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

С. В. Ярмолик, В. Н. Ярмолик

КРИПТОГРАФИЯ И ОХРАНА КОММЕРЧЕСКОЙ ИНФОРМАЦИИ

Методическое пособие по выполнению лабораторных работ
для студентов специальностей 1-40 01 02 – 02 и 1-26 02 03
дневной и заочной форм обучения

Минск БГУИР 2010

УДК
ББК

Р е ц е н з е н т

Ярмолик, С. В.

Я75 Криптография и охрана коммерческой информации: метод. пособие по выполнению лаб. работ для студ. спец. 1-40 01 02 – 02 и 1-26 02 03 дневн. и заоч. форм обуч. / С. В. Ярмолик, В. Н. Ярмолик. – Минск: БГУИР, 2010. – 32 с.

ISBN

В пособие рассматриваются практические вопросы криптографического преобразования информации в компьютерных системах. Рассмотрены актуальные вопросы предметной области – алгоритмы шифрования с открытым ключом; алгоритмы электронной цифровой подписи и хеширования; идентификация личности с помощью алгоритмов доказательства с нулевым разглашением. По каждой теме приводятся теоретические сведения, практические способы реализации алгоритмов, примеры и набор заданий.

УДК
ББК

ISBN

СОДЕРЖАНИЕ

1. ПРОСТЕЙШИЕ АЛГОРИТМЫ ШИФРОВАНИЯ

1.1. Шифрование методами перестановок

Суть методов перестановки состоит в том, что исходное сообщение M делится на блоки данных, состоящих из d символов $M = m_1,...,m_d\ ,m_{d+1},\ldots,m_{2d},...$, после чего полученные блоки исходного текста преобразуются в соответствии с некоторым вектором, функцией или фигурой. В результате получим $E_k(M) = m_{f(1)},\ldots,m_{f(d)},m_{f(d+1)},\ldots,m_{f(2d)},...$

Простейшим представителем этого метода является **метод "железнодорожной изгороди"**, при использовании которого исходное сообщение преобразуется в соответствии с фигурой, напоминающей по форме железнодорожную изгородь, откуда и пошло его название. В этом случае символы исходного текста записываются в виде, напоминающем по форме забор, а символы зашифрованного текста считываются из полученной записи построчно. Пример шифрование этим методом приведен на рис. 1.1.

M = это лабораторная работа по киоки

↓

Э О Н О И
Т Б Р Р А Б Т К О
О А А О Я А А О К
Л Т Р П И

↓

C = ЭОНОИ ТБРРА БТКОО ААОЯА АОКЛТ РПИ

Рис. 1.1. Шифрование методом "Железнодорожная изгородь"

M = "Это лабораторная работа по КиОКИ" является исходным текстом, а *C* = "ЭОНОИ ТБРРА БТКОО ААОЯА АОКЛТ РПИ" – соответствующим ему шифротекстом. "Высота изгороди" *K* является ключом, который в приведенном примере равен 4. Для того чтобы расшифровать полученный текст, необходимо выполнить действия обратные выполненным при шифровании и использовать тот же ключ.

Одной из наиболее известных модификаций метода перестановки является "**столбцовый метод"**, при котором исходное сообщение записывается в таблицу построчно, а затем считывается оттуда столбцами согласно некоторому вектору, задающему порядок считывания. Этот вектор может быть задан с помощью ключевого слова или фразы, буквам которого назначаются номера в соответствии с алфавитом, а если буква встречается несколько раз, то нумерация определяется порядком следования повторяющихся букв в ключевом слове. Например, пусть у нас есть исходный текст *M* = "Это

лабораторная работа по КиОКИ" и ключ K = "КРИПТОГРАФИЯ". Запишем текст в таблицу и считаем его по столбцам, порядок считывания которых задан ключом K = "КРИПТОГРАФИЯ":

К	Р	И	П	Т	О	Г	Р	А	Ф	И	Я
5	8	3	7	10	6	2	9	1	11	4	12
Э	Т	О	Л	А	Б	О	Р	А	Т	О	Р
Н	А	Я	Р	А	Б	О	Т	А	П	О	К
И	О	К	И								

При этом получим следующий шифротекст С = "ААООО ЯКООЭ НИББЛ РИТАО РТААТ ПРК".

Во время первой мировой войны немецкие военные использовали двойной столбцовый шифр, при котором одно сообщение шифровалось дважды с один и тем же или с двумя разными ключами. Так для приведенного выше примера, зашифруем полученный ранее шифротекст, используя другой ключ K_2 = = "ШИФРОВАНИЕ". Тогда получим C = "КИРЯР ПЭОАИ ТОАОТ КОЛТО БАОБА АНР".

Ш	И	Ф	Р	О	В	А	Н	И	Е
10	4	9	8	7	2	1	6	5	3
А	А	О	О	О	Я	К	О	О	Э
Н	И	Б	Б	Л	Р	И	Т	А	О
Р	Т	А	А	Т	П	Р	К		

Такая система является ненадежной, и она была взломана французами. Им требовалось несколько дней после введения нового ключа для его вычисления. Это стало известно немцам, и они изменили используемый шифр на другой 18 ноября 1914.

Другой метод, использовавшийся Германией во время первой мировой войны, – это **метод поворачивающейся решетки**. Суть его состоит в том, что исходный текст записывался на бумаге через отверстия решетки, которая по мере заполнения поворачивалась на 90° градусов.

1	2	3	1
3	4	4	2
2	4	4	3
1	3	2	1

1	2	3	4	1
4	5	6	5	2
3	6	7	6	3
2	5	6	5	4
1	4	3	2	1

1	2	3	4	5	1
5	6	7	8	6	2
4	8	9	9	7	3
3	7	9	9	8	4
2	6	8	7	6	5
1	5	4	3	2	1

Рис. 1.2. Примеры построения матриц
для метода поворачивающейся решетки

Сделать такую решетку достаточно легко. Строится матрица, в которой ячейки нумеруются согласно правилу, что ячейкам, при повороте на 90° занимающим одинаковое положение, присваивается один и тот же номер. На рис. 1.2. приведены примеры таких матриц размером 4x4, 5x5, 6x6. Затем вырезается по одному квадрату для каждого номера.

Например, возьмем матрицу 4x4 и вырежем следующие ячейки:

X			
			X
		X	
	X		

Теперь возьмем наше исходное сообщение, например, *M* = "ШИФРОТЕКСТ" и, используя решетку, зашифруем его. В пустые клеточки можно вставить ничего не значащие буквы (см. рис. 1.3). На выходе получим шифротекст *C* = "ШВСОТ ТГИАЕ ФДЕРК Б".

Ш			
			И
		Ф	
	Р		

Ш			**О**
Т			И
	Е	Ф	
	Р	**К**	

Ш		**С**	О
Т	**Т**		И
А	Е	Ф	
	Р	К	**Б**

Ш	**В**	С	О
Т	Т	**Г**	И
А	Е	Ф	**Д**
Е	Р	К	Б

Рис. 1.3. Шифрование методом "Поворачивающейся решетки"

1.2. Шифрование методами подстановок

Криптографические системы, основанные на методе подстановки, можно разделить на четыре основных класса:

1. Одноалфавитный шифр подстановки (шифр простой замены) – шифр, при котором каждый символ открытого текста заменяется некоторым фиксированным при данном ключе символом того же алфавита. Примером может служить шифр Цезаря.

2. Однозвучный шифр подстановки (омофонный) похож на одноалфавитный за исключением того, что символ открытого текста может быть заменен одним из нескольких возможных символов. К данным шифрам относят шифр Билла.

3. Полиграммный шифр подстановки заменяет не один символ, а целую группу символов. Примерами таких шифров являются шифр Плейфейра, шифр Хилла.

4. Многоалфавитный шифр подстановки состоит из нескольких шифров простой замены. Например, шифр Виженера, одноразовый блокнот.

Одноалфавитный шифр подстановки характеризуется тем, что каждому символу алфавита исходного текста в соответствии ставиться другой символ из шифроалфавита. Криптографическим ключом такой системы является таблица соответствия исходного алфавита алфавиту подстановки. Например, для английского алфавита существует $26! \approx 4*10^{26}$ различных криптографических ключей.

Примером такого шифра является **шифр Цезаря**. Так, шифр Цезаря заменяет каждый символ алфавита исходного текста на сдвинутый относительно него символ того же алфавита на k позиций вправо, при этом k является ключом шифра. Т.е. в алгоритме Цезаря i-й символ алфавита заменяется $(i+k)$-м по модулю n символом, где n – количество букв алфавита. Для английского языка $n = 26$, для русского – 33, для ASCII-кодов – 256. Юлий Цезарь использовал подобную систему для $k = 3$, откуда и пошло название данного шифра. Аналитически криптосистема Цезаря описывается выражением 1.1.

$$E_k(i) = (i+k) \bmod n. \qquad (1.1)$$

Например, в соответствии с приведенным выражением буква 'a' исходного английского алфавита, имеющая номер $i = 0$, заменяется буквой 'D', имеющей номер $(i+k) \bmod n = (0+3) \bmod 26 = 3$, а буква 'z' ($i = 25$) заменяется буквой 'C', имеющей номер $(i+k) \bmod 26 = (25+3) \bmod 26 = 2$.

Алгоритм дешифрования имеет вид (1.2):

$$D_k(i) = (i+n-k) \bmod n. \qquad (1.2)$$

Существуют более сложные методы подстановки. Шифраторы, основанные на умножении номера каждого символа исходного текста на значение ключа k (**метод децимации**), описываются следующим отношением 1.3:

$$E_k(i) = (i*k) \bmod n, \qquad (1.3)$$

где n и k должны быть взаимно простыми числами, т.е. у них не должно быть общих делителей кроме 1.

Комбинацией двух приведенных выше методов шифрования является **афинное преобразование**, при котором уже используются два ключа:

$$E_k(i) = (i*k_1+k_2) \bmod n. \qquad (1.4)$$

Рассмотренные выше шифры простой замены легко взламываются с помощью криптографических атак, основанных на анализе частот появления символов в шифротексте. Так в естественном языке частота встречи букв разная и некоторые из них встречаются чаще, чем другие, и это является

ключом для криптоаналитика. Проанализировав частоту встречи символов в шифротексте, можно сделать вывод о соответствии им символов исходного алфавита.

Омофонные шифраторы обеспечивают простейшую защиту от таких атак. Омофононые шифры являются одноалфавитными, хотя при этом каждому символу исходного текста ставится в соответствие несколько подстановочных элементов – омофонов, количество которых прямо пропорционально частоте использования данного символа в исходных текстах. При шифровании каждый символ исходного текста заменяется омофоном, случайным образом выбранным из множества омофонов, соответствующих данному конкретному символу.

Предположим, что в качестве омофонов для английских букв использованы двухзначные целые числа, лежащие в диапазоне между 00 и 99. Количество омофонов для конкретного символа алфавита исходных текстов выбирается пропорционально относительной частоте букв английского языка в исходных текстах, кроме того, один и тот же омофон используется как подстановочный элемент только для одной буквы английского языка. Возможное сопоставление целых двухзначных чисел омофонов выбранным буквам английского языка приведено ниже:

A 23, 25, 97, 95, 89, 33, 12, 11, 34
C 87, 41
G 44, 77, 35, 51
H 59, 90, 00, 26, 36
O 66, 02, 15, 22, 09, 83, 54
P 04, 58
R 38, 07, 94, 30, 56, 67
T 55, 71, 72, 80, 01, 12, 29, 50, 68
Y 88

Тогда для исходного текста *M* = "CRYPTOGRAPHY" возможный вариант шифротекста имеет вид *C* = "87 07 88 58 72 54 51 30 97 04 00 88".

Улучшение качества омофонного шифратора достигается увеличением количества омофонов, используемых при шифровании, однако здесь необходимо отметить, что это приводит к усложнению процедур шифрования и дешифрования.

Биграммный шифр Плейфейра, который применялся в Великобритании во время первой мировой войны, – наиболее известный полиграммный шифром замены. Суть полиграммных алгоритмов состоит в том, что одновременно шифруется не один, а сразу несколько символов, что также позволяет видоизменить частотные зависимости характерные для исходных текстов.

Шифр Плейфейра является биграммным и при шифровании рассматривается два символа. Основой данного шифра является шифрующая

таблица со случайно расположенными буквами алфавита исходных текстов (см. рис. 1.4). Такая таблица представляет собой сеансовый ключ. Для удобства запоминания шифрующей таблицы можно использовать ключевое слово или фразу, которые записывают в начальные строки таблицы.

Для случая английского языка шифрующая таблица задается матрицей 5×5, состоящей из 25 позиций с символами алфавита английского языка (позиция для символа 'J' соответствует позиции для символа 'I').

C	R	Y	P	T
O	G	A	H	B
D	E	F	I/J	K
L	M	N	Q	S
U	V	W	X	Z

Рис. 1.4. Таблица шифра Плейфейра

Процедура шифрования включает следующие этапы. Сначала исходный текст M разбивается на пары символов $M = m_1m_2, m_3m_4, \ldots$ (биграммы), после чего полученные биграммы $m_1m_2, m_3m_4,\ldots$ открытого текста M преобразуется с помощью шифрующей таблицы в последовательность биграмм $c_1c_2, c_3c_4, \ldots$ шифротекста C по следующим правилам:

1. Если буквы биграммы исходного текста m_i и m_{i+1} находятся в одной и той же строке шифрующей матрицы, то c_i и c_{i+1} представляют собой два символа справа от m_i и m_{i+1}, соответственно (см. рис. 1.5). Здесь первый столбец матрицы, является столбцам справа по отношении к последнему столбцу. Например, если m_im_{i+1} = YT то c_ic_{i+1} = PC.

2. Если m_i и m_{i+1} находятся в одном и том же столбце, то c_i и c_{i+1} принимают значения символов ниже m_i и m_{i+1}, соответственно. Первая строка считается строкой ниже последней. Например, если m_im_{i+1} = VG то c_ic_{i+1} = RE.

3. Если m_i и m_{i+1} находятся в различных строках и столбцах, то c_i и c_{i+1} соответствуют двум другим углам прямоугольника, имеющего m_i и m_{i+1}, в качестве двух исходных углов, при этом c_i находится в той же строке что и m_i, а c_{i+1} находится в той же строке что и m_{i+1}. Например, если m_im_{i+1} = ZF, то c_ic_{i+1} = WK как видно из диаграммы на рис. 1.5.

C	**R**	**Y**	**P**	**T**
O	**G**	A	H	B
D	**E**	**F**	I/J	**K**
L	M	N	Q	S
U	**V**	**W**	X	**Z**

Рис. 1.5. Шифрование алгоритмом Плейфейра

4. Если $m_i = m_{i+1}$ тогда пустой символ (например, 'X') вставляется в исходный текст между m_i и m_{i+1}, чтобы устранить равенство $m_i = m_{i+1}$. Например, если $m_i m_{i+1}$ = SS, тогда $m_i m_{i+1} m_{i+2}$ = SXS и соответственно $c_i c_{i+1}$ = QZ.

5. Если исходный текст имеет нечетное число знаков, пустой символ добавляется в конец текста для получения четного числа символов исходного текста.

Например, для исходный текст M = "CIPHERTEXT", в результате шифрования согласно алгоритма Плейфейра, используя матрицу, приведенную на рис. 1.4, в качестве ключа, получим шифротекст C = "PDHIMGRKZP".

Следует отметить, что шифрование биграммами существенно повышает стойкость шифров к взлому, однако частотные свойства распределения биграмм по-прежнему является ключом для злоумышленника.

С целью упрощения процедур шифрования и дешифрования была предложена модификация данного метода путем использования четырех шифрующих матриц, как представлено на рис. 1.6. Причем две матрицы (второй и четвертый квадрант) используются для задания символов исходного текста, а матрицы первого и третьего квадранта для получения символов шифротекста. Подобная модификация предполагает меньшее число правил по сравнению с шифратором Плейфейра.

M	W	X	Y	N		W	O	M	L	H
V	A	P	K	L		U	A	N	K	I
U	R	B	O	Z		S	B	C	Z	Y
E	F	Q	C	I		Q	P	D	E	Z
T	**S**	G	H	D		R	T	V	**F**	G
A	K	O	N	I		P	R	M	O	N
Z	B	L	P	H		I	D	S	E	F
U	T	C	M	G		H	G	C	T	Y
X	**S**	W	D	F		K	W	L	**B**	Z
Y	R	V	Q	E		V	Q	X	U	A

Рис. 1.6. Шифрование модифицированным алгоритмом Плейфейра

При шифровании биграммы $m_i m_{i+1}$ = SB получим $c_i c_{i+1}$ = FS.

Шифр Вижинера, является шифром многоалфавитной подстановки и использует развитие идеи Цезаря. Сутью данного алгоритма шифрования является чередование использования таблиц подстановки в зависимости от последовательности символов используемого ключа. Этот шифр можно описать таблицей шифрования, называемой таблицей Виженера. На рис. 1.6 приведен пример таблицы Виженера для английского языка.

В таблице Виженера каждая строка представляет собой циклически сдвинутую на один символ предыдущую строку таблицы таким образом, что каждая строка по своей сути является таблицей подстановки шифратора Цезаря для конкретного значения ключа.

Верхняя строка таблицы Виженера используется для задания символов исходных текстов, а левый столбец для задания символов криптографического ключа. При шифровании исходного сообщения его записывают в строку, а под ним ключевое слово либо фразу. Если ключ оказался короче исходного текста, то его циклически повторяют необходимое число раз. На каждом шаге шифрования в верхней строке таблицы Виженера находят очередную букву исходного текста, а в левом столбце – очередное значение символа ключа. В результате очередная буква шифротекста находится на пересечении столбца определенного символом исходного текста и строки, соответствующей строке символа ключа.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	a	B	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
B	b	C	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a
C	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b
D	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c
E	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d
F	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
G	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f
H	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g
I	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h
J	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i
K	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j
L	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k
M	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l
N	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m
O	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n
P	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
Q	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
R	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
S	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r
T	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s
U	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t
V	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u
W	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
X	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
Y	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x
Z	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y

Рис. 1.7. Таблица Виженера для английского языка

При шифровании слова *M* = "Cryptography" по методу Виженера для ключа "MODE" предварительно исходный текст и ключевое слова запишем в виде двух строк.

M =	c	r	y	p	t	o	g	r	a	p	h	y
K =	M	O	D	E	M	O	D	E	M	O	D	E
C =	O	F	B	T	F	C	J	V	M	D	K	C

Тогда первая буква исходного текста 'c' определяет третий столбец таблицы Виженера, а буква 'M' ключа тринадцатую строку таблицы, на пересечении которых находится символ шифротекста 'O'. Аналогично для остальных букв. Получим шифротекст C = "OFBTF CJVMD KC".

Различают три возможных варианта использования криптографического ключа:

1. Прямое использование, которое было рассмотрено выше.

2. Прогрессивный ключ, при повторном применении которого символы ключа циклически сдвигаются на одну позицию в упорядоченном алфавите символов. Например, ключ "MODE" при повторном его использовании по прогрессивной схеме будет иметь вид "NPEF", а при третьем – "OQFG", и так далее.

Для шифрования сообщения "Cryptography", используем ключ "MODE" и прогрессивную схему его применения и получим.

M =	c	r	y	p	t	o	g	r	a	p	h	y
K =	M	O	D	E	N	P	E	F	O	Q	F	G
C =	O	F	B	T	G	D	K	W	O	F	M	E

3. Самогенерирующийся ключ, при котором в качестве его последующих символов используется исходный текст.

Для шифрования сообщения "Cryptography", используем самогенерирующийся ключ "MODE". В результате имеем.

M =	c	r	y	p	t	o	g	r	a	p	h	y
K =	M	O	D	E	C	R	Y	P	T	O	G	R
C =	O	F	B	T	V	F	E	G	T	D	N	P

Задание для выполнения лабораторной работы №1

1. Изучить теоретический материал по лабораторной работе.

2. Реализовать шифраторы и дешифраторы на основе трех рассмотренных перестановочных методов.

3. Реализовать шифратор и дешифратор подстановочного метода согласно варианту, выданному преподавателем (см. табл. 1.1).

Таблица 1.1

Вариант	№1	№2	№3
Шифр	Шифр Цезаря	Шифр Плейфейра	Шифр Вижинера

2. КРИПТОГРАФИЧЕСКИЕ СИСТЕМЫ С ОТКРЫТЫМ КЛЮЧОМ

Первые криптографические системы с открытым ключом или ассиметричные криптосистемы появились в конце 1970-х годов. От классических симметричных алгоритмов они отличаются тем, что для шифрования данных используется один ключ, обычно его называют открытый или публичный ключ, а для дешифрования – другой, секретный или закрытый, ключ.

Диффи и Хеллман, которые впервые предложили и описали криптосистему с открытым ключом, выявляют следующие требования к криптосистемам:

1. Вычислительно легко создавать пару открытый ключ (K_o), закрытый ключ (K_c).

2. Вычислительно легко, имея открытый ключ и незашифрованное сообщение M, создать соответствующее зашифрованное сообщение: $C = E_{ko}[M]$.

3. Вычислительно легко дешифровать сообщение, используя закрытый ключ: $M = D_{Kc}[C] = D_{Kc}[E_{Ko}[M]]$.

4. Вычислительно невозможно, зная открытый ключ K_o, определить закрытый ключ K_c.

5. Вычислительно невозможно, зная открытый ключ K_o и зашифрованное сообщение C, восстановить исходное сообщение M.

Можно добавить шестое требование, хотя оно не выполняется для всех алгоритмов с открытым ключом:

6. Шифрующие и дешифрующие функции могут применяться в любом порядке, т.е. $M = E_{Ko}[D_{Kc}[M]] = D_{Kc}[E_{Ko}[M]]$.

Таким образом, данные, зашифрованные открытым ключом, можно расшифровать только секретным ключом. Следовательно, открытый ключ может распространяться через обычные коммуникационные сети и другие открытые каналы, что устраняет главный недостаток стандартных криптографических алгоритмов: необходимость использовать специальные каналы связи для распределения ключей.

2.1. Криптосистема RSA

В настоящее время лучшим и наиболее популярным криптографическим алгоритмом с открытым ключом считается RSA, название которого получено по именам его создателей: Rivest, Shamir и Adelman.

Наиболее важной частью алгоритма RSA, как и других алгоритмов с открытым ключом, является процесс создания пары открытый/секретный ключи. В RSA он состоит из следующих шагов:

1. Случайным образом выбираются два секретных простых числа p и q таких, что $p \approx q$.

2. Вычисляется их произведение $r = p*q$.

3. Вычисляется функция Эйлера для r. Функция Эйлера для произвольного x представляет собой число взаимно простых с x чисел меньших, чем x. Для простого x $\varphi(x) = x–1$, а для числа представляющего собой произведение двух простых чисел $x = p*q$: $\varphi(x) = (p–1)*(q–1)$.

4. Выбирается целое значения открытой экспоненты e такой, что $1<e<\varphi(r)$ и $(e, \varphi(r)) = 1$.

5. Вычисляется значение секретной экспоненты d, которая должна удовлетворять условию $(e*d) \bmod \varphi(r) = 1$ (т.е., d является мультипликативной инверсной по модулю $\varphi(r)$ для e).

Таким образом, ключом шифрования K_o является пара значений (e, r), а ключом дешифрования K_c – (d, r). Значение параметра r, так же как и значение e являются общедоступной информацией, в то время как значения параметров p, q и d хранятся в секрете.

Перед шифрованием исходное сообщения M необходимо разбить на блоки $M = m_1,m_2,m_3\ldots$, где m_i представляет собой число в диапазоне от 0 до $r–1$. Сам процесс шифрования открытым ключом $K_o = (e, r)$ последовательности чисел m_i происходит согласно формуле:

$$c_i = (m_i^e) \bmod r, \qquad (2.1)$$

где последовательность чисел c_i представляет собой шифротекст.

Чтобы расшифровать эти данные секретным ключом $K_c = (d, r)$, необходимо выполнить следующие вычисления:

$$m_i = (c_i^{\,d}) \bmod r. \qquad (2.2)$$

В результате будет получено множество чисел m_i, которые представляют собой исходный текст.

Приведем простой пример использования метода RSA для шифрования сообщения "CAT". Сначала получим ключи:

1. Выберем $p = 3$, $q = 11$.

2. Вычислим $r = 3*11 = 33$.

3. Вычислим $\varphi(r) = (p–1)*(q–1) = 2*10 = 20$.

4. Выберем открытую экспоненту e, которая является взаимно простой с $\varphi(r) = 20$, например, $e = 7$.

5. На основе e и $\varphi(r)$ вычислим закрытую экспоненту d. Для этого можно использовать расширение алгоритма Евклида:

```
EUCLIDEX(a; b)
d0:=a; d1:=b;
x0:=1; x1:=0;
y0:=0; y1:=1;
while d1>1 do
begin
q:=d0 div d1;
d2:=d0 mod d1;
x2:=x0-q*x1;
y2:=y0-q*y1;
d0:=d1; d1:=d2;
```

```
x0:=x1; x1:=x2;
y0:=y1; y1:=y2;
end
return (x1; y1; d1)
```

Расширенный алгоритм Евклида позволяет вычислить числа x_1 и y_1, для которых выполняется равенство $x_1*a+y_1*b = d_1$, где d_1 = НОД(a, b). Если a и b взаимно простые, и $a>b$, то y_1 является мультипликативным инверсным по модулю a для b, т. е. $y_1*b \bmod a = 1$. Используя данный алгоритм, можно вычислим d, положив a, равным $\varphi(r)$, и b, равным $e = 7$. Если значение y_1 получилось отрицательным, то для получения корректного значения d необходимо добавить к y_1 значение a (или $\varphi(r)$).

В нашем случае имеем $d = y_1 = 3$.

6. Представим шифруемое сообщение как последовательность целых чисел в диапазоне 2...27 (для английского языка). Пусть букве 'A' соответствует число 2, 'B' – 3, 'C' – 4 и так далее. Тогда сообщение M = "CAT" можно представить в виде последовательности чисел $m_1,m_2,m_3 = \{4, 2, 21\}$. Зашифруем сообщение, используя открытый ключ $K_o = (7, 33)$:

$c_1 = (m_1^e) \bmod r = (4^7) \bmod 33 = 16384 \bmod 33 = 16,$
$c_2 = (m_2^e) \bmod r = (2^7) \bmod 33 = 128 \bmod 33 = 29,$
$c_3 = (m_3^e) \bmod r = (21^7) \bmod 33 = 1801088541 \bmod 33 = 21.$

7. Для расшифровки полученного сообщения $C = \{16, 29, 21\}$ с помощью секретного ключа $K_c = (3, 33)$, необходимо выполнить действия:

$m_1 = (c_1^e) \bmod r = (16^3) \bmod 33 = 4096 \bmod 33 = 4,$
$m_2 = (c_2^e) \bmod r = (29^3) \bmod 33 = 24389 \bmod 33 = 2,$
$m_3 = (c_3^e) \bmod r = (21^3) \bmod 33 = 9261 \bmod 33 = 21.$

Таким образом, в результате расшифровки сообщения получено исходное сообщение {4, 2, 21} = "CAT".

Для возведения в степень $x = a^z \bmod n$ можно использовать алгоритм быстрого возведения в степень по модулю:

```
function fast_exp(a,z,n)
begin
a1:=a
z1:=z
x:=1
while z1<>0 do
begin
while (z1 mod 2)=0 do
begin
z1:=z1 div 2
a1:=(a1*a1) mod n
end
z1:=z1-1
x:=(x*a1) mod n;
end
fast_exp:=x
end
```

Криптостойкость алгоритма RSA основывается на двух математических трудно решаемых задачах, для которых не существует эффективного способа их решения. Первая из них заключается в том, что невозможно вычислить исходный текст из шифротекса, так как для этого надо извлечь корень степени e по модулю числа r (найти дискретный логарифм). Данную задачу в настоящее время невозможно решить за полиномиальное время. С другой стороны практически невозможно найти секретный ключ, зная открытый, поскольку для этого необходимо решить сравнение $e*d \equiv 1 \pmod{\varphi(r)}$. Для его решения нужно знать делители целого числа r, т.е. разложить число r на сомножители. Задача разложения на множители, задача факторизации числа, в настоящее время также не имеет эффективного (полиномиального) решения, но пока не было доказано, что алгоритма решения данной задачи не существует.

На практике же применяются 1024–2048-битные числа r, однако предсказывается, что в скором времени 1024-битные ключи будут взломаны, а 2048-битные — будут криптостойкими до 2030. Поэтому лучше использовать 3072-битные ключи, если необходимо обеспечить секретность вплоть до 2030г. Так, если взять ключ r длиной 300 бит, то разложить на множитель его можно за несколько часов на обычном персональном компьютере.

2.2. Криптосистема Эль-Гамаля

Данная криптосистема, предложенная в 1984 году, лежит в основе стандартов электронной цифровой подписи в США и России.

Как и для алгоритма RSA, для криптосистемы Эль-Гамаля необходимо перед шифрованием вычислить пару открытый/закрытый ключ. Это происходит по следующему алгоритму:

1. Генерируется случайное простое число p.

2. Выбирается произвольное целое число g, являющееся первообразным корнем по модулю p. Первообразным корнем по модулю некоторого числа p является целое число g такое, что $g^{\varphi(p)} = 1 \bmod p$ и $g^l \neq 1 \bmod p$ при $1 \le l \le \varphi(p)-1$.

3. Выбирается случайное целое положительное число $x < p-1$, не равное 1.

4. Вычисляется $y = g^x \bmod p$.

Открытым ключом K_o является тройка (p, g, y), закрытым ключом K_c – число x.

Сообщение $m \in [0, p-1]$ шифруется так:

1. Выбирается случайное секретное число $k \in (1, p-1)$, взаимно простое с $p-1$.

2. Вычисляется два значения a и b:

$$\begin{aligned} a &= g^k \bmod p, \\ b &= y^k m \bmod p, \end{aligned} \qquad (2.3)$$

где m является исходным сообщением, а пара чисел (a, b) – шифротекстом.

Как видно, полученный шифротекст в два раза длиннее исходного сообщения. Также следует отметить, что для разных сообщений m_1 и m_2 следует использовать и разные значения случайного k. Например, если для двух текстов m_1 и m_2 было использовано одно и то же значение k, то получим, что $\frac{b_1}{b_2} = \frac{m_1}{m_2}$, и значение m_2 может быть легко вычислено злоумышленником, при известном m_1.

Зная закрытый ключ x, исходное сообщение можно вычислить из шифротекста (a, b) по формуле:

$$m = ba^{-x} \bmod p \qquad (2.4)$$

Приведем пример шифрования сообщения $m = 4$ с помощью алгоритма Эль-Гамаля. Для начала сгенерируем ключи:

1. Выберем простое число $p = 13$.

2. Подберем для него такое g, которое являлось бы первообразным корнем по модулю p. Для этого используем алгоритм нахождения случайного первообразного корня по модулю p:

```
in: p, q1, q2, ..., qk // q1,q2,...,qk - все простые делители числа p-1
out: g
Root (p, q1, q2, ..., qk)
begin
g:=random(p-1); // генерируем случайное число из диапазона [2, p-1]
for (i:=1; i<k+1; i++)
       if ( exp(g,(p-1)/qi)== 1 (mod p)) return Root (p, q1, q2, ..., qk);
return g;
end
```

Исходя из алгоритма, получаем $g = 7$.

3. Закрытым ключом x возьмем число 5.

4. Вычисляем $y = g^x \bmod p = 7^5 \bmod 13 = 11$.

Открытым ключом является K_o = (13, 7, 11), закрытым ключом K_c – число 5.

Далее зашифруем сообщение m = 4, для чего сгенерируем случайное число $k = 7$, взаимно простое с $p-1$, и вычислим по формуле 2.3 значения a и b:

$a = g^k \bmod p = 7^7 \bmod 13 = 6$,
$b = y^k m \bmod p = 11^7 * 4 \bmod 13 = 8$.

На выходе получаем шифротекст (6, 8).

Вычислим исходное сообщение из шифротекста (a, b) по формуле 2.4:

$$m = \frac{b}{a^x} \bmod p = \frac{6}{8^5} \bmod 13 = \frac{6}{32768} \bmod 13 = 6 * 32768^{\varphi(p)-1} \bmod 13 = 6 * 32768^{11} \bmod 13 = 4 .$$

Криптостойкость криптосистемы Эль-Гамаля основана на том, что невозможно вычислить закрытый ключ, зная открытый, за полиномиальное время, так как для этого необходимо вычислить дискретный логарифм, т.е. по известным значениям p, g и y вычислить x, который бы удовлетворял сравнению: $y \equiv g^x \bmod p$.

2.3. Криптосистема Рабина

Криптосистема, разработанная Рабином, подобно RSA основывается на трудной проблеме факторизации больших целых чисел или на проблеме извлечения квадратного корня по модулю составного числа $n = p*q$. Эти две задачи являются эквивалентными, т. е. зная простые делители числа n, можно извлечь квадратные корни по модулю n, а умея извлекать квадратные корни по модулю n, – разложить n на простые множители.

Генерация ключей в криптосистеме Рабина происходит следующим образом:

1. Выбираются два разных случайных простых числа p и q таких, что $p \approx q$. При этом они должны удовлетворять условию: $p \equiv q \equiv 3 \bmod 4$. Выполнение данного условия необходимо для упрощения вычисления квадратного корня по модулю p и q при дешифрации сообщения.

2. Вычисляется $n = p*q$.

3. Выбирается случайное число $b<n$.

Открытым ключом будут n и b, в то время как простые числа p и q – закрытым.

Для получения шифротекста c_i необходимо выполнить следующие действия над исходным сообщением M, которое также разбивается на блоки $m_1 m_2 m_3 ..$, $(0 \le m_i \le n-1)$:

$$c_i = m_i(m_i+b) \bmod n. \tag{2.5}$$

Процесс шифрования в алгоритме Рабина происходит намного быстрее, чем в других криптосистемах с открытым ключом.

Для дешифрации же нужно решить квадратное уравнение вида $m_i^2 + b*m_i - c_i = 0 \pmod n$. Как известно, общее решение такого уравнения будет иметь вид:

$$m = \frac{-b + \sqrt{D}}{2} (\bmod n), \tag{2.6}$$

где $D = b^2+4c \pmod n$.

Вычислить квадратный корень из D по модулю числа $n=p*q$ можно с помощью китайской теоремы об остатках, для чего необходимо знать закрытые ключи p и q. Вычислив $\sqrt{D}$, получим четыре результата d_1, d_2, d_3, d_4, и,

подставив в 2.6, получим m_1, m_2, m_3, m_4. Главным недостатком криптосистемы Рабина является то, что неизвестно который из четырех результатов, m_1, m_2, m_3, m_4 равен исходному m_i. Если сообщение написано по-русски, выбрать правильное m_i, нетрудно. С другой стороны, если сообщение является потоком случайных битов, способа определить, какое m_i, – правильное, нет. Одним из способов решить эту проблему служит добавление к сообщению перед шифрованием известного заголовка.

Приведем пример работы данного алгоритма. Пусть нам надо зашифровать сообщение 'P'. 'P' является 17-ой буквой алфавита, поэтому $m = 17$. Сначала сгенерируем ключи:

1. Выберем два простых числа $p = 11$ и $q = 19$. При этом $11 \bmod 4 = 3$ и $19 \bmod 4 = 3$. Т.е. условие $p \equiv q \equiv 3 \bmod 4$ выполняется.

2. Вычислим наш открытый ключ: $n = p*q = 11*19 = 209$.

Далее зашифруем сообщение $m = 17$. Для этого вычислим по формуле 2.5 $c = m*(m+b) \bmod n = 17*(17+173) \bmod 209 = 95$. Таким образом, получили шифротекст $c = 95$.

Для расшифровки сначала вычислим D по формуле 2.6:

$$D = b^2 + 4c(\bmod n) = 173^2 + 4*95(\bmod 209) = 4 .$$

Далее для вычисления квадратного корня из D=4 по модулю составного числа $n = p*q$ выполним следующие действия:

1. Вычислим m_p и m_q по формуле:

$$\begin{aligned} m_p &= \sqrt{D} \bmod p = D^{\frac{p+1}{4}} \bmod p = 4^{\frac{11+1}{4}} \bmod 11 = 4^3 \bmod 11 = 9, \\ m_q &= \sqrt{D} \bmod q = D^{\frac{q+1}{4}} \bmod q = 4^{\frac{19+1}{4}} \bmod 19 = 4^5 \bmod 19 = 17. \end{aligned} , \qquad (2.7)$$

2. Вычислим значения y_p и y_q таких, что $y_p*11+y_q*19 = 1$ по расширенному алгоритму Евклида. Для этого принимаем $a = 11$, $b = 19$. Тогда на выходе получим, что $y_p = x_1 = 7$, а $y_q = y_1 = -4$.

3. Далее вычисляем квадратные корни из D по модулю n по формулам:

$$d_1 = (y_p * p * m_q + y_q * q * m_p) \, mod \, n$$

$$d_2 = n - d_1$$

$$d_3 = (y_p * p * m_q - y_q * q * m_p) \, mod \, n$$

$$d_4 = n - d_3$$

4. Теперь необходимо вычислить значения согласно формуле 2.6 с одним условием:

Если (di - b) mod 2 = 0, тогда:

$$m_i = \frac{-b + d_i}{2} mod \, n$$

А иначе, если (di - b) mod 2 != 0, тогда:

$$m_i = \frac{-b + n + d_i}{2} mod\ n$$

На выходе получаем корни:
m1 = 17 (верный), m2 = 19, m3 = 74 и m4 = 171.

Как видно из предыдущих вычислений, после работы алгоритма на одном шифруемом значении у нас получается 4 корня и только один из них верный. Есть ли способ расшифровать информацию однозначно? Есть!

Использование на практике:

Все файлы есть набор байтов от 0 до 255. Значит, что шифровать придется числа от 0 до 255. Возьмем число 17 из примера на прошлой странице, а так же допустим, что p=523, q=3 а b=1 и пропустим все это через алгоритм выше. Получим четыре значения m: 1551, **17**, 540, 1028. Видно, что среди этих корней лишь один меньше, чем 256 (m < 256). Совпадение? Попробуем другие числа, побольше: p = 5003,q = 5227 и b = 1234. Получаем: 26149430, **17**, 495314, 25654133. И снова лишь один корень! Очевидно, что при использовании достаточно больших p и q среди выходных корней будет лишь один, меньше 256. Или нет? Попробуем p=523, q= 3 а b=2, получим четыре m: **17**, 1550, 1550, **17**. У нас получилось два подходящих корня, но нам нужен только один! Это значит, что если во время проверки неравенства mi < 256 мы находим корень, то нужно прекращать проверку.
Для однозначной расшифровки любых файлов Вы можете использовать любое 0 < b < 10533, и p,q больше 3 и 3511 соответственно (всего лишь рекомендация). Эти значения должны давать однозначную расшифровку. Значения меньше будут давать на некоторых байтах неверные значения. Так же, возможно, кое-где придется использовать длинную арифметику.

Примечание:

- p*q=n > 256
- b - натуральное число
- При создании шифро-файла зашифрованные значения могут не "влезать" в байт: снова возьмем число 17, p =31,q=43 а b=176. Зашифрованное значение получилось 615, что явно больше 255,а значит, записать в один байт файла это значение мы не можем. Выход из этого положения такой: записывать в шифро-файл не байты, а полученные значения через пробел, например: 615 23 176 2945 ... То есть при расшифровке считывать файл нужно будет не побайтово, а по содержимому.

3. ЭЛЕКТРОННАЯ ЦИФРОВАЯ ПОДПИСЬ

3.1. Функция хеширования

Функцией хеширования h называется преобразование данных, переводящее строку M произвольной длины в значение $m=h(M)$ (хеш-образ или дайджест сообщения) некоторой фиксированной длины.

Хорошая хеш-функция должна удовлетворять следующим условиям:

1. Хеш-функция $h(M)$ должна быть чувствительна к любым изменениям входной последовательности M.

2. Хеш-функция $h(M)$ должна применяться к блоку данных любой длины.

3. Хеш-функция $h(M)$ создает выход фиксированной длины.

4. Для данного значения $h(M)$ должно быть невозможным нахождение значения M.

5. Для данного значения $h(M)$ должно быть невозможным нахождение M', такого, что $h(M') = h(M)$.

6. Вычислительно невозможно найти произвольную пару (M_1, M_2) такую, что $h(M_1) = h(M_2)$.

7. Вероятность возникновения ситуации, называемой коллизией, когда для различных входных последовательностей M_1 и M_2 совпадают значения их хеш-образов: $h(M_1) = h(M_2)$, должна быть чрезвычайно мала.

При построении хеш-образа входная последовательность M разбивается на блоки M_i фиксированной длины и обрабатывается по блочно по формуле:

$$H_i = f(H_{i-1}, M_i). \qquad (3.1)$$

Хеш-значение, вычисленное в результате обработки последнего блока сообщения, становится хеш-образом всего сообщения.

В качестве примера рассмотрим упрощенный вариант хеш-функции следующего вида:

$$H_i = (H_{i-1} + M_i)^2 \bmod n, \qquad (3.2)$$

где $n = p \cdot q$, p и q – большие простые числа, H_0 – произвольное начальное значение, M_i – i-й блок сообщения $M = \{M_1, M_2, \ldots, M_k\}$.

Например, вычислим хеш-образ для строки "БГУИР". Для перехода от символов к числовым значениям будем использовать следующее соответствие: 'А' – 1, 'Б' – 2, 'В' – 3, …, 'Я' – 33. Тогда сообщение M примет вид $M = \{2, 4, 21, 10, 18\}$. Выберем два простых числа $p = 17$и $q = 19$, тогда модуль $n = 323$. Пусть H_0 будет равен 100. Тогда используя 3.2, получим:

$$H_1 = (H_0+M_1)^2 \bmod n = (100+2)^2 \bmod 323 = 10404 \bmod 323 = 68,$$
$$H_2 = (H_1+M_2)^2 \bmod n = (68+4)^2 \bmod 323 = 5184 \bmod 323 = 16,$$
$$H_3 = (H_2+M_3)^2 \bmod n = (16+21)^2 \bmod 323 = 1369 \bmod 323 = 77,$$

$H_4 = (H_3+M_4)^2 \bmod n = (77+10)^2 \bmod 323 = 7569 \bmod 323 = 140,$
$H_5 = (H_4+M_5)^2 \bmod n = (140+18)^2 \bmod 323 = 24964 \bmod 323 = 93.$

Таким образом, хеш-образ сообщения "БГУИР" будет $h(M)=H_5=93$.

3.2. Электронная цифровая подпись

Цифровая подпись для электронных документов играет ту же роль, что и подпись, поставленная от руки в документах на бумаге: это данные, присоединяемые к передаваемому сообщению, подтверждающие, что владелец подписи составил или заверил это сообщение. Получатель сообщения с помощью цифровой подписи может проверить, что автором сообщения является именно владелец подписи, и что в процессе передачи не была нарушена целостность полученных данных.

При разработке механизма цифровой подписи возникают следующие задачи:

1.Формирование подписи таким образом, чтобы её невозможно было подделать.

2. Обеспечение возможности проверки того, что подпись действительно принадлежит указанному субъекту.

3. Предотвращение отказа субъекта от своей подписи.

3.2.1. Классическая схема создания цифровой подписи

До того, как будет происходить формирование цифровой подписи, отправитель должен сгенерировать два ключа: открытый K_o и секретный K_c. При этом закрытый ключ должен быть известен только тому, кто подписывает сообщения, а открытый — любому желающему проверить подлинность сообщения.

При создании цифровой подписи по классической схеме отправитель должен выполнить следующие действия.

1. Вычислить хеш-образ m исходного сообщения M при помощи хеш-функции h.

2. Вычислить цифровую подпись S по хеш-образу сообщения с использованием секретного ключа K_c создания подписи.

3. Сформировать новое сообщение (M, S), состоящее из исходного сообщения и добавленной к нему цифровой подписи.

Получив подписанное сообщение (M', S), получатель должен выполнить следующие действия для проверки подлинности подписи и целостности полученного сообщения:

1. Вычислить хеш-образ m' сообщения M' при помощи хеш-функции h.

2. С использованием открытого ключа проверки подписи (K_o) извлечь хеш-образ m сообщения из цифровой подписи S.

3. Сравнить вычисленное значение m' с извлеченным из цифровой подписи значением хеш-образа m. Если хеш-образы совпадают, то подпись признается подлинной.

Фальсификация сообщения при его передаче по каналу связи возможна при получении злоумышленником секретного ключа K_c или за счет проведения успешной атаки против хеш-функции. Используемые в реальных приложениях хеш-функции обладают характеристиками, делающими атаку против цифровой подписи практически не осуществимой. Например, хеш-функция SHA-1, принятая в США в качестве стандарта в 1995 году, формирующая 160-битовый хеш-образ при обработке сообщения блоками по 512 бит. С 2010 происходит переход от использования хеш-функции SHA-1 на использование SHA-2, которая может формировать хеш-образ длиной 224, 256, 512 или 1024 бит.

3.2.2. Алгоритм цифровой подписи RSA

Первой и наиболее известной во всем мире конкретной системой электронной цифровой подписи стала система RSA, математическая схема которой была разработана в 1977г. в Массачусетском технологическом институте США.

Для формирования подписи по алгоритму RSA сначала необходимо вычислить пару ключей: секретный ключ и открытый ключ, как это делается для криптосистемы RSA:

1. Выбираются два случайных простых числа p и q таких, что $p \approx q$.
2. Вычисляется их произведение $r = p*q$.
3. Вычисляется функция Эйлера для r $\varphi(r) = (p-1)*(q-1)$.
4. Выбирается открытая экспонента e такая, что $1<e<\varphi(r)$ и $(e, \varphi(r)) = 1$.
5. Вычисляется секретная экспонента d, удовлетворяющая условию $(e*d) \bmod \varphi(r) = 1$.

Пару значений $K_o=(e, r)$, которая является открытым ключом, автор передает партнерам по переписке для проверки его цифровых подписей. Значение $K_c = (d, r)$ сохраняется автором как секретный ключ подписи.

Если отправителю необходимо подписать сообщение M перед его отправкой, он сжимает сообщение M с помощью хеш-функции h в целое число m: $m = = h(M)$. Затем вычисляет цифровую подпись S под электронным документом M на основе хеш-образа m и секретного значения d:

$$S = m^d \bmod r. \qquad (3.3)$$

Пара (M, S) передается получателю как электронный документ M, подписанный цифровой подписью S, причем подпись S сформирована обладателем секретного ключа (d, r).

После приема пары (M', S) получатель вычисляет хеш-образ сообщения M' двумя различными способами. Прежде всего, он восстанавливает хеш-образ

m, применяя криптографическое преобразование подписи S с использованием открытого ключа (e, r):

$$m = S^e \bmod r. \qquad (3.4)$$

Кроме того, он находит результат хеширования m' принятого сообщения M' с помощью такой же хеш-функции h: $m' = h(M)$.

Если вычисленные значения совпадают, т. е. $h(M) = S^e \bmod r$, то получатель признает пару (M', S) подлинной.

Например, подпишем сообщение "БГУИР". Сначала получим его хеш-образ. Как показано выше, он равен $h(M)$=93. Далее сгенерируем открытый и закрытый ключи:

1. Выберем $p = 17$, $q = 19$.
2. Вычислим $r = 17*19 = 323$.
3. Вычислим $\varphi(r) = (p-1)*(q-1) = 16*18 = 288$.
4. Выберем открытую экспоненту $e = 43$, взаимно простую с $\varphi(r) = 288$.
5. На основе e и $\varphi(r)$ вычислим закрытую экспоненту $d = 67$, используя расширенный алгоритм Евклида.

Тогда открытый будет равен (43, 323), а закрытый – (67, 323). Далее подписываем сообщение:

$$S = m^d \bmod r = 93^{67} \bmod 323 = 206.$$

После чего отправляем сообщение состоящие из самого текста и подписи {БГУИР, 206}. Пусть при передачи сообщение было изменено, и получатель получил {Б**Р**УИР, 206}. Для проверки подписи сначала он вычисляет хеш-образ полученного сообщения "БРУИР":

$$H_1 = (H_0+M_1)^2 \bmod n = (100+2)^2 \bmod 323 = 10404 \bmod 323 = 68,$$
$$H_2 = (H_1+M_2)^2 \bmod n = (68+18)^2 \bmod 323 = 7396 \bmod 323 = 290,$$
$$H_3 = (H_2+M_3)^2 \bmod n = (290+21)^2 \bmod 323 = 96721 \bmod 323 = 144,$$
$$H_4 = (H_3+M_4)^2 \bmod n = (144+10)^2 \bmod 323 = 23716 \bmod 323 = 137,$$
$$H_5 = (H_4+M_5)^2 \bmod n = (137+18)^2 \bmod 323 = 24025 \bmod 323 = 123.$$

С другой стороны из цифровой подписи с помощью известного ему открытого ключа (43, 323) получатель вычисляет хеш-образ, переданный отправителем:

$$S = m^e \bmod r = 206^{43} \bmod 323 = 93.$$

Так как два вычисленных значений 123 и 93 не равны, то подпись признается недействительной.

3.2.3. Алгоритм цифровой подписи DSA

Алгоритм DSA (Digital Signature Algorithm – алгоритм цифровой подписи) был предложен Национальным институтом стандартов и технологий в августе 1991. Данный алгоритм вместе с криптографической хеш-функцией SHA-1 является частью DSS (Digital Signature Standard – стандарт цифровой подписи) − криптографического стандарта электронной цифровой подписи, используемой в США. DSA основан на трудности вычисления дискретных логарифмов и базируется на схеме, первоначально представленной Эль-Гамалем и Шнорром.

Алгоритма цифровой подписи DSA состоит в следующем. Сначала необходимо получить секретный и открытый ключи, для этого выполнить следующие действия:

1. Выбрать большое простое число q.

2. Выбрать простое число p такое, что q является делителем $(p-1)$.

3. Подобрать число g такое, что для него верно $g = h^{(p-1)/q} \bmod p$, где h − некоторое произвольное число из интервала $(1, p-1)$, и при этом $g>1$. В большинстве случаев значение $h = 2$ удовлетворяет этому требованию.

4. Закрытый ключ отправителя x выбирается случайно из интервала $(0, q)$.

5. Открытый ключ вычисляется из закрытого ключа по формуле:

$$y = g^x \bmod p. \tag{3.5}$$

Вычислить y по известному x довольно просто (используя алгоритм быстрого возведения в степень). Однако, имея открытый ключ y, вычислительно невозможно определить x, который является дискретным логарифмом y по основанию g.

Открытой информацией являются значения p, q и y, закрытой – x. При этом значения p и q могут быть общими для группы пользователей, а значеие y и x – для каждого свое.

Подпись сообщения выполняется по следующему алгоритму:

1. Получаем хеш-образ исходного сообщения $h(M)$. При использовании формулы 3.2 вычисления необходимо выполнять по модулю числа q.

2. Выбирается случайное число k из $(0, q)$, уникальное для каждого подписи.

3. Вычисляется значение r и s по формулам:

$$\begin{aligned} r &= (g^k \bmod p) \bmod q, \\ s &= k^{-1}(h(M)+x*r) \bmod q. \end{aligned} \tag{3.6}$$

4. Если одно из полученных значений r или s будет равно 0, то необходимо повторить вычисления для другого значения k. Иначе, подписью будет пара значений (r, s).

Таким образом сообщение с подписью будет иметь вид $\{M, r, s\}$.

Для того чтобы проверить подлинность подписи, сначала из полученного сообщения $\{M', r, s\}$ вычисляется хеш-образ $h(M')$, после чего находят значение v, используя формулы 3.7. Подпись признается подлинной, если $v = r$.

$$\begin{aligned} w &= s^{-1} \bmod q, \\ u_1 &= h(M)*w \bmod q, \\ u_2 &= r*w \bmod q, \\ v &= (g^{u1}*y^{u2} \bmod p) \bmod q. \end{aligned} \qquad (3.7)$$

Приведем пример данного алгоритма подписи. Возьмем приведенное выше сообщение "БГУИР", хеш-образ которого равен 93. Далее сгенерируем открытый и закрытый ключи для создания подписи. Для этого выберем случайные простые числа q и p, пусть они будут равны соответственно 107 и 643. Как видно $p-1$ (642) делится на q (107) без остатка. Тогда число будет g равно 64. Далее выберем случайное число x = 45, которое будет секретным ключом и храниться в секрете, и вычислим для него открытый ключ по формуле 3.5: $y = = g^x \bmod p = 64^{45} \bmod 643 = 181$. Значение y является открытой информацией.

Вычислим цифровую подпись для сообщения. Для этого возьмем его хеш-образ $h(M) = 93$, сгенерируем случайное число $k = 31$, и вычислим r, s по формулам 3.6:

$$r = (g^k \bmod p) \bmod q = (64^{31} \bmod 643) \bmod 107 = 36,$$

$$s = k^{-1}(h(m) + x*r) \bmod q = \frac{1}{31}(93 + 45*36) \bmod 107 = 31^{\varphi(q)-1} * 1713 \bmod 107 = 38.$$

Так как оба полученных значения r и s не равны 0, то подпись будет равна паре значений (36, 38). И отправляемое сообщение будет иметь вид: {БГУИР, 36, 38}.

Для проверки подлинности подписи получатель выполняет следующие действия. Сначала он вычисляет хеш-образ сообщения "БГУИР", которое равно 93. Далее вычисляет значение v по формулам 3.8.

$$\begin{aligned} w &= s^{-1} \bmod q = 38^{105} \bmod 107 = 31, \\ u_1 &= h(M)*w \bmod q = 93*31 \bmod 107 = 101, \\ u_2 &= r*w \bmod q = 36*31 \bmod 107 = 46, \\ v &= (g^{u1}*y^{u2} \bmod p) \bmod q = (64^{101}*181^{46} \bmod 643) \bmod 107 = 36. \end{aligned}$$

Так как $r = v$ (36 = 36), то подпись является подлинной.

Задание для выполнения лабораторной работы №3

1. Изучить теоретический материал по лабораторной работе.

2. Реализовать программу вычисления и проверки электронной цифровой подписи для варианта №1 по схеме RSA, для варианта №2 по схеме DSA. Для вычисления хеш-образа сообщения использовать функцию 3.2.

4. ДОКАЗАТЕЛЬСТВО С НУЛЕВЫМ ЗНАНИЕМ

В реальном мире для доказательств подлинности часто используются физические символы: паспорта, водительские права, кредитные карточки и т.д. Эти символы содержат что-то, связывающее их с конкретным человеком: обычно фотографию или подпись, но с той же вероятностью это может быть отпечаток пальца, снимок сетчатки глаза или рентгеновский снимок челюсти.

Использовать доказательства с нулевым знанием для доказательства идентичности было впервые предложено Уриелем Файгом, Амосом Фиатом и Ади Шамиром. Закрытый ключ пользователя становится функцией его "идентичности". Используя доказательство с нулевым знанием, он доказывает, что знает свой закрытый ключ, не раскрывая его, и, таким образом, свою идентичность. Таким образом, доказательство с нулевым знанием – это интерактивный протокол, позволяющий одной из сторон (проверяющему) убедиться в достоверности какого-либо утверждения (обычно математического), не получив при этом никакой другой информации от второй стороны (доказывающего).

Доказательство с нулевым знанием должно обладать тремя свойствами:

1. Полнота: если утверждение действительно верно, то доказывающий убедит в этом проверяющего.

2. Корректность: если утверждение неверно, то даже нечестный доказывающий не сможет убедить проверяющего за исключением пренебрежимо малой вероятности.

3. Нулевое разглашение: если утверждение верно, то любой даже нечестный проверяющий не узнает ничего кроме самого факта, что утверждение верно.

4.1. Алгоритм Фиата-Шамира

Одним из наиболее известных протоколов идентификации личности с помощью доказательства с нулевым знанием является протокол, предложенный Амосом Фиатом и Ади Шамиром. Стойкость данного протокола основывается на сложности извлечения квадратного корня по модулю достаточно большого составного числа n, факторизация которого неизвестна.

Доверенный центр T выбирает и публикует модуль $n = p*q$, где p, q – простые числа и держатся в секрете, при этом n достаточно большое число, разложить на множители которое трудно. Каждый пользователь A выбирает секретное $s \in [1, n-1]$ взаимно простое с n. Затем вычисляется открытый ключ:

$$v = s^2 \pmod{n}. \qquad (4.1)$$

Полученное v регистрируется центром доверия в качестве открытого ключа пользователя A, а значение s является секретом A. Именно знание этого секрета s необходимо доказать A некоторой стороне B (проверяющему) без его

разглашение за t раундов (аккредитаций). Каждая аккредитация состоит из следующих этапов:

1. A выбирает случайное $r \in [1, n-1]$ и отсылает $x = r^2 \pmod n$ стороне B.

2. B случайно выбирает бит $e \in \{0, 1\}$ и отсылает его A.

3. A вычисляет y

$$y = r*s^e \pmod n. \qquad (4.2)$$

и отправляет его обратно к B.

4. Сторона B проверяет равенство $y^2 \equiv x*v^e \pmod n$. Если оно верно, то происходит переход к следующему раунду протокола, иначе доказательство не принимается.

Вероятность того, что пользователь A не знает секрета s, но убеждает в обратном проверяющего B будет оцениваться вероятностью равной $p = 2^{-t}$, где t – число аккредитаций. Для достижения высокой достоверности его выбирают достаточно большим ($t = 20 - 40$). Таким образом, B удостоверяется в знании A тогда и только тогда, когда все t раундов прошли успешно.

Приведем пример работы алгоритма Фиата-Шамира на маленьких числах. Пусть доверенный центр выбрал простые $p = 11$ и $q = 19$, тогда $n = 11*19 = 209$. Пользователь A выбирает $s = 43$. Откуда по формуле 4.1 вычисляем открытый ключ $v = 43^2 \pmod{209} = 1849 \pmod{209} = 177$. Таким образом, в секрете держаться 11, 19 и 43, а 177 и 209 публикуются открыто. Теперь проведем пример аккредитации для доказательства стороне B, что открытый ключ v принадлежит именно A тем, что A знает секрет s:

1. А выбирает $r = 38$ и считает $x = 38^2 \pmod{209} = 1444 \pmod{209} = 190$.

2. B выбирает бит $e \in \{0, 1\}$ и отсылает его A.

3. Если B отправил $e = 0$, то A согласно 4.2 возвращает $y = r \pmod n = 38$, иначе A возвращает $y = r*s \pmod n = 38*43 \pmod{209} = 1634 \pmod{209} = 171$.

4. B проверяет полученные значения. Если e было выбрано 0, то $y^2 = 38^2 \pmod{2099} = 190$ и $x = 190$, что подтверждает знание s стороной A. Если e же было выбрано 1, то $y^2 = 171^2 \pmod{209} = 29241 \pmod{209} = 190$ и $x*v = 190 * 177 \pmod{209} = 33630 \pmod{209} = 190$.

Для того чтобы этот протокол корректно выполнялся, A никогда не должен повторно использовать значение x. Если бы A поступил таким образом, а B во время другого цикла отправил бы A на шаге 2 другой случайный бит r, то B бы имел оба ответа A. После этого B может вычислить значение s, и ему будет известен секретный ключ Алисы.

4.2. Алгоритм Гиллу-Кискатра

Алгоритм Гиллу-Кискатра является развитием схемы Фиата-Шамира. Он позволяет сократить число аккредитаций t, часто $t = 1$, и объем используемой памяти.

Согласно ему, доверенный центр выбирает и публикует модуль $n = p*q$, где p, q − простые секретные числа, найти которые, зная n, невозможно, и открытую экспоненту $v \geq 3$ такую, что $(v, \varphi(n)) = 1$ ($\varphi(n) = (p-1)(q-1)$). Используя эти данный вычисляется секретная экпонента

$$s = v^{-1} \bmod \varphi(n). \tag{4.3}$$

Параметры (v, n) являются общедоступными и могут быть использованы для генерации открытого и закрытого ключей группой пользователей.

Далее каждому пользователю доверенным центром выдается некоторый публичный идентификатор его личности Id, при этом $1<Id<n$, и вычисляется некоторый секрет s_A:

$$s_A = Id^{-s} \bmod n. \tag{4.4}$$

A посылает проверяющему B свои атрибуты Id. И для доказательства того, что Id − это именно его идентификатор, пользователь A должен убедить B, что ему известен секрет s_A. Вот этот протокол доказательства:

1. A выбирает случайное целое $r \in [1, n-1]$ и отправляет $x = r^v \bmod n$ проверяющему B.

2. B выбирает случайное целое $e \in [1, v-1]$ и отсылает его A.

3. A вычисляет

$$y = rs_A^e \bmod n \tag{4.5}$$

и посылает y B.

4. B вычисляет

$$z = y^v Id^e \bmod n \tag{4.6}$$

идентификатор A, и если $z = x$, то подлинность A доказана.

Приведем пример работы данного протокола на маленьких числах, для этого возьмем $n = p*q = 11*19 = 209$ и $v = 43$, и вычислим s согласно формуле 4.3: $s = v^{-1} \bmod \varphi(n) = 43^{-1} \bmod 180 = 43^{\varphi(n)-1} \bmod 180 = 43^{47} \bmod 180 = 67$. Далее, присвоим пользователю A идентификатор $Id = 31$ и вычислим его секрет s_A по формуле 4.4: $s_A = Id^{-s} \bmod n = 31^{-67} \bmod 209 = 31^{67*(\varphi(n)-1)} \bmod 209 = 31^{67*179} \bmod 209 = 179$. Для доказательства знания секрета s_A без его разглашение пользователь A действует согласно схеме:

1. A выбирает случайное целое $r = 23$ и отправляет $x = 23^{43} \bmod 209 = 177$ проверяющему B.

2. B выбирает случайное целое $e = 17$ и отсылает 17 A.

3. A вычисляет по формуле 4.5 $y = rs_A^e \bmod n = 23*179^{17} \bmod 209 = 86$ и посылает его B.

4. Так как $y^v Id^e \bmod n = 86^{43}31^{17} \bmod 209 = 177$ и $x = 177$, то подлинность A доказана.

4.3. Алгоритм Шнорра

Алгоритм Шнорра является альтернативой алгоритмам Фиата-Шамира и Гиллу-Кискатра. Его особенностью является то, что позволяет проводить предварительные вычисления, что удобно при малых вычислительных ресурсах. Надежность данного алгоритма основывается на сложности вычисления дискретного логарифма.

Доверенный центр выбирает два простых числа p и q таких, что $p-1$ делится без остатка на q, и выбирается элемент $g \neq 1$ такой, что для него верно $g = = h^{(p-1)/q} \bmod p$, где h − некоторое произвольное число из интервала $(1, p-1)$. Параметры (p, q, g) являются открытой информацией и свободно публикуются. Они могут быть общими для группы пользователей. Также выбирается параметр t $(40 \le t < q)$, определяющий уровень безопасности.

Далее доказывающая сторона A выбирает секрет $s \in [1, q-1]$ и вычисляет

$$v = g^{-s} (\bmod p), \tag{4.7}$$

где v является открытым ключом A, который посылается доверенному центру и там публикуется как открытое значение пользователя A.

Для доказательства знания секрета s проверяющему B пользователь A выполняет следующие действия:

1. A выбирает случайное $r \in [1, q-1]$ и вычисляет

$$x = g^r \ (\bmod p). \tag{4.8}$$

Далее, x отсылает стороне B.

2. Сторона B отсылает случайное e, где $1 \le e \le 2^t$.

3. A возвращает B $y = (s*e+r) \ (\bmod q)$.

4. B вычисляет

$$z = g^y * v^e \ (\bmod p). \tag{4.9}$$

Если $z = x$, то идентификация считается успешной.

Например, выберем простые p=643 и q=107, при этом выполняется условие, что 107|642. Далее вычислим g из условия $g = h^{(p-1)/q} \bmod p$, где h − некоторое произвольное число из интервала $(1, p-1)$. В данном случае для $h = 2$ получим $g = 64$.

Сторона A выбирает закрытый ключ s=43 и вычисляет по формуле 4.7 открытый ключ $v = g^{-s} \ (\bmod p) = 64^{-43} \bmod 643 = 623$.

Доказательство происходит следующим образом:

1. Сторона A случайным образом выбирает доказательство r=27 и отсылает $x = g^r (\mathrm{mod}\ p) = 64^{27} \bmod 643 = 231$ стороне B.

2. B отсылает A вызов e=19.

3. A отсылает B $y = (s*e+r) \ (\mathrm{mod}\ q) = (43*19 + 27) \ (\mathrm{mod}\ 107) = 95$.

4. B вычисляет $g^y * v^e (\mathrm{mod}\ p) = 64^{95} * 623^{19} \ (\mathrm{mod}\ 643) = 231$, что равно x.

Задание для выполнения лабораторной работы №4

1. Изучить теоретический материал по лабораторной работе.

2. Реализовать алгоритм доказательства с нулевым знанием по алгоритму согласно своему варианту (см. табл. 4.1)

Таблица 4.1

Вариант	№1	№2	№3
Алгоритм	Фиата-Шамира	Гиллу-Кискатра	Шнорра

ЛИТЕРАТУРА

1. Венбо, М. Современная криптография: теория и практика / М. Венбо. – М. : Издательский дом «Вильямс», 2005. – 768 с.
2. Виноградов, И.М. Основы теории чисел / И.М. Виноградов. – М.: Наука, 1981. – 176 с.
3. Гинзбург, А.И. Пластиковые карты / А.И. Гинзбург. – СПб. : "Питер", 2004. – 128 с.
4. Грибунин, В.Г. Цифровая стеганография / В.Г. Грибунин, И. Н. Оков, И. В. Туринцев. – М. : СОЛОН-Пресс, 2002. – 272 с.
5. Деднев, М.А. Защита информации в банковском деле и электронном бизнесе / М.А. Деднев, Д.В. Дыльнов. − М. : Кудиц-Образ,2004. – 512 с.
6. Кнут, Д. Искусство программирования для ЭВМ: В 3– х томах. Получисленные методы / Д. Кнут – М. : Мир. Т.2. – 724 с.
7. Лидл, Р. Конечные поля / Р. Лидл, Т. Нидеррайтер – М. : Мир, 1988. – 820 с.
8. Молдовян, Н.А. Введение в криптосистемы с открытым ключом: Проблематика криптографии; элементы теории чисел; двухключевые криптосистемы и др.: Учебное пособие для вузов / Н.А. Молдовян, А.А. Молдовян.- СПб.: Издательство БВХ–Петербург, 2005. – 288 с.
9. Романец, Ю.В. Защита информации в компьютерных системах и сетях / Ю.В. Романец, П.А. Тимофеев, В.Ф. Шаньгин. – М.: Радио и связь, 1999. – 328 с.
10. Смарт, Н. Криптография / Н. Смарт. – М. : Техносфера, 2005. – 528 с.
11. Харин, Ю.С. Компьютерный практикум по математическим методам защиты информации / Ю.С. Харин, С.В. Агиевич.– Минск : БГУ, 2001. – 190 с.
12. Харин, Ю.С. Математические основы криптологии / Ю.С. Харин, В.И. Берник, Г.В. Матвеев. - Минск : БГУ, 1999. – 319 с.
13. Харин, Ю.С. Математические и компьютерные основы криптологии / Ю.С. Харин, В.И. Берник, Г.В. Матвеев, С.В. Агиевич. – Минск : Новое Знание, 2003. – 382 с.
14. Шеннон, К. Работы по теории информации и кибернетике / К. Шеннон. – М. : Издательство иностранной литературы, 1963. – 830 с.
15. Шнайер, Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнаейр. – М. : ТРИУМФ, 2002. – 816 с.
16. Ярмолик, В.Н. Элементы теории информации. Практикум для студентов специальности “Программное обеспечение информационных технологий” / В.Н. Ярмолик, А.П. Занкович, С.С. Портянко. – Минск : БГУИР, 2007. – 40 с.
17. Ярмолик, В.Н. Криптография, стеганография и охрана авторского права / В.Н. Ярмолик, С.С. Портянко, С.В. Ярмолик. – Минск : Издательский центр БГУ, 2007. – с.

Св. план 2010, поз. 49

Учебное издание

Ярмолик Светлана Вячеславовна
Ярмолик Вячеслав Николаевич

КРИПТОГРАФИЯ И ОХРАНА КОММЕРЧЕСКОЙ ИНФОРМАЦИИ

Методическое пособие по выполнению лабораторных работ
для студентов специальностей 1-40 01 02 – 02 и 1-26 02 03
дневной и заочной форм обучения

Редактор
Корректор

Подписано в печать | Формат 60×84 1/16. | Бумага офсетная.
Гарнитура «Таймс». | Оптечатано на ризографие | Усл. печ. л. 2.
Уч.-изд. л. 2. | Тираж 100 экз. | Заказ .

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