2.3 Перечень тем лабораторных занятий по дисциплине Методы защиты информации

Основная цель проведения лабораторных занятий состоит в закреплении теоретического материала курса, приобретении навыков выполнения эксперимента, обработки экспериментальных данных, анализа результатов, грамотного оформления отчетов.

№ темы по п.1	Наименование лабораторной работы	Содержание
3	Лабораторная работа № 1 Симметричная криптография. Стандарт шифрования ГОСТ 28147-89	Реализовать программные средства шифрования и дешифрования текстовых файлов при помощи стандарта шифрования ГОСТ 28147-89 в следующих режимах: 1) Простой замены 2) Гаммирования 3) Гаммирования с обратной связью 4) Генерации имитоприставок
3	Лабораторная работа № 2 Симметричная криптография. СТБ 34.101.31-2011	Реализовать программные средства шифрования и дешифрования текстовых файлов при помощи алгоритма СТБ 34.101.31-2011 в различных режимах
4	Лабораторная работа № 3 Асимметричная криптография. Криптосистема Рабина	Реализовать криптостойкое программное средство шифрования и дешифрования текстовых файлов при помощи Криптосистемы Рабина
4	Лабораторная работа № 4 Асимметричная криптография. Алгоритм Мак-Элиса	Реализовать программные средства шифрования и дешифрования текстовых файлов при помощи алгоритма Мак-Элиса для криптостойких размеров порождающей матрицы
5	Лабораторная работа № 5 Хеш-функции	Реализовать программное средство контроля целостности сообщений с помощью вычисления хеш-функции и алгоритма ГОСТ 34.11 и SHA 1
6	Лабораторная работа № 6 Цифровая подпись	Реализовать программное средство формирования и проверки ЭЦП на базе алгоритма ГОСТ 34.10
7	Лабораторная работа № 7 Криптография с ис-пользованием эллипти-ческих кривых	Реализовать схему шифрования (дешифрования) для аналога алгоритма Эль-Гамаля на основе эллиптических кривых
8	Лабораторная работа № 8 Стеганографические методы	Реализовать программное средство, сокрытия (извлечения) текстового сообщения в (из) JPEG изображение(я) на основе метода сокрытия в частотной области изображения

По результатам работы студентом должен быть представлен и защищен отчет.

Содержание отчета включает:

1. Титульный лист, подписанный студентом и преподавателем с оценкой после защиты работы

2. Введение, содержащее общие теоретические сведения и индивидуальную постановку задачи.

3. Полную блок – схему разработанного алгоритма.

4. Распечатку скриншотов результатов ввода данных и исполнения программы.

5. Распечатку программного кода.

6. Вывод о решении поставленной задачи и иные сведения, относящиеся к процессу ее решения.

Лабораторная работа № 1. Симметричная криптография. Стандарт шифрования ГОСТ 28147-89

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Общие сведения

ГОСТ 28147-89 «*Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования*» — устаревший государственный стандарт СССР (позже межгосударственный стандарт СНГ), описывающий алгоритм симметричного блочного шифрования и режимы его работы.

Является примером DES-подобных криптосистем, созданных по классической итерационной схеме Фейстеля.

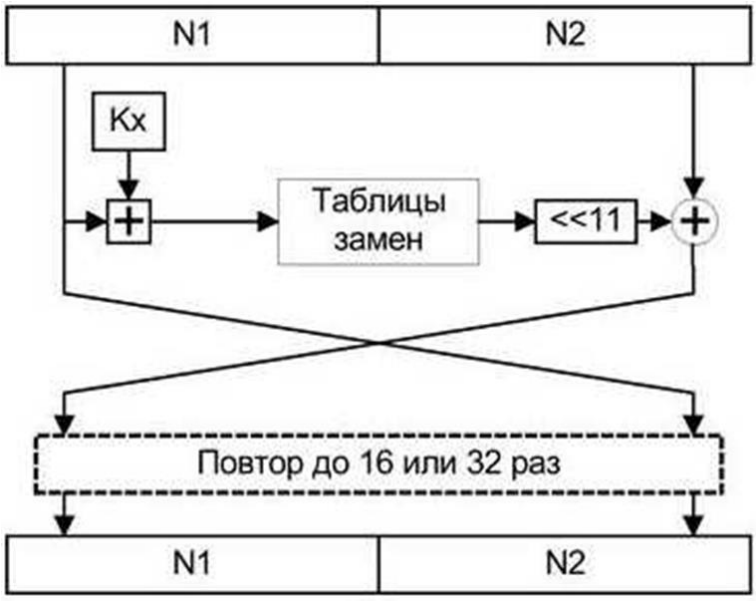
ГОСТ 28147-89 представляет собой симметричный 64-битовый блочный алгоритм с 256-битовым ключом.

Этот алгоритм криптографического преобразования данных предназначен для аппаратной и программной реализации, удовлетворяет криптографическим требованиям и до 1983 года не накладывал ограничений на степень секретности защищаемой информации.

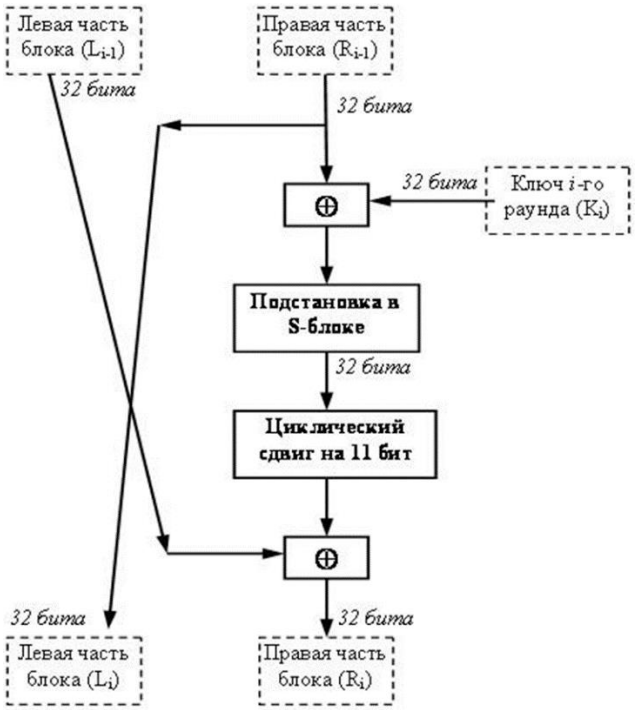
Описание работы ГОСТ 28147-89

Схема работы алгоритма ГОСТ 28147-89 следующая. Данные, подлежащие зашифровке, разбивают на 64-разрядные блоки.

Эти блоки разбиваются на два субблока N1 и N2 по 32 бит



- Структура одного раунда ГОСТ 28147-89\



Шифруемый блок данных разбивается на две части, которые затем обрабатываются как отдельные 32-битовые целые числа без знака.

Сначала правая половина блока и подключ раунда складываются по модулю 2^{32}.

Затем производится поблочная подстановка.

32-битовое значение, полученное на предыдущем шаге (обозначим его S), интерпретируется как массив из восьми 4-битовых блоков кода: $S=(S_0,S_1,S_2,S_3,S_4,S_5,S_6,S_7)$.

Далее значение каждого из восьми блоков заменяется на новое, которое выбирается **по таблице замен.**

Таблица замен ГОСТ 28147-89

Номер S-блока \ x_{ij}	Значение y_{ij}															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	4	10	9	2	13	8	0	14	6	11	1	12	7	15	5	3
2	14	11	4	12	6	13	15	10	2	3	8	1	0	7	5	9
3	5	8	1	13	10	3	4	2	14	15	12	7	6	0	9	11
4	7	13	10	1	0	8	9	15	14	4	6	12	11	2	5	3
5	6	12	7	1	5	15	13	8	4	10	9	14	0	3	11	2
6	4	11	10	0	7	2	1	13	3	6	8	5	9	12	15	14
7	13	11	4	1	3	15	5	9	0	10	14	7	6	8	2	12
8	1	15	13	0	5	7	10	4	9	2	3	14	6	11	8	12

В каждой строке таблицы замен записаны числа от 0 до 15 в произвольном порядке без повторений.

Значения элементов таблицы замен взяты от 0 до 15, так как в четырех битах, которые подвергаются подстановке, может быть записано целое число без знака в диапазоне от 0 до 15.

Значение блока S1 (четыре младших бита 32-разрядного числа S) заменится на число, стоящее на позиции, номер которой равен значению заменяемого блока.

Например, в этом случае S1=0 заменится на 4, если S1=1, то оно заменится на 10 и т.д.

После выполнения подстановки все 4-битовые блоки снова объединяются в единое 32-битное слово, которое затем циклически сдвигается на 11 битов влево.

Наконец, с помощью побитовой операции "сумма по модулю 2" результат объединяется с левой половиной, вследствие чего получается новая правая половина Ri.

Новая левая часть Li берется равной младшей части преобразуемого блока: Li= Ri-1.

Полученное значение преобразуемого блока рассматривается как результат выполнения одного раунда алгоритма шифрования.

Процедуры шифрования и расшифрования

ГОСТ 28147-89 является блочным шифром, поэтому преобразование данных осуществляется блоками в так называемых базовых циклах.

Базовые циклы заключаются в многократном выполнении для блока данных основного раунда, рассмотренного нами ранее, с использованием разных элементов ключа и отличаются друг от друга порядком использования ключевых элементов.

В каждом раунде используется один из восьми возможных 32-разрядных подключей.

Рассмотрим процесс создания подключей раундов. В ГОСТ эта процедура очень проста, особенно по сравнению с DES. 256-битный ключ K разбивается на восемь 32-битных подключей, обозначаемых K0, K1, K2, K3, K4, K5, K6, K7. Алгоритм включает 32 раунда, поэтому каждый подключ при

шифровании используется в четырех раундах в последовательности, представленной в таблице

Таблица 1. Последовательность использования подключей при шифровании

Раунд	1	2	3	4	5	6	7	8
Подключ	K0	K1	K2	K3	K4	K5	K6	K7
Раунд	9	10	11	12	13	14	15	16
Подключ	K0	K1	K2	K3	K4	K5	K6	K7
Раунд	17	18	19	20	21	22	23	24
Подключ	K0	K1	K2	K3	K4	K5	K6	K7
Раунд	25	26	27	28	29	30	31	32
Подключ	K7	K6	K5	K4	K3	K2	K1	K0

Процесс расшифрования производится по тому же алгоритму, что и шифрование. Единственное отличие заключается в порядке использования подключей Ki. При расшифровании подключи должны быть использованы в обратном порядке, а именно, как указано в таблице

Таблица 2. Последовательность использования подключей при расшифровании

Раунд	1	2	3	4	5	6	7	8
Подключ	K0	K1	K2	K3	K4	K5	K6	K7
Раунд	9	10	11	12	13	14	15	16
Подключ	K7	K6	K5	K4	K3	K2	K1	K0
Раунд	17	18	19	20	21	22	23	24
Подключ	K7	K6	K5	K4	K3	K2	K1	K0
Раунд	25	26	27	28	29	30	31	32
Подключ	K7	K6	K5	K4	K3	K2	K1	K0

ГОСТ 28147-89 Режимы работы

Режим простой замены: все блоки шифруются независимо друг от друга с разными подключами в разных раундах. Для одинаковых блоков сообщения M блоки шифртекста будут одинаковыми.

Режим гаммирования: В регистры N1 и N2 записывается 64-битовая синхропосылка (вектор инициализации) и шифруется с использованием СК. Результат подается на вход регистров и снова шифруется с использованием ключа. Получается «одноразовый блокнот».

В режиме гаммирования с обратной связью для заполнения регистров N1 и N2, начиная со 2-го блока, используется результат зашифрования предыдущего блока открытого текста.

Рисунок 4. Работа криптосистемы в режиме гаммирования

ЗАДАНИЕ:

1. Изучить теоретические сведения.
2. Реализовать программные средства шифрования и дешифрования текстовых файлов при помощи стандарта шифрования ГОСТ 28147-89 в следующих режимах:

- ✓ Простой замены
- ✓ Гаммирования
- ✓ Гаммирования с обратной связью
- ✓ Генерации имитовставок

Лабораторная работа № 2. Симметричная криптография. СТБ 34.101.31-2011

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

2.1. Общие сведения

Настоящий стандарт определяет семейство криптографических алгоритмов, предназначенных для обеспечения конфиденциальности и контроля целостности данных. Обрабатываемыми данными являются двоичные слова (сообщения).

Криптографические алгоритмы стандарта построены на основе базовых алгоритмов шифрования блока данных.

Криптографические алгоритмы шифрования и контроля целостности делятся на восемь групп:

1) алгоритмы шифрования в режиме простой замены;
2) алгоритмы шифрования в режиме сцепления блоков;
3) алгоритмы шифрования в режиме гаммирования с обратной связью;
4) алгоритмы шифрования в режиме счетчика;
5) алгоритм выработки имитовставки;
6) алгоритмы одновременного шифрования и имитозащиты данных;
7) алгоритмы одновременного шифрования и имитозащиты ключа;
8) алгоритм хеширования.

Первые четыре группы предназначены для обеспечения конфиденциальности сообщений. Каждая группа включает алгоритм зашифрования и алгоритм расшифрования.

Стороны, располагающие общим ключом, могут организовать конфиденциальный обмен сообщениями путем их зашифрования перед отправкой и расшифрования после получения. В режимах простой замены и сцепления блоков шифруются сообщения, которые содержат хотя бы один блок, а в режимах гаммирования с обратной связью и счетчика – сообщения произвольной длины.

Пятый алгоритм предназначен для контроля целостности сообщений с помощью имитовставок – контрольных слов, которые определяются с использованием ключа. Стороны, располагающие общим ключом, могут организовать контроль целостности при обмене сообщениями путем добавления к ним имитовставок при отправке и проверки имитовставок при получении. Проверка имитовставок дополнительно позволяет стороне-получателю убедиться в том, что сторона-отправитель знает ключ, т. е. позволяет проверить подлинность сообщений.

2.2. Блок-схема алгоритма шифрования в режиме простой замены

Блок-схема алгоритма на i-ом такте шифрования представлена на рисунке 1.

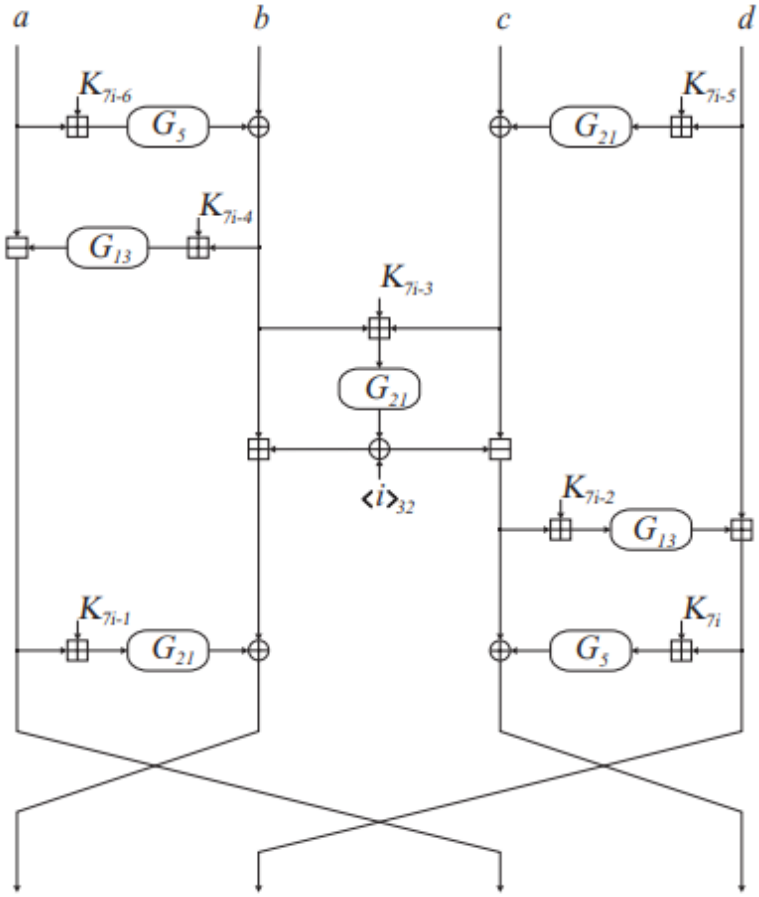


Рисунок 2.1 – Вычисления на i-м такте шифрования

Входными данными алгоритмов зашифрования и расшифрования являются блок $X \in \{0,1\}^{128}$ и ключ $\theta \in \{0,1\}^{256}$.

Выходными данными является блок $Y \in \{0,1\}^{128}$ — результат зашифрования либо расшифрования слова X на ключе $\theta : Y = F_\theta(X)$ либо $Y = F_\theta^{-1}(X)$.

Входные данные для шифрования подготавливаются следующим образом:

- Слово X записывается в виде $X = X_1 \| X_2 \| X_3 \| X_4, X_i \in \{0,1\}^{32}$.
- Ключ θ записывается в виде $\theta = \theta_1 \| \theta_2 \| \theta_3 \| \theta_4 \| \theta_5 \| \theta_6 \| \theta_7 \| \theta_8, \theta_i \in \{0,1\}^{32}$ и определяются тактовые ключи $K_1 = \theta_1, K_2 = \theta_2, K_3 = \theta_3, K_4 = \theta_4, K_5 = \theta_5, K_6 = \theta_6, K_7 = \theta_7, K_8 = \theta_8, K_9 = \theta_1, \ldots\ldots, K_{56} = \theta_8.$

Обозначения и вспомогательные преобразования

Преобразование $G_r : \{0,1\}^{32} \to \{0,1\}^{32}$ ставит в соответствие слову $u = u_1 \parallel u_1 \parallel u_2 \parallel u_3 \parallel u_4, u_i \in \{0,1\}^8$, слово

$$G_r(u) = RotHi^r(H(u_1) \parallel H(u_2) \parallel H(u_3) \parallel H(u_4)).$$

$RotHi^r$ циклический сдвиг влево на r бит.

$H(u)$ операция замены 8-битной входной строки подстановкой с рисунка 4.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	B1	94	BA	C8	0A	08	F5	3B	36	6D	00	8E	58	4A	5D	E4
1	85	04	FA	9D	1B	B6	C7	AC	25	2E	72	C2	02	FD	CE	0D
2	5B	E3	D6	12	17	B9	61	81	FE	67	86	AD	71	6B	89	0B
3	5C	B0	C0	FF	33	C3	56	B8	35	C4	05	AE	D8	E0	7F	99
4	E1	2B	DC	1A	E2	82	57	EC	70	3F	CC	F0	95	EE	8D	F1
5	C1	AB	76	38	9F	E6	78	CA	F7	C6	F8	60	D5	BB	9C	4F
6	F3	3C	65	7B	63	7C	30	6A	DD	4E	A7	79	9E	B2	3D	31
7	3E	98	B5	6E	27	D3	BC	CF	59	1E	18	1F	4C	5A	B7	93
8	E9	DE	E7	2C	8F	0C	0F	A6	2D	DB	49	F4	6F	73	96	47
9	06	07	53	16	ED	24	7A	37	39	CB	A3	83	03	A9	8B	F6
A	92	BD	9B	1C	E5	D1	41	01	54	45	FB	C9	5E	4D	0E	F2
B	68	20	80	AA	22	7D	64	2F	26	87	F9	34	90	40	55	11
C	BE	32	97	13	43	FC	9A	48	A0	2A	88	5F	19	4B	09	A1
D	7E	CD	A4	D0	15	44	AF	8C	A5	84	50	BF	66	D2	E8	8A
E	A2	D7	46	52	42	A8	DF	B3	69	74	C5	51	EB	23	29	21
F	D4	EF	D9	B4	3A	62	28	75	91	14	10	EA	77	6C	DA	1D

Рисунок 2.2 – Преобразование H

Подстановка $H : \{0,1\}^8 \to \{0,1\}^8$ задается фиксированной таблицей. В таблице используется шестнадцатеричное представление слов $u \in \{0,1\}^8$

$\boxplus$ и $\boxminus$ операции сложения и вычитания по модулю 2^{32}

Для зашифрования блока X на ключе θ выполняются следующие шаги:

1. Установить $a \leftarrow X_1, b \leftarrow X_2, c \leftarrow X_3, d \leftarrow X_4$.

2. Для i = 1,2,… ,8 выполнить:

1) $b \leftarrow b \oplus G_5(a \boxplus K_{7i-6})$;
2) $c \leftarrow c \oplus G_{21}(d \boxplus K_{7i-5})$;
3) $a \leftarrow a \boxminus G_{13}(b \boxplus K_{7i-4})$;
4) $e \leftarrow G_{21}(b \boxplus c \boxplus K_{7i-3}) \oplus \langle i \rangle_{32}$;
5) $b \leftarrow b \boxplus e$;
6) $c \leftarrow c \boxminus e$;
7) $d \leftarrow d \boxplus G_{13}(c \boxplus K_{7i-2})$;
8) $b \leftarrow b \oplus G_{21}(a \boxplus K_{7i-1})$;
9) $c \leftarrow c \oplus G_5(d \boxplus K_{7i})$;
10) $a \leftrightarrow b$;
11) $c \leftrightarrow d$;
12) $b \leftrightarrow c$.

3. Установить $Y \leftarrow b \| d \| a \| c$.
4. Возвратить Y.

Для расшифрования блока X на ключе θ выполняются следующие шаги:

1. Установить $a \leftarrow X_1, b \leftarrow X_2, c \leftarrow X_3, d \leftarrow X_4$.
2. Для i = 8,7,… ,1 выполнить:

1) $b \leftarrow b \oplus G_5(a \boxplus K_{7i})$;
2) $c \leftarrow c \oplus G_{21}(d \boxplus K_{7i-1})$;
3) $a \leftarrow a \boxminus G_{13}(b \boxplus K_{7i-2})$;
4) $e \leftarrow G_{21}(b \boxplus c \boxplus K_{7i-3}) \oplus \langle i \rangle_{32}$;
5) $b \leftarrow b \boxplus e$;
6) $c \leftarrow c \boxminus e$;
7) $d \leftarrow d \boxplus G_{13}(c \boxplus K_{7i-4})$;
8) $b \leftarrow b \oplus G_{21}(a \boxplus K_{7i-5})$;
9) $c \leftarrow c \oplus G_5(d \boxplus K_{7i-6})$;
10) $a \leftrightarrow b$;
11) $c \leftrightarrow d$;
12) $a \leftrightarrow d$.

3. Установить $Y \leftarrow c \| a \| d \| b$.
4. Возвратить Y.

.....

2.3 Шифрование в режиме гаммирования с обратной связью

Входными данными алгоритмов зашифрования и расшифрования являются сообщение $X \in \{0, 1\}^*$, ключ O$\in\{0, 1\}^{256}$ и синхропосылка $S \in \{0, 1\}^{128}$.

Выходными данными является слово $Y \in \{0, 1\}^{|X|}$ — результат зашифрования либо расшифрования X на ключе θ при использовании синхропосылки S.

Входное сообщение X записывается в виде $X = X_1 \| X_2 \| \ldots \| X_n$, $|X_1| = |X_2| = \ldots = |X_{n-1}| = 128$, $|X_n| <= 128$.

При шифровании словам Xi ставятся в соответствие слова $Y_i \in \{0, 1\}^{|Xi|}$, из которых затем составляется Y .

При зашифровании используется вспомогательный блок $Y_0 \in \{0, 1\}^{128}$, а при расшифровании — вспомогательный блок $X0 \in \{0, 1\}^{128}$

Зашифрование сообщения X на ключе θ при использовании синхропосылки S состоит в выполнении следующих шагов:

1. Установить $Y_0 \leftarrow S$.
2. Для $i = 1, 2, \ldots, n$ выполнить: $Y_i \leftarrow X_i \oplus L|X_i|(F(Y_{i-1}))$.
3. Установить $Y \leftarrow Y_1 \| Y_2 \| \ldots \| Y_n$.
4. Возвратить Y.

Расшифрование сообщения X на ключе θ при использовании синхропосылки S состоит в выполнении следующих шагов:

1. Установить $X_0 \leftarrow S$.
2. Для $i = 1, 2, \ldots, n$ выполнить: $Y_i \leftarrow X_i \oplus L|X_i|(F\theta(X_{i-1}))$.
3. Установить $Y \leftarrow Y_1 \| Y_2 \| \ldots \| Y_n$.
4. Возвратить Y.

ЗАДАНИЕ:

1. Изучить теоретические сведения.
2. Реализовать программные средства шифрования и дешифрования текстовых файлов при помощи стандарта шифрования СТБ 34.101.31-2011 в режимах:

✓ Простой замены

✓ Гаммирования с обратной связью

Лабораторная работа № 3
Асимметричная криптография. Криптосистема Рабина

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

3.1. Общие сведения

Криптосистема Рабина — криптографическая система с открытым ключом, безопасность которой обеспечивается сложностью поиска квадратных корней в кольце остатков по модулю составного числа. Безопасность системы, как и безопасность метода RSA, обусловлена сложностью разложения на множители больших чисел. Зашифрованное сообщение можно расшифровать 4 способами. Недостатком системы является необходимость выбора истинного сообщения из 4-х возможных.

Общий вид криптосистемы Рабина представлена на рисунке 1.

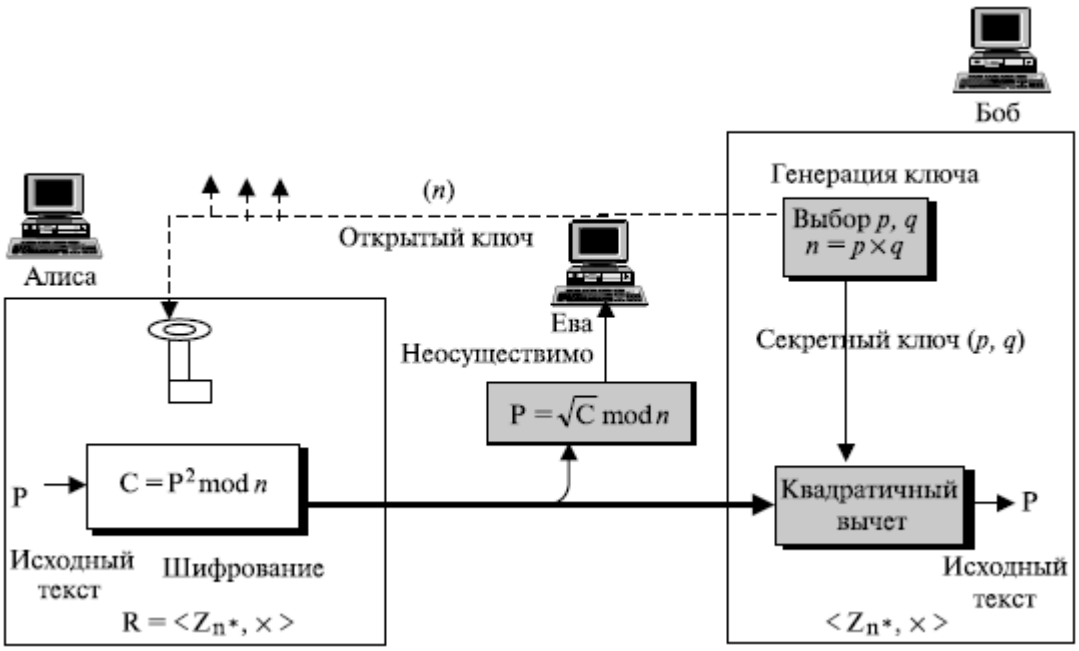


Рисунок 1 – Криптосистема Рабина общий вид

3.2 Генерация ключа

Система Рабина, как и любая асимметричная криптосистема, использует открытый и закрытый ключи. Открытый ключ используется для шифрования сообщений и может быть опубликован для всеобщего обозрения. Закрытый ключ необходим для расшифровки и должен быть известен только получателям зашифрованных сообщений.

Процесс генерации ключей следующий:

1 выбираются два случайных числа p и q с учётом требований:

1) числа должны быть большими;
2) числа должны быть простыми;
3) должны выполняться условия: p=4k+3; q=4k+3 и $p \neq q$, где kЄN

2 вычисляется число $n = p \cdot q$;

3 число n — открытый ключ; числа p и q — закрытый.

Пример. Пусть $p = 7$ и $q = 11$. Тогда $n = p \cdot q = 7 \cdot 11 = 77$. Число $n = 77$ — открытый ключ, а числа $p = 7$ и $q = 11$ — закрытый. Получатель сообщает отправителям число 77. Отправители шифруют сообщение, используя число 77, и отправляют получателю. Получатель расшифровывает сообщение с помощью чисел 7 и 11. Приведённые ключи плохи для практического использования, так как число 77 легко раскладывается на простые множители (7 и 11).

3.3 Шифрование

Исходное сообщение m (текст) шифруется с помощью открытого ключа — числа n по следующей формуле $c = m^2 \bmod n$.

Благодаря использованию умножения по модулю скорость шифрования системы Рабина больше, чем скорость шифрования по методу RSA, даже если в последнем случае выбрать небольшое значение экспоненты.

Пример (продолжение). Пусть исходным текстом является $m = 20$. Тогда зашифрованным текстом будет: $c = m^2 \bmod n = 20^2 \bmod 77 = 400 \bmod 77 = 15$.

3.4 Расшифрование

Для расшифровки сообщения необходим закрытый ключ — числа p и q. Процесс расшифровки выглядит следующим образом:

1 используя алгоритм Евклида, из уравнения $y_p * p + y_q * q = 1$ находят числа y_p и y_q;

2 далее, используя китайскую теорему об остатках, вычисляют четыре числа:

1) $r = (y_p * p * m_q + y_q * q * m_p) \; mod \; n$
2) $-r = n - r$
3) $s = (y_p * p * m_q - y_q * q * m_p) \; mod \; n$
4) $-s = n - s$

Одно из этих чисел является истинным открытым текстом m.

Пример (окончание). В результате расшифровки получаем: m ∈ { 64 , 20 , 13 , 57 }. Видим, что один из корней является исходным текстом m.

ЗАДАНИЕ:

1) Изучить теоретические сведения
2) Реализовать криптостойкое программное средство шифрования и дешифрования текстовых файлов при помощи Криптосистемы Рабина

Лабораторная работа № 4
Асимметричная криптография. Алгоритм Мак-Элиса

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

4.1 Общие сведения

McEliece — криптосистема с открытыми ключами на основе теории алгебраического кодирования, разработанная в 1978 году Робертом Мак-Элисом. Это была первая схема, использующая рандомизацию в процессе шифрования. Алгоритм основан на сложности декодирования полных линейных кодов.

Криптосистема имеет несколько преимуществ, например, над RSA. Шифрование и дешифрование проходит быстрее и с ростом длины ключа степень защиты данных растет гораздо быстрее. McEliece применим также в задачах аутентификации.

Идея, лежащая в основе данной системы, состоит в выборе корректирующего кода, исправляющего определенное число ошибок, для которого существует эффективный алгоритм декодирования. С помощью секретного ключа этот код «маскируется» под общий линейный код, для которого задача декодирования не имеет эффективного решения.

4.2 Генерация ключа

В системе Мак-Элиса параметрами системы, общими для всех абонентов, являются числа k, n, t. Для получения открытого и соответствующего секретного ключа каждому из абонентов системы следует осуществить следующие действия:

1 Выбрать порождающую матрицу G = Gkn двоичного (n,k)-линейного кода, исправляющего t ошибок, для которого известен эффективный алгоритм декодирования.

2 Случайно выбрать двоичную невырожденную матрицу S = Sk.

3 Случайно выбрать подстановочную матрицу P = Pn.

4 Вычислить произведение матриц G1 = S · G · P.

Открытым ключом является пара (G1, t), секретным – тройка (S, G, P).

4.3 Шифрование

Для того чтобы зашифровать сообщение M, предназначенное для абонента A, абоненту B следует выполнить следующие действия:

Представить M в виде двоичного вектора длины k.

Выбрать случайный бинарный вектор ошибок Z длиной n, содержащий не более t единиц.

Вычислить бинарный вектор C = M · GA + Z и направить его абоненту A.

4.4 Расшифрование

Получив сообщение C, абонент A вычисляет вектор C1 = C · P-1, с помощью которого, используя алгоритм декодирования кода с порождающей матрицей G, получает далее векторы M1 и M = M1 · S-1.

В качестве кода, исправляющего ошибки в системе Мак-Элиса, можно использовать код Гоппы. Известно, что для любого неприводимого полинома g(x) степени t над полем GF(2m) существует бинарный код Гоппы длины n = 2m и размерности k ≥ n – mt, исправляющий до t ошибок включительно, для которого имеется эффективный алгоритм декодирования.

Рекомендуемые параметры этой системы - n = 1024, t = 38, k > 644 – приводят к тому, что открытый ключ имеет размер около 219 бит, а длина сообщения увеличивается при шифровании примерно в 1,6 раза, в связи с чем данная система не получила широкого распространения.

ЗАДАНИЕ:

1. Изучить теоретические сведения
2. Реализовать криптостойкое программное средство шифрования и дешифрования текстовых файлов при помощи Криптосистемы Мак-Элиса.

Лабораторная работа № 5. Хеш-функции

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

5.1 Общие сведения

Хеш-функции получили широкое распространение в разнообразных алгоритмах быстрого поиска информации.

Однако с появлением криптографии у них появилась вторая, ничуть не меньшая, область применения.

Хеш-функцией (англ, hash — мелко измельчать и перемешивать) называется необратимое преобразование данных, обладающее следующими свойствами:

на вход алгоритма преобразования может поступать двоичный блок данных произвольной длины;

на выходе алгоритма получается двоичный блок данных фиксированной длины;

значения на выходе алгоритма распределяются по равномерному закону по всему диапазону возможных результатов;

при изменении хотя бы одного бита на входе алгоритма его выход значительно меняется: в идеальном случае инвертируется произвольная половина бит.

Основное, но не единственное, предназначение хеш-функций в криптографии — вычисление "неподделываемых" контрольных сумм документов.

Действительно, если для алгоритма хеширования выполняются требования невозможности подобрать иной документ с той же хеш-суммой и невозможности подобрать два документа с произвольной одинаковой хеш-суммой, то хеш-сумма становится уникальной характеристикой документа:

5.2 Алгоритм вычисления хеш-функции ГОСТ 34.11

В алгоритме ГОСТ 34.11 используются следующие преобразования:

X-преобразование. На вход функции X подаются две последовательности длиной 512 бит каждая, выходом функции является XOR этих последовательностей.

S-преобразование. Функция S является обычной функцией подстановки. Каждый байт из 512-битной входной последовательности заменяется соответствующим байтом из таблицы подстановок π.

P-преобразование. Функция перестановки. Для каждой пары байт из входной последовательности происходит замена одного байта другим.

L-преобразование. Представляет собой умножение 64-битного входного вектора на бинарную матрицу A размерами 64x64.

Функция сжатия.

Основным элементом любой хеш-функции является функция сжатия.

Опишем используемую в новом стандарте функцию сжатия g n в виде алгоритма. Пусть h, N и m — 512-битные последовательности. Для вычисления функции g(N, m, h) необходимо проделать следующие шаги:

1) Вычислить значение K = h ⊕ N
2) Присвоить значение K = S(K)
3) Присвоить значение K = P(K)
4) Присвоить значение K = L(K)
5) Вычислить t = E(K, m)
6) Присвоить значение t = h ⊕ t
7) Вычислить значение G = t ⊕ m
8) Вернуть G в качестве результата вычисления функции g(N, m, h)

Функция E(K, m) выполняет нижеприведенные действия:

1) Вычислить значение state = K ⊕ m
2) Для i=0 по 11 выполнить:

– присвоить значение state = S(state);
– присвоить значение state = P(state);
– присвоить значение state = L(state);
– вычислить K=KeySchedule(K, i);
– присвоить значение state = state ⊕ K.

3) Вернуть state в качестве результата.

Функция KeySchedule(K, i) отвечает за формирование временного ключа K на каждом раунде функции E(K, m) и производит следующие вычисления:

1) Присвоить значение K = K ⊕ C[i].
2) Присвоить значение K = S(K).
3) Присвоить значение K = P(K).
4) Присвоить значение K = L(K).
5) Вернуть K в качестве результата функции.

Вычисление хеш-функции.

Для любого входного сообщения M:

1) Присвоить начальные значения внутренних переменных:

Для хеш-функции с длиной выхода 512 бит: $h=iv=0x00^{64}$

Для хеш-функции с длиной выхода 256 бит: $h=iv=0x01^{64}$

$N = 0^{512}$

$\Sigma = 0^{512}$

2) Проверить следующее условие:

длина сообщения M<512. Если условие выполняется перейти к пункту 3). В противном случае выполнить последовательность вычислений:

m — последние 512 бит сообщения M

h = g(N, m, h)

$N = (N + 512) \bmod 2^{512}$

$\Sigma = (\Sigma + m) \bmod 2^{512}$

Обрезать M, убрав последние 512 бит.

Перейти к шагу 2 .

3) Произвести дополнение сообщения M до длины в 512 бит по следующему правилу:

$m = 0^{511-|M|}||1||M$, где |M| — длина сообщения M в битах.

Вычислить h = g(N, m, h).

Вычислить $N = (N + |M|) \bmod 2^{512}$

Вычислить $\Sigma = (\Sigma + m) \bmod 2^{512}$

Вычислить h = g(0, h, N)

Вычислить h = g(0, h, Σ)

Для хеш-функции с длиной выхода в 512 бит возвращаем h в качестве результата. Для функции с длиной выхода 256 бит возвращаем MSB 256 (h).

5.3 Алгоритм вычисления хеш-функции SHA-1

Алгоритм SHA-1 (Secure Hash Algorithm) предложен Институтом Стандартизации США NIST как стандарт хеширования в гражданской криптографии. Этот алгоритм был призван дать еще больший запас прочности к криптоатакам.

SHA-1 реализует хеш-функцию, построенную на идее функции сжатия. Входами функции сжатия являются блок сообщения длиной 512 бит и выход предыдущего блока сообщения.

Выход представляет собой значение всех хеш-блоков до этого момента. Иными словами хеш блока Mi равен hi = f (Mi , hi−1). Хеш-значением всего сообщения является выход последнего блока.

Алгоритм получает на входе сообщение максимальной длины 2^{64} бит и создает в качестве выхода *дайджест сообщения* длиной 160 бит.

Шаг 1: добавление недостающих битов

Сообщение добавляется таким образом, чтобы его длина была кратна 448 по модулю 512 (длина ≡ 448 mod 512).

Добавление осуществляется всегда, даже если сообщение уже имеет нужную длину.

Таким образом, число добавляемых битов находится в диапазоне от 1 до 512.

Добавление состоит из единицы, за которой следует необходимое количество нулей.

Шаг 2: добавление длины

К сообщению добавляется блок из 64 битов. Этот блок трактуется как беззнаковое 64-битное целое и содержит длину исходного сообщения до добавления.

Результатом первых двух шагов является сообщение, длина которого кратна 512 битам.

Расширенное сообщение может быть представлено как последовательность 512-битных блоков $Y_0, Y_1, \ldots, Y_{L-1}$, так что общая длина расширенного сообщения есть L * 512 бит.

Таким образом, результат кратен шестнадцати 32-битным словам.

Шаг 3: инициализация SHA-1 буфера

Используется 160-битный буфер для хранения промежуточных и окончательных результатов *хеш-функции*.

Буфер может быть представлен как пять 32-битных регистров A, B, C, D и E.

Эти регистры инициализируются следующими шестнадцатеричными числами:

A = 67452301

B = EFCDAB89

C = 98BADCFE

D = 10325476

E = C3D2E1F0

Шаг 4: обработка сообщения в 512-битных (16-словных) блоках

Основой алгоритма является модуль, состоящий из 80 циклических обработок, обозначенный как H_{SHA}.

Все 80 циклических обработок имеют одинаковую структуру.

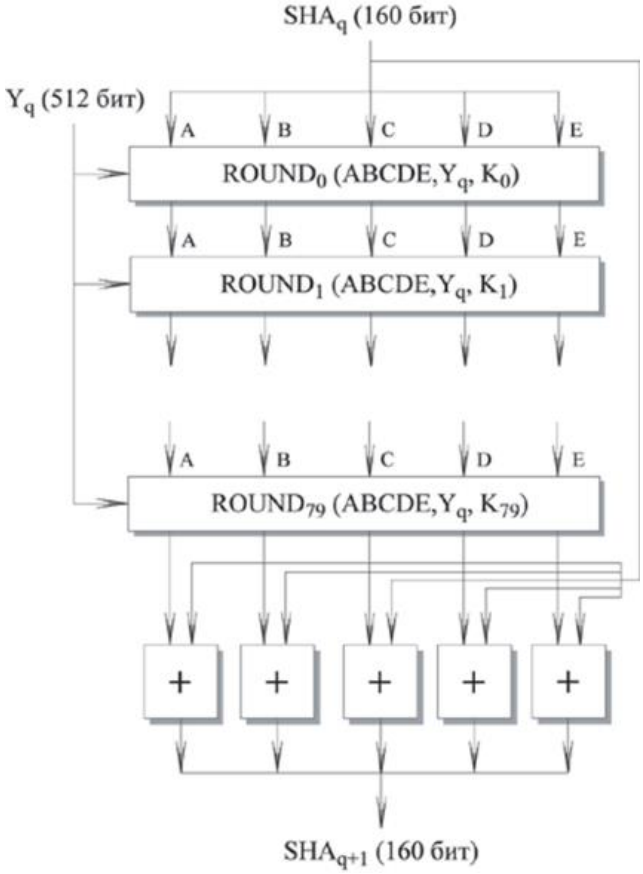


Рисунок 5,1 Обработка очередного 512-битного блока

Каждый цикл получает на входе текущий 512-битный обрабатываемый блок Y_q и 160-битное значение буфера *ABCDE*, и изменяет содержимое этого буфера.

В каждом цикле используется дополнительная константа K_t, которая принимает только четыре различных значения:

$0 <= t <= 19$ $\mathbf{K_t = 5A827999}$ (целая часть числа $[2^{30} \times 2^{1/2}]$)

$20 <= t <= 39$ $\mathbf{K_t = 6ED9EBA1}$ (целая часть числа $[2^{30} \times 3^{1/2}]$)

$40<=t<=59$ $\mathbf{K_t=8F1BBCDC}$ (целая часть числа $[2^{30} x5^{1/2}]$)

$60<=t<=79$ $\mathbf{K_t=CA62C1D6}$ (целая часть числа $[2^{30} x10^{1/2}]$)

Для получения SHA_{q+1} выход 80-го цикла складывается со значением SHA_q.

Сложение по модулю 2^{32} выполняется независимо для каждого из пяти слов в буфере с каждым из соответствующих слов в SHA_q.

Шаг 5: выход

После обработки всех 512-битных блоков выходом L-ой стадии является 160-битный *дайджест сообщения*.

Рассмотрим более детально логику в каждом из 80 циклов обработки одного 512-битного блока.

Каждый цикл можно представить в виде:

$$\mathbf{A,B,C,D,E(CLS_5(A)+f_t(B, C, D)+E+W_t+K_t),A,CLS_{30}(B),C,D}$$

где

A, B, C, D, E - пять слов из буфера.

t - номер цикла, $0 <= t <= 79$.

f_t - элементарная *логическая функция*.

CLS_s - циклический левый сдвиг 32-битного аргумента на s битов.

W_t - 32-битное слово, полученное из текущего входного 512-битного блока.

K_t - дополнительная константа.

$+$ - сложение по модулю 2^{32}.

Логика выполнения отдельного цикла SHA-1

Каждая элементарная функция получает на входе три 32-битных слова и создает на выходе одно 32-битное слово.

Элементарная функция выполняет набор побитных логических операций, т.е. n-ый бит выхода является функцией от n-ых битов трех входов. Функции следующие:

Номер цикла	*ft (B, C, D)*
$(0 <= t <= 19)$	$(B \wedge C) \vee (\neg B \wedge D)$
$(20 <= t <= 39)$	$B \oplus C \oplus D$
$(40 <= t <= 59)$	$(B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$
$(60 <= t <= 79)$	$B \oplus C \oplus D$

На самом деле используются только три различные функции.

Для $0 <= t <= 19$ функция является условной:

$$\text{if B then C else D.}$$

Для $20 <= t <= 39$ и $60 <= t <= 79$ функция создает бит *четности*.

Для $40 <= t <= 59$ функция является истинной, если два или три аргумента истинны.

Получение входных значений каждого цикла из очередного блока/

Двухбитные слова W_t получаются из очередного 512-битного блока сообщения следующим образом.

Первые 16 значений W_t берутся непосредственно из 16 слов текущего блока. Оставшиеся значения определяются следующим образом:

$$W_t = W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3}$$

В первых 16 циклах вход состоит из 32-битного слова данного блока. Для оставшихся 64 циклов вход состоит из *XOR* нескольких слов из блока сообщения.

Алгоритм *SHA-1* можно суммировать следующим образом:

$$\mathbf{SHA_0 = IV}$$

$$\mathbf{SHA_{q+1} = \Sigma 32(SHA_q, ABCDE_q),}$$

Где:

IV - начальное значение буфера *ABCDE*;

ABCDE$_{\mathbf{q}}$ - результат обработки q-того блока сообщения;

L - число блоков в сообщении, включая поля добавления и длины;

Σ32 - сумма по модулю 2^{32}, выполняемая отдельно для каждого слова буфера;

SHA - значение *дайджеста сообщения*.

ЗАДАНИЕ:

1. Изучить теоретические сведения
2. Реализовать самостоятельно (без использования готовых библиотек и функций) программное средство контроля целостности сообщений с помощью вычисления хеш-функции и алгоритма ГОСТ 34.11 и SHA 1.