

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО
ОБРАЗОВАНИЯ
«ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ - УЧЕБНО-НАУЧНО-
ПРОИЗВОДСТВЕННЫЙ КОМПЛЕКС»

УЧЕБНО-НАУЧНО-ИССЛЕДОВАТЕЛЬСКИЙ ИНСТИТУТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

В.Т. Еременко, С.А. Шпичак, П.Н. Рязанцев

КРИПТОГРАФИЧЕСКИЕ МЕТОДЫ ЗАЩИТЫ ИНФОРМАЦИИ

Рекомендовано ФГБОУ ВПО «Госуниверситет - УНПК»
для использования в учебном процессе в качестве учебного пособия
для высшего профессионального образования

Орел 2015

УДК 621.391
ББК 32.81я73
Е70

Рецензенты:

доктор технических наук, профессор кафедры «Информационные системы» Федерального государственного бюджетного образовательного учреждения высшего профессионального образования «Государственный университет — Учебно-научно-производственный комплекс»

В.И. Раков,

кафедра «Системы информационной безопасности»
Федерального государственного бюджетного образовательного учреждения высшего профессионального образования
«Брянский государственный технический университет»

М.Ю Рытов

Еременко В.Т.

Е70 Криптографические методы защиты информации: учебное пособие для высшего профессионального образования / В.Т. Еременко, С.А. Шпичак, П.Н. Рязанцев – Орел: ФГБОУ ВПО «Госуниверситет – УНПК», 2015. – 125 с.

В учебном пособии рассматриваются исторические аспекты развития криптографии, общие вопросы теории криптографии, стойкости криптографических систем, принципы построения криптоалгоритмов, криптографических хеш-функций и организации сетей засекреченной связи. Большое внимание уделено рассмотрению основ криптоанализа и перспективных направлений в развитии криптографии.

Учебное пособие предназначено для студентов специальностей предназначен для бакалавров, обучающихся по направлению подготовки 090900.62 «Информационная безопасность» и 210700.62 «Инфокоммуникационные технологии и систем связи», из изучающих дисциплины Б.3.А.15 «Криптографические методы защиты информации», Б.3.В.2.1 «Основы криптографии» и смежных с ними, слушателей курсов повышения квалификации по проблемам защиты информации, а также может быть полезно специалистам, занимающимся решением задач обеспечения информационной безопасности.

УДК 621.391

ББК 32.81я73

© ФГБОУ ВПО «Госуниверситет-УНПК», 2015

СОДЕРЖАНИЕ

Введение	5
1. Введение в криптографию	6
1.1. Краткая история развития криптографических методов.....	6
1.2. Основные понятия криптографии	19
Контрольные вопросы	30
2. Стойкость криптографических систем	31
2.1. Модели шифров и открытых текстов	31
2.2. Криптографическая стойкость шифров.....	40
2.3. Имитостойкость и помехоустойчивость шифров.....	48
Контрольные вопросы	55
3. Принципы построения симметричных криптографических алгоритмов.....	56
3.1. Виды симметричных шифров. Особенности программной и аппаратной реализации.	56
3.2. Принципы построения блочных шифров.....	57
3.3. Современные блочные криптоалгоритмы.....	61
3.4. Принципы построения поточных шифров	74
3.5. Современные поточные криптоалгоритмы.....	82
3.6. Режимы использования шифров.	84
Контрольные вопросы	89
4. Принципы построения асимметричных криптографических алгоритмов.....	90
4.1. Математические основы асимметричной криптографии.	90
4.2. Примеры современных асимметричных шифров.	107
Контрольные вопросы	122
5. Криптографические хэш-функции и электронно-цифровая подпись	123
5.1. Криптографические хэш-функции.....	123
5.2. Электронно-цифровая подпись	133
Контрольные вопросы	147
6. Организация сетей засекреченной связи	148
6.1. Протоколы распределения ключей.	148
6.2. Особенности использования вычислительной техники в криптографии.	163
Контрольные вопросы	168
7. Криптоанализ и перспективные направления в криптографии	169

7.1. Основные методы криптоанализа	169
7.2. Перспективные направления в криптографии.....	180
Контрольные вопросы	198
ЛИТЕРАТУРА.....	199
Приложение А.	201
Приложение Б.....	204

ВВЕДЕНИЕ

В настоящее время возрастает актуальность защиты информации во всех сферах человеческой деятельности: на государственной службе, в бизнесе, науке и даже в личной жизни. Исходя из анализа свойств информации, становится очевидным, что при обеспечении информационной безопасности объекта, прежде всего, следует надежно защищать носители информации от непреднамеренной и несанкционированной деятельности людей, связанной с информацией, хранимой на объекте защиты в условиях бесконтрольного доступа. Среди мер по защите информации важное значение придается криптографической защите информации, основанной на использовании математических приемов и методов.

Учебное пособие состоит из введения, семи глав и списка литературы.

Его целью является изучение исторических аспектов развития криптографии, общие вопросы теории криптографии, стойкости криптографических систем, принципы построения криптоалгоритмов, криптографических хеш-функций и организации сетей засекреченной связи.

1. ВВЕДЕНИЕ В КРИПТОГРАФИЮ

1.1. Краткая история развития криптографических методов

Исторически сложились и дошли до наших дней три подхода к защите информации:

1. *Физическая защита носителя информации.*

Данный подход предполагает использование комплекса различных средств защиты, а также нестандартной передачи и хранения информации (голуби, курьеры, сейф, проводная и кабельная связь, «радиовыстрел», СИЧ-передача и др.). Одновременно разрабатывались и методы уничтожения информации при угрозе ее захвата, а также методы обнаружения тайной перлюстрации. Идеалом этого метода является создание защищенного канала передачи информации, к которому противник физически не в состоянии получить доступ.

2. *Стеганография.*

Подход предполагает применение комплекса средств, при помощи которых скрывается сам факт передачи информации (симпатические чернила, «микроточка» и т.д.). Идеал этого подхода – создание скрытого канала связи. Здесь заслуживает внимания предложение греческого полководца Энея (Спарта, V – IV вв. до н. э.). Наряду с известным «диском Энея», который представлял собой шифр замены букв на расстояние между узелками на послании – нити, он предложил и такой способ. На «невинном» тексте, который обычно писался на деревянных дощечках, покрытых воском, мелкими, малозаметными точками отмечались буквы, несущие секретное сообщение.

Эту идею уже в XVI веке повторил английский государственный деятель Ф. Бэкон. Вместо точек он предложил двухшрифтовый шифр, где использовались два малоразличимых шрифта. Заслуга Бэкона заключается также и в том, что он первый предложил кодировать буквы латинского алфавита двоичным кодом. Эта идея широко используется и в наши дни.

Дальнейшее развитие идей Энея привело к появлению так называемых книжных шифров. Немцы в начале XX века

использовали в агентурной переписке эти шифры, в которых вместо невидимых точек, проколов в тексте прикрытия значимые буквы выделялись невидимой точкой, поставленной с использованием симпатических чернил. Недостаток книжных шифров – необходимость постоянно иметь с собой заранее оговоренную книгу. Поэтому появились его разновидности: стихотворный шифр, шифр по слову-лозунгу и т. д. Здесь стеганография объединяется с криптографией. Ключом шифра является книга, стихотворение, слово-лозунг и т.д. Сам шифрованный текст имеет невинный вид.

3. Криптография.

Этот подход предполагает использование шифров. Идеал – использование открытого канала связи. При этом противник, зная о факте передачи и имея физический доступ к информации, не может понять ее смысл, если не владеет секретным ключом [1].

В настоящее время широко используется термин «криптология». Данная наука включает в себя защиту (криптография) и нападение (криптоанализ, дешифрование). Наряду с шифрами (с XIV – XV веков) широко использовались коды (номенклаторы). В кодах единицами текста, подлежащими замене, являлись не отдельные буквы, а слова, фразы и целые предложения.

В настоящее время все эти подходы нередко используются в комплексе, особенно при защите наиболее важных секретов.

Перейдем к краткому изложению исторической эволюции шифров.

В Древнем мире появились два основных вида шифров:

- шифры замены;
- шифры перестановки.

Классическим примером шифра замены является **шифр Цезаря** (около I века до нашей эры). Этот шифр задается подстановкой следующего вида (применительно к латинскому алфавиту):



Верхний алфавит – это алфавит открытого текста. Нижний алфавит, представляющий собой циклический сдвиг верхнего алфавита влево на три шага, есть алфавит шифрованного текста.

Буква *A* открытого текста заменялась на букву *D* в шифрованном, буква *B* – на букву *E* и т.д. Все буквы открытого текста заменялись по подобной подстановке. В связи с этим этот шифр в последующем стал называться шифром простой (одноалфавитной) замены.

В наши дни под шифром Цезаря понимаются все шифры, в которых нижняя строка является циклическим сдвигом верхней на произвольное число шагов. Однако поскольку этот сдвиг не меняется в процессе шифрования, то шифр остается шифром простой (одноалфавитной) замены.

Дальнейшее развитие шифра Цезаря шло в разных направлениях. Прежде всего алфавит шифрованного текста стал не сдвигом алфавита открытого текста, а произвольным набором букв алфавита (перемешивание алфавита).

Шифр усложнился, в нем появилось $z! = 26! \approx O(10^{26})$ ключей. Однако он оставался шифром простой замены. Появилось и практическое неудобство. Перемешанный алфавит нижней строки невозможно запомнить, поэтому и отправитель сообщения, и его получатель должны были иметь постоянно при себе ключ (алфавит нижней строки). Здесь есть риск: тайная перлюстрация ключа или просто его потеря.

Возникла идея построения «псевдослучайного» алфавита шифрованного текста. В качестве ключа брался легко запоминающийся пароль, например, слово «THE TABLE». Это слово без повторов букв выписывалось в нижней строке:

THEABL.....

Затем к нему добавлялись пропущенные буквы алфавита:

THEABLCDF.....Z.

Но это опять-таки был шифр простой замены.

Известно несколько разновидностей шифров простой замены:

- *атбаиш* – шифр некоторых фрагментов библейских текстов. Правило зашифрования: i -я буква алфавита ($i = \overline{1, n}$) заменялась буквой с номером $n - i + 1$, где n – число букв алфавита. Это один из простейших шифров простой замены;

- *табличка и диск Энея* – приспособления с нанесенным алфавитом (ключевой элемент), на которые наматывалась нить с

узелками напротив соответствующих букв. Шифр-величины – расстояния между узелками;

- *квадрат Полибия* (Древняя Греция) – таблица с буквами алфавита, заменяемыми номерами строки и столбца (i, j). Один из простейших шифров простой замены.

Арабы во времена Средневековья нашли эффективный метод дешифрования шифров простой замены. Они использовали частотные характеристики открытых текстов (букв, биграмм), которые отражались в зашифрованном. Первое упоминание метода – в 1412 г. в разделе 14-томной энциклопедии «Шауба аль-Аща», автор – *Шехаб аль-Кашиканди*. Они также предложили метод опробования вероятных слов и словосочетаний в открытом тексте.

Дальнейшее развитие шифров замены пошло по пути появления многоалфавитности зашифрованного текста, а также введения так называемого биграммного шифрования. Впервые эту идею (биграммности) предложил в 1563 году итальянец *Порта*. Открытый текст разбивался на пары букв (при необходимости к нему добавлялась произвольная буква алфавита). Шифром являлась таблица размером $Z \cdot Z$ ($26 \cdot 26 = 676$ клеток). В каждую клетку помещался один из 676 заранее придуманных экзотических знаков (геометрические фигуры, образы и т.д.). Первая буква пары определяла строку таблицы, вторая – ее столбец. На их пересечении и находился зашифрованный знак. Ввиду громоздкости и сложности практического применения этот шифр не нашел распространения. Весомый вклад в биграммное шифрование внес англичанин *Уитстон*. Он использовал в 1854 году в своем шифре *PlayFair* (честная игра) простой квадрат размером $5 \cdot 5 = 25$ (редуцированный латинский алфавит, в котором буква *J* отождествлялась с буквой *I*). В таблице в произвольном (ключевом) порядке располагались 25 букв алфавита. Биграмма, лежащая в одной строке, заменялась биграммой из букв, находящихся циклически справа от букв открытого текста. Биграмма, лежащая в одном столбце, также шифровалась по правилу циклического сдвига столбца сверху вниз. Биграммы, расположенные в разных строках и столбцах, заменялись на буквы, стоящие в противоположных углах прямоугольника, построенного на исходной биграмме. Это было существенным усилением шифров

побуквенной замены. Тем не менее этот шифр является шифром простой замены, но на уровне биграмм.

Другое развитие идеи замены заключалось в появлении так называемых многоалфавитных шифров, в которых алфавит шифрованного текста менялся в процессе шифрования. И первым здесь был итальянец *Альберти* (1466 г.). Конструкция его шифра была достаточно простой. Два диска, насаживались на одну ось. Внешний диск – неподвижный, на нем нанесен естественный алфавит (алфавит открытого текста). Внутренний диск – подвижный, на нем нанесен алфавит в заранее оговоренном порядке (алфавит шифрованного текста). Перед началом шифрования внутренний диск ставился в заранее оговоренное угловое положение, что определяло исходную замену. Затем, после зашифрования заранее оговоренного количества букв, внутренний диск смещался на некоторое оговоренное расстояние (например на единицу по часовой стрелке). Таким образом, алфавит шифрованного текста менялся (по отношению к неподвижному алфавиту внешнего диска).

Следующий шаг в развитие идеи многоалфавитности шифров внес в 1518 году аббат *Тритемий* (Германия). Он предложил использовать шифр, вошедший в историю как *таблица Тритемия*. Строится квадрат размером $Z \cdot Z$ ($26 \cdot 26$). Первая строка квадрата – естественный алфавит, вторая строка – сдвиг первой (по циклу) на одну букву влево. Последующие строки также являются циклическими сдвигами предыдущей строки. При шифровании первой буквы открытого текста используется первая строка, при шифровании второй – вторая и т.д. При зашифровании $(Z+1)$ -й

(27-й) буквы производится возврат к первой строке. Таким образом, алгоритм шифрования является периодическим с периодом Z . Алгоритмы шифрования и расшифрования различны.

Английский адмирал *Бофор* предложил модификацию шифра Тритемия, в которой алгоритмы зашифрования и расшифрования совпадают. Первая строка таблицы представляет собой обратный (инверсный) порядок расположения букв алфавита. Строки сдвигаются не влево, а вправо. При использовании такой таблицы расшифрование сводится к повторному шифрованию полученного шифрованного текста. Так появились так называемые

обратимые шифры. Они значительно проще в практической реализации. Одним из недостатков шифра Тритемия является жестко фиксированный порядок использования строк. Итальянцы *Порта* и *Беллазо* предложили использовать случайный порядок использования строк. Для легкого запоминания этого порядка вводится ключ – лозунг (слово). Строки таблицы используются в соответствии с этим словом (при его периодическом продолжении на всю длину шифруемого текста). Этот алфавит шифрования обобщил и широко опубликовал французский дипломат *Виженер*. В настоящее время шифр известен именно под его именем.

Француз *де Виари* выдвинул идею формализации процесса шифрования по шифру Виженера. По сути дела, он предложил математическое представление процесса шифрования. В современном изложении эта идея сводится к следующему.

Буквы алфавита и лозунга (ключа) заменяются на их номера в естественном алфавите. Периодическая последовательность букв лозунга в последующем получила название «гамма шифра». Некоторые исследователи связывают это название с именем французского дипломата *Гамбетты*, который одним из первых использовал такую формализацию.

Итак, пусть шифруется открытый текст:

$$p = p_1 p_2 \dots p_n, p_i \in A = \{0, 1, 2, \dots, 25\}$$

(применительно к латинскому алфавиту).

Гамма шифра имеет вид:

$$k = k_1 k_2 \dots k_m, k_i \in A$$

Тогда шифрование по шифру Виженера имеет следующее представление. При использовании таблицы Тритемия:

$$C = E(p) = (p + k_i) \bmod 26,$$

где p – i -я буква шифрованного текста.

При расшифровании имеем:

$$p = D(C) = (C - k_i) \bmod 26.$$

Таким образом, при зашифровании и расшифровании применяются разные операции (сложение и вычитание).

При использовании таблицы Бофора получим:

$$C = E(p) = (k - p) \bmod 26;$$

$$p = D(C) = (k - C) \bmod 26.$$

Таким образом, при шифровании, и расшифровании используется одна и та же операция (вычитание). Слабость указанных шифров заключается в короткой периодичности гаммы шифра.

Офицер прусской армии *Казисский* (XIX в.), а затем и известный криптограф США *Фридман* продемонстрировали эффективные способы дешифрования шифров короткопериодического гаммирования. Поэтому в дальнейшем возникла проблема создания гаммы большого периода, образуемой из короткого ключа-лозунга. Эта идея и реализуется в современных шифраторах. Значительный вклад в развитие многоалфавитных шифров внес в начале XIX века американец *Джефферсон* (будущий президент США). Шифратор Джефферсона представлял собой 25 – 36 дисков одинакового размера с перемешанными алфавитами на боковой поверхности. Множество ключей – расстановка букв на дисках, расстановка дисков на оси, выбор набора дисков из запаса.

Предложенный способ (прибор) дискового шифрования получил широкое распространение в XX веке (дисковые шифраторы, «Энигма», «Хагелин»).

Еще одно направление развития шифров замены – появление шифров так называемой многозначной замены (не путать с шифрами многоалфавитной замены). В этих шифрах алфавит шифрованного текста был значительно больше алфавита открытого текста. В нем использовались геометрические фигуры, числа и т.д. Одна и та же буква получала несколько шифр-обозначений, которые не совпадали с обозначениями других букв. При шифровании эти шифр-обозначения выбирались произвольно из соответствующих множеств. При этом «сглаживались» частотные характеристики открытого текста.

В дальнейшем развитии многозначные шифры породили шифры пропорциональной замены, в которых количество обозначений для буквы открытого текста было пропорционально частоте появления этой буквы. В шифртексте в этом случае все шифр-обозначения появлялись примерно с одинаковой частотой. Эти шифры, возникшие в Средние века, дошли до XX века.

Примеры шифров многозначной замены:

- *омофоны* (Италия, XIV в.). Гласным буквам соответствует несколько различных шифр-величин. Автор – Чикко Симонетти;
- *шифры пропорциональной замены* (Италия, XV в.). Каждой букве ставится в соответствие несколько шифр-величин в зависимости от частоты, с которой эта буква встречается в тексте. Автор – Габриэль де Лавинда. (Аналог – Миланский ключ, 1469 г.);
- *шифр Ардженти* (Италия, XVII в.). Внедрено смещение алфавита сдвигом от ключевого слова, введены «пустышки» для изменения частотных характеристик текста, *многозначная замена*, буквенный код.

Наряду с развитием шифров побуквенного шифрования (поточных шифров) появились так называемые блочные шифры. Предвестниками их стали биграммные шифры (Порта, Плейфер). Затем блоки (группы одновременно шифруемых букв) заметно увеличились. Примером таких шифров – шифры перестановки. Эти шифры, по сути дела являются, шифрами поточной замены, но в значительно более мощном алфавите (весь блок букв естественного алфавита рассматривается как буква более мощного алфавита). Шифр перестановки в этом случае становится шифром простой замены в расширенном алфавите.

Шифры перестановки также получили заметное развитие и дошли до наших дней. Классический пример такого шифра – *шифр Сцитала* (изобретение спартанцев, примерно IV век до нашей эры). Итальянец *Кардано* уже в середине XVI столетия предложил новый шифр перестановки – «решетку Кардано». В основе лежало решение кубического уравнения вида $x^3 + px + q = 0$, предложенного другим итальянским математиком Тарталья:

$$x = \sqrt[3]{-\frac{q}{2} + \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}} + \sqrt[3]{-\frac{q}{2} - \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}}$$

квадратная или прямоугольная решетка, своими вырезами однократно покрывающая всю площадь квадрата после четырех поворотов.

Далее появились шифры перестановки по лозунгу. Здесь буквы шифра переставлялись по ключу-лозунгу, по которому составлялся так называемый номерной ряд. Пусть, например, лозунг – слово «THE TABLE». В нем восемь букв. Пронумеруем буквы лозунга в соответствии с их алфавитным порядком. Получим

номерной ряд: 75381264. Открытый текст разбивается на блоки из восьми букв. Первая буква блока встает на 5-е место, вторая –

на 6-е, третья на 3-е, четвертая – на 8-е и т.д. При необходимости открытый текст добавляется произвольным набором букв до числа, кратного 8. К XX веку эти шифры усложнились. Появились так называемые маршрутные перестановки, шифры вертикальной перестановки и т.д.

На развитие криптографии оказывает большое влияние научно-технический прогресс. Изобретение книгопечатания породило шифры, основанные на ключе-книге (книжный шифр, книжная гамма, книжный код и т.д.). Появление телеграфа, радио, телефона дало мощный стимул к разработке новых шифров. Особое влияние на развитие криптографии оказали ЭВМ, распространившиеся в середине XX века. Первая в мире современная ЭВМ («Колосс», Великобритания) была разработана в процессе проведения английской операции «Ультра» – дешифрования основного шифратора фашистской Германии («Энигма»). Появились новые виды шифраторов:

- *телефонные шифраторы (скремблеры)* – обработка аналогового сигнала (инверсия, смещение, деление диапазона, шумовые маскировки, временные перестановки), комбинация криптографических и стеганографических методов. После появления цифровой телефонии – классические криптографические (дискретные) методы;

- *электронные шифраторы, программные реализации.* Аппаратные реализации на основе микросхем СБИС. Программные реализации – несколько позже. Самая известная и заметная программная реализация – *PGP* Фила Циммермана.

Развитие математики привело к становлению криптографии как точной математической науки. *Клод Шеннон* в 1944 году в своем труде «Теория секретных систем связи» формализовал задачи синтеза и анализа шифров. Там же были введены понятия принципов рассеивания и перемешивания (конфузия, диффузия), теоретической и практической стойкости, совершенного шифра. Введено рассмотрение языка как вероятностный процесс. Предложена концепция избыточной информации естественного языка. (The, of, and, to, a, in, that, it, is, i – более 25 % слов любого английского текста в любой криптограмме.)

Определена теоретическая мера стойкости – энтропийная характеристика (неопределенность шифра по открытому сообщению). Энтропия показывает, насколько «близка» средняя криптограмма к единственному «решению». Также введено понятие расстояния единственности – минимального размера криптограммы, необходимого для однозначного расшифрования (для совершенного шифра – бесконечность). Разделены понятия теоретической и практической стойкости.

Задолго до Шеннона советский академик *А.А. Марков* предложил марковские модели (зависимость появления букв в тексте от предыдущих букв), в 60-е годы *А.Н. Колмогоров* – различие удельной энтропии на букву для текстов разного смыслового содержания.

В середине XX века наметилась тенденция опережающего развития методов защиты (шифрования и расшифрования) по отношению к методам средств нападения (дешифрования). В связи с этим стали активно разрабатываться новые подходы к получению информации, способствующей облегчению дешифрования (помимо обычного перехвата сообщений).

В XX веке криптографическая деятельность вышла из-под контроля государства. Появились частные фирмы – производители шифртехники на продажу и их специфические пользователи (корпорации, банки и т.д.). Наряду с лояльными пользователями криптографические методы начали применять и криминальные структуры. Государство не может отказаться от контроля за их деятельностью. Появилась проблема вида: «государственная и негосударственная» криптография.

Был разработан ряд криптографических стандартов. Первым шагом явилась публикация DES – стандарта на симметричный блочный комбинированный криптоалгоритм. В XXI веке выходит стандарт AES. Также публикуются стандарты цифровой подписи DSS и распределения ключей X.509. Ряд таких стандартов, как: EES – стандарт с депонированием ключей, микросхема Clipper, плата Fortezza, и явились попыткой правительства США (администрация Клинтона) взять под контроль ключевую систему в криптосистемах связи. В нашей стране также вышли государственные стандарты на блочный алгоритм шифрования, криптографическую хеш-функцию и схему электронно-цифровой подписи.

В XIX веке криптографические методы начали широко применяться в новых областях исследования информационных источников. Например, методы криптографии помогли исследователям так называемых «мертвых языков». Утраченные письменность и устный язык восстанавливаются, в частности, и методами криптоанализа.

Шифры широко используются в литературных произведениях, кинофильмах (особенно детективного содержания). Они вносят интригу в повествование, заинтересовывают читателя (зрителя). С точки зрения профессионального криптоанализа эти «криптографические вставки» являются дилетантскими, но несут положительную нагрузку – заинтересовывают потребителя информацией в новом для них средстве защиты информации.

В XX веке сформировались основные требования к шифрам, включающие в себя следующую триаду:

1) *Криптографическая стойкость* – способность шифра противостоять попыткам его дешифрования. Это основное требование.

2) *Имитостойкость* – способность шифра противостоять попыткам «дезинформации под шифром» (имитостойкости). Цель имитации – внедрение противником дезинформации под шифром. В истории есть немало примеров такой тактики нападения.

3) *Помехоустойчивость* – способность шифра противостоять действию помех в канале связи. Эти помехи могут носить и искусственный характер с целью срыва зашифрованной связи. Есть примеры шифров, которые не получили широкого распространения только потому, что они при наличии даже ограниченных помех (искажений) не позволяли применять их в сетях засекреченной связи.

Примером могут служить многоалфавитные поточные шифры. При пропуске хотя бы одной буквы в зашифрованном тексте весь последующий текст на приеме не расшифровывается. Разумеется, это только основные требования, существуют и другие (затраты времени на шифрование – расшифровывание, т.е. быстроедействие, особые требования к каналам связи, затраты на засекречивание и т.д.).

В 1976-1978 г. г. революционные идеи У. Диффи и М. Хеллмана привели к появлению асимметричной криптографии. В

настоящее время криптосистемы делятся на симметричные (классические) и ассиметричные (с открытым ключом) [1]. На рис. 1. приведена схема передачи шифрованной информации при помощи симметричного шифра.

Источник сообщений – владелец информации, осуществляющий преобразование информации (*шифрование*) и передачу *получателю* шифрованных сообщений с целью защиты информации от *противника*, который может наблюдать передаваемые по каналу связи сообщения. Исходная информация называется *открытым текстом* (x), шифрованная информация – *шифртекстом* (y). Стороны, обменивающиеся информацией, по усмотрению выбирают *шифр* и *ключи* (k). Секретные ключи предварительно рассылаются по защищенному каналу связи. Симметричные системы используют один *секретный ключ* для зашифрования и расшифрования.

Законный получатель информации осуществляет *расшифрование* полученных сообщений.

Противник – субъект, не имеющий права ознакомления с содержанием передаваемой информации. Противник пытается завладеть защищаемой информацией, для чего предпринимает действия, называемые атаками. *Пассивные атаки* связаны с анализом трафика, перехватом и *дешифрованием* сообщений.

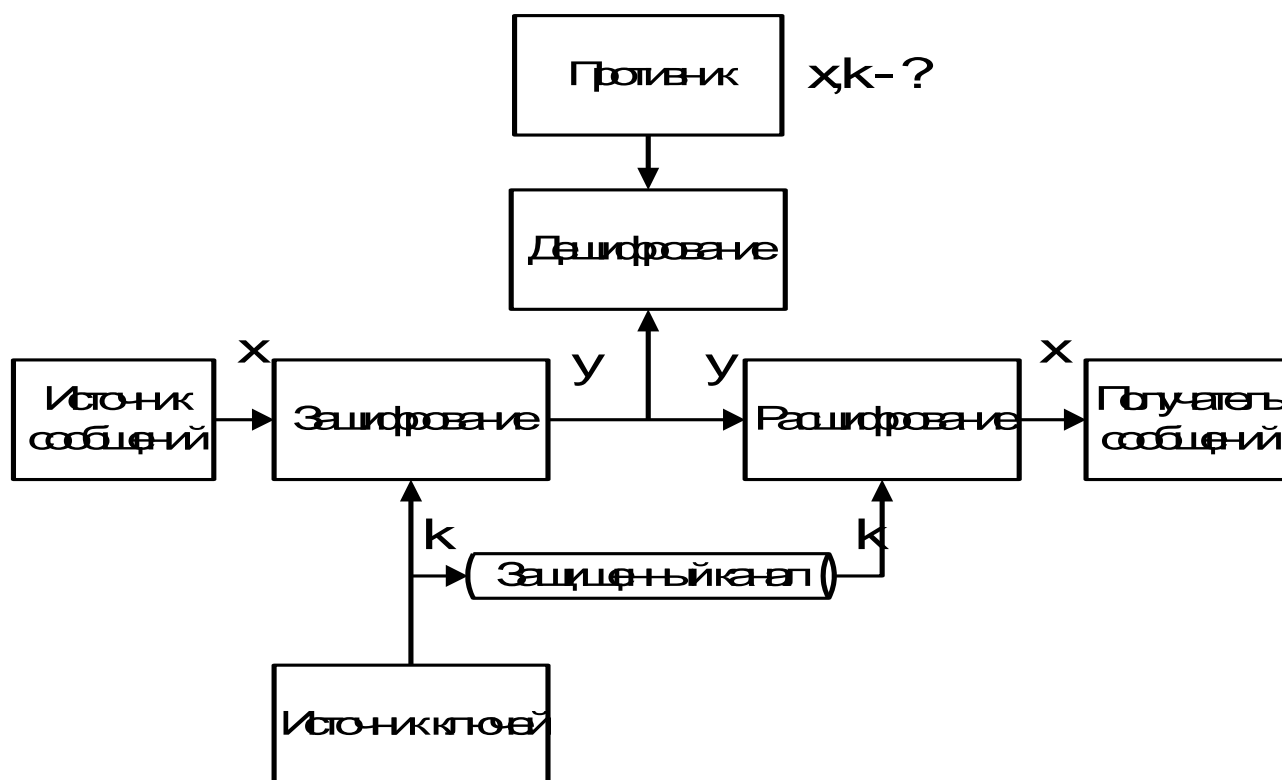


Рис. 1. Схема симметричного шифрования.

Каждое шифрующее преобразование однозначно определяется ключом и описывается *криптографическим алгоритмом*. Идея асимметричного шифрования состоит в том, что при расшифровании используется алгоритм, который может отличаться от алгоритма, применяемого для зашифрования. Соответственно могут различаться и ключи, которые используются для зашифрования и расшифрования.

Схема передачи информации при асимметричном шифровании приведена на рис.2. Ассиметричные системы используют два разных ключа: *открытый ключ* (k_o) для зашифрования и *личный (секретный) ключ* (k_p) для расшифрования.

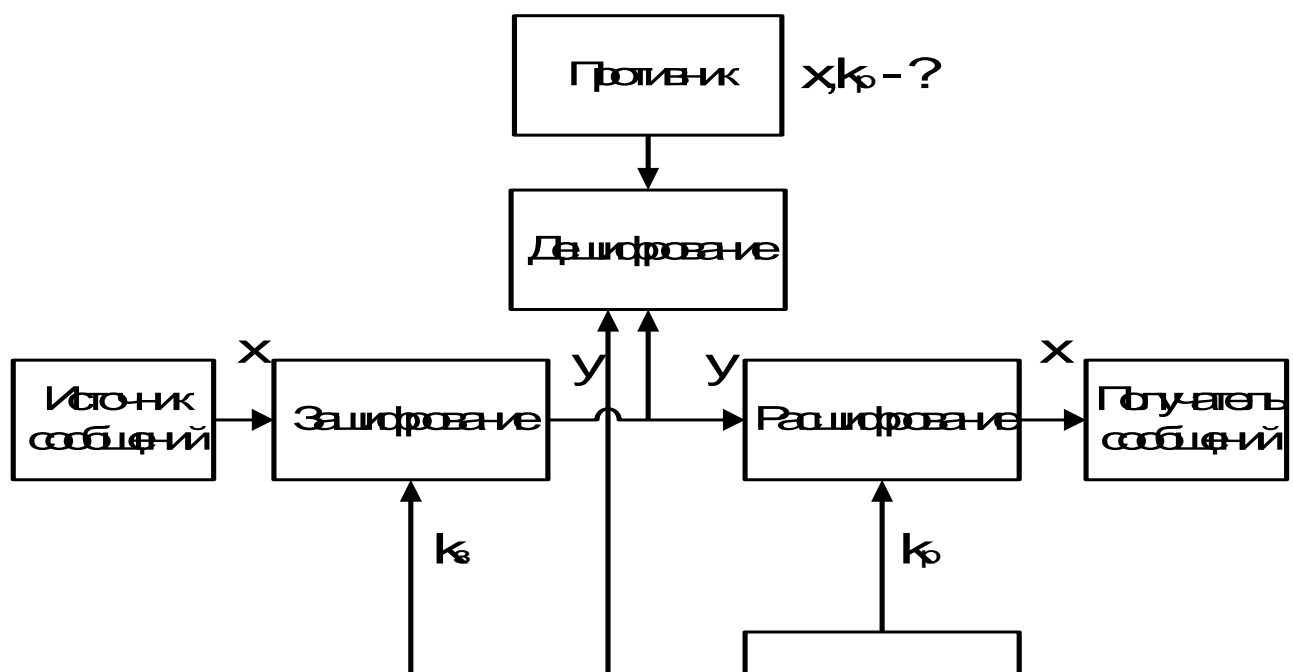


Рис.2. Схема асимметричного шифрования.

В связи с развитием современной и сложной техники защиты информации появилась потребность в подготовке соответствующих специалистов, разрабатывающих и реализующих эту технику на практике. В целях обучения таких специалистов для органов государственной службы в нашей стране создан ИКСИ Академии ФСБ РФ. Одновременно азы защиты информации преподают уже многие вузы страны (МГУ, МИРЭА, РГГУ, МИФИ и др.). Однако следует отметить, что доступ к наиболее важным криптографическим секретам государство охраняет режимом строгой секретности (как, впрочем, и в ведущих странах Запада). Слушатели ИКСИ получают возможность ознакомиться с некоторыми из этих секретов.

Параллельно с развитием криптографической деятельности развивались и соответствующие государственные структуры. В нашей стране государственные криптографические службы прошли довольно сложный путь эволюции. Многочисленные реорганизации повредили их нормальному развитию. В последнее время основным органом нашей страны в области информационной безопасности являлось Федеральное агентство правительственной связи и информации (ФАПСИ) при Президенте РФ, ныне расформированное и в основном переданное в структуру ФСБ РФ. В США стабильно развивается Агентство национальной безопасности (АНБ США); в Великобритании – штаб-квартира правительственной связи и т.д.

1.2. Основные понятия криптографии

Термины и определения. *Криптография* – область науки, техники и практической деятельности, связанная с разработкой, применением и анализом *криптографических систем* защиты информации (рис. 3). Основными функциями криптографических систем является обеспечение *конфиденциальности* и *аутентификации* различных аспектов информационного взаимодействия. Источником угроз при решении криптографических задач считаются преднамеренные действия противника или недобросовестного участника информационного взаимодействия, а не случайные искажения информации вследствие помех, отказов и т.п.

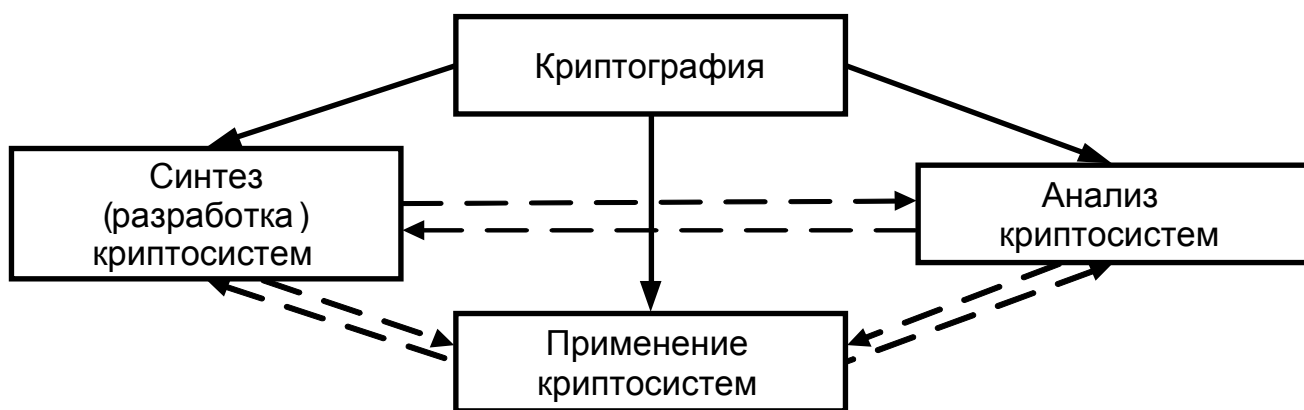


Рис.3. Структура криптографии

Конфиденциальность – защищенность информации от ознакомления с ее содержанием со стороны лиц, не имеющих права доступа к ней.

Аутентификация – установление (проверка и подтверждение) подлинности различных аспектов информационного взаимодействия: сеанса связи, сторон (идентификация), содержания (имитозащита) и источника (установление авторства) передаваемых сообщений, времени взаимодействия и т.д. Является важной составной частью проблемы обеспечения достоверности получаемой информации. Особенно остро эта проблема стоит в случае не доверяющих друг другу сторон, когда источником угроз может служить не только третья сторона (противник), но и сторона, с которой осуществляется информационное взаимодействие.

Система криптографическая (криптосистема) – система обеспечения безопасности защищенной сети, использующая криптографические средства. В качестве подсистем может включать системы шифрования, идентификации, имитозащиты, цифровой подписи и др., а также ключевую систему, обеспечивающую работу остальных систем (рис. 4).

В основе выбора и построения криптосистемы лежит условие обеспечения криптографической стойкости. В зависимости от ключевой системы различают симметричные и асимметричные криптосистемы.

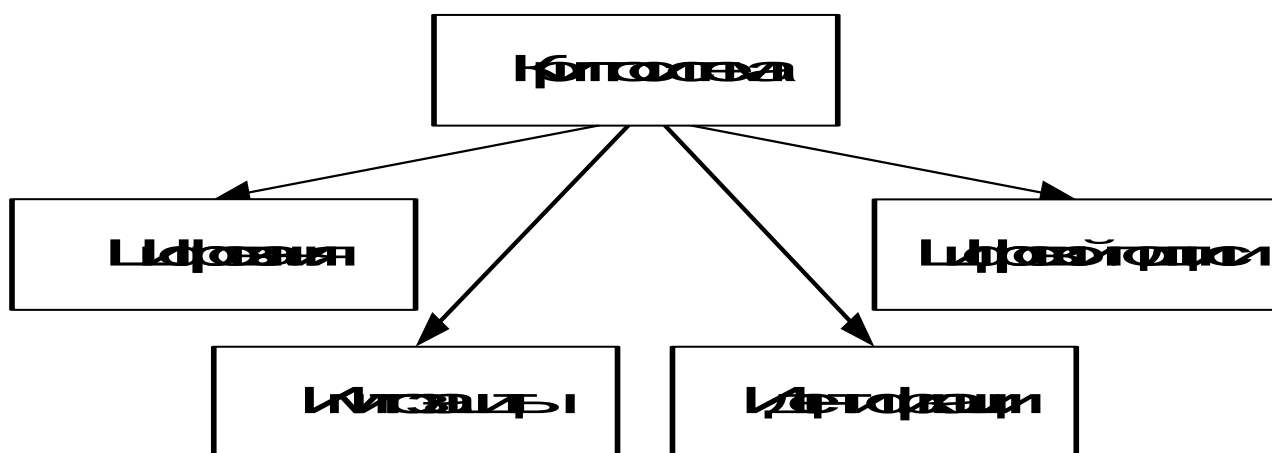


Рис.4. Структура криптосистем

Стойкость криптографическая – свойство криптографической системы, характеризующее ее способность противостоять атакам противника, как правило, с целью получить ключ, открытое сообщение или навязать ложное сообщение.

Симметричные криптосистемы – криптосистемы с симметричными (секретными) ключами. Симметричность означает здесь, что ключи, задающие пару взаимно обратных криптографических преобразований, могут быть получены один из другого с небольшой трудоемкостью. Стойкость симметричной криптосистемы определяется трудоемкостью, с которой противник может вычислить любой из секретных ключей, и оценивается при общепринятом допущении, что противнику известны все элементы криптосистемы, за исключением секретного ключа.

Асимметричные криптосистемы – криптосистемы с асимметричными (секретными и открытыми) ключами. Асимметричность означает здесь, что из двух ключей, задающих

пару взаимно обратных криптографических преобразований, один является секретным, а другой – открытым. Открытые ключи известны всем участникам защищенной сети и противнику, но каждый участник сети хранит в тайне собственный секретный ключ. Стойкость асимметричной криптосистемы определяется трудоемкостью, с которой противник может вычислить секретный ключ, исходя из знания открытого ключа и другой дополнительной информации о криптосистеме.

Система шифрования (шифрсистема) – криптографическая система обеспечения конфиденциальности, предназначенная для защиты информации от ознакомления с ее содержанием лиц, не имеющих права доступа к ней, путем шифрования информации. Математическая модель шифрсистемы включает способ кодирования исходной и выходной информации, шифр и ключевую систему (рис. 5).

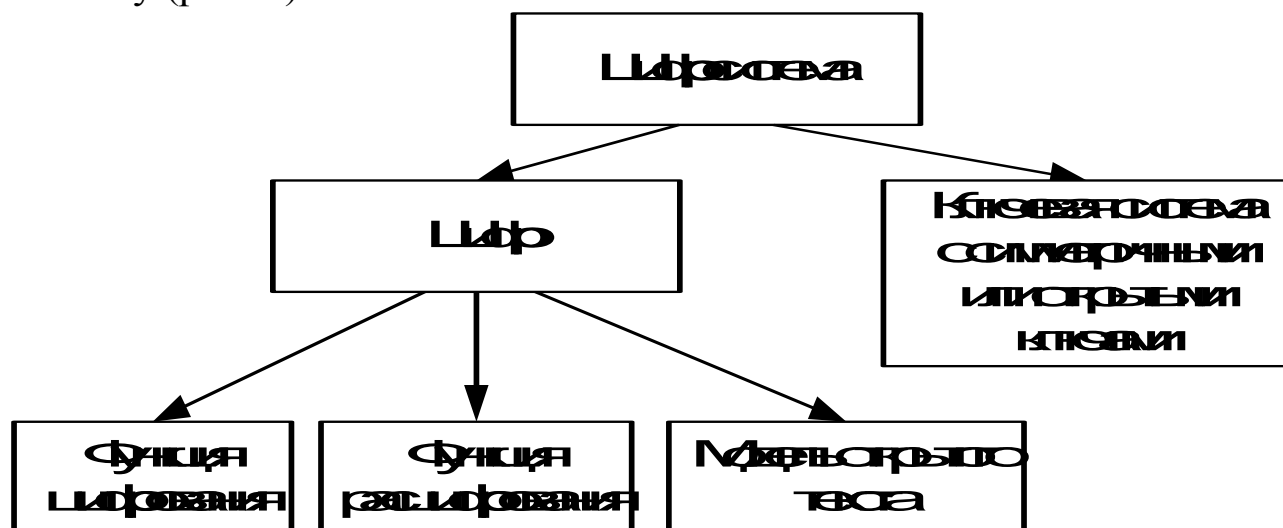


Рис.5. Структура шифрсистемы.

Шифр – семейство обратимых преобразований множества открытых сообщений во множество зашифрованных сообщений и обратно, каждое из которых определяется некоторым параметром, называемым ключом. Математическая модель шифра включает две функции: шифрования и расшифрования, и модель множества открытых сообщений. В зависимости от способа представления открытых сообщений различают блочные, поточные и другие шифры. Основными требованиями, определяющими качество шифра, являются: криптографическая стойкость, имитостойкость, помехоустойчивость и др.

Имитостойкость – способность противостоять активным атакам со стороны противника, целью которых является навязывание ложного или подмена передаваемого сообщения или хранимых данных.

Помехоустойчивость – способность сохранять устойчивую работу при наличии помех в канале связи.

Функция шифрования – осуществляет преобразование множества открытых сообщений во множество зашифрованных сообщений, зависящее от ключа.

Функция расшифрования – преобразует множество зашифрованных сообщений во множество открытых сообщений, зависящее от ключа, является обратным к преобразованию, осуществляемому функцией шифрования.

Алгоритм шифрования – алгоритм, реализующий функцию шифрования.

Алгоритм расшифрования – алгоритм, реализующий функцию расшифрования.

Система цифровой подписи – криптографическая система, выполняющая функцию аутентификации источника сообщения или документа и предназначенная для защиты от отказа субъектов от некоторых из ранее совершенных ими действий. Например, отправитель может отказаться от факта передачи сообщения, утверждая, что его создал сам получатель, а получатель легко может модифицировать, подменить или создать новое сообщение, а затем утверждать, что оно получено от отправителя.

Математическая модель системы цифровой подписи включает схему цифровой подписи и ключевую систему (рис. 6).

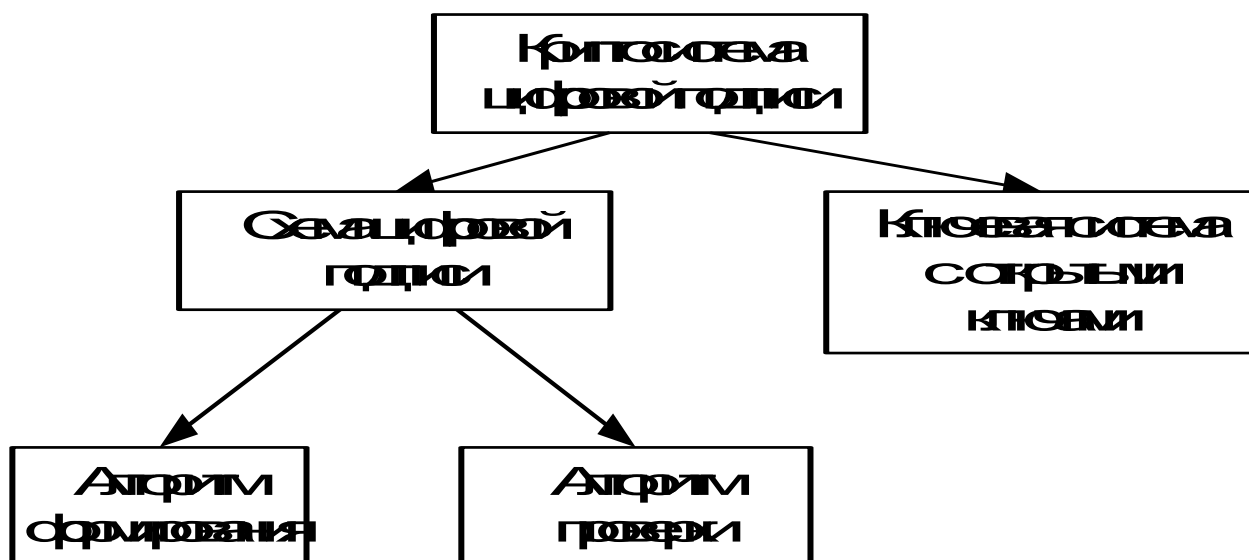


Рис.6. Структура криптосистемы цифровой подписи.

Схема цифровой подписи состоит из двух алгоритмов: один – для формирования; второй – для проверки подписи. Надежность схемы цифровой подписи определяется сложностью следующих трех задач для лица, не являющегося владельцем секретного ключа: *подделки подписи*, т. е. вычисления значения подписи под заданным документом; *создания подписанного сообщения*, т. е. нахождения хотя бы одного сообщения с правильным значением подписи; *подмены сообщения*, т. е. подбора двух различных сообщений с одинаковыми значениями подписи.

Алгоритм формирования цифровой подписи – алгоритм, в качестве исходных данных которого используются сообщение, ключ подписи и параметры схемы цифровой подписи, а в результате формируется цифровая подпись.

Алгоритм проверки цифровой подписи – алгоритм, в качестве исходных данных которого используются подписанное сообщение, ключ проверки и параметры схемы цифровой подписи, а результатом является заключение о правильности или ошибочности цифровой подписи.

Цифровая подпись (сообщения или электронного документа) представляет собой конечную цифровую последовательность, зависящую от самого сообщения или документа и от секретного ключа, известного только подписывающему субъекту, предназначенную для установления авторства. Предполагается, что цифровая подпись должна быть легко проверяемой без получения

доступа к секретному ключу. При возникновении спорной ситуации, связанной с отказом подписывающего от факта подписи некоторого сообщения либо с попыткой подделки подписи, третья сторона должна иметь возможность разрешить спор. Цифровая подпись позволяет решить следующие три задачи: осуществить аутентификацию источника данных; установить целостность сообщения или электронного документа; обеспечить невозможность отказа от факта подписи конкретного сообщения.

Система ключевая определяет порядок использования криптографической системы и включает системы установки и управления ключами (рис. 7).

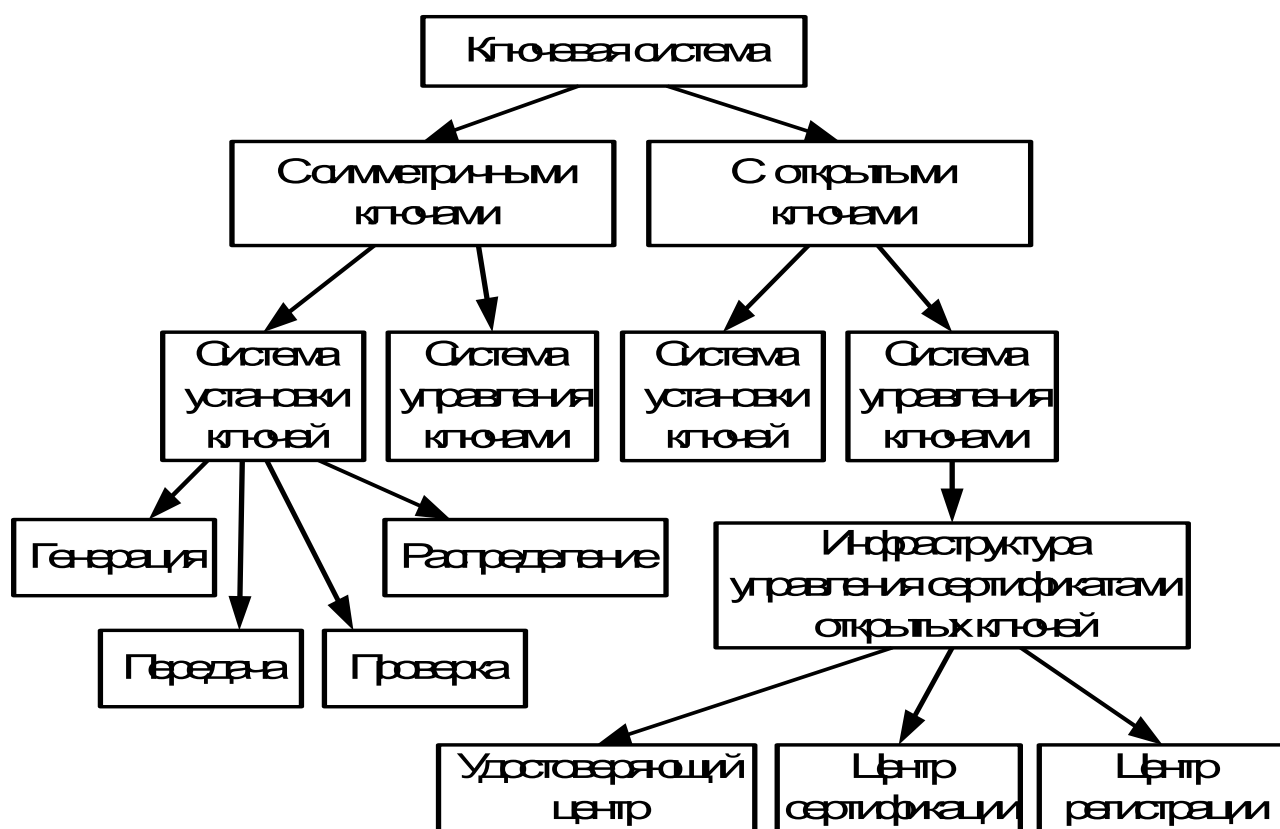


Рис.7. Структура ключевой системы

Система ключевая симметричной криптосистемы основана на использовании симметричных (секретных) ключей. Главными проблемами таких систем являются построение *системы установки ключей* и обеспечение их сохранности для сетей с большим числом абонентов.

Система ключевая асимметричной криптосистемы основана на использовании асимметричных ключей [21], состоящих из пары

– открытого и секретного (закрытого) ключей. Основные проблемы подобных систем состоят в построении *системы управления ключами*, как правило, представляющей собой инфраструктуру управления сертификатами открытых ключей, включающую центры регистрации и сертификации. Функции обоих центров могут объединяться в одном удостоверяющем центре.

Система установки ключе определяет алгоритмы и процедуры генерации, распределения, передачи и проверки ключей.

Система управления ключами устанавливает порядок использования, смены, хранения и архивирования, резервного копирования и восстановления, замены или изъятия из обращения скомпрометированных, а также уничтожения старых ключей. Целью управления ключами является нейтрализация таких угроз, как: компрометация конфиденциальности секретных ключей; компрометация аутентичности секретных или открытых ключей; несанкционированное использование секретных или открытых ключей, например использование ключа, срок действия которого истек.

Жизненный цикл ключей – последовательность стадий, которые проходят ключи от момента генерации до уничтожения. Включает такие стадии, как: генерация ключей, регистрация пользователей и ключей, инициализация ключей, период действия, хранение ключа, замена ключа, архивирование, уничтожение ключей, восстановление ключей, отмена ключей.

Протокол – алгоритм, в котором участвуют две или более стороны, обменивающиеся между собой сообщениями.

Протокол распределения ключей – протокол, в результате выполнения которого взаимодействующие стороны (участники, группы участников) получают необходимые для функционирования криптографической системы ключи. Различают следующие типы протоколов распределения ключей: протоколы передачи (уже сгенерированных) ключей; протоколы (совместной) выработки общего ключа (открытое распределение ключей); схемы предварительного распределения ключей. В зависимости от порядка взаимодействия сторон выделяют *двусторонние протоколы*, в которых стороны осуществляют передачу ключей при непосредственном взаимодействии (протоколы типа «точка –

точка»), и *протоколы с централизованным распределением ключей*, предусматривающие наличие третьей стороны, играющей роль доверенного центра.

Открытое распределение ключей (согласование ключа, выработка общего значения ключа) – протокол, позволяющий двум абонентам выработать общий секретный ключ путем обмена сообщениями по открытому каналу связи без передачи какой-либо общей секретной информации, распределяемой заранее. Важным преимуществом открытого распределения является то, что ни один из абонентов заранее не может определить значение ключа, так как ключ зависит от сообщений, передаваемых в процессе обмена.

Схема предварительного распределения ключей состоит из двух алгоритмов: распределения исходной ключевой информации и формирования ключа. С помощью первого алгоритма осуществляется генерация исходной ключевой информации. Эта информация включает открытую часть, которая будет передана всем сторонам или помещена на общедоступном сервере, а также секретные части каждой стороны. Второй алгоритм предназначен для вычисления действующего значения ключа для взаимодействия между абонентами по имеющейся у них секретной и общей открытой части исходной ключевой информации. Применяется для уменьшения объема хранимой и распределяемой секретной ключевой информации.

Схема предварительного распределения ключей должна быть устойчивой, т. е. учитывать возможность раскрытия части ключей при компрометации, обмане или сговоре абонентов, и гибкой – допускать возможность быстрого восстановления путем исключения скомпрометированных и подключения новых абонентов.

Классификация шифров. По характеру преобразований шифры делятся на два класса: *шифры замены* и *шифры перестановки*, есть также *композиционные шифры* – последовательное применение двух предыдущих (рис. 8).

Шифры замены бывают *симметричные* (ключи зашифрования и расшифрования совпадают) и *асимметричные* (ключи различаются).

В зависимости от того, является ли правило зашифрования однозначной или многозначной функцией, шифры замены делят на шифры *однозначной и многозначной замены*.

Единицы открытого текста называют *шифр-величинами*, а единицы шифрованного текста – *шифр-обозначениями*.

В зависимости от размера шифр-величин шифры замены делятся на *поточные* ($n = 1$) и *блочные* ($n > 1$).

В отдельный подкласс многоалфавитных шифров выделяют *шифры гаммирования*, которые отличаются тем, что основной криптоалгоритм используется для выработки ключевой последовательности – гаммы, а достаточно простая функция шифрования применяются для наложения ключевой последовательности на шифр-величины.

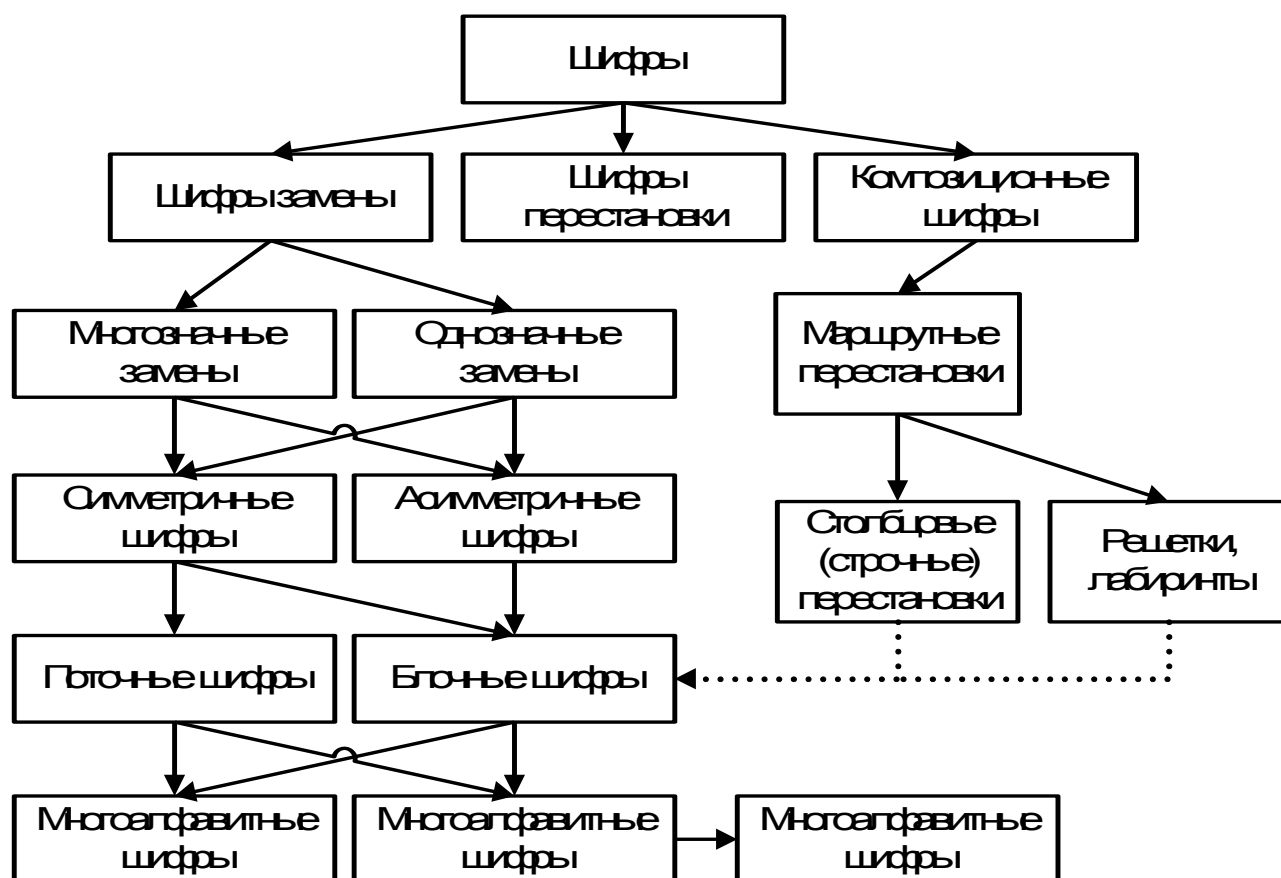


Рис.8. Классификация шифров.

Основой шифров перестановки служат маршрутные перестановки, которые делятся на *столбцовые/строчные (вертикальные)* перестановки, а также *решетки, лабиринты* и пр. В настоящее время шифры перестановки в чистом виде практически

не используются, но сами перестановки являются составляющей частью блочных криптоалгоритмов.

Композиционные шифры представляют собой сочетание в применении шифров замены и перестановки, а также блочных и поточных шифров. Четкой границы класса композиционных шифров не существует. Композиционным может стать любой блочный и поточный алгоритм в зависимости от режима шифрования. Кроме того, в настоящее время активно используются композиции симметричных и асимметричных шифров, такие как: RSA – OAEP, RSA – FDH, RSA – PSS, где симметричная составляющая служит для усиления слабых в реализации мест асимметричных алгоритмов [21].

Характер криптографической деятельности. *Нарушения защиты* (атаки) делятся на две основные группы – пассивные и активные атаки (рис. 9.)

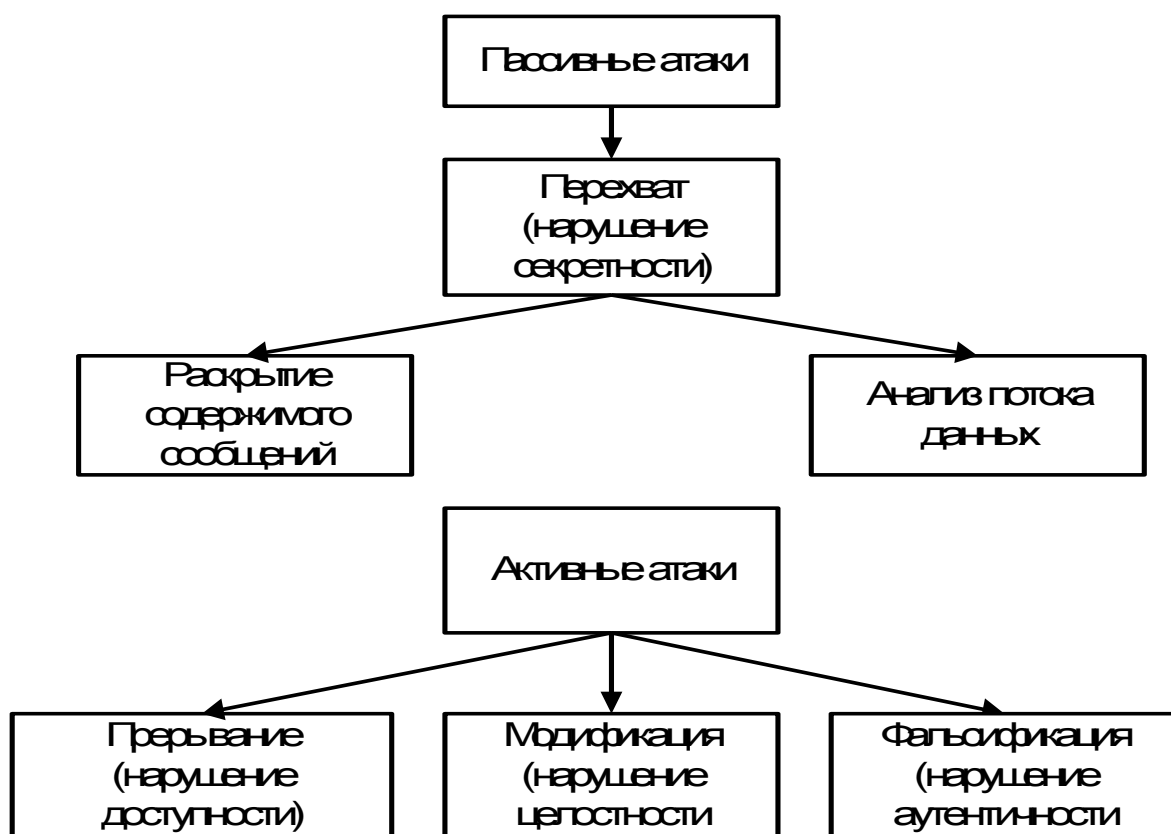


Рис.9. Классификация атак.

Прерывание – нарушение доступности. Ресурс становится недоступным либо непригодным к использованию.

Перехват – нарушение конфиденциальности. К ресурсу открывается несанкционированный доступ.

Модификация – Нарушение целостности. Не только несанкционированный доступ + модификация ресурса.

Фальсификация – нарушение аутентичности. Внесение в систему подложного ресурса.

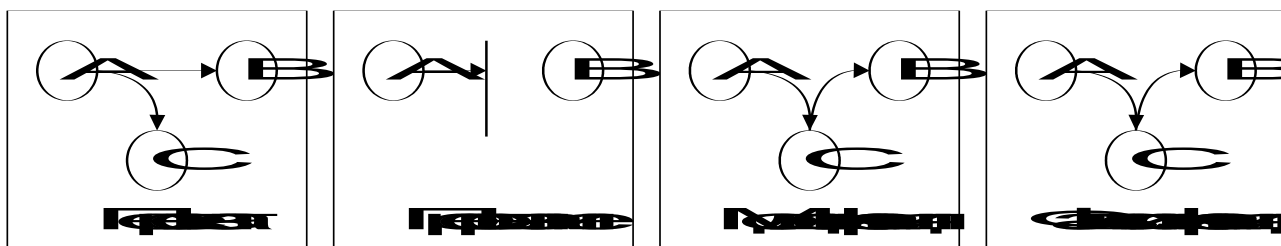


Рис.10. Схемы пассивных и активных атак.

Соответственно предотвращаемым атакам делятся и сервисные службы защиты информации:

- *конфиденциальность* – защита данных от пассивных атак;
- *аутентификация* – надежная идентификация подлинного источника информации;
- *целостность* – защита от модификации;
- *невозможность отречения* – недопущение отказа от факта отправки сообщения, электронно-цифровая подпись;
- *управление доступом* – возможность контроля доступа к ресурсам;
- *доступность* – предупреждение отказов и восстановление доступности.

Контрольные вопросы

1. Дайте понятие шифра простой замены.
2. Какие шифры легли в основу шифров короткопериодичной замены?
3. Какие шифры привели к развитию шифров биграммной замены?
4. Перечислите основные виды криптографических систем.
5. Из каких компонентов состоит ключевая система?
6. Чем различаются понятия многозначной и многоалфавитной замены?

7. В чем различие между симметричным и асимметричным шифрованием?
8. Какова разница между пассивной и активной атаками?

2. СТОЙКОСТЬ КРИПТОГРАФИЧЕСКИХ СИСТЕМ

2.1. Модели шифров и открытых текстов

Математические модели шифра и открытого текста требуются в криптографии для получения и доказательства точных математических результатов. Модели делятся на два класса – алгебраические (детерминированные) и вероятностные (стохастические).

Алгебраические модели шифров. Алгебраическая модель предложенная К. Шенноном.

Пусть X, K, Y – конечные множества открытых текстов, ключей и шифртекстов соответственно, $|X| > 1, |K| > 1, |Y| > 1$, $E_k: X \rightarrow Y$ и $D_k: E_k(X) \rightarrow X$ – правила зашифрования и расшифрования, отвечающие ключу $k \in K$, $E = \{E_k: k \in K\}$, и $D = \{D_k: k \in K\}$. Через $E_k(X)$ обозначили множество $E = \{E_k(x): x \in X\}$. Если ключ $k \in K$ представляется в виде пары $k = (k_z, k_p)$, где k_z – ключ зашифрования, а k_p – ключ расшифрования (причем $k_z \neq k_p$), то E_k понимается как E_{k_z} , а D_k – как D_{k_p} .

Определение: Алгебраической моделью шифра назовем совокупность $\Sigma_A = (X, K, Y, E, D)$ введенных множеств, для которых выполняются условия:

- 1) при любых $x \in X$ и $k \in K$ выполняется равенство $D_k(E_k(x)) = x$;
- 2) справедливо равенство $E_k(X)$.

Условие 1 отвечает требованию однозначности расшифрования. Условие 2 означает, что любой элемент $y \in Y$ может быть представлен в виде $E_k(x)$ для подходящих элементов $x \in X$ и $k \in K$. В общем случае E_k могут быть многозначными отображениями, но здесь ограничиваемся изучением лишь *однозначных шифров*, получивших наибольшее распространение.

Шифр называется *эндоморфным*, если $Y = X$. Для эндоморфного шифра правило зашифрования E_k осуществляет биективное отображение множества X на себя. В этом случае

удобно рассматривать правила зашифрования как подстановки множества X , и опускать индекс в записи E_k , предполагая, что правила зашифрования пронумерованы ключами. По сути, вся информация об эндоморфном шифре содержится во множествах X и K .

Определение: Подстановочной моделью эндоморфного шифра назовем упрощенную совокупность $\Sigma_{\Pi} = (X, E)$

Примеры.

1. Шифр простой замены в алфавите A :

Пусть $A^i, K \subseteq S(A)$,

где $S(A)$ – симметрическая группа подстановок множества A и L – натуральное число.

$\forall k \in K, x = (x_1, \dots, x_l), y = (y_1, \dots, y_l): E_k(x) = (k(x_1), \dots, k(x_l)), D_k(y) = (k^{-1}(y_1), \dots, k^{-1}(y_l))$,

где k^{-1} – подстановка, обратная k .

Подстановочной моделью данного эндоморфного шифра является совокупность (X, E) , для которой $e(x_1, \dots, x_l) = k(x_1), \dots, k(x_l)$. Здесь $x = x_1, \dots, x_l \in X, e \in E$, а k – ключ, поставленный в соответствие подстановке e .

2. Шифр перестановки.

Пусть $X = Y = A_L, K \subseteq S_L$,

где S_L – симметрическая группа подстановок множества $\{1, 2, \dots, L\}$.

$\forall k \in K, x = (x_1, \dots, x_L), y = (y_1, \dots, y_L): E_k(x) = (x_{k(1)}, \dots, x_{k(L)}), D(y) = (y_{k^{-1}(1)}, \dots, y_{k^{-1}(L)})$,

где k^{-1} – подстановка, обратная k .

3. Шифр RSA.

Пусть $n = pq$ (q и p – простые), $X = Y = Z_n$ – кольцо вычетов по модулю n .

$K = \{(n, p, q, a, b): a, b \in Z_n, n = pq, ab \equiv 1 \pmod{\varphi(n)}\}$,

где φ – функция Эйлера.

$k = (k_z, k_p)$, где $k_z = (n, b)$ и $k_p = (n, p, q, a)$ – ключи. Правила зашифрования и расшифрования для $x \in X$ и $y \in Y$ определяются формулами $E_{k_z}(x) = x^b \bmod n$, $D_{k_p}(y) = y^a \bmod n$.

Уточним алгебраическую модель шифра, выбрав за основу модель, в которой открытыми и шифрованными текстами служат *шифр-величины* и *шифр-обозначения*, с которыми оперирует шифр. Назовем выбранную модель опорным шифром. Будем шифровать фрагменты открытого текста одним из априорно заданных правил зашифрования, составляющих опорный шифр. Назовем эти отображения простыми заменами. Простые замены будем выбирать при помощи вспомогательной последовательности, которую назовем ключевым потоком. Ключевой поток может определяться случайным образом или вычисляться детерминированно в зависимости от выбранного ключа шифра. Большинство шифров использует детерминированный ключевой поток.

Определение: *Опорным шифром назовем совокупность $\Sigma = (U, K, V, E, D)$, состоящую из: U и V – множеств шифр-величин и шифр-обозначений, с которыми оперирует шифр; K – множества ключей (номеров простых замен); E и D – множеств простых замен $E_k : U \rightarrow V, k \in K$, и обратных к ним отображений $D_k : E_k(U) \rightarrow U$.*

Если шифр использует s простых замен, то примем в качестве множества ключей опорного шифра множество целых чисел $K = \{0, 1, \dots, s - 1\}$. Опорный шифр определяет способ зашифрования фрагментов открытого текста – шифр-величин. Для работы с последовательностью шифр-величин требуется **степень** опорного шифра.

Определение: *l -й степенью опорного шифра назовем совокупность множеств $\Sigma^l = (U^l, K^l, V^l, E^{(l)}, D^{(l)})$,*

где U^l, K^l, V^l – декартовы степени; множество $E^{(l)}$ состоит из отображений ~~$E_k : U \rightarrow V$~~ , таких, что для ~~$u \in U$~~ и ~~$k \in K$~~

множество $D^{(l)}$ состоит из отображений $D_{\bar{k}} : E_{\bar{k}}(U^l) \rightarrow U^l, \bar{k} \in K^l$, таких, что для $\bar{v} = v_1 \dots v_l \in E_{\bar{k}}(U^l)$ и $\bar{k} = k_1 \dots k_l \in K^l$

$$D_{\bar{k}}(\bar{v}) = D_{k_l}(v_l) \dots D_{k_1}(v_1) \in U^l, D_{k_i} D_{k_j} = D_{k_j} D_{k_i}, i = 1, l.$$

Теперь рассмотрим построение ключевого потока, т. е. последовательности $k_1 \dots k_l$, $k_j \in K$, $i = \overline{1, l}$, номеров простых замен, используемых для зашифрования открытого текста $\bar{u} = u_1 \dots u_l$, $u_i \in U^l$, $i \in \overline{1, l}$.

Имеется два принципиально разных способа построения такой последовательности. В первом случае она строится случайно. Устройство, вырабатывающее случайный ключевой поток, называется случайным генератором ключевого потока. Формально его можно представить отображением $\psi: N \rightarrow K^*$, (K^* – множество слов конечной длины в алфавите C), ставящим в соответствие натуральному числу l последовательность из l испытаний некоторой случайной величины, принимающей значение из K с некоторыми ненулевыми вероятностями. Простейший случайный генератор – игровая рулетка.

Во втором случае шифр имеет некоторое, априорно заданное, конечное множество ключей K . Каждому ключу $k \in K$ и натуральному числу $l \in N$ ставится в соответствие однозначно определенный ключевой поток $k_1 \dots k_l$, $k_i \in K$, $i = \overline{1, l}$. Этот поток вырабатывается детерминированным генератором ключевого потока.

Определение: пусть $\psi: K \times N \rightarrow K^*$ – произвольное отображение, такое, что для любых $k \in K$, $l \in N$ и некоторых $k_i \in K$, $i = \overline{1, l}$, $\psi(k, l) = k_1 \dots k_l$, причем $\{\psi(k, l), k \in K\} = K$.

Назовем последовательность $k_1 \dots k_l$ ключевым потоком, отвечающим ключу k и числу l , а само отображение ψ – детерминированным генератором ключевого потока.

Если шифр использует случайный генератор, ключами шифра считают сами ключевые потоки. В таком случае, допуская вероятность зашифрования последовательности шифр-величин любой конечной длины, получаем шифр с бесконечным множеством ключей. Если же шифр использует детерминированный генератор ключевого потока, множеством ключей является конечное множество K . Характер генератора ключевого потока делит все множество шифров на два класса.

Определение: Совокупность $\Sigma_H = (\Sigma^l, l \in N; \psi)$ степеней опорного шифра, в которых для зашифрования последовательностей шифр-величин u_1, \dots, u_l по

правилу $E_{\bar{k}}(\bar{u}) = E_{k_1}(u_1) \dots E_{k_l}(u_l) \in V^l, E_{k_i} E, i = \overline{1, l}$ ключевой поток строится с помощью случайного генератора ψ , назовем алгебраической моделью шифра с неограниченным ключом (или просто шифром с неограниченным ключом).

Определение: Совокупность $\Sigma_0 = (\Sigma^l, l \in \mathbb{N}; \psi)$ степеней опорного шифра, в которых для зашифрования последовательностей шифр-величин u_1, \dots, u_l по правилу $E_{\bar{k}}(\bar{u}) = E_{k_1}(u_1) \dots E_{k_l}(u_l) \in V^l, E_{k_i} E, i = \overline{1, l}$ ключевой поток строится с помощью детерминированного генератора ψ , назовем алгебраической моделью шифра с ограниченным ключом (или просто шифром с ограниченным ключом).

Криптографические свойства шифра с ограниченным ключом определяются в первую очередь свойствами его генератора ключевого потока. Например, если $\psi(k, l) = k' \dots k'$ для любого $k \in K$ и подходящего $k' \in K$, то получаем «слабый» шифр простой замены. Если $\psi(k, l) = k_1 \dots k_p k_1 \dots k_p \dots$ представляет собой периодическую последовательность (в которой $|\{k_1 \dots k_p\}| \geq 2$), то получаем более стойкий шифр периодической замены, например шифр Виженера.

Вероятностные модели шифров. Введем теперь вероятностную модель шифра. Определим априорные распределения вероятностей $P(X)$, $P(K)$ на множествах X и K соответственно. Тем самым для любого $x \in X$ определена вероятность $p_X(x) \in P(X)$ и для любого $k \in K$ – вероятность $p_K(k) \in P(K)$, причем выполняются равенства

$$\sum_{x \in X} p_X(x) = 1 \text{ и } \sum_{k \in K} p_K(k) = 1.$$

В тех случаях, когда требуется знание распределений $P(X)$ и $P(K)$, мы будем пользоваться *вероятностной моделью* Σ_B , состоящей из пяти множеств, связанных условиями 1) и 2) предыдущего определения алгебраической модели шифра, и двух вероятностных распределений:

$$\Sigma_B = (X, K, Y, E, D, P(X), P(K)).$$

Распределение $P(Y)$ индуцируется распределениями $P(X)$ и $P(K)$ согласно формуле полной вероятности:

$$p_Y(y) = \sum_{\substack{(x,k) \in X \times K: \\ Ek(x)=y}} p_X(x) \cdot p_K(k)$$

В большинстве случаев множества X и Y представляют собой объединения декартовых степеней A и B соответственно, так что для некоторых натуральных L и L_1

$$X = \bigcup_{i=1}^L A^i, \quad Y = \bigcup_{i=1}^{L_1} B^i.$$

Множества A и B называют соответственно алфавитом открытого текста и алфавитом шифрованного текста. Другими словами, открытые и шифрованные тексты записываются привычным образом в виде последовательности букв.

Принята также формулировка вероятностной модели шифра в которой вместо включения в совокупность распределений случайных величин множества X, K, Y , рассматриваются как случайные величины $\tilde{X}, \tilde{K}, \tilde{Y}$, полагая при этом случайные величины \tilde{X}, \tilde{K} независимыми.

Определение: Пусть для $l \in \mathbb{N}$ ~~$\mathcal{U}^l, \mathcal{K}^l, \mathcal{V}^l$~~ совокупность, состоящая из случайных величин ~~$\mathcal{U}, \mathcal{K}, \mathcal{V}$~~ , множеств правил зашифрования и расшифрования E^l, D^l , для которой выполняются условия ~~$\mathcal{U}^l, \mathcal{K}^l, \mathcal{V}^l$~~ при любых $\bar{u} \in U^l, \bar{k} \in K^l$.

Тогда вероятностной моделью шифра с неограниченным ключом назовем семейство

$$\tilde{\Sigma}_l = (\tilde{\Sigma}^{(l)}, l \in \mathbb{N}; \psi),$$

где ψ – случайный генератор ключевого потока.

Аналогично вводится вероятностная модель шифра с ограниченным ключом, только вместо множества K^l всех ключевых потоков длины l рассмотрим множество $K^{(l)}$ возможных ключевых потоков длины l .

Определение: Пусть для $l \in \mathbb{N}$ ~~$\mathcal{U}^l, \mathcal{K}^l, \mathcal{V}^l$~~ совокупность, состоящая из случайных величин ~~$\mathcal{U}, \mathcal{K}, \mathcal{V}$~~ , множеств правил зашифрования и расшифрования E^l, D^l , где распределение $P(K^{(l)})$ определяется формулой $p_{K^{(l)}}(\bar{k}) = \sum_{\hat{e} \in K_l(\bar{k})} p_K(\hat{e})$ и

при любых $\bar{u} \in U^l$

$$P\{\tilde{U}^{(i)} = \bar{u}\} > 0, P\{\tilde{K}^i = \bar{k}\} > 0$$

Тогда вероятностной моделью шифра с ограниченным ключом назовем семейство

$$\{\psi^k\},$$

где ψ – детерминированный генератор ключевого потока.

Математические модели открытых сообщений.

Применяются для математических исследований свойств шифров и для автоматизации криптоанализа. Учет частот m -грамм приводит к следующей модели открытого текста.

Открытое сообщение – последовательность знаков (слов) некоторого алфавита.

Различают естественные (языки), и специальные (цифровые, буквенно-цифровые) алфавиты.

Естественные алфавиты могут быть нормальными и смешанными (*систематически перемешанные* на основе правила и *случайные*). Смешанные используются в качестве нижней строки подстановки.

Примеры специальных алфавитов:

- код Бодо – 32 значный 5-битовый алфавит для телетайпов и телексов. 26 английских букв, пробел, скобки, вопрос и плюс;
- духбуквенный алфавит Ф. Бэкона – двоичный код из букв А и В;
- код ASCII – 256-символьный 8-битовый алфавит. Может быть в различных формах записи (десятичная, двоичная и пр.).

Частотные характеристики шифра. Наиболее важная характеристика – *избыточность* открытого текста.

Более простые характеристики – это:

- повторяемость букв, пар букв (биграмм), m -грамм;
- сочетаемость букв друг с другом (гласные-согласные и пр.).

Такие характеристики устанавливаются на основе эмпирического анализа текстов достаточно большой длины.

Эксперимент по оценке вероятности появления в тексте фиксированных m -грамм (для небольших m). Подсчет чисел вхождений каждой из n^m возможных биграмм в достаточно длинных открытых текстах $T = t_1 t_2 \dots t_l$, составленных из букв алфавита $\{a_1, a_2, \dots, a_n\}$.

При этом просматриваются подряд идущие m -граммы текста:

$$t_1 t_2 \dots t_m, t_2 t_3 \dots t_{m+1}, \dots, t_{l-m+1} t_{l-m+2} \dots t_l.$$

Если $\phi_{a_1 a_2 \dots a_m}^{i_1 i_2 \dots i_m}$ – число появлений m -граммы $a_1 a_2 \dots a_m$ в тексте T , а L – общее число подсчитанных m -грамм, то при достаточно больших L

$$P_{a_1 a_2 \dots a_m}^{i_1 i_2 \dots i_m} = \frac{\phi_{a_1 a_2 \dots a_m}^{i_1 i_2 \dots i_m}}{L}.$$

При анализе сочетаемости букв друг с другом используют понятие *условной вероятности* (зависимость появления буквы в тексте от предыдущих букв).

Для условных вероятностей выполняются неравенства:

$$P_{a_1 a_2 \dots a_m}^{i_1 i_2 \dots i_m} \leq P_{a_1 a_2 \dots a_m}^{i_1 i_2 \dots i_{m-1}}.$$

Была отмечена устойчивая закономерность чередования гласных и согласных. Зависимость появления букв текста вслед за несколькими предыдущими ощутима на глубину в 30 знаков, после чего практически отсутствует.

Вероятностная модель m -го приближения. Пусть $P^{(m)}(A)$ – массив, состоящий из приближений для вероятностей $p(b_1 b_2 \dots b_m)$ появления m -грамм $b_1 b_2 \dots b_m$ в открытом тексте, $m \in \mathbb{N}$, $A = \{a_1, \dots, a_n\}$ – алфавит открытого текста, $b_i \in A$, $i = \overline{1, m}$. Тогда источник открытого текста генерирует последовательность $c_1, c_2, \dots, c_k, c_{k+1}, \dots$ знаков алфавита A , в которой m -грамма $c_1 c_2 \dots c_m$ появляется с вероятностью $p(c_1 c_2 \dots c_m) \in P^{(m)}(A)$.

Вероятностная модель первого приближения. Последовательность знаков c_1, c_2, \dots в которой каждый знак c_i , $i = 1, 2, \dots$, появляется с вероятностью $p(c_i) \in P^{(1)}(A)$, независимо от других знаков.

В такой модели открытый текст имеет вероятность

$$P(c_1 c_2 \dots c_k) = \prod_{i=1}^k p(c_i).$$

Вероятностная модель второго приближения. Первый знак c_1 имеет вероятность $p(c_1) \in P^{(1)}(A)$, а каждый следующий знак c_i зависит от предыдущего и появляется с вероятностью

$$p(c_i | c_{i-1}) = \frac{p(c_{i-1} c_i)}{p(c_{i-1})},$$

где $p(c_{i-1} c_i) \in P^{(2)}(A)$, $p(c_{i-1}) \in P^{(1)}(A)$, $i = 2, 3, \dots$

Другими словами, модель открытого текста второго приближения представляет собой *простую однородную цепь*

Маркова. В такой модели открытый текст $c_1c_2\dots c_l$ имеет вероятность

$$P(c_1c_2\dots c_l) = \prod_{i=1}^l P(c_i | c_1\dots c_{i-1}).$$

С общих позиций открытый текст рассматривается как стационарный эргодический случайный процесс с дискретным временем и конечным числом состояний.

Критерии распознавания открытого текста строятся на основе моделей открытого текста двумя методами:

- путем различения статистических гипотез;
- на базе ограничений по запретным или ожидаемым сочетаниям букв (БЪ и пр.).

Первый подход. Открытый текст – реализация независимых испытаний случайной величины, значениями которой являются буквы алфавита $A = \{a_1, \dots, a_n\}$, появляющиеся в соответствии с распределением вероятностей $P(A) = (p(a_1), \dots, p(a_n))$. Требуется определить, является ли случайная последовательность $c_1c_2\dots c_l$ букв алфавита A открытым текстом или нет.

Пусть H_0 – гипотеза, состоящая в том, что данная последовательность – открытый текст, H_1 – альтернативная гипотеза. В простейшем случае последовательность $c_1c_2\dots c_l$ можно рассматривать при гипотезе H_1 как случайную и равновероятную, значениями которой являются буквы алфавита $A = \{a_1, \dots, a_n\}$, появляющиеся в соответствии с распределением вероятностей $Q(A) = (q(a_1), \dots, q(a_n))$.

Наиболее мощный критерий различения двух простых гипотез – *лемма Неймана – Пирсона*. Также может использоваться и теорема Фробениуса.

Возможны ошибки двух родов:

- ошибка первого рода (открытый текст принят за случайный набор знаков), ее вероятность $\alpha = P(H_0 | H_1)$;
- ошибка второго рода (случайный набор знаков принимается за открытый текст), ее вероятность $\beta = P(H_1 | H_0)$.

Второй подход. Критерий запретных m -грамм. Устроен просто. Отбирается некоторое число s редких m -грамм, которые объявляются запретными. Теперь, последовательно просматривая

все m -граммы анализируемой последовательности $c_1c_2\dots c_l$, объявляем ее случайной, как только в ней встретится одна из запретных k -грамм. Весьма эффективны, несмотря на простоту.

Распознавание открытого текста производится также на основе особенностей нетекстовых сообщений (файловые метки и пр.).

2.2. Криптографическая стойкость шифров

Попытки противника по добыванию зашифрованной информации называют криптоатаками. В симметричных криптосистемах рассматривают следующие криптоатаки:

1) Атака на основе шифртекста.

Криптоаналитик располагает шифртекстами $y_1 = E_{k_1}(x_1), \dots, y_m = E_{k_m}(x_m)$, отвечающими неизвестным открытым текстам различных сообщений. Требуется определить хотя бы одно из сообщений $x_i, i = \overline{1, m}$, (или соответствующий ключ k_i), исходя из необходимого числа m криптограмм, или убедиться в своей неспособности сделать это. В качестве частных случаев возможно совпадение ключей $k_1 = \dots = k_m$ или совпадение открытых текстов

$$x_1 = \dots = x_m.$$

2) Атака на основе известного открытого текста.

Криптоаналитик располагает парами ~~открытых~~ - ~~шифрованных~~ открытых и отвечающих им шифрованных текстов. Требуется определить ключ k_i для хотя бы одной из пар. В частном случае, когда $k_1 = \dots = k_m$, требуется определить ключ k или, убедившись в своей неспособности сделать это, определить открытый текст x_{m+1} еще одной криптограммы $y_{m+1} = E_k(x_{m+1})$, зашифрованный на том же ключе.

3) Атака на основе выбранного открытого текста.

Эта атака отличается от предыдущей лишь тем, что криптоаналитик имеет возможность выбора открытых текстов x_1, \dots, x_m . Цель атаки та же, что и предыдущей. Подобная атака возможна в случае, когда криптоаналитик имеет доступ к шифратору передающей стороны, или в системах опознавания свой – чужой

4) Атака на основе выбранного шифртекста.

Атака отличается от второй атаки тем, что криптоаналитик имеет возможность выбора шифртекстов $\mathcal{M}_1, \dots, \mathcal{M}_m$. Цель атаки та же, что и во втором случае. Подобная атака возможна, когда криптоаналитик имеет доступ к шифратору принимающей стороны.

Атаки на основе выбранных текстов считаются наиболее опасными. Иногда к указанным атакам добавляют и другие.

Правило Керкгоффса – компрометация системы не должна причинять неудобств ее пользователям (надежность определяется только секретностью ключа).

В криптографии различают два подхода к стойкости: *теоретическую и практическую (или вычислительную) стойкость*.

Теоретико-информационный подход к оценке криптостойкости шифров. Энтропия и избыточность языка. Свойства текстов изучаются методами теории информации, разработанной К. Шенноном. Ключевое понятие – энтропия, устанавливаемая функцией от вероятностного определения и характеризующая количество неопределенности или информации в случайном эксперименте. Неопределенность и информация измеряются одной и той же мерой. Применительно к независимым испытаниям случайной величины ξ с распределением вероятностей $\xi = \begin{pmatrix} a_1, a_2, \dots, a_n \\ p_1, p_2, \dots, p_n \end{pmatrix}$ энтропия $H(\xi)$ определяется формулой

$$H(\xi) = - \sum_{i=1}^n p_i \cdot \log_2 p_i.$$

Единицей количества информации считается 1 бит. При $p_i = 1/n$ при всех $i = \overline{1, n}$

$$H(\xi) = \log_2 n.$$

Мерой среднего количества информации, приходящейся на одну букву открытого текста языка Λ (рассматриваемого как источник случайных текстов), служит величина H_Λ , называемая энтропией языка Λ . Она вычисляется последовательными приближениями позначных моделей текста: H_1, H_2, \dots, H_r .

Для каждого языка значение H_Λ стремится к определенному пределу (после $r = 30$ предел уже устанавливается):

$$H_{\Lambda} = \lim_{r \rightarrow \infty} \frac{H_r}{r},$$

при этом формула

$$R_{\Lambda} = 1 - \frac{H_{\Lambda}}{\log 2}$$

определяет *избыточность языка* R_{Λ} .

Разговорные языки имеют весьма большую избыточность. Избыточность текста в 75 % означает, что при оптимальном кодировании текста (например использование кодов Хаффмана, Фано или др.) его можно сжать до четверти без потери информации.

Энтропию можно определить и по-другому. Для n -буквенного алфавита число текстов длиной L , удовлетворяющих статистическим ограничениям, равно (при достаточно больших L) не $n^L = 2^{L \log_2 n} = 2^{L H_0}$, как это было бы, если бы имели право брать *любые* наборы из L букв, а всего лишь

$$n^L \cdot 2^{-L R_{\Lambda}}.$$

По сути это – приближенное число осмысленных текстов длиной L для данного языка Λ . Исходя из этого, можно определить энтропию языка по формуле

$$H_{\Lambda} = \frac{1}{L} \log_2 \left(\frac{n^L \cdot 2^{-L R_{\Lambda}}}{n^L} \right)$$

Расстояние единственности. При дешифровании криптограмм может возникнуть ситуация, в которой несколько найденных ключей дают осмысленный текст. Например, криптограмму WNAJW, полученную при помощи шифра Цезаря, порождают два открытых текста RIVER и ARENA, отвечающих величинам сдвига (ключам) 5 и 11 соответственно. Из этих ключей один является *истинным*, а другой – *ложным*. Найдем оценку для числа ложных ключей. Для этого рассмотрим связь между энтропиями вероятностных распределений $P(X)$, $P(K)$, $P(Y)$, заданных на компонентах X , K , Y произвольного шифра Σ_b [2].

Назовем условную энтропию $H(K/Y)$ *неопределенностью шифра Σ_b по ключу*. Она измеряет среднее количество информации о ключе, которое дает шифртекст. Аналогично вводится *неопределенность шифра по открытому тексту* $H(X/Y)$. Эти величины являются мерой теоретической стойкости шифра.

Минимально возможным значением неопределенности $H(X/Y)$ является нуль.

что возможно только в тех случаях, когда $\log p(x|y) = 0$ или $p(x|y) = 1$ для всех x, y , т. е. если $p(x|y) = 1$ при некоторых x, y . Это означает, что по данному y можно получить существенную информацию об x , что свидетельствует о слабости шифра.

Идеальной является ситуация, когда $H(X/Y) = H(X)$. Именно в этом случае шифр можно было бы назвать совершенным.

Связь между энтропиями компонент шифра дает формула неопределенности шифра по ключу:

полученная К. Шенноном. Она позволяет получить оценку среднего числа ложных ключей.

Введем обозначение $K(y) = \{k \in K : \exists x \in X, E_k(x) = y\}$ – множество ключей, для каждого из которых y является результатом зашифрования некоторого осмысленного текста длиной L . Если располагаем криптограммой y , то число ложных ключей равно

$|K(y)| - 1$, так как лишь один из допустимых ключей является истинным. Определим среднее число ложных ключей k_L (относительно всех возможных шифртекстов длиной L) по формуле

$$\hat{e}_L = \sum_{y \in Y} p(y) \cdot |K(y)| - 1.$$

Теорема: для любого рассматриваемого шифра Σ_B с равновероятными ключами при достаточно больших значениях L в алфавите из n букв имеет место неравенство

$$k_L \geq \frac{|K|}{n^{L \cdot R_A}} - 1,$$

где R_A – избыточность данного языка.

Назовем расстоянием единственности для шифра Σ_B натуральное число L_0 , для которого ожидаемое число ложных ключей k_L равно нулю. По сути, расстояние единственности есть средняя длина шифртекста, необходимая для однозначного восстановления истинного ключа (без каких-либо временных ограничений на время его нахождения).

Непосредственно из предыдущего неравенства следует, что

$$\frac{|K|}{k_L + 1} \leq n^{L \cdot R_A},$$

откуда при $\kappa_L = 0$ получаем $|K| \leq n^{L_{R_A}}$, и, следовательно,

$$L \geq \frac{\log K}{R_A \log 2}.$$

Минимально возможное значение в этом неравенстве принимается за L_0 . Таким образом,

$$L_0 = \left\lceil \frac{\log K}{R_A \log 2} \right\rceil.$$

Например для шифра простой замены с параметрами $n = 26$, $|K| = 26!$, $R_A = 0,5$ (примерно соответствует английскому языку) получим оценку:

$$L_0 = \left\lceil \frac{88}{0,547} \right\rceil = 3.$$

Это значит, что для шифра простой замены для английского языка в среднем по криптограмме длиной около 40 символов можно однозначно определить открытый текст.

Для совершенных шифров типа одноразовых блокнотов расстояние единственности $L_0 = \infty$.

Теоретическая стойкость шифров. При анализе теоретической стойкости шифров отвлекаются от объема реальных затрат на дешифрование. Основным критерием является возможность получения на основе шифртекста *вероятностной информации* об открытом тексте или используемом ключе. Для теоретически стойких (совершенных) шифров сама задача дешифрования становится бессмысленной. Никакой метод криптоанализа, включая полный перебор ключей, не позволяет не только определить ключ или открытый текст, но даже получить о них какую-либо информацию (за исключением длины открытого текста).

Априорная вероятность открытого текста – безусловная вероятность.

Апостериорная вероятность открытого текста – вероятность (по шифртексту) при условии использования соответствующего шифра.

Стойкость по отношению к атаке на основе единственного шифртекста. К. Шеннон назвал шифр *совершенным*, если шифртекст не дает никакой вероятностной информации об открытом тексте на языке вероятностной модели ~~Шеннона~~.

Определение. Назовем шифр Σ_v совершенным, если для любых $x \in X, y \in Y$ выполняется равенство $p(x / y) = p_X(x)$.

Утверждение. Если шифр Σ_v – совершенный, то $|X| \leq |Y| \leq |K|$.

Теорема (К. Шеннон): пусть Σ_v – шифр, для которого $|X| = |Y| = |K|$. Тогда шифр Σ_v – совершенный тогда и только тогда, когда выполняются два условия:

- 1) для любых $x \in X, y \in Y$ существует единственный ключ $k \in K$, для которого $E_k(x) = y$;
- 2) распределение вероятностей $P(K)$ – равномерное, т. е. для любого ключа $p_K(k) = \frac{1}{|K|}$.

По сути, теорема описывает шифры табличного гаммирования со случайными равновероятными ключами.

Практическая стойкость шифров. Центровым понятием в практической стойкости по Шеннону является рабочая характеристика шифра, представляющая собой средний объем работы $W(N)$, необходимый для определения ключа по криптограмме, состоящей из N букв, причем $N > L_0$ (объем перехвата перевалил за расстояние единственности), измеренный в удобных элементарных операциях.

Ценность большинства данных со временем снижается, поэтому важно, чтобы рабочая характеристика шифра превышала по стоимости защищаемую информацию.

Сложность взлома алгоритмов классифицируется по категориям:

- *Полное вскрытие.* Криптоаналитик находит ключ k , такой, что $D_k(x) = y$.
- *Глобальная дедукция.* Криптоаналитик находит альтернативный алгоритм A , эквивалентный $D_k(x)$ без знания k .
- *Случайная (или частичная) дедукция.* Криптоаналитик находит (крадет) открытый текст для перехваченного зашифрованного сообщения.
- *Информационная дедукция.* Криптоаналитик добывает некоторую информацию о ключе или открытом тексте. Такой

информацией могут быть несколько битов ключа, сведения о форме открытого текста и пр.

Алгоритм *безусловно стоек*, если восстановление невозможно при любом объеме шифртекста, полученного криптоаналитиком. На поверку безусловно стойки только одноразовые блокноты. Все остальные криптосистемы теоретически можно вскрыть методом *грубой силы* (прямой подбор ключа, лобовая атака).

Вычислительная стойкость по отношению к лобовой атаке зависит от длины ключа.

Вопросами вычислительной стойкости по отношению к другим методам криптоанализа (вероятностный, линейный, дифференциальный и пр.) занимается теория сложности вычислений.

Термодинамические ограничения. Идеальный компьютер затрачивает на каждую установку и сброс бита $4,4 \cdot 10^{-16}$ эрг энергии. Энергия, излучаемая солнцем за год равна $1,21 \cdot 10^{41}$ эрг. Этого достаточно, чтобы совершить $2,7 \cdot 10^{56}$ перемен бита, чего хватает для пробега 187-разрядным счетчиком всех значений.

Лобовое вскрытие 256-битового ключа невозможно пока компьютеры построены из обычной материи и работают в обычном пространстве.

Таблица 1.

Оценки времени лобового вскрытия для симметричных шифров.

Стоимость	Длина ключа, бит.					
	40	56	64	80	112	128
\$100 000	2 сек.	35 ч	1 год	70 000 лет	10^{14} лет	10^{19} лет
\$1 млн	0,2 сек.	3,5 ч	37 дней	7000 лет	10^{13} лет	10^{18} лет
\$10 млн	0,02 сек.	21 мин	4 дня	700 лет	10^{12} лет	10^{17} лет
\$100 млн	2 мсек.	2 мин	9 ч	70 лет	10^{11} лет	10^{16} лет
\$1 млрд	0,2 мсек.	13 сек.	31 мин	7 лет	10^{10} лет	10^{15} лет
\$10 млрд	0,02 мсек.	1 сек.	5,4 мин	245 дней	10^9 лет	10^{14} лет
\$100 млрд	2 мсек.	0,1 сек.	32 сек.	24 дня	10^8 лет	10^{13} лет
\$1 трлн	0,2 мсек.	0,01 сек.	3 сек.	2,4 дня	10^7 лет	10^{12} лет
\$10 трлн	0,02 мсек.	1 мсек.	0,3 сек.	6 ч	10^6 лет	10^{11} лет

Таблица 2.

Оценки среднего времени лобового вскрытия в «Китайской лотерее»

Страна	Население	Число телевизоров, радиоприемников	Время взлома	
			56 бит	64 бита
Китай	1 190 431 000	257 000 000	280 сек.	20 ч
США	260 714 000	739 000 000	97 сек.	6,9 ч
Ирак	19 890 000	4 730 000	4,2 ч	44 дня
Израиль	5 051 000	3 640 000	5,5 ч	58 дней
Вайоминг	470 000	1 330 000	15 ч	160 дней
Невада	6100	17 300	48 дней	34 года

Таблица 3.

Длины симметричных и открытых ключей, равных по устойчивости к лобовому вскрытию.

Длина симметричного ключа (бит)	Длина открытого ключа (бит)
56	384
64	512
80	768
112	1792
128	2304

Таблица 4.

Расстояние единственности по симметричному ключу

Длина ключа (бит)	Расстояние единственности (символов)
40	5,9
56	8,2
64	9,4
80	11,8
128	18,8
256	37,6

2.3. Имитостойкость и помехоустойчивость шифров

Имитостойкость шифров. Имитация и подмена сообщения. Помимо пассивных действий со стороны противника возможны активные действия, состоящие в попытках подмены или имитации сообщения. Если передается шифрованное сообщение $y \in Y$ (полученное из открытого текста $x \in X$ на ключе $k \in K$), то противник может заменить его на y' , отличный от y . При этом он будет рассчитывать на то, что на действующем ключе k новая криптограмма при расшифровании будет воспринята как некий осмысленный открытый текст x' , отличный от x , чем больше вероятность этого события, тем успешнее будет попытка имитации.

Имитостойкость шифра определим как его способность противостоять попыткам противника по имитации или подмене. Естественной мерой имитостойкости шифра служит вероятность соответствующего события:

$D_k(y') \in X$ – для попытки имитации сообщения;

$(D_k(y') \in X) \wedge (y' \neq y)$ – для попытки подмены сообщения.

В соответствии с этим введем следующие обозначения: $p(Dk(y') \in X)$, которые назовем соответственно вероятностью имитации и вероятностью подмены. Полагая, что противник выберет ту попытку, которая с большей вероятностью приводит к успеху, вводят также вероятность навязывания формулой $p_i = \max\{p_{ei}, p_{im}\}$.

Для шифров с равновероятными ключами можно получить общие оценки введенных вероятностей.

Утверждение 1. Для шифра Σ_B с равновероятными ключами имеет место достижимая оценка $p_{ei} \geq \frac{|X|}{|Y|}$. Для эндоморфного шифра с равновероятной гаммой $p_{im} = 1$.

Это неравенство поясняет широко используемый для имитозащиты способ введения избыточности в передаваемое сообщение, например дополнительных «добавок» к передаваемому сообщению типа аутентификаторов или имитовставок.

Утверждение 2. Для шифра Σ_B с равновероятными ключами имеет место достижимая оценка

$$p_{im} \geq \frac{|X|}{|Y|}.$$

Определим совершенную имитостойкость (теоретически лучшую защиту от имитации или подмены), достижимую при данной величине $|Y|$ множества допустимых криптограмм и при произвольном распределении $P(K)$ на множестве ключей. Для этого вводится понятие *граница Симмонса*.

Обозначим через $I(Y, K)$ взаимную информацию между Y и K , т. е. величину, определяемую формулой $I(Y, K) = H(Y) - H(Y / K)$.

Утверждение 3. Имеет место достижимая оценка

$$I(Y, K) \geq H(Y) - H(Y / K)$$

Равенство, определяемое как совершенная имитостойкость, достигается при одновременном выполнении двух условий:

1. Вероятность $p(y)$ того, что y окажется допустимой криптограммой, не зависит от y .

2. Для каждой криптограммы $y \in Y$ вероятность $p(y / k)$ одинакова при всех k , для которых $D_k(y) \in X$.

Следует отметить, что даже при совершенной имитостойкости вероятность навязывания мала лишь при большой величине $I(Y, K)$, т. е. в том случае, когда криптограмма дает значительную информацию о ключе. Информация, которую дает Y относительно K , есть мера того, в какой степени ключ используется для обеспечения имитостойкости.

Способы обеспечения имитостойкости. Основной причиной отсутствия какой-либо имитостойкости шифра гаммирования является то, что множество возможных открытых текстов длиной l совпадает с множеством всех слов длиной l в алфавите шифр-величин. Для обеспечения имитозащиты эндоморфных шифров в открытый текст намеренно вводится избыточная информация. Это делается для выделения множества открытых текстов так, чтобы соответствующая структура легко распознавалась, но не могла быть воспроизведена оппонентом без знания некоторого секрета, которым обычно является ключ зашифрования.

Например, к каждому сообщению перед зашифрованием можно добавить контрольную сумму вычисляемую с помощью известной функции F . Отправитель сообщения вычисляет значение F от открытого текста x , присоединяет это значение к x и шифрует полученную комбинацию. Получатель расшифровывает поступивший массив, рассматривая результат как сообщение с

присоединенной контрольной суммой. После чего он применяет к полученному сообщению функцию F , чтобы воспроизвести контрольную сумму. Если она равна контрольной сумме, поступившей с сообщением, сообщение признается подлинным (или аутентичным). Маловероятно, чтобы случайная последовательность знаков могла быть признана аутентичной.

Более надежный способ используется в военном протоколе аутентификации, принятом в США. Отправитель и получатель сообщения имеют опечатанный пакет со случайной последовательностью символов, вырабатываемой компетентным органом. Каждый из участников связи отвечает за защиту своего опечатанного пакета и имеет инструкцию не вскрывать его, пока не потребуются аутентификация сообщения. Кроме того, отправитель и получатель имеют общий секретный ключ. При аутентификации сообщения отправитель вскрывает пакет, дополняет сообщение символами этой секретной последовательности, а затем шифрует полученное сообщение, используя секретный ключ. Для шифрования обычно используется симметричный шифр. Получатель после расшифрования сообщения (с помощью своей копии ключа) вскрывает пакет и производит аутентификацию.

Сообщение интерпретируется как аутентичное только тогда, когда при расшифровании будут получены символы секретной последовательности. Если используется стойкое шифрование, то оппоненту (который не знает ключа) при осуществлении активной атаки не остается ничего другого, как случайным образом выбирать шифртекст в надежде, что он будет принят получателем как аутентичный. Если секретная последовательность состоит, например, из r битов, то вероятность того, что при расшифровании случайно выбранный оппонентом зашифрованный текст даст сообщение, заканчивающееся неизвестной ему, но правильной последовательностью, будет составлять величину 2^{-r} .

Коды аутентификации. Другой метод нашел распространение при аутентификации электронной передачи фондов в Федеральной резервной системе США. Подобные передачи должны быть аутентифицированы с использованием процедуры, которая фактически реализована в криптографическом алгоритме, определенном в стандарте шифрования данных США (ранее DES, теперь – AES). Аутентификатор генерируется в режиме,

называемом шифровании со сцеплением блоков. В этом режиме сообщение разбивается на 64-битные блоки – $M = M_1M_2...M_n$, которые последовательно шифруются следующим образом. Блоки шифртекста $C_1, C_2, ..$ вырабатываются по рекуррентной формуле

$$C_i = E_k (C_{i-1} \oplus M_i),$$

при этом вектор C_0 полагается равным начальному вектору IV (Initial Vector). Начальный вектор меняется ежедневно и хранится в секрете. Схематично этот режим изображен на рис. 11.

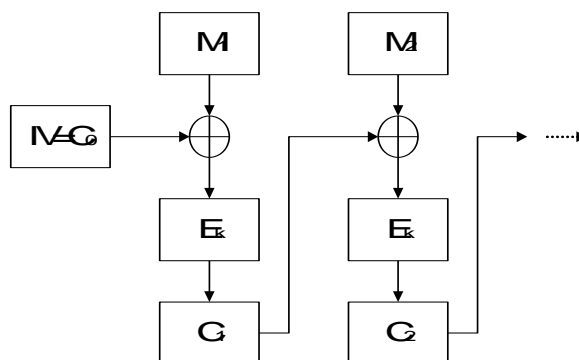


Рис.11. Режим выработки имитовставки.

Процедура повторяется до тех пор, пока не будут обработаны все блоки текста. Последний блок шифртекста – C_n является функцией секретного ключа, начального вектора и каждого бита текста, независимо от его длины. Этот блок называют кодом аутентичности сообщения (КАС) и добавляют к сообщению в качестве аутентификатора. Само расширенное аутентификатором сообщение передается обычно открытым, хотя может быть и зашифрованным, если требуется секретность.

Любой владелец секретного ключа и начального вектора может проверить правильность такого аутентификатора. Нужно лишь повторить ту же процедуру шифрования. Оппонент, однако, не может ни осуществить генерацию аутентификатора, который бы воспринимался получателем как подлинный, для добавления его к ложному сообщению, ни отделить аутентификатор от истинного сообщения для использования его с измененным или ложным сообщением. В обоих случаях вероятность того, что ложное сообщение будет интерпретироваться как подлинное, равна вероятности «угадывания» аутентификатора, т. е. 2^{-64} . Предложенный способ делает аутентификатор сложной функцией информации, которую он аутентифицирует. В таком случае

подмножество допустимых сообщений состоит из тех пар «текст – КАС», которые могут успешно пройти проверку на соответствие аутентификатора тексту с использованием ключа.

Оба способа имитозащиты передаваемых сообщений, связанные с введением избыточности, моделируются так называемыми кодами аутентификации (или А-кодами). В случае если расширенное аутентификатором сообщение шифруется, говорят об А-кодах с секретностью. Если расширенное сообщение передается в открытом виде, говорят об А-кодах без секретности.

Помехостойкость шифров. Помимо целенаправленных искажений передаваемой шифрованной информации возможны также искажения, происходящие за счет наличия помех в канале связи. Такие помехи могут привести к искажениям и даже потере некоторых знаков используемого алфавита. Если искаженный знак не является знаком используемого алфавита, то на приеме факт искажения легко установить. В противном случае факт искажения может быть установлен лишь при расшифровании, когда искажение в шифртексте ведет к потере части или даже всего открытого текста. Так же проявляется и потеря знаков шифртекста.

Прежде всего интересен вопрос о свойствах самого шифра, позволяющих не распространять искажений при расшифровании. Ограничимся только рассмотрением эндоморфных ($X=Y$) шифров и искажений двух типов:

1. Замена знаков знаками того же алфавита.
2. Потеря знаков или появление дополнительных знаков того же алфавита.

Шифры, не распространяющие искажений типа «замена знаков». Будем рассматривать шифры, описываемые алгебраической моделью $\Sigma_A = (X, K, Y, E, D)$, в которой

$$X = Y = \bigcup_{i=1}^L A^i, \text{ причем для любых } x \in X \text{ и } k \in K \text{ длина } y = E_k(x)$$

совпадает с длиной x .

Мерой значительности последствий искажений типа замены знаков является метрика на множестве сообщений $X=Y$. Простейшей является метрика Хэмминга μ , определяемая формулой

$$\mu(x, y) = \sum_{i=1}^{\lambda} \alpha_{x_i, y_i},$$

$$x = (x_1, \dots, x_{\lambda}), y = (y_1, \dots, y_{\lambda}) \in X,$$

$$\alpha_{x_i, y_i} = \begin{cases} 1 & x_i \neq y_i \\ 0 & x_i = y_i \end{cases}$$

Так как для эндоморфного шифра каждое правило зашифрования E_k представляет собой биекцию $E_k : X \rightarrow X$, то будем пользоваться *подстановочной моделью шифра* – $\Sigma_{\Pi} = (X, E)$, в которой множество $E = \{e_k : k \in K\}$ рассматривается как множество подстановок $e : X \rightarrow X, e \in E$.

Шифр $\Sigma_{\Pi} = (X, E)$ не распространяет искажений типа замены знаков и является помехостойким, если для любых $x, y \in A^{\lambda}$ и любого $e \in E$ выполняется неравенство $\mu(e^{-1}x, e^{-1}y) \leq \mu(x, y)$.

Подстановки $e \in E$, удовлетворяющие предыдущему равенству, называются *изометриями на X* .

Теорема А. А. Маркова. Биекция $e \in E$ является изометрией на X тогда и только тогда, когда $e = R \cdot \tilde{I}_{j_1, \dots, j_{\lambda}}$ для подходящих преобразований множества X :

$$\tilde{I}_{j_1, \dots, j_{\lambda}}(a_1, \dots, a_{\lambda}) = (a_{j_1}, \dots, a_{j_{\lambda}}),$$

$$R(a_1, \dots, a_{\lambda}) = (R_1(a_1), \dots, R_{\lambda}(a_{\lambda})),$$

где $(j_1, \dots, j_{\lambda})$ – перестановка чисел $1, 2, \dots, \lambda$;

$R_i \in S(A)$ – некоторые фиксированные подстановки множества $A, a_i \in A, \lambda \in \overline{1, L}, i \in \overline{1, \lambda}$.

Согласно теореме Маркова, в классе эндоморфных шифров, не изменяющих длины сообщений, не распространяют искажения типа замены знаков, например шифры перестановки, поточные шифры однозначной замены, а также их композиции типа «шифр замены – шифр перестановки».

Шифры, не распространяющие искажений типа «пропуск – вставка знаков». Приведем теорему, рассматривающую подстановочную модель шифра.

Теорема: если $\Sigma_{\Pi} = (X, E)$ – шифр, не распространяющий искажений типа «пропуск – вставка», то для любого $e \in E$ либо $e = \pi_L$, либо $y = \pi_L \cdot f$ (при подходящем $\pi \in S(A)$), где π_L – отображение множества X в себя, определенное для любого $a = (a_1, \dots, a_{\lambda}) \in X$ формулой ~~$\pi_L(a) = (\pi(a_1), \dots, \pi(a_{\lambda}))$~~ (π – некоторая подстановка

множества A), а f – отображение множества X в себя, меняющее порядок следования букв любого слова на противоположный: $f(a_1, \dots, a_\lambda) = (a_\lambda, \dots, a_1)$.

Всякий шифр, не распространяющий искажений типа «пропуск – вставка знака», есть либо шифр простой замены либо произведение шифра простой замены и частного вида шифра перестановки, заключающейся в инверсной записи текста (справа налево).

Следовательно, все сложные шифры распространяют искажения типа «пропуск – вставка», и в данном случае борьба с такими искажениями криптографическими методами невозможна, необходимо применять иные способы повышения помехоустойчивости (введение избыточности – контрольные суммы и пр.).

Практические вопросы повышения надежности. Совместно с алгоритмом шифрования данных целесообразно использовать алгоритм сжатия по следующим причинам:

- криптоанализ опирается на избыточность шифртекста, а сжатие файла перед шифрованием эту избыточность снижает;
- шифрование занимает много времени, а потому сжатие файла до шифрования ускоряет весь процесс.

Важно запомнить, что сжатие должно выполняться именно до шифрования. (Сжимаемость шифртекста можно использовать как неплохой тест алгоритма шифрования. Если шифртекст можно сжать, значит, алгоритм не слишком надежен.)

Вырабатывать имитовставку можно до сжатия файла и сжимать вместе с ним с целью повышения стойкости шифра (имитовставка повышает избыточность текста) или же выполнять после шифрования с целью ускорения процесса аутентификации сообщения.

Помехоустойчивое кодирование, повышающее избыточность текста, наоборот, целесообразно применять после шифрования данных.

Последовательность применения имитовставки и кодирования приведена на рис.12.



Рис. 12. Практическое применение имитовставки в сочетании со сжимающим и помехоустойчивым кодированием.

Контрольные вопросы

1. Перечислите основные составляющие алгебраической модели шифра.
2. Для чего используются вероятностные модели шифров?
3. Для чего применяются модели открытых текстов?
4. Каковы критерии на открытый текст?
5. В чем различие между теоретической и практической криптостойкостью шифров?
6. В чем различие между имитацией и подменой сообщения?
7. Основные способы обеспечения имитостойкости?
8. Какие бывают виды искажений при передаче сообщения?

3. ПРИНЦИПЫ ПОСТРОЕНИЯ СИММЕТРИЧНЫХ КРИПТОГРАФИЧЕСКИХ АЛГОРИТМОВ

3.1. Виды симметричных шифров. Особенности программной и аппаратной реализации

Работа симметричных шифров включает в себя два преобразования:

$$C = E_k(m) \text{ и } m = D_k(C),$$

где m – открытый текст;

E – шифрующая функция;

D – расшифровывающая функция;

k – секретный ключ;

C – шифртекст;

Функции, зашифрования/расшифрования общеизвестны и стойкость всей системы зависит только от секретности ключа k . Этот принцип называют также принципом Керкгоффса.

Современные симметричные алгоритмы шифрования делятся на блочные и поточные, (табл. 5).

Таблица 5.

Особенности блочных и поточных шифров.

Блочные шифры	Поточные шифры
Работают во всех режимах шифрования	Поддерживают не все режимы шифрования
Тяжелы для математического анализа	Простое проектирование и математический анализ.
Легко реализуются как аппаратно, так и программно	Практически непригодны к программной реализации для битового потока.
Имеют не очень высокое быстродействие	Имеют высокое быстродействие
Стойкость шифра зависит от длины ключа, количества раундов шифрования, размера блока и рассеивающих и перемешивающих характеристик алгоритма.	Стойкость шифра зависит от статистических характеристик и периода переполнения генератора ключевой последовательности

Блочные шифры оперируют с данными путем фиксированного преобразования крупных блоков открытого текста, а поточные – путем динамического преобразования отдельных цифр открытого текста.

При применении различных режимов шифрования блочные шифры могут быть также реализованы как поточные, и наоборот.

Основой блочного шифра является *алгоритм преобразования*, основой поточного – *генератор ключевой последовательности*.

3.2. Принципы построения блочных шифров

Базовые шифрующие преобразования. Алфавитом, на котором действует блочный шифр, является множество двоичных векторов-блоков открытого текста одинаковой длины (64, 128 и т.д.). Так как с увеличением мощности алфавита энтропия на один знак также увеличивается, криптографическая стойкость алгоритма возрастает. Блочные шифры реализуются путем многократного применения к блокам открытого текста некоторых базовых преобразований.

Требования к базовым преобразованиям:

- 1) возможность простой реализации (как аппаратной, так и программной);
- 2) способность давать при небольшом числе итераций аналитически сложные преобразования;
- 3) соответствие критериям рассеивания и перемешивания;
- 4) соответствие критерию лавинного эффекта.

Принцип *лавинного эффекта*: – применение шифрующего преобразования к наборам аргументов, отличающихся в незначительном числе позиций, должно приводить к существенному изменению результата.

Принцип *перемешивания* (diffusion): криптографические преобразования должны обеспечивать максимальное усложнение статистической и аналитической взаимосвязи между открытым и шифртекстом.

Принцип *рассеивания* (confusion): криптографические преобразования должны обеспечивать максимальное усложнение статистической и аналитической взаимосвязи между шифртекстом и ключом.

Перемешивающие преобразования – сложные в криптографическом отношении локальные преобразования над отдельными частями шифрующих блоков.

Рассеивающие преобразования – простые преобразования, переставляющие между собой части блоков текста.

Применяемые математические преобразования (операции):

- *Перестановки* – смена местами битовых значений разрядов подблоков на основе фиксированных подстановок.

- *Битовые сдвиги* – сдвиг вправо ($X \gg Y$) и сдвиг влево ($X \ll Y$) битовых значений разрядов подблока X (соответствует умножению на 2 и целочисленному делению на 2) на величину, определяемую значением подблока Y .

- *Циклические битовые сдвиги* – циклический сдвиг вправо ($X \ggg Y$) и циклический сдвиг влево ($X \lll Y$) битовых значений разрядов подблока X (ключа, шифртекста) на величину, определяемую значением подблока Y .

- *S-подстановки* – подстановки значений на основе матриц подстановки, где элементы битового блока используются в качестве параметров (координат элементов) матрицы подстановки. S-подстановки могут быть сужающими, расширяющими и однозначными в зависимости от размерного соотношения (в битах) входных и выходных подблоков данных.

- *Побитовая инверсия* – инверсия ($\sim X$) всех битовых регистров блока X .

- *Побитовое сложение по модулю 2* – операция XOR ($X \oplus Y$) над всеми битовыми регистрами подблоков X и Y .

- *Сложение по модулю 2^n* (n – длина блока в битах) – двоичное сложение ($X \boxplus Y$) подблоков с отбрасыванием значений старших регистров переноса.

- *Умножение по модулю 2^{n+1}* – двоичное умножение ($X \odot Y$) с отбрасыванием значений старших регистров переноса, причем нулевое значение подблока принимается равным $0 = 2^n$.

- *Сложные операции* – нелинейные подстановки, логарифмирование и экспоненцирование в конечном поле и пр.

Сеть Фейстеля. Она служит структурной основой построения большинства современных блочных криптоалгоритмов и является

по сути методом смешивания текущей части шифруемого блока с результатом некоторой функции, вычисленной от другой независимой части того же блока. Эта методика получила широкое распространение, поскольку обеспечивает выполнение требования о многократном использовании ключа и материала исходного блока информации.

Действие, состоящее из однократного вычисления образующей функции и последующего наложения ее результата на другую ветвь с обменом их местами, называется циклом или раундом (англ. *round*) сети Фейстеля. Оптимальное число раундов N – от 8 до 32. Важно то, что увеличение количества раундов значительно увеличивает криптостойкость любого блочного шифра к криптоанализу.

Классическая схема одного раунда сети Фейстеля показана на рис. 13.

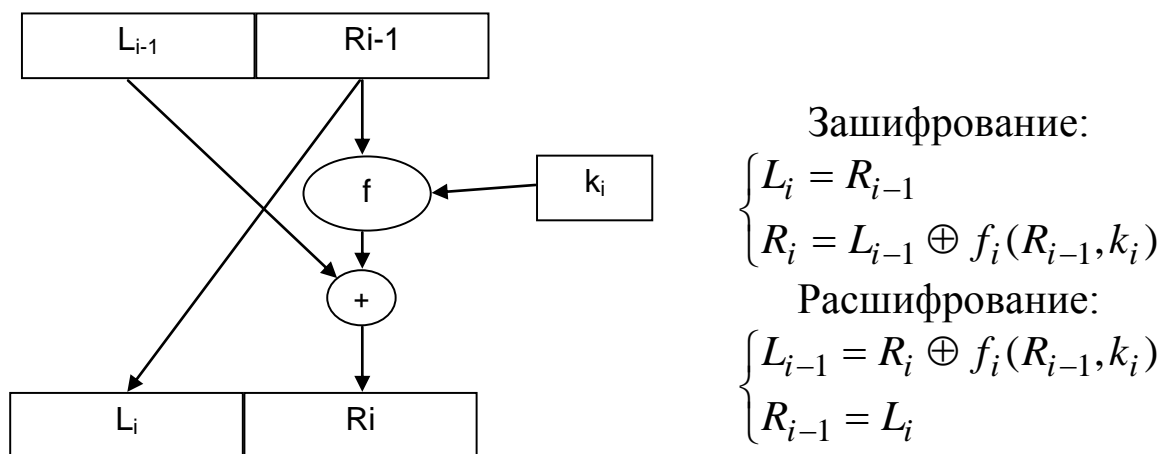


Рис.13. Сеть Фейстеля

Независимые потоки информации, порожденные из исходного блока, называются ветвями сети. В классической схеме их две. Функция f называется образующей.

Данная схема является обратимой. Сеть Фейстеля обладает тем свойством, что даже если в качестве образующей функции f будет использовано необратимое преобразование, то и в этом случае вся цепочка будет восстанавливаема. Это происходит вследствие того, что для обратного преобразования сети Фейстеля не нужно вычислять функцию f^{-1} .

Благодаря применению финального преобразования (перестановка левой и правой частей блока) сеть Фейстеля симметрична. Использование операции XOR, обратимой своим же повтором, и инверсия последнего обмена ветвей делают возможным раскодирование блока той же сетью Фейстеля, но с инверсным порядком применения подключей k_i . Для обратимости сети Фейстеля не имеет значения, является ли число раундов четным или нечетным. В большинстве реализаций схемы, в которых оба вышеперечисленных условия (операция XOR и уничтожение последнего обмена) сохранены, прямое и обратное преобразования производятся одной и той же процедурой.

В настоящее время чаще применяют модификацию сети Фейстеля для большего числа ветвей. Это в первую очередь связано с тем, что при больших размерах кодируемых блоков (128 и более бит) становится неудобно работать с математическими функциями по модулю 64 и выше. Как известно, основные единицы информации, обрабатываемые процессорами на сегодняшний день, – это байт и двойное машинное слово 32-бита. Поэтому все чаще и чаще в блочных криптоалгоритмах встречается сеть Фейстеля с четырьмя ветвями (рис. 14). Самый простой принцип ее модификации изображен на рис. 14, *а*). Для более быстрого перемешивания информации между ветвями (а это основная проблема сети Фейстеля с большим количеством ветвей) применяются две модифицированные схемы, называемые «type – 2» и «type – 3» (рис. 14, *б-в*).

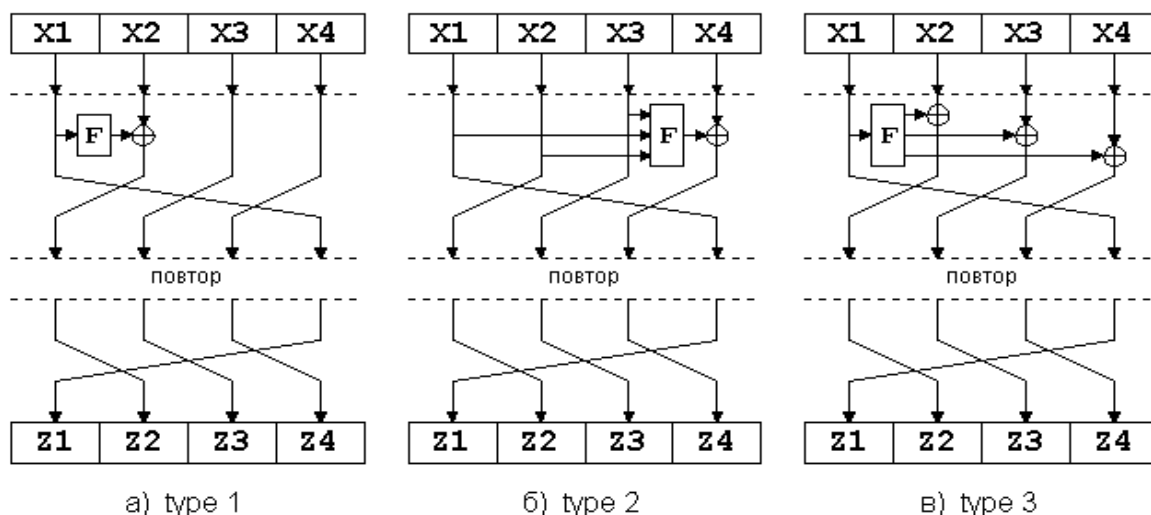


Рис.14. Сеть Фейстеля на четыре ветви.

3.3. Современные блочные криптоалгоритмы

Основные параметры блочных криптоалгоритмов.

Параметрами, характеризующими современные блочные криптографические алгоритмы, являются размер блока шифртекста (определяющий энтропию алфавита сообщения), размер ключевого пространства (определяющий энтропию криптосистемы), количество раундов, структурная схема криптоалгоритма, применяемые криптографические преобразования (математические операции) (табл. 6).

Алгоритм DES. Это 64-битовый блочный алгоритм с 56-битовым ключом. Он построен на классической сети Фейстеля и выполняется в течение 16 раундов (рис.15). Кроме того, он имеет начальную и завершающую перестановки.

При подстановке через *S*-блок на вход подается 48-битовый блок, разбитый на восемь 6-битовых подблоков, соответствующих каждой своей *S*-матрице. Первый и последний биты подблока объединяются в 2-битовое число, определяющее строку *S*-матрицы, а средние четыре бита объединяются в 4-битовое число, определяющее столбец. На выход подается 4-битовое значение соответствующего элемента *S*-матрицы.

Таблица 6.

Основные параметры распространенных блочных
криптоалгоритмов.

№ n/ n	Название	Размер блока, бит	Размер ключа, бит	Кол-во раундов	Основа алгоритма	Применяемые операции
1	2	3	4	5	6	7
1	DES	64	56	16	Сеть Фейстеля (две ветви)	XOR, S-подстановки, перестановки
2	3DES	64	112,168	48	Алгоритм DES	-'
3	IDEA	64	128	8	Мультипликативно-аддитивная структура (четыре ветви)	XOR, сложение по модулю 2^{16} , умножение по модулю $2^{16}+1$
4	Blowfish	64	32-448	16	Модифицированная сеть Фейстеля (две ветви)	XOR, сложение по модулю 2^{32} , S-подстановки
5	RC5	32, 64, 128	8-2040	1?255	Оригинальная структура (две ветви)	XOR, сложение/вычитание, циклический сдвиг
6	CAST	64,128	40-256	3-32	Сеть Фейстеля с переменной F (две ветви)	XOR, сложение/вычитание, циклический сдвиг, S-подстановки
7	RC2	64	8-1024	18	Оригинальная структура (четыре ветви)	XOR, сложение, циклический сдвиг, побитовое «И» побитовое дополнение
8	ГОСТ 28147-89	64	256	32	Сеть Фейстеля (две ветви)	XOR, сложение, циклический сдвиг, S-подстановки
9	TEA	64	128	32	Несимметричная сеть Фейстеля (две ветви)	XOR, сложение, битовый сдвиг
10	MARS	128	128,192, 256	16+16	Сеть Фейстеля (четыре ветви)	XOR, сложение, циклический сдвиг, S-подстановки
11	RC6	128	128,192, 256	20	Сеть Фейстеля (четыре ветви)	XOR, сложение, циклический сдвиг, преобразование T
12	Serpent	128	128,192, 256	32	Сеть Фейстеля (четыре ветви)	XOR, битовый сдвиг, циклический сдвиг, S-подстановки

1	2	3	4	5	6	7
13	Twofish	128	128,192, 256	?	Алгоритм Blowfish, сеть Фейстеля (четыре ветви), преобразование Адамара	XOR, сложение, циклический сдвиг, S-подстановки
14	Rijndael (AES)	128	128,192, 256	10-14	Табличные преобразования подблоков	XOR, S-подстановки, сдвиг строк, перемешивание в столбцах
15	Base King	192	?	?	Алгоритм 3-WAY	?
16	SAFER	64	64,128	?	Итеративные раунды, псевдопреобразования Адамара	Логарифмирование и экспоненцирование в конечном поле, XOR, сложение
17	3-WAY	96	96	?	Оригинальная процедура	Линейная подстановка на основе сдвигов и XOR, перестановки, нелинейная подстановка 3-бит блоков

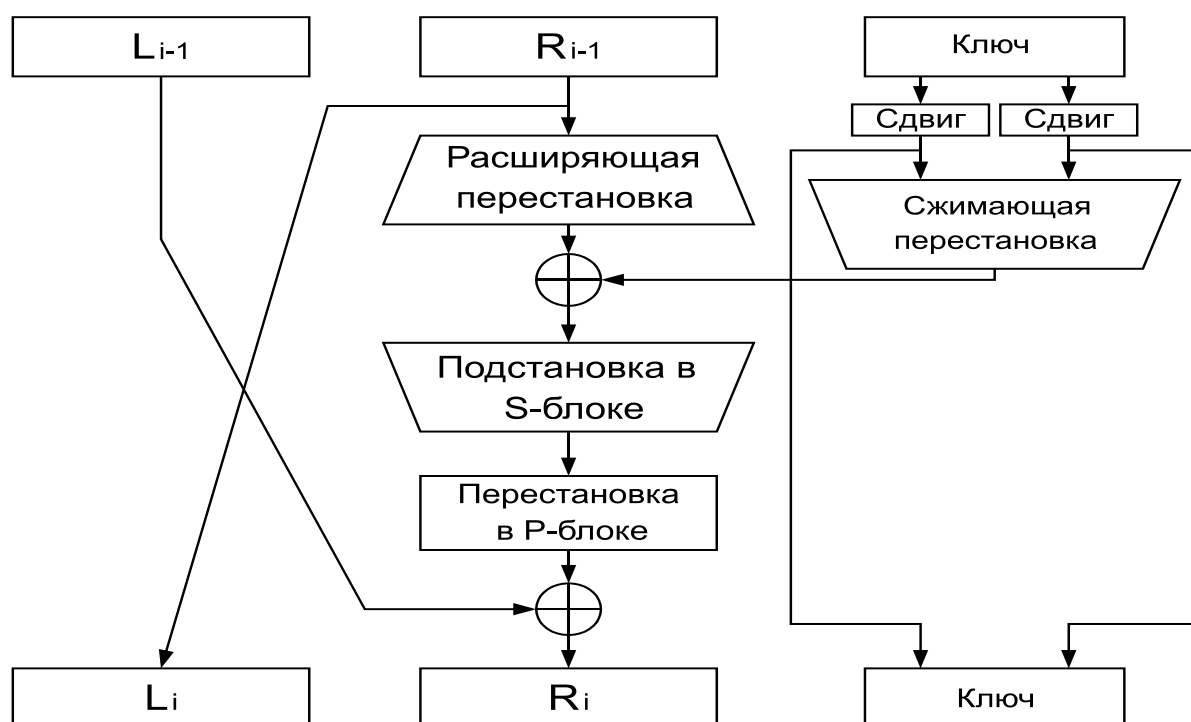
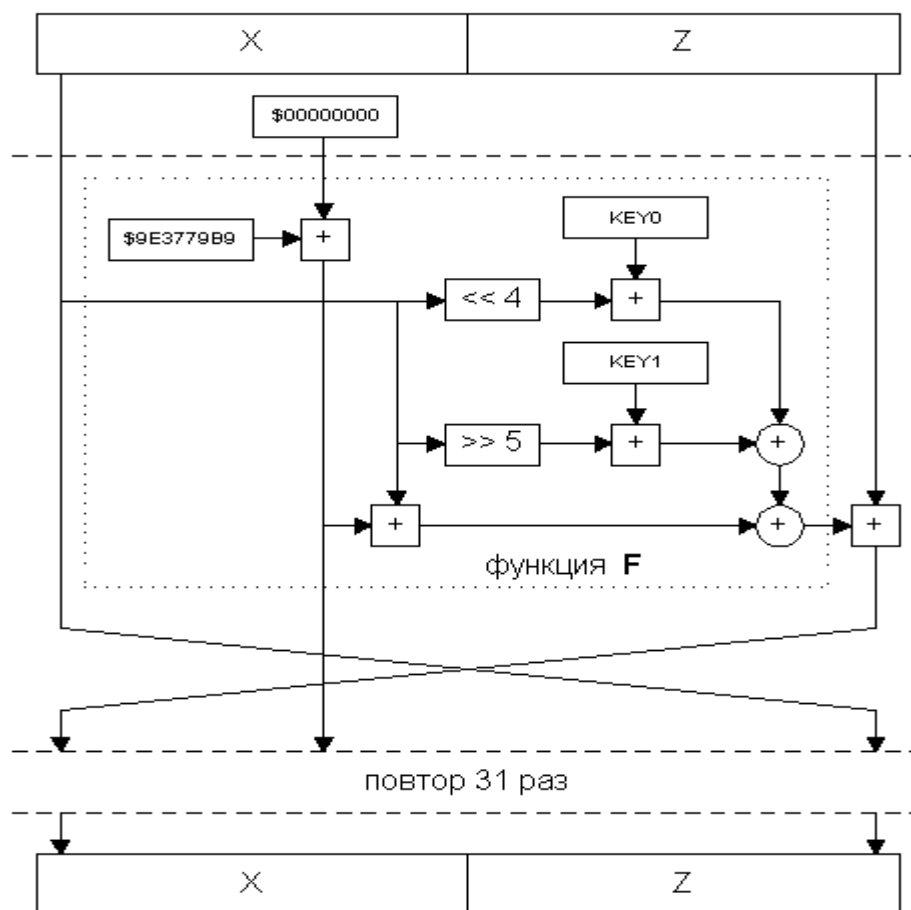


Рис.15. Раунд шифрования DES

Алгоритм DES являлся коммерческим стандартом шифрования США до 2000 года, когда ему на смену пришел AES. В

настоящее время алгоритм применяется в варианте *3DES* – каждый блок шифруется трижды на разных ключах. Фактическая длина ключа составила 168 бит.



Параметры алгоритма: размер блока – 64 бита; длина ключа – 128 бит.

Образующая функция F обратима.

Ниже приведен код криптоалгоритма на языке программирования PASCAL:

```
type TLong2=array[0.. 1] of longint;
    TLong2x2=array[0.. 1] of TLong2;
const Delta=$9E3779B9;
var key:TLong2x2;
procedure EnCryptRouting(var data);
var y,z,sum:longint; a:byte;
begin
y:=TLong2 (data) [0];z:=TLong2 (data) [1];sum:=0;
for a:=0 to 31 do
begin
inc(sum,Delta);
inc(y,((z shl 4)+key[0,0]) xor (z+sum) xor
((z shr 5)+key[0,1]));
inc(z,((y shl 4)+key[1,0]) xor (y+sum) xor
((y shr 5)+key[1,1]));
end;
TLong2 (data) [0]:=y;TLong2 (data) [1]:=z
end;
```

Отличительной чертой криптоалгоритма TEA является его размер. Простота операций, отсутствие табличных подстановок и оптимизация под 32-разрядную архитектуру процессоров позволяют реализовать его на языке ASSEMBLER в предельно малом объеме кода. Недостатком алгоритма является некоторая медлительность, вызванная необходимостью повторять цикл Фейстеля 32 раза (это нужно для тщательного «перемешивания данных» из-за отсутствия табличных подстановок).

Международный алгоритм IDEA. Данный алгоритм получает на входе 64-битовый блок открытого текста (в процессе шифрования он разбивается на четыре 16-битовых подблока) и 128-битовый ключ, из которого генерируется 52 16-битовых подключа шифрования $Z_1..Z_{52}$ по шесть подключей на каждый из восьми раундов и четыре подключа для выходного преобразования. На выходе генерируется 64-битовый блок шифрованного текста. Алгоритм расшифрования IDEA полностью повторяет структуру алгоритма шифрования: в качестве входных данных использует 64-битовый блок шифрованного текста, и тот же 128-битовый ключ, из которого генерируются 53 16-битовых подключей расшифрования

$U_1..U_{52}$. В результате работы алгоритм расшифрования должен генерировать 64-битовый блок открытого текста.

Алгоритм шифрования представляет собой сложную комбинацию смешанного использования трех различных операций. Каждая из операций предполагает два 16-битовых входных значения, в результате обработки которых получается одно 16-битовое выходное значение. Вот эти операции:

1. Побитовое исключающее «ИЛИ» (XOR, сумма по модулю 2), обозначаемое символом \oplus .

2. Сложение целых чисел по модулю 2^{16} (по модулю 65536) с входными и выходными значениями, рассматриваемыми как

16-битовые целые числа без знака. Эта операция обозначается символом \boxplus .

3. Умножение целых чисел по модулю $2^{16} + 1$ (по модулю 65537) с входными и выходными значениями, рассматриваемыми как 2-битовые целые числа без знака, за исключением блока состоящего из одних нулей, который интерпретируется как 2^{16} . Эта операция обозначается символом \odot .

Упрощенно операция реализуется следующим отображением:

$$ab \bmod (2^{16} + 1) =$$

$$= \begin{cases} (ab \bmod 2^{16}) - (ab \div 2^{16}), & \text{при } (ab \bmod 2^{16}) \geq (ab \div 2^{16}) \\ (ab \bmod 2^{16}) - (ab \div 2^{16}) + 2^{16} + 1, & \text{при } (ab \bmod 2^{16}) \leq (ab \div 2^{16}) \end{cases}$$

$ab \bmod 2^{16}$ соответствует 16 наименее значимым битам ab , а $(ab \div 2^{16})$ является простым сдвигом ab вправо на 16 битов. Особенность данной операции состоит в том, что она образует группу (каждый элемент обратим, включая $0 \equiv 2^{16}$). Эти операции являются несогласованными:

1) Никакие две из этих трех операций не подчиняются дистрибутивному закону.

2) Никакие две из этих трех операций не подчиняются ассоциативному закону.

Вычисление подключей IDEA. В алгоритме IDEA используется 128-битовый ключ Z , который должен быть как у отправителя, так и у получателя сообщения. Из этого до начала шифрования или расшифрования генерируются 52 16-битовых подключа. При этом применяется следующая схема. Первые восемь подключей,

обозначенные $Z_1..Z_8$, образуются непосредственно из ключа: Z_1 равен первым 16 битам ключа, Z_2 – следующим 16 и т.д.

Затем к ключу шифрования применяется циклический сдвиг влево на 25 битов, и создается восемь следующих подключей. Эта процедура повторяется до тех пор, пока не будут получены все 52 подключа.

Подключи расшифрования $U_1..U_{52}$ получаются из ключей шифрования по схеме,

где Z_i^{-1} – мультипликативное обращение Z_i по модулю $2^{16}+1$ (простое число) т. е.

$$Z_i \odot Z_i^{-1} = 0000000000000001.$$

$-Z_i$ – аддитивное обращение Z_i по модулю 2^{16} , т. е.

$$Z_i \boxplus (-Z_i) = 0000000000000000.$$

Первые четыре подключа для i -го раунда дешифрования получаются из первых четырех подключей $(10 - i)$ -го раунда шифрования, если 9-м раундом считать выходное преобразование. Первый и четвертый подключи дешифрования равны мультипликативным обращениям по модулю $2^{16} + 1$ первого и четвертого подключей шифрования соответственно. Для раундов со 2-го по 8-й второй и третий подключи дешифрования равны аддитивным обращениям по модулю 2^{16} третьего и второго подключей шифрования соответственно. Для раундов 1 и 9 второй и третий подключи дешифрования равны аддитивным обращениям по модулю 2^{16} второго и третьего подключей шифрования соответственно.

Для первых восьми раундов два последних подключа i -го раунда дешифрования равны двум последним подключам $(9 - i)$ -го раунда шифрования.

Зашифрование и расшифрование в IDEA. Процесс шифрования предполагает два раунда и выходное преобразование. Один раунд состоит из преобразования входных блоков и субшифрования (рис. 17). Основой субшифрования является строительный блок алгоритма – мультипликативно-аддитивная (МА) структура.

Раунд начинается с преобразования, которое с помощью операции сложения и умножения связывает четыре входных подблока с четырьмя подключами. Это преобразование представлено серым прямоугольником вверху рисунка. Четыре

выходных блока этого преобразования связываются операцией XOR с целью получения двух 2-битовых блоков, которые затем подаются на вход структуры МА, представленной на рис. 17 нижним серым прямоугольником. Кроме того, структура МА получает на входе два подключа, а в результате обработки всех полученных данных на выходе этой структуры генерируется два 2-битовых значения.

Наконец, четыре блока, полученных на выходе первого преобразования, связываются с помощью операции XOR с двумя блоками, полученными на выходе структуры МА, и в результате имеется четыре выходных блока ($W_{11}..W_{14}$) данного раунда (рис. 18).

Выходное преобразование имеет структуру, подобную структуре той части предыдущего раунда шифрования, которая представлена на рис. 17 верхним серым прямоугольником. Единственное отличие в том, что второй и третий входные подблоки перед обработкой меняются местами. Фактически это означает отмену операции обмена, выполненной в конце второго раунда. Наличие этих лишних, по сути, перестановок обеспечивает возможность применить одну и ту же структуру как для шифрования, так и для расшифрования. Кроме того, в отличие от предыдущих раундов, на данной стадии используется не шесть подключей, а только четыре.

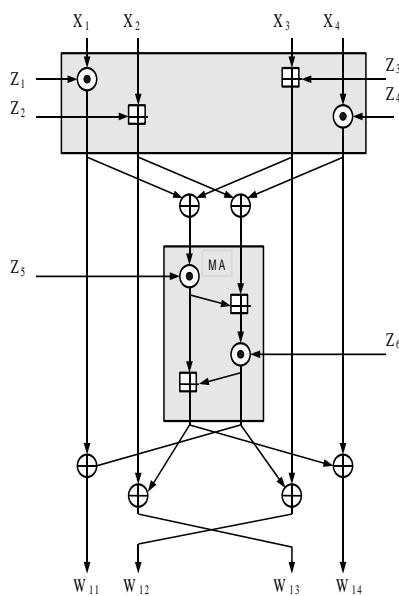


Рис.17. Раунд шифрования IDEA

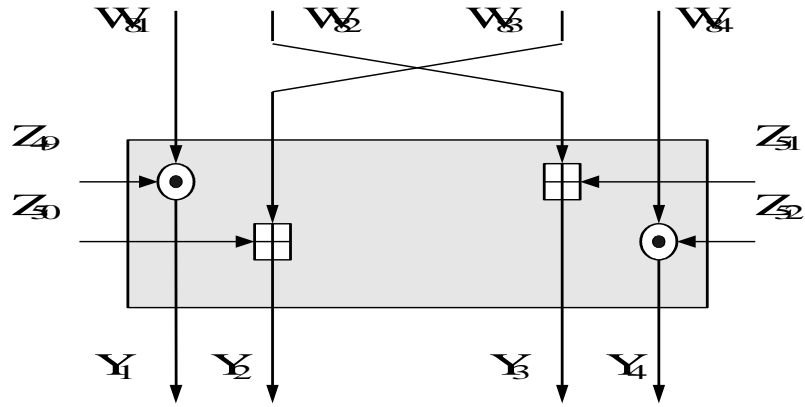


Рис.18. Выходное преобразование IDEA.

Подробно рассмотрим работу алгоритма. Выходы первого раунда расшифрования обозначим как $(V_{11}..V_{14})$, промежуточные выходы первого раунда зашифрования и расшифрования соответственно как $(I_{11}..I_{14})$ и $(J_{11}..J_{14})$.

При шифровании на выходе функции выходного преобразования имеем:

$$Y_1 = W_{81} \odot Z_{49}; Y_2 = W_{83} \boxplus Z_{50}; Y_3 = W_{82} \boxplus Z_{51}; Y_4 = W_{84} \odot Z_{52}.$$

При расшифровании на выходе первой подстадии первого раунда получаем:

$$J_{11} = Y_1 \odot U_1; J_{12} = Y_2 \boxplus U_2; J_{13} = Y_3 \boxplus U_3; J_{14} = Y_4 \odot U_4.$$

Заменив соответствующие значения эквивалентными получим:

$$J_{11} = Y_1 \odot Z_{49}^{-1} = W_{81} \odot Z_{49} \odot Z_{49}^{-1} = W_{81};$$

$$J_{12} = Y_2 \boxplus -Z_{50} = W_{83} \boxplus Z_{50} \boxplus -Z_{50} = W_{83};$$

$$J_{13} = Y_3 \boxplus -Z_{51} = W_{82} \boxplus Z_{51} \boxplus -Z_{51} = W_{82};$$

$$J_{14} = Y_4 \odot Z_{52}^{-1} = W_{84} \odot Z_{52} \odot Z_{52}^{-1} = W_{84}.$$

Таким образом, выходные значения первой подстадии первого раунда расшифрования совпадают с входными данными последней стадии шифрования, за исключением того, что второй и третий блоки оказываются переставленными. Следующие соотношения:

$$W_{81} = I_{81} \oplus MA_R(I_{81} \oplus I_{83}, I_{82} \oplus I_{84});$$

$$W_{82} = I_{83} \oplus MA_R(I_{81} \oplus I_{83}, I_{82} \oplus I_{84});$$

$$W_{83} = I_{82} \oplus MA_L(I_{81} \oplus I_{83}, I_{82} \oplus I_{84});$$

$$W_{84} = I_{84} \oplus MA_L(I_{81} \oplus I_{83}, I_{82} \oplus I_{84}),$$

где $MA_R(X, Y)$ обозначает правое выходное значение структуры МА для входных значений X и Y , а $MA_L(X, Y)$ – соответственно левое выходное значение.

Тогда

$$\begin{aligned}
V_{11} &= J_{11} \oplus MA_R(J_{11} \oplus J_{13}, J_{12} \oplus J_{14}) = W_{81} \oplus MA_R(W_{81} \oplus W_{82, 83} \oplus W_{84}) = \\
&= I_{81} \oplus MA_R(I_{81} \oplus I_{83}, I_{82} \oplus I_{84}) \oplus MA_R [I_{81} \oplus MA_R(I_{81} \oplus I_{83}, I_{82} \oplus I_{84}) \oplus \\
&\oplus I_{83} \oplus MA_R(I_{81} \oplus I_{83}, I_{82} \oplus I_{84}), \\
&I_{82} \oplus MA_L(I_{81} \oplus I_{83}, I_{82} \oplus I_{84}) \oplus I_{84} \oplus MA_L(I_{81} \oplus I_{83}, I_{82} \oplus I_{84})] = \\
&= I_{81} \oplus MA_R(I_{81} \oplus I_{83}, I_{82} \oplus I_{84}) \oplus MA_R(I_{81} \oplus I_{83}, I_{82} \oplus I_{84}) = I_{81}.
\end{aligned}$$

Точно так же получаем $V_{11} = I_{81}$; $V_{12} = I_{83}$; $V_{13} = I_{82}$; $V_{14} = I_{84}$.

Таким образом, выходные данные второй подстадии процесса расшифрования совпадают с входными значениями предпоследней стадии процесса шифрования, за исключением перестановки второго и третьего блоков. Далее можно показать, что соотношение сохраняется для всех последующих подстадий вплоть до $V_{81} = I_{11}$; $V_{82} = I_{13}$; $V_{83} = I_{12}$; $V_{84} = I_{14}$. Наконец, входное преобразование процесса расшифрования эквивалентно преобразованию первой подстадии шифрования, за исключением перестановки второго-третьего блоков, и результаты зашифрования/расшифрования совпадут.

Алгоритм AES (Rijndael). Алгоритм – победитель конкурса AES, объявленный в 2000 году, был разработан двумя бельгийскими криптографами Дименом и Рийменом. Эта криптосистема не является обобщением шифра Фейстеля. В основе алгоритма – повторяющиеся раунды, каждый из которых состоит из замен, перестановок и прибавления ключа. Кроме того, AES использует сильную математическую структуру: большинство его операций основано на арифметике поля F_{2^8} [16]. Однако в отличие от DES зашифрование и расшифрование в этом алгоритме – процедуры разные. Элементы поля F_{2^8} хранятся в памяти компьютера в виде 8-битовых векторов (байтов). Например последовательность «1000 0011b» соответствует многочлену $X^7 + X + 1$ над полем F_2 или шестнадцатичному числу «83h».

Арифметические операции в поле F_{2^8} соответствуют операциям над двоичными многочленами из $F_2[X]$ по модулю неприводимого многочлена $m(X) = X^8 + X^4 + X^3 + X + 1$.

В алгоритме AES 32-битовые слова отождествляются с многочленами степени 3 из $F_{2^8}[X]$. Отождествление делается в формате «перевертыш», т. е. старший бит соответствует младшему

коэффициенту многочлена. Так, например, слово $a_0||a_1||a_2||a_3$ соответствует многочлену $a_3X^3 + a_2X^2 + a_1X + a_0$.

Арифметика в алгоритме совпадает с арифметическими действиями в кольце многочленов $\mathbb{F}_{2^8}[X]$ по модулю многочлена $M(X) = X^4 + 1$. Заметим, что многочлен $M(X) = (X + 1)^4$ приводим, и, следовательно, арифметические действия в алгоритме отличны от операций поля, в частности, бывают пары ненулевых элементов, произведение которых равно нулю [16].

AES – настраиваемый блочный алгоритм, который может работать с блоками из 128, 192 или 256 битов. Для каждой комбинации размера блоков и ключа определено свое количество раундов. Ниже рассматривается самый простой вариант алгоритма, при котором блоки, как и ключ, состоят из 128 битов. В этом случае в алгоритме выполняется 10 раундов.

Алгоритм оперирует с внутренней байтовой матрицей размером 4 x 4, называемой матрицей состояний:

$$S = \begin{bmatrix} s_{00} & s_{01} & s_{02} & s_{03} \\ s_{10} & s_{11} & s_{12} & s_{13} \\ s_{20} & s_{21} & s_{22} & s_{23} \\ s_{30} & s_{31} & s_{32} & s_{33} \end{bmatrix},$$

которую обычно записывают как вектор 32-битовых слов. Каждое слово в векторе представляет собой столбец матрицы. Подключи также хранятся в виде матрицы 4 x 4:

$$K = \begin{bmatrix} k_{00} & k_{01} & k_{02} & k_{03} \\ k_{10} & k_{11} & k_{12} & k_{13} \\ k_{20} & k_{21} & k_{22} & k_{23} \\ k_{30} & k_{31} & k_{32} & k_{33} \end{bmatrix}.$$

Операции алгоритма AES. Раундовая функция алгоритма действует с использованием четырех операций.

SubBytes. В алгоритме есть два типа S-блоков. Один применяется при зашифровании, а другой – при расшифровании.

S-блоки имеют прозрачную математическую структуру. Они поочередно обрабатывают строки матрицы состояний $s = [s_7, \dots, s_0]$, воспринимая их как элементы поля \mathbb{F}_{2^8} . Их работа состоит из двух шагов;

1. Вычисляется мультипликативный обратный к элементу $s \in \mathbf{F}_{2^8}$ и записывается как новый байт $x = [x_7, \dots, x_0]$. По соглашению элемент $[0, \dots, 0]$, не имеющий обратного, остается неизменным.

2. Битовый вектор x при помощи линейного преобразования над полем \mathbf{F}_2 переводится в вектор y :

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix},$$

служащий выходом S-блока. Действия S-блока на стадии расшифрования состоят в обратном линейном преобразовании и вычислении мультипликативного обратного. Эти преобразования байтов можно осуществить, используя табличный поиск или микросхему, реализующую вычисление обратных элементов в \mathbf{F}_{2^8} и линейные преобразования.

ShiftRows. Операция осуществляет циклический сдвиг матрицы состояний. Каждая из ее строк сдвигается на свое число позиций. В рассматриваемой версии шифра это преобразование имеет вид:

$$\begin{Bmatrix} \begin{bmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{bmatrix} & \begin{bmatrix} x_0 & x_1 & x_2 & x_3 \\ x_5 & x_6 & x_7 & x_4 \\ x_9 & x_{10} & x_{11} & x_8 \\ x_{13} & x_{14} & x_{15} & x_{12} \end{bmatrix} \end{Bmatrix}$$

Обратная операция – тоже циклический сдвиг, но в противоположном направлении. Данная операция является рассеивающим преобразованием на протяжении нескольких раундов.

MixColumns. Операция в сочетании с предыдущей служит рассеивающим преобразованием и обеспечивает зависимость каждого выходного байта от каждого входного.

Каждый столбец матрицы представляется в виде многочлена степени 3 с коэффициентами из \mathbf{F}_{2^8} : $a(X) = a_0 + a_1X + a_2X^2 + a_3X^3$.

Новый столбец получается умножением многочлена $a(X)$ на фиксированный многочлен

$$c(x) = '02h' + '01h' \cdot X + '01h' \cdot X^2 + '03h' \cdot X^3$$

по модулю многочлена $M(X) = X^4 + 1$. Так как умножение на многочлен – линейная операция, ее можно представить в виде действия матрицы:

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} '02h' & '03h' & '01h' & '01h' \\ '01h' & '02h' & '03h' & '01h' \\ '01h' & '01h' & '02h' & '03h' \\ '03h' & '01h' & '01h' & '02h' \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}.$$

Матрица коэффициентов невырождена над \mathbf{F}_{2^8} , поэтому операция обратима, а обратимое к ней действие реализуется матрицей, обратной к выписанной.

AddRoundKey. Сложение с подключом осуществляется побайтово по модулю 2 для каждого байта матрицы состояний с соответствующим элементом матрицы подключа. Обратная операция, очевидно, совпадает с исходной.

Структура раундов. Шифрование в алгоритме AES запишется в псевдокоде следующим образом:

```
AddRoundKey(S, K[0]);
For i:= 1 to 9 do Begin SubBytes(S); ShiftRows(S);
MixColumns(S); AddRoundKey(S, K[i]) End;
SubBytes(S);
ShiftRows(S);
AddRoundKey(S, K[10])
```

Блок открытого текста, предназначенный для шифрования, записывается в виде матрицы состояний S . Полученный в результате алгоритма шифротекст представляется той же матрицей. В последнем раунде операция *MixColumns* не осуществляется.

```
Процедура расшифрования в псевдокоде:
AddRoundKey(S, K[10]);
InverseSubBytes(S);
InverseShiftRows(S);
For i:= 1 to 9 do Begin AddRoundKey(S, K[i]);
InverseMixColumns(S); InverseShiftRows(S);
InverseSubBytes(S) End;
```

AddRoundKey(S, K[0])

Развертывание ключа. Основной ключ алгоритма состоит из 128 битов, а нужно произвести 10 подключей K_1, \dots, K_{10} , каждый из которых включает в себя четыре 32-битовых слова. Здесь используется константа раунда RC_i , вычисляющаяся по правилу

$$RC_i = X_i \pmod{X^8 + X^4 + X^3 + X + 1}.$$

Обозначим i -й подключ через $(W_{4i}, W_{4i+1}, W_{4i+2}, W_{4i+3})$. Основной ключ алгоритма делится на четыре 32-битовых слова

$(k0, k1, k2, k3)$, после чего подключи получаются в результате выполнения проведенного ниже алгоритма. В нем через RotBytes обозначена процедура циклического сдвига слова влево на 1 байт, а через SubBytes – применение S-блока из этапа шифрования к каждому байту слова:

```
W[0]:=K[0]; W[1]:=K[1]; W[2]:=K[2]; W[3]:=K[3];  
For i:=1 to 10 do Begin T:=RotBytes(W[4*i-1]); T:=SubBytes(T);  
T:=T $\wedge$ RC[i];  
W[4*i]:=W[4*i-4]  $\wedge$  T;  
W[4*i+1]:=W[4*i-3]  $\wedge$  W[4*i];  
W[4*i+2]:=W[4*i-2]  $\wedge$  W[4*i+1];  
W[4*i+3]:=W[4*i-1]  $\wedge$  W[4*i+2]; End
```

3.4. Принципы построения поточных шифров

Поточные системы шифрования, в отличие от блочных, не обладают таким недостатком, как возможность применения криптоанализа «со словарем», так как осуществляют познающее шифрование в алфавитах небольшой мощности. Кроме того, практика показывает, что пока лишь поточные шифры обеспечивают максимальные скорости шифрования, что важно при магистральном шифровании больших потоков информации.

Синхронизация поточных шифрсистем. Многоалфавитные поточные шифры не распространяют ошибок при искажении отдельных знаков шифртекста, но оказываются неустойчивыми к пропускам/вставке знаков шифртекста, поскольку это приводит к неправильному расшифрованию всего текста, следующего за пропущенным/вставленным знаком. Поэтому с учетом наличия

помех приходится решать проблему *синхронизации* процедур зашифрования и расшифрования.

По способу решения этой проблемы поточные шифрсистемы делят на *синхронные* и системы с *самосинхронизацией*.

В синхронных поточных шифрсистемах синхронизация достигается при помощи вставки в передаваемое сообщение специальных *маркеров*. В результате искажение не распространяется за пределы периода вставки маркеров.

Другое решение состоит в *реинициализации* состояний как шифратора отправителя, так и шифратора получателя при некотором согласованном условии.

Примеры синхронных систем – регистры сдвига с обратной связью, дисковые шифраторы или шифрмашинка Б. Хагелина С-36.

Системы с самосинхронизацией чаще всего используются в режиме шифрования с обратной связью по шифртексту (CFB), в котором текущее состояние системы зависит от некоторого числа N предыдущих знаков шифртекста. В этом случае потерянный в канале знак влияет на N последовательных состояний. После приема N правильных последовательных знаков из канала связи состояние приемного шифратора становится идентичным состоянию передающего шифратора.

Структура поточных шифрсистем. Поточная система шифрования состоит из двух основных блоков:

- управляющего блока (генератор ключевой последовательности – гаммы).
- шифрующего блока (реализует преобразование, накладывающее гамму на шифртекст).

Обычно управляющая гамма представляет собой псевдослучайную последовательность (ПСП), удовлетворяющую некоторой рекуррентной зависимости. В общем случае рекуррентная последовательность (на заданном множестве A) определяется формулой $x(i + m) = f(x(i), \dots, x(i + m - 1))$, $i \geq 0$,

в которой $f: A^m \rightarrow A$ – некоторая функция от m переменных.

Для получения рекуррентных последовательностей используются различные датчики *псевдослучайных чисел*, например *линейные* и *степенные конгруэнтные генераторы*.

В настоящее время большинство генераторов ПСП реализовано на основе регистров сдвига с линейными функциями обратной связи РСЛОС. Применяются также регистры сдвига с нелинейными функциями обратной связи.

Нарушения требований, предъявляемых к блокам поточной шифрсистемы, приводит к появлению аналитических или статистических слабостей алгоритма шифрования.

Требования к управляющему блоку таковы:

- период управляющей гаммы должен превышать максимально возможную длину открытых сообщений, подлежащих шифрованию;
- статистические свойства управляющей гаммы должны приближаться к свойствам случайной равновероятной последовательности;
- в управляющей гамме должны отсутствовать простые аналитические зависимости между близко расположенными знаками;
- криптографический алгоритм получения знаков управляющей гаммы должен обеспечивать высокую сложность определения секретного ключа.

К шифрующему блоку предъявляют следующие требования: применение алгоритма шифрования должно носить универсальный характер и не зависеть от вида шифруемой информации.

Иногда выдвигается дополнительное требование: способ построения шифрующего блока должен обеспечивать криптографическую стойкость шифра при перекрытиях управляющей гаммы, в частности при повторном использовании ключей.

Выполнение данных требований является необходимым, но не достаточным условием криптографической стойкости поточного шифра.

Регистр сдвига с обратной связью. Он состоит из двух частей: регистра сдвига и *функции обратной связи* (рис. 19).

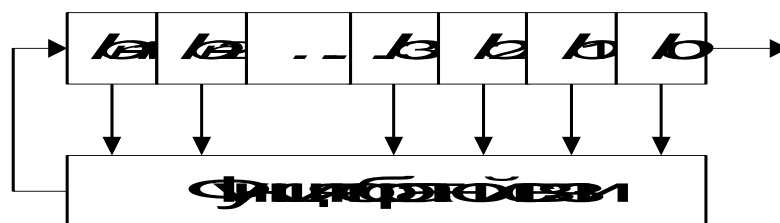


Рис 19. Регистр сдвига с обратной связью.

В общем случае регистр сдвига представляет собой последовательность некоторых элементов кольца или поля. Наиболее часто применяются битовые регистры сдвига. Длина такого регистра выражается числом битов. При каждом извлечении бита все биты регистра сдвигаются вправо на одну позицию. Новый старший бит рассчитывается как функция всех остальных битов регистра. Выходом обычно является младший значащий бит. Периодом регистра сдвига называют длину выходной последовательности до начала ее повторения.

Простейший тип регистров сдвига – регистр сдвига с линейной обратной связью (РСЛОС). Обратная связь – простая операция XOR над некоторыми битами регистра. Перечень этих битов определяется характеристическим многочленом и называется *последовательностью отводов*. Иногда такую схему называют *конфигурацией Фибоначчи* (рис. 20).

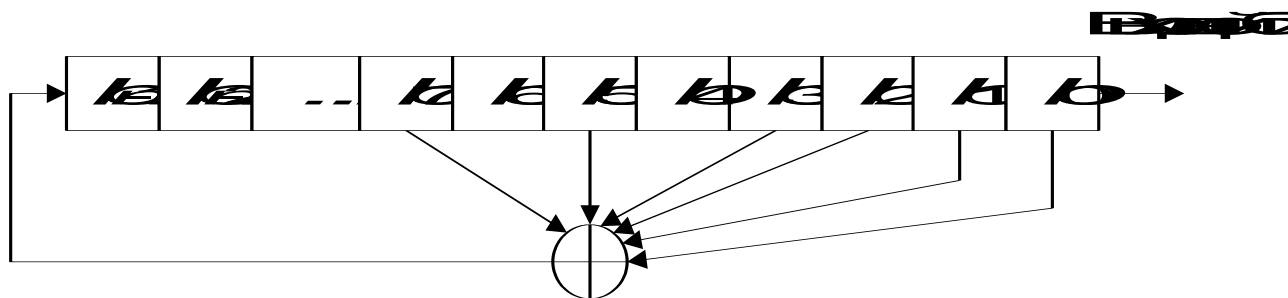


Рис.20. РСЛОС конфигурации Фибоначчи.

При программной реализации РСЛОС пользуются модифицированной схемой: для генерации нового значащего бита вместо использования битов последовательности отводов над каждым ее битом выполняется операция XOR с выходом генератора, заменяя старый бит последовательности отводов. Такую модификацию иногда называют *конфигурацией Галуа* (рис. 21).

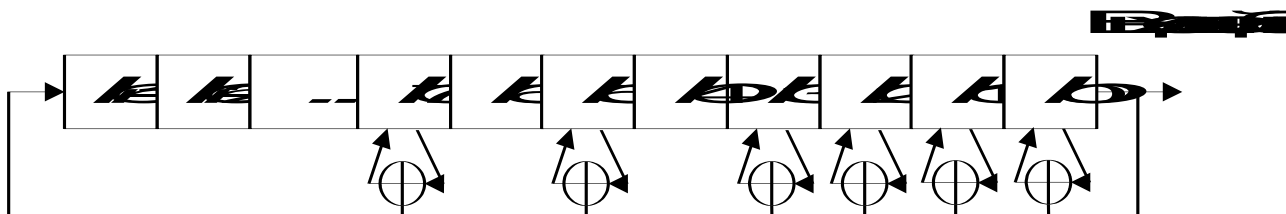


Рис.21. РСЛОС конфигурации Галуа.

n -битовый РСЛОС может находиться в одном из $2^n - 1$ внутренних состояний. Это значит, что теоретически такой регистр может генерировать псевдослучайную последовательность с периодом $2^n - 1$ битов (заполнение нулями совершенно бесполезно). Прохождение всех $2^n - 1$ внутренних состояний возможно только при определенных последовательностях отводов. Такие регистры называют РСЛОС с максимальным периодом. Для обеспечения максимального периода РСЛОС необходимо, чтобы его характеристический многочлен был *примитивным* по модулю 2.

Степень многочлена является длиной регистра сдвига. Примитивный многочлен степени n – это такой *неприводимый* многочлен, который является делителем $x^{2^n-1} + 1$, но не является делителем $x^d + 1$ для всех d , являющихся делителями $2^n - 1$. (При обсуждении многочленов термин *простое число* заменяется термином *неприводимый многочлен*.) Характеристический многочлен РСЛОС (см. рис. 20 и 21):

$$x^{32} + x^7 + x^5 + x^3 + x^2 + x + 1$$

примитивен по модулю 2. Период такого регистра будет максимальным, циклически проходя все $2^{32} - 1$ значений до их повторения. Чаще используются прореженные многочлены, т.е. у которых есть только некоторые коэффициенты. Наиболее популярны трехчлены.

Важным параметром генератора на базе РСЛОС является *линейная сложность*. Она определяется как длина n самого короткого РСЛОС, который может имитировать выход генератора. Линейная сложность важна, поскольку при помощи простого *алгоритма Берленкэмпа – Мэсси* можно воссоздать такой РСЛОС, проверив всего $2n$ битов гаммы. С определением нужного РСЛОС поточный шифр фактически взламывается.

Помимо РСЛОС применяются и регистры сдвига с нелинейной обратной связью, обратной связью по переносу и пр.

Ряд генераторов разработан на основе теоретико-числового подхода (генераторы Блума – Микали, RSA, BBS, сжимающие, аддитивные генераторы и пр.).

Математическое обеспечение синтеза поточных криптографических алгоритмов разработано более полно и подробно по сравнению с блочными криптоалгоритмами. Тем не менее для создания поточных шифров зачастую используют блочные криптоалгоритмы в режимах OFB или CFB.

Алгоритм Берленкэмпа-Мэсси. Пусть P – некоторое поле, e – единица поля P . Обозначим через $u = (u_0, u_1, \dots, u_{l-1})$ начальный отрезок произвольной последовательности u элементов поля P . Будем говорить, что многочлен

$$G(x) = \sum_{j=0}^{l-1} u_j x^j \in P[x]$$

вырабатывает отрезок $u(0, l-1)$, если

$$G(x) \mid \sum_{j=0}^{\infty} u_j x^j \in P[x],$$

т. е. данный отрезок последовательности является отрезком некоторой линейной рекуррентной последовательности с характеристическим многочленом $G(x)$. Алгоритм Берленкэмпа-Мэсси строит многочлен $G(x)$ наименьшей степени, вырабатывающий отрезок $u(0, l-1)$.

Определим функцию *умножения последовательности на многочлен*. Для произвольного многочлена $H(x) = \sum_{j=0}^n h_j x^j$ последовательности v положим $H(x) \cdot v = w$,

$$\text{где } w = \sum_{j=0}^{\infty} w_j x^j \in P[x]$$

Многочлен $G(x)$ вырабатывает $l \geq m$ знаков последовательности u , если выполняется равенство

$$G(x) \mid \sum_{j=0}^{l-1} u_j x^j \in P[x]$$

т. е. если первые $l - m$ знаков последовательности v равны нулю, а следующий за ними отличен от нуля.

Для многочлена $G(x) \in P[x]$ степени m и последовательности u введем параметры $l_u(G)$ и $k_u(G) = l_u(G) - m \in \mathbb{N} \cup \{0, +\infty\}$, где $l_u(G)$

– число знаков последовательности u , вырабатываемых многочленом $G(x)$. Ясно, что $k_u(G)$ – максимальное число первых подряд идущих нулей в последовательности $G(x) \cdot u$ (либо ∞).

Будем индуктивно строить последовательность многочленов $G_0(x), G_1(x), \dots$ неубывающих степеней $0 = m_0 < m_1 \leq m_2 \leq \dots$.

Начальные условия: $G_0(x) = e, m_0 = 0$.

Этап 1. Если

$$G_0(x) \cdot u = 0,$$

то полагаем

$$G_1(x) = G_0(x) \cdot u,$$

Если $G_1(x) \cdot u = 0$, т. е. если $k_u(G_1) = \infty$, то $G_1(x)$ – искомым минимальный многочлен ЛРП u . В противном случае строим $G_2(x)$.

Этап $t + 1$. пусть многочлены $G_0(x), \dots, G_t(x)$ уже построены, и степень многочлена $G_j(x)$ равна m_j , причем

$0 = m_0 < m_1 \leq m_2 \leq \dots m_t$. Пусть выполняются соотношения

$$G_0(x) \cdot u = 0, \dots, G_t(x) \cdot u = 0,$$

Определим число $s = s(t)$ так, чтобы выполнялись условия $m_t = m_{t-1} = \dots = m_s + 1 > m_s$ (такое s найдется, так как $m_1 > m_0$).

$$\text{Положим } G_{t+1}(x) = \begin{cases} G_t(x) - x^{k_s - k_t} \cdot u_t(k_t) \cdot u_s(k_s)^{-1} G_s(x), & \text{если } k_t k_s, \\ x^{k_t - k_s} G_t(x) - u_t(k_t) \cdot u_s(k_s)^{-1} G_s(x), & \text{если } k_t k_s. \end{cases}$$

Если $G_{t+1}(x) \cdot u = 0$, то нужный многочлен построен. В противном случае строим $G_{t+2}(x)$.

Теорема: если u – линейная рекуррентная последовательность над полем P с минимальным многочленом степени m , то $F(x) = G_t(x)$ для некоторого подходящего значения $\tau \leq 2m - 1 - k_0$.

Усложнение линейных рекуррентных последовательностей. Несмотря на достаточно большой период и хорошие статистические качества, линейные рекуррентные последовательности имеют простое строение (ярко выраженная аналитическая связь между предыдущими и последующим элементами последовательности). Поэтому в криптографических приложениях используют различные способы усложнения аналитического строения линейных рекуррент.

Основной подход, применяемый при проектировании генератора гаммы на базе РСЛОС, прост. Сначала берется один или несколько РСЛОС, обычно с разными длинами и различными характеристическими многочленами. (Если длины взаимно просты, а все многочлены обратной связи примитивны, то у сконструированного генератора будет максимальный период.) Ключ является начальным состоянием регистров. Выходной бит представляет собой функцию, желательно нелинейную, некоторых битов РСЛОС.

Если выходной бит является функцией единственного РСЛОС, то генератор называется *фильтрующим генератором* (рис. 22).

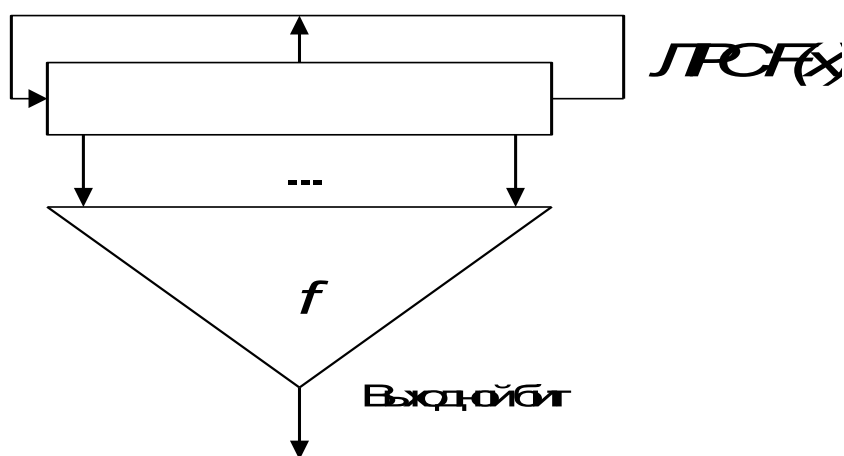


Рис.22. Фильтрующий генератор.

Если выходной бит является функцией нескольких РСЛОС, то генератор называется *комбинирующим генератором* (рис. 23).

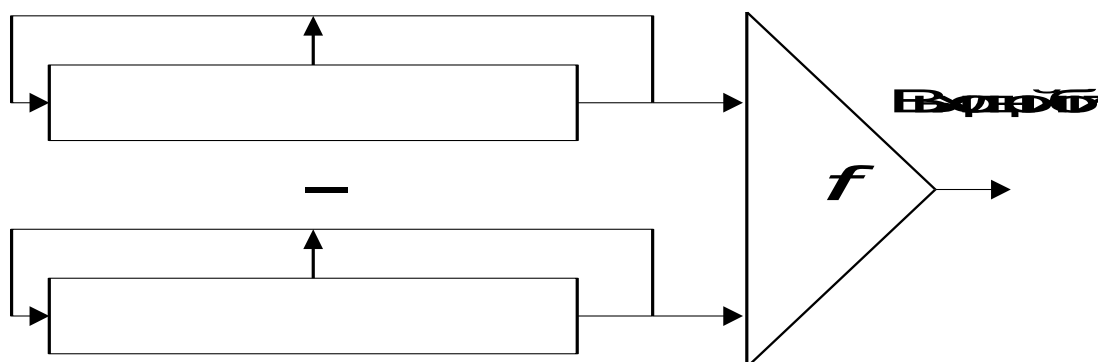


Рис.23. Комбинирующий генератор.

Еще один тип генераторов представляет собой композицию РСЛОС. В данной схеме выход одного из регистров подается на вход другого регистра (рис 24).

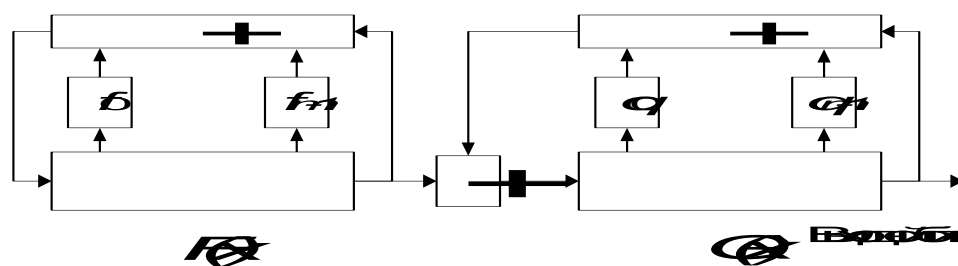


Рис.24. Генератор на основе композиции регистров сдвига.

Кроме перечисленных схем усложнения применяются и другие схемы (с динамическим изменением закона рекурсии с элементами памяти и пр.).

3.5. Современные поточные криптоалгоритмы.

Краткие сведения о некоторых наиболее известных поточных криптоалгоритмах, считающихся относительно стойкими, сведены в табл. 7

Таблица 7.

Стойкие поточные криптоалгоритмы.

№ пп.	Алгоритм	Описание
1	Каскад Голлмана	Усиленная версия генератора «старт-стоп». Последовательное соединение РСЛОС.
2	A5	Комбинация трех РСЛОС с изменяемым управлением тактированием
3	Nanoteq	На основе одного РСЛОС
4	Rambutan	Предположительно на основе РСЛОС
5	Pike	На основе трех аддитивных генераторов
6	Mush	Взаимно сжимающий генератор
7	RC4	S-блочная замена в режиме OFB. Эффективен для программной реализации
8	SEAL	На основе таблиц SHA. Эффективен для программной реализации. Наиболее быстрый алгоритм
9	WAKE	S-блочная замена в режиме CFB. Эффективен для программной реализации
10	Генератор Блюма–Микали	На основе трудности решения задачи дискретного логарифмирования
11	Генератор RSA	На основе трудности решения задачи факторизации
12	Генератор BBS	На основе трудности решения задачи факторизации

Алгоритм Гиффорда. Алгоритм построен на основе единственного 8-байтового регистра сдвига и работает в режиме OFB. Ключевая гамма генерируется побайтово. Схема работы алгоритма приведена на рис. 25.

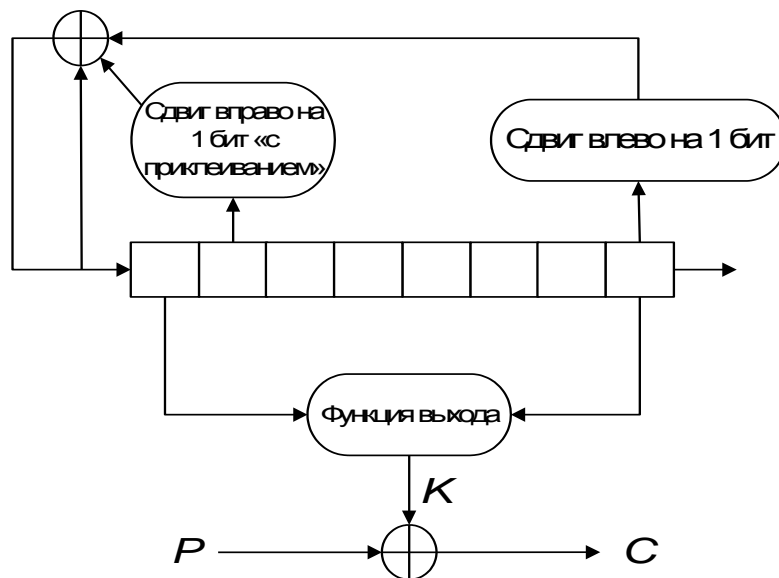


Рис.25. Алгоритм Гиффорда.

Характеристический многочлен регистра не примитивен и потому может быть вскрыт.

Алгоритм А5. Алгоритм построен на основе трех РСЛОС длиной 19, 22 и 23 с последовательностью «стоп/вперед», управляющей движением регистров (рис. 26). В каждом такте сдвигаются по меньшей мере два регистра.

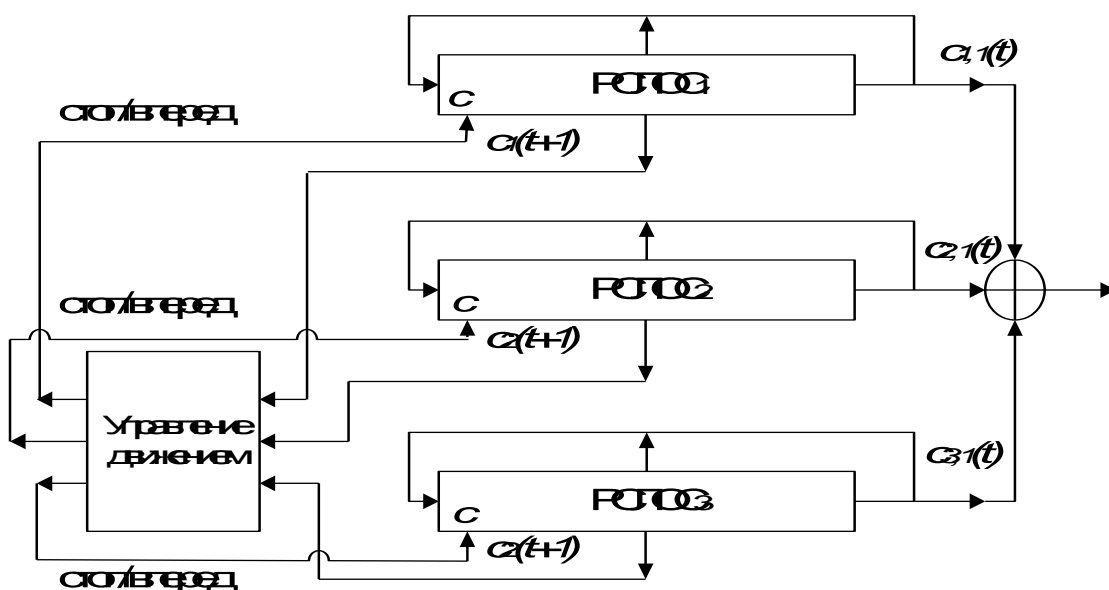


Рис.26. Алгоритм А5.

Алгоритм используется в системах связи стандарта GSM для шифрования трафика между абонентом и базовой станцией.

3.6. Режимы использования шифров.

Для решения разнообразных криптографических задач блочные шифры используют в нескольких режимах работы.

Алгоритм DES может использоваться в четырех режимах:

- *режим электронной кодовой книги* (ECB – Electronic Code Book);
- *режим сцепления блоков* (CBC – Cipher Block Chaining);
- *режим обратной связи по шифртексту* (CFB – Cipher Feed Back);
- *режим обратной связи по выходу* (OFB – Output Feed Back).

Режим электронной кодовой книги (ECB) отвечает обычному использованию DES как блочного шифра, осуществляющего некоторую простую замену блоков открытого текста. В режиме сцепления блоков (CBC) каждый блок C_i , $i > 1$, шифртекста перед очередным зашифрованием складывается по модулю 2 со следующим блоком открытого текста M_i (рис. 27). При этом вектор C_0 полагается равным начальному вектору IV (*Initial Vector*). Начальный вектор меняется ежедневно и хранится в секрете. Блоки C_1, C_2, \dots вырабатываются по рекуррентной формуле

$$C_i = DES_k(C_{i-1} \oplus M_i).$$

В режимах CFB и OFB алгоритм DES функционирует по аналогии с «шифром Вернама», в режиме OFB – как синхронный шифр, в режиме CFB – как шифр с самосинхронизацией.

В режиме CFB вырабатывается блочная «гамма» Z_0, Z_1, \dots , причем Z_0 полагается равным начальному вектору IV , а при $i \geq 1$ блоки гаммы удовлетворяют соотношению $Z_i = DES_k(C_{i-1})$ (рис. 28). Блоки открытого текста шифруются по правилу $C_i = M_i \oplus Z_i$, $i \geq 1$.

В режиме OFB вырабатывается блочная «гамма» Z_0, Z_1, \dots , причем Z_0 полагается равным начальному вектору IV , а при $i \geq 1$ блоки гаммы удовлетворяют соотношению $Z_i = DES_k(Z_{i-1})$ (рис. 29). Блоки открытого текста шифруются по правилу $C_i = M_i \oplus Z_i$, $i \geq 1$.

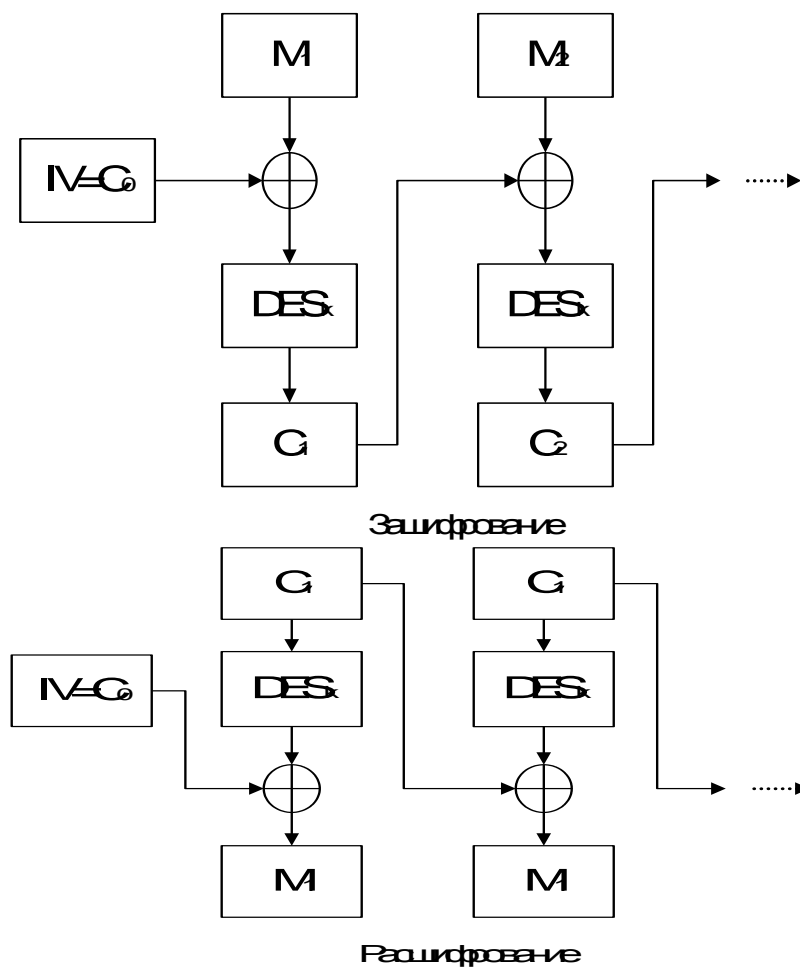


Рис.27. Режим CBC.

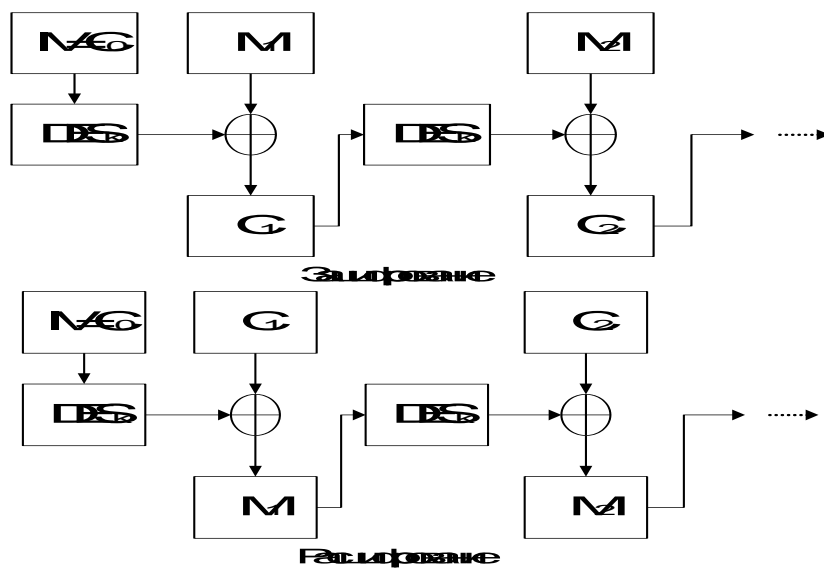


Рис.28. Режим CFB.

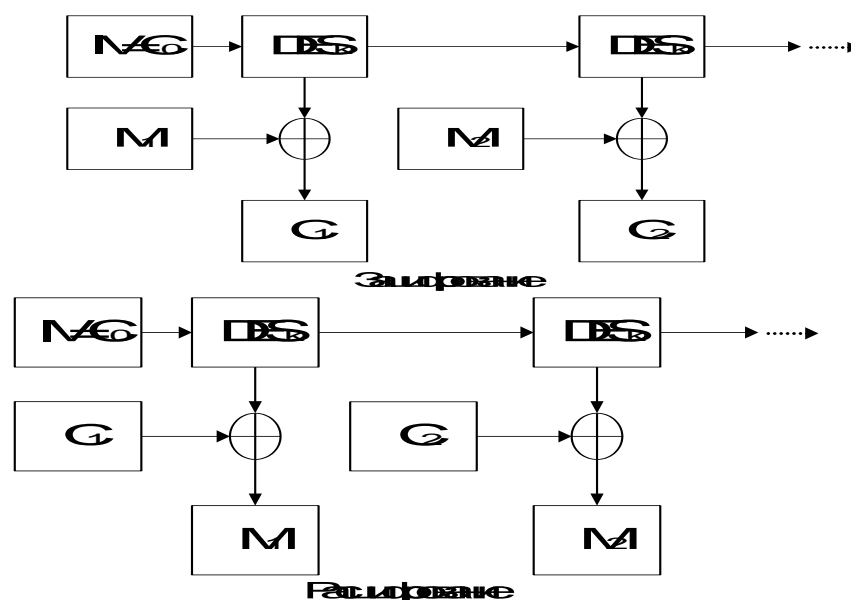


Рис.29. Режим OFB.

Кроме перечисленных режимов, DES имеет также *режим m -битовой обратной связи* $1 \leq m \leq 64$. Этот режим оперирует с m -битовыми блоками (рис. 30). Входной блок (64-битовый регистр сдвига) вначале содержит вектор инициализации IV , «выровненный по правому краю».

Блоки открытого текста шифруются по правилу $C_i = M_i \oplus P_i$, где P_i – вектор, состоящий из m старших битов блока $DES_k(C_{i-1})$. Обновление заполнения регистра сдвига осуществляется путем отбрасывания старших m битов и дописывания справа вектора P_i .

Указанные режимы имеют свои достоинства и недостатки. Основное достоинство режима ECB – простота реализации недостаток – возможность проведения *криптоанализа «со словарем»*. Дело в том что вследствие большой избыточности в открытом тексте вполне возможны повторения 64-битовых блоков. Это приводит к тому, что одинаковые блоки открытого текста будут представлены идентичными блоками шифртекста, что дает криптоаналитику возможность при наличии достаточно большого числа пар открытого и зашифрованного текста восстанавливать с большой вероятностью блоки открытого текста по шифртексту.

В режимах ECB и OFB искажение при передаче одного 64-битового блока шифртекста C приводит к искажению после расшифрования соответствующего блока M_i открытого текста, но не влияет на следующие блоки. Это свойство используется для шифрования информации, предназначенной для

передачи по каналам связи с большим числом искажений. Вместе с тем при использовании режима OFB остается открытым вопрос о периоде получаемой выходной гаммы, который в некоторых предположениях может составлять величину порядка 2^{32} .

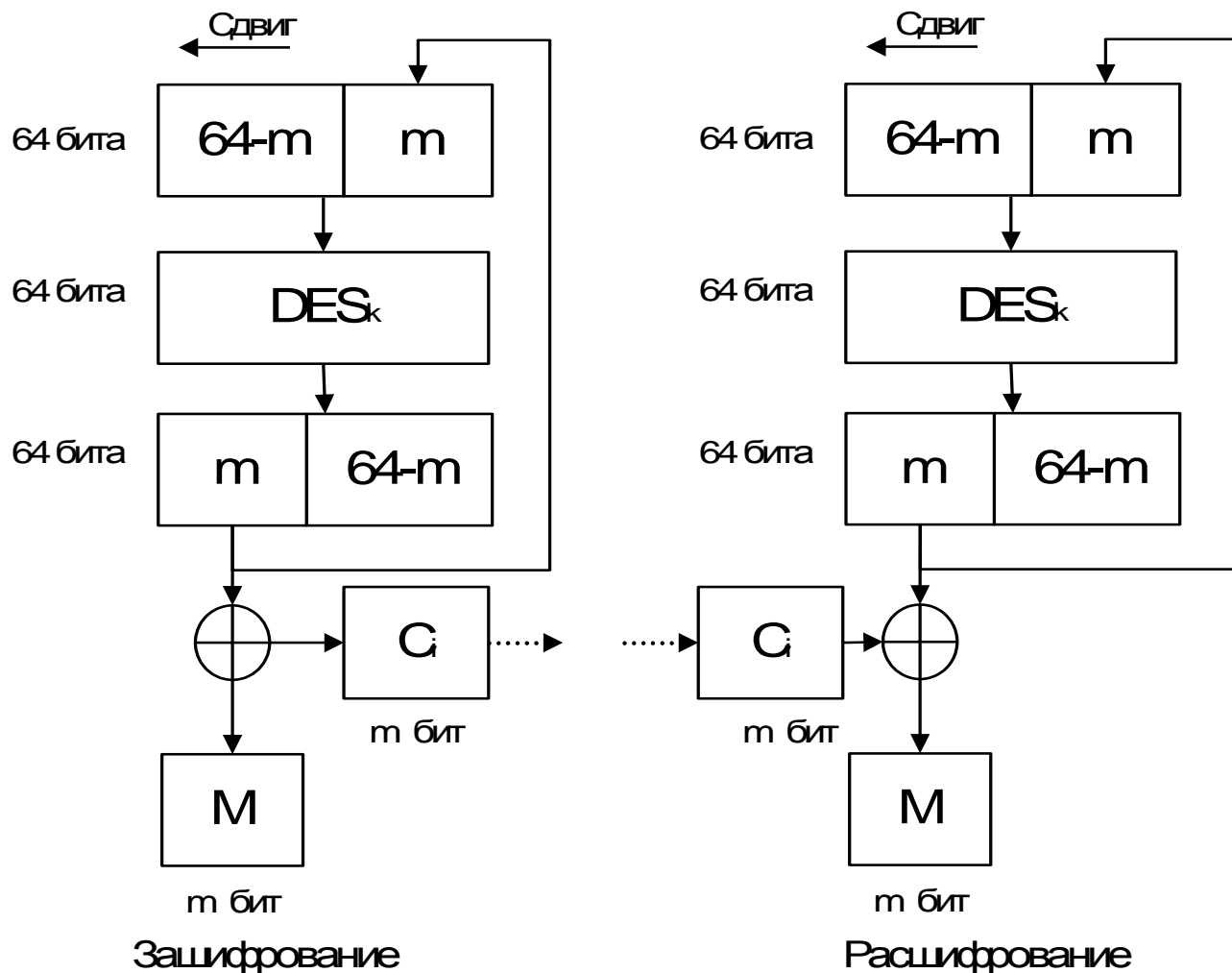


Рис.30. Режим m -битовой обратной связи.

В режимах CBC и CFB искажение при передаче одного блока шифртекста C_i приводит к искажению на приеме не более двух блоков открытого текста – M_i, M_{j+1} . В то же время изменение блока M_i приводит к тому, что C_i и все последующие блоки шифртекста будут искажены. Это свойство оказывается полезным для целей аутентификации. Такие режимы применяются для выработки кода аутентификации сообщения. Так, в режиме CBC берется вектор инициализации, состоящий из одних нулей. Затем с помощью ключа k вырабатываются блоки C_1, \dots, C_n шифртекста. Кодом аутентификации сообщения (КАС) служит блок C_n .

Если требуется обеспечить лишь целостность сообщения, отправитель передает блоки M_1, \dots, M_n вместе с C_n . Тогда противнику, желающему изменить сообщение, нужно соответствующим образом изменить и блок C_n . Возможность этого маловероятна, если только противник не располагает секретным ключом k .

Если же нужно обеспечить шифрование и аутентификацию, то отправитель сначала использует ключ k_1 для выработки КАС, затем шифрует последовательность блоков M_1, \dots, M_n , $M_{n+1} = \text{КАС}$ на втором ключе k_2 и получает последовательность блоков C_1, \dots, C_n, C_{n+1} . Получатель должен сначала расшифровать C_1, \dots, C_n, C_{n+1} на ключе k_2 , а затем проверить (с помощью k), что M_{n+1} – это КАС для M_0, \dots, M_n .

Можно поступить и иначе: сначала использовать k для зашифрования M_1, \dots, M_n , получая C_1, \dots, C_n , а затем k_2 – для получения КАС. Получатель же будет использовать k_2 для проверки КАС, а затем k – для расшифрования C_1, \dots, C_n .

Во всех перечисленных режимах вместо алгоритма DES может быть использован любой алгоритм блочного шифрования, в частности российский стандарт ГОСТ 28147-89.

В российском стандарте также предусмотрено несколько режимов использования: *режим простой замены*, *режим шифрования с обратной связью* и *режим гаммирования с обратной связью*, которые являются аналогами соответственно режимов ECB, CBC и CFB. Для того чтобы избавиться от указанной выше проблемы неопределенности длины периода гаммы в режиме OFB, в российском стандарте введен режим гаммирования, при котором блочный шифр используется в качестве узла усложнения некоторой последовательности гарантированного периода (рис. 31). Для выработки этой последовательности обычно применяются линейные регистры сдвига или счетчики по некоторому модулю.

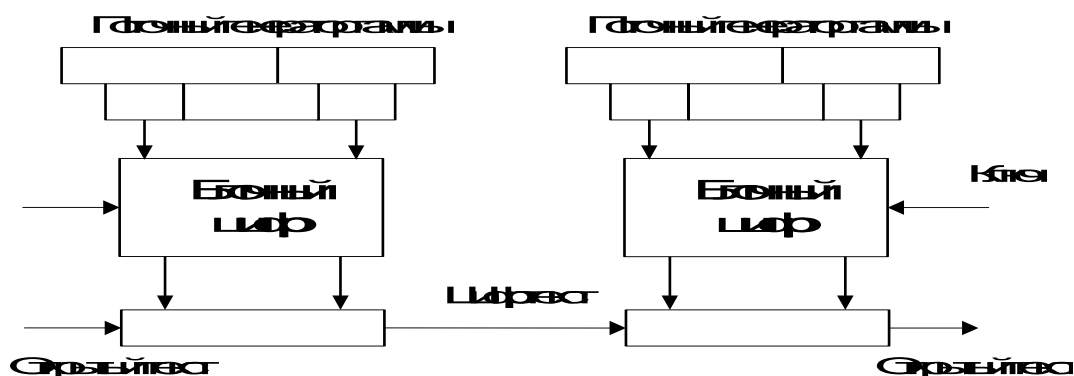


Рис.31. Режим гаммирования.

Уравнение шифрования имеет вид

$$C_i = M_i \oplus F(\gamma_i), \quad i = 1, 2, \dots,$$

где F – преобразование, осуществляемое блочным шифром;

γ_i – блоки, сформированные узлом выработки исходной гаммы из начального вектора γ_1 , который передается в начале сообщения в открытом или зашифрованном виде.

Ошибка при передаче, состоящая в искажении символа, приводит при расшифровании к искажению только одного блока, поэтому сохраняются все преимущества шифра гаммирования.

Контрольные вопросы

1. В чем различие между поточными и блочными шифрами?
2. Какие шифры удобнее в программной, а какие в аппаратной реализации?
3. Какие требования предъявляются к шифрующим преобразованиям блочных шифров?
4. В чем суть рассеивающих и перемешивающих преобразований при блочном шифровании?
5. Каковы основные параметры блочных шифров?
6. Какие виды поточных шифров могут быть эффективно реализованы программно?
7. Какие требования предъявляются к генераторам псевдослучайной последовательности?
8. Какие требования предъявляются к функции шифрования поточного шифра?
9. Какие режимы шифрования не распространяют искажений?

4. ПРИНЦИПЫ ПОСТРОЕНИЯ АСИММЕТРИЧНЫХ КРИПТОГРАФИЧЕСКИХ АЛГОРИТМОВ

4.1. Математические основы асимметричной криптографии

Преимущество систем с открытым ключом состоит в том, что ключ не требуется сохранять в тайне. Необходимо лишь обеспечить его аутентичность, что сделать, как правило, легче, чем обеспечить рассылку секретных ключей.

Системы шифрования с открытым ключом осуществляют блочное шифрование, поэтому открытый текст перед шифрованием разбивается на блоки.

Асимметричные системы шифрования обеспечивают значительно меньшие скорости шифрования, нежели симметричные, в силу чего они обычно используются для шифрования сообщений, а пересылаемых сеансовых секретных ключей, которые затем применяются в симметричных системах шифрования.

Для понимания принципов построения асимметричных криптосистем необходимо повторить некоторые понятия алгебры и алгоритмы теории чисел [21].

Свойства операций. Рассмотрим основные свойства арифметических операций, определенных на некотором множестве A .

$*$, \circ – обозначение операций.

Замкнутость: $\forall a, b \in A: a * b \in A$

Ассоциативность: $\forall a, b, c \in A: (a * b) * c = a * (b * c)$

Наличие нейтрального элемента $\Theta \in A$: ~~$\forall a \in A: a * \Theta = a$~~

Существование обратного элемента $\bar{a} \in A$: ~~$\forall a \in A: a * \bar{a} = \Theta$~~

Коммутативность: ~~$\forall a, b \in A: a * b = b * a$~~

Дистрибутивность операции \circ относительно операции $*$:

$$\forall a, b, c \in A: (a * b) \circ c = a \circ c * b \circ c$$

Группы, кольца.

*Определение. Группа – множество G с операцией, которая: замкнута, обладает нейтральным элементом, ассоциативна, относительно нее каждый элемент обладает обратным. Группу с коммутативной операцией называют **коммутативной** или **абелевой**.*

Практически все группы в криптографии – абелевы.
Обозначение группы: $(G, \text{знак операции})$.

Мультипликативная группа (G, \cdot) :

Групповая операция – умножение \cdot ;

Нейтральный элемент – единица 1 ;

Обратный элемент – a^{-1} ;

Многократное применение операции – возведение в степень:

$$a^5 = a \cdot a \cdot a \cdot a \cdot a.$$

Аддитивная группа $(G, +)$:

Групповая операция – сложение $+$;

Нейтральный элемент – ноль 0 ;

Обратный элемент – $-a$;

Многократное применение операции – умножение:

$$5 \cdot a = a + a + a + a + a.$$

Образующая g – такой элемент группы, что любой другой элемент может быть получен путем многократного применения к нему групповой операции. Запись: $G = \langle g \rangle$.

В мультипликативной группе: ~~$a \cdot b = b \cdot a$~~

В аддитивной группе: ~~$a + b = b + a$~~

Определение. Кольцо – множество R с двумя операциями: сложением и умножением, в котором обе операции замкнуты, ассоциативны, обладают нейтральным элементом, связаны законом дистрибутивности, а сложение коммутативно и для каждого элемента есть обратный по сложению. Обозначение кольца $(R, \cdot, +)$.

В коммутативном кольце операция умножения дополнительно обладает свойством коммутативности.

Основное кольцо, важное для криптологии, – коммутативное кольцо остатков от деления на натуральное число n , большее единицы, которое также называют кольцом вычетов по модулю n и обозначают \mathbf{Z}_n или $\mathbf{Z}/n\mathbf{Z}$.

Функция Эйлера. Поле. Теоремы Эйлера – Лагранжа и Ферма. Одна из центральных задач арифметики остатков – решение сравнения [13]:

$$a \cdot x \equiv b \pmod{n}.$$

⊙ Если $\text{НОД}(a, n) = 1$, то существует ровно одно решение, и оно находится с помощью a^{-1} :

$$\text{так как } a \cdot a^{-1} \equiv 1 \pmod{n},$$

то, домножив обе части сравнения на a^{-1} , получим

$$x \equiv b \cdot a^{-1} \pmod{n}.$$

⊙ Если $\text{НОД}(a, n) = g \neq 1$ и $g \mid b$, то сравнение имеет g решений.

Чтобы их найти, нужно разделить исходное сравнение на g :

$$a' \cdot x' \equiv b' \pmod{n'},$$

где $a' = a / g$, $b' = b / g$, $n' = n / g$.

Если x' – решение полученного сравнения, то решение исходного сравнения – любое число вида

$$x = x' + i \cdot n',$$

где $i = 0, 1, \dots, g - 1$.

⊙ В других случаях решений нет.

Пример:

$7 \cdot x \equiv 3 \pmod{143}$ – одно решение;

$11 \cdot x \equiv 3 \pmod{143}$ – решений нет;

$11 \cdot x \equiv 22 \pmod{143}$ – 11 решений.

Ситуация с единственным решением наиболее интересна для криптологии, поэтому важно знать количество элементов кольца, взаимно простых с его модулем. Оно дается *функцией Эйлера* φ и равно $\varphi(n)$. Значение $\varphi(n)$ можно легко найти по разложению числа n на простые множители. Если

$$n = \prod_{i=1}^n p_i^{e_i},$$

где p_i – различные простые числа, то

$$\varphi(n) = n \cdot \prod_{i=1}^n \left(1 - \frac{1}{p_i}\right).$$

Функция Эйлера для любого $n > 2$ имеет четное значение.

\mathbb{Z}_n^* – наибольшее подмножество элементов, образующих группу по умножению $\#\mathbb{Z}_n^* = \varphi(n)$ (количество элементов мультипликативной группы кольца вычетов по модулю n равно $\varphi(n)$).

Особый интерес представляют частные случаи:

1. Если p простое, то $\varphi(p) = p - 1$.
2. Если p и q – простые и $p \neq q$, то $\varphi(p \cdot q) = (p - 1)(q - 1)$.
3. Если p – простое число, то любой элемент в \mathbb{Z}_p (или $\mathbb{Z}/p\mathbb{Z}$) обладает мультипликативным обратным. Кольца с такими свойствами называются полями.

Определение. Полем называется множество $(F, \cdot, +)$ с двумя операциями, обладающее следующими свойствами:

- $(F, +)$ – абелева группа с нейтральным элементом 0;
- $(F \setminus \{0\}, \cdot)$ – абелева группа с нейтральным элементом 1;
- $(F, \cdot, +)$ удовлетворяет закону дистрибутивности (умножение дистрибутивно относительно сложения).

Поле – коммутативное кольцо, в котором каждый ненулевой элемент обратим. Каждый ненулевой элемент кольца \mathbb{Z}_p взаимно прост с p , так как p – простое, и, следовательно, обратим. Значит,

\mathbb{Z}_p – конечное поле, которое обычно называют полем вычетов по модулю p и обозначают $\mathbb{F}_p = \{0, 1, \dots, p - 1\}$. Мультипликативная группа $F_p = \{1, \dots, p - 1\}$ поля вычетов содержит все ненулевые элементы \mathbb{F}_p .

Теорема Лагранжа: ~~Группа вычетов по модулю p является циклической группой.~~

Обобщение Эйлера для малой теоремы Ферма [13]:

$x^{(n)} \equiv 1 \pmod{n}$, при $\text{НОД}(x, n) = 1$.

Малая теорема Ферма: ~~$a^{p-1} \equiv 1 \pmod{p}$.~~

Конечные поля. Целые числа по простому модулю не единственный пример конечных полей. Более общий тип полей используется при рассмотрении таких криптосистем, как AES, поточных шифров на основе РСЛОС и криптосистем на основе эллиптических кривых.

Рассмотрим множество многочленов от переменной x с коэффициентами из \mathbb{F}_p . Это множество обозначается через $\mathbb{F}_p[x]$ и образует кольцо относительно естественных операций суммы и умножения многочленов. Особый интерес представляет случай

$$p = 2.$$

Пример. В кольце $F_2[x]$ выполнены равенства:

$$(1+x+x^2) \cdot (x+x^3) \equiv 1+x^2 \pmod{x^4+1}$$

Зафиксируем многочлен $f(x)$ и будем рассматривать остальные элементы кольца $F_p[x]$ по модулю $f(x)$. Как и натуральные числа по модулю n , возможные остатки от деления на многочлен $f(x)$ будут образовывать кольцо. Оно обозначается через $F_p[x]/f(x)F_p[x]$ (по аналогии с $\mathbb{Z}/n\mathbb{Z}$).

Пример. $f(x) = x^4 + 1$ и $p = 2$. Тогда $(1+x+x^2) \cdot (x+x^3) \equiv 1+x^2 \pmod{x^4+1}$,

так как $x^4 \equiv -1$.

По аналогии с целыми числами по модулю n , где рассматривалось сравнение $a \cdot x \equiv b \pmod{n}$, можно поставить аналогичный вопрос и для многочленов. Пусть a, b и

f – многочлены из $F_p[x]$. Существование решения уравнения $a \cdot \alpha \equiv b \pmod{f}$

также зависит от НОД (a, f) , и возможны три ситуации.

Наиболее интересна ситуация, когда $\text{НОД}(a, f) = 1$.

Многочлен называется *неприводимым*, если у него нет делителей, отличных от него самого и констант. Неприводимость многочленов – то же самое, что и простота целых чисел. Кольцо $F_p[x]/f(x)F_p[x]$ является конечным полем тогда и только тогда, когда многочлен $f(x)$ неприводим.

Рассмотрим два неприводимых многочлена над полем F_2 :

$$f_1(x) = x^7 + x + 1; \quad f_2(x) = x^7 + x^3 + 1.$$

Возникают два конечных поля:

$$F_1 = F_2[x]/f_1(x)F_2[x]; \quad F_2 = F_2[x]/f_2(x)F_2[x].$$

Каждое состоит из 2^7 двоичных многочленов (каждый имеет ровно семь коэффициентов, равных единицы или нулю), степень которых не превосходит 6. Сложение в обоих полях выглядит одинаково, поскольку при вычислении суммы складываются коэффициенты многочленов по модулю 2. А вот умножаются элементы этих полей по-разному:

$$(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1) \cdot (x^6 + x^5 + x^4 + x^3 + x^2 + x + 1) \equiv 1 \pmod{f_1(x)}$$

$$(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1) \cdot (x^6 + x^5 + x^4 + x^3 + x^2 + x + 1) \equiv x^6 + x^5 + x^4 + x^3 + x^2 + x + 1 \pmod{f_2(x)}$$

Действительно ли различны поля F_1 и F_2 или это только кажущееся различие?

Изоморфны ли поля F_1 и F_2 ?

Изоморфизм: отображение $\varphi: F_1 \rightarrow F_2$, удовлетворяющее двум требованиям:

$$\varphi(a+b) = \varphi(a) + \varphi(b),$$
$$\varphi(ab) = \varphi(a)\varphi(b).$$

Для построения изоморфизма между F_1 и F_2 достаточно показать, как выражается корень $f_2(x)$ в виде многочлена от корня $f_1(x)$.

Изоморфизм существует между любыми двумя конечными полями с одинаковым числом элементов. Все конечные поля совпадают либо с целыми числами по простому модулю, либо с многочленами по модулю неприводимого многочлена.

Теорема: *существует единственное (с точностью до изоморфизма) конечное поле с числом элементов, равным степени простого числа.*

Конечное поле из $q = p^d$ элементов обозначается символом \mathbf{F}_q или $\text{GF}(q)$ (поле Галуа из q элементов).

Любое конечное поле K содержит в себе экземпляр поля целых чисел по некоторому простому модулю p , который называется *простым подполем* поля K . Число элементов простого подполя называется характеристикой поля и обозначается через $\text{char } K$. В частности $\text{char } \mathbf{F}_p = p$. На конечном поле характеристики p можно определить отображение Фробениуса:

$$\Phi: \mathbf{F}_q \rightarrow \mathbf{F}_q, \Phi(\alpha) = (\alpha^p),$$

которое является автоморфизмом (изоморфизмом поля с самим собой). Множество элементов из \mathbf{F}_q , остающихся неподвижными при действии Φ , совпадает с его простым подполем:

$$\{\alpha \in \mathbf{F}_q \mid \alpha^p = \alpha\} = \mathbf{F}_p.$$

Это утверждение – обобщение малой теоремы Ферма на случай любых конечных полей [13].

Ненулевые элементы конечного поля составляют конечную абелеву циклическую группу \mathbf{F}_q^* . Образующая этой группы называется *примитивным элементом* конечного поля. Примитивный элемент есть в любом конечном поле, их может быть

и несколько. Всегда можно найти такой элемент $g \in \mathbf{F}_q$, что любой ненулевой элемент $\alpha \in \mathbf{F}_q$ будет представляться в виде $\alpha = g^x$ при некотором целом показателе x .

Пример. В поле из восьми многочленов

~~$\mathbf{F}_2[x]$~~

В нем существует семь ненулевых элементов, а именно

$$1, \alpha, \alpha + 1, \alpha^2, \alpha^2 + 1, \alpha^2 + \alpha, \alpha^2 + \alpha + 1,$$

где α – корень многочлена $x^3 + x + 1$ (искусственно введенный элемент, удовлетворяющий соотношению $\alpha^3 + \alpha + 1 = 0$, в котором все действия выполняются по модулю 2).

Тогда:

$$\alpha^1 = \alpha;$$

$$\alpha^2 = \alpha^2;$$

$$\alpha^3 = \alpha + 1;$$

$$\alpha^4 = \alpha^2 + \alpha;$$

$$\alpha^5 = \alpha^2 + 1;$$

$$\alpha^6 = \alpha^2 + \alpha + 1;$$

$$\alpha^7 = 1$$

и α – примитивный элемент поля \mathbf{F}_{2^3} . Целые числа по модулю p также имеют примитивный элемент, так как \mathbf{F}_p – конечное поле.

Основные алгоритмы. Нахождение НОД не составляет проблем, если известно разложение чисел на простые множители или многочленов на неприводимые многочлены, но разложение на простые множители или неприводимые многочлены (факторизация) – очень трудоемкая операция.

Алгоритм разложения чисел на простые множители

(рис. 32). Алгоритм А (разложение на простые множители путем деления). По данному положительному целому числу N этот алгоритм находит простые множители $p_1 \leq p_2 \leq \dots \leq p_t$ числа N . В этом методе используется вспомогательная последовательность пробных делителей $d = d_0 < d_1 < d_2 < d_3 < \dots$, которая включает в себя все простые числа $< \sqrt{N}$ (и, если это удобно, может содержать числа, не являющиеся простыми). Последовательность чисел d_i должна также содержать, по крайней мере, одно значение, такое, что $d_k > \sqrt{N}$.

A1. [Начальная установка.] Присвоить $t \leftarrow 0, k \leftarrow 0, n \leftarrow N$. (В ходе выполнения алгоритма переменные t, k, n подчинены следующим условиям: « $n = N/p_1 \dots p_t$ и n не имеет простых множителей, меньших d_k »)

A2. [$n = 1$?] Если $n = 1$, алгоритм заканчивается.

A3. [Разделить.] Присвоить $q \leftarrow \lfloor n/d_k \rfloor, r \leftarrow n \bmod d_k$. (Здесь q и r – соответственно частное и остаток от деления числа n на d_k .)

A4. [Остаток равен нулю?] Если $r \neq 0$, то перейти к шагу A6.

A5. [Множитель найден.] Увеличить t на единицу и присвоить $p_t \leftarrow d_k, n \leftarrow q$. Возвратиться к шагу A2.

A6. [Частное мало?] Если $q > d_k$, увеличить k на единицу и возвратиться к шагу A3.

A7. [n – простое число.] Увеличить t на единицу, присвоить $p_t \leftarrow n$ и завершить выполнение алгоритма.

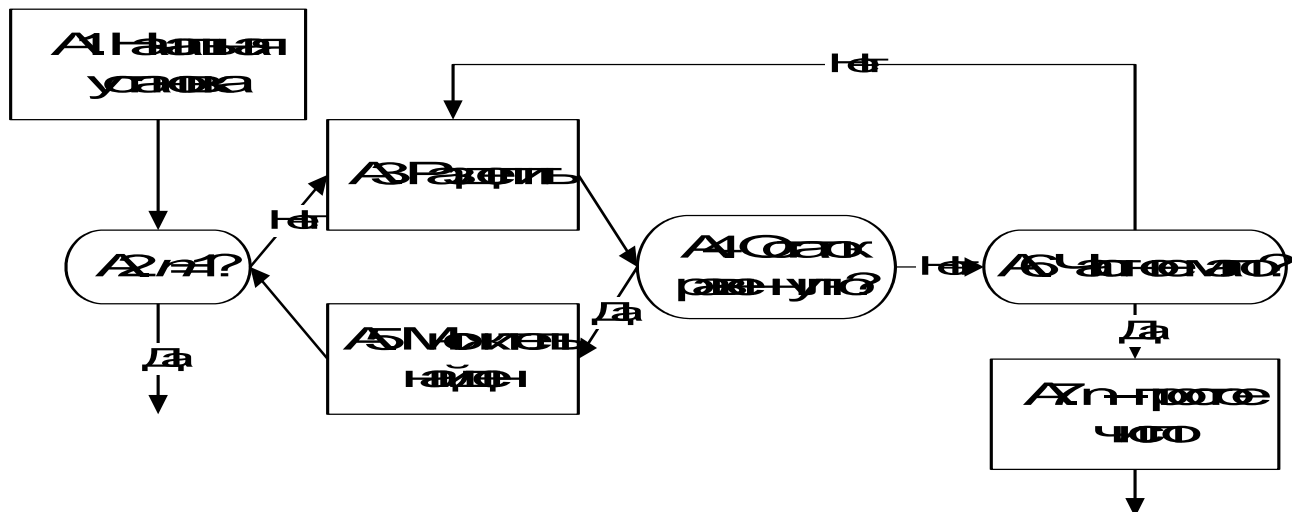


Рис.32. Алгоритм разложения числа на простые множители.

Пример. Разложение на простые множители числа $N = 25852$. Сразу же находим, что $N = 2 \cdot 12926$; следовательно, $p_1 = 2$. Далее, $12926 = 2 \cdot 6463$, так что $p_2 = 2$. Но теперь число $n = 6463$, не делится на числа 2, 3, 5, ..., 19; находим, что $n = 23 \cdot 281$; значит, $p_3 = 23$. В итоге имеем: $281 = 12 \cdot 23 + 5$ и $12 \leq 23$, т. е. $p_4 = 281$.

Алгоритм Евклида. Разделить целое число a на число b с остатком – это значит найти такие числа q и r с $0 \leq r < |b|$, при которых выполняется равенство $a = q \cdot b + r$.

Разделить многочлен f на многочлен g с остатком – это значит найти такие многочлены q и r с $0 \leq \deg r < \deg g$, при которых выполняется равенство $f = q \cdot g + r$.

Для вычисления НОД($r_0 = a$, $r_1 = b$) производится деление с остатком по следующей схеме:

$$a = r_0 = q_1 r_1 + r_2,$$

$$b = r_1 = q_2 r_2 + r_3,$$

.....

$$r_{m-2} = q_{m-1} r_{m-1} + r_m,$$

$$r_{m-1} = q_m r_m.$$

Работа алгоритма Евклида основана на том, что отображение ~~$f = q \cdot g + r$~~ сохраняет наибольший общий делитель.

Отображение работает до $a = \text{НОД}$, $b = 0$.

$$\begin{aligned} \text{Пример. } \text{НОД}(21, 12) &= \text{НОД}(21 \pmod{12}, 12) = \text{НОД}(9, 12) = \\ &= \text{НОД}(12 \pmod{9}, 9) = \text{НОД}(3, 9) = \text{НОД}(9 \pmod{3}, 3) = \\ &= \text{НОД}(0, 3) = 3. \end{aligned}$$

Работа бинарного алгоритма нахождения НОД основана на следующем ряде отображений, также сохраняющих НОД [12]:

$$\hat{\text{НОД}}(a, b) \left\{ \begin{array}{ll} 2 \cdot \hat{\text{НОД}}(a/2, b/2), & \text{если } a \text{ и } b \text{ четны}; \\ \hat{\text{НОД}}((a-b)/2, b), & \text{если } a \text{ и } b \text{ нечетны}; \\ \hat{\text{НОД}}(a/2, b), & \text{если } a \text{ четно, } b \text{ нечетно}; \\ \hat{\text{НОД}}(a, b/2), & \text{если } a \text{ нечетно, } b \text{ четно}. \end{array} \right.$$

Отображения работают до $(a-b)/2 = 0$. НОД = b .

Пример.

$$\begin{aligned} \text{НОД}(21, 12) &= \text{НОД}(21, 12/2) = \\ &= \text{НОД}(21, 6) = \text{НОД}(21, 6/2) = \\ &= \text{НОД}(21, 3) = \text{НОД}((21-3)/2, 3) = \\ &= \text{НОД}(9, 3) = \text{НОД}((9-3)/2, 3) = \\ &= \text{НОД}(3, 3) = \text{НОД}((3-3)/2, 3) = \\ &= \text{НОД}(0, 3) = 3 \end{aligned}$$

Во всех вышеперечисленных рассуждениях принимается $a \geq b$.

Расширенный алгоритм Евклида. Позволяет найти не только НОД(a, b), но и мультипликативный обратный к b по модулю a .

Преобразуем формулы для алгоритма Евклида, выразив все остатки r_i через a и b :

$$r_2 = r_0 - q_1 r_1 = a - q_1 b,$$

$$r_3 = r_1 - q_2 r_2 = b - q_2(a - q_1) = -q_2 a + (1 + q_1 q_2)b,$$

.....

$$r_{i-2} = s_{i-2}a + t_{i-2}b,$$

$$r_{i-1} = s_{i-1}a + t_{i-1}b,$$

$$r_i = r_{i-2} - q_{i-1}r_{i-1} = a(s_{i-2} - q_{i-1}s_{i-1}) + b(t_{i-2} - q_{i-1}t_{i-1})$$

.....

$$r_m = s_m a + t_m b.$$

Расширенный алгоритм Евклида по данным целым числам a и b выдает r_m , s_m и t_m , так что [9].

$$r_m = \text{НОД}(a, b) = s_m a + t_m b.$$

$$\text{При } \text{НОД}(a, b) = 1, \quad 1 \equiv t_m b \pmod{a}; \quad b^{-1} \equiv t_m \pmod{a}.$$

$$\text{Пример. } r_2 = 5 = 19 - 2 \cdot 7,$$

$$r_3 = 2 = 7 - 5 = 7 - (19 - 2 \cdot 7) = -19 + 3 \cdot 7,$$

$$r_4 = 1 = 5 - 2 \cdot 2 = (19 - 2 \cdot 7) - 2 \cdot (-19 + 3 \cdot 7) = 3 \cdot 19 - 8 \cdot 7.$$

$$\text{Отсюда } 1 \equiv -8 \cdot 7 \pmod{19}, \text{ так что } 7^{-1} \equiv -8 \equiv 11 \pmod{19}.$$

Алгоритмы нахождения НОД и мультипликативного обратного по модулю. Алгоритм Евклида. Алгоритм A (алгоритм Евклида в современной редакции). По данным неотрицательным целым числам u и v этот алгоритм находит их наибольший общий делитель.

A1. [$v = 0?$] Если $v = 0$, то выполнение алгоритма заканчивается, а в качестве результата возвращается число u .

A2. [Взять $u \bmod v$.] Присвоить $r \leftarrow u \bmod v$, $u \leftarrow v$, $v \leftarrow r$ и вернуться к шагу A1. (В результате выполняемых на этом шаге операций значение v уменьшается, значение $\text{НОД}(u, v)$ остается неизменным.)

Бинарный алгоритм НОД (рис. 33). Алгоритм B (бинарный алгоритм нахождения наибольшего общего делителя). По данным неотрицательным целым числам u и v этот алгоритм находит их наибольший общий делитель (использует знаковое представление чисел).

В1. [Найти степень 2.] Присвоить $k \leftarrow 0$, затем повторно присваивать $k \leftarrow k + 1$, $u \leftarrow u / 2$, $v \leftarrow v / 2$ нуль или более раз до тех пор, пока одно или оба числа u и v станут нечетными.

В2. [Начальная установка.] (Исходные значения чисел u и v уже разделены на 2^k , и, по крайней мере, одно из текущих значений нечетно.) Если нечетно u , то присвоить $t \leftarrow -v$ и перейти к шагу В4. В противном случае присвоить $t \leftarrow u$.

В3. [Уменьшить t наполовину.] (Здесь t четно и не нуль.) Присвоить $t \leftarrow t / 2$.

В4. [t четно?] Если t четно, то вернуться к шагу В3.

В5. [Установить $\max(u, v)$ заново.] Если $t > 0$, то присвоить $u \leftarrow t$, в противном случае присвоить $v \leftarrow -t$. (Большее из чисел u и v заменяется на $|t|$, за исключением, возможно, первого выполнения этого шага.)

В6. [Вычесть.] Присвоить $t \leftarrow u - v$. Если $t \neq 0$, то вернуться к шагу В3. В противном случае алгоритм останавливает выполнение, а на выходе будет $u \cdot 2^k$.

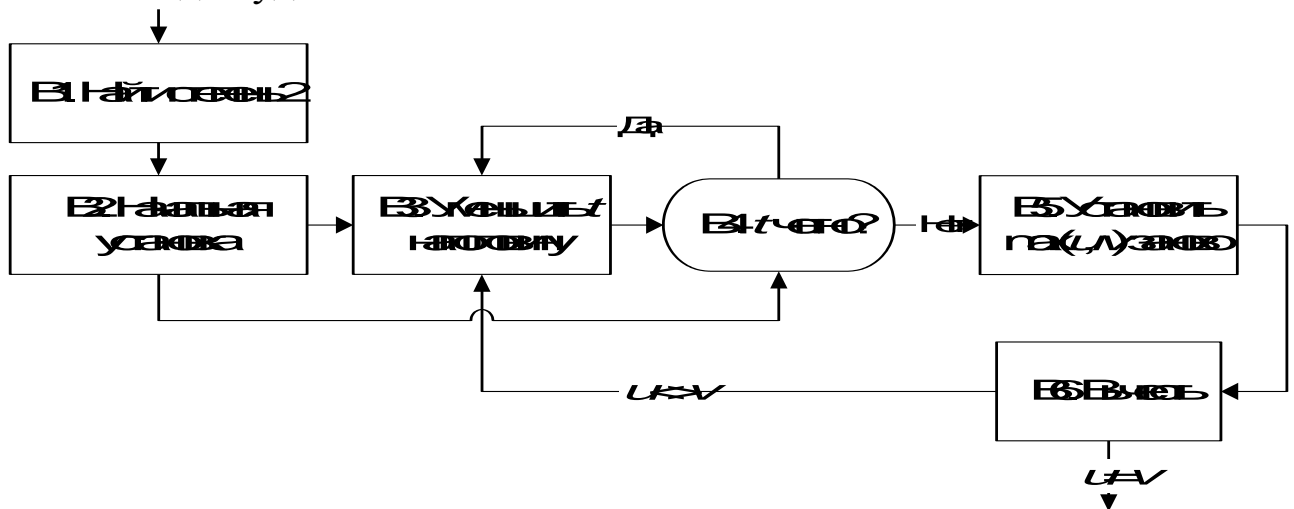


Рис.33. Бинарный алгоритм НОД.

Расширенный алгоритм Евклида. Алгоритм X (обобщенный алгоритм Евклида). Для заданных целых чисел u и v этот алгоритм определяет вектор (u_1, u_2, u_3) , такой, что $uu_1 + vu_2 = u_3 = \text{НОД}(u, v)$. В процессе вычисления используются вспомогательные векторы (v_1, v_2, v_3) , (t_1, t_2, t_3) ; действия производятся таким образом, что в течение всего процесса вычисления выполняются соотношения:

$$ut_1 + vt_2 = t_3; uu_1 + vu_2 = u_3; uv_1 + vv_2 = v_3.$$

X1. [Начальная установка.] Присвоить $(u_1, u_2, u_3) \leftarrow (1, 0, u)$, $(v_1, v_2, v_3) \leftarrow (0, 1, v)$.

X2. [$v_3 = 0$?] Если $v_3 = 0$, то выполнение алгоритма заканчивается.

X3. [Разделить и вычесть.] Присвоить $q \leftarrow \lfloor u_3/v_3 \rfloor$, затем присвоить

$$(t_1, t_2, t_3) \leftarrow (u_1, u_2, u_3) - (v_1, v_2, v_3) \cdot q,$$

$$(u_1, u_2, u_3) \leftarrow (v_1, v_2, v_3),$$

$$(v_1, v_2, v_3) \leftarrow (t_1, t_2, t_3).$$

Возвратиться к шагу X2.

Расширенный бинарный алгоритм НОД. Алгоритм Y. Модификация алгоритма X с использованием принципов алгоритма В. Для заданных целых чисел u и v этот алгоритм определяет вектор (u_1, u_2, u_3) , такой, что $uu_1 + vu_2 = u_3 = \text{НОД}(u, v)$. В процессе вычисления используются вспомогательные векторы (v_1, v_2, v_3) ,

(t_1, t_2, t_3) ; в течение всего процесса вычисления, так же как и в X, выполняются соотношения:

$$ut_1 + vt_2 = t_3; uu_1 + vu_2 = u_3; uv_1 + vv_2 = v_3.$$

(Используется знаковое представление чисел.)

Y1. [Найти степень 2.] То же, что в шаге В1.

Y2. [Начальная установка.] Присвоить $(u_1, u_2, u_3) \leftarrow (1, 0, u)$, $(v_1, v_2, v_3) \leftarrow (v, 1 - u, v)$. Если число u нечетно, присвоить $(t_1, t_2, t_3) \leftarrow (0, -1, -v)$ и перейти к шагу Y4. В противном случае присвоить $(t_1, t_2, t_3) \leftarrow (1, 0, u)$.

Y3. [Выполнить половинное деление t_3 .] Если t_1 и t_2 четны, присвоить $(t_1, t_2, t_3) \leftarrow (t_1, t_2, t_3)/2$; иначе присвоить $(t_1, t_2, t_3) \leftarrow (t_1 + v, t_2 - u, t_3)/2$. (В последнем случае $t_1 + v$ и $t_2 - u$ будут оба четными.)

Y4. [t_3 четно?] Если t_3 четно, вернуться к шагу Y3.

Y5. [Переустановить $\max(u_3, v_3)$.] Если t_3 положительно, присвоить $(u_1, u_2, u_3) \leftarrow (t_1, t_2, t_3)$; иначе присвоить $(v_1, v_2, v_3) \leftarrow (v - t_1, -u - t_2, -t_3)$.

Y6. [Вычесть.] Присвоить $(t_1, t_2, t_3) \leftarrow (u_1, u_2, u_3) - (v_1, v_2, v_3)$. Затем, если $t_1 \leq 0$, присвоить $(t_1, t_2) \leftarrow (t_1 + v, t_2 - u)$. Если $t_3 \neq 0$, вернуться к шагу Y3. В противном случае закончить выполнение алгоритма с результатом, равным $(u_1, u_2, u_3 \cdot 2^k)$.

Китайская теорема об остатках. Для двух сравнений.

Система сравнений

$$x \equiv a \pmod{m},$$

$$x \equiv b \pmod{n}.$$

при $\text{НОД}(m, n) = 1$ имеет единственное решение по модулю $m \cdot n$, которое определяется по формулам:

$$T = m^{-1} \pmod{n},$$

$$u = (b - a)T \pmod{n},$$

$$x \pmod{m \cdot n} = a + u \cdot m.$$

Пример.

$$x \equiv 4 \pmod{7},$$

$$x \equiv 3 \pmod{5}.$$

$$T = 7^{-1} \pmod{5} = 2^{-1} \pmod{5} = 3 \pmod{5},$$

$$u = (3 - 4) \cdot 3 \pmod{5} = 4 \cdot 3 \pmod{5} = 2 \pmod{5},$$

$$x \pmod{35} = 4 + 2 \cdot 7 = 18.$$

Общий случай КТО. Пусть m_1, \dots, m_r и a_1, \dots, a_r — целые числа, причем все m_i попарно взаимно просты [16].

Нужно найти такой x по модулю $M = m_1 \cdot m_2 \cdot \dots \cdot m_r$, что $x = a_i \pmod{m_i}$ для всех i .

Решение единственно и равно:

$$x = \sum_{i=1}^r a_i M_i y_i \pmod{M}$$

где $M_i = M / m_i$ и $y_i = M_i^{-1} \pmod{m_i}$.

Пример.

$$x \equiv 5 \pmod{7},$$

$$x \equiv 3 \pmod{11},$$

$$x \equiv 10 \pmod{13}.$$

$$M_1 = 143, y_1 = 5,$$

$$M_2 = 91, y_2 = 4,$$

$$M_3 = 77, y_3 = 12.$$

$$x = \sum_{i=1}^r a_i M_i y_i \pmod{M} = 715 \cdot 5 + 364 \cdot 3 + 924 \cdot 10 \pmod{1001} = 894.$$

Символы Лежандра и Якоби. Извлечение корней. Пусть p – простое число, большее двух. Рассмотрим отображение

$$F_p \rightarrow F_p, \alpha \mapsto \alpha^2,$$

сопоставляющее каждому элементу поля его квадрат. На множестве ненулевых элементов поля F_p это отображение в точности «два-в-один», т. е. если из ненулевого элемента $x \in F_p$ можно извлечь квадратный корень, то таких корней у него ровно двум, и, кроме того, ровно половина элементов из F_p являются полными квадратами. Полные квадраты в F_p называются *квадратичными вычетами* по модулю p . Множество всех квадратичных вычетов по модулю p является подгруппой порядка $(p-1)/2$ в мультипликативной группе F_p . Элементы мультипликативной группы F_p , из которых нельзя извлечь квадратный корень, называются *квадратичными невычетами*.

Для выявления полных квадратов по модулю p вводится символ Лежандра $\left(\frac{a}{p}\right)$.

Он равен нулю, если a делится на p , $+1$, если a – квадратичный вычет по модулю p ; -1 , если a – квадратичный невычет.

Символ Лежандра легко вычисляется по формуле

$$\left(\frac{a}{p}\right) = a^{\frac{p-1}{2}} \pmod{p}.$$

Но использование этой формулы сопряжено с вычислениями больших степеней, и на практике предпочитают пользоваться *законом квадратичной взаимности*

$$\left(\frac{p}{q}\right) \left(\frac{q}{p}\right) = (-1)^{\frac{p-1}{2} \frac{q-1}{2}}.$$

Иными словами,

$$\left(\frac{q}{p}\right) = \begin{cases} -\left(\frac{p}{q}\right), & \text{при } p = q = 3 \pmod{4} \\ \left(\frac{p}{q}\right), & \text{в других случаях.} \end{cases}$$

При вычислении символа Лежандра большую помощь оказывают следующие дополнительные формулы:

$$\left(\frac{2}{p}\right) = (-1)^{(p^2-1)/8}.$$

1. Выбрать наугад такое n , что $\left(\frac{n}{p}\right) = -1$.

3. Положим $y = n^q \pmod{p}$, $r = e$, $x = a^{(q-1)/2} \pmod{p}$.

4. Положим $b = ax^2 \pmod{p}$, $x = ax \pmod{p}$.

5. Пока $b \not\equiv 1 \pmod{p}$:

- найти наименьшее число m , такое, что $\mathcal{B}^{3''} = \{m, p\}$
- положить $\mathcal{B}^{3''} = \{m, p\}$
- положить $\mathcal{B}^{3''} = \{m, p\}$

6. Вывести x .

Если $p \equiv 3 \pmod{4}$, то для извлечения квадратного корня из a можно использовать формулу

$$x_{el}^{G=1/4}(\eta)$$

формула дает правильный ответ, потому что

REFPES.

Символ Лежандра определен только в случае простого знаменателя. Если же знаменатель составной, то вводится *символ Якоби*, обобщающий символ Лежандра.

Пусть n – нечетное число, большее двух, и

$$n = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k} .$$

Символ Якоби определяется через символы Лежандра простых делителей числа n следующим образом:

$$\overline{a} \overline{a} \overline{a} \overline{a} \dots$$

Символ Якоби можно вычислять так же, как и символ Лежандра, опираясь на тождество, выведенное из закона квадратичной взаимности:

$$\left(\frac{a}{n}\right) = \left(\frac{n}{a}\right)$$

где $a = 2^e a_1$ и a_1 нечетно.

Полезно помнить еще несколько формул, справедливых при нечетном n :

$$\left(\frac{1}{n}\right) = 1,$$

$$\left(\frac{2}{n}\right) = (-1)^{(n^2-1)/8},$$

$$\left(\frac{-1}{n}\right) = (-1)^{(n-1)/2}.$$

Это дает быстрый алгоритм вычисления символа Якоби и, соответственно, символа Лежандра, как его частный случай, без разложения на множители. Единственное, что нужно сделать, – выделить максимальную степень двойки.

Пример.

$$\left(\frac{a}{n}\right) = \left(\frac{n}{a}\right)$$

Символ Лежандра $\left(\frac{a}{p}\right)$ сообщает, является ли a полным квадратом по модулю простого числа p . Символ Якоби ничего не утверждает о возможности извлечения квадратного корня из a по модулю составного числа n . Если a в действительности квадрат по модулю n , то символ Якоби будет равен $+1$. Однако из равенства $\left(\frac{a}{n}\right) = +1$ нельзя сделать вывод о том, что a – полный квадрат. Несмотря на благоприятное равенство, квадратный корень из a может не извлекаться.

Извлечение корня из числа по модулю составного $n = p \cdot q$ в предположении, что разложение n на простые множители известно и a – полный квадрат по модулю n , т. е.

$$\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = 1$$

Сначала извлекается корень из a по модулю p и обозначается через s_p (таких корней два, как будет показано позже). Затем

извлекается квадратный корень из a по модулю q и обозначается s_q (таких корней также два). Наконец, для вычисления искомого корня применяем китайскую теорему об остатках к системе

$$x \equiv s_p \pmod{p},$$

$$x \equiv s_q \pmod{q}.$$

Пример. Вычислим корень из $a = 217$ по модулю $n = 221 = 13 \cdot 17$.

Квадратные корни из a по модулям 13 и 17 соответственно равны $s_{13} = 3$ и $s_{17} = 8$. Опираясь на китайскую теорему об остатках, получаем $s = 42$. Проверим правильность данного решения:

$$s^2 = 42^2 \equiv 217 \pmod{221}.$$

Существуют еще три квадратных корня из $a = 217$ по модулю $n = 221$, поскольку n имеет два простых делителя. Для их отыскания применим КТО к трем системам с коэффициентами:

$$s_{13} = 10, s_{17} = 8;$$

$$s_{13} = 3, s_{17} = 9;$$

$$s_{13} = 10, s_{17} = 9$$

и получим полный ответ: 42, 94, 127, 179.

4.2. Примеры современных асимметричных шифров

Криптосистема RSA. Система построена на следующих функциях [21]:

- *Односторонняя функция* – умножение двух больших простых чисел: $N = p \cdot q$.

Обратная задача – *задача факторизации*, (является сложной).

- *Односторонняя функция с секретом* – возведение в степень по модулю составного $N = p \cdot q$ с фиксированной степенью E : $C = m^E \bmod N$.

Обратная задача – *задача извлечения корня степени E по модулю N* : $m = \sqrt[E]{C} \bmod N$.

Задача является сложной, если неизвестно разложение N на множители (секрет); простой – при известном разложении N .

В дальнейшем секретные элементы криптосистем обозначаются строчными (p , q , m и пр.) латинскими буквами, открытые – прописными (N , C , E и пр.).

Предположим, абонент А считает нужным разрешить всем желающим отправлять ему секретные сообщения, расшифровать которые способен только он. Для этого выбирается два больших простых числа p и q , которые держатся в секрете, и публикуется их произведение $N = p \cdot q$, которое называют модулем алгоритма. Кроме того, выбирается шифрующая экспонента E , удовлетворяющая условию

$$\text{НОД}(E, (p-1), (q-1)) = 1.$$

Как правило, E берут равным 3, 17 или 65537. Пара, доступная всем желающим, – это (N, E) . Для выбора секретного ключа применяют расширенный алгоритм Евклида к паре чисел

E и $(p-1)(q-1)$, получая при этом расшифровывающую экспоненту d . Найденная экспонента удовлетворяет соотношению

$$E \cdot d = 1 \pmod{(p-1)(q-1)} = 1 \text{ [9]}.$$

Секретным ключом является тройка (d, p, q) . Фактически можно было бы выбросить простые делители p и q из ключа и помнить лишь о d и всем числе N . Но, это снизит скорость алгоритма.

Допустим, пользователь В намерен зашифровать сообщение, адресованное А. Он сверяется с открытым ключом и представляет

сообщение в виде числа m , строго меньшего модуля N алгоритма. Шифртекст C получается из m по следующему правилу:

$$C = m^E \pmod{N}.$$

Абонент А, получив шифrogramму, расшифровывает ее, возводя число C в степень d : $m = C^d \pmod{N}$.

Равенство имеет место в связи с тем, что порядок группы:

$$\#(\mathbf{Z}/N\mathbf{Z})^* = \varphi(N) = (p-1)(q-1),$$

потому по теореме Лагранжа (Эйлера – Ферма)

$$x^{(p-1)(q-1)} = 1 \pmod{N}$$

для любого числа $x \in (\mathbf{Z}/N\mathbf{Z})^*$ [13]. Поскольку E и d взаимно обратны по модулю $(p-1)(q-1)$, при некотором целом числе s получается равенство

$$Ed - s(p-1)(q-1) = 1.$$

Следовательно,

$$C^d = (m^E)^d = m^{Ed} = m^{1+s(p-1)(q-1)} = m \cdot m^{s(p-1)(q-1)} = m \pmod{N}.$$

Пример. Пусть $p = 7$ и $q = 11$. Тогда $N = 77$, а $(p-1)(q-1) = 6 \cdot 10 = 60$. В качестве открытой шифрующей экспоненты возьмем число $E = 37$, поскольку $\text{НОД}(37, 60) = 1$. Применяя расширенный алгоритм Евклида, найдем $d = 13$, так как $37 \cdot 13 = 481 = 1 \pmod{60}$.

Для зашифрования сообщения численное представление которого имеет вид $m = 2$:

$$C = m^E \pmod{N} = 2^{37} \pmod{77} = 51.$$

Расшифрование происходит аналогично:

$$m = C^d \pmod{N} = 51^{13} \pmod{77} = 2.$$

Взаимосвязь компонентов RSA. Задача RSA: Даны числа C и E , последнее из которых удовлетворяет соотношению:

$$\text{НОД}(E, (p-1)(q-1)) = 1.$$

Требуется найти такое число m , что

$$m^E = C \pmod{N}.$$

Лемма. Задача RSA не сложнее проблемы факторизации.

Доказательство. Применяя алгоритм факторизации, разложим число N на простые множители, вычислим значение функции Эйлера $\Phi = \varphi(N)$ и найдем $D = 1/E \pmod{\Phi}$.

Теперь, зная число D , легко восстановить m , поскольку

$$C^D = m^{ED} = m^{1 \pmod{\Phi}} = m \pmod{N}.$$

Отсюда следует, что при известных p и q легко находятся d и m .

Самые большие числа, которые к настоящему времени удается разложить на множители за разумное время, имеют 500 двоичных знаков. В связи с этим для обеспечения стойкости систем среднего срока действия рекомендуют брать модули шифрования порядка 1024 бит. Для систем большого срока действия следует выбирать модули, состоящие из 2048 бит.

Секретная экспонента и проблема факторизации. Лемма. Если известна секретная экспонента d алгоритма RSA, соответствующая открытому ключу (N, E) , то число N можно эффективно разложить на множители.

Доказательство. При некотором целом s имеет место равенство

$$Ed - s(p - 1)(q - 1) = 1.$$

Возьмем произвольное целое число $X \neq 0$. Тогда $X^{Ed-1} = 1 \pmod{N}$.

Вычисляем квадратный корень Y_1 по модулю N :

$$Y_1 = \sqrt[Ed-1]{X} \pmod{N}$$

что можно сделать, поскольку $Ed - 1$ известно и четно. Приходим к тождеству $Y_1^2 = X^2 \pmod{N}$ которое можно использовать для определения делителей числа N с помощью вычисления $\text{НОД}(Y_1 - 1, N)$. Однако это будет работать только если $Y_1 \neq \pm 1 \pmod{N}$.

Предположим, что $Y_1 = \pm 1 \pmod{N}$. Если $Y_1 = -1 \pmod{N}$, вернемся к началу и выберем другое число X . Если $Y_1 = 1 \pmod{N}$, то можно взять еще один квадратный корень:

$$Y_2 = \sqrt[Ed-1]{Y_1} \pmod{N}$$

Опять получаем

$$Y_2^2 = Y_1^2 = X^2 \pmod{N}$$

откуда $\text{НОД}(Y_2 - 1, N)$ – делитель числа N . К сожалению, здесь тоже может оказаться, что $Y_2 = \pm 1 \pmod{N}$. Тогда придется повторить все снова.

Алгоритм необходимо повторять до тех пор, пока не разложим N на множители и не придем к числу $(ED - 1)/2^t$, которое уже не

будет делиться на 2. В последнем случае придется вернуться к началу алгоритма, выбрать новое случайное значение X и все повторить.

Отсюда следует, что при известном d легко найти p и q .

Алгоритм, приведенный в доказательстве, – типичный пример алгоритма Лас-Вегаса, вероятностного по своей природе, которая проявляется в процессе работы. Но если уж ответ отыщется, то он будет обязательно верным.

Пример. Входные данные задачи RSA:

$N = 1441499$, $E = 17$ и $d = 507905$.

Напомним, что предполагаем известной расшифровывающую экспоненту d , которая обычно хранится в секрете. Опираясь на предыдущий алгоритм, найдем числа N . Положим

$$T_1 = (Ed - 1)/2 = 4317192,$$

$$X = 2.$$

Для вычисления Y_1 , сделаем преобразования:

~~$$Y_1 = (X^{T_1}) \bmod N = 1$$~~

Поскольку Y_1 оказался равным 1, нам нужно взять

$$T_2 = T_1/2 = (Ed - 1)/4 = 2158596 \text{ и } Y_2 = 2^{T_2}.$$

Теперь ~~$$Y_2 = (X^{T_2}) \bmod N = 1$$~~

Снова нужно повторять предыдущие шаги, что приведет нас к $T_3 = (Ed - 1)/8 = 1079298$

~~$$Y_3 = (X^{T_3}) \bmod N = 1$$~~

Отсюда ~~$$Y_4 = (X^{T_4}) \bmod N = 1$$~~

и можем найти простой делитель числа N , вычислив $\text{НОД}(Y_3 - 1, N) = 1423$.

Значение функции Эйлера $\varphi(N)$ и проблема факторизации.

Лемма. Значение $\Phi = \varphi(N)$ позволяет эффективно разложить число N на множители.

Доказательство. Имеем $\Phi = (p - 1)(q - 1) = N - (p + q) + 1$.

Следовательно, положив $S = N + 1 - \Phi$, получим $S = p + q$.

Нужно определить числа p или q , опираясь на их сумму S и произведение N . Рассмотрим многочлен

$$f(X) = (X - p)(X - q) = X^2 - SX + N.$$

Теперь можно найти как p , так и q , решая квадратное уравнение $f(X) = 0$ стандартным образом:

$$X^2 - SX + N = 0$$

Отсюда следует, что при известном $\varphi(N)$ легко найти p и q .

Пример. Открытый модуль $N = 18923$ криптосистемы RSA.

Известно $\Phi = \varphi(N) = 18648$. Вычисляем:

$$S = p + q = N + 1 - \Phi = 276.$$

Соответствующий многочлен имеет вид:

$$f(X) = X^2 - SX + N = X^2 - 276X + 18923,$$

а его корни – $p = 149$, $q = 127$.

Слабые моменты реализации RSA. Разделенный модуль.

Поскольку арифметика остатков – дорогое удовольствие с точки зрения компьютера, весьма заманчиво разработать систему шифрования [16], в которой пользователи разделяют общий модуль N , но применяют различные открытые и закрытые экспоненты (E_i, d_i) .

Одна из причин, побуждающая это делать, – ускорить алгоритмы зашифрования и расшифрования в аппаратных средствах, специально настроенных на определенный модуль N . Однако это неудачная идея, поскольку пасует перед двумя типами нападающих: внутреннего, т. е. законного пользователя системы, и внешнего.

Предположим, что нападающим является один из законных клиентов криптосистемы, – пользователь номер 1. Он может найти значение секретной экспоненты пользователя 2 – d_2 . Сначала он вычисляет p и q по известному ему d_1 . Затем злоумышленник находит $\varphi(N) = (p - 1)(q - 1)$ и, наконец, при помощи расширенного алгоритма Евклида раскрывает значение d_2 по формуле

$$d_2 = \frac{1}{E_2} \pmod{\varphi(N)}$$

Предположим, что теперь атакующий не принадлежит к пользователям криптосистемы, использующей общий модуль. Допустим, происходит рассылка одинакового (циркулярного) сообщения m двум клиентам криптосистемы, открытые ключи которых: (N, E_1) и (N, E_2) .

Противник, нападающий извне, видит зашифрованные сообщения C_1 и C_2 ,

где ~~1514~~

Он может вычислить

~~8189~~

и восстановить сообщение m по следующей схеме:

$$\tilde{N}_1^{T_1} \tilde{N}_2^{-T_2} = m^{E_1 T_1} m^{-E_2 T_2} = m^{1+E_2 T_2} m^{-E_2 T_2} = m^{1+E_2 T_2 - E_2 T_2} = m^1 = m.$$

Пример. $N = N_1 = N_2 = 18923$, $E_1 = 11$ и $E_2 = 5$.

Предположим, что перехвачены шифртексты $C_1 = 1514$ и $C_2 = 8189$, соответствующие одному открытому тексту m . Тогда противник вычисляет $T_1 = 1$ и $T_2 = 2$, после чего раскрывает исходную информацию: ~~1514~~

Вывод. При использовании общего модуля N любой законный пользователь системы может восстановить секретную экспоненту d другого пользователя, а внешний противник сможет читать циркулярные сообщения m .

Малые шифрующие экспоненты. Иногда в криптосистемах RSA с целью экономии затрат на шифрование используются небольшие шифрующие экспоненты E . Это тоже создает дополнительные проблемы, связанные с криптостойкостью. Предположим, что есть три пользователя с различными модулями шифрования N_1 , N_2 и N_3 и одинаковой шифрующей экспонентой $E = 3$. Пусть некто посылает им одно сообщение m , зашифрованное тремя разными открытыми ключами.

Нападающий видит три криптограммы:

$$C_1 = m^3 \pmod{N_1};$$

$$C_2 = m^3 \pmod{N_2};$$

$$C_3 = m^3 \pmod{N_3}$$

и с помощью китайской теоремы об остатках находит решение системы

$$\{X = C_i \pmod{N_i} \mid i = 1, 2, 3\}$$

в виде

$$X = m^3 \pmod{N_1 N_2 N_3}.$$

Но поскольку $m^3 < N_1 N_2 N_3$, целые числа X и m^3 должны совпадать. Поэтому, вычисляя обычный кубический корень из X , нападающий раскрывает сообщение m .

Пример. $N_1 = 323$, $N_2 = 299$, $N_3 = 341$

и перехваченные сообщения $C_1 = 50$, $C_2 = 299$, $C_3 = 1$.

Чтобы восстановить исходное сообщение m , находим решение системы $X = 300763 \pmod{N_1 N_2 N_3}$,

после чего извлекаем обычный кубический корень:

$$m = X^{1/3} = 67.$$

Вывод. При использовании малой шифрующей экспоненты E противник может восстановить циркулярное сообщение m для E пользователей.

Обе вышеуказанные атаки позволяют раскрыть текст, не прибегая к разложению на множители. Наиболее важный вывод из них – необходимо дополнять оригинальные сообщения случайными цифрами перед их зашифрованием.

При практической реализации RSA нужно решить следующие задачи:

1. Генерацию больших простых чисел p и q с проверкой выполнения соответствующих требований к ним.
2. Выбор экспонент e и d с соблюдением соответствующих требований.
3. Реализацию операции вычисления $x^y \pmod{n}$.

Генерация большого простого числа:

- 1) Сгенерировать случайное n -битовое число p .
- 2) Установить старший и младший биты равными единице. (Старший бит гарантирует требуемую длину простого числа, а младший – обеспечивает его нечетность).
- 3) Убедиться, что p не делится на малые простые числа: 3, 5, 7, 11 и т.д. Во многих практических реализациях проверяется делимость p на все простые числа, меньшие 256. Наиболее надежна проверка делимости на все простые числа, меньшие 2000. Эти проверки можно выполнить на основе массива этих простых чисел.
- 4) Выполнить тест простоты Рабина – Миллера для некоторого случайного числа a [22]. Если p проходит тест, сгенерировать другое случайное число a и повторить тест. Для ускорения проверки можно выбирать относительно небольшие значения a . Выполнить тест минимум пять раз. Если p не проходит один из тестов, то перейти к шагу 1.

Другой путь – не генерировать случайное число p в каждом тесте, а последовательно перебирать все нечетные числа, начиная со случайно выбранного числа, пока не найдется простое число.

Тест Рабина – Миллера. Выбрать для тестирования число p . Вычислить b – число делений $p - 1$ на 2 (то есть 2^b – это наибольшая степень числа 2, на которую делится $p - 1$). Затем вычислить m , такое, что $p = 1 + 2^b \cdot m$.

- 1) Выбрать случайное число a , меньшее p .
- 2) Установить $j = 0$ и $z = a^m \bmod p$.
- 3) Если $z = 1$ или если $z = p - 1$, то p проходит тест и может быть простым числом.
- 4) Если $j > 0$ и $z = 1$, то p не относится к простым числам.
- 5) Установить $j = j + 1$. Если $j < b$ и $z \neq (p - 1)$, установить $z = z^2 \bmod p$ и вернуться к шагу 4. Если $z = p - 1$, то p проходит тестирование и может быть простым числом.
- 6) Если $j = b$ и $z \neq (p - 1)$, то p не относится к простым числам.

Выбор экспонент e и d :

1) Так как единственное требование к e – отсутствие общих множителей с числом $\varphi(n) = (p - 1) \cdot (q - 1)$, то логично выбирать небольшое e , легко разложимое на простые множители и осуществлять проверку делением $\varphi(n)$ на множители, e . Либо выбирать простое e . Неплохо также выбирать e с минимальным количеством единиц в двоичном представлении, что ускорит операцию возведения в степень, реализованную следующим алгоритмом.

2) Так как очевидно, что $d = e^{-1}(\bmod p)$, то d находим при помощи расширенного алгоритма Евклида.

Алгоритм вычисления $x^y(\bmod n)$:

- 1) Представим y в двоичной системе счисления $y = y_0 2^r + \dots + y_{r-1} 2 + y_r$, где y_i , цифры в двоичном представлении равные нулю или единице, $y_0 = 1$.
- 2) Положим $x_0 = x$ и затем для $i = 1, \dots, r$ вычислим $x_i = x_{i-1}^2 \cdot x^{y_i}(\bmod n)$.
- 3) x_r есть искомый вычет $x^y(\bmod n)$.

Криптосистема Эль-Гамала. Односторонняя функция – возведение в степень с фиксированным модулем P и основанием G :

$$H = G^x \pmod{P}$$

Обратная задача – *задача дискретного логарифмирования*

$x = \text{ind}_{G,P} H$ – является сложной.

Шифрование по Эль-Гамалю использует конечные поля. Существует и аналогичная система на эллиптических кривых.

В отличие от RSA в алгоритме Эль-Гамалю имеются некоторые открытые параметры, которые могут быть применены большим числом пользователей. Они называются *параметрами домена* и выглядят следующим образом:

- P – большое простое число, т. е. число, насчитывающее около 1024 бит, такое, что $P - 1$ делится на другое, среднее простое число Q , лежащее неподалеку от 2^{160} ;
- G – элемент мультипликативной группы поля F_P , порядок которой делится на Q , причем $G^{(P-1)/Q} \pmod{P} \neq 1$.

Все параметры домена, т. е. P , Q и G , выбираются таким образом, чтобы элемент $G^{(P-1)/Q}$ был образующей абелевой группы A порядка Q . Информация об этой группе открыта и используется большим числом пользователей.

После выбора параметров домена определяют открытый и закрытый ключи. Закрытым ключом может априори быть любое натуральное число x , а открытый ключ получается по следующей формуле: $H = G^x \pmod{P}$.

Каждый из пользователей RSA должен генерировать два больших простых числа для определения ключевой пары, что является довольно громоздкой задачей, а в системе Эль-Гамалю для построения ключевой пары достаточно найти какое-нибудь случайное число и сделать сравнительно несложные вычисления в арифметике остатков [16].

Сообщение в этой системе представляется ненулевым элементом поля $m \in F_P$. Для его шифрования поступают следующим образом:

- генерируют случайный эфемерный ключ k ;
- вычисляют $C_1 = G^k$;
- находят $C_2 = m \cdot H^k$;
- выдают получившийся шифртекст в виде пары $C = (C_1, C_2)$.

При каждом шифровании применяется свой кратковременный ключ k . Поэтому, шифруя одно сообщение дважды, получают разные шифртексты.

Чтобы расшифровать пару данных $C = (C_1, C_2)$, производят следующие преобразования:

$$\frac{C_1 \cdot H^k}{G^k} \quad \frac{C_2 \cdot H^k}{G^k}$$

Пример. $Q = 101$, $P = 809$ и $G = 3$.

Легко проверить, что Q действительно делит число $P - 1$, а порядок элемента G в группе F_P^* делится на Q . Порядок элемента G равен 808, поскольку $3^{808} = 1 \pmod{P}$, и ни при каких меньших степенях такого равенства не получается. В качестве пары открытого и закрытого ключа выберем $x = 68$ и

$H = G^x = 3^{68} = 65 \pmod{P}$. Допустим, нам нужно зашифровать сообщение, численное представление которого равно: $m = 100$. Поступаем следующим образом.

- Генерируем случайный эфемерный ключ $k = 89$.
- Находим $C_1 = G^k = 3^{89} = 345 \pmod{P}$.
- Получаем $C_2 = m \cdot H^k = 100 \cdot 65^{89} = 517 \pmod{P}$.
- Отправляем шифртекст $C = (345, 517)$.

Адресат сможет восстановить текст, делая также вычисления:

$$\frac{C_2 \cdot 517}{C_1 \cdot 345}$$

Последнее равенство получается чуть более сложно, чем в вещественных числах: сначала число 345 возводится в степень 68 по модулю 809, вычисляется мультипликативный обратный к результату по этому же модулю, а затем найденный обратный умножается на 517.

Криптосистема Рабина. Система базируется на трудности задачи факторизации больших целых чисел, а точнее на трудности извлечения квадратного корня по модулю составного числа

$$N = p \cdot q.$$

Эти задачи эквивалентны:

- зная простые делители числа N , можно легко извлекать корни по модулю N ;

- умея извлекать корни по модулю N , можно легко разложить N на составные множители.

Такую систему можно считать в некотором отношении более криптостойкой, чем RSA. Процесс шифрования в системе Рабина происходит намного быстрее, чем практически в любой другой криптосистеме с открытым ключом [22]. Однако, несмотря на эти преимущества, система Рабина используется все же реже, чем RSA.

Выбираются два простых числа p и q , удовлетворяющие условию $p = q = 3 \pmod{4}$ [22].

Такой специальный вид чисел сильно ускоряет процедуру извлечения корней по модулю p и q . Закрытым ключом системы является пара (p, q) .

Для определения соответствующего открытого ключа берут произведение $N = p \cdot q$ и генерируют случайное целое число

$B \in \{0, \dots, N-1\}$. Открытый ключ – это пара (N, B) .

Для шифрования сообщения m в алгоритме Рабина вычисляют $C = m(m + B) \pmod{N}$ [22].

Таким образом, шифрование состоит из операций сложения и умножения по модулю N , что обеспечивает более высокую скорость шифрования, чем в RSA, даже если в последней выбирают небольшую шифрующую экспоненту.

Расшифрование в этом алгоритме гораздо более сложное. По существу, нужно вычислить

$$m = \sqrt{\frac{C - B}{4}} \pmod{N}$$

На первый взгляд, здесь не требуется никакой секретной информации, но для извлечения корней по модулю N очень и очень полезно знать разложение N на простые множители. Поскольку N – произведение двух простых чисел, существует четыре возможных квадратных корня из числа по модулю N . Поэтому при расшифровании получается четыре возможных варианта открытого текста. Чтобы выбор был более определенным, стоит к открытому тексту добавлять некую избыточную информацию.

Объясним, почему при расшифровании действительно получаем m . Напомним, что шифртекст имеет вид

$$C = m(m + B) \pmod{N}.$$

Поэтому

$$\begin{aligned} \sqrt{\frac{B^2}{4} + C} - \frac{B}{2} &= \sqrt{\frac{B^2 + 4m(m+B)}{4}} - \frac{B}{2} = \\ &= \sqrt{\frac{4m^2 + 4Bm + B^2}{4}} - \frac{B}{2} = \\ &= \sqrt{\frac{(2m+B)^2}{4}} - \frac{B}{2} = \frac{2m+B}{2} - \frac{B}{2} = m \end{aligned}$$

Конечно, предполагаем, что выбрали правильный корень из четырех.

Пример. Открытый и закрытый ключи имеют вид:

- $p = 127, q = 131,$

- $N = 16637$ и $B = 12345.$

Для шифрования сообщения $m = 4410$ вычисляем

$$C = m(m + B) \pmod{N} = 4633.$$

При обратной операции сначала находим

$$T = B^2/4 + C \pmod{N} = 1500, \text{ а затем извлекаем квадратные}$$

корни из T по модулю p и q :

$$\sqrt{T \pmod{p}} = \pm 22 \pmod{p}$$

После этого, применяя китайскую теорему об остатках к паре $\pm 22 \pmod{p}$ и $\pm 37 \pmod{q}$, находим квадратный корень из T по модулю N : $\sqrt{T \pmod{N}} = \pm 4410$

Четыре варианта расшифрования: 4410, 5851, 15078, или 16519, – получаются из формулы

$$s = \frac{B}{2} \pm \frac{1}{2} \sqrt{\frac{B^2}{4} + C}.$$

Рюкзачные криптосистемы. Задача об укладке рюкзака.

Задано множество $\{v_i\}$ из k натуральных чисел и целое число S .

Требуется найти такое k -разрядное число $n = (n_{k-1}n_{k-2} \dots n_1n_0)_2$ (где $n_i \in \{0, 1\}$ – суть значения разрядов в двоичной записи числа n), что $\sum_{i=0}^{k-1} n_i v_i = S$, если такое n существует.

В зависимости от набора $\{v_i\}$ и числа S такого решения может не быть вообще, решений может быть несколько или единственное. Такая задача является сложной.

Частный случай – задача об укладке быстровозрастающего рюкзака. Это случай, когда величины v_i , будучи упорядочены в

порядке возрастания, обладают тем свойством, что каждое число больше суммы всех предыдущих.

Такая задача имеет эффективный алгоритм решения:

1. Положим $W = S$ и $j = k$.
2. Начиная с n_{j-1} и последовательно уменьшая j , полагаем все $n_j = 0$ до тех пор, пока не придем к первому такому значению j (обозначим его через i_0), что $v_{i_0} \leq W$. Положим $n_{i_0} = 1$.
3. Заменяем W на $W - v_{i_0}$, положим $j = i_0$ и, если $W > 0$, переходим к шагу 2.
4. Если $W = 0$, то цель достигнута. Если $W > 0$ и все оставшиеся v_i больше W , то ясно, что решения $n = (n_{k-1}n_{k-2}\dots n_1n_0)_2$ задачи не существует. Если решение есть, то оно единственное.

Пример. Набор $\{2, 3, 7, 15, 31\}$ является быстрорастущим набором, а $S = 24$. Тогда, проходя пятерку значений справа налево, видим, что $n_4 = 0$ ($31 > 24$), $n_3 = 1$ ($15 < 24$, и в этот момент заменяем 24 на $24 - 15 = 9$), $n_2 = 1$ ($7 < 9$, и в этот момент заменяем 9 на $9 - 7 = 2$), $n_1 = 0$ ($3 > 2$), $n_0 = 1$ ($2 = 2$). Таким образом, получаем $n = (01101)_2 = 13$.

Криптосистема Меркля – Хеллмана. Предположим, что элементы открытого текста имеют в качестве своих числовых эквивалентов k -разрядные двоичные числа n .

Каждый пользователь выбирает быстрорастущий набор $\{v_0, \dots, v_{k-1}\}$, целое число $m > \sum_{i=0}^{k-1} v_i$ и целое число a , такое что $a < 0 < m$ и $\text{НОД}(a, m) = 1$.

После этого вычисляются $b = a^{-1} \pmod{m}$ и k -элементный набор $\{W_i\} = \{W_0, \dots, W_{k-1}\}$, определяемый равенством $W_i = av_i \pmod{m}$. Пользователь держит числа $\{v_i\}$, m , a и b в секрете, а набор $\{W_i\}$ делает общеизвестным. Таким образом, ключом зашифрования является набор $\{W_0, \dots, W_{k-1}\}$, а ключом расшифрования – пара (b, m) , которая вместе с ключом зашифрования позволяет определить набор $\{v_0, \dots, v_{k-1}\}$.

Желающий передать сообщение $n = (n_0 \dots n_{k-1})_2$ пользователю с ключом шифрования $\{W_i\}$ вычисляет

$$N = \sum_{i=0}^{k-1} n_i W_i$$

и передает это число.

Чтобы прочесть это сообщение, пользователь сначала находит $s = bC$:



поскольку $bW_i \equiv bav_i \equiv v_i \pmod{m}$.

Теперь можно воспользоваться приведенным выше алгоритмом для быстровозрастающего рюкзака и найти единственное решение $(n_0 \dots n_{k-1})_2 = n$ задачи о подмножестве $\{v_i\}$ с суммой, равной s .

Пример. Элементы открытого текста – двоичные представления букв 26-буквенного алфавита от «А» = 0 = $(00000)_2$ до «Z» = 25 = $(11001)_2$.

Секретный быстровозрастающий набор $\{v_0, \dots, v_{k-1}\} = \{2, 3, 7, 15, 31\}$.

Выберем $m = 61$, $a = 17$, тогда $b = 18$, и открытый ключ шифрования:

$\{W_0, \dots, W_{k-1}\} = \{34, 51, 58, 11, 39\}$.

Чтобы послать сообщение «WNY», корреспондент должен вычислить:

«W» = $(10110)_2 \rightarrow 51 + 58 + 39 = 148$;

«H» = $(00111)_2 \rightarrow 34 + 51 + 58 = 143$;

«Y» = $(11000)_2 \rightarrow 11 + 39 = 50$.

$N = n_1, n_2, n_3 = 148, 143, 50$.

Чтобы прочитать сообщение N , сначала умножают эти числа на 18 и приводят результаты по модулю 61, получают $S = s_1, s_2, s_3 = 41, 12, 46$. Далее, пользуясь алгоритмом для быстрорастущего рюкзака для всех s_i , восстанавливают сообщение:

$(10110)_2, (00111)_2, (11000)_2$

Шифрсистема Мак-Элиса. 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.

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 $G = G_{k \times n}$ двоичного (n, k) -линейного кода, исправляющего t ошибок, для которого известен эффективный алгоритм декодирования.

2) Случайно выбрать двоичную невырожденную матрицу $S = S_{k \times k}$.

3) Случайно выбрать подстановочную матрицу $P = P_{n \times n}$.

4) Вычислить произведение матриц $G_1 = S \cdot G \cdot P$.

Открытым ключом является пара (G_1, t) , секретным – тройка (S, G, P) .

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

1) Представить M в виде двоичного вектора длины k .

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

3) Вычислить бинарный вектор $C = M \cdot G_A + Z$ и направить его абоненту A .

Получив сообщение C , абонент A вычисляет вектор $C_1 = C \cdot P^{-1}$, с помощью которого, используя алгоритм декодирования кода с порождающей матрицей G , получает далее векторы

M_1 и $M = M_1 \cdot S^{-1}$.

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 $GF(2^m)$ существует бинарный код Гоппы длиной $n = 2^m$ и размерностью $k \geq n - mt$, исправляющий до t ошибок включительно, для которого имеется эффективный алгоритм декодирования. В настоящее время не известны эффективные алгоритмы дешифрования системы Мак-Элиса, использующей код Гоппы, при правильном выборе параметров системы.

Рекомендуемые параметры этой системы – $n = 1024, t = 38$,

$k > 644$ – приводят к тому, что открытый ключ имеет размер около 2^{19} бит, а длина сообщения увеличивается при шифровании примерно в 1,6 раза, в связи с чем данная система не получила широкого распространения.

Контрольные вопросы

1. Почему операции для криптографических преобразований должны обладать свойством замкнутости?
2. Приведите случаи, при которых функция Эйлера легко вычислима.
3. В чем различие между кольцом вычетов по модулю натурального числа и простым конечным полем?
4. Для чего используется расширенный (обобщенный) алгоритм Евклида?
5. Для решения какого вида систем сравнений используется китайская теорема об остатках?
6. В чем различие между символами Лежандра и Якоби?
7. Укажите два способа извлечения квадратных корней в простом конечном поле?
8. Может ли в криптосистеме RSA шифрующая экспонента быть четной?
9. Какие требования к модулю P предъявляются в криптосистеме Эль-Гамала?
10. Сколько вариантов расшифрования сообщения дает криптосистема Рабина?

5. КРИПТОГРАФИЧЕСКИЕ ХЕШ-ФУНКЦИИ И ЭЛЕКТРОННО-ЦИФРОВАЯ ПОДПИСЬ

5.1. Криптографические хеш-функции

Хеш-функции – это функции, предназначенные для «сжатия» произвольного сообщения или набора данных, записанного, как правило, в двоичном алфавите, в некоторую битовую комбинацию фиксированной длины, называемую *сверткой*.

В криптографии хеш-функции применяются для задач:

- построения систем контроля целостности данных при их передаче или хранении;
- аутентификации источника данных.

При решении первой задачи для каждого набора данных вычисляется значение хеш-функции (называемое *кодом аутентификации сообщения* или *имитовставкой*), которое передается или хранится вместе с самими данными. При получении данных пользователь вычисляет значение свертки и сравнивает его с имеющимся контрольным значением. Несовпадение говорит о том, что данные были изменены. Для выработки имитовставки обычно применяют хеш-функции, значение которых зависит от секретного ключа пользователя. Этот ключ должен быть известен передающей и проверяющей сторонам. Такие хеш-функции называют *ключевыми*.

Имитовставки, формируемые с помощью ключевых хеш-функций, не должны позволять противнику создавать поддельные (сфабрикованные) сообщения (*fabrication*) при атаках типа *имитация* (*impersonation*) и модифицировать передаваемые сообщения (*modification*) при атаках типа *подмена* (*substitution*).

При решении второй задачи – аутентификации источника данных, – обмен данными происходит между не доверяющими друг другу сторонами. В такой ситуации применяют схемы цифровой подписи, позволяющие осуществлять аутентификацию источника данных. Как правило, при этом сообщение, прежде чем быть подписано личной подписью, основанной на секретном ключе пользователя, «сжимается» с помощью хеш-функции, выполняющей функцию кода обнаружения ошибок. В данном

случае хеш-функция не зависит от секретного ключа и может быть фиксирована и известна всем. Основными требованиями к ней являются гарантии невозможности подмены подписанного документа, а также подбора двух различных сообщений с одинаковым значением хеш-функции (в этом случае говорят, что такая пара сообщений образует *коллизия*).

Блочно-итерационные и шаговые функции. Пусть X – множество сообщений (последовательности символов некоторого алфавита, как правило, двоичного); Y – множество двоичных векторов фиксированной длины.

Хеш-функцией называется всякая функция $h: X \rightarrow Y$, легко вычисляемая и такая, что для любого сообщения M значение

$h(M) = H$ (*свертка*) имеет фиксированную битовую длину.

Обычно число возможных сообщений значительно превосходит число возможных значений сверток, в силу чего для каждого значения свертки имеется большое множество прообразов, т. е. сообщений с заданным значением хеш-функции. При случайном и равновероятном выборе сообщений условие равномерности распределения значений хеш-функции эквивалентно наличию одинакового числа прообразов для каждого значения свертки.

Как правило, хеш-функции строят на основе так называемых *одношаговых сжимающих функций* $y = f(x_1, x_2)$ двух переменных, где x_1 и y – двоичные векторы длины m и n соответственно, причем n – длина свертки. Для получения значения $h(M)$ сообщение M сначала разбивается на блоки длины m (при этом если длина сообщения не кратна m , то последний блок неким специальным образом дополняется до полного), а затем к полученным блокам

M_1, M_2, \dots, M_N применяют следующую последовательную процедуру вычисления свертки:

$$H_0 = v,$$

$$H_i = f(M_i, H_{i-1}), i = 1, \dots, N,$$

$$h(M) = H_N.$$

Здесь v – некоторый фиксированный начальный вектор. Если функция f зависит от ключа, то этот вектор можно положить равным нулевому вектору. Если же функция f не зависит от ключа, то для исключения возможности перебора коротких сообщений

(при попытках обращения хеш-функции) этот вектор можно составить из фрагментов, указывающих дату, время, номер сообщения и т. п. Такие хеш-функции называют *блочными-итерационными*. При данном подходе свойства хеш-функции h полностью определяются свойствами одношаговой сжимающей функции.

Особо выделяют два важных типа криптографических хеш-функций – *ключевые и бесключевые*. Первые применяются в системах с симметричными ключами. Ключевые хеш-функции называют *кодами аутентификации сообщений (КАС) (message authentication code (MAC))*. Они дают возможность без дополнительных средств гарантировать как правильность источника данных, так и целостность данных в системах с доверяющими друг другу пользователями.

Бесключевые хеш-функции называются *кодами обнаружения ошибок (modification detection code (MDC) или manipulation detection code, message integrity code (MIC))*. Они дают возможность с помощью дополнительных средств (шифрование, использование защищенного канала или цифровой подписи) гарантировать целостность данных. Эти хеш-функции могут применяться в системах как с доверяющими, так и не доверяющими друг другу пользователями.

Ключевые функции хеширования. В криптографических приложениях к ключевым функциям хеширования предъявляются следующие требования:

- невозможность фабрикаций;
- невозможность модификации.

Первое требование означает высокую сложность подбора сообщения с правильным значением свертки; второе – высокую сложность подбора для заданного сообщения с известным значением свертки другого сообщения с правильным значением свертки.

Иногда эти свойства объединяют в одно более сильное свойство – *вычислительную устойчивость*. Это требование означает высокую сложность подбора для заданного множества сообщений $\{x_1, \dots, x_t\}$ (быть может, пустого) с известными значениями сверток еще одного сообщения x , $x \neq x_i$, $i = 1, \dots, t$, с

правильным значением свертки (возможен случай $h(x) = h(x_i)$, $i \in \{1, \dots, t\}$).

Словосочетание «высокая сложность» означает такую вычислительную сложность задачи, при которой ее решение с использованием вычислительной техники за реальное время невозможно.

Ключевые функции применяются в ситуациях, когда стороны доверяют друг другу и могут иметь общий секретный ключ. Обычно в этих условиях не требуется, чтобы система обеспечивала защиту в случае отказа получателя от факта получения сообщения или его подмены. Поэтому от ключевых хеш-функций не требуется устойчивости к коллизиям.

Обычные атаки на ключевые хеш-функции заключаются в имитации, т. е. в передаче сфабрикованных сообщений в пустом канале, а также в подмене передаваемых сообщений с целью навязывания приемной стороне ложных сообщений.

В качестве примера рассмотрим широко распространенную хеш-функцию, построенную на основе одношаговой сжимающей функции вида

$$f_k(x, H) = E_k(x \oplus H),$$

где E_k – алгоритм блочного шифрования.

Для вычисления значения $h(M)$ сообщение M представляется в виде последовательности n -битовых блоков M_1, M_2, \dots, M_N . Если при этом длина сообщения не кратна длине одного блока, то последний блок неким специальным образом дополняется до полного блока.

Алгоритм вычисления свертки имеет следующий вид:

$$H_0 = 0;$$

$$H_i = E_k(M_i \oplus H_{i-1}), i = 1, \dots, N;$$

$$h(M) = H_N.$$

Данный алгоритм фактически совпадает с режимом шифрования со сцеплением блоков СВС с той лишь разницей, что в качестве результата берется не весь шифртекст H_1, H_2, \dots, H_N , а только его последний блок. Такой режим в ГОСТ 28147-89 называется *режимом выработки имитовставки*.

Еще одной основой для построения ключевых хеш-функций могут служить бесключевые хеш-функции. При этом для

вычисления значения свертки ключ приписывается к исходному сообщению.

Если ключ просто дописывать в начало или в конец исходного сообщения, то это может приводить к потенциальным слабостям, позволяющим в некоторых случаях осуществлять модификацию сообщений. В связи с этим предпочтительными являются способы введения ключа, при которых ключ вставляется в сообщение не один, а, по крайней мере, два раза:

$$H = h(k, y, M, k) \quad \text{или} \quad H = h(k, y_1, h(k, y_2, M)),$$

где y , y_1 и y_2 – дополнения ключа k до размера, кратного длине блока n . Для определенных бесключевых хеш-функций h такой подход позволяет строить эффективно вычисляемые и устойчивые к атакам ключевые хеш-функции. Недостатком такого метода является слишком большая длина n свертки. Дело в том, что для целей проверки целостности обычно выбирают длину свертки n в пределах 32 – 64, а для аутентификации необходимо условие $n > 128$.

Существуют также ключевые хеш-функции, не использующие какую-либо основу типа блочного шифрования или вычисления бесключевой хеш-функции, а разработанные независимо, с учетом эффективной реализации на современных ЭВМ, например ключевая хеш-функция, используемая в алгоритме МАА (*Message Authenticator Algorithm*), утвержденном стандартом ISO 8731-2.

Бесключевые функции хеширования. Как правило требуется, чтобы бесключевые хеш-функции обладали такими свойствами, как:

- 1) однонаправленность, (по $Y = h(X)$ трудно определить X);
- 2) устойчивость к коллизиям, (по X трудно найти X' , такое, что $h(X) = h(X')$);
- 3) устойчивость к нахождению второго прообраза (трудно найти пару X, X' , такие, что $h(X) = h(X')$).

Например, хеш-функция CRC-32, представляющая собой контрольную сумму, является линейным отображением и поэтому не удовлетворяет ни одному из этих трех свойств.

Использование в качестве бесключевой хеш-функции функции, построенной на основе алгоритма блочного шифрования

в режиме выработки имитовставки, также нецелесообразно, так как обратимость блочного шифрования позволяет подбирать входное сообщение для любого значения свертки при фиксированном и общеизвестном ключе.

Для построения примера хеш-функции, удовлетворяющей свойству 1, рассмотрим функцию, заданную формулой

$$g_k(x) = E_k(x) \oplus x, \text{ где } E_k - \text{алгоритм блочного шифрования.}$$

Такая функция является однонаправленной по обоим аргументам. Поэтому на ее основе можно построить хеш-функцию, определив одношаговую сжимающую функцию одной из следующих формул:

$$f(x, H) = E_H(x) \oplus x; f(x, H) = E_x(H) \oplus H.$$

Первая из этих функций лежит в основе российского стандарта хеш-функции, а вторая – в основе американского стандарта SHA.

Обе приведенные выше шаговые функции хеширования принадлежат семейству:

$$f(X, Y) = E_{\alpha_1 X \oplus \alpha_2 Y}(\beta_1 X \oplus \beta_2 Y) \oplus \gamma_1 X \oplus \gamma_2 Y, \quad \alpha_i, \beta_i, \gamma_i \in \{0, 1\}.$$

Не все функции данного семейства стойки к коллизиям. Стойкие к известным методам криптоанализа конструкции шаговых функций хеширования сведены в табл 8.

Таблица 8.

Шаговые функции хеширования $f(X, Y)$, использующие блочное шифрование E_k (длина блока равна длине ключа, или ключ дополнен до длины блока)

Шаг 1	Шаг 2	Шаг 3
$E_Y(X) \oplus X$	$E_X(Y) \oplus Y$	$E_{X \oplus Y}(X) \oplus X$
$E_Y(X \oplus Y) \oplus X \oplus Y$	$E_X(X \oplus Y) \oplus X \oplus Y$	$E_{X \oplus Y}(Y) \oplus Y$
$E_Y(X) \oplus X \oplus Y$	$E_X(Y) \oplus X \oplus Y$	$E_{X \oplus Y}(X) \oplus Y$
$E_Y(X \oplus Y) \oplus X$	$E_X(X \oplus Y) \oplus Y$	$E_{X \oplus Y}(Y) \oplus X$

Трудоемкость подбора прообраза для однонаправленной функции или трудоемкость поиска второго прообраза оцениваются величиной $O(2^n)$. В то же время трудоемкость поиска коллизии оценивается величиной $O(2^{n/2})$, так как в данной ситуации применима атака, основанная на парадоксе «дней рождений»

Значением любой из хеш-функций, построенных из приведенных одношаговых сжимающих функций, является вектор длиной n , равной размеру блока. В случае если эта величина оказывается недостаточной, ее можно увеличить, заменив одношаговую функцию f на функцию f' с удвоенной размерностью значений. Это можно сделать, например, путем двукратного применения функции f с последующим перемешиванием полублоков согласно формуле

$$f'(x, H_1, H_2) = \pi(f(x, H_1), f(x, H_2)),$$

в которой π переставляет произвольные полублоки a, b, c, d по правилу $\pi((a,b),(c,d)) = (a,d,c,b)$. Такой подход реализован в конструкции одношаговой функции MDC-2.

Другие примеры бесключевых хеш-функций дают известные алгоритмы MD-4, MD-5 и SHA. Они оперируют с блоками длиной n , совпадающей с длиной результирующего значения свертки, причем $n = 128$ для алгоритма MD-4 и $n = 160$ для MD-5 и SHA. Указанные алгоритмы спроектированы специально с учетом эффективной реализации на 32-разрядных ЭВМ.

При их использовании исходное сообщение M разбивается на блоки длиной $m = 512$ бит. Последний блок формируется путем дописывания к концу сообщения комбинации 10...0 до получения блока размером 448 бит, к которому затем добавляется комбинация из 64 бит, представляющая битовую длину сообщения. Затем вычисляется значение свертки с использованием одношаговой сжимающей функции заданной формулой:

$$f(x, H) = E_x(H) \oplus H,$$

где x – блок сообщения длиной $m = 512$ бит;

H – блок из n бит;

E_x – некоторое преобразование множества блоков.

Значение начального вектора определяется в описании преобразования E_x . В стандарте хеш-функции ГОСТ Р 34.11-94 приняты значения $n = m = 512$. Одношаговая сжимающая функция $f(x, H)$, используемая для вычисления последовательности значений $H_i = f(x_i, H_{i-1})$, построена на базе четырех параллельно работающих схем блочного шифрования (ГОСТ 28147-89), каждая из которых имеет 256-битовый ключ и оперирует с блоками размером 64 бита. Каждый из ключей вычисляется в соответствии с некоторой линейной функцией от блока исходного сообщения x_i и значения

H_{i-1} . Значение H_i является линейной функцией от результата шифрования блока исходного сообщения x_i и значения H_{i-1} . После вычисления значения H_N для последовательности блоков M_1, M_2, \dots, M_N применяют еще два шага вычисления согласно формуле

$$H = h(M) = f(Z \oplus M_N, f(L, H_N)),$$

где Z – сумма по модулю два всех блоков сообщения;

L – длина сообщения.

Схемы использования ключевых и бесключевых функций.

Схемы использования ключевых хеш-функций (кодов аутентичности сообщений) приведены на рис. 34.

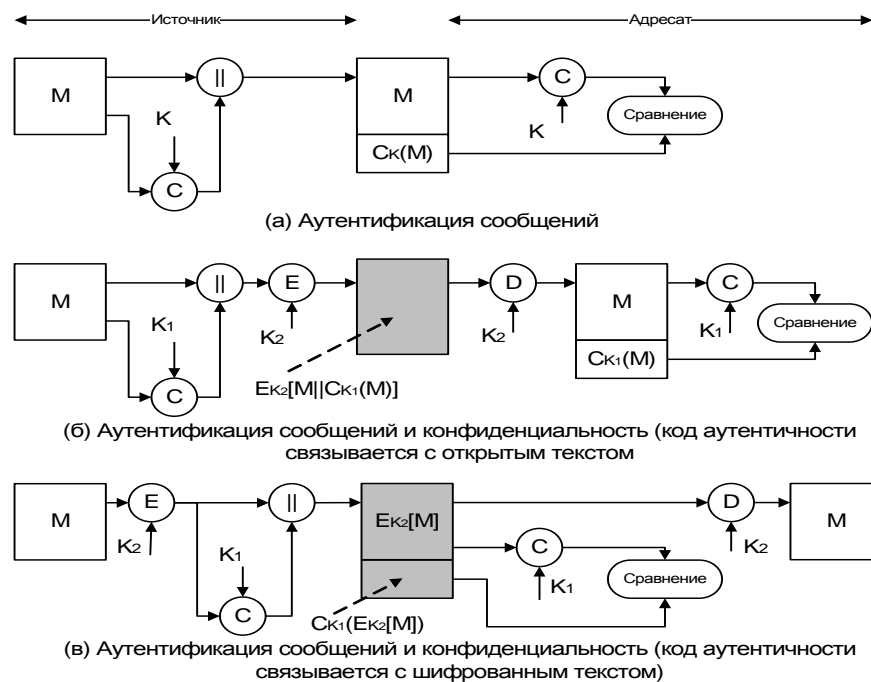


Рис.34. Схемы применения ключевых хеш-функций.

Схема *a* применяется в случае необходимости обеспечить целостность открытых сообщений, а схемы *б* и *в* – в случаях передачи шифрованных сообщений.

Случаи использования кодов аутентичности сообщений:

1. Циркулярная рассылка сообщения нескольким адресатам. Аутентичность проверяет корневой узел на основе известного ему секретного ключа.
2. Аутентификация сообщений проводится принимающей стороной на выборочной основе из-за загруженности.
3. Аутентификация компьютерных программ (в частности, антивирусная).

4. Важность обеспечить аутентичность управляющих системных запросов.

5. Разделение функций аутентификации и конфиденциальности.

6. Защита целостности сообщения после его получения, при хранении в открытом виде.

Схемы использования бесключевых хеш-функций приведены на рис.35.

В схемах *a* и *б* хеш-функция используется совместно с симметричной системой шифрования.

В схемах *в* и *г* хеш-функция используется совместно с асимметричной системой шифрования.

В схемах *д* и *е* хеш-функция используется совместно с симметричной системой шифрования как код аутентичности сообщения с секретным элементом *S*.

В схемах *a*, *г*, *е* обеспечивается и аутентичность и конфиденциальность, а в схемах *б*, *в*, *д* – только аутентичность.

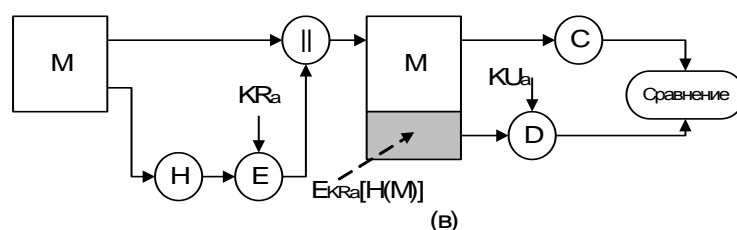
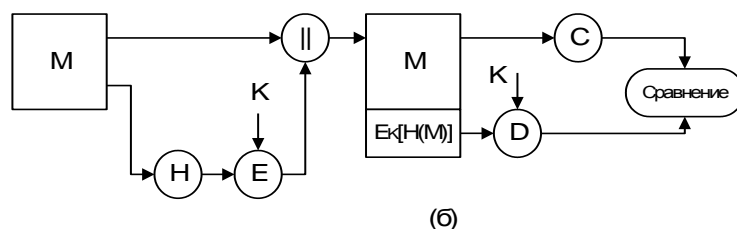
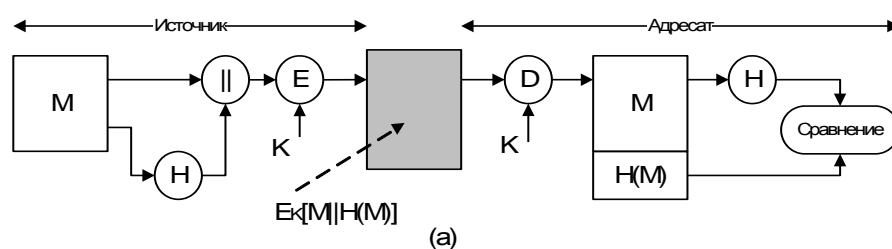
В табл. 9 даны сведения по основным современным алгоритмам хеширования

Таблица 9.

Современные алгоритмы хеширования.

№ пп	Название	Длина свертки (бит)	Тип	Основа алгоритма.
1	2	3	4	5
1	Snefru	128, 256	бесключевая	Сжимающая функция на основе блочного 512-раз-рядного шифрования. два-три прохода в 64 раунда
2	MD4	128	бесключевая	Оригинальная блочно-итерационная 512-разрядная функция (три раунда)
3	MD5	128	бесключевая	Оригинальная блочно-итерационная 512-разрядная функция (четыре раунда)
4	SHA	160, 256, 512	бесключевая	Оригинальная блочно-итерационная 512-разрядная функция (четыре раунда)
5	RIPEMD	128, 160	бесключевая	Разновидность MD4
6.	HAVAL	Переменная	бесключевая	Разновидность MD5

1	2	3	4	5
7	MDC-2	Две длины блока (DES)	бесключевая	Две ветви блочного криптоалгоритма
8	MDC-4	Две длины блока (DES)	бесключевая	Две ветви блочного криптоалгоритма
9	ГОСТ	256 (четыре длины блока ГОСТ)	бесключевая	Четыре ветви блочного криптоалгоритма ГОСТ
10	Схема Девиса – Майера	128 (две длины блока IDEA)	бесключевая	Две ветви блочного криптоалгоритма IDEA
11	СВС-МАС	Длина блока	ключевая	Блочный криптоалгоритм
12	МАС	32	ключевая	Блочный криптоалгоритм
13	НМАС	Переменная	ключевая	Бесключевая хеш-функция MD5, SHA или RIPEMD



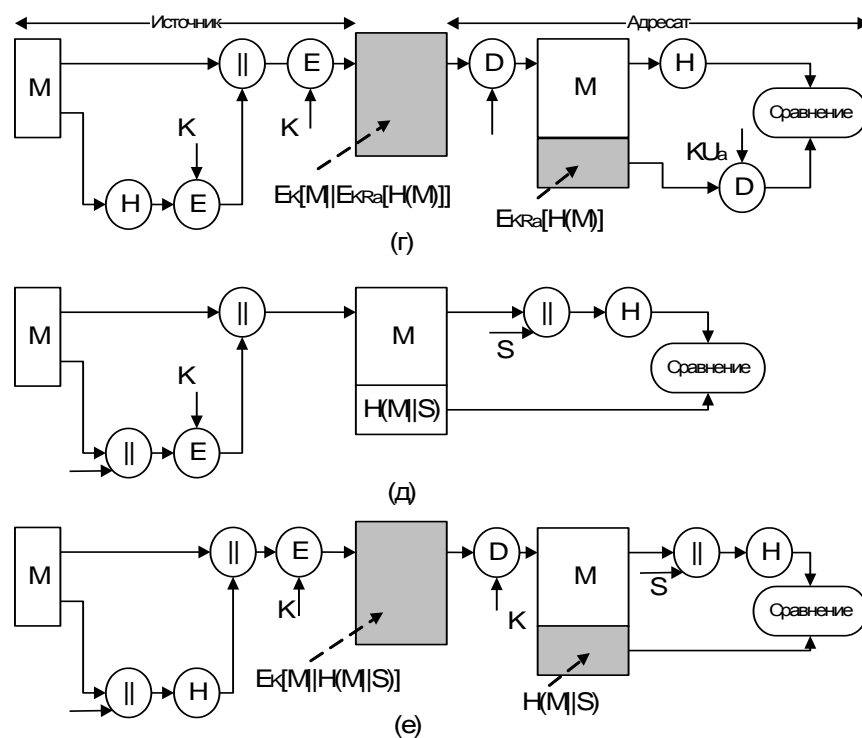


Рис.35. Схемы применения бесключевых хеш-функций.

5.2. Электронно-цифровая подпись.

Задачи и особенности электронно-цифровой подписи.

Цифровая подпись для сообщения является числом, зависящим от самого сообщения и от некоторого секретного, известного только подписывающему субъекту, ключа. При этом предполагается, что она должна быть легко проверяемой и что осуществить проверку подписи должен иметь возможность каждый без получения доступа к секретному ключу. При возникновении спорной ситуации, связанной с отказом подписывающего от факта подписи им некоторого сообщения либо с попыткой подделки подписи, третья сторона должна иметь возможность разрешить спор.

Цифровая подпись позволяет решить следующие задачи:

- осуществить аутентификацию источника сообщения;
- установить целостность сообщения;
- обеспечить невозможность отказа от факта подписи

конкретного сообщения.

Использование термина «подпись» в данном контексте оправдано тем, что цифровая подпись имеет много общего с обычной собственноручной подписью на бумажном документе. Собственноручная подпись также решает три перечисленные

задачи, однако между обычной и цифровой подписями имеются существенные различия (табл. 10).

Таблица 10.

Особенности собственноручной и цифровой подписи.

<i>Собственноручная подпись</i>	<i>Цифровая подпись</i>
Не зависит от подписываемого текста, всегда одинакова	Зависит от подписываемого текста, практически всегда разная
Неразрывно связана с подписывающим лицом, однозначно определяется его психофизическими свойствами, не может быть утеряна	Определяется секретным ключом, принадлежащим подписывающему лицу, может быть утеряна владельцем
Неотделима от носителя (бумаги), поэтому отдельно подписывается каждый экземпляр документа	Легко отделима от документа, поэтому верна для всех его копий
Не требует для реализации дополнительных механизмов	Требует дополнительных механизмов, реализующих алгоритмы ее вычисления и проверки
Не требует создания поддерживающей инфраструктуры	Требует создания доверенной инфраструктуры сертификатов открытых ключей

Для реализации схемы цифровой подписи необходимы два алгоритма:

- алгоритм вычисления цифровой подписи;
- алгоритм ее проверки.

Главные требования к этим алгоритмам заключаются в исключении возможности получения подписи без использования секретного ключа и гарантировании возможности проверки подписи без знания какой-либо секретной информации.

Надежность схемы цифровой подписи определяется сложностью следующих задач:

- *подделки подписи, т. е.* нахождения значения подписи под заданным документом лицом, не являющимся владельцем секретного ключа;
- *создания подписанного сообщения, т. е.* нахождения хотя бы одного сообщения с правильным значением подписи;
- *подмены сообщения, т. е.* подбора двух различных сообщений с одинаковыми значениями подписи.

Принципиальной сложностью, возникающей при использовании цифровой подписи на практике, является проблема создания *инфраструктуры открытых ключей*.

Сложилась практика заключения договоров между участниками информационного взаимодействия с применением цифровых подписей. В таком договоре должно быть четко указано:

- кто должен нести ответственность в случае, если подписанные сделки не состоятся;
- кто должен нести ответственность в случае, если система окажется ненадежной и будет взломана, т. е. будет выявлен факт подделки секретного ключа;
- какова ответственность уполномоченного по сертификатам в случае, если открытый ключ будет сфальсифицирован;
- какова ответственность владельца секретного ключа в случае его утраты;
- кто несет ответственность за плохую реализацию системы в случае повреждения или разглашения секретного ключа;
- каков порядок разрешения споров и т. п.

Имеется несколько принципиально различных подходов к созданию схем цифровой подписи. Их можно разделить на три группы:

- 1) схемы на основе систем шифрования с открытыми ключами;
- 2) схемы со специально разработанными алгоритмами вычисления и проверки подписи;
- 3) схемы на основе симметричных систем шифрования.

Ассиметричные алгоритмы цифровой подписи на основе RSA. Существуют две основные схемы асимметричной цифровой подписи:

- схема с восстановлением сообщения (при проверке подписи восстанавливается исходный текст);
- схема с дополнением (подпись передается вместе с исходным текстом).

Формально схема цифровой подписи состоит из двух преобразований:

- секретного преобразования подписи s ;
- открытого преобразования проверки V .

Схема с восстановлением сообщения. Абонент A , посылая сообщение M , вычисляет $S = s(M)$ и передает результат S , где S – цифровая подпись на сообщении M . Секретность сообщения в

Таблица 11.

Современные алгоритмы цифровой подписи.

№пп	Название	Схема	Примечание
1	DSA	Эль-Гамалья, Шнорра	Стандарт DSS
2	ECDSA	Эль-Гамалья, Шнорра, вычисления в группе точек эллиптической кривой над конечным полем	Международный стандарт
3	RSA	RSA	Стандарт ISO 979
4	ГОСТ 94	Эль-Гамалья, Шнорра	ГОСТ Р 34.10 – 94
5	ГОСТ 2001	Эль-Гамалья, Шнорра, вычисления в группе точек эллиптической кривой над конечным полем	ГОСТ Р 34.10 – 2001
6	ESIGN	Оригинальная схема, комбинация принципов RSA и DSA	NTT Japan
7	СТБ	Эль-Гамалья, Шнорра	Стандарт СТБ 1176.2- 99 (Республика Беларусь)

данный момент не рассматривается, важна лишь его аутентичность. Если существенна и конфиденциальность сообщения, то его можно зашифровать (до или после подписания), используя, например, открытый ключ адресата.

Получатель подписи S применяет открытое преобразование проверки V к S и получает на выходе процедуры сообщение M и некоторый бит v , который отвечает за результат проверки подписи $M, v = V(S)$. Если результат проверки положителен, то адресат получает уверенность в следующем:

- в целостности сообщения, т. е. в том, что оно не было изменено при передаче;
- в его аутентичности, т. е. в том, что оно было послано именно отправителем;
- в отсутствии отказа от авторства (ренегатства): отправитель не сможет утверждать, что не посылал сообщения.

Последнее свойство важно для электронной коммерции (невозможно отрицать факт подписания чека и пр.).

Схема с восстановлением сообщения на основе RSA.

Алгоритм шифрования RSA можно непосредственно использовать в качестве алгоритма подписи с восстановлением сообщения:

1) Отправитель применяет расшифровывающее преобразование RSA, чтобы поставить подпись, беря сообщение и возводя его в закрытую степень d :

$$S = M^d \pmod{N}.$$

2) Получатель использует зашифровывающее преобразование RSA на основе открытой экспоненты E и восстанавливает оригинальное сообщение:

$$M = S^E \pmod{N}.$$

При этом встает вопрос о проверке правильности восстановленного сообщения. Если исходное сообщение было написано на естественном языке, то можно проверить, что восстановленное сообщение написано на том же языке. Это не очень удачное решение, так как RSA работает с числовым эквивалентом сообщения. Лучшим выходом служит добавление к сообщению некоторой избыточной информации.

Предположим, сообщение D состоит из t битов, а модуль N алгоритма RSA насчитывает k битов, причем $t < k - 32$. Тогда прибавляем к $D(k - t)/8$ байтов слева и получаем строку байтов вида

$$M = 00\|01\|FF\|FF...\|FF\|00\|D,$$

после чего подпись вычисляется посредством формулы $M^d \pmod{N}$. При проверке подписи правильность восстановления сообщения M подтверждается корректностью дополнения.

Схема подписи с восстановлением сообщения хороша для небольших сообщений (например при обмене ключами Диффи – Хеллмана) [1]. Если сообщение большое, его приходится разбивать на блоки и подписывать каждый блок отдельно, что порождает целый ряд дополнительных неудобств. В таком случае применяют схему подписи с дополнением.

Схема с дополнением. Абонент A , посылая сообщение M , сначала вычисляет $H(M)$, где H – криптографическая однонаправленная хеш-функция. Функция H является общедоступной.

Подписывающее преобразование s применяется к $H(M)$:

$$S = s(H(M)),$$

и результат S передается адресату вместе с сообщением M .

Получатель подписи (M, S) применяет функцию H к сообщению M и открытое преобразование проверки V к S и получает на выходе процедуры значение $v = (H(M), V(S))$,

где результирующий бит проверки подписи v получается при сравнении значения $H'(M)$, полученного из дополнения-подписи S и значения $H(M)$, полученного при применении функции H к тексту сообщения M .

Схема с дополнением на основе RSA. Предположим, дано длинное сообщение M для визирования. Сначала вычисляется $H(M)$, и потом применяется преобразование подписи RSA к хеш-свертке $H(M)$, т. е. подпись получается так:

$$S = (H(M))^d \pmod{N}.$$

Наконец, подпись и само сообщение передаются вместе в виде пары (M, S) .

Проверка пары (M, S) состоит из трех этапов:

1) «Зашифрование» S с помощью открытой экспоненты RSA для получения H' : $H'(M) = S^E \pmod{N}$.

2) Вычисление $H(M)$ по M .

3) Проверка равенства $H'(M) = H(M)$. Если оно верно, то подпись законна. В противном случае – незаконна.

Алгоритм цифровой подписи Фиата – Шамира. Алгоритм основан на сложности задач факторизации больших целых чисел и извлечения квадратного корня в кольце вычетов. В данном алгоритме реализуется цифровая подпись с дополнением.

Пусть H – некоторая хеш-функция, преобразующая исходное сообщение в битовую строку длиной m [13]. Выбирают два простых числа p и q и вычисляют $N = p \cdot q$. В качестве секретного ключа каждый абонент должен сгенерировать m различных случайных чисел $a_1, a_2, \dots, a_m \in \mathbb{Z}_N$. Открытым ключом объявляется набор чисел $B_1, B_2, \dots, B_m \in \mathbb{Z}_N$, где $B_i = a_i^2 \pmod{N}$.

Алгоритм вычисления цифровой подписи для сообщения M состоит в выполнении ряда действий:

1. Выбрать случайное число r , $1 < r < N - 1$.

2. Вычислить $u = r^2 \pmod{N}$.

3. Вычислить $H(M, u) = S = (S_1, S_2, \dots, S_m)$.

4. Вычислить $T = \prod_{i=1}^m S_i a_i \pmod{N}$.

5. Подписью для M положить пару (S, T) .

Алгоритм проверки подписи заключается в следующем:

1) По открытому ключу $B_1, B_2, \dots, B_m \bmod N$ и значению T вычислить

$$W = \sum_{i=1}^m B_i T^i$$

2) Вычислить $H(M, W) = S'$.

3) Проверить равенство $S = S'$.

Достоинствами описанной схемы являются возможность выработки цифровых подписей для нескольких различных сообщений с использованием одного секретного ключа, а также сравнительная простота алгоритмов вычисления и проверки подписи. Попытка компрометации данной схемы сталкивается с необходимостью решения сложной задачи нахождения квадратных корней по модулю N . Недостатком схемы является большая длина ключа, которая определяется числом m . Если двоичная запись числа N содержит l знаков, то длина закрытого ключа составляет ml бит, а открытого ключа – $(m + 1)l$ бит. При этом необходимо учитывать, что для обеспечения достаточной стойкости данной схемы цифровой подписи числа m и l должны иметь в своей двоичной записи несколько сотен бит.

Пример. Если $p = 5$, $q = 7$, $N = 35$, то возможными квадратичными вычетами являются:

1: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 1, 6, 29, 34$.

4: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 2, 12, 23, 33$.

9: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 3, 17, 18, 32$.

11: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 9, 16, 19, 26$.

14: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 7, 28$.

15: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 15, 20$.

16: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 4, 11, 24, 31$.

21: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 14, 21$.

25: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 5, 30$.

29: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 8, 13, 22, 27$.

30: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 10, 25$.

Обратными значениями (mod 35) и их квадратными корнями являются: $B_i = 1, 4, 9, 11, 16, 29$; $(B_i)^{-1} = a_i^2 = 1, 9, 4, 16, 11, 29$; $a_i = 1, 3, 2, 4, 9, 8$

У чисел 14, 15, 21, 25 и 30 нет обратных значений по модулю 35, так как они не взаимно простые с 35. Это верно, так как должно быть $(p-1)(q-1)/4 = (5-1)(7-1)/4 = 6$ квадратичных вычетов по модулю 35, взаимно простых с 35. Поэтому НОД($x, 35$) должен быть равен единице.

Для хеш-функции H со сверткой длины $m = 4$ выберем открытый ключ $\{B_i\} = \{4, 11, 16, 29\}$;

и соответственно закрытый ключ $\{a_i\} = \{3, 4, 9, 8\}$.

Выбираем $r = 16$, вычисляем $u = r^2 = 16^2 \bmod 35 = 11$.

Допустим, значение хеш-свертки сообщения M и u составило: $S = H(M, u) = (1, 1, 0, 1)$.

Вычисляем:

$$T = 16 \cdot (3^1 \cdot 4^1 \cdot 9^0 \cdot 8^1) \bmod 35 = 31.$$

При проверке подписи находят:

$$W = 31^2 \cdot (4^1 \cdot 11^1 \cdot 16^0 \cdot 29^1) \bmod 35 = 11 = u.$$

Но, так как проверяющий не знает u , он для проверки вычисляет хеш-свертку: $S' = H(M, W) = (1, 1, 0, 1) = S$.

Алгоритм цифровой подписи Эль-Гамала. Схема цифровой подписи Эль-Гамала основана на сложности вычисления дискретных логарифмов в конечном поле.

Так же как и для шифрования Эль-Гамала, выбираются параметры системы:

- P – большое простое число (порядка 1024-2048 бит);
- Q – простой делитель числа $P - 1$ (порядка 160 бит);
- G – порождающий элемент мультипликативной группы

$$\langle G \rangle = F_P^*.$$

Секретным ключом является x – целое случайное число в диапазоне $2 \leq x \leq P - 2$.

Открытый ключ Y рассчитывается по формуле:

$$Y = G^x \bmod P.$$

Подпись для сообщения M вычисляется при помощи следующего алгоритма:

1. Выбрать случайный эфемерный сеансовый ключ k , $2 \leq k \leq P - 2$, взаимно простой с $P - 1$, $\text{НОД}(k, P - 1) = 1$.
2. Вычислить $R = G^k \pmod{P}$.
3. Вычислить $S = (M - xR)k^{-1} \pmod{P - 1}$.
4. Подписью сообщения M служит пара (R, S) .

Если сообщение M большое, то при вычислении S используют его хеш-значение $H(M)$.

Алгоритм проверки подписи заключается в проверке сравнения:

$$Y^R R^S \equiv G^M \pmod{P}.$$

Основным достоинством данной схемы является возможность выработки цифровых подписей для большого числа сообщений с использованием одного секретного ключа.

Пример. Выберем $P = 11$ и $G = 2$, а закрытый ключ $x = 8$.

Вычислим:

$$Y = G^x \pmod{P} = 2^8 \pmod{11} = 3.$$

Открытым ключом являются $Y = 3$, $G = 2$, и $P = 11$. Чтобы подписать $M = 5$, сначала выберем случайное $k = 9$. Убеждаемся, что $\text{НОД}(9, 10) = 1$. Вычисляем:

$$R = G^k \pmod{P} = 2^9 \pmod{11} = 6.$$

Далее с помощью расширенного алгоритма Евклида находим $k^{-1} = 9 \pmod{11 - 1}$ и S :

$$S = (M - xR)k^{-1} \pmod{P - 1} = (5 - 8 \cdot 6) \cdot 9 \pmod{11 - 1} = 3.$$

Подпись составит пару $(6, 3)$.

Для проверки подписи убедимся, что:

~~$$Y^R R^S \equiv G^M \pmod{P};$$~~

$$3^6 \cdot 6^3 \pmod{11} = 2^5 \pmod{11}.$$

Схема Эль-Гамала послужила образцом для построения большого семейства во многом сходных по своим свойствам схем подписи. В их основе лежит проверка сравнения вида

~~$$G^A R^B \equiv G^C \pmod{P};$$~~

в котором тройка (A, B, C) совпадает с одной из перестановок чисел $\pm M, \pm S, \pm R$.

Например, исходная схема Эль-Гамала получается при

$$A = M, B = -R \text{ и } C = S.$$

Еще одно достоинство данного семейства – это возможность уменьшения длины подписи путем замены пары чисел (R, S) на

пару чисел $(R \bmod Q, S \bmod Q)$. При этом проверочное равенство по модулю P следует заменить на модифицированное равенство по модулю Q :



На основе семейства данных схем построены стандарты цифровой подписи DSS и ГОСТ Р34.10-94.

Кроме того, данное семейство может быть модифицировано для работы с группой точек эллиптической кривой.

Алгоритм цифровой подписи Шнорра. Принадлежит семейству цифровых подписей на дискретных логарифмах и может быть обобщен на эллиптические кривые.

Параметры системы совпадают с параметрами P , Q , G системы Эль-Гамала.

Закрытый ключ x – целое число из интервала $1 < x < Q - 1$. Открытый ключ определяется формулой $Y = G^x \pmod{P}$. Чтобы подписать сообщение в алгоритме Шнорра, поступают следующим образом:

- 1) Выбирают эфемерный ключ k из промежутка $1 < k < Q - 1$.
- 2) Вычисляют соответствующий открытый ключ $R = G^k \bmod P$.
- 3) Находят $E = H(M||R)$. Значение функции зависит как от сообщения, так и от эфемерного ключа.
- 4) Вычисляют $S = k + x \cdot E \pmod{Q}$.

Полученная таким образом пара (E, S) является искомой подписью.

Для проверки подписи вычисляют:

$$R' = G^S Y^E \pmod{P}; H(M||R').$$

Подпись корректна, если верно равенство $E = H(M||R')$. (Возможен вариант, в котором закрытый ключ вычисляется по формуле $Y = G^x \pmod{P}$, и тогда проверочное уравнение принимает форму $R' \equiv G^S Y^E \pmod{P}$.)

Пример. Параметры домена $Q = 101$, $P = 607$ и $G = 601$.

Чтобы зафиксировать ключевую пару, положим $x = 3$ и $Y = G^x \pmod{P} = 601^3 \pmod{607} = 391$.

Затем генерируем эфемерный ключ $k = 65$ и вычисляем:

$$R = G^k \pmod{P} = 601^{65} \pmod{607} = 223.$$

Теперь находим хеш-значение $E = H(M||R)$. Допустим, что при этом получилось $E = 93$. Тогда второй компонент подписи выглядит как $S = k + x \cdot E \pmod{Q} = 65 + 3 \cdot 93 \pmod{101} = 41$.

При проверке подписи (E, S) теоретически можно вычислить $R' \cdot Y^E \pmod{P} = 223 \cdot 391^{93} \pmod{607} = 172$;

$$G^S \pmod{P} = 601^{41} \pmod{607} = 172.$$

Но так как R' не передается напрямую, а только в виде свертки $E = H(M||R)$, придется вычислять:

$$R' = G^S Y^{-E} \pmod{P} = 601^{41} \cdot (391^{93})^{-1} \pmod{607} = 172 \cdot (537)^{-1} \pmod{607} = 223.$$

Соответственно совпадут и хеш-значения $H(M||R) = H(M||R') = 93$.

Алгоритм цифровой подписи Ниберга – Руппеля. Данный алгоритм также базируется на сложности дискретного логарифмирования и является примером схемы с восстановлением сообщения.

Все схемы подписи с восстановлением сообщения используют не однонаправленную хеш-функцию H , а открытую функцию избыточности F , которая является легко обратимой. В качестве простого примера можно взять $F(M) = (M||M)_2$.

Параметры системы совпадают с параметрами P, Q, G системы Эль-Гамала.

Закрытый ключ x – целое число из интервала $1 < x < Q - 1$. Открытый ключ определяется формулой $Y = G^x \pmod{P}$.

Алгоритм подписи Ниберга – Руппеля состоит в следующем:

1. Берут случайное число k из промежутка $1 < k < Q - 1$ и вычисляют $R = G^k \pmod{P}$.

2. Находят $E = F(M) \cdot R \pmod{P}$.

3. Определяют $S = x \cdot E + k \pmod{Q}$.

Подпись представляет собой пару (E, S) . Исходя из этой пары, нужно:

- убедиться в том, что подпись принадлежит пользователю с открытым ключом Y ;

- восстановить сообщение M .

В процедуре проверки подписи Ниберга – Руппеля по паре (E, S) и открытому ключу отправителя Y вычисляют:

$$U_1 = G^S Y^{-E} \pmod{P} = R.$$

$$U_2 = E(U_1)^{-1} \pmod{P}.$$

После этого убеждаются, что U_2 является значением функции избыточности $U_2 = F(M)$. Если это равенство ложно, то подпись отклоняют. В противном случае восстанавливают сообщение по правилу $M = f^{-1}(U_2)$ и принимают подпись.

Пример. Параметры домена $Q = 101$, $P = 607$ и $G = 601$.

Чтобы зафиксировать ключевую пару, положим $x = 3$ и

$$Y = G^x \pmod{P} = 601^3 \pmod{607} = 391.$$

Чтобы подписать сообщение $M = 12$ (в нашем примере M должно лежать на отрезке $[0, 15]$), выбираем эфемерный закрытый ключ $k = 45$ и вычисляем:

$$R = G^k \pmod{P} = 601^{45} \pmod{607} = 143.$$

Допустим, $F(M) = M + 2^4$.

$$\text{Тогда } F(M) = 204 \text{ и } E = F(M) \cdot R \pmod{P} = 204 \cdot 143 \pmod{607} = 36,$$

$$S = x \cdot E + k \pmod{Q} = 3 \cdot 36 + 45 \pmod{101} = 52.$$

Подписью является пара $(E, S) = (36, 52)$.

Проверка подписи и восстановление сообщения: вычисляется

$$U_1 = G^S Y^E \pmod{P} = 601^{52} \cdot (391^{36})^{-1} \pmod{607} = 195 \cdot 284 \pmod{607} = 143 = R.$$

Далее находим:

$$U_2 = E(U_1)^{-1} \pmod{P} = 36 \cdot (143)^{-1} \pmod{607} = 36 \cdot 208 \pmod{607} = 204.$$

Теперь необходимо убедиться, что найденное число U_2 представимо в виде $M + 2^4 \cdot M$ для некоторого целого числа $M \in [0, 15]$. U_2 действительно так представимо, поэтому подпись корректна.. Сообщение $M = 12$ восстанавливается из уравнения

$$M + 2^4 \cdot M = 204.$$

Алгоритм цифровой подписи DSA. Рассмотрим данный алгоритм более подробно, так как на его основе построен стандарт цифровой подписи DSS. DSA – алгоритм подписи с дополнением, в котором собственно подпись состоит из двух 160-битовых целых чисел R и S . Число R является функцией 160-разрядного случайного числа k , которое называют эфемерным ключом, изменяемым с каждым новым сообщением. Число S – функция от сообщения секретного ключа x , принадлежащего подписывающему лицу, числа R и эфемерного ключа k .

Аналогично алгоритму Эль-Гамала здесь есть несколько параметров домена. Для начала фиксируется 160-битовое простое число Q , а затем выбирается такое большое простое число P , лежащее между 2^{512} и 2^{2048} , что $P - 1$ делится на Q . Наконец генерируется случайное число $H < P$, и вычисляется $G = H^{\frac{P-1}{Q}}$.

Если $G = 1$, то переходят к другому случайному H до тех пор, пока не получат $G \neq 1$. Этим обеспечивается выполнение следующего условия: G – порождающий элемент группы F_P^* порядка Q , т. е. $G^Q = 1 \pmod{P}$.

После выбора параметров домена (P, Q, G) каждый пользователь генерирует свой собственный закрытый подписывающий ключ x , удовлетворяющий неравенству $0 < x < Q$. Соответствующим открытым ключом служит число Y , вычисляемое по правилу $Y = G^x \pmod{P}$.

Процедура выбора пользовательской ключевой пары существенно проще, чем в RSA, поскольку она требует всего лишь одного возведения в степень в числовом поле.

Подписание сообщения осуществляется по следующему алгоритму:

- 1) Вычисляют значение хеш-функции $H = H(M)$.
- 2) Выбирают случайный эфемерный ключ k , $0 < k < Q$.
- 3) Определяют $R = (G^k \pmod{P}) \pmod{Q}$.
- 4) Находят $S = \frac{H + xR}{k} \pmod{Q}$

Подписью сообщения M служит пара (R, S) , имеющая в общей сложности 320 двоичных знаков.

Для проверки подписи (R, S) на сообщении M выполняют следующие действия:

- вычисляют значение хеш-функции $H = H(M)$,
 - определяют $A = H/S \pmod{Q}$ и $B = R/S \pmod{Q}$,
 - находят $V = (G^A Y^B \pmod{P}) \pmod{Q}$,
- где Y – открытый ключ автора сообщения.
- Подпись считается корректной только тогда, когда $V = R$.

Пример. Параметры домена $Q = 13$, $P = 4Q + 1 = 53$ и $G = 16$. Предположим, что ключевая пара имеет вид $x = 3$ и $Y = G^x \pmod{P} = 16^3 \pmod{53} = 15$. Если хотим подписать сообщение с хеш-значением $H = 5$, то сначала нужно выбрать эфемерный ключ $k = 2$

и найти $R = (G^k \pmod{P}) \pmod{Q} = (16^2 \pmod{53}) \pmod{13} = 44 \pmod{13} = 5$, $S = (H + xR)/k \pmod{Q} = (5 + 3 \cdot 5) \cdot 2^{-1} \pmod{13} = (5 + 3 \cdot 5) \cdot 7 \pmod{13} = 10$.

Для проверки подписи получатель определяет:

$$A = H/S \pmod{Q} = 5 \cdot 10^{-1} \pmod{13} = 5 \cdot 4 = 7;$$

$$B = R/S \pmod{Q} = 5 \cdot 10^{-1} \pmod{13} = 7;$$

$$V = (G^A Y^B \pmod{P}) \pmod{Q} = (16^7 \cdot 15^7 \pmod{53}) \pmod{13} = (49 \cdot 42 \pmod{53}) \pmod{13} = 5.$$

Ввиду равенства $V = R = 5$ делаем вывод о корректности подписи.

Симметричные (одноразовые) цифровые подписи. Рассмотренные системы цифровой подписи имеют один потенциальный недостаток. Он состоит в возможности построения новых эффективных алгоритмов для решения этих математических задач. Поэтому в реальных схемах длину ключа выбирают с определенным превышением необходимой величины для обеспечения достаточного запаса стойкости. Это, в свою очередь, значительно усложняет алгоритмы вычисления и проверки подписи. Поэтому представляется весьма привлекательной задача построения схем цифровой подписи на основе симметричных систем шифрования, свободных от подобных недостатков.

Например, схема цифровой подписи Диффи – Лампорта на основе симметричных систем шифрования [1].

Пусть требуется подписать сообщение $M = m_1 m_2 \dots m_n$, $m_i \in \{0,1\}$, $i = 1, \dots, n$. Согласно схеме Диффи – Лампорта, подписывающий сначала выбирает $2n$ случайных секретных ключей $K = [(k_{10}, k_{11}), \dots, (k_{n0}, k_{n1})]$ для используемой им симметричной шифрсистемы, затем n пар случайных чисел $S = [(S_{10}, S_{11}), \dots, (S_{n0}, S_{n1})]$, где $S_{ij} \in \{0,1\}$, $i = 1, \dots, n$, $j = 0,1$ и вычисляет значения ~~$R = [R_{10}, R_{11}), \dots, (R_{n0}, R_{n1})]$~~ .

Наборы S и $R = [(k_{10}, k_{11}), \dots, (k_{n0}, k_{n1})]$ являются открытыми и помещаются в общедоступном месте так, чтобы каждый мог прочитать их, но записать их туда мог бы только автор подписи.

Подпись для сообщения M имеет вид $(k_{1m_1}, \dots, k_{nm_n})$. Чтобы убедиться в ее правильности, следует проверить равенства ~~$R_{ij} = S_{ij} \oplus k_{im_i}$~~ .

Недостатком этой схемы является слишком большой размер подписи, который может превышать размер самого подписываемого сообщения. Имеется несколько способов избавиться от этого недостатка:

- Во-первых, можно хранить не $2n$ значений секретных ключей, а лишь один секретный ключ k . Для этого можно воспользоваться, например, одной из следующих схем формирования последовательности K :

$$k_{ij} = E_k(i, j), \quad j = 0, 1, \quad i = 1, \dots, n;$$
$$k_{ij} = \begin{cases} E_k(i-1, 1), & j = 0, \\ E_k(i, 0), & j = 1, \end{cases} \quad k_{10} = E_k(0), \quad i = 1, \dots, n.$$

- Во-вторых, можно аналогичным образом свернуть набор открытых значений S .

- В-третьих, можно подписывать не само сообщение, а его свертку, если воспользоваться какой-либо хеш-функцией.

- Можно использовать и другие подходы, позволяющие сократить как длину подписи, так и размеры открытого и секретного ключей.

Вместе с тем подобные модификации не устраняют главного недостатка рассматриваемых подписей, состоящего в том, что после проверки подписи либо весь секретный ключ, либо его часть становятся известными проверяющему. Поэтому данная схема цифровой подписи является, по существу, одноразовой.

Контрольные вопросы

1. Различия между шаговой и блочно-итерационной функциями.
2. Основные требования к бесключевым хеш-функциям.
3. Каковы основные требования к ключевым хеш-функциям?
4. Отличия между собственноручной и электронной подписями.
5. Какие бывают схемы реализации подписей на основе RSA?
6. Суть модификации Шнорра для схемы подписи Эль-Гамала.
7. Почему симметричные электронные подписи называют одноразовыми?

6. ОРГАНИЗАЦИЯ СЕТЕЙ ЗАСЕКРЕЧЕННОЙ СВЯЗИ

6.1. Протоколы распределения ключей

Выделяют следующие типы протоколов распределения ключей:

- протоколы передачи (уже сгенерированных) ключей;
- протоколы (совместной) выработки общего ключа (открытое распределение ключей);
- схемы предварительного распределения ключей.

Различают также протоколы распределения ключей между отдельными участниками и между группами участников информационного взаимодействия.

Передача ключей с использованием симметричного шифрования. Имеются протоколы, в которых стороны осуществляют передачу ключей при непосредственном взаимодействии, т. е. двусторонние протоколы, или протоколы типа «точка – точка» и протоколы с централизованным распределением ключей, в которых предусмотрена третья сторона, играющая роль доверенного центра.

Двусторонние протоколы. Различают протоколы, в которых стороны заранее располагают какой-либо известной им обоим секретной информацией, и протоколы, не требующие этого условия.

Пусть стороны A и B заранее обладают общей секретной информацией. Допустим, что это – секретный ключ k_{AB} .

Тогда для передачи ключа k стороны могут использовать одностороннюю передачу:

$$A \rightarrow B: E_{k_{AB}}(k, t, B),$$

где E – алгоритм шифрования;

t – метка времени;

B – идентификатор абонента B (для краткости вместо $id(B)$ будем использовать лишь один символ B).

Если не передавать метки времени, то злоумышленник может осуществить повторную передачу того же сообщения. Если же не указывать идентификатора адресата, то злоумышленник может

вернуть отправителю перехваченное сообщение, что в некоторых ситуациях может быть опасным, поскольку абонент A не сможет установить, что это сообщение получено не от абонента B .

В приведенном протоколе вместо шифрования можно было использовать ключевую хеш-функцию, зависящую от общего ключа:

$$A \rightarrow B: k \oplus h_{kAB}(t, B).$$

Если дополнительно требуется аутентификация сеанса, то можно использовать следующий протокол типа «запрос – ответ»:

$$1. B \rightarrow A: r_B.$$

$$2. A \rightarrow B: E_{kAB}(k, r_B, B),$$

где r_B – случайное число, сгенерированное абонентом B и переданное абоненту A в начале сеанса.

При использовании хеш-функции подобный протокол может выглядеть так:

$$1. B \rightarrow A: r_B.$$

$$2. A \rightarrow B: k \oplus h_{kAB}(r_B, B).$$

Если требуется двусторонняя аутентификация, можно модифицировать последний протокол, предоставив возможность стороне A путем генерации своего случайного числа r_A и введения его в сообщение на шаге 2 протокола убедиться в том, что он имеет дело именно с абонентом B .

Исходный протокол можно модифицировать таким образом, чтобы искомый ключ k не генерировался одной стороной, а являлся результатом двустороннего обмена.

Пусть абонентами A и B , помимо случайных чисел r_A и r_B , генерируют также случайные числа k_A и k_B соответственно. Тогда в результате выполнения протокола

$$1. B \rightarrow A: r_B.$$

$$2. A \rightarrow B: E_{kAB}(k_A, r_A, r_B, B).$$

$$3. B \rightarrow A: E_{kAB}(k_B, r_B, r_A, A)$$

каждая из сторон может вычислить общий ключ с помощью некоторой функции f по правилу $k = f(k_A, k_B)$. В этом протоколе ни одна из сторон не может знать заранее значение ключа.

Приведем теперь «бесключевой» протокол А. Шамира, позволяющий передать ключ без использования какой-либо общей секретной информации [13]. Пусть имеется некоторое

коммутирующее шифрующее преобразование E . Это означает, что при всех сообщениях x и ключах k_1 и k_2 выполняется равенство

$$E_{k_1}(E_{k_2}(x)) = E_{k_2}(E_{k_1}(x))$$

Тогда пользователи A и B могут реализовать следующий трехшаговый протокол для передачи секретного ключа k от A к B :

1. $A \rightarrow B: E_{k_A}(k)$.
2. $B \rightarrow A: E_{k_B}(E_{k_A}(k))$.
3. $A \rightarrow B: D_{k_A}(E_{k_B}(E_{k_A}(k)))$.

В этом протоколе можно использовать не каждое коммутирующее преобразование E . Например, легко заметить, что для преобразования $E_k(k) = k \oplus \Gamma$ протокол оказывается заведомо нестойким. Поэтому в протоколе Шамира рекомендуется использовать преобразование вида $E_{k_A}(k) = k^a \bmod p$ [13], в котором константа a определяется ключом k_A .

Трехсторонние протоколы. Рассмотрим протоколы распределения ключей между парами участников с использованием третьей стороны T , называемой центром. В этом качестве обычно выступает некоторый выделенный узел сети или сервер, которому доверяют все участники. Центр T хранит ключи всех абонентов сети. Поэтому схема ключевых взаимоотношений графически представляет собой звезду.

Один из первых протоколов такого типа заключается в выполнении следующих шагов:

1. $A \rightarrow T: A, B, r_A$.
2. $T \rightarrow A: E_{k_{AT}}(r_A, B, k, E_{k_{BT}}(k, A))$.
3. $A \rightarrow B: E_{k_{BT}}(k, A)$.
4. $B \rightarrow A: E_k(r_B)$.
5. $A \rightarrow B: E_k(r_B - 1)$.

В результате выполнения трех первых шагов протокола пользователи A и B получают сгенерированный центром T общий ключ k для организации взаимодействия. Четвертый и пятый шаги предназначены для аутентификации пользователя A и подтверждения правильности получения ключа обеими сторонами.

Слабость этого протокола заключается в возможности повторной передачи абоненту B сообщения, переданного на шаге 3. При этом абонент B не имеет возможности установить, что полученный ключ k уже был использован. Поэтому в случае

компрометации этого ключа злоумышленник может аутентифицироваться и передавать сообщения от имени A .

Недостаток этого протокола устранен в протоколе *Kerberos*. Рассмотрим сначала базовый протокол, применяемый в протоколе аутентификации и распределения ключей *Kerberos*:

1. $A \rightarrow T: A, B, r_A$.
2. $T \rightarrow A: E_{k_{BT}}(k, r_A, L, B)$, билет.
3. $A \rightarrow B$: билет, аутентификатор.
4. $B \rightarrow A: E_k(t, k_B)$.

Здесь «билетом» названа величина $E_{k_{BT}}(k, A, L)$, «аутентификатором» – величина $E_k(A, t, k_A)$, t – метка времени; L – период времени действия билета; r_A – случайное число, сгенерированное абонентом A и вставленное в передаваемое сообщение для взаимной аутентификации; k_A и k_B – случайные числа, сгенерированные абонентами A и B соответственно и используемые либо в качестве ключа шифрования информации другой стороне, либо для выработки общего ключа $k_{AB} = f(k_A, k_B)$ с помощью некоторой функции f .

В полном протоколе *Kerberos* описанный выше базовый протокол используется два раза. Дело в том, что в нем предусмотрено два сервера. Первый – сервер аутентификации обозначаемый AS , выдает так называемые билеты для получения билетов (tgt), содержащие ключи, предназначенные для длительного использования. Второй сервер, TGS , – сервер выдачи билетов выдает обычные билеты для доступа к сетевым ресурсам и обращения к другим пользователям.

Сообщения, передаваемые согласно этому протоколу, выглядят следующим образом (рис. 36):

1. $A \rightarrow AS: A, TGS, r_A$.
2. $AS \rightarrow A: E_{k_{A,AS}}(k_{A,TGS}, r_A, L_1, TGS)$, tgt .
3. $A \rightarrow TGS: tgt$, аутентификатор₁, B, r_A' .
4. $TGS \rightarrow A: E_{k_{A,TGS}}(k, r_A', L_2, B)$, билет.
5. $A \rightarrow B$: билет, аутентификатор₂.
6. $B \rightarrow A: E_k(t_2, k_B)$,

где

$$tgt = E_{k_{AS,TGS}}(k_{A,TGS}, A, L_1),$$

$$\text{аутентификатор}_1 = E_{k_{A,TGS}}(A, t_1),$$

$$\text{билет} = E_{k_{B,TGS}}(K, A, L_2),$$

$$\text{аутентификатор}_2 = E_k(A, t_2, k_A).$$

Благодаря введению второго сервера нагрузка на первый сервер уменьшается во много раз. Первый сервер должен быть наиболее защищенным, поскольку он хранит главные ключи всех пользователей. Серверов второго типа может быть несколько, и они могут соответствовать определенной подсети или определенному типу ресурса.

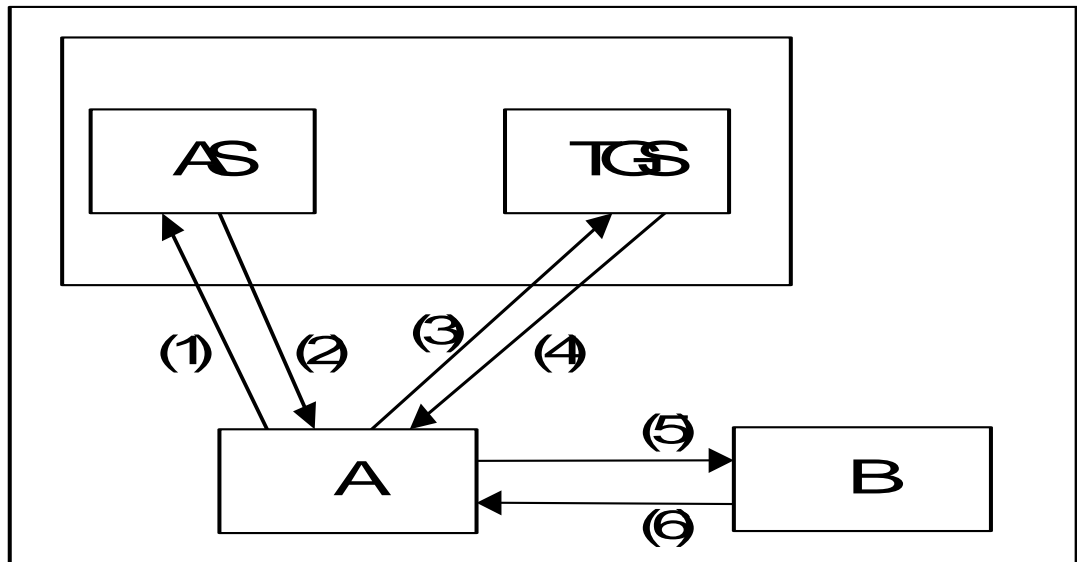


Рис.36. Протокол Kerberos.

Приведем еще один протокол распределения ключей с использованием сервера, предпочтительный для случая, когда сервер находится в более удобном расположении для второго абонента. Протокол состоит в выполнении следующих действий:

1. $A \rightarrow B: r, A, B, E_{k_{AT}}(r_A, r, A, B).$
2. $B \rightarrow T: r, A, B, E_{k_{AT}}(r_A, r, A, B), E_{k_{BT}}(r_B, r, A, B).$
3. $T \rightarrow B: E_{k_{AT}}(r_A, k), E_{k_{BT}}(r_B, k).$
4. $B \rightarrow A: E_{k_{AT}}(r_A, \kappa).$

Пользователь A генерирует два случайных числа: первое (r_A) используется, как и раньше, для взаимной аутентификации, а второе (r) – для аутентификации сеанса связи (вместо него может быть использована метка времени).

Этот протокол можно дополнить еще одним шагом для обеспечения взаимной аутентификации сторон и подтверждения правильности полученного ключа:

5. $B \rightarrow A: E_{k_{AT}}(r_A, k), E_k(r, r_B).$

6. $A \rightarrow B: E_k(r)$.

Передача ключей с использованием асимметричного шифрования. Рассмотрим варианты использования асимметричного шифрования для передачи секретных ключей симметричных криптосистем.

Протоколы без использования цифровой подписи. Для передачи ключа k можно использовать следующий одношаговый протокол:

$A \rightarrow B: E_{kB}(k, t, A)$,

где E – алгоритм шифрования с открытым ключом;

t – метка времени, вставляемая для предотвращения возможности повторного использования ключа.

Для осуществления взаимной аутентификации и подтверждения правильности получения ключа можно воспользоваться протоколом из [78]:

1. $A \rightarrow B: E_B(k_1, A)$.

2. $B \rightarrow A: E_A(k_1, k_2)$.

3. $A \rightarrow B: E_B(k_2)$.

Производя расшифрование полученных сообщений на втором и третьем шагах, стороны убеждаются в том, что они имеют дело с нужной стороной и что другая сторона правильно расшифровала полученное значение ключа.

Протоколы с использованием цифровой подписи. При использовании цифровой подписи аутентифицированный протокол передачи ключей может содержать только одно сообщение и иметь, например, один из следующих трех видов:

• $A \rightarrow B: E_B(k, t, S_A(B, k, t))$

(шифрование подписанного ключа);

• $A \rightarrow B: E_B(k, t, S_A(B, k, t))$

(зашифрование и подпись ключа);

• $A \rightarrow B: t, E_B(A, k), S_A(B, t, E_B(A, k))$

(подпись зашифрованного ключа).

Сертификаты открытых ключей. Как правило, при использовании открытых ключей хранят не сами ключи, а их сертификаты. *Сертификат* представляет собой набор данных $C_A = (A, k_A, t, S_{kTA}(A, k_A, t))$, состоящий из идентификатора абонента A ,

его открытого ключа k_A и, быть может, еще какой-либо дополнительной информации, например, времени t выдачи сертификата и срока его действия, заверенных цифровой подписью доверенного центра TA или заслуживающего доверия лица. Сертификат предназначен для исключения возможности подмены открытого ключа при его хранении или пересылке.

Получив такой сертификат и проверив цифровую подпись, можно убедиться в том, что открытый ключ действительно принадлежит данному абоненту.

Международный стандарт ССІТТ Х.509 определяет следующий протокол аутентификации с одновременным распределением ключей:

1. $A \rightarrow B: C_A, D_A, S_A(D_A).$
2. $B \rightarrow A: C_B, D_B, S_B(D_B).$
3. $A \rightarrow B: r_B, B, S_A(r_B, B),$

где C_A и C_B – сертификаты сторон;

S_A и S_B – цифровые подписи сторон;

$D_A = (t_A, r_A, B, data_1, E_B(k_1)); D_B = (t_B, r_B, A, r_A, data_2, E_A(k_2))$ – наборы передаваемых и подписываемых данных. В поле $data$ заносится дополнительная информация для аутентификации источника. Третий шаг протокола требуется стороне B для подтверждения того, что она действительно взаимодействует со стороной A .

Открытое распределение ключей. Оно позволяет двум абонентам выработать общий секретный ключ путем динамического взаимодействия на основе обмена открытыми сообщениями без какой-либо общей секретной информации, распределяемой заранее. Важным преимуществом открытого распределения является также то, что ни один из абонентов заранее не может определить значения ключа, так как ключ зависит от сообщений, передаваемых в процессе обмена.

Первый алгоритм открытого распределения ключей был предложен У. Диффи и М. Хеллманом. Для его выполнения стороны должны договориться о значениях большого простого числа $[1] p$ и образующего элемента α мультипликативной группы $Z_p = \{1, 2, \dots, p-1\}$. Для выработки общего ключа k они должны сгенерировать случайные числа $x, 1 \leq x \leq p-2$, и $y, 1 \leq y \leq p-1$

соответственно, затем обмениваться сообщениями в соответствии с протоколом:

1. $A \rightarrow B: \alpha^x \bmod p$,
2. $B \rightarrow A: \alpha^y \bmod p$.

Искомый общий ключ теперь вычисляется по формуле

$$k = (\alpha^y)^x = (\alpha^x)^y \bmod p.$$

Недостатком этого протокола является возможность атаки типа «злоумышленник в середине», состоящей в следующем. Предположим, что злоумышленник имеет возможность осуществлять подмену передаваемых абонентами сообщений.

Рассмотрим два протокола, устраняющих этот недостаток. Первый протокол, называемый STS (*station-to-station*), предполагает, что пользователи применяют цифровую подпись, которой подписываются передаваемые по протоколу Диффи–Хеллмана сообщения [1]:

1. $A \rightarrow B: \alpha^x \bmod p$.
2. $B \rightarrow A: \alpha^y \bmod p, E_k(S_B(\alpha^y, \alpha^x))$.
3. $A \rightarrow B: E_k(S_A(\alpha^x, \alpha^y))$.

Здесь S_A и S_B – цифровые подписи пользователей A и B соответственно, k – искомый общий ключ. Они позволяют гарантировать достоверность получения сообщения именно от того пользователя, от которого это сообщение получено. Шифрование значений подписей пользователей введено для того, чтобы обеспечить взаимное подтверждение правильности вычисления значения ключа.

Еще один подход также предполагает наличие у абонентов открытых ключей, но вместо цифровой подписи предлагается использовать модифицированную процедуру выработки общего ключа. Рассмотрим протокол MTI (Мацумото–Такашима–Имаи).

Предположим, что пользователи A и B имеют секретные ключи a , $1 \leq a \leq p - 2$, и b , $1 \leq b \leq p - 2$, соответственно и публикуют свои открытые ключи $z_A = \alpha^a \bmod p$ и $z_B = \alpha^b \bmod p$. Для выработки общего секретного ключа k они должны сгенерировать случайные числа x , $1 \leq x \leq p - 2$, и y , $1 \leq y \leq p - 2$ соответственно, а затем обмениваться следующими сообщениями:

1. $A \rightarrow B: \alpha^x \bmod p$.
2. $B \rightarrow A: \alpha^y \bmod p$.

Искомый общий ключ вычисляется по формуле



Теперь любая подмена сообщений приведет к тому, что все стороны получают различные значения ключа, что, в свою очередь, приведет к невозможности чтения всей передаваемой информации.

Предварительное распределение ключей. Большинство криптографических систем требуют проведения предварительного распределения секретных ключей. Для предварительного распределения стороны могут обмениваться ключами при личной встрече, либо поручить доставку ключей специально назначенному доверенному курьеру либо использовать для передачи некоторый выделенный защищенный канал. В зависимости от назначения криптографической системы иногда оказывается удобным распределять не сами ключи, а некоторые вспомогательные ключевые материалы, на основании которых каждый участник или группа участников могут самостоятельно вычислить необходимый ключ, используя для этого некоторую установленную заранее процедуру.

Схемы предварительного распределения ключей в сети связи. Если число абонентов сети засекреченной связи невелико, то и число распределяемых ключей также невелико. Для больших же сетей распределение ключей становится очень серьезной проблемой. Она заключается в том, что для сети, в которой работают n абонентов, необходимо выработать заранее и хранить в дальнейшем $n(n-1)/2$ ключей. Кроме того, каждому абоненту сети необходимо передать ключи для связи с остальными $n - 1$ абонентами, которые абонент должен постоянно хранить. Например, для сети со 100 абонентами нужно сгенерировать и хранить почти 5000 ключей, причем каждый абонент при этом должен хранить у себя 99 ключей.



Для уменьшения объема хранимой ключевой информации применяются различные *схемы предварительного распределения ключей* в сети связи. Их суть заключается в том, что в действительности вначале происходит распределение не самих ключей, а некоторых вспомогательных ключевых материалов, занимающих меньшие объемы. На основании этих материалов каждый абонент сети может самостоятельно вычислить по некоторому алгоритму необходимый для связи ключ. Такой подход

позволяет уменьшить объемы как хранимой, так и распределяемой секретной информации.

В качестве примера рассмотрим *схему Блома* распределения ключей между n абонентами, для которой процедура вычисления ключа заключается в вычислении значения некоторого симметрического многочлена над конечным полем.

Выберем поле F , имеющее конечное, но достаточно большое число элементов, и зафиксируем n различных элементов $r_1, \dots, r_n \in F$, отличных от нуля. Каждый элемент r_i припишем i -му абоненту сети, $i = \overline{1, n}$. Эти элементы не являются секретными и могут храниться на общедоступном сервере сети. Выберем теперь многочлен над полем F степени $2m$, $1 \leq m < n$, вида $f(x, y) = \sum_{i=0}^m \sum_{j=0}^m a_{ij} x^i y^j$,

где $a_{ij} = a_{ji}$, $i \neq j$, $i, j = \overline{0, m}$.

Его коэффициенты являются секретными и должны храниться только в центре распределения ключей. Каждый абонент A получает в качестве ключевых материалов набор , состоящий из коэффициентов многочлена .

Для связи между абонентами A и B теперь можно использовать общий ключ k_{AB} :

$$k_{AB} = k_{BA} = f(r_A, r_B) = g_B(r_A) = g_A(r_B),$$

вычисляемый по формуле

$$k_{AB} = \sum_{i=0}^m \sum_{j=0}^m a_{ij} r_A^i r_B^j$$

в матричном виде:

$$k_{AB} = (r_A \ r_A^2 \ \dots \ r_A^m) \cdot \begin{pmatrix} a_{00} & a_{01} & \dots & a_{0m} \\ a_{10} & a_{11} & \dots & a_{1m} \\ \dots & \dots & \dots & \dots \\ a_{m0} & a_{m1} & \dots & a_{mm} \end{pmatrix} \cdot \begin{pmatrix} 1 \\ r_B \\ r_B^2 \\ \dots \\ r_B^m \end{pmatrix}$$

где матрица $A = (a_{ij})_{m \times m}$ составлена из коэффициентов многочлена $f(x, y)$ и является симметричной.

При использовании данной схемы каждый абонент должен хранить $m+1$ секретных значений вместо $n-1$, общее же число секретных коэффициентов многочлена f равно $m(m+1)/2$.

Для заданного числа m схема Блома дает минимальное по объему количество хранимых у абонента ключевых материалов.

Схема предварительного распределения ключей KDP (key distribution patterns) основана на схеме пересечений множеств. Пусть имеется n , $n > 2$ абонентов (пользователей) и множество секретных ключей K , $|K| = q$. Будем считать, что все ключи пронумерованы числами $1, 2, \dots, q$. Выберем некоторое семейство $\{S_1, \dots, S_n\}$ подмножеств множества $\{1, 2, \dots, q\}$. Предварительно абоненту i по защищенному каналу передается множество секретных ключей с номерами из подмножества S_i , $i = \overline{1, n}$. Таким образом, семейство $\{S_1, \dots, S_n\}$ представляет собой таблицу с номерами ключей каждого пользователя. Хотя данная таблица является несекретной, она должна быть защищена от модификаций и подделок.

Если абонент i хочет связаться с абонентом j , то он использует для выработки общего ключа множество ключей, номера которых содержатся в пересечении $S_i \cap S_j$. Если каждый ключ представлен некоторой битовой строкой, то для формирования общего связного ключа можно взять, например, их сумму или значение некоторой хеш-функции от строки, составленной из ключей, номера которых входят в пересечение множеств $S_i \cap S_j$.

Схемой распределения ключей типа KDP, или $KDP(n, q)$ -схемой, назовем всякое семейство $\{S_1, \dots, S_n\}$ подмножеств множества K , удовлетворяющее следующему условию:

если при некоторых $i, j, r \in \{1 \leq i < j \leq n\}$ выполнено включение $S_i \cap S_j \subseteq S_r$, то либо $i = r$, либо $j = r$.

Это условие означает, что общий ключ двух абонентов не должен быть известным никакому другому абоненту.

Семейство подмножеств называется *семейством Шпернера*, если ни одно из них не содержится в другом.

Семейство $\{S_1, \dots, S_n\}$ подмножеств множества K , $|K| = q$, образует $KDP(n, q)$ -схему в том и только в том случае, если множество $\{S_i \cap S_j \mid 1 \leq i < j \leq n\}$ образует семейство Шпернера.

Если подмножества $\{S_1, \dots, S_m\}$ множества K , $|K| = q$ образуют семейство Шпернера, то

$$m \leq C_q^{\lfloor \frac{q}{2} \rfloor}.$$

Равенство достигается только в случае, если множество $\{S_1, \dots, S_m\}$ совпадает с множеством всех w -элементных подмножеств множества K , где $w = q/2$ при четном q и $w = (q+1)/2$ или $(q-1)/2$ при нечетном q .

Для любой $KDP(n, q)$ -схемы каждый абонент должен иметь не менее $\log_2 n$ ключей. Если $n \geq 4$, то $q \geq 2 \log_2 n$.

Схемы разделения секрета. Подобная схема представляет собой схему предварительного распределения ключей между уполномоченными группами пользователей, в которой ключ заранее определен и одинаков для каждой уполномоченной группы. При этом каждый пользователь получает свою *долю*, или «часть секрета». Схема включает два протокола: протокол формирования долей (разделения секрета) и распределения их между пользователями и протокол восстановления секрета группой пользователей. Схема должна позволять восстанавливать ключ только тем группам пользователей, которые имеют на это полномочия, и никакая другая группа не должна иметь возможности для восстановления ключа или получения о нем какой-либо информации.

Основное назначение схемы разделения секрета – защита ключа от потери. Обычно для защиты от потери делают несколько копий ключа. С возрастанием числа копий ключа возрастает вероятность его компрометации. Если число копий мало, то велик риск потери ключа. Поэтому лучше «разделить» ключ между несколькими лицами так, чтобы допускалась возможность восстановления ключа при различных обстоятельствах несколькими уполномоченными группами с заранее оговоренным составом участников. Тем самым исключается риск безвозвратной потери ключа.

Еще одно положительное качество схем разделения секрета заключается в разделении ответственности за принятие решения, которое автоматически вводится при определении состава уполномоченных групп. Такая коллективная ответственность нужна для многих приложений, включая принятие важных решений, касающихся применения систем оружия, подписания корпоративных чеков или допуска к банковскому хранилищу.

В простейшем случае, когда имеется только одна группа, состоящая из t пользователей, уполномоченная формировать ключ, схему разделения секрета можно построить следующим образом. Предположим, что ключ представляет собой двоичный вектор s длиной m . Выберем случайным образом t векторов s_1, \dots, s_t так, чтобы их сумма совпадала с вектором s , и распределим их между пользователями. Теперь, собравшись вместе, они могут легко восстановить значение ключа s , в то время как никакая группа, состоящая из меньшего числа пользователей, не сможет этого сделать. Действительно, в данном случае отсутствие хотя бы одной доли приводит к полной неопределенности относительно значения секрета, поскольку для каждого значения искомого секрета найдется возможный вариант значения отсутствующей доли.

Заметим, что если бы в предыдущем примере просто разбили вектор на t частей, то такая схема не могла быть схемой разделения секрета, так как знание любой доли давало бы частичную информацию о секрете s .

Другой пример схемы разделения секрета дает *пороговая схема Шамира*. Пусть $1 < t \leq n$. Схема разделения секрета между n пользователями называется (n, t) -пороговой [13], если любая группа из t пользователей может восстановить секрет, в то время как никакая группа из меньшего числа пользователей не может получить никакой информации о секрете.

Для построения (n, t) -пороговой схемы А. Шамир предложил использовать многочлен степени $t - 1$ над конечным полем с достаточно большим числом элементов [13]. Многочлен степени $t - 1$ можно однозначно восстановить по его значениям в t различных точках, но при этом меньшее число точек использовать для интерполяции нельзя.

Выберем поле F и зафиксируем n различных несекретных элементов $r_1, \dots, r_n \in F$, отличных от нуля. Каждый элемент r_i приписан i -му абоненту сети, $i = \overline{1, n}$. Выберем также t случайных элементов a_0, \dots, a_{t-1} поля F и составим из них многочлен $f(x)$ над полем F степени $t - 1$, $1 < t \leq n$:

$$f(x) = \sum_{i=0}^{t-1} a_i x^i.$$

Положим $s = f(0) = a_0$. Вычислим теперь значения $s_1 = f(r_1), \dots, s_n = f(r_n)$ и распределим в качестве долей между участниками наборы $\{s_i\}_{i=1}^n$.

Для восстановления секрета S можно воспользоваться интерполяционной формулой Лагранжа. Пусть имеются t пар (x_i, y_i) , где $y_i = f(x_i)$. Тогда формула Лагранжа имеет вид:

$$f(x) = \sum_{i=0}^{t-1} y_i \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}.$$

Так как $s = f(0)$, то из формулы Лагранжа получаем равенства $f(x) = \sum_{i=0}^{t-1} y_i c_i$, $c_i = \prod_{j \neq i} \frac{x_j}{x_j - x_i}$,

причем коэффициенты c не зависят от коэффициентов многочлена $f(x)$ и могут быть вычислены заранее.

С помощью полученной формулы любая группа из t пользователей может легко восстановить секрет. В то же время можно доказать, что никакая группа из меньшего числа пользователей не может получить никакой информации о секрете.

Схема Шамира удобна тем, что она позволяет легко увеличивать число пользователей. Для этого не нужно ничего менять, кроме множества $\{r_1, \dots, r_n\}$, к которому следует добавить новые элементы r_{n+1}, \dots, r_{n+w} . Компрометация одной доли делает из (n, t) -пороговой схемы $(n-1, t-1)$ -пороговую схему [13].

Способы установления ключей для конференц-связи. Еще один тип распределения ключей между группами пользователей дают протоколы распределения ключей для проведения конференц-связи. Несмотря на внешнюю схожесть с протоколами разделения секрета, они имеют несколько принципиальных отличий. Если протоколы разделения секрета осуществляют предварительное распределение одного и того же ключевого значения (секрета) по секретным каналам между привилегированными группами пользователей, то протоколы конференц-связи осуществляют динамическое распределение ключей по открытым каналам связи между привилегированными группами пользователей. При этом ключи должны быть различными для каждой группы.

Тривиальный пример распределения ключей для проведения конференц-связи дает использование централизованного распределения ключей с помощью одного из трехсторонних

протоколов передачи ключей, используемых для симметричных шифрсистем. Для реализации такого подхода нужно выделить одного из пользователей группы и возложить на него функции центра генерации и распределения ключей. Естественно, что при этом возрастают требования к доверенности и безопасности выделенного пользователя, что вносит серьезную асимметрию между участниками конференц-связи.

Другой подход основан на использовании идеи открытого распределения ключей.

Приведем примеры протоколов, в которых все участники группы имеют одинаковые полномочия и выполняют симметричные функции. Простейший пример такого протокола для группы из трех участников можно получить, слегка модифицировав протокол открытого распределения ключей Диффи – Хеллмана. Участники протокола заранее договариваются о значениях большого простого числа p и образующего элемента α мультипликативной группы $Z_p^* = \{1, 2, \dots, p-1\}$ [1]. Для выработки общего ключа k пользователи A , B и C должны сгенерировать соответственно случайные числа x , y и z , $1 \leq x, y, z \leq p-2$. Затем они должны обмениваться сообщениями согласно следующему протоколу:

1. $A \rightarrow B: X = \alpha^x \bmod p$.
2. $B \rightarrow C: Y = \alpha^y \bmod p$.
3. $C \rightarrow A: Z = \alpha^z \bmod p$.
4. $A \rightarrow B: Z' = Z^x \bmod p$.
5. $B \rightarrow C: X' = X^y \bmod p$.
6. $C \rightarrow A: Y' = Y^z \bmod p$.

Искомый общий ключ $k = \alpha^{xyz} \bmod p$ теперь вычисляется пользователями A , B и C по формулам:

$$k = (Y')^x \bmod p;$$

$$k = (Z')^y \bmod p;$$

$$k = (X')^z \bmod p$$

соответственно.

Протокол формирования общего ключа для конференц-связи группы из t пользователей U_0, \dots, U_{t-1} . Как и в предыдущем протоколе, каждый пользователь U_i должен сгенерировать секретное случайное число r_i , $1 \leq r_i \leq p-2$ и вычислить открытую экспоненту $\tilde{r}_i = \alpha^{r_i} \bmod p$. Положим

$$A_i = \alpha^{r_0 r_1 + r_1 r_2 + \dots + r_{i-1} r_0} \bmod p = A_0 A_1 \dots A_{i-1} \bmod p.$$

Тогда общий ключ k имеет вид

$$A_i = \alpha^{r_0 r_1 + r_1 r_2 + \dots + r_{i-1} r_0} \bmod p = A_0 A_1 \dots A_{i-1} \bmod p.$$

Протокол состоит из следующих шагов:

1. Каждый пользователь U рассылает z_i остальным $t - 1$ пользователям.

2. Каждый пользователь U вычисляет значение $x_i \in \mathbb{Z}_m$ и рассылает его остальным $t - 1$ пользователям.

3. Каждый пользователь U_i вычисляет значение общего ключа k по формуле

$$k = \alpha^{x_1 z_1 + x_2 z_2 + \dots + x_t z_t} \bmod p.$$

Протокол требует передачи $2t(t - 1)$ сообщений, причем каждый пользователь должен отправлять сообщения всем остальным. Можно модифицировать протокол для случая обмена сообщениями по схеме двунаправленного кольца.

Рассмотренный протокол не решает задачи аутентификации, поскольку в нем не заложено процедур для взаимной аутентификации сторон.

6.2. Особенности использования вычислительной техники в криптографии

Методы применения шифрования данных в локальных вычислительных сетях. В настоящее время применяется два основных метода шифрования при передаче информации – абонентское шифрование и канальное шифрование (рис. 37).

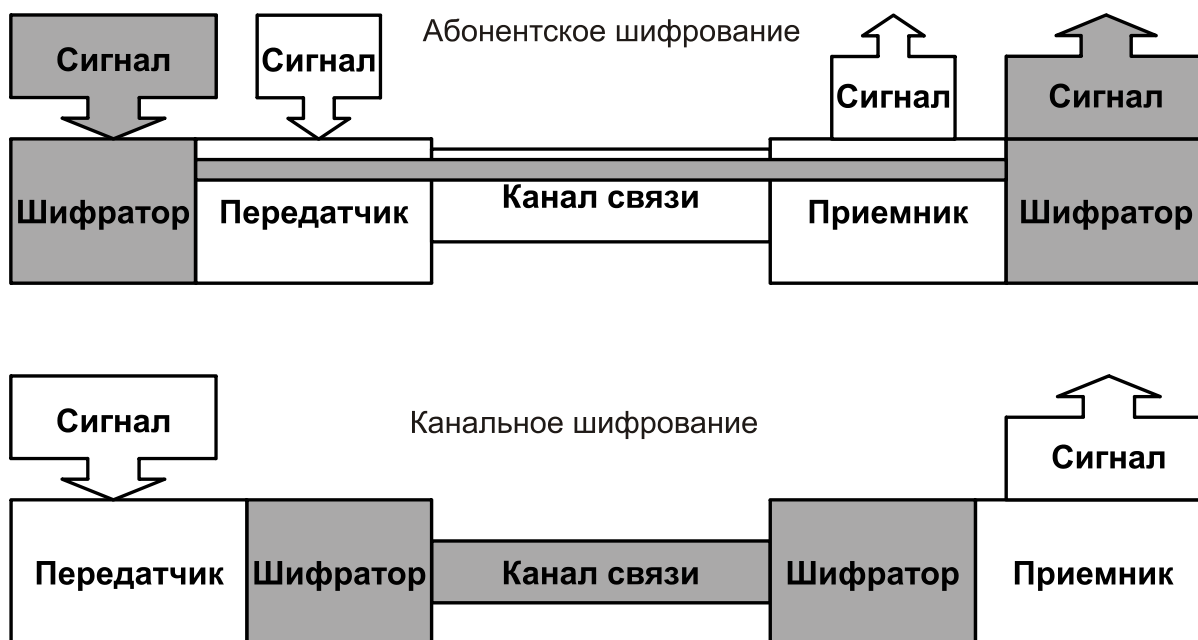


Рис.37. Схемы абонентского и канального шифрования

При абонентском шифровании для передачи данных используется открытый канал. Это позволяет параллельно обрабатывать на компьютере и конфиденциальные данные и открытую информацию. При использовании этого метода обычно используют программные реализации шифраторов.

При канальном шифровании шифруется весь передаваемый по сети трафик, как конфиденциальные данные, так и открытая информация. При таком подходе рационально применение аппаратного шифрования.

Хранение данных в компьютере. Хранимая в компьютере конфиденциальная информация делится на две группы секретов: ключи и данные.

Для обеих групп возможны два вида хранения:

- долгосрочное хранение – внешние носители;
- краткосрочное хранение – оперативная память.

Обеспечение секретности данных при долгосрочном хранении. В последнее время для характеристики способа шифрования хранимой на диске информации все чаще применяют два термина: «прозрачное шифрование» и «непрозрачное шифрование». На деле же речь идет о двух подходах к шифрованию – защите всего логического диска или же защите каждого файла в отдельности.

Каждый из подходов имеет свои достоинства и недостатки (табл 12).

Задачи обеспечения секретности и целостности данных и ключей при краткосрочном хранении. При разработке программного обеспечения по защите данных с использованием криптографических алгоритмов необходимо уделять особое внимание решению следующих задач:

Таблица 12.

Подходы к защите данных при долговременном хранении.

	Защита диска	Защита файла
Достоинства	<p>Доступ ко всему содержимому диска по одному ключу.</p> <p>Невозможность аналитической оценки содержимого.</p> <p>«Прозрачность» шифрования</p>	<p>Гибкость работы с файлами.</p> <p>Возможность использовать любой режим и алгоритм шифрования.</p> <p>Экономия вычислительной мощности</p>
Недостатки	<p>Рационально использовать только режим шифрования «Электронно-цифровой книги».</p> <p>Большие затраты вычислительной мощности.</p> <p>Негибкость работы</p>	<p>Шифрование каждого файла на своем ключе.</p> <p>Возможность аналитической оценки содержимого дисков.</p> <p>«Непрозрачность» шифрования</p>

1. *Уничтожение состояния* – одно из основных правил. Как только какая-то информация станет ненужной, она подлежит уничтожению. Уничтожать данные необходимо до потери контроля над их носителем. Что касается краткосрочного хранения, это в

первую очередь очистка соответствующих областей оперативной памяти.

2. *Уничтожение файла подкачки.* Большинство систем виртуальной памяти не предпринимают серьезных попыток по шифрованию данных при переносе их в файл подкачки, а сам файл может стать достоянием противника.

3. *Очистка кеша.* В нем могут храниться копии секретных данных. В принципе опасность утечки данных из кеша невелика, так как к нему имеет доступ только код операционной системы. Степень безопасности определяется степенью доверия к операционной системе.

4. *Предупреждение удерживания данных в памяти* – простое перезаписывание данных не всегда удаляет их из памяти. Оперативная память может не полностью стираться при выключении питания. Доступ к ранее «удаленным» данным может быть осуществлен в недокументированных тестовых режимах. Частичным решением проблемы является метод boojum. При этом методе данные в оперативной памяти хранятся не в явном виде. Для хранения данных m используется случайная величина R той же длины и в два разных участка памяти записываются R и $R \oplus m$. Для вызова данных производится вычисление $m = R \oplus (R \oplus m)$. Значение R меняется, например, каждые 100 сек..

5. *Предупреждение доступа других программ* – наиболее сложно решаемая проблема. Windows дает возможность подключения отладчика к любому процессу. UNIX-системы позволяют создавать дампы памяти, а супервизоры вообще могут получить прямой доступ к любому участку памяти.

6. *Обеспечение целостности данных* – в основном проблема аппаратного обеспечения. В компьютере с 1 Гб памяти появление случайной ошибки данных в памяти следует ожидать каждые 32 часа.

Обеспечение секретности ключей при долгосрочном хранении. Основные способы хранения ключевой информации:

1. *Хранение на логическом диске ПК.*

Наиболее неудачное решение. При работе одного пользователя с несколькими компьютерами количество хранимых

экземпляров ключа возрастает. Соответственно растет и риск доступа к ним сторонних пользователей.

2. Хранение в памяти человека.

Недостатками данного метода является проблема запоминания стойких по длине и статистическим свойствам ключевых слов. Возникает дилемма: простой ключ легко запомнить, но он не стоек, и наоборот – сложный ключ стоек, но труден к запоминанию. Существует два подхода к решению этой дилеммы. Первый подход – использование в качестве пароля легко запоминающейся идентификационной фразы наподобие: «Розовые занавески порхают над океанами». Ограничение данного подхода – при помощи идентификационных фраз трудно хранить в человеческой памяти более 128 бит энтропии.

Второй подход – использование программно реализуемого метода «подсаливания и растягивания» (salt&stretch). Первый этап – добавление «соли». Соль – случайное число s , хранящееся вместе с данными, зашифрованными с помощью пароля. По возможности необходимо использовать 256-битовое значение s .

Второй этап – растягивание пароля p . Достаточно долгий процесс вычисления с использованием любой криптографически сильной функции хеширования h :

$$\begin{aligned}x_0 &:= 0, \\x_i &:= h(x_{i-1} \parallel p \parallel s) \text{ для } i = 1, \dots, r, \\K &:= x_r.\end{aligned}$$

Полученное значение K применяется для фактического шифрования данных. Параметр r – количество итераций данного алгоритма, которое должно быть максимально возможным.

3. Использование портативных хранилищ.

Использование флеш-карт, дискет, даже клочков бумаги для внешнего хранения ключей. Недостаток заключается в необходимости надежного хранения носителя.

4. Использование идентификаторов безопасности.

Более удачное и более дорогостоящее решение, – использование мини-компьютера, носимого с собой (смарт-карта, USB ключ, таблетка e-token и пр.). При этом сам идентификатор при потере обеспечивает достаточную защиту от злоумышленников. В частности, доступ к хранимым на нем ключам в свою очередь может быть защищен паролем.

Подобные устройства защищены также и от физического взлома. (Стопроцентной защиты, конечно, не существует, но весь вопрос во времени и стоимости взлома.) Одна из проблем – халатное поведение пользователей, оставляющих свой идентификатор подключенным к компьютеру при покидании рабочего места. Другой недостаток – пароль защиты вводится не непосредственно в само устройство, а через компьютер, что не гарантирует защиты от программных закладок.

5. *Безопасный пользовательский интерфейс.*

Развитие предыдущей идеи. В качестве идентификатора используются носимые ПК типа Palm, КПК, iPhone и пр. Метод становится еще более затратным.

6. *Использование биометрических параметров.*

Сканирование подписей, отпечатков пальцев, сетчатки глаза и пр. Надежность (и стоимость) методов все более возрастает с увеличением сложности биометрических сканеров.

7. *Системы однократной регистрации.*

Предоставление пользователю одного простого мастер-ключа, используемого для расшифровки остальных ключей, применяемых в различных приложениях. Достигается определенная прозрачность, но стойкость данной системы определяется стойкостью мастер-ключа. Метод очень напоминает хранение всех яиц в одной корзине.

8. *Разделение секрета.*

Применяется для совместного разделенного хранения ключа несколькими лицами.

Защита от атак с использованием побочных каналов. Существует целый класс атак, называемых атаками с использованием побочных каналов, основные из них – это:

- Атаки измерения энергии и радиоизлучения (особенно успешны при взломе смарт-карт);
- Атаки измерения времени (тайминг-атаки, дающие представление об ориентировочном размере секретного ключа).

Защита от атак первого вида криптографическими средствами не обеспечивается. Необходимо предусматривать экранирование физически при помощи конструктивных доработок.

Для борьбы с тайминг-атаками применяются два метода:

- добавление в конец каждого вычисления случайной задержки;
- стандартизация времени выполнения операции.

Следует учитывать особенности современных компиляторов по оптимизации программного кода. При компиляции программы в некоторых средах программирования компилятор удаляет все «лишние по его мнению» инструкции.

Контрольные вопросы

1. Какие бывают виды криптографических протоколов?
2. Для чего предназначен «билет» в протоколе Kerberos?
3. С какой целью применяется протокол X.509?
4. Схема предварительного распределения ключей.
5. Для чего служит схема разделения секрета?
6. В чем различие между канальным и абонентским шифрованием?
7. Перечислите атаки с использованием побочных каналов.

7. КРИПТОАНАЛИЗ И ПЕРЕСПЕКТИВНЫЕ НАПРАВЛЕНИЯ В КРИПТОГРАФИИ

7.1. Основные методы криптоанализа

Атаки на симметричные криптоалгоритмы. Существует множество типов атак на симметричные алгоритмы шифрования и хеш-функции, каждый из которых обладает своей степенью сложности:

1. *Атака с использованием только зашифрованного текста (ciphertext-only attack).*

Это самая сложная атака, при которой злоумышленник обладает минимальным объемом информации. Цель атаки – нахождение ключа и/или открытого текста.

2. *Атака с известным открытым текстом (known plaintext attack).*

Злоумышленник знает один или произвольное количество открытых текстов и соответствующих им шифртекстов. Цель атаки – нахождение ключа и/или открытых текстов, зашифрованных на том же ключе.

3. *Атаки с избранным открытым текстом (chosen plaintext attack): автономная (offline) и оперативная (online).*

Злоумышленник имеет возможность сам выбирать один или произвольное количество открытых текстов и вычислять соответствующий ему шифртекст. Набор открытых текстов подготавливается заранее (автономно) или генерируется в процессе атаки (оперативно). Цель атаки – нахождение ключа.

4. *Атака с избранным зашифрованным текстом (chosen ciphertext attack).*

Злоумышленник имеет возможность выбирать как открытый текст (получая соответствующий ему зашифрованный текст), так и зашифрованный (с получением соответствующего ему открытого текста). Цель атаки – нахождение ключа.

5. *Различающие атаки (distinguishing attack).* Это любые нетривиальные методы, позволяющие обнаружить различие между идеальным и реальным шифром. Цель атаки – дискредитация криптосистемы.

6. *Атаки на основе коллизий.*

- *Атака на основе парадокса задачи о днях рождения (birthday attack) на хеш-функции.*

- *Двусторонняя атака, или «встреча посередине» (meet-in-the-middle attack), на код аутентификации.*

Цель – подмена-имитация сообщений.

Отдельно различают *атаки с использованием активного аппаратного воздействия на криптосистему* при помощи различных излучений, вызывающих помехи и ошибки. Данные атаки возникли в связи с массовым использованием интеллектуальных электронных карточек (smart cards).

Метод прямого перебора (brute force). Данный метод используется при атаках типа 1 и 2. В случае атаки типа 2 на основе полного шифртекста или его фрагмента и соответствующего ему открытого текста осуществляют последовательное расшифрование на всем множестве возможных ключей до совпадения открытого текста. При этом длина известного фрагмента открытого текста должна превышать расстояние единственности. В случае атаки типа 1 при неизвестном открытом тексте вычислительная сложность метода увеличивается, так как необходимо введение дополнительного критерия на «осмысленность» открытого текста.

Метод прямого перебора возможен только при известном алгоритме шифрования, и алгоритм, лежащий в его основе, является экспоненциальным.

Стойкость криптосистемы по отношению к методу прямого перебора полностью определяется мощностью ключевого пространства $|K|$ или энтропией ключа (при переборе со словарем). Минимальные требования к энтропии ключа современных симметричных криптосистем – 128 бит. В настоящее время рекомендуется использовать 256-битовые ключи.

Методы криптоанализа с использованием теории статистических решений. Цель статистического криптоанализа – получение открытого текста или некоторой вероятностной информации о нем. Существуют два основных метода статистического криптоанализа:

Таблица 13.

Мощностные характеристики пространства ключей.

Криптосистема	$ K $	Среднее время перебора для INTEL ASCI RED
DES	$7,21 \cdot 10^{16}$	9,4 ч
IDEA	$3,4 \cdot 10^{38}$	$1,3 \cdot 10^{21}$ лет
ГОСТ 28147-89	$1,16 \cdot 10^{77}$	$1,7 \cdot 10^{58}$ лет

-метод максимального правдоподобия (метод частотного анализа), базирующийся на логарифмической функции правдоподобия, которая строится на таблицах частотных характеристик предполагаемого открытого текста и шифртекста;

-байесовский метод, основанный на предположении о случайном ключе с известным распределением вероятностей, функции потерь, функционала риска и байесовской оценки, минимизирующей функционал риска.

Анализ поточных криптосистем. При проведении криптоанализа поточных алгоритмов решаются следующие задачи:

1. Распознавание ЛРП.

Если s_0, s_1, \dots – линейная рекуррентная последовательность с минимальным многочленом $m(\lambda)$ степени n , то *ганкелев определитель*

$$D_t^{(k)} = \begin{vmatrix} s_t & s_{t+1} & s_{t+k-1} \\ s_{t+1} & s_{t+2} & s_{t+k} \\ \dots & \dots & \dots \\ s_{t+k-1} & s_{t+k} & s_{t+2k-2} \end{vmatrix}$$

равен нулю для всех $k \geq n + 1$ и всех $t \geq 0$.

С другой стороны, если s_0, s_1, \dots – последовательность независимых случайных величин $0 < p(s_t = 0) < 1$, то

$$D_t^{(k)} \rightarrow 0$$

при $T \rightarrow \infty$ для любого фиксированного натурального k .

2. Оценка параметров ЛРП.

Производится на основе алгоритма Берленкэмп – Мэсси, который позволяет по $2k$ -отрезку s_0, \dots, s_{2k-1} линейной рекуррентной

последовательности определить минимальный многочлен $m(\lambda)$, если степень последнего не превышает k .

3. Определение начального состояния ЛРП.

Производится на основе опробования различных начальных векторов S_0 и сравнения результатов генерации s_0, s_1, \dots с известными фрагментами перехваченной искаженной последовательности (шифртекста). Задача упрощается при малом количестве ненулевых коэффициентов характеристического многочлена, что свойственно часто употребляемым прореженным многочленам.

4. Корреляционный анализ (применяется для комбинированных генераторов различных типов).

Анализ блочных криптосистем. На сегодняшний день двумя самыми известными методами анализа блочных криптосистем являются дифференциальный (разностный) и линейный криптоанализ. Первый реализует атаку типа 3, второй – атаку типа 2.

Дифференциальный криптоанализ. Первые открытые упоминания встречаются в литературе с 1990 года в работах Мерфи, Бихема и Шамира. Методика анализа строится индивидуально для каждого алгоритма и основана на знании пар сообщений m и m' , для которых известны соответствующие шифртексты $E_k(m)$ и $E_k(m')$ и XOR-разности между ними с рассмотрением разности между промежуточными частями блоков сообщений.

Для двухветвевой сети Фейстеля $\Delta m_i = m_i \oplus m'_i$. При этом для каждого раунда в отдельности справедливо:

$$\Delta m_{i+1} = m_{i+1} \oplus m'_{i+1} = \Delta m_{i-1} \oplus [f(m_i, K_i) \oplus f(m'_i, K_i)].$$

Предполагается, что для многих пар входных данных функции f , имеющих одну и ту же XOR-разность, при использовании одного и того же подключа одинаковой оказывается и XOR-разность для соответствующих выходных пар. Формально говорят, что X *влечет* Y с вероятностью p , если для относительной доли p входных пар с XOR-разностью X для соответствующих выходных пар XOR-разность оказывается равной Y . Предположим теперь, что имеется ряд значений X , которые с высокой вероятностью влекут определенные разности выходных значений. Тогда если с большой

вероятностью известны значения Δm_{i-1} и Δm_i то с большой вероятностью можно определить и Δm_{i-1} . А если удастся найти достаточно много таких значений разности, становится возможным определить подключ, используемый функцией f .

Общая стратегия дифференциального криптоанализа основывается на представленной выше методике для каждого раунда шифрования в отдельности. Процедура начинается с рассмотрения двух сообщений открытого текста m и m' с заданной разностью и имеет своей целью получение вероятностных оценок разностей промежуточных результатов последовательно раундов за раундом с тем, чтобы определить вероятностную оценку для разности соответствующих шифрованных текстов. На самом деле получаются вероятностные оценки для двух 32-битовых половинок сообщения ($m_{17} \parallel m_{16}$).

После этого выполняется шифрование m и m' , чтобы определить реальные разности при неизвестном ключе, а затем сравнить результат с вычисленными вероятностными оценками. Если результаты совпадают, т. е. $E_k(m) \oplus E_k(m') = (\Delta m_{17} \parallel m_{16})$, то можно ожидать, что вероятностные оценки для всех раундов соответствуют реальности. Это позволяет сделать определенные заключения о некоторых битах ключа. Чтобы определить все биты ключа, процедуру приходится повторять много раз.

Линейный криптоанализ. Более новым направлением в криптоанализе является линейный криптоанализ. Данный метод заключается в построении линейной аппроксимации преобразования, выполняемого в ходе шифрования DES. Он позволяет найти ключ DES при наличии 2^{47} известных фрагментов открытого текста, тогда как для дифференциального криптоанализа требуется 2^{47} фрагментов избранного открытого текста.

Пусть для шифра с n -битовыми блоками открытого и шифрованного текста и m -битовым ключом блок открытого текста будет обозначен $P[1], \dots, P[n]$, блок шифрованного текста – $C[1], \dots, C[n]$, а ключ – $K[1], \dots, K[n]$. Тогда определим

$$A[i, j, \dots, k] = A[i] \oplus A[j] \oplus \dots \oplus A[k].$$

Целью линейного криптоанализа является нахождение подходящего *линейного* уравнения вида

$$P[\alpha_1, \alpha_2, \dots, \alpha_a] \oplus C[\beta_1, \beta_2, \dots, \beta_b] = K[\gamma_1, \gamma_2, \dots, \gamma_c],$$
 выполняющегося с вероятностью $p \neq 0,5$ (где $x = 0$ или 1 , $1 \leq a$, $b \leq n$, $1 \leq c \leq m$, а α , β и γ представляют фиксированные битовые позиции в блоках). Чем больше значение p отклоняется от $0,5$, тем более подходящим считается уравнение.

После этого вычисляются значения левой части данного уравнения для большого числа пар соответствующих фрагментов открытого и зашифрованного текста. Если результат оказывается равным нулю более чем в половине случаев, полагают, что $K[\gamma_1, \gamma_2, \dots, \gamma_c] = 0$. Если же в большинстве случаев получается единицу, полагают, что $K[\gamma_1, \gamma_2, \dots, \gamma_c] = 1$.

Достаточное количество полученных таким образом равенств образует систему, решением которой определяется ключ. Ввиду линейности получаемых уравнений проблему можно решать для каждого раунда шифрования в отдельности, а полученные результаты затем объединить.

Атаки на хеш-функции и коды аутентичности. *Атака «дней рождения».* Этот тип атаки основан на том факте, что одинаковые значения, называемые также *коллизиями (collisions)*, появляются намного быстрее, чем можно было ожидать. В системе финансовых транзакций, в которой для обеспечения безопасности каждой транзакции применяется новый 64-битовый ключ аутентификации (Для простоты предполагается, что шифрование не используется.), существует 2^{64} возможных значения ключа (это больше, чем $18 \cdot 10^{18}$).

Отследив около 2^{32} транзакций, злоумышленник может предположить, что две из них используют один и тот же ключ. Допустим, что сообщение-заголовок, передаваемое в ходе каждой транзакции, всегда одинаково. Если две транзакции используют один и тот же ключ аутентификации, тогда значения МАС первых сообщений этих транзакций будут совпадать, что легко отследит злоумышленник. Зная, что обе транзакции используют один и тот же ключ аутентификации, он сможет вставлять сообщения из более старой транзакции в более новую во время выполнения последней. Поскольку ложные сообщения успешно пройдут аутентификацию, они будут приняты, что является очевидным взломом системы финансовых транзакций.

В общем случае, если элемент может принимать N различных значений, ожидать первой коллизии можно после случайного выбора приблизительно \sqrt{N} элементов. В большинстве случаев говорят об n -битовых значениях. Поскольку n -битовый элемент может иметь 2^n возможных значения, необходимо извлечь $\sqrt{2^n} = 2^{n/2}$ элементов множества, чтобы надеяться на возникновение коллизии. Назовем это оценкой $2^{n/2}$ или *оценкой парадокса задачи о днях рождения (birthday bound)*.

Двусторонняя атака. Это является разновидность атак, в основе которых лежит парадокс задачи о днях рождения, носит также название *атаки «встреча на середине» (meet-in-the-middle attacks)*. (В совокупности оба типа атак называются *атаками на основе коллизий (collision attacks)*.) Этот тип атак более распространен и результативен.

В системе финансовых транзакций, в которой для каждой транзакции применяется новый 64-битовый ключ, используя двустороннюю атаку, злоумышленник может еще более продвинуться во взломе системы. Для этого он случайным образом выбирает 2^{32} различных 64-битовых ключа. По каждому из них он подсчитывает значение МАС для сообщения-заголовка. Полученное значение МАС вместе с соответствующим ключом помещается в таблицу.

Затем злоумышленник прослушивает каждую транзакцию и проверяет, не окажется ли значение МАС первого сообщения этой транзакции в его таблице. Если такая транзакция находится, значит, с высокой долей вероятности ее ключом аутентификации является тот самый ключ, который был сгенерирован злоумышленником и помещен в таблицу вместе с соответствующим значением МАС. Теперь, когда злоумышленник обладает ключом аутентификации транзакции, он может вставлять в нее собственные сообщения с любым нужным ему текстом. (Предыдущий тип атаки позволял злоумышленнику вставлять только сообщения, взятые из более старой транзакции.)

После 2^{32} транзакций злоумышленник может ожидать появления транзакции, использующей ключ из его таблицы. Таким образом, злоумышленнику придется проделать 2^{32}

предварительных подсчета и прослушать 2^{32} транзакции. Это намного меньше, чем перебирать все 2^{64} возможных ключа.

Различие между атакой, в основе которой лежит парадокс задачи о днях рождения, и двусторонней атакой состоит в следующем. В первом случае ждем, когда одно и то же значение появится дважды в одном множестве элементов. В двусторонней атаке есть два множества элементов, и ждем, когда эти множества пересекутся. И в том и в другом случае ожидать первого результата можно примерно через одинаковое количество элементов.

Двусторонняя атака является более гибкой, чем атака, в основе которой лежит парадокс задачи о днях рождения. Предположим, что элементы обоих множеств могут принимать N возможных значений. Пусть в первом множестве содержится P элементов, а во втором – Q элементов. Из них можно образовать PQ различных пар, в которых один элемент будет принадлежать первому множеству, а другой – второму множеству. Для каждой пары вероятность того, что значения ее элементов совпадут, равна $1/N$. Можем ожидать возникновения коллизии, когда значение PQ/N приближается к единице. В этом случае наиболее эффективным выбором будет

$$P \cong Q \cong N.$$

Двусторонняя атака обладает определенной гибкостью в отношении выбора P и Q . Иногда элементы одного множества легче получить, чем элементы второго, поэтому размер множеств может быть и неодинаков. Единственным требованием является соблюдение условия $PQ \cong N$. Можно выбрать $P \cong N^{1/3}$, а $Q \cong N^{2/3}$. В приведенном выше примере злоумышленник может составить таблицу из 2^{40} значений МАС и ожидать первого совпадения уже после 2^{24} прослушанных транзакций.

Атаки на асимметричные криптосистемы. Атаки на криптосистемы, основанные на сложности задачи факторизации. Очевидное направление криптоанализа систем шифрования и цифровой подписи, основанных на сложности решения задачи факторизации, – разработка перспективных методов факторизации целых чисел.

Среди методов факторизации целых чисел наиболее известные субэкспоненциальные и с экспоненциальной сложностью.

К последним относятся:

1. Метод пробных делений.
2. Алгоритм Ферма.
3. $(p - 1)$ -метод Полларда.
4. δ -метод Полларда.
5. Метод Шермана – Лемана.
6. Алгоритм Ленстры.
7. Алгоритм Полларда – Штрассена.
8. $(p + 1)$ -метод Уильямса.
9. Метод Шэнкса.

Практическая значимость данных алгоритмов, как правило, невелика. Наибольшие успехи в решении задачи факторизации достигнуты с помощью субэкспоненциальных методов, таких как:

1. Метод Диксона.
2. Алгоритм Бриллхарта – Моррисона.
3. Метод квадратичного решета QS (quadratic sieve), предложенный К. Померансом в 1981 году; эффективен для чисел $n < 10^{110}$.
4. Методы Шнорра – Ленстры и Ленстры – Померанса (активно на практике не использовались).
5. Обобщенный метод решета числового поля GNFS (General number field sieve). На сегодняшний день лучший по показателям из официально опубликованных методов факторизации, предложен в 1990 году.

6. Алгоритм Ленстры для факторизации с помощью эллиптических кривых (аналогичен $(p - 1)$ -методу Полларда, но применяются вычисления в группе точек эллиптической кривой).

Последнее достижение – факторизация числа RSA-155, потребовавшая:

- использования 160 рабочих станций SGI и Sun (175-400 МГц);
- восемь процессоров SGI Origin 2000 (250 МГц);
- 120 персональных компьютеров Pentium II (300-450 МГц);
- четыре кластера Digital\Compaq (500 МГц);
- девять недель для настройки параметров GNFS;
- 5,2 месяца для сбора вспомогательных пар чисел;

- 3,2 Гб оперативной памяти, 224 часа вычислений на Cray C916 для обработки матрицы размером 6699191×6711336 с 417132631 ненулевыми элементами.

Таблица 14.

Рекорды факторизации на примере системы RSA.

Число	Дата	Сложность, MIPS-лет	Алгоритм
RSA-100	Апрель 1991	7	QS
RSA-110	Апрель 1992	75	QS
RSA-120	Июнь 1993	830	QS
RSA-129	Апрель 1994	5000	QS
RSA-130	Апрель 1996	500	GNFS
RSA-140	Февраль 1999	2000	GNFS
RSA-155	Август 1999	8000	GNFS

Возможные атаки на систему RSA:

1. Факторизация модуля $n = p \cdot q$.

2. Решение сравнения $y \equiv x^e \pmod{n}$ для конкретного y – нахождение корня степени e из y по модулю p (задача не менее сложная, чем задача факторизации).

3. Метод повторного шифрования. Он сводится к поиску ключа d' для дешифрования на основе того, что существует такое натуральное число m , что $e^m \equiv 1 \pmod{(n)}$. Производится последовательное повторное зашифрование сообщения y : $y_1 = y$, $y_i \equiv y_{i-1}^e \pmod{n}$, $i > 1$ – до тех пор, пока не получим опять значения y : $y \equiv y_m \pmod{n}$; решением будет $x = y_{m-1}$. (Метод эффективен при неправильном подборе параметров RSA.)

4. Метод факторизации модуля системы на основе дискредитированного ключа d . (Нельзя использовать одинаковый модуль для многопользовательских систем, при потере хотя бы

одного ключа вскрывается вся система.) Кроме того, знание хотя бы одного секретного ключа многопользовательской системы с единым модулем позволяет найти любой другой секретный ключ даже без факторизации n .

5. Определение секретного ключа d простым перебором (при малом d).


6. Определение секретного ключа d с использованием непрерывных дробей по теореме Винера (при $d < \sqrt[4]{n}$).

7. Бесключевое чтение циркулярных сообщений на общей открытой экспоненте e и трех взаимно простых модулях n_1, n_2, n_3 посредством решения системы из трех сравнений.

Атаки на криптосистемы, основанные на сложности задачи дискретного логарифмирования. Очевидное направление криптоанализа систем шифрования и цифровой подписи, основанных на сложности решения задачи дискретного логарифмирования, – разработка перспективных методов дискретного логарифмирования.

Наиболее известные методы дискретного логарифмирования:

1. Алгоритм согласования.
2. Алгоритм Полига – Хеллмана.
3. δ -метод Полларда для дискретного логарифмирования.
4. Дискретное логарифмирование в простых полях (алгоритмы Адлемана, COS).
5. Дискретное логарифмирование в полях Галуа (алгоритмы index-calculus, Эль-Гамала, Копперсмита).
6. Алгоритмы решета числового поля для дискретного логарифмирования.

Логарифмирование по простому модулю  (алгоритм COS с гауссовыми целыми).

Криптоанализ рюкзачных систем. Доказано, что существует алгоритм полиномиальной сложности получения открытого текста по шифртексту на основе так называемой косозубой функции. Алгоритм известен под названием LLL (Ленстры–Ленстры–Ловаса).

7.2. Перспективные направления в криптографии

Эллиптические кривые. Проективная плоскость $P^2(K)$ над полем K определяется как множество троек (X, Y, Z) , не равных одновременно нулю элементов $X, Y, Z \in K$, на котором введено отношение эквивалентности:

$$(X, Y, Z) \sim (\lambda X, \lambda Y, \lambda Z) \text{ для любых } \lambda \in K^*.$$

Так, например, две точки $(4, 1, 1)$ и $(5, 3, 3)$ эквивалентны в $P^2(\mathbb{F}_7)$.

Класс эквивалентности троек называется проективной точкой. Эллиптическая кривая E – это множество точек проективной плоскости, удовлетворяющих однородному уравнению Вейерштрасса

$$E: F(X, Y, Z) = -X^3 + Y^2Z + a_1XYZ - a_2X^2Z + a_3YZ^2 - a_4XZ^2 - a_6Z^3 = 0 \quad [3] \quad \text{с } a_i \in K.$$

Это уравнение называют также длинной формулой Вейерштрасса. Кривая должна быть неособой в том смысле, что частные производные $\frac{\partial F}{\partial X}, \frac{\partial F}{\partial Y}, \frac{\partial F}{\partial Z}$ не должны обращаться в нуль одновременно ни в одной ее точке.

Множество K -рациональных точек кривой E (точек, удовлетворяющих уравнению кривой) обозначается через $E(K)$.

Кривая имеет только одну точку, чья координата $Z = 0$, а именно $(0, 1, 0)$. Ее принято называть бесконечно удаленной точкой (или точкой на бесконечности) и обозначать символом O . Для удобства часто пользуются аффинной версией уравнения Вейерштрасса [3]:

$$E: Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6, \text{ с } a_i \in K.$$

K -рациональные точки в аффинном случае – это решения уравнения в K^2 и бесконечно удаленная точка O .

Переход от аффинных к проективным координатам:

- точка на бесконечности всегда переходит в бесконечно удаленную точку, как при переходе от аффинных координат к проективным, так и наоборот;

- проективная точка (X, Y, Z) кривой, отличная от бесконечно удаленной ($Z \neq 0$), переходит в аффинную точку с координатами $(X/Z, Y/Z)$;

- чтобы найти проективные координаты аффинной точки (X, Y) , не лежащей на бесконечности, достаточно выбрать произвольное значение $Z \in K^*$ и вычислить $(X \cdot Z, Y \cdot Z, Z)$.

Иногда удобнее пользоваться модифицированной формой проективной плоскости, когда проективные координаты (X, Y, Z) представляют аффинную точку $(X/Z^2, Y/Z^3)$.

Для эллиптической кривой вводятся следующие константы

$$b_2 = a_1^2 + 4a_2,$$

$$b_4 = a_1 a_3 + 2a_4,$$

$$b_6 = a_2^3 + 4a_6,$$

$$[5]: \quad b_8 = a_1^2 a_6 + 4a_2 a_6 - a_1 a_3 a_4 + a_2 a_5^2 - a_4^2,$$

$$c_4 = b_2^2 - 24a_4,$$

$$c_6 = -b_2^3 + 36b_2 b_4 - 216a_6.$$

Дискриминант кривой E определяется по формуле

$$\Delta = -b_2^2 b_4^2 - 4b_2^2 b_6^2 - 4b_4^2 b_6^2 - 1728a_6^2.$$

Если $\text{char } K \neq 2, 3$, то дискриминант можно вычислить следующим образом:

$$\Delta = \frac{c_4^3 - c_6^2}{1728}.$$

Деление на $1728 = 2^6 3^3$ имеет смысл только в тех полях, чья характеристика отлична от 2 и 3. Кривая E – неособая тогда и только тогда, когда $\Delta \neq 0$. Далее рассматриваются только неособые кривые. Для неособых кривых вводится j -инвариант:

$$j(E) = \frac{c_4^3}{\Delta}.$$

Он тесно связан с понятием изоморфизма эллиптических кривых.

Кривая E с координатами X и Y изоморфна над полем K кривой E' с координатами X', Y' (обе заданы уравнением Вейерштрасса), если найдутся такие константы $r, s, t \in K$ и $u \in K^*$, что при замене переменных $X = u^2 X' + r$, $Y = u^3 Y' + s u^2 X' + t$ кривая E перейдет в кривую E' [3]. Изоморфизм кривых определен относительно поля K . Изоморфизм эллиптических кривых является отношением эквивалентности.

Лемма. Изоморфные над полем K кривые имеют один и тот же j -инвариант. С другой стороны, любые кривые с совпадающими j -инвариантами изоморфны над алгебраическим замыканием \bar{K} поля K . Иными словами j -инвариант разделяет классы

эквивалентности отношения изоморфизма над алгебраическим замыканием \bar{K} поля K .

Групповой закон. Рассмотрим для $\text{char } K \neq 2, 3$ замену переменных



переводящую кривую, заданную длинной формулой Вейерштрасса в изоморфную ей кривую [3], определяемую короткой формулой Вейерштрасса $E: Y^2 = X^3 + aX + b$ при некоторых $a, b \in K$. На таких представителях классов изоморфных эллиптических кривых можно наглядно ввести групповой закон методом хорд и касательных.

Сложение точек определяется с помощью хорд. Пусть P и Q – две точки кривой (рис. 38). Соединим их прямой линией. Она обязательно пересечет кривую в какой-то третьей точке R , поскольку пересекаем кубическую кривую прямой. Точка R будет определена над тем же полем, что сама кривая и исходные точки P и Q . Отразим затем точку R относительно горизонтальной оси координат и получим точку, определяемую над основным полем. Последняя точка и будет суммой $(P + Q)$.

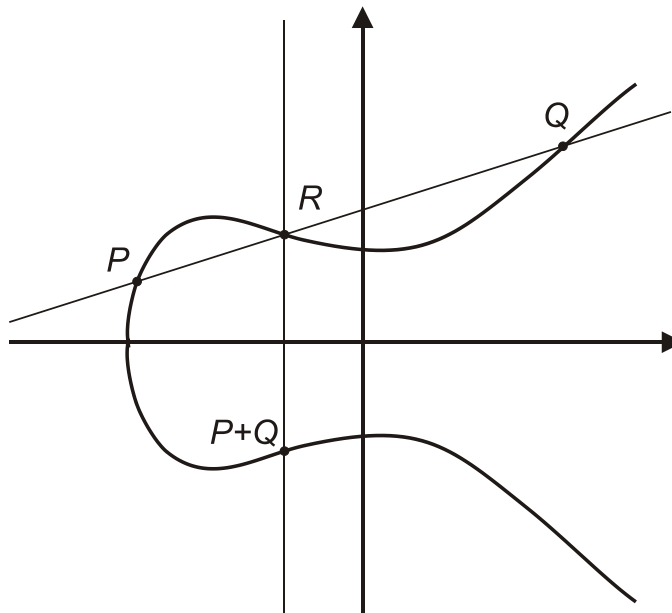


Рис.38. Групповой закон. Сложение точек.

Касательные служат для удвоения точек (используя хорду, нельзя сложить точку с собой). Пусть P – произвольная точка эллиптической кривой. Проведем касательную к кривой в точке P .

Она пересечет кривую в какой-то третьей точке R (кубическая кривая пересекается по трем точкам с учетом кратности пересечения). Отобразив R относительно горизонтальной оси, получим точку $P = P + P$ [2]. Вертикальная касательная в точке P «пересекает» кривую в бесконечно удаленной точке. В этой ситуации $P + P = O$, и говорят, что P – точка порядка 2.

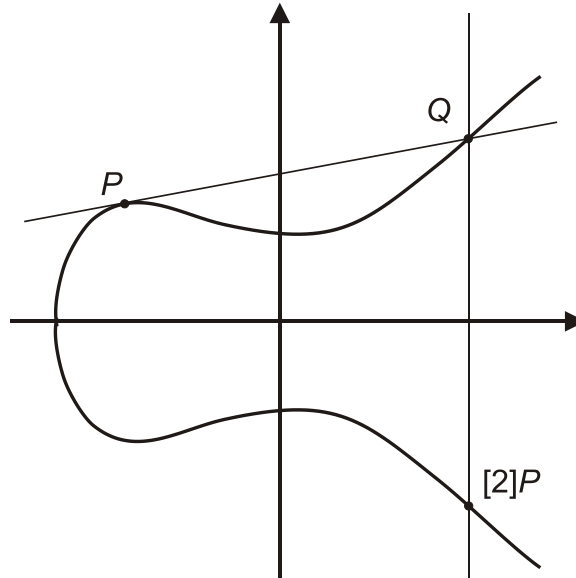


Рис.39. Групповой закон. Удвоение точек.

Метод хорд и касательных наделяет эллиптическую кривую структурой абелевой группы с бесконечно удаленной точкой в качестве нейтрального (единичного) элемента, т. е. нуля. Определение операций можно легко перенести на случай общей эллиптической кривой, заданной длинной формой Вейерштрасса (в частности характеристика поля может быть любой). Необходимо только заменить отражение относительно оси абсцисс на симметрию относительно прямой $Y = a_1X + a_3$ [3].

Алгебраические формулы, реализующие сложение точек по методу хорд и касательных. Лемма. Пусть E – эллиптическая кривая, определяемая уравнением $E: Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6$, на которой выбраны точки $P_1(x_1, y_1)$ и $P_2(x_2, y_2)$. Точка $-P_1$ имеет координаты $-P_1(x_1 - y_1 - a_1x_1 - a_3)$.

Введем коэффициенты

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}, \quad \mu = \frac{y_2 + y_1 + a_1x_1 + a_3}{x_2 - x_1}$$

при $x_1 \neq x_2$

и 

если $x_1 = x_2$, но $P_2 \neq -P_1$. Если $P_3(x_3, y_3) = P_1 + P_2 \neq O$, то x_3 и y_3 вычисляются по формулам:

$$x_3 = \frac{(y_2 - y_1)(x_1 + x_2) + x_1^2 + x_2^2}{2y_1 y_2},$$

Фиксируем натуральное число m и обозначим через $[m]$ отображение кривой на себя, сопоставляющее каждой точке P ее кратное $[m]P$, т. е.

$$\underbrace{P + P + \dots + P}_m = [m]P$$

Это отображение – основа криптографических систем, опирающихся на эллиптическую кривую, поскольку можно легко вычислить, но крайне сложно обратить, т. е. по данным координатам $P(x, y)$ и $[m]P(x', y')$ найти m очень трудно. (Конечно сложность обращения предполагает специальный выбор эллиптической кривой и соблюдение других условий [5].)

Эллиптические кривые над конечными полями. Количество \mathbf{F}_q рациональных точек над эллиптической кривой конечно. Обозначим его $\#E(\mathbf{F}_q)$. Ожидаемое число точек кривой близко к $q + 1$, и можно положить $\#E(\mathbf{F}_q) = q + 1 - t$, где «дефект» t называется следом отображения Фробениуса в q [5].

Теорема Хассе. След отображения Фробениуса удовлетворяет неравенству $|t| \leq 2\sqrt{q}$.

Есть два частных случая криптографически непригодных эллиптических кривых:

- кривая $E(\mathbf{F}_q)$ называется аномальной, если ее след Фробениуса равен единице, т. е. $\#E(\mathbf{F}_q) = q$. Эта кривая особенно неудобна, когда q – простое число;

- кривая $E(\mathbf{F}_q)$ называется суперсингулярной, если характеристика p поля \mathbf{F}_p делит след отображения Фробениуса t . Таких кривых также следует избегать в криптографии.

Выбирая кривую для шифрования, нужно стремиться к тому, чтобы число ее точек делилось на достаточно большое простое число. В связи с этим необходимо научиться вычислять порядок группы. Порядок произвольной группы $E(\mathbf{F}_q)$ над любым полем вычисляется за полиномиальное время.

Информация о порядке группы также существенна для оценки стойкости протокола, основанного на соответствующей кривой.

Одним из достоинств эллиптических кривых является то, что они составляют большое число возможных групп. Можно менять как основное поле, так и коэффициенты уравнения кривой. Отыскать эллиптическую кривую с хорошими криптографическими свойствами для создания безопасного протокола достаточно легко.

Как правило, реализация криптографических систем, основанных на эллиптической кривой, базируется на поле \mathbb{F}_{2^n} , чья характеристика равна 2, или на поле \mathbb{F}_p с большим простым числом p .

Проективные координаты. Одна из проблем, возникающих при использовании формул группового закона как при большой, так и при минимальной характеристике поля, связана с необходимостью деления. Деление в конечном поле считается дорогой операцией, так как включает в себя некий вариант расширенного алгоритма Евклида, который хотя и имеет приблизительно ту же сложность, что и умножение, однако не может быть реализован достаточно эффективно.

Во избежание операции деления применяют проективные координаты. При этом уравнение кривой записывается через три координаты (X, Y, Z) вместо двух (X, Y) . Однако вместо стандартного варианта уравнения кривой используется уравнение вида $E: Y^2 + a_1XYZ + a_3YZ^4 = X^3 + a_2X^2Z^2 + a_4XZ^4 + a_6Z^6$.

Точка на бесконечности здесь также имеет координаты $(0, 1, 0)$, но переход от аффинных координат к проективным осуществляется по правилу ~~$(X, Y, Z) \rightarrow (X, Y, Z)$~~ .

Выбор таких координат обусловлен стремлением сделать арифметические операции более эффективными.

Сжатие точек. Во многих криптографических протоколах возникает необходимость хранить в памяти или передавать по сети отдельные точки эллиптической кривой. В аффинных координатах это можно сделать при помощи двух элементов поля: координат x и y . Однако экономнее применять так называемую технику сжатия точек.

Метод сжатия точек работает благодаря тому, что уравнение кривой в аффинных координатах при фиксированном значении x

превращается в квадратное уравнение относительно координаты y . Значит, вместо двух координат для идентификации точки кривой можно хранить в памяти компьютера только координату x и еще некий двоичный параметр b , сообщающий о том, какое именно значение координаты y нужно брать.

Кривые над полем характеристики $p > 3$. Пусть основное поле $K = \mathbf{F}_q$ с $q = p^n$, где $p > 3$ – простое число и $n \geq 1$.

Уравнение кривой над таким полем можно представить в виде короткой формулы Вейерштрасса $E: Y^2 = X^3 + aX + b$ [3].

Ее дискриминант равен: $\Delta = -16(4a^3 + 27b^2)$, а j -инвариант – $j(E) = -1728(4a)^3 / \Delta$.

Формулы группового закона:

– $P_1 = (x_1, y_1)$ и, если $P_3(x_3, y_3) = P_1 + P_2 \neq O$, то координаты x_3, y_3 вычисляются следующим образом: ~~$x_3 = x_1^2 + x_2^2 + a$~~

где при $x_1 \neq x_2$ $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$,

а при $x_1 = x_2, y_1 \neq 0$ $\lambda = \frac{3x_1^2 + a}{2y_1}$.

В проективных координатах формулы сложения точек эллиптической кривой, заданной уравнением

$E: Y^2 = X^3 + aXZ^2 + bZ^6$, над полем характеристики $p > 3$ выглядят как ~~$x_3 = x_1^2 + x_2^2 + a$~~

где тройка координат ~~(x_3, y_3, z_3)~~ вычисляется последовательно по правилу:

$$\lambda_1 = x_1 z_2^2, \quad \lambda_2 = x_2 z_1^2,$$

$$\lambda_3 = \lambda_1 - \lambda_2, \quad \lambda_4 = y_1 z_2^3,$$

$$\lambda_5 = y_2 z_1^3, \quad \lambda_6 = \lambda_4 - \lambda_5,$$

$$\lambda_7 = \lambda_4 + \lambda_2, \quad \lambda_8 = \lambda_4 + \lambda_5,$$

$$z_3 = z_1 z_2 \lambda_3, \quad x_3 = \lambda_6^2 - \lambda_7 \lambda_8,$$

$$\lambda_9 = \lambda_7 \lambda_8 - 2x_3, \quad y_3 = (\lambda_9 \lambda_6 - \lambda_8 \lambda_3^3) / 2$$

Здесь нет ни одной операции деления, кроме деления на 2, которое легко заменяется умножением на заранее вычисленное число $2^{-1} \pmod{p}$.

Удвоение точек ~~(x_3, y_3, z_3)~~ упрощается с помощью формул:

1. Для каждого такого значения x , что $0 \leq x < p$, вычисляется $x^3 + ax + b \pmod{p}$.

2. Для каждого из полученных на предыдущем шаге значений выясняется, имеет ли это значение квадратный корень по модулю p (вычисляется символ Лежандра). Если нет, то в $E_p(a, b)$ нет точек с этим значением x . Если же корень существует, имеется два значения y , соответствующих операции извлечения квадратного корня (исключением является случай, когда единственным таким значением оказывается $y = 0$). Эти значения (x, y) и будут точками $E_p(a, b)$.

Таблица 15.

Точки на эллиптической кривой $E_{23}(1, 1)$

(0, 1)	(1, 7)	(3, 10)	(4, 0)	(5, 4)	(6, 4)	(7, 11)
(0, 22)	(1, 16)	(3, 13)	O	(5, 19)	(6, 19)	(7, 12)
(9, 7)	(11, 3)	(12, 4)	(13, 7)	(17, 3)	(18, 3)	(19, 5)
(9, 16)	(11, 20)	(12, 19)	(13, 16)	(17, 20)	(18, 20)	(19, 18)

Пример. Сложение и удвоение точек данной группы.
 $P = (3, 10)$, $Q = (9, 7)$.

Сложение:

$$\lambda = \frac{7 - 10}{9 - 3} = \frac{-3}{6} = \frac{-1}{2} \equiv 11 \pmod{23}$$

$$x_3 = 1^2 - 3 - 9 = -11 \equiv 12 \pmod{23}$$

$$y_3 = 1(7 - (-6)) = 13 \equiv 13 \pmod{23}$$

$$P + Q = (12, 13)$$

Удвоение:

$$\lambda = \frac{3(3^2) + 1}{2 \cdot 10} = \frac{5}{20} \equiv 6 \pmod{23}$$

$$x_3 = 6^2 - 3 - 9 = 3 \equiv 3 \pmod{23}$$

$$y_3 = 6(3 - 10) = -42 \equiv 12 \pmod{23}$$

$$2P = (3, 12)$$

Умножение определяется как повторное применение операции сложения [4]:

$$P = P + P + P + P.$$

Кривые над полем характеристики 2. Пусть основное поле

$K = \mathbf{F}_q$ с $q = 2^n$ при $n \geq 1$. В этом случае j -инвариант кривой вычисляется по формуле $j(E) = 4^2/\Delta$. Условие $j(E) = 0$, т. е. $a_1 = 0$, в характеристике 2 равносильно суперсингулярности кривой E , а такие кривые в криптографии не используются, поэтому полагаем, что $j(E) \neq 0$.

В этих предположениях представитель любого класса изоморфизма эллиптических кривых над \mathbf{F}_q записывается уравнением

$$E: Y^2 + XY = X^3 + a_2X^2 + a_6,$$

где $a_6 \in \mathbf{F}_q^*$ и $a_2 \in \{0, \gamma\}$. Здесь γ – фиксированный элемент поля \mathbf{F}_q , удовлетворяющий соотношению $\sum_{i=1}^{n-1} \gamma^{2^i} = 1$.

Формулы группового закона: $-P_1 = (x_1, y_1 - x_1)$ и, если

$P_3(x_3, y_3) = P_1 + P_2 \neq 0$, то

$$\begin{aligned} x_3 &= x_1^2x_2^2 + x_1^2x_2 + x_1x_2^2 + x_1x_2 + x_1 + x_2 + 1, \\ y_3 &= (x_1 + x_2)(x_1^2x_2^2 + x_1^2x_2 + x_1x_2^2 + x_1x_2 + x_1 + x_2 + 1) + x_1^2x_2^2 + x_1^2x_2 + x_1x_2^2 + x_1x_2 + x_1 + x_2 + 1. \end{aligned}$$

где при $x_1 \neq x_2$

$$\begin{aligned} x_3 &= x_1^2x_2^2 + x_1^2x_2 + x_1x_2^2 + x_1x_2 + x_1 + x_2 + 1, \\ y_3 &= (x_1 + x_2)(x_1^2x_2^2 + x_1^2x_2 + x_1x_2^2 + x_1x_2 + x_1 + x_2 + 1) + x_1^2x_2^2 + x_1^2x_2 + x_1x_2^2 + x_1x_2 + x_1 + x_2 + 1. \end{aligned}$$

а при $x_1 = x_2 \neq 0$

$$\begin{aligned} x_3 &= x_1^2x_2^2 + x_1^2x_2 + x_1x_2^2 + x_1x_2 + x_1 + x_2 + 1, \\ y_3 &= (x_1 + x_2)(x_1^2x_2^2 + x_1^2x_2 + x_1x_2^2 + x_1x_2 + x_1 + x_2 + 1) + x_1^2x_2^2 + x_1^2x_2 + x_1x_2^2 + x_1x_2 + x_1 + x_2 + 1. \end{aligned}$$

В проективных координатах формулы сложения точек эллиптической кривой, заданной уравнением

$$E: Y^2 + XYZ = X^3 + a_2X^2Z^4 + a_6Z^6,$$

над полем характеристики $p = 2$ выглядят

как

где тройка координат (X_3, Y_3, Z_3) вычисляется последовательно

по правилу:

$$\begin{aligned} \lambda_1 &= X_1Z_2^2, & \lambda_2 &= X_2Z_1^2, \\ \lambda_3 &= \lambda_1 + \lambda_2, & \lambda_4 &= Y_1Z_2^2, \\ \lambda_5 &= X_2Z_1^2, & \lambda_6 &= \lambda_4 + \lambda_5, \\ \lambda_7 &= Z_1\lambda_3, & \lambda_8 &= \lambda_6X_2 + \lambda_7Y_2, \\ Z_3 &= \lambda_7Z_2, & \lambda_9 &= \lambda_6 + Z_3, \\ X_3 &= a_2Z_3^2 + \lambda_6\lambda_9 + \lambda_3^4, & Y_3 &= \lambda_9X_3 + \lambda_8\lambda_7^2. \end{aligned}$$

Координаты удвоенной точки

~~определяются по~~
~~правилу:~~

Сжатие точек эллиптической кривой над полем [5] характеристики 2. Дана точка $P(x, y)$ на эллиптической кривой. Если $y = 0$, то можно положить $b = 0$. В противном случае вычисляют $z = y / x$ и присваивают переменной b самый младший двоичный разряд числа z . Для восстановления y по данной паре (x, b) в случае $x \neq 0$ вычисляют

$$\alpha x + \frac{\alpha}{x}$$

и обозначают через β одно из решений уравнения $z^2 + z = \alpha$.

Если наименьший двоичный разряд числа β совпадает с b , то $y = x\beta$. В противном случае $y = x(\beta - 1)$.

Алгоритм цифровой подписи EC-DSA. Алгоритм DSA можно обобщить на произвольную конечную абелеву группу A , в которой сложно решается задача дискретного логарифмирования, порождающий элемент G имеет простой порядок $Q > 2^{160}$ и существует открытая функция $F: A \rightarrow \mathbb{Z}/Q\mathbb{Z}$. В последнее время широкое распространение получили криптосистемы на эллиптических кривых [15]. Аналог алгоритма DSA, обобщенный для работы с эллиптическими группами точек, носит название EC-DSA.

Различия в средах оперирования DSA и EC-DSA сведены в табл. 16.

Таблица 16.

Параметры DSA и EC-DSA.

Параметр	DSA	EC-DSA
A	$\langle G \rangle \subset \mathbb{F}_p^*$	$\langle P \rangle \subset E(\mathbb{F}_p)$
G	$G \in \mathbb{F}_p^*$	$P \in E(\mathbb{F}_p)$
Y	G^x	$[x]P$
F	$(\text{mod } Q)$	x -координата по модулю Q

В обобщенной версии DSA каждый пользователь тоже генерирует секретный подписывающий ключ x и вычисляет

открытый по формуле: $Y = [x]P$. Подпись получается в результате выполнения следующего алгоритма:

- 1) Вычисляется хеш-значение $H = H(M)$.
- 2) Выбирается эфемерный ключ k , $0 < k < Q$.
- 3) Определяется $R = F([k]P)$ и $S = (H + xR)/k \pmod{Q}$.

Подписью сообщения M служит пара (R, S) .

Для проверки подписи (R, S) на сообщении M осуществляют следующую процедуру:

- вычисляют хеш-значение $H = H(M)$;
- определяют $A = H/S \pmod{Q}$ и $B = R/S \pmod{Q}$;
- находят $V = F([A]P + [B]Y)$, где Y – открытый ключ подписавшего сообщение;
- подпись считается корректной, если $V = R$.

Пример. Возьмем за основу эллиптическую кривую над полем F_{199} вида: $E: Y^2 = X^3 + X + 3$.

Число ее элементов равно $Q = 197$, т. е. является простым. Поэтому соответствующая группа – циклическая, а координаты ее образующей $P = (1, 76)$. Выберем $x = 29$ в качестве закрытого ключа. Тогда соответствующий открытый ключ будет равен:

$$Y = [x]P = [29](1, 76) = (113, 191).$$

Предположим, что владелец закрытого ключа намерен подписать сообщение с хеш-значением $H(M) = 68$. Тогда ему нужно сгенерировать эфемерный ключ, который выберем $k = 153$, и вычислить $R = x\text{-коорд.}([k]P) = x\text{-коорд.}([153](1, 76)) = x\text{-коорд.}((185, 35)) = 185$.

Далее он находит

$S = (H(M) + x \cdot R)/k \pmod{Q} = (68 + 29 \cdot 185)/153 \pmod{197} = 78$. Подпись, которую он посылает вместе с сообщением, – это пара $(R, S) = (185, 78)$.

Чтобы проверить подпись, вычисляем:

$$A = H(M)/S \pmod{Q} = 68/78 \pmod{197} = 112,$$

$$B = R/S \pmod{Q} = 185/78 \pmod{197} = 15.$$

После этого определяем

$$Z = [A]P + [B]Y = [112](1, 76) + [15](113, 191) = (111, 60) + (122, 140) = (185, 35).$$

Подпись оказалась верной, поскольку $R = 185 = x\text{-коорд.}(Z)$.

Квантовая криптография. Идея квантовой криптографии основывается, главным образом, на физике фотонов. Фотон во время своего движения производит колебания (рис. 40). Все четыре фотона летят в одном направлении, но в каждом случае угол колебаний различен. Угол колебаний называется поляризацией фотона, и лампочкой накаливания создаются фотоны всех поляризаций, что означает, что у части фотонов колебания будут происходить вверх-вниз, у части фотонов – влево-вправо, а у остальных колебания будут происходить при любых углах между этими направлениями. Для простоты предположим, что фотоны обладают только четырьмя возможными поляризациями, которые обозначим \leftrightarrow \updownarrow \nearrow \nwarrow .

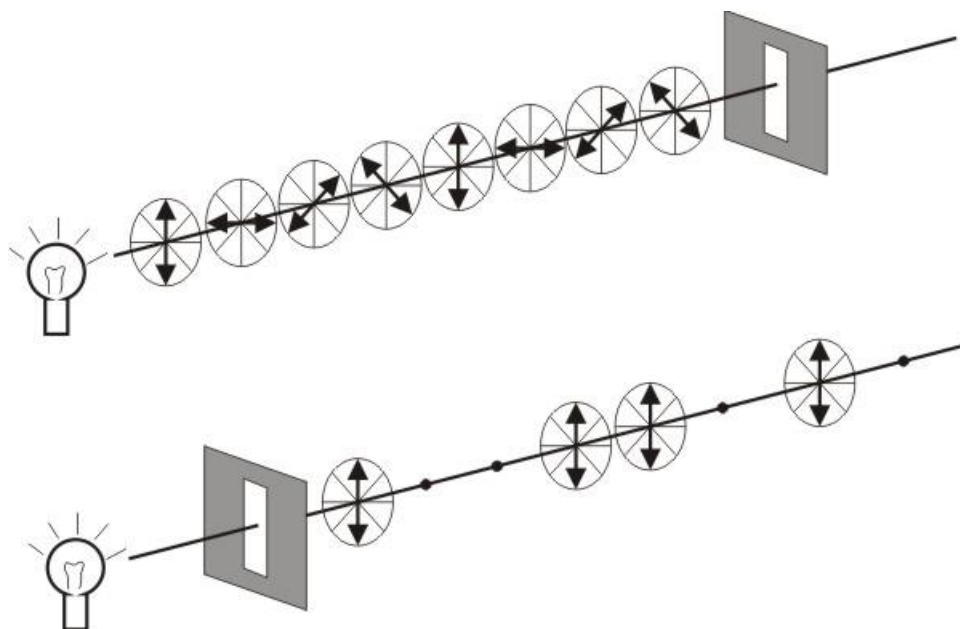


Рис.40. Прохождение поляризованных квантов через вертикальную решетку.

Если на пути фотонов установить фильтр, называющийся поляризационным, то выходящий пучок света будет состоять из фотонов, которые колеблются в одном определенном направлении; другими словами, все фотоны будут иметь одну и ту же поляризацию. Любой фотон, поляризованный в том же направлении, что и поляризация поляризационного фильтра, заведомо пройдет через него без изменений, а фотоны,

поляризованные в направлении, перпендикулярном к фильтру, будут задержаны.

Из диагонально поляризованных фотонов, попадающих на поляризационный фильтр с вертикальной поляризацией, половина из них будет задержана, а половина пройдет через фильтр, причем те, которые пройдут, приобретут вертикальную поляризацию. На верхней части рисунка показаны восемь фотонов, попадающих на поляризационный фильтр с вертикальной поляризацией, а на нижней части показано, что через фильтр благополучно прошли только четыре из восьми фотонов. Прошли все вертикально поляризованные фотоны и половина диагонально поляризованных фотонов, а все горизонтально поляризованные фотоны задержаны.

Именно такая способность задерживать определенные фотоны и объясняет, каким образом действуют поляроидные солнцезащитные очки.

Идея использования поляризованных фотонов для организации криптографически защищенного канала связи принадлежит американским ученым С. Беннету и Ж. Брассарду. Они задались вопросом, что произойдет, если зашифрованное сообщение будет представлено, а затем передано с помощью поляризованных фотонов. Сущность метода состоит в том, что противник не сможет безошибочно прочесть зашифрованное сообщение, а раз не сможет прочесть зашифрованное сообщение, то не сможет и дешифровать его.

Рассмотрим схему организации такого канала. Допустим, необходимо отправить зашифрованное сообщение, которое состоит из последовательности 1 и 0. Вместо этих 1 и 0 посылаются фотоны с определенными поляризациями. Есть две возможные схемы, с помощью которых можно связать поляризации фотонов с 1 или 0.

В первой схеме, называемой *ортогональной* или *+-схемой*, для представления 1 посылаем \uparrow , а для представления 0 – \leftrightarrow . Во второй схеме, называемой *диагональной* или \times -схемой, для представления 1 посылаем \nearrow , а для представления 0 – \searrow .

При отправке сообщения, представленного в двоичном виде, будем постоянно переключаться с одной схемы на другую непредсказуемым образом. Так что двоичное сообщение 1101101001 может быть передано следующим образом:

Сообщение	1	1	0	1	1	0	1	0	0	1
Схема	+	×	+	×	×	×	+	+	×	×
Передача	↑	↗	↔	↗	↗	↘	↑	↔	↘	↗

Мы передаем первую единицу с использованием +-схемы, а вторую 1 – с использованием × -схемы. Так что в обоих случаях передается единица, но всякий раз она представляется различным образом поляризованными фотонами.

Если противник захочет перехватить это сообщение, то ему потребуется определить поляризацию каждого фотона. Чтобы измерить поляризацию каждого фотона, противник должен решить, каким образом сориентировать свой поляризационный фильтр по мере прихода каждого фотона. Он не может знать наверняка, какой схемой воспользовались для каждого из фотонов, поэтому наугад выбирает ориентацию поляризационного фильтра, которая окажется неверной в половине случаев. А следовательно, она не сможет точно определить содержимое передаваемого сообщения.

Для реализации симметричной схемы шифрования достаточно использовать в качестве секретного ключа последовательность поляризации (вторая строка таблицы – +×+××+××). Но здесь столкнемся с проблемой распределения ключей. А при использовании короткопериодичной гаммы поляризации проявятся еще и все слабости, присущие шифрам гаммирования.

На основе этого получен вывод, что для квантовой криптографии требуется три подготовительных этапа. Хотя эти этапы не включают в себя отправку зашифрованного сообщения, с их помощью осуществляется безопасный обмен ключом, с помощью которого позднее можно будет зашифровать сообщение.

Этап 1. Абонент А начинает передавать абоненту В случайную последовательность из 1 и 0 (биты), используя для этого случайным образом выбираемые ортогональные (горизонтальная и вертикальная поляризации) и диагональные поляризационные схемы. На рис. 41 показана такая последовательность фотонов, движущихся к В.

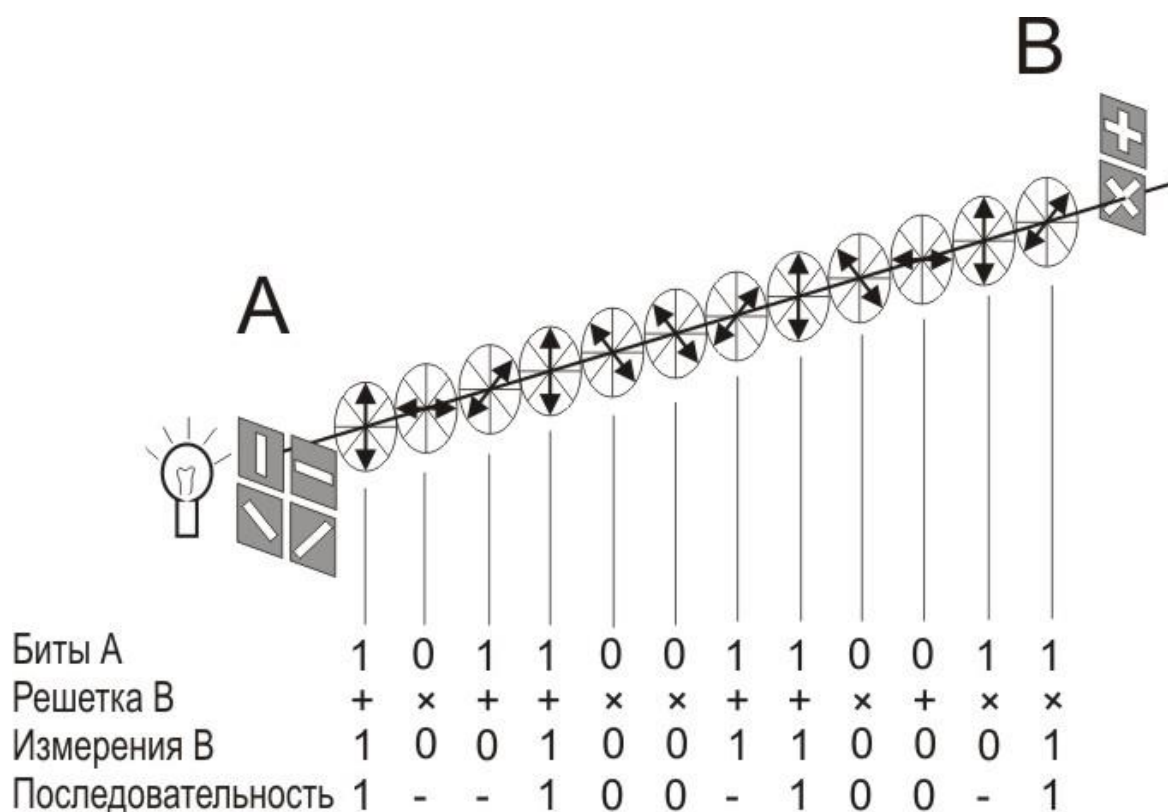


Рис.41. Квантовый канал обмена ключами.

Этап 2. Абонент В должен измерить поляризацию этих фотонов. Поскольку он не имеет представления, какой поляризационной схемой абонент А пользовался для каждого из фотонов, то в произвольном порядке выбирает +- и x-решетку. Иногда абонент В выбирает правильный детектор, иногда – нет. Если В воспользуется не той решеткой, то он вполне может неправильно распознать фотон абонента А. В табл. 17 указаны все возможные случаи.

К примеру, в верхней строке для посылки 1 абонент А использует ортогональную схему и поэтому передает \uparrow ; далее В, используя правильную решетку, определяет \uparrow и выписывает 1 в качестве первого бита последовательности. В следующей строке действия А те же самые, но В теперь использует неверную решетку, и поэтому он может определить \nearrow или \searrow , что означает, что либо он верно выпишет 1, либо неверно – 0.

Таблица 17.

Возможные ошибки према поляризованных квантов.

Схема А	Бит А	Фотон А	Решетка В	Ошибка	Фотон В	Бит В	Ошибка
Ортогональная	1	↑	+	Нет	↑	1	Нет
			×	Да	↗	1	Нет
	↘	0			Да		
	0	↔	+	Нет	↔	0	Нет
			×	Да	↗	1	Да
					↘	0	Нет
Диагональная	1	↗	+	Нет	↑	1	Нет
			↔	0	Да		
	×	Да	↗	1	Нет		
	0	↘	+	Нет	↑	1	Да
			↔	0	Нет		
×			Да	↘	0	Нет	

Этап 3. К этому моменту абонент А уже отправил последовательность 1 и 0, а В уже определил их; какие-то правильно, какие-то – нет. После этого абоненты А и В созваниваются по обычной незащищенной линии и А сообщает В, какую поляризационную схему он использовал для каждого фотона, но не как он поляризовала каждый из фотонов. Так, он может сказать, что первый фотон был послан с использованием ортогональной схемы, но не скажет, какой это был фотон: \uparrow или \leftrightarrow ;

В сообщает А, в каких случаях он угадал с правильной поляризационной схемой. В этих случаях он, несомненно, измерил правильную поляризацию и верно выписал 1 или 0. В конечном счете А и В игнорируют все те фотоны, для которых В пользовался неверной схемой, и используют только те из них, для которых он угадал с правильной схемой. В действительности они создали новую, более короткую последовательность битов, состоящих только из правильных измерений. Весь этот этап изображен в виде таблицы в нижней части рис. 41.

Благодаря этим трем этапам, абонентам А и В удалось образовать общую согласованную последовательность цифр: 11001001. Ключевым для этой последовательности является то, что она случайна, поскольку получена из исходной последовательности абонента А, которая сама была случайной. Более того, события, когда В использует правильный детектор, сами являются случайными. Поэтому данная согласованная последовательность может использоваться в качестве случайного ключа. И вот теперь-то можно начать процесс зашифрования.

Эта согласованная случайная последовательность может использоваться в качестве ключа для шифра одноразового шифрблокнота.

Во время передачи абонентом А поляризованных фотонов противник пытается измерить их, но он не знает, использовать ли

+/- или \times -решетку. В половине случаев выбор решетки будет неверным. Это приведет к исчезновению фотонов из канала, что однозначно при выполнении этапов 2 и 3 приводит к несовпадению данных и указывает на присутствие противника в канале.

Итак, квантовая криптография является системой, которая обеспечивает секретность связи, не позволяя противнику безошибочно прочесть сообщение между абонентами. Более того, если противник попытается осуществить перехват, то абоненты смогут обнаружить его присутствие. Тем самым квантовая криптография дает абонентам возможность обмениваться информацией и согласовать одноразовый шифрблокнот совершенно конфиденциальным образом, после чего они смогут использовать его в качестве ключа для зашифрования сообщения. Этот способ состоит из пяти основных этапов:

1) А посылает В последовательность фотонов, а абонент В измеряет их.

2) А сообщает В, в каких случаях он измерил их правильно. (Хотя А и сообщает В, когда он выполнил правильное измерение, он не сообщает ему, каков должен быть правильный результат, так что, даже если противник и просматривает канал, это не представляет ровным счетом никакой опасности.)

3) Чтобы создать пару идентичных одноразовых шифрблокнотов, А и В отбрасывают те измерения, которые В

выполнил неверно, и используют те из них, которые он выполнил правильно.

4) А и В проверяют неприкосновенность своих одноразовых шифрблокнотов путем сличения нескольких цифр.

5) Если процедура проверки показала удовлетворительные результаты, они могут использовать одноразовый шифрблокнот для зашифрования сообщения. Если же проверка выявила ошибки, то им становится известно, что противник осуществил перехват фотонов, и им следует начать все заново.

Контрольные вопросы

1. Перечислите типы атак на симметричные криптоалгоритмы.
2. В чём разница между дифференциальным и линейным криптоанализом?
3. В чем суть атаки на основе парадокса о днях рождения?
4. Перечислите отличия атаки факторизацией от атаки дискретным логарифмированием.
5. В чем заключается преимущество использования асимметричных криптосистем на основе эллиптических кривых?
6. В чем основная идея квантовой криптографии?

ЛИТЕРАТУРА

1. Основы криптографии: учебное пособие / А.П. Алферов, А.Ю. Зубов, А.С. Кузьмин, А.В. Черемушкин. – М.: Гелиос-АРВ, 2001. – 480 с.
2. Бабенко, Л. К. Современные алгоритмы блочного шифрования и методы их анализа: учебное пособие / Л. К. Бабенко, Е. А. Ищукова. – М.: Гелиос-АРВ, 2006. – 376 с.
3. Элементарное введение в эллиптическую криптографию: Алгебраические и алгоритмические основы / А.А. Болотов, С.Б. Гашков, А.Б. Фролов, А.А. Часовских. – М.: КомКнига, 2006. – 328 с.
4. Элементарное введение в эллиптическую криптографию: Протоколы криптографии на эллиптических кривых / А.А. Болотов, С.Б. Гашков, А.Б. Фролов, А.А. Часовских. – М.: КомКнига, 2006. – 280 с.
5. Василенко, О.Н. Теоретико-числовые алгоритмы в криптографии/ О.Н. Василенко. – М.: МЦНМО, 2003. – 328 с.
6. Глухов, М. М. Алгебра. Т.1 / М. М. Глухов, В. П. Елизаров, А. А. Нечаев. – М.: Гелиос-АРВ, 2003. – 336 с.; ил.
7. Глухов, М. М. Алгебра. Т.2 / М. М. Глухов, В. П. Елизаров, А. А. Нечаев. – М.: Гелиос-АРВ, 2003. – 416 с.; ил.
8. Запечников, С.В. Криптографические протоколы и их применение в финансовой и коммерческой деятельности: учебное пособие / С.В. Запечников. – М.: Горячая линия Телеком, 2007. – 320 с.
9. Кнут, Д. Искусство программирования. Т.1–3: [пе. с англ.] /Д. Кнут. – М.: Вильямс, 2001
10. Коблиц, Н. Курс теории чисел и криптографии: [пе. с англ.] /Н. Коблиц. –М.: ТВП, 2001. – 254 с.
11. Мао, В. Современная криптография. Теория и практика / В. Мао. – М.: Вильямс, 2005. – 768 с.
12. Маховенко, Е.Б. Теоретико-числовые методы в криптографии: учебное пособие / Е.Б. Маховенко. – М.: Гелиос-АРВ, 2006. – 320 с.
13. Рябко, Б.Я. Криптографические методы защиты информации: учебное пособие/ Б.Я. Рябко, А.Н. Фионов. – М.: Горячая линия Телеком, 2005. – 229 с.

14. Сингх, С. Книга шифров: тайная история шифров и их расшифровки / С. Сингх; пер. с англ. А. Галыгина. – М.: АСТ: Астрель, 2007. – 447 с.
15. Сمارт, Н. Криптография: [пе. с англ.] / Н. Смарт. – М.: Техносфера, 2005. – 528 с.
16. Столлингс В. Криптография и защита сетей. Принципы и практика: [пе. с англ.] 2-е изд / В. Столингс. – М.: Вильямс, 2001. – 672 с.
17. Тилборгван, Х. К. А. Основы криптологии. Профессиональное руководство и интерактивный учебник / Х. К. А. Тилборгван. – М.: Мир, 2006. – 471 с.
18. Фергюсон, Н. Практическая криптография / Н. Фергюсон, Б. Шнайер. – М.: Диалектика, 2005. – 424 с.
19. Фомичев, В. М. Дискретная математика и криптология / В. М. Фомичев. – М.: ДИАЛОГ-МИФИ, 2003. – 400 с.
20. Харин, Ю.С. Математические и компьютерные основы криптологии / Ю.С. Харин, В.И. Берник, Г.В. Матвеев, С.В. Агиевич. – Мн.: Новое знание, 2003. – 382 с.
21. Черемушкин, А.В. Лекции по арифметическим алгоритмам в криптографии / А.В. Черемушкин. – М.: МЦНМО, 2002. – 104 с.
22. Шнайер, Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си: [пе. с англ.] / Б. Шнайер. – М.: ТРИУМФ, 2003. – 816 с.

Приложение А – перестановки DES

Начальная и завершающая перестановки DES

Начальная перестановка (IP)

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

Перестановка, обратная начальной (IP-1)

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

Расширяющая перестановка DES:

Перестановка с расширением (E)

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

Сжимающая перестановка DES:

Вторая перестановка с выбором (PC-2)

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	54	46	42	50	36	29	32

Подстановка в S-блоке DES:

		S-матрицы															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S1	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S2	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S3	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S4	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S5	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S6	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S7	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S8	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Перестановка в P-блоке DES:

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

Преобразования ключа DES.

Начальная перестановка 64-битового ключа (каждый восьмой бит – бит четности):

Первая перестановка с выбором (PC-1)

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

Циклические сдвиги влево:

Таблица сдвигов влево (вверху – раунд, внизу – число сдвигов)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	1
1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Приложение Б – IDEA

Зашифрование и расшифрование в IDEA.

Шифрование		
Стадия	Обозначение	Эквивалент
Раунд 1	$Z_1 Z_2 Z_3 Z_4 Z_5 Z_6$	$Z[1..96]$
Раунд 2	$Z_7 Z_8 Z_9 Z_{10} Z_{11} Z_{12}$	$Z[97..128, 26..89]$
Раунд 3	$Z_1 Z_2 Z_3 Z_4 Z_5 Z_6$	$Z[90..128, 1..25, 51..82]$
Раунд 4	$Z_7 Z_8 Z_9 Z_{10} Z_{11} Z_{12}$	$Z[83..128, 1..50]$
Раунд 5	$Z_1 Z_2 Z_3 Z_4 Z_5 Z_6$	$Z[76..128, 1..43]$
Раунд 6	$Z_7 Z_8 Z_9 Z_{10} Z_{11} Z_{12}$	$Z[44..75, 101..128, 1..36]$
Раунд 7	$Z_1 Z_2 Z_3 Z_4 Z_5 Z_6$	$Z[37..100, 126..128, 1..29]$
Раунд 8	$Z_7 Z_8 Z_9 Z_{10} Z_{11} Z_{12}$	$Z[30..125]$
Преобразование	$Z_{49} Z_{50} Z_{51} Z_{52}$	$Z[23..86]$
Расшифрование		
Стадия	Обозначение	Эквивалент
Раунд 1	$U_1 U_2 U_3 U_4 U_5 U_6$	$Z_{49}^{-1} - Z_{50} - Z_{51} Z_{52}^{-1} Z_{47} Z_{48}$
Раунд 2	$U_7 U_8 U_9 U_{10} U_{11} U_{12}$	$Z_{43}^{-1} - Z_{45} - Z_{44} Z_{46}^{-1} Z_{41} Z_{42}$
Раунд 3	$U_{13} U_{14} U_{15} U_{16} U_{17} U_{18}$	$Z_{37}^{-1} - Z_{39} - Z_{38} Z_{40}^{-1} Z_{35} Z_{36}$
Раунд 4	$U_{19} U_{20} U_{21} U_{22} U_{23} U_{24}$	$Z_{31}^{-1} - Z_{33} - Z_{32} Z_{34}^{-1} Z_{29} Z_{30}$
Раунд 5	$U_{25} U_{26} U_{27} U_{28} U_{29} U_{30}$	$Z_{25}^{-1} - Z_{27} - Z_{26} Z_{28}^{-1} Z_{23} Z_{24}$
Раунд 6	$U_{31} U_{32} U_{33} U_{34} U_{35} U_{36}$	$Z_{19}^{-1} - Z_{21} - Z_{20} Z_{22}^{-1} Z_{17} Z_{18}$
Раунд 7	$U_{37} U_{38} U_{39} U_{40} U_{41} U_{42}$	$Z_{13}^{-1} - Z_{15} - Z_{14} Z_{16}^{-1} Z_{11} Z_{12}$
Раунд 8	$U_{43} U_{44} U_{45} U_{46} U_{47} U_{48}$	$Z_7^{-1} - Z_9 - Z_8 Z_{10}^{-1} Z_5 Z_6$
Преобразование	$U_{49} U_{50} U_{51} U_{52}$	$Z_1^{-1} - Z_2 - Z_3 Z_4^{-1}$

Учебное издание

Еременко Владимир Тарасович

Шпичак Сергей Александрович

Рязанцев Павел Николаевич

КРИПТОГРАФИЧЕСКИЕ МЕТОДЫ ЗАЩИТЫ ИНФОРМАЦИИ

Учебное пособие

Редактор Т. Д. Васильева
Технический редактор

Федеральное государственное бюджетное образовательное
Учреждение высшего профессионального образования
«Государственный университет – учебно-научно-
производственный комплекс»

Подписано к печати	Формат 60х90 1/16
Усл. печ. л.	Тираж экз.
Заказ №	

Отпечатано с готового оригинал-макета

кафедра «Электроника, вычислительная техника и
информационная безопасность»

Телефон: 8-9066646161, 8-9065701666

Учебное издание

Еременко Владимир Тарасович

Шпичак Сергей Александрович

Рязанцев Павел Николаевич

КРИПТОГРАФИЧЕСКИЕ МЕТОДЫ ЗАЩИТЫ ИНФОРМАЦИИ

Учебное пособие