Министерство образования и науки Российской Федерации

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Приоритетный национальный проект «Образование»
Национальный исследовательский университет

Ю.В. ВЕТРОВ С.Б. МАКАРОВ

Криптографические методы защиты информации в телекоммуникационных системах

Рекомендовано Учебно-методическим объединением по университетскому политехническому образованию в качестве учебного пособия для студентов высших учебных заведений, обучающихся по направлению подготовки магистров Техническая физика

Санкт-Петербург
Издательство политехнического университета
2011

УДК 621.391.7:004.056.55(075.8)
ББК 32.811.4я73
В 393

Рецензенты:

М.А Сиверс, д.т.н., проф., зав. каф. «Радиопередающие устройства и средства подвижной связи» Санкт-Петербургского государственного университета им. проф. М.А. Бонч-Бруевича.

А.Ф. Крячко, д.т.н., проф., начальник каф. «Телеметрические системы» Военно-космической академии им. А.Ф. Можайского

Ветров Ю.В. **Криптографические методы защиты информации в телекоммуникационных системах:** учеб. пособие / Ю.В. Ветров, С.Б. Макаров. — СПб.: Изд-во Политехн. ун-та, 2011. — 174 с.

ISBN

Предлагаемое учебное пособие посвящено изложению криптографическим методов защиты информации, применяемых в телекоммуникационных системах связи.

В пособии рассматриваются методы шифрования сообщений, в том числе блочные шифры, поточные шифры и др. Даются элементарные сведения о вычислениях в конечных полях. Анализируются классические криптосистемы с секретным ключом и криптографические системы с открытым ключом. Значительное внимание уделено защите информации в телекоммуникаицоных системах стандартов GSM, UMTS и CDMA 2000. В учебном пособии приведено описание лабораторной работы «Криптографические методы защиты информации».

Учебное пособие рекомендовано Учебно-методическим объединением по университетскому политехническому образованию в качестве учебного пособия для студентов высших учебных заведений, обучающихся по направлению подготовки «Техническая физика». Оно также может быть полезно студентам, обучающимся по направлениям «Радиотехника» и «Инфокоммуникационные технологии и системы связи».

Печатается по решению редакционно-издательского совета Санкт-Петербургского государственного политехнического университета.

ISBN

Оглавление

Предисловие

Учебное пособие предназначено для студентов высших учебных заведений, обучающихся по направлениям подготовки «Техническая физика». Цель данного учебного пособия заключается в том, чтобы дать теоретические сведения, необходимые для понимания принципов использования криптографии, применяемой в телекоммуникационных системах передачи информации.

Во введении после краткого исторического экскурса даются определения основных терминов, используемых в криптографии.

В первом разделе рассматривается классификация криптографических систем, подробно рассматривается структура Фейстеля, лежащая в основе современных композиционных блочных шифров, поточные шифры, используемые в системах мобильной связи.

Во втором разделе рассматриваются математические основы криптографии: вычисления в конечных полях, в частности, в полях многочленов, методы решения диофантовых уравнений, принципы использования китайской теоремы об остатках, как числовом поле, так и в поле многочленов.

В третьем разделе рассматриваются системы с открытым ключом, используемые в системах электронной торговли, в банковских системах, функции хеширования и цифровой подписи.

В четвертом разделе рассматриваются механизмы обеспечения безопасности в телекоммуникационной системе передачи данных стандарта GSM. В частности, протоколы аутентификации сети, аутентификации пользователя, функции шифрования, алгоритмы имитационной защиты информации.

В пятом разделе рассматриваются механизмы обеспечения безопасности в системах мобильной связи третьего поколения UMTS и CDMA2000.

В шестом разделе приводится описание лабораторной работы «Криптографические методы защиты информации», в которой предлагается программа экспериментальных исследований, поддерживающих основные теоретические разделы математических основ криптографии, методы построения композиционных блочных шифров, поточных шифров, используемых в телекоммуникационных системах.

В пособии излагаются основы криптографии с позиции применения ее основных методов и алгоритмов в телекоммуникационных системах передачи информации. В пособии нет подробного анализа основных теоретических положений классической криптографии, но даются элементарные сведения из общей математической теории построения многочленов, позволяющие понять механизмы обеспечения безопасности в цифровых мобильных системах связи стандартов GSM, UMTS и CDMA2000.

Авторы считают, что опыт, полученный ими в процессе многолетнего ведения соответствующих курсов лекций, лабораторного практикума, руководства дипломным проектированием помог им в процессе работы над учебным пособием. Профессиональное удовлетворение, как считают авторы, они смогут получить, если учебное пособие станет кладезем практических знаний в разработке современных защищенных телекоммуникационных систем передачи данных и связи.

— Ах, если б знать! — заплакал Морж. — Проблема так сложна!
Льюис Кэролл. Алиса в Зазеркалье

Введение

Проблема тайной передачи сообщений существует столько времени, сколько существует письменность, более того, первоначально письменность сама по себе была методом тайной передачи информации, поскольку была доступна лишь избранным. Для реализации тайной передачи сообщения от одного адресата к другому существует два направления: во-первых, можно попытаться скрыть сам факт передачи сообщения, например, методами тайнописи, во-вторых, можно так преобразовать сообщение, чтобы постороннему лицу была недоступна информация, заключенная в сообщении. Первым направлением занимается *стеганография*, вторым — *криптография*.

В учебном пособии будут рассматриваться криптографические методы защиты информации, передаваемой по телекоммуникационным системам. Для защиты информации используется, прежде всего, шифрование. При шифровании происходит преобразование данных в вид, недоступный для чтения без соответствующей информации (ключа шифрования). Задача состоит в том, чтобы обеспечить конфиденциальность, скрыв информацию от лиц, которым она не предназначена, даже если они имеют доступ к зашифрованным данным. Кроме этого, используются криптографические методы контроля целостности переданной информации и криптографические методы аутентификации абонента, позволяющие убедиться, что информация передана именно данному абоненту.

Роль шифров в истории развития человечества огромна. Шифры решали результаты сражений и приводили к смерти королей и королев. Характерной является история Марии Стюард [1]. Ей вменяли в вину организацию заговора с целью убийства королевы Елизаветы I, чтобы завладеть короной Англии. Нужно было доказать, что Мария была душой заговора, а посему заслуживает смерти. Однако ее положение не было безнадежно, поскольку она была осторожна, и вся ее переписка с заговорщиками была зашифрована. Шифр превратил ее слова в бессмысленный набор символов. Если содержание писем оставалось тайным, то письма не могли использоваться как свидетельство против нее. Однако все это было бы так только при условии, что ее шифр не был раскрыт.

Дешифрирование — процесс, обратный шифрованию, т. е. преобразование шифрованных данных в открытый вид.

Хотя криптография и стеганография являются независимыми, но для обеспечения максимальной секретности, чтобы и зашифровать, и скрыть сообщение, можно пользоваться обеими. Впервые криптографический метод защиты информации и стеганография в телекоммуникационных системах стал применяться, по-видимому, во Второй мировой войне. Метод назывался микроточкой, с помощью которого фотографическим способом сжималась страница текста в точку диаметром менее одного миллиметра. Эта микроточка прикреплялась поверх обычной точки в конце предложения в совершенно безобидном письме. Обнаружить первую микроточку удалось, получив предупреждение о том, где следует искать крошечный блик на поверхности письма, указывающий, что в этом месте прикреплен кусочек глянцевой фотопленки. В последствие оказалось возможным прочитать содержимое большинства перехваченных микроточек, за исключением тех случаев, когда были предпринято дополнительное шифрование сообщения перед сжатием.

В криптографии на первом этапе ее развития существовали два направления, известные как *перестановка* и *замена.* При перестанов-

ке буквы сообщения просто переставляются, образуя анаграмму. Для очень короткого сообщения, состоящего, например, из одного слова, такой способ весьма ненадежен, поскольку существует крайне ограниченное число возможных способов перестановки горстки букв. Однако по мере увеличения количества букв число возможных перестановок стремительно растет, и восстановить исходное сообщение становится невозможным, если не известен точный способ шифрования. Для обеспечения эффективности способ перестановки букв должен быть заранее оговорен отправителем сообщения и его получателем, но он должен храниться в секрете от противника. Один из способов перестановки букв был реализован в самом первом из известных шифровальных устройств — спартанской *скитале*. Упоминание об этом устройстве восходит к пятому веку до н. э. Скитала (или сцитала) представляла собой деревянный цилиндр (или конус), вокруг которого наматывалась полоска кожи или пергамента.

Рис. 1. Скитала

Отправитель писал сообщение по всей длине скиталы, а затем разматывал полоску, на которой после этого оставался бессмысленный набор букв. Сообщение оказывалось зашифрованным. Сообщение можно будет прочесть, только если намотать эту полоску вокруг другой скиталы такого же диаметра. Вестник брал кожаную полоску и обычно прятал сообщение, используя полоску как пояс, буквами внутрь, то есть кроме зашифровывания применял также и стеганографию. Чтобы получить исходное сообщение, адресат просто наматывал полоску кожи вокруг скиталы того же диаметра, что и скитала, которой пользовался отправитель. В 404 году до н. э. к спартанскому полководцу Лисандру привели вестника, окровавленного и

еле держащегося на ногах, одного из пяти оставшихся в живых после, крайне опасного путешествия из Персии. Вестник передал свой пояс Лисандру, который намотал его вокруг своей скиталы и прочитал, что Фарнабаз собирается напасть на него. Благодаря скитале Лисандр успел подготовиться к нападению и отбил его.

Альтернативой перестановке является замена. Одно из первых описаний шифрования с помощью замены дается в «Кама-сутре», тексте, написанном в четвертом веке н. э. священником-брамином Ватсьяной и основанном на манускриптах, относящихся к четвертому веку до н. э. Согласно «Кама-сутре», женщины должны овладеть 64 искусствами, такими, как приготовление пищи и напитков, искусство одевания, массаж, приготовление ароматов. В этот список также входят менее очевидные искусства: колдовство, игра в шахматы, переплетное дело и плотницкое дело. Под номером 45 в списке стоит искусство тайнописи, предназначенное для того, чтобы помочь женщинам скрыть подробности своих любовных связей. Один из рекомендуемых способов заключается в том, чтобы расположить попарно буквы алфавита случайным образом, а затем заменять каждую букву в исходном сообщении ее парной. Если применить этот принцип к латинскому алфавиту, то мы можем расположить буквы попарно следующим образом:

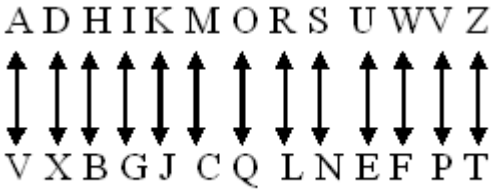

Рис. 2. Шифр замены

Тогда вместо текста *meet at midnight* (встретимся в полночь) отправитель напишет *CUUZ VZ CGXSGIBZ***.** Такой вид тайнописи называется шифром замены, поскольку каждая буква в исходном тексте заменяется другой буквой, так что этот шифр диаметрально противоположен шифру перестановки. При перестановке каждая буква остается сама

собой, но меняет свое местоположение, в то время как при замене каждая буква меняется на другую букву, но остается на своем месте.

Первое документально подтвержденное использование шифра замены в военных целях появилось в «Галльских войнах» Юлия Цезаря. Каждая букву в послании заменяется буквой, стоящей в алфавите на три позиции дальше. Наиболее часто пользуются терминами *алфавит открытого текста,* то есть алфавит, используемый для создания исходного, незашифрованного сообщения, и *шифралфавит*, буквы которого подставляются вместо букв алфавита открытого текста. Если алфавит открытого текста расположить над шифралфавитом, то станет ясно, что шифралфавит сдвинут на три позиции, и поэтому такой вид замены часто называется *шифром сдвига Цезаря* или просто шифром Цезаря. Хотя Светоний упоминает только о шифре Цезаря со сдвигом на три позиции, ясно, что, осуществляя сдвиг на 1..25 позиций, можно создать для латинского алфавита 25 различных шифров. Если же мы не будем ограничиваться сдвигом алфавита, а будем рассматривать шифралфавит как любую возможную перестановку букв алфавита открытого текста, то мы сможем создать гораздо большее количество различных шифров.

При этом шифр Цезаря является частным случаем моноалфавитной подстановки, а ключевая матрица является матрицей сдвига. Шифр Цезаря достаточно легкий. Если известен открытый текст и соответствующий ему зашифрованный текст, то ключ вычисляется непосредственно как разность номеров соответствующих букв открытого текста и зашифрованного текста. Если известны только зашифрованные тексты, то ключ вычисляется с помощью частотного анализа. Частотный анализ основан на том, что буквы алфавита появляются с различной частотой. Сравнивая частоту появления букв зашифрованного текста и частоту появления букв исходного алфавита, можно вычислить матрицу подстановки.

Изобретение многоалфавитной замены сделало процедуру шифрования устойчивой к вскрытию. Примером многоалфавитной

замены является *шифр Блеза Виженера*, дипломата 16 века, который развивал и совершенствовал криптографические системы. Этот шифр основан на таблице Виженера, представленной на рис. 3.

Ключ	**Исходный алфавит**
	абвгдежзиклмнопрстуфхцчшщыъэюя_
а	абвгдежзиклмнопрстуфхцчшщыъэюя_
б	_абвгдежзиклмнопрстуфхцчшщыъэюя
в	я_абвгдежзиклмнопрстуфхцчшщыъэю
г	юя_абвгдежзиклмнопрстуфхцчшщыъэ
..	----------
я	вгдежзиклмнопрстуфхцчшщыъэюя_аб
_	бвгдежзиклмнопрстуфхцчшщыъэюя_а

Рис. 3. Таблица Виженера

В этой таблице первая строка представляет собой исходный алфавит, дополненный символом пробела. Левый столбец таблицы представляет собой набор символов ключа. Вторая строка соответствует букве «**а**» ключа и нулевому сдвигу исходного алфавита, по существу, повторяя первую строку таблицы. Третья строка соответствует букве «**б**» ключа и циклическому сдвигу на один символ второй строки таблицы. Четвертая строка соответствует букве «**в**» ключа и циклическому сдвигу на один символ третьей строки таблицы. Построение строк продолжается до букв «**я**» и пробел «**_**» ключа.

При шифровании сообщения его выписывают в строку, а под каждой буквой открытого текста записывают буквы ключа. Если ключ оказался короче сообщения, то его циклически повторяют. Зашифрованный текст получают, находя символ открытого текста в первой строке таблицы и заменяя его соответствующим символом, находящимся на этом же месте, из строки, соответствующей букве ключа в левом столбце таблицы Виженера. Например, применяя, ключ АГАВА, к сообщению ПРИЕЗЖАЮ ШЕСТОГО, получаем следующее зашифрованное сообщение (рис. 4):

ПРИЕЗЖ А Ю _ Ш Е С Т О Г О открытый текст
АГАВАА Г А В А А Г А В А А ключ
ПНИГЗЖ ЮЮ ЮШ Е О Т М ГО зашифрованный текст

Рис. 4. Зашифрованное с помощью таблицы Виженера сообщение

В компьютере операция шифрования соответствует сложению ASCII символов сообщения и ключа по некоторому модулю. Для шифрования и дешифрирования требуется ключ. Процесс передачи шифрованной информации иллюстрируется рис. 5.

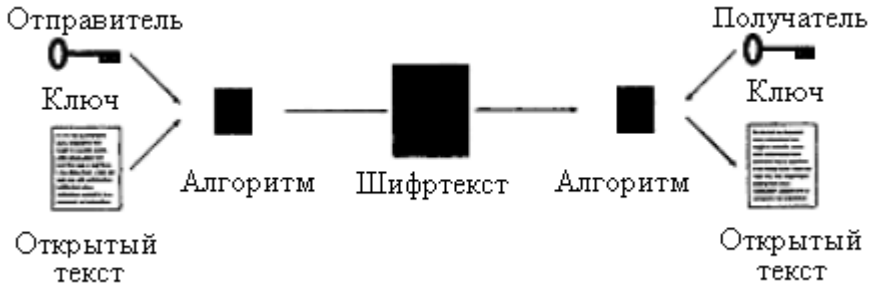

Рис. 5. Процесс передачи информации

Простейшей и, в то же время, наиболее надежной из всех систем шифрования является так называемая *схема однократного использования*. Принцип работы такой системы заключается в следующем. Формируется m-разрядная случайная двоичная последовательность, которая является ключом шифра, известного отправителю и получателю сообщения. Отправитель производит побитовое сложение по модулю 2 символов ключа и m-разрядной двоичной последовательности, соответствующей пересылаемому сообщению:

$$c_i = (p_i + k_i) \bmod 2,$$

где p_i, k_i, c_i — очередной i-й бит, соответственно, исходного сообщения, ключа и шифрованного сообщения. Процесс расшифровывания сводится к повторной генерации ключевой последовательности и наложения ее на шифрованные данные:

$$p_i = (c_i + k_i) \bmod 2.$$

Если ключ является фрагментом случайной двоичной последовательности с равномерным законом распределения, так, что длина ключа равна длине исходного сообщения, и ключ используется только один раз, то такой шифр является *абсолютно стойким* [5].

Абсолютная стойкость рассмотренной схемы оплачивается слишком большой ценой: такое устройство весьма дорогое и непрактичное. Одним из вариантов подобного устройства является шифровальный блокнот.

Методы криптографии используются не только для шифрования. Важной задачей также является установление подлинности сообщения или аутентификация (например, при подписании документов). Наиболее известной является цифровая подпись, которая связывает подписанный документ с владельцем определенного ключа, а цифровая дата связывает документ со временем его создания. Криптографические методы защиты информации в телекоммуникационных системах используются в финансовых операциях, таких, как например, электронная торговля. Термин «электронная торговля» подразумевает банковские операции, управление счетами и совершение покупок, а также некоторые другие действия, осуществляемые с помощью сети Интернет (например, заказ авиабилетов, бронирование мест в гостиницах, вызов такси, перевод денег с одного счета на другой и т. д.). Однако передача при расчетах по сети Интернет номера кредитной карточки вызывает необходимость защитить его от перехвата. Эта задачу также решается криптографическими средствами путем шифрования номера кредитной карточки (или другой ценной информации) для передачи по открытому телекоммуникационному каналу связи. Кроме того, криптографические методы позволяют защитить весь сеанс связи, и тогда вся пересылаемая по открытым каналам шифрованная информация будет недоступна для несанкционированного доступа. В этом случае Web-сервер интернет-магазина, получив шифрованную информацию, дешифрирует ее и принимает платеж, а пользователь может не опасаться, что номер кредитной

карточки (или другая ценная информация) будет использован не по назначению. Поскольку количество платежей, проводимых по сети интернет и беспроводным телекоммуникационным сетям, возрастает, защита информации от несанкционированного доступа становится одной из главных задач, которые возникают при построении современных сетей передачи данных.

Введем некоторые термины, необходимые для рассмотрения методов использования криптографии в телекоммуникационных системах.

Система криптографическая (криптосистема) **—** система обеспечения безопасности системы связи, использующая криптографические средства. В качестве подсистем может включать системы шифрования, аутентификации, имитационной защиты, цифровой подписи.

Система шифрования **—** система обеспечения конфиденциальности, предназначенная для защиты информации от ознакомления с ее содержанием лиц, не имеющих права доступа к ней, путем шифрования информации. Математическая модель системы включает способ кодирования исходной и выходной информации, шифр и ключевую систему.

Система ключевая **—** определяет порядок использования криптографической системы и включает системы установки и управления ключами.

Симметричные криптографические системы (симметричное шифрование, симметричные шифры, криптографические системы с секретным ключом) **—** способ шифрования, в котором для шифрования и дешифрирования применяется один и тот же ***секретный*** криптографический ключ, который никогда не передается по открытому каналу связи.

Криптографическая система с открытым ключом *(или* ***асимметричное шифрование)*** **—** система при которой используется пара ключей, причем ***открытый ключ*** передаётся по открытому (т. е.

незащищённому, доступному для наблюдения) каналу, а криптографическая стойкость обеспечивается наличием ***закрытого*** ключа.

Система аутентификации **—** криптографическая система, выполняющая функцию идентификации сторон в процессе информационного взаимодействия. Математическая модель системы включает протокол аутентификации.

Система имитационной защиты (обеспечения целостности) информации **—** криптографическая система, выполняющая функцию аутентификации содержания сообщения или документа и предназначенная для защиты от несанкционированного изменения информации

Система цифровой подписи **—** криптографическая система, выполняющая функцию аутентификации источника сообщения или документа.

Ни одна вещь не возникает беспричинно, но все возникает на каком-нибудь основаниии в силу необходимости.

Демокрит

1. Методы шифрования сообщений

Шифрование и дешифрование сообщений происходит на входе и выходе канала передачи данных (рис. 1.1). Несанкционированный пользователь анализирует процесс передачи информации по каналу связи и имеет возможность формировать свои собственные сообщения.

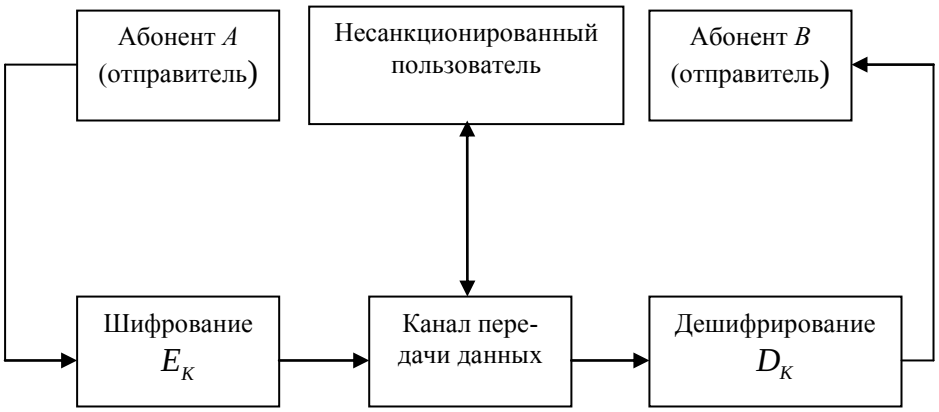

Рис. 1.1. Канал передачи данных

Процедуры шифрования и дешифрирования можно представить в следующем виде:

$$c = E_{k_e}(p);$$

$$p = D_{k_d}(c),$$

где p и c соответственно — открытый и зашифрованный тексты, k_e и k_d — ключи шифрования и дешифрирования; E_{k_e} и D_{k_d} — функции шифрования с ключом k_e и дешифрирования с ключом k_d соответст-

венно, причем для любого открытого текста справедливо соотношение

$$D_{k_d}(E_{k_e}(p)) = p.$$

На рис. 1.2 приведена классификация методов шифрования информации.

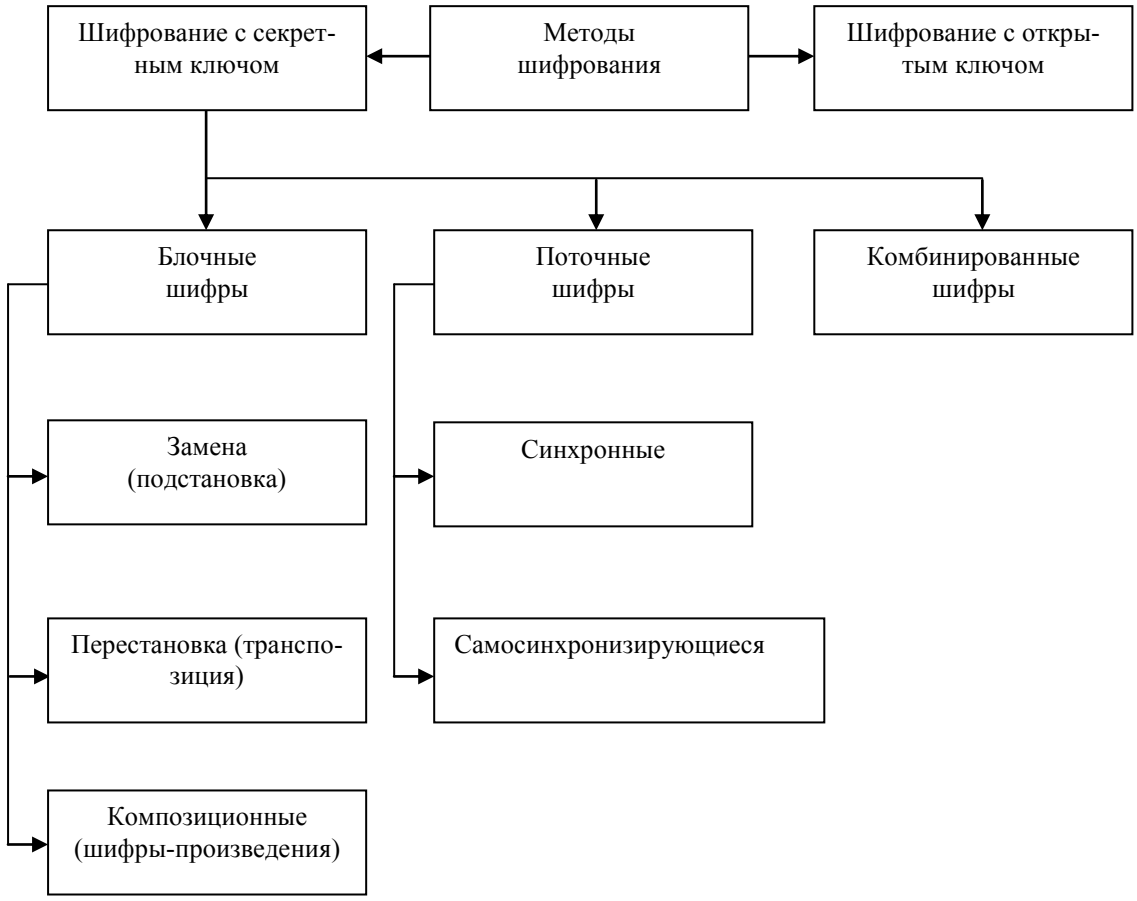

Рис. 1.2. Методы шифрования информации

Различают два типа алгоритмов шифрования: симметричные (с закрытым или секретным ключом) и асимметричные (с открытым ключом). В первом случае ключ шифрования совпадает с ключом дешифрирования. В асимметричных алгоритмах для шифрования и дешифрирования используются различные ключи, причем знание одного из них не дает практической возможности определить другой.

1.1. Блочные шифры

К блочным шифрам относятся шифр Цезаря, являющийся примером моноалфавитной подстановки, и шифр Виженера, являющийся примером многоалфавитной подстановки. Как правило, в виду низкой криптографической стойкости эти шифры в чистом виде не используются. Криптосистема с однократным ключом, обладающая абсолютной криптографической стойкостью, неудобна для практического применения. Поэтому блочные шифры строятся на основе многократного применения относительно простых криптографических преобразований, к которым относятся подстановки и перестановки. Примером такой структуры является конструкция Фейстеля.

1.1.1. Конструкция Фейстеля

Блочные шифры представляют собой семейство обратимых преобразований блоков (частей фиксированной длины) исходного текста. В процессе шифрования информация делится на порции величиной от одного до сотен бит. Блочные шифры наиболее распространены на практике. Под N-разрядным блоком будем понимать последовательность из нулей и единиц длины N:

$$x = (x_0, x_1, x_2, \ldots, x_{N-1}),$$

где x можно интерпретировать как вектор, двоичное представление целого числа или как многочлен. Под блочным шифром будем понимать отображение (см. разд. 2)

$$T : x \to y, \text{ где } y = (y_0, y_1, y_2, \ldots y_{N-1}),$$

представляющие собой последовательность из нулей и единиц длиной N.

Хотя блочные шифры можно при таком описании рассматривать как частный случай шифра подстановок на алфавитах большого объема $L = 2^N$, однако практически всегда их рассматривают отдельно.

Рассмотрим более подробно методы, заложенные в основу построения композиционных блочных шифров. Конструкция Фейстеля

лежит в основе шифров DES, ГОСТ 28147-89, Lucifer, FEAL, Blowfish и др.

При таком методе шифрования блок p_i открытого текста разбивается на две равные части — правую и левую. Представим шифруемый блок данных p_i в виде пары полублоков в два раза меньшего размера $p_i = (L, R)$. Очевидно, что при этом длина блока должна быть четной.

Конструкция Фейстеля представлена на рис. 1.3, где k_1, k_2, … — ключи шифрования, а f — криптографическая функция.

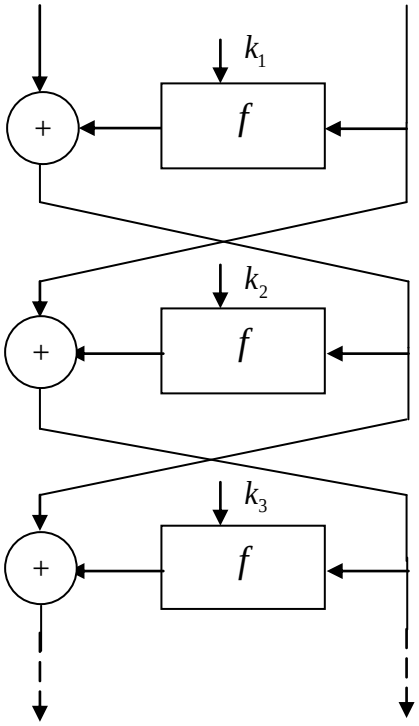

Рис. 1.3. Конструкция Фейстеля

Для того чтобы конструкция Фейстеля могла быть использована как для шифрования, так и для дешифрования функция f должна являться *инволюцией,* т. е. $f(f(x)) = f(x)$ для всех x. Например, функция $f(x) = x$ является инволюцией, функция $f(x) = x + c$ является инволюцией, где c произвольная константа, а суммирование осуществляется по модулю 2.

На каждом цикле одна из частей открытого текста подвергается преобразованию при помощи функции f и вспомогательного ключа k_i. Результат операции суммируется по модулю 2 с другой частью. Затем левая и правая части меняются местами. Преобразования на каждом цикле идентичны. Процедура дешифрования (рис. 1.4, *б*) аналогична процедуре шифрования (рис. 1.4, *а*), однако ключи k_i выбираются в обратном порядке.

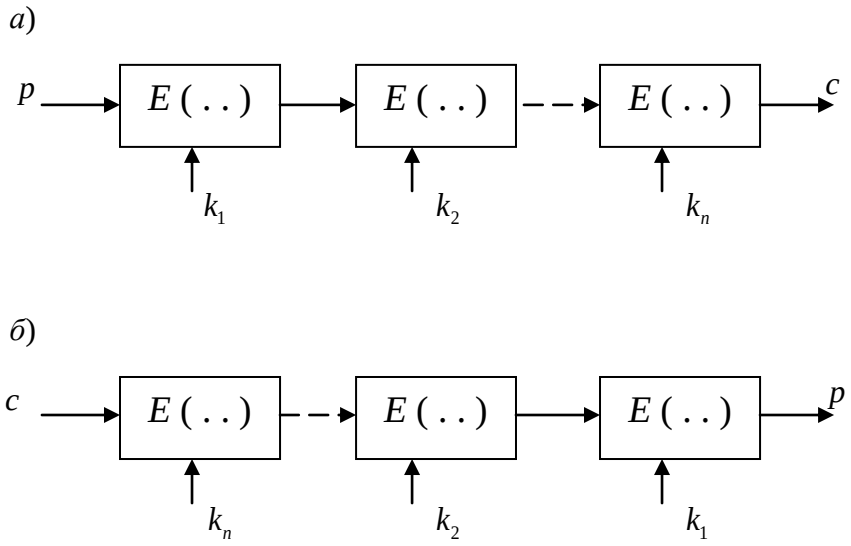

Рис. 1.4. Процессы шифрования и дешифрования

В результате преобразования блока открытого текста p рис. 1.4, *а*, получаем блок шифрованного текста

$$c = E_{k_n}\left(E_{k_{n-1}}\left(\ \ldots E_{k_2}\left(E_{k_1}(p)\right)\ldots\right)\right).$$

Исходный текст может быть восстановлен в результате преобразования (рис. 1.4, *б*):

$$p = E_{k_1}\left(E_{k_2}\left(\ \ldots E_{k_{n-1}}\left(E_{k_n}(p)\right)\ldots\right)\right).$$

Таким образом, завершается процесс шифрования и дешифрования сообщения.

1.1.2. Композиционные блочные шифры

Многократное применение операций подстановки и перестановки позволяет значительно повысить криптографическую стойкость и обеспечить два свойства, которые должны быть присущи стойким шифрам: *рассеивание и перемешивание.*

Рассеивание предполагает распространение влияния одного символа открытого текста, а также одного символа ключа на значительное количество символов шифрованного текста. Целью перемешивания является усложнение зависимости между ключом и шифрованным текстом. Наиболее известны шифр DES и шифр ГОСТ 28147-89, построенные на основе конструкции Фейстеля. Рассмотрим подробнее основные идеи, лежащие в основе этих двух шифров, которые фактически отличаются лишь численными параметрами.

Композиционный шифр определяется семейством преобразований F_i, где $i = 1, 2, ..., r$. Преобразование F_i называется раундом шифрования, причем каждому преобразованию соответствует свой ключ шифрования k_i, который называется *раундовым ключом*.

На рис. 1.5 более подробно рассмотрена структура раундовой функции шифрования. В блочных криптосистемах раундовые шифры строятся с использованием операций подстановки двоичных кодов небольшой разрядности (схемы, реализующие эту нелинейную операцию, называются *S*-блоками) и перестановки элементов двоичных кодов (*P*-блоки). От свойств кодов, записанных в *S*-блоках, в первую очередь зависит стойкость всей системы.

Важным достоинством многих составных шифров является их симметричность относительно операций шифрования и дешифрирования, которые по этой причине могут быть реализованы на одном устройстве. Переход от одного режима к другому обеспечивается заменой последовательности раундовых ключей на другие ключи.

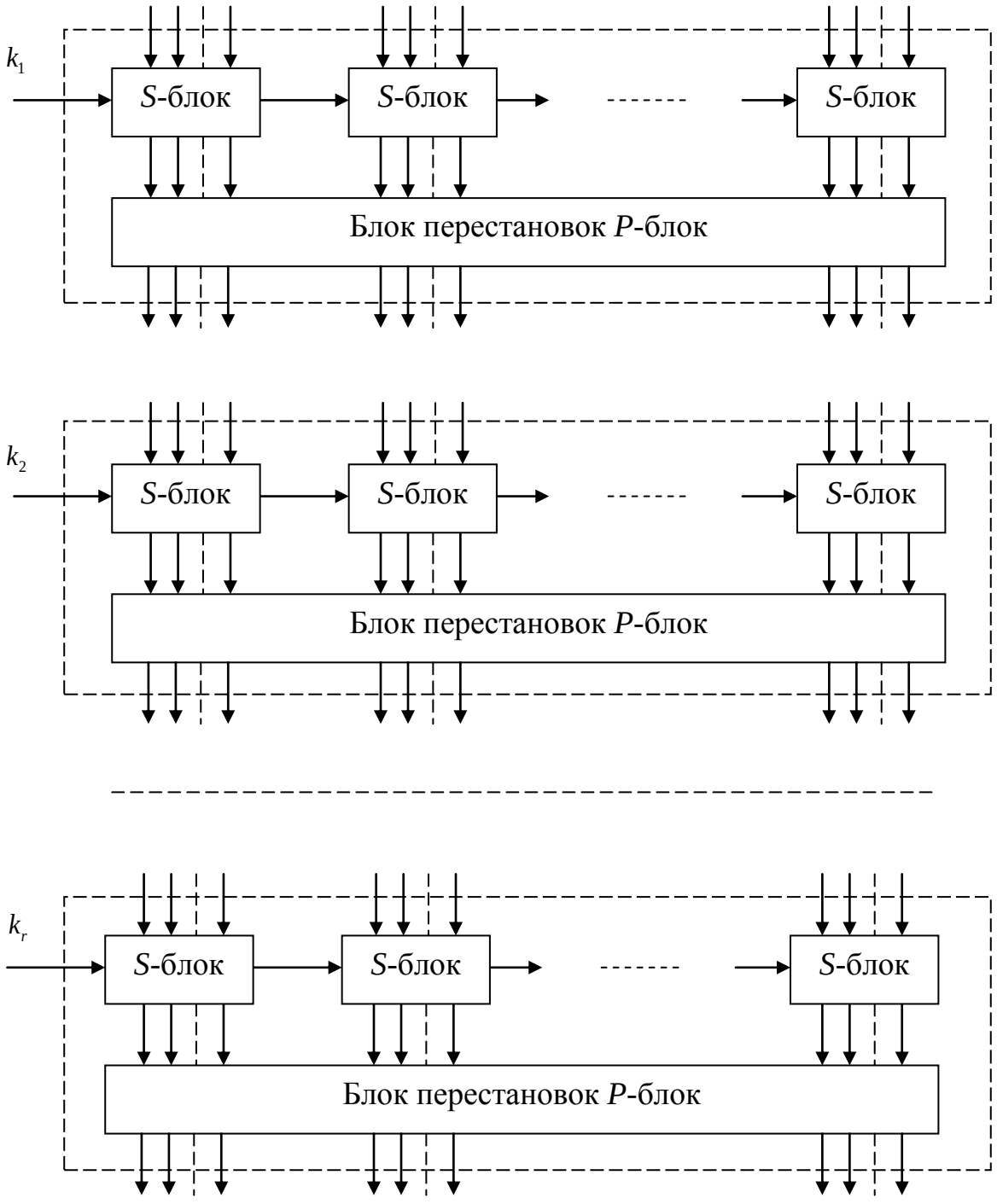

Рис. 1.5. Структура раундовой функции шифрования

Представим шифруемый блок данных p_i в виде пары полублоков в два раза меньшего размера, как это предусмотрено в конструкции Фейстеля:

$$p_i = (L, R)$$

Выполним шифрование старшего полублока L(Left) с помощью младшего R(Right), используя некоторую криптографическую функцию f_i, зависящую от раундового ключа k_i, и обратимую бинарную операцию, например, операцию $\oplus$ — сложения по модулю 2, над полублоками данных. Для подготовки к следующему аналогичному раунду выполняется перестановка правой и левой частей блока. Таким образом, раундовая функция шифрования будет иметь следующий вид (рис. 1.6):

$$F_i(p_i) = F_i(L, R) = (R, L \oplus f_i(R)).$$

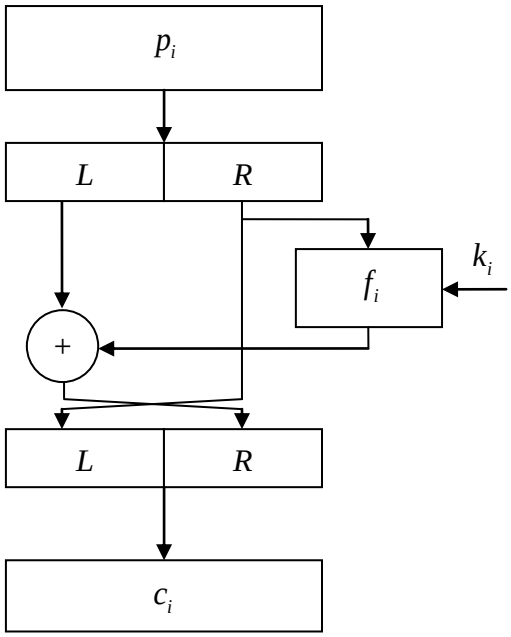

Рис. 1.6. Раундовая функция шифрования

Для раундовой функции шифрования можно построить обратное или дешифрирующее преобразование:

$$F_i^{-1}(c_i) = F_i^{-1}(L, R) = (R, L \oplus f_i(R)).$$

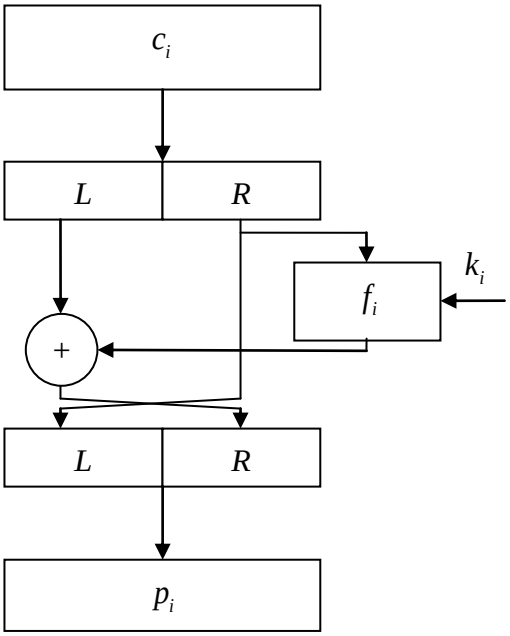

Рис. 1.7. Раундовая функция дешифрирования

Композиционный шифр, использующий раундовые функции такого вида, называется *композиционным шифром, построенным в соответствии с конструкцией Фейстеля*. Если функции f_i являются инволюцией, то прямое преобразование и обратное преобразование совпадают, а меняется только порядок используемых ключей. На рис. 1.6 и 1.7 рассмотрены раундовые преобразования в режиме простой замены. Из одного блока открытого текста после криптографического преобразования получается один блок шифрованного текста.

При использовании композиционных блочных шифров применяются различные схемы шифрования, известные под названием рабочих *режимов шифрования.* Режимы шифрования позволяют реализовать дополнительные, отсутствующие в исходной конструкции блочного шифра, функции и, тем самым, повысить криптографическую стойкость.

Композиционные методы шифрования обеспечивают зависимость каждого блока шифрованного текста не только от соответствующего блока открытого текста, но и от его номера. На рис. 1.8 показана схема шифрования в режиме *сцепления блоков шифрованного*

текста — CBC (Ciphertext Block Chaining), а на рис 1.9 — схема дешифрирования. На этих рисунках p_1, p_2, ..., p_m — блоки открытого текста, c_0, c_1, …, c_m — блоки шифрованного текста, E_1, E_2, ..., E_m — блоки шифрования, D_1, D_2, …, D_m — блоки дешифрирования.

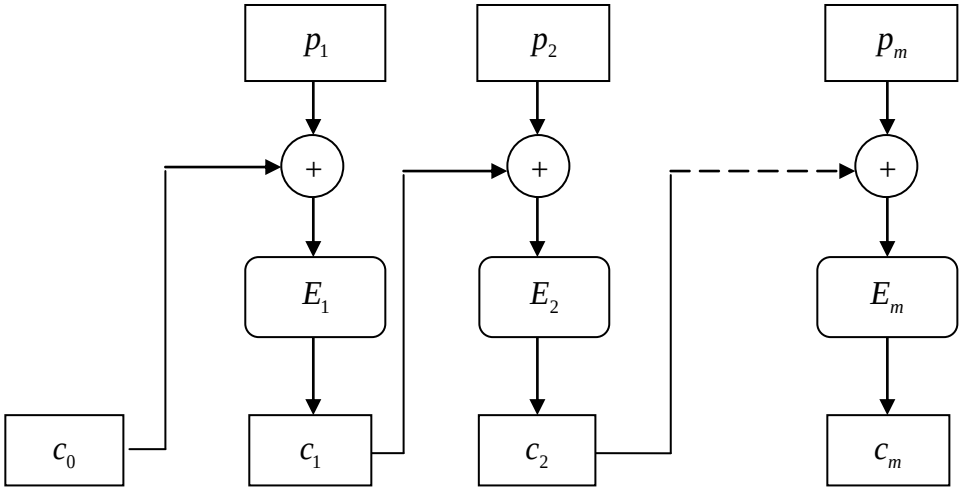

Рис. 1.8. Схема шифрования в режиме сцепления блоков шифрованного текста

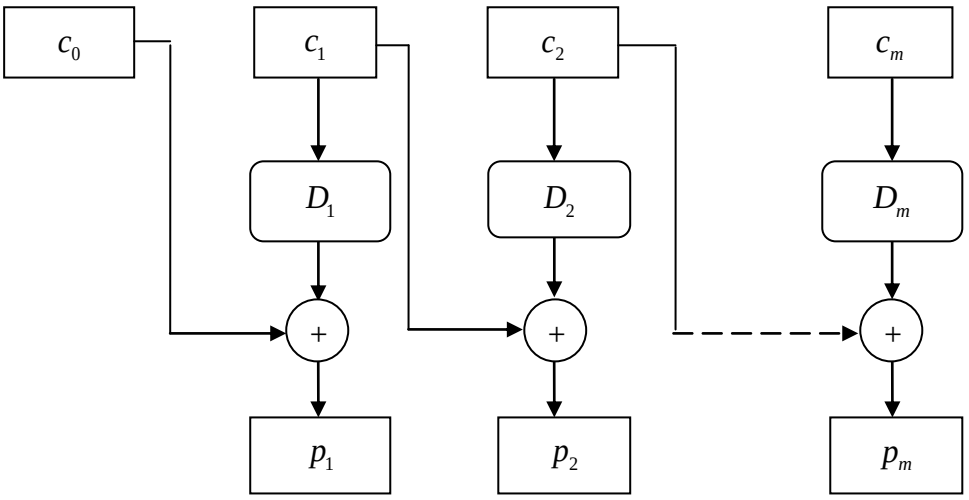

Рис. 1.9. Схема дешифрирования в режиме сцепления блоков шифрованного текста

Уравнения шифрования и дешифрирования в режиме CBC имеют вид

$$c_i = E_i\,(p_i + c_{i-1});$$

$$p_i = D_i\,(c_i) + c_{i-1};\ i = 1, 2, ..., m.$$

Отличительными особенностями режима СВС являются зависимость при шифровании i-го блока шифрованного текста от всех предшествующих блоков открытого текста и зависимость при дешифровании каждого блока открытого текста только от двух блоков шифрованного текста (c_{i-1} и c_i).

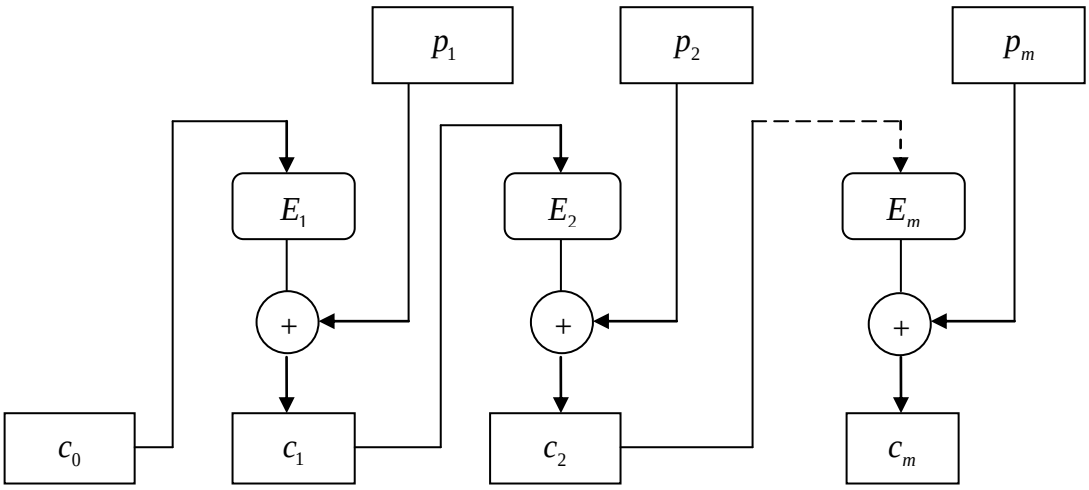

Рис. 1.10. Схема шифрования в режиме «обратная связь по шифрованному тексту»

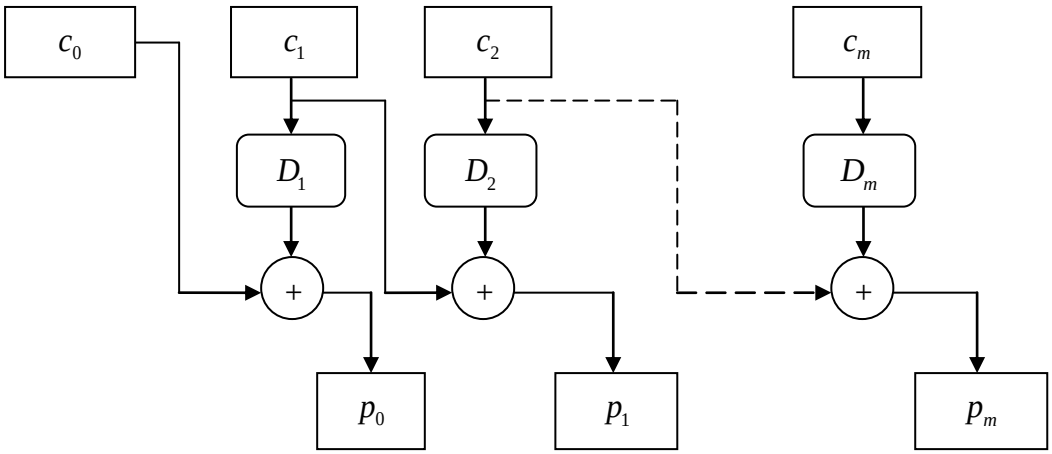

Рис. 1.11. Схема дешифрирования в режиме «обратная связь по шифрованному тексту»

Кроме рассмотренного выше режима шифрования CBC существует и множество других режимов шифрования, обеспечивающих повышенную криптографическую стойкость телекоммуникационной системы передачи информации. Рассмотрим, например, режим шифрования, который называется «режим обратная связь по шифрованному тексту» CFB (Ciphertext Feedback) (рис. 1.10 и 1.11).

Уравнения шифрования и дешифрирования имеют вид:

$$c_i = p_i + E_i(c_{i-1});$$

$$p_i = c_i + D_i(c_{i-1}); D_i = E_i, \ i = 1, 2, ..., m.$$

Заметим, что начальное состояние c_0 можно рассматривать как некоторое начальное состояние (синхропосылка). Кроме этих двух режимов в криптосистемах с секретным ключом, например, шифрах DES, ГОСТ 28147-89 и др. для сцепления блоков могут использоваться и другие схемы шифрования:

• режим обратной связи по выходу — OFB (Output Feedback);

• сцепление блоков открытого текста — PBC (Plaintext Block Chaining);

• обратная связь по открытому тексту — PFB (Plaintext Feedback);

• усиленное сцепление блоков шифрованного текста.

1.2. Шифр DES (Федеральный стандарт США)

Разновидностью шифра Фейстеля является созданный в 1974 г. шифр DES (Data Encryption Standard) и предложенный в качестве стандарта шифрования данных в государственных и частных организациях США. Шифр DES имеет длину блока исходных данных p равную 64 битам и ключ сложения по модулю 2 длиной 56 бит. Ключ, реализующий подстановку, является ключом длительного пользования, который выбирается по специальным критериям.

Схема преобразования представлена на рис. 1.12. На этом рисунке: p — блок открытого текста. В этом блоке предполагается, что открытый текст разбивается на части по 64 бита. Каждые 64 бита подвергаются всем преобразованиям, которые приведены на рис. 1.12.

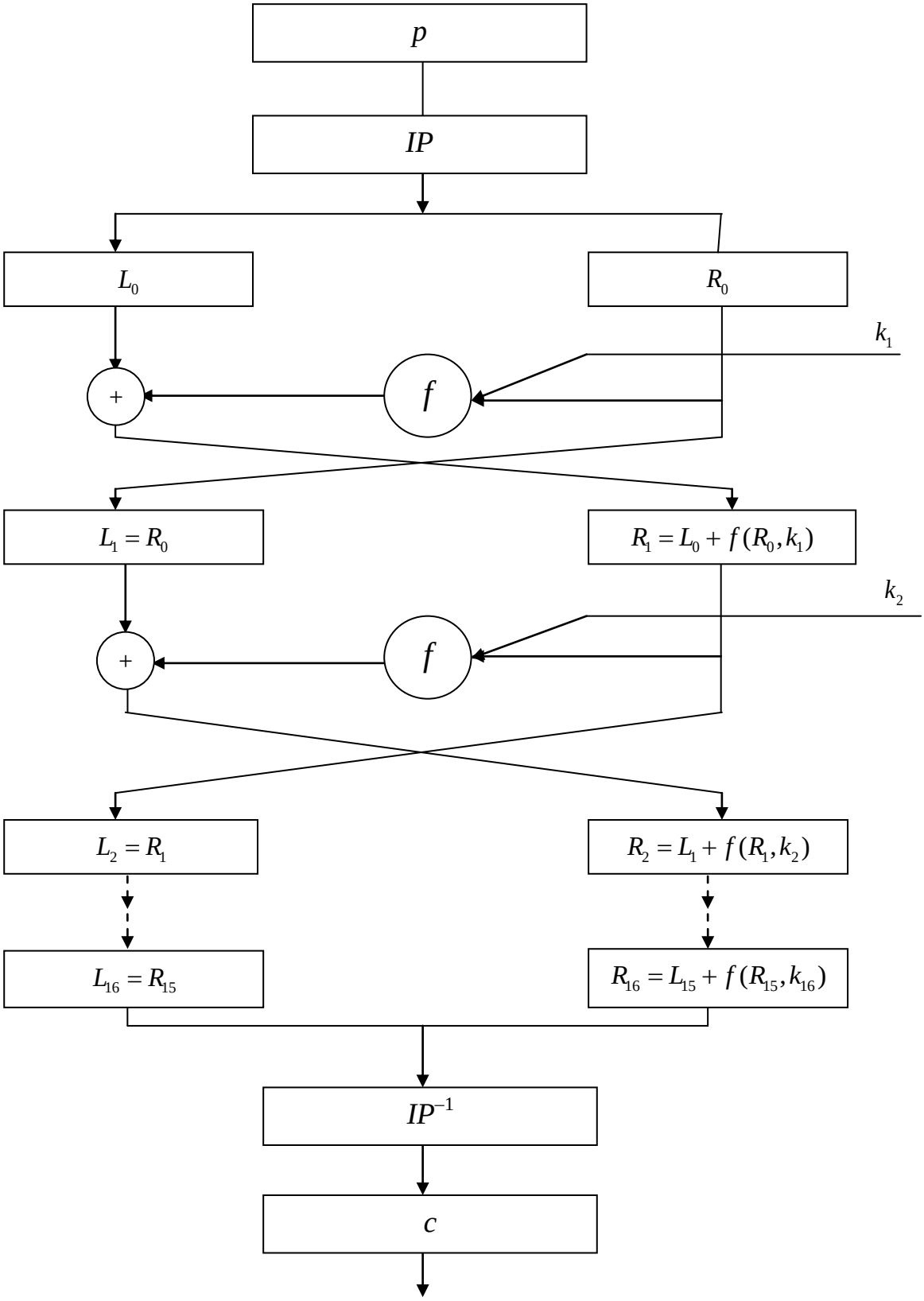

Рис. 1.12. Структура шифра DES

Это соответствует режиму шифрования «прямая замена», когда 64 бита обрабатываются независимо друг от друга и после всех преобразований появляется 64 бита зашифрованного текста. На выходе схемы преобразования находится блок c — блок шифрованного текста. Кроме этих блоков в схему преобразования входят: IP — блок начальной перестановки, IP^{-1}— блок конечной перестановки, f — функция криптографического преобразования, L — левые полублоки, R_i — правые полублоки, $i = 0, 1, 2, \ldots, 15$.

Блок открытого текста (64 бита) подвергается начальной перестановке, согласно табл. 1.1. В соответствии с этой таблицей 58 бит открытого текста становится первым битом, 50 бит становится вторым битом, 42 бит — третьим, а первый бит открытого текста перемещается на 40 позицию и т. д. После блока начальной перестановки полученные, 64 бита разбиваются на две половины по 32 бита, и получаются, соответственно, левый полублок L и правый полублок R.

Таблица 1.1

Блок начальной перестановки

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

Биты, находящиеся в правом полублоке R, подвергаются криптографическому преобразованию с помощью функции $f(R, k)$, зависящей от ключа и складываются по модулю 2 с данными из левого полублока. Затем данные в этих полублоках меняются местами. Эта процедура в шифре DES повторяется 16 раз.Таким образом, в общем виде раундовая функция шифрования на i-м шаге представлена на рис. 1.13.

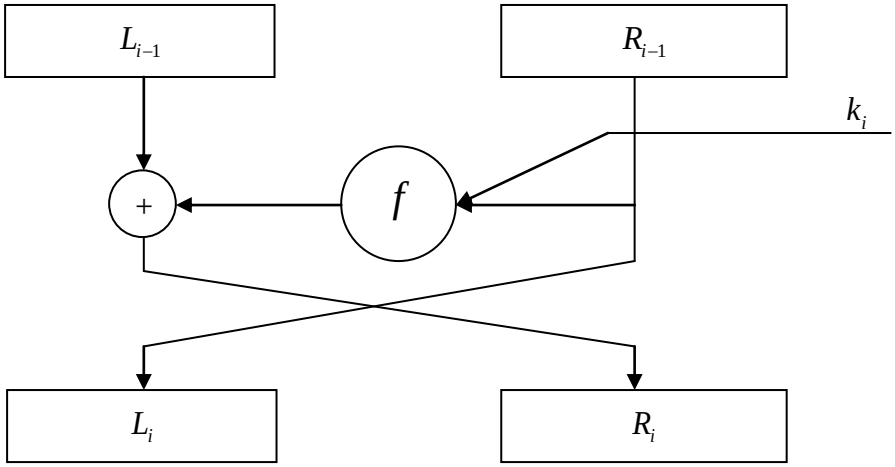

Рис. 1.13. Раундовая функция *i*-го шага шифрования DES

На этом рисунке L_i и R_i — левый и правый полублоки, полученные в результате i-й итерации, k_i — 48-битный ключ, а f — функция криптографического преобразования, выполняющая все подстановки, перестановки и операцию сложения по модулю 2 с ключом. Один цикл криптографического преобразования можно обобщенно записать следующим образом:

$$(L_i, R_i) = (R_{i-1}, L_{i-1} + f(R_{i-1}, k_i)).$$

Алгоритм криптографического преобразования функции f более подробно рассмотрен на рис. 1.14. На этом рисунке S — блок замен и P — блок перестановки, которые соответствуют эквивалентным блокам, представленным на общей структурной схемы раунда шифрования, приведенной на рис. 1.5.

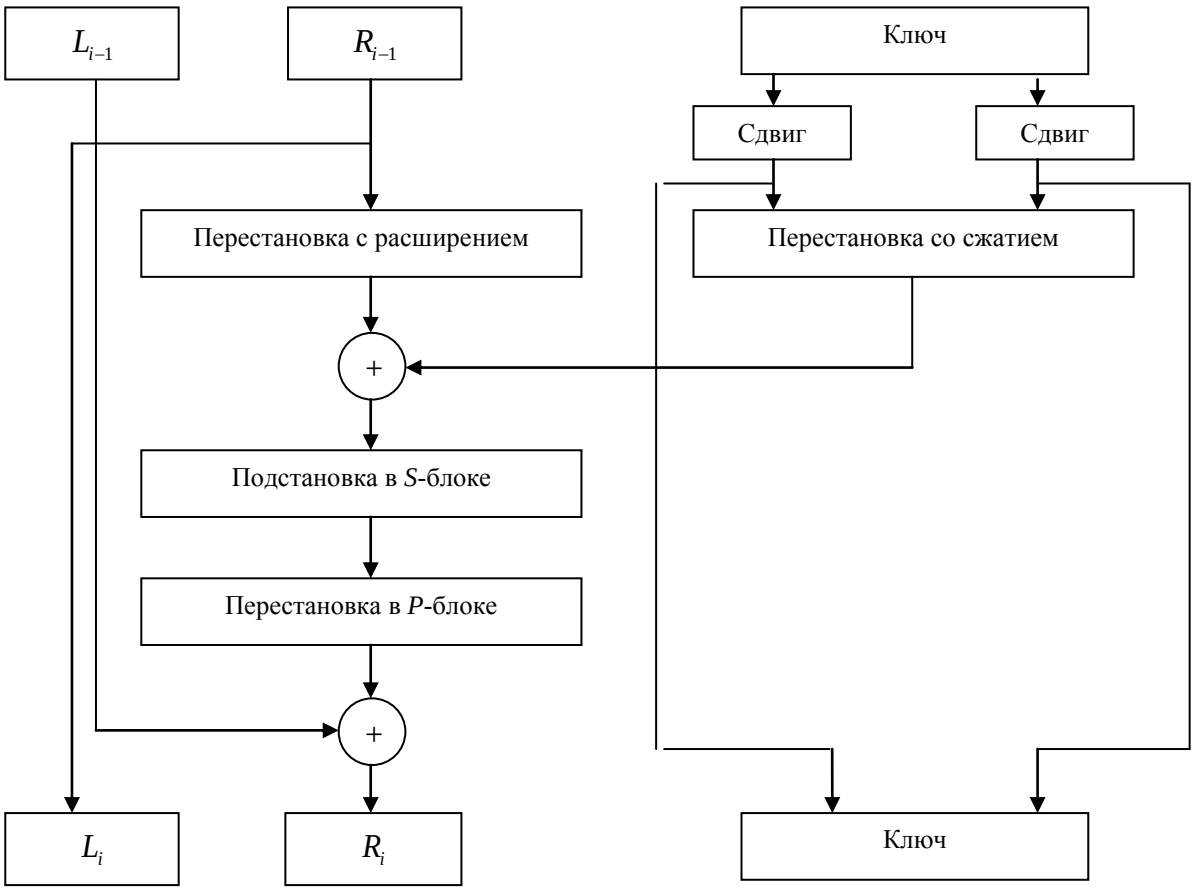

Рис. 1.14. Раундовая функция шифрования DES с учетом ключевой информации

На каждом раунде шифрования предыдущий правый полублок R_{i-1} становится новым левым полублоком L_i, а с правым полублоком осуществляются следующие преобразования.

- Правый полублок R_{i-1} разбивается на 8 тетрад по 4 бита. Тетрады по циклическому закону дополняются крайними битами из соседних тетрад до 6 разрядных слов так, что получается 8 шестиразрядных слов, что в данной схеме обозначено как перестановка с расширением. Формально правило обработки 32 битов в операции перестановки с расширением отражено в табл. 1.2. Эта процедура позволяет привести размер правой половины в соответствие с размером 48-битового ключа.

Таблица 1.2

Перестановка с расширением

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

- Полученное после перестановки с расширением 48-разрядное слово суммируется по модулю 2 с 48 битами ключа, которые выбираются по специальному списку из 56 бит ключевого массива.

- После суммирования с ключом 48 бит разбиваются на 8 блоков из 6 восьмиразрядных символов, которые поступают на вход *S*-блока четырехразрядной подстановки, содержащего 8 видов подстановок (рис. 1.15.), причем каждый блок поступает на свой блок подстановки. Крайний левый и крайний правый биты каждого из восьми шестиразрядных символов дают сочетание от 00 до 11, которое определяет номер используемой строки в блоке подстановки (табл. 1.3–1.10), а оставшиеся четыре символа определяют номер столбца. В результате подстановки каждый символ полублока заменяется соответствующей тетрадой из блока (*S*-блока) подстановки, где 48-битный вход представляет собой результат перестановки с расширением правого полублока. S_i — блоки замен, где 32-битный выход представляет собой результат подстановки в блоке замен *S*. Замена осуществляется по правилам, приведенным в табл. 1.3–1.11, где каждому из восьми шестиразрядных символов соответствует свой блок замен: первому символу — блок S_1, второму S_2, и т. д. В блоке замен шестиразрядный символ заменятся десятичным числом, выбранным из соответствующей таблицы замен из соответствующей строки и столбца. Десятичные числа в таблице замен лежат в интервале от нуля до пятнадцати, что соответствует тетраде бит, принимающих значения в интервале от 0000 до 1111. Подстановка с помощью *S*-блоков

является тем элементом DES, который реализует нелинейную операцию. Именно эти блоки в большей степени, чем все остальные действия определяют криптографическая стойкость шифра DES. Другие элементы алгоритма шифрования выполняют линейные операции и относительно легко поддаются анализу.

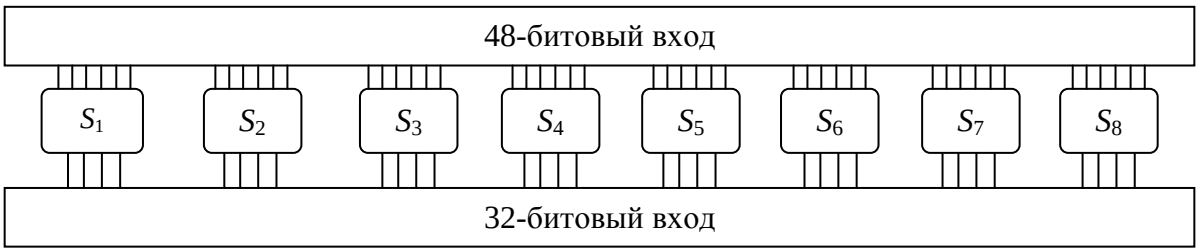

Рис. 1.15. Структура блока замен

Таблица 1.3

Блок замен S_1

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

Таблица 1.4

Блок замен S_2

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

Таблица 1.5

Блок замен S_3

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

Таблица 1.6

Блок замен S_4

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

Таблица 1.7

Блок замен S_5

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

Таблица 1.8

Блок замен S_6

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

Таблица 1.9

Блок замен S_7

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

Таблица 1.10

Блок замен S_8

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

- Результат подстановки в блоках замен *S*, состоящий из 32 бит, полученный в предыдущей операции, подвергается перестановке, согласно табл. 1.11.

Таблица 1.11

Блок перестановки *P*-блок

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

- Полученные в результате перестановки биты в правом полублоке суммируются по модулю 2 с битами, находящимися в левом полублоке L_{i-1}. Таким образом, получается правый полублок R_i, и начинается новый криптографический раунд. Всего проводится 16 раундов, после чего данные в полублоке *R* и данные в полублоке *L* объединяются.
- Полученный после 16 раундов шифрования блок текста (64 бита) подвергается заключительной перестановке, согласно табл. 1.12. В соответствии с этой таблицей 40 бит текста становится первым битом, 8 бит становится вторым битом, 48 бит — третьим, а первый бит текста перемещается на 58 позицию и т. д.

Таблица 1.12

Блок заключительной перестановки

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

Вернемся к рассмотрению рис. 1.14. Алгоритм, который создает ключ для каждого цикла, является циклическим алгоритмом. Объем ключевого пространства в шифре DES равен 2^{56}. Будем называть ключевым массивом последовательность из 56 бит. Эта последовательность в каждом раунде циклически сдвигается вправо. Число позиций сдвига в каждом раунде определяется массивом m из 16 элементов

$$m = \{0, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1\}.$$

Значение элемента в этом массиве соответствует числу позиций сдвига на каждом раунде шифрования.

Ключевой массив определяется для каждой итерации в следующей последовательности:

- 56 бит ключа K разделяются на 2 блока: левый LK_1 и правый RK_1 по 28 бит каждый;
- производится циклический сдвиг $m[1]$ раз блоков LK_1 и RK_1 для получения блоков LK_2 и RK_2, а затем с помощью сдвига $m[2]$ получаются блоки LK_2 и RK_2 и далее по формуле

$$LK_{i+1} = m[i+1]\, LK_i;\ \ RK_{i+1} = m[i+1]\, RK_i.$$

Величина сдвига $m[i]$ зависит от номера итерации. Далее производится перестановка со сжатием, согласно табл. 1.13.

Таблица 1.13

Блок перестановки со сжатием

14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

Согласно этой таблице из 56 бит выбирается только 48 бит ключевого массива, причем 14 бит становится первым, 17 — вторым и т. д. Полученные, в результате 48 бит ключа складываются по модулю 2 в каждом раунде с 48-битовым массивом данных полублока R, полученным после блока перестановки с расширением (рис. 1.14).

Рассмотрим в качестве примера ситуацию, когда блок открытого текста состоит из 8 байт (64 бита). Значения цифр в этом блоке p_{i-1} равны величинам 1, 2, 3, 4, 5, 6, 7, 8. Согласно рассмотренному выше алгоритму обработки входного буфера открытого текста, цифры в блоке открытого текста разбиваются на две равные половины по 32 бита. Тогда левой половине L_{i-1} соответствуют байты 1, 2, 3, 4, а правой половине R_{i-1} байты 5, 6, 7, 8. В алгоритме обработки предполагается, что в каждом раунде данные полублоков меняются местами, то новые данные в левом полублоке равны предыдущим данным, находившимся в правом полублоке:

$$L_i = R_{i-1} = \{5, 6, 7, 8\},$$

а новое значение данных в правом полублоке определяется выражением

$$R_i = L_{i-1} + f(R_{i-1}, k_i).$$

Вычислим величину R_i, согласно алгоритму обработки, представленному на рис. 1.12.

До перестановки с расширением байтам 5, 6, 7, 8 соответствует строка битов вида

00000101 00000110 00000111 00001000.

После перестановки с расширением по табл. 1.2 имеем массив из 48 битов разбитых на 8 групп по 6 бит каждая:

000000 001010 100000 001100 000000 001110 100001 010000.

Предположим, что ключевой массив из 48 бит состоит только из нулей, тогда операция суммирования по модулю 2 блока, полученного после перестановки с расширением с 48 битами ключа, не изменит этот массив. При входе в блок замен крайний левый и крайний правый биты каждого из шестиразрядных символов дают сочетание от 00 до 11, которое определяет номер используемой строки в блоке подстановки, а оставшиеся четыре символа определяют номер столбца. Запишем эти шестиразрядные символы в виде пар чисел, где первое число соответствует номеру строки в блоке замены, а четыре средних бита (средняя тетрада) соответствует номеру столбца в таблице замен

(00), (05), (20), (06), (00), (07), (30), (08).

Тогда первой паре чисел (00) после блока замены S_1 (табл. 1.3) соответствует число 14, а последней паре (08) после блока замены S_8 (табл. 1.9) соответствует число 10. Таким образом, после таблицы замен получим следующие 8 десятичных чисел

14, 11, 13, 9, 2, 8, 6, 10.

Записывая их в шестнадцатеричной системе счисления, получим 8 шестнадцатеричных чисел

E, B, D, 9, 2, 8, 6, A.

Этим числам соответствует строка из 32 битов.

После блока перестановки, в соответствии с табл. 1.13, полученные биты записываются в шестнадцатеричном виде

DC9B C7C0.

Окончательно, после суммирования с битами, находящимися в левом полублоке, имеем зашифрованный текст, находящийся в правом полублоке. Этот текст имеет следующий вид в шестнадцатеричной системе счисления:

DD99 C4C4.

Биты в левом полублоке записываются в виде

05060708.

1.3. Шифр ГОСТ 28147-89 (Всероссийский стандарт)

Шифр ГОСТ 28147-89 (далее будем обозначать этот шифр как шифр ГОСТ) является классическим примером итерационного блочного шифра Фейстеля с разрядностью блока данных, равной 64 бита. Объем ключевого пространства ГОСТ равен 2^{256}. Ключевая информация представляет собой массив данных размерности 256 бит. Каждый массив из 256 бит представляется как массив из восьми 32 разрядных элементов $\{k_m\}$, $m = 0, 1, 2, ..., 7$. Таблица замен S представляет собой набор из восьми одномерных массивов $\{S_m\}$, $m = 0, 1, 2, ..., 7$, так называемых узлов замены, каждый из которых определяет алгоритм работы 4-разрядного блока подстановок (S-блока). На вход блока замен поступает 32 битовая последовательность, которая разбивается на восемь 4 битовых подпоследовательностей. Каждый S-блок представляет собой таблицу из шестнадцати строк, каждая из которых содержит по четыре бита заполнения в строке. Заполнение в строке представляет собой некоторую перестановку чисел от 0 до 15. Первая 4 битовая подпоследовательность попадает на вход первого S-блока, вторая 4 битовая подпоследовательность поступает на вход второго S-блока и т. д.

Пусть соответствующий S-блок представляет собой последовательность из 16 десятичных чисел в интервале от 0 до 15, расположенных в следующем порядке: 1, 15, 13, 0, 5, 7, 10, 4, 9, 2, 3, 14, 6, 11, 8, 12. Если на вход S-блока подается 4 битовая подпоследовательность, равная 0, то на его выходе будет 1. При подаче на вход S-блока 4 битовой подпоследовательности, равной 4 на выходе будет 5. Если на вход S-блока подается 4 битовая подпоследовательность, равная 12, то на выходе будет 6 и т. д. Пример реализации блока замен шифра ГОСТ приведен в табл. 1.14.

Таблица 1.14

Таблица замен шифра ГОСТ

Номер *S*-блока	Значение															
1	4	10	9	2	13	8	0	14	6	11	1	12	7	15	5	3
2	14	11	4	12	6	13	15	10	2	3	8	1	0	7	5	9
3	5	8	1	13	10	3	4	2	14	15	12	7	6	0	9	11
4	7	13	10	1	0	8	9	15	14	4	6	12	11	2	5	3
5	6	12	7	1	5	15	13	8	4	10	9	14	0	3	11	2
6	4	11	10	0	7	2	1	13	3	6	8	5	9	12	15	14
7	13	11	4	1	3	15	5	9	0	10	14	7	6	8	2	12
8	1	15	13	0	5	7	10	4	9	2	3	14	6	11	8	12

Выходы всех восьми *S*-блоков объединяются в 32-битное слово. После этого слово циклически сдвигается влево (к старшим разрядам) на 11 битов. Все восемь *S*-блоков могут быть различными.

Более подробно структура основного шага криптопреобразования шифра ГОСТ показана рис. 1.16, где d^i — 64-битовый блок данных на *i*-м раунде криптопреобразования, *L* и *R* — соответственно, левый и правый 32 битовые полублоки блока d^i, *X* — 32-разрядный ключевой массив, « + » — операция побитового сложения по модулю 2.

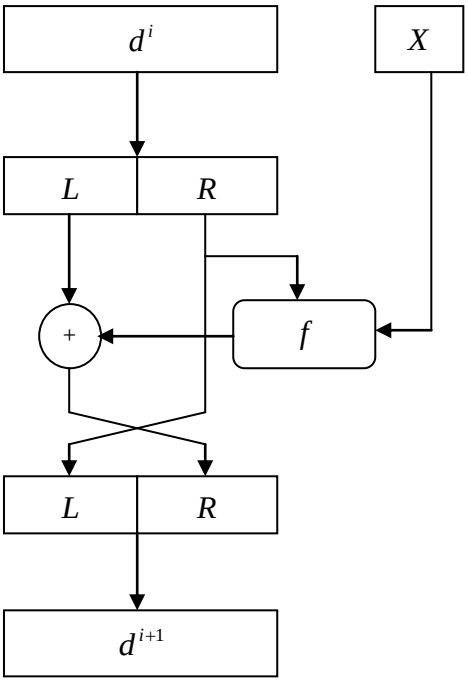

Рис. 1.16. Раундовая функция шифра ГОСТ

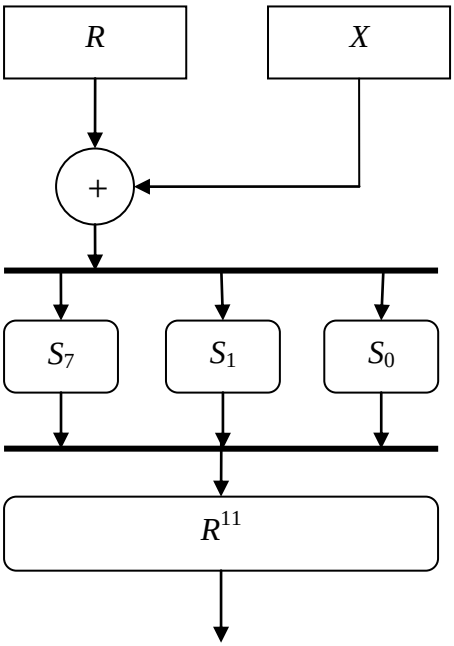

Рис. 1.17. Раундовая функция шифра ГОСТ с учетом блока замен

В качестве исходных данных каждый шаг криптообразования получает 64 разрядный бок данных $d = (L, R)$ и 32 разрядный ключ X, в качестве которого используется один из элементов ключевого массива k_m (рис. 1.17). Более подробно преобразования, осуществляемые с правой половиной R в функции f, представлены на рис. 1.17, где суммирование осуществляется по модулю $2^{32,}$ а R^{11} представляет собой циклический сдвиг на 11 разрядов влево.

В ходе выполнения одного шага криптопреобразования данные в правой половине блока R подвергаются следующим преобразованиям.

- Сложение по модулю 2^{32} данных полублока R и ключа X.
- Разбиение результата S на восемь 4 битовых блоков, поблочной замене по таблице замен, формированию из получившихся блоков нового значения S.
- Циклический сдвиг результата S на 11 разрядов влево.
- Поразрядное сложение по модулю 2 результата S и данных, записанных в левой половине блока L.
- Данные в правой половине блока R становятся новым значением элемента L, а значение результата предыдущей операции, описанной в предыдущем пункте, становится новым значением элемента R (рис. 1.16).

В шифре ГОСТ предусмотрено, что последовательно осуществляется 32 таких раунда.

Описанный выше алгоритм функционирования называется режимом простой замены, когда блоки открытого текста обрабатываются независимо друг от друга. Кроме того, в шифре ГОСТ предусмотрены режимы работы, аналогичные рассмотренным выше режимам шифра DES, которые обладают повышенной криптографической стойкостью. Шифр DES являлся прототипом для шифра ГОСТ 28147-89.

1.4. Поточные шифры

Структура простейшего устройства [3] для реализации поточного метода шифрования по битам представлена на рис. 1.18.

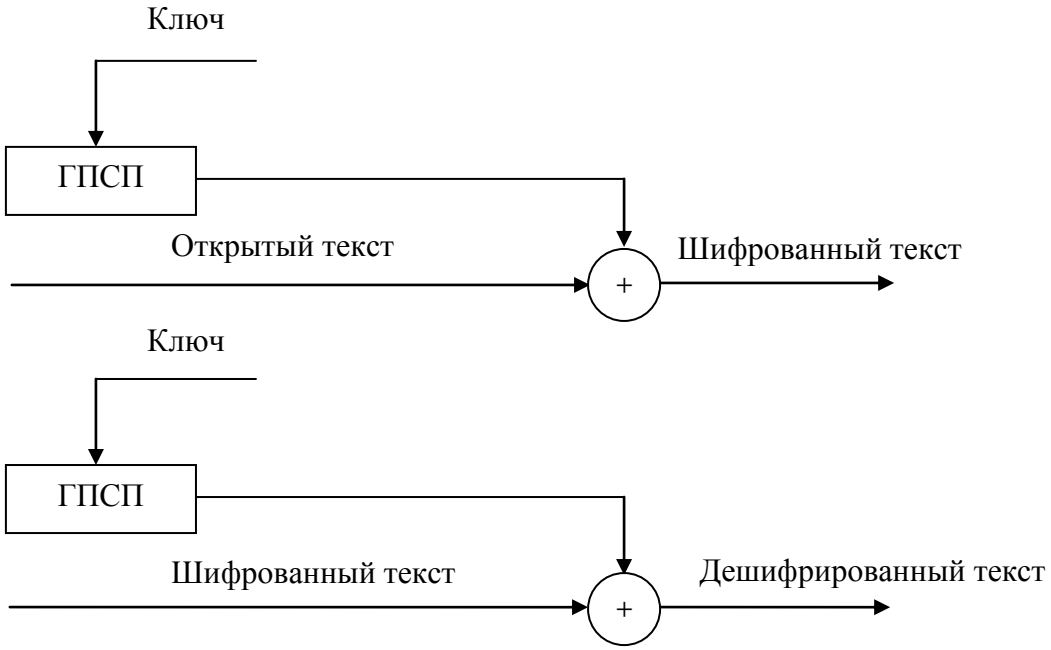

Рис. 1.18. Структура поточного шифра

На этом рисунке: символом + обозначена операция сложения по модулю 2, ГПСП — генератор псевдослучайной последовательности, построенный обычно на вычислениях в поле Галуа $GF(p^m)$, образованного делением на неприводимый многочлен степени m (разд. 3). В синхронных поточных шифрах ключ формируется независимо от входной информационной последовательности.

Стойкость криптографической системы целиком зависит от статистических свойств [4] генератора псевдослучайной последовательности (ГПСП). Разумеется, в случае, когда используется последовательность истинно случайных чисел бесконечной длины, обеспечивается идеальная криптографическая стойкость телекоммуникационной системы передачи информации. Если псевдослучайная последовательность имеет небольшой период повторения, то криптографиче-

ская стойкость такого шифра относительно невелика. Реальная криптографическая стойкость потоковых шифров зависит от конкретной реализации ГПСП. Потоковые шифры наиболее пригодны для шифрования непрерывных потоков данных, например, в телекоммуникационных сетях передачи информации.

Существуют два основных типа потоковых шифров: самосинхронизирующиеся и синхронные. Основная идея самосинхронизирующихся потоковых шифров (CipherTextAutoKey) заключается в том, что внутреннее состояние генератора псевдослучайной последовательности является функцией фиксированного числа предшествующих битов шифрованного текста. Поскольку внутреннее состояние зависит только от n бит шифрованного текста, генератор псевдослучайной последовательности на приемной стороне войдет в синхронизм с генератором псевдослучайной последовательности на передающей стороне после получения определенного числа бит. С учетом этого передача сообщений выглядит следующим образом.

Каждое сообщение предваряется случайным заголовком определенной длины. Самосинхронизирующиеся потоковые шифры уязвимы для атак типа «воспроизведение», когда записывается достаточное число бит шифрованного текста, и затем они передаются на приемную сторону. Приемная сторона входит в состояние синхронизма, и не существует средств, чтобы определить, что эти данные искажены несанкционированным абонентом.

При использовании синхронных шифров выходные значения генератора ПСП не зависят от исходного или шифрованного текста. Основная сложность заключается в необходимости синхронизации генераторов ПСП на приемной и передающей сторонах. Если в процессе синхронизации неправильно принят хоть один бит, то вся последовательность не может быть расшифрована и необходима повторная синхронизация.

Наиболее часто ГПСП поточных шифров строятся на основе класса вычетов многочленов по модулю $p(x)$ неприводимого многочлена

степени n, которое образует поле Галуа $GF(p^n)$ с конечным числом элементов. В этой связи необходимо рассмотреть методы практической реализации вычислений в полях многочленов.

1.4.1. Реализация вычислений в полях многочленов с помощью регистров с прямыми и обратными связями

Рассмотрим операцию умножения многочлена

$$f(x) = f_n x^n + f_{n-1} x^{n-1} + \ldots + f_1 x + f_0$$

на многочлен

$$h(x) = h_N x^N + h_{N-1} x^{N-1} + \ldots + h_1 x + h_0.$$

Непосредственно перемножая многочлены и приводя подобные члены при одинаковых степенях, получаем:

$$y(x) = f(x)\, h(x) = y_{n+N} x^{n+N} + y_{n+N-1} x^{n+N-1} + \ldots + y_1 x + y_0,$$

где

$$y_0 = f_0 h_0;$$
$$y_1 = f_0 h_1 + f_1 h_0;$$
$$y_2 = f_0 h_2 + f_1 h_1 + f_2 h_0;$$
$$\ldots .$$

Для выполнения операции умножения многочленов используются схемы на основе регистров сдвига (рис. 1.19 и 1.20).

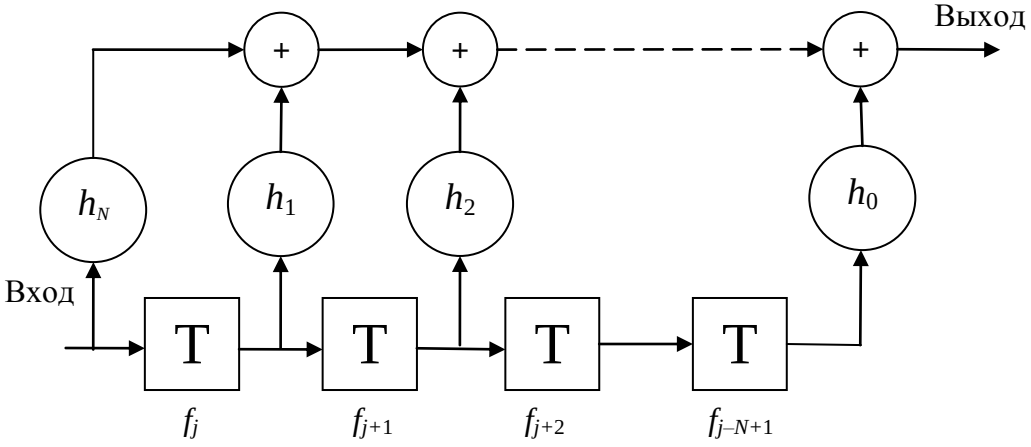

Рис. 1.19. Структурная схема устройства умножения многочленов на базе регистра сдвига

При поступлении на вход устройства первого коэффициента многочлена $f(x)$, равного f_n, на выходе появляется значение первого коэффициента произведения $f_n\, h_N$. В этот момент времени все ячейки регистра сдвига, куда записываются значения f_j, f_{j+1}, f_{j+2} и т. д., содержат нули. В следующий момент времени на входе появляется значение второго коэффициента f_{n-1}, а значение f_n содержится в первой ячейке регистра сдвига. На выходе устройства умножения многочленов появляются значения произведений $f_{n-1}\, h_N + f_n\, h_{N-1}$ и так далее.

Рассмотренный выше алгоритм умножения многочленов может быть реализован на регистре сдвига (рис. 1.20) с включенными параллельно коэффициентами многочлена $h(x)$. В начальном состоянии в регистре сдвига записаны нулевые значения чисел. При появлении на входе устройства значения f_n происходит сложение $f_n h(x)$ с содержимым регистра сдвига. В результате сдвига регистра на один такт производится умножение на x. После появления на входе устройства значения f_{n-1} происходит сложение $f_{n-1} h(x)$ с содержимым регистра, и снова производится умножение на x.

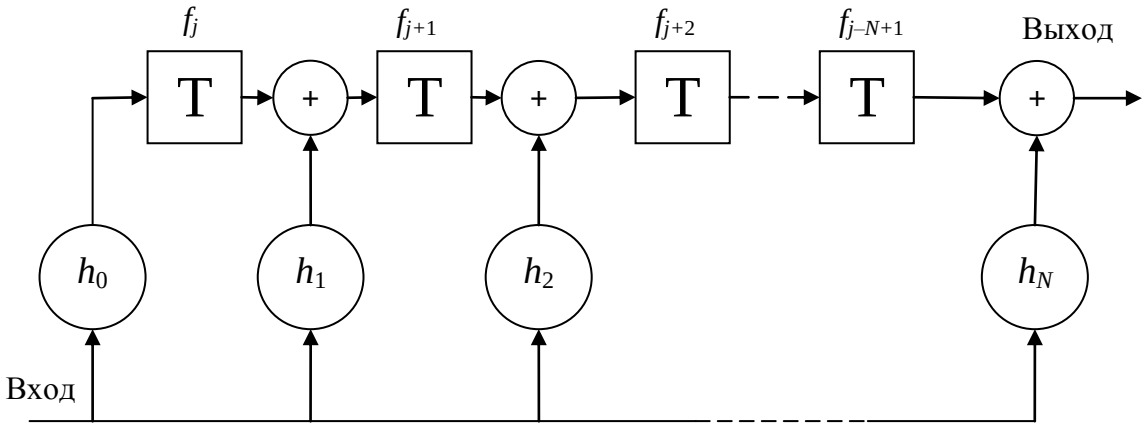

Рис. 1.20. Структурная схема устройства умножения многочленов с параллельным включением коэффициентов многочлена

Рассмотрим операцию деления многочлена

$$f(x) = f_n x^n + f_{n-1} x^{n-1} + \ldots + f_1 x + f_0$$

на многочлен

$$h(x) = h_N x^N + h_{N-1} x^{N-1} + ... + h_1 x + h_0.$$

На рис. 1.21 приведена структурная схема устройства деления многочленов на базе регистров сдвига.

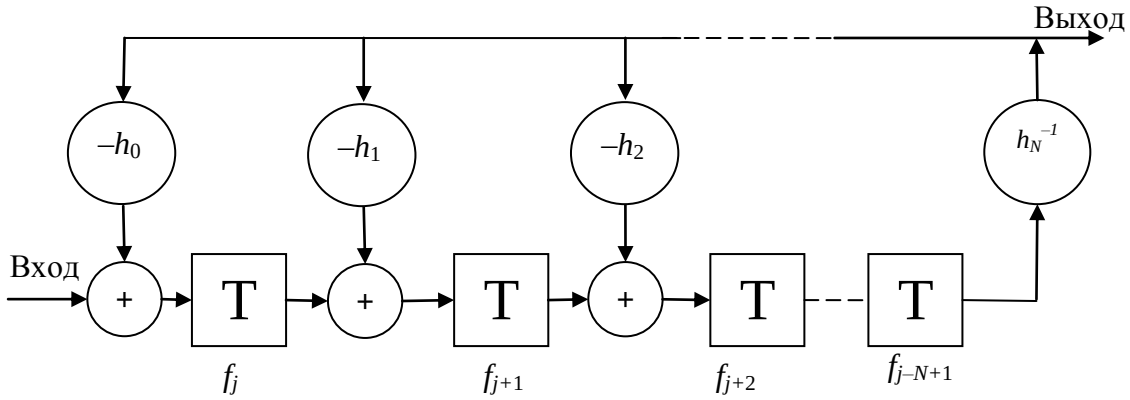

Рис. 1.21. Структурная схема устройства деления многочленов

В начальном состоянии в регистре сдвига находятся нулевые значения и выходной сигнал равен нулю для первых N сдвигов. После этого появляется первое ненулевое выходное значение равное $f_n (h_N)^{-1}$. Процесс деления заканчивается после n сдвигов, причем на выходе формируется частное от деления многочленов, а в регистре сдвига будет находиться остаток от деления.

В качестве примера на рис. 1.22 приведено схема устройства деления многочлена $f(x)$ на многочлен $h(x) = x^6 + x^5 + x^4 + x^3 + 1$.

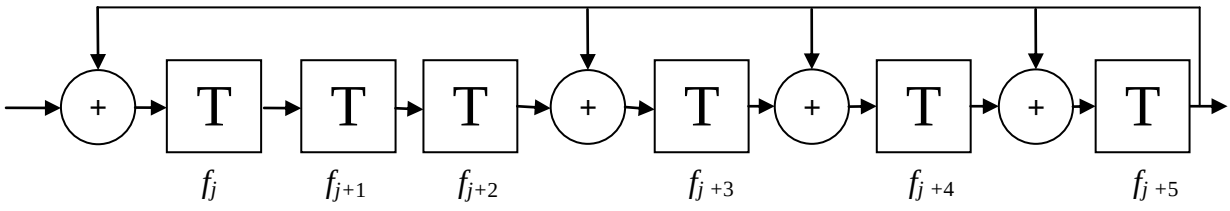

Рис. 1.22. Пример деления многочленов

1.4.2. Регистры с линейными обратными связями

Общая схема регистра с линейными обратными связями приведена на рис. 1.23. При ненулевом начальном состоянии регистра при

сдвиге данных, записанных в ячейках регистра, сигнал на выходе циклически повторяется, причем период повторения зависит от того, какой многочлен соответствует включенным обратным связям.

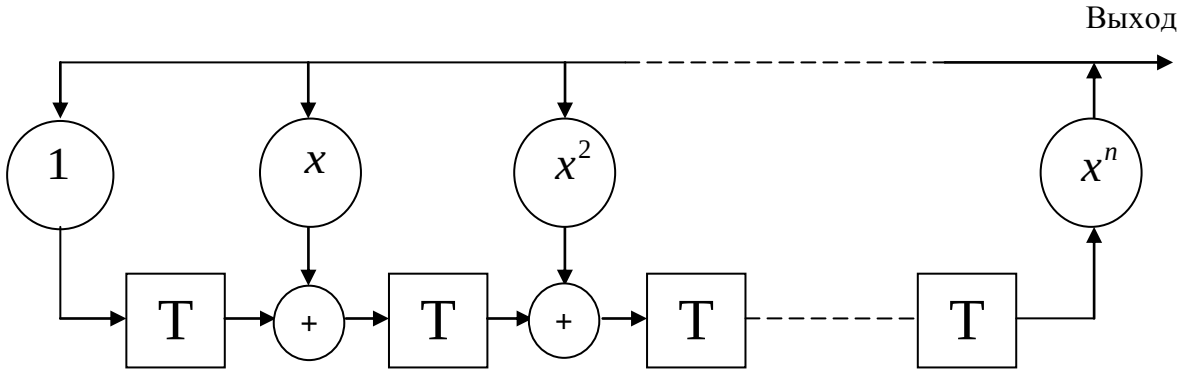

Рис. 1.23. Общая схема регистра с линейными обратными связями

Обратные связи включаются путем соединения шины «Выход» через электронные ключи «1», «x», «x^2»…«x^n» с соответствующими сумматорами по модулю 2. Например, структурная схема регистра с линейными обратными связями вида $x^6 + x^5 + x^4 + x^3 + 1$ совпадает со структурной схемой деления многочленов, приведенной на рис. 1.22.

Схемы на базе регистров с линейными обратными связями используются в качестве генераторов псевдослучайных последовательностей, датчиков случайных чисел, модуляторов частоты или фазы широкополосных сигналов и других устройствах. В зависимости от вариантов включения обратной связи и начальных состояний регистра сдвига на выходе устройства будут формироваться различные формы псевдослучайных последовательностей символов. Если включенные обратные связи соответствуют неприводимому многочлену, то все возможные состояния регистра образуют поле многочленов $GF(2^m)$ и число таких состояний оказывается равным $q = 2^m - 1$, где

m — степень многочлена. Любое другое включение обратных связей приводит к уменьшению числа состояний.

1.4.3. Последовательности максимальной длины

В криптографии регистры с линейными обратными связями, имеющие период повторения последовательностей, равный $2^m - 1$, используются, как генераторы псевдослучайных последовательностей (ГПСП). При достаточно большом периоде повторения можно считать, что на выходе этих устройств значения 0 или 1 появляются независимо друг от друга с вероятностью 0,5. Такие генераторы ГПСП реализуются на регистре с линейными обратными связями (рис. 1.23).

Рассмотрим последовательность максимальной длины на примере многочлена $x^4 + x + 1$. Это многочлен является примитивным. Его корень $\alpha = x$ является примитивным элементом поля $GF(2^4)$, и его степени образуют мультипликативную группу максимального порядка, т. е степени примитивного элемента $\alpha^0, \alpha^1, \alpha^2, ..., \alpha^{15}$, где $\alpha^0 = 1$ и $\alpha^{15} = 1$ однозначно соответствуют всем ненулевым элементам поля $GF(2^4)$.

Схема деления на многочлен $x^4 + x + 1$ приведена на рис. 1.24.

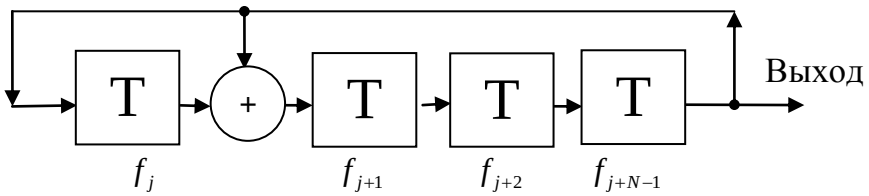

Рис. 1.24. Структурная схема генератора последовательности максимальной длины на основе многочлена $x^4 + x + 1$

Заметим, что единице, сдвигаемой из ячейки старшего разряда регистра сдвига, соответствует элемент α^4**,** и он заменяется элементом $\alpha + 1$.

Другой вариант генератора последовательности максимальной длины схемы приведен на рис. 1.25.

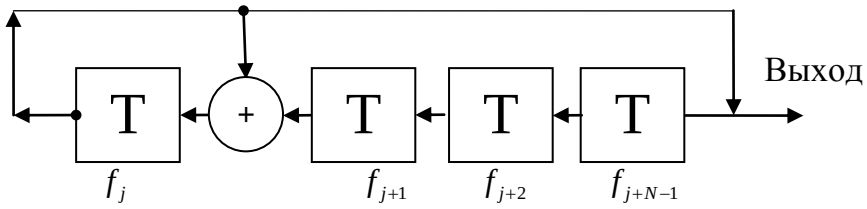

Рис. 1.25. Структурная схема генератора ПСП на основе деления многочлена

Данное устройство деления многочленов также может быть использовано в качестве генератора последовательности максимальной длины. В этом случае форма генерируемой последовательности полностью определяется ненулевым начальным состоянием регистра. Сдвиг влево соответствует делению на α, так что единица, выходящая из ячейки младшего разряда, дает значение α^{-1}, заменяемое эквивалентным значением $\alpha^3 + 1$.

1.4.4. Оценка качества последовательностей максимальной длины

Для оценки качества генераторов псевдослучайной последовательности [4] используются две группы тестов.

Графические тесты. В соответствии этим тестом строятся определенные графические зависимости, *гистограммы,* и по их виду делается вывод о свойствах тестируемой последовательности. Гистограмма позволяет определить равномерность появления символов в тестируемой последовательности, а также оценить частоту появления конкретного символа.

Оценочные тесты. На основе оценочных тестов делается заключение о степени близости статистических свойств анализируемой и истинно случайной последовательности, например, в соответствии с критерием χ^2 (хи-квадрат).

Рассмотрим понятие «критерий χ^2». Пусть результаты n испытаний таковы, что их можно разбить на k интервалов. Обозначим через p_i вероятность, того, что результат испытания попадает в i-й интервал, а через y_i — число испытаний которые реально попали в i-й интервал. Тогда можно написать следующее соотношение:

$$v = \sum_{i=1}^{k} \frac{(y_i - np_i)^2}{np_i}.$$

Для оценки результата используются таблицы распределения χ^2.

К оценочным тестам относится тест на равномерность появления символов в последовательности максимальной длины. При этом анализируется число n байт в последовательности. Один байт соответствует целому числу, находящемуся в интервале 0…255. В этой связи определяется, сколько раз y_i встречается в n испытаниях каждый символ s_i (i = 0, 1, 2, ..., 255), после чего применяется критерий χ^2:

$$v = \sum_{i=0}^{255} \frac{(y_i - n/256)^2}{(n/256)}.$$

Используется тест на равномерность распределения символов 0 и 1 в последовательности максимальной длины. Для этого подсчитывается число нулей (y_0) и число единиц (y_1) в последовательности символов, состоящей из n бит. Далее применяется критерий χ^2.

1.4.5. CRC-код

В криптографических системах для обеспечения целостности передачи информации используется CRC-код (cyclic redundancy check, проверка циклической суммы), выполняющий функцию аутентификации содержания сообщения или документа и предназначенный для защиты от несанкционированного изменения информации. Для построения CRC-кода используется алгоритм вычисления контрольной суммы, основанный на делении информационного многочлена на примитивный многочлен. Популярность CRC-кода обусловлена тем, что проверка контрольной суммы просто реализуема, легко анализи-

руется и позволяет обнаруживать ошибки, вызванные наличием шума в каналах передачи данных. Общая структурная схема устройства генератора CRC-кода приведена на рис. 1.26.

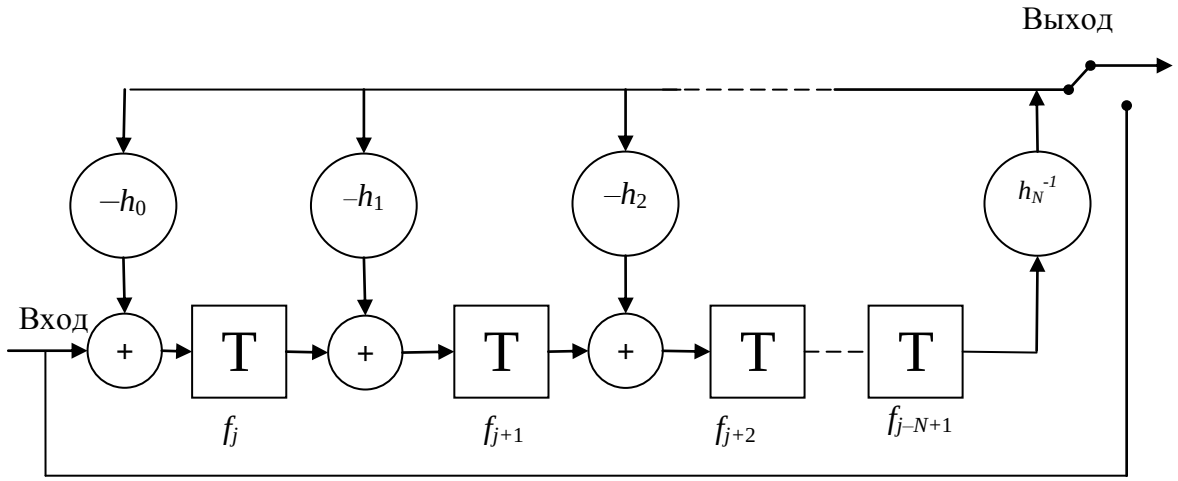

Рис. 1.26. Структурная схема генератора CRC-кода

Исходная информационная последовательность символов подается на вход устройства и эта же последовательность поступает на его выход (переключатель находится в нижнем положении). Далее переключатель соединяет верхнюю шину генератора CRC-кода с выходом устройства. Таким образом, происходит считывание CRC-кода, и каждый информационный блок дополняется CRC-кодом. Этот код представляет собой результат вычисления контрольной суммы (КС), основанный на делении многочленов.

Принцип вычисления КС основан на использовании свойств двоичного многочлена, в виде которого представляется исходная битовая последовательность блока данных. При этом каждый бит такой последовательности соответствует одному из полиномиальных коэффициентов. Например, десятичное число 90 (1011010 в двоичной записи) соответствует многочлену следующего вида:

$$P(x) = 1 \cdot x^6 + 0 \cdot x^5 + 1 \cdot x^4 + 1 \cdot x^3 + 0 \cdot x^2 + 1 \cdot x^1 + 0 \cdot x^0.$$

Подобным же образом в виде многочлена может быть представлен любой из блоков обрабатываемых данных.

При вычислении контрольного кода по методу КС используется свойство поведения многочленов, позволяющее выполнять с ними любые арифметические действия. Контрольный код рассчитывается, как остаток от деления по модулю 2 многочлена, полученного из исходной битовой последовательности на другой, заранее определённый многочлен (такой многочлен называется *порождающим* или *примитивным*):

$$R(x) = P(x)x^r \bmod G(x),$$

где $R(x)$ — контрольный код многочлена $P(x)$, $P(x)$ — исходный многочлен (информационный), $G(x)$ — порождающий многочлен, r — степень порождающего многочлена.

В качестве стандартных порождающих многочленов используются следующие многочлены.

CRC-32:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1.$$

CRC-24:

$$G(x) = x^{24} + x^{23} + x^{14} + x^{12} + x^8 + 1.$$

CRC-16:

$$G(x) = x^{16} + x^{15} + x^2 + 1.$$

CRC-8:

$$G(x) = x^8 + x^7 + x^6 + x^4 + x^2 + 1.$$

CRC-4:

$$G(x) = x^4 + x^3 + x^2 + x + 1.$$

1.4.6. Пример реализация поточного шифра

Структурная схема простейшей реализации алгоритма шифрования поточным шифром приведена на рис. 1.27. На вход устройства подается информационная последовательность символов. С выхода этого устройства снимается зашифрованная последовательность символов.

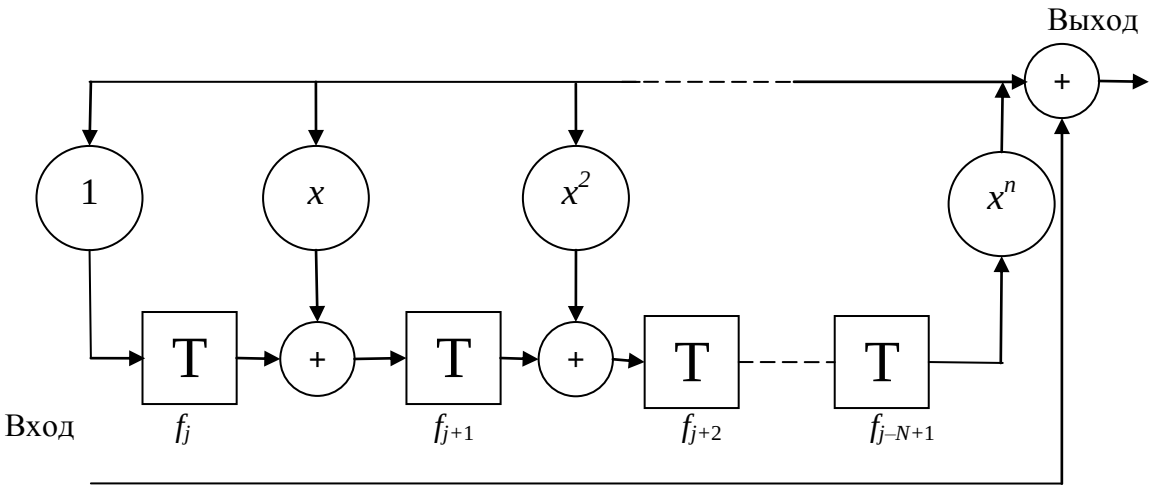

Рис. 1.27. Реализация поточного шифра

При шифровании каждый бит информационной последовательности складывается по модулю 2 со случайным битом с выхода генератора псевдослучайной последовательности. Качество шифрования оценивается методом сравнения энтропий информационного и зашифрованного текстов [4], поскольку после шифрования текст должен состоять из случайных наборов символов (байт) и обладать максимальной энтропией.

Энтропия исходного алфавита определяется выражением

$$H(A) = -\sum_{i=1}^{m} p_i \log_2 p_i ,$$

где A — множество букв алфавита, p_i — вероятность появления i-й буквы алфавита, а m — объем (количество букв) алфавита. Так, для русского языка можно считать, что $m = 32$, и, если считать, что буквы алфавита равновероятны, то $H(A) = 5$. Эту величину можно считать верхней оценкой значения энтропии для текстов, написанных на русском языке.

В действительности буквы русского языка не равновероятны. Например, буквы «о» и «а» встречаются в тексте чаще, чем буква «ф». Это обстоятельство уменьшает значение энтропии примерно до значения $H(A) = 4{,}35$. При записи информации в текстовый файл каждая

буква записывается в символьном виде, где каждый символ соответствует одному байту (8 бит). Если считать появление символов в последовательности равновероятным, то можно получить оценку для максимального значения энтропии $H(A) = 8$. При шифровании информации поточным шифром можно считать, что происходит суммирование по модулю 2 каждого бита информационного текста со случайной последовательностью нулей и единиц. Выходной шифрованный текст будет представлять собой также случайную последовательность нулей и единиц. При расчете энтропии такой последовательности по байтам получим значение $H(A) = 8$. При расчете энтропии этой же последовательности по битам значение $H(A) = 1$.

1.4.7. Синхронный потоковый шифр RC4

С помощью синхронного потокового шифра RC4 входная информация шифруется по блокам объемом по 256 бит. В этом шифре ключевая последовательность не зависит от исходного текста и представляет собой массив из 256 байт, которые принимают случайные значения в интервале от 0 до 255. Структура алгоритма включает блок замены S размерностью 8×256 бит: S_0, S_1, S_{255}. Каждый байт блока замены описывается целым числом, принимающим значения в интервале от 0 до 255. Таким образом, в блоке замен S осуществляется перестановка чисел 0, 1, …, 255 в зависимости от ключа переменной длины.

Алгоритм работы синхронного потокового шифра RC4 представлен на рис. 1.28. Для реализации этого алгоритма используются два счетчика I и J и ячейки S_t, S_i и S_j блока замен S (рис. 1.28).

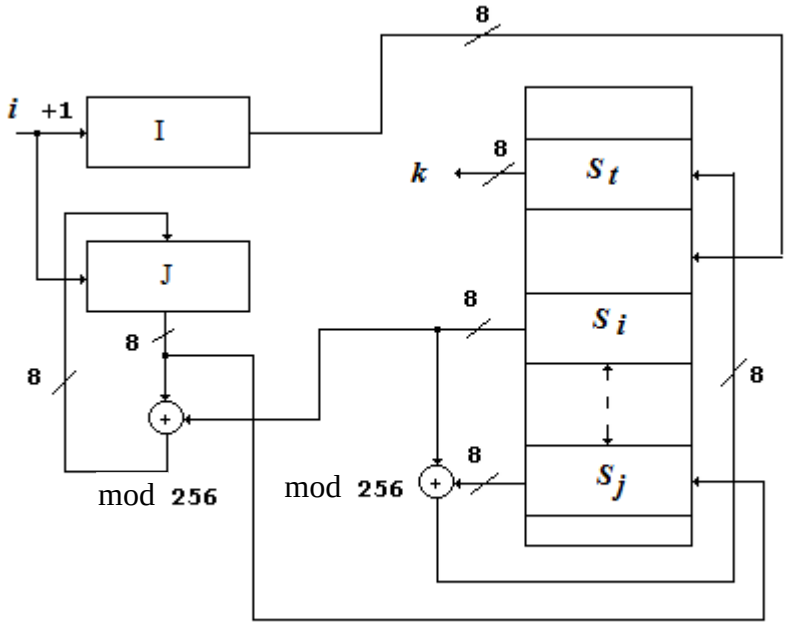

Рис. 1.28. Алгоритм работы синхронного потокового шифра RC4

На каждом такте работы первого счетчика I вычисляется выражение $i = (i + 1) \bmod 256$. На каждом такте второго счетчика J вычисляется выражение $j = (j + S_i) \bmod 256$. Затем данные, находящиеся в ячейках блока замен S_i и S_j, обмениваются между собой, и эта операция обозначается как $S_i \leftrightarrow S_j$. Далее вычисляется адрес ячейки блока замен $t = (S_i + S_j) \bmod 256$. Байт ключевой последовательности k представляет собой содержимое ячейки S_t блока замен $k = S_t$. Соответствующий байт шифрованного текста образуется сложением байта открытого текста p с байтом ключевой последовательности k:

$$c = (p + k) \bmod 256.$$

Перед началом выполнения алгоритма шифрования производится начальная инициализация блока замены. Ячейки блока замен заполняются данными линейно $S_i = i$. После начальной инициализации заполняется еще один массив, размером 256 байт, с помощью ключа k_i. Если длина ключа меньше 256, то ключ повторяется необходимое

число раз. Программа инициализации выполняет следующие действия.

- Инициализируется нулевым значением счетчик $j = 0$.
- Осуществляется перемешивание данных в блоке замен S согласно следующему выражению: для всех $i = 0, 1, 2, \ldots, 255$ вычисляется $j = (j + S_i + k_i) \bmod 256$, и ячейки блока замен S_i и S_j обмениваются своим содержимым.

Вопросы и задания

1. Сформулируйте определение криптосистем симметричных (с секретным ключом) и асимметричных (с открытым ключом).
2. На блоки какой длины разбивается открытый текст при использовании блочного шифра DES?
3. Сформулируйте определение поточного шифра.
4. Перечислите, какими свойствами обладают композиционные блочные шифры.
5. Нарисуйте блок схему конструкции Фейстеля.
6. Какому требованию должна удовлетворять функция криптографического преобразования в конструкции Фейстеля?
7. Какие свойства позволяют обеспечить рабочие режимы шифрования композиционных блочных шифров?
8. Каким требованиям должен удовлетворять генератор псевдослучайной последовательности, входящий в структуру поточного шифра?
9. Как осуществляется проверка качества генератора псевдослучайной последовательности по критерию хи-квадрат?
10. Нарисуйте структурные схемы, используемые для умножения и деления многочленов.
11. Напишите на языке высокого уровня программу шифрования по алгоритму шифра DES.

Без веры в то, что возможно охватить реальность нашими теоретическими построениями, без веры во внутреннюю гармонию нашего мира не могло быть никакой науки. Эта вера есть и всегда остается основными мотивами всякого научного творчества. **А.Эйнштейн и Л.Инфельд**

Эволюция физики (1938 г.)

2. Элементарные сведения о вычислениях в конечных полях

2.1. Элементы теории множеств

В телекоммуникационных системах связи для защиты передаваемой информации от несанкционированного доступа используются криптографические системы, алгоритм работы которых основан на вычислениях в конечных полях Галуа. В этой связи необходимо напомнить некоторые основные понятия теории множеств.

Множество считается заданным, если определено правило, по которому можно однозначно сказать принадлежит тот или иной элемент данному множеству или нет. Например, корректны определения: множество целых чисел, множество вещественных чисел, множество предметов в данной комнате, множество предметов красного цвета и т. д. Понятие множество — одно из наиболее абстрактных понятий в алгебре. Множество может быть конечным (состоять из конечного числа элементов), бесконечным (множество целых чисел) и даже пустым (не содержать ни одного элемента). Примером конечного множества является любой алфавит. Например, латинский алфавит, т. е. множество букв $\{a, \ldots, z\}$ является конечным.

Алгебраической структурой называется множество с заданными на нем действиями. Таким образом, структура — это набор $\{A, O_1, O_2, ..., O_n\}$, где A — множество, а $O_1, O_2, ..., O_n$ — заданные на нем действия, например, операция O_1 – сложение, а операция O_2 — умножение.

Если заданы два множества A и B, то вводится понятие отображения T. Отображение T считается заданным ($T: A \to B$), если задано правило, по которому для любого элемента a, принадлежащего A ($a \in A$), существует элемент b, принадлежащий B ($b \in B$). Понятие отображения более общее, чем понятие функции.

Отображение T называется биективным или взаимно-однозначным, если для любого a, принадлежащего A ($a \in A$), существует единственное b, принадлежащее B ($b \in B$), а также верно и обратное утверждение, т. е. для любого $b \in B$ существует единственное $a \in A$, т. е. задано обратное отображение T^{-1} ($T^{-1}: B \to A$).

Пусть на множестве G для каждой пары элементов a, b определена бинарная операция, которую, например, обозначим знаком «*». Это может быть как операция сложения, так и операция умножения, такая, что каждой паре элементов $a,b \in G$ ставится в соответствие элемент $c = a * b$. Такое множество называется *группой*, если операция «*» удовлетворяет следующим условиям.

- Замкнутость по отношению к операции «*», т. е. элемент $c = a * b$ принадлежит множеству G ($c \in G$).
- Ассоциативность $(a * b) * c = a * (b * c)$.
- Наличие единицы (нейтрального элемента), т. е. такого элемента, что $a * e = e * a = a$.
- Для любого элемента a, принадлежащего множеству G, существует обратный элемент a^{-1} такой, что $a^{-1} * a = e$. Если операция «*» является операцией умножения, то нейтральный элемент $e = 1$. Для операции сложения обратный элемент является отрицательным числом и нейтральный элемент $e = 0$ так, что $a + 0 = 0 + a = a$.

Если для любых двух элементов группы $a * b = b * a$, то группа называется *коммутативной* или *абелевой*. Если множество содержит конечное число элементов n, то группа называется *конечной* (группа порядка n), в противном случае группа называется *бесконечной*. Бесконечная группа может быть счетной (множество целых чисел) или несчетной (множество вещественных чисел).

Кольцо R представляет собой множество с двумя определенными на нем операциями, условно обозначаемыми как сложение «+» и умножение «*». В кольце должны выполняться следующие свойства.

- R является коммутативной группой по отношению к операции « + »; нейтральный элемент — 0; операции « + » и « * » в кольце неравноправны.
- В кольце R обеспечивается замкнутость по отношению к операции «*».
- Существует нейтральный элемент по отношению к операции «*», называемый и обозначаемый «1».
- В кольце R обеспечивается выполнение ассоциативности по отношению к операции «*», т. е. $a * (b * c) = (a * b) * c$.
- Обе операции должны обеспечивать выполнение свойства дистрибутивности:

$$a * (b + c) = ab + ac.$$

Рассмотрим, например, операцию вычисления по модулю целого числа p (остаток от деления на p). Такая операция преобразует множество целых чисел расположенных на числовой оси в порядке возрастания (..., –99, –98, ..., –2, –1, 0, 1, 2, ..., 98, 99, …) в кольцо 0, 1, …, p – 1. В этом кольце определены операции сложения, вычитания по модулю p и умножения по модулю p.

Пусть заданы числа $p = 3$, $a = 5$, $b = 7$. Эти числа преобразуются в остаток от деления на p, т. е. $a = 2$, $b = 1$. Например, $c = a + b = 2 + 1 = 3$. Вычисляя остаток после деления на p, получим результат $c = 0$. Если результат получается отрицательным, то к нему необходимо прибавить число p столько раз, сколько потребуется для

получения положительного результата операции. Этот результат попадет в кольцо 0, 1, …, $p - 1$. Аналогичным способом в кольце выполняется и операция умножения.

Относительно обычных операций сложения и умножения множество целых чисел образует кольцо. В кольце целых чисел сложение и умножение возможно всегда, а деление не всегда. Целое число s делится на целое число r (r делит s), если $s = r * a$. Этот факт записывается символом $r\backslash s$.

Положительное число $p > 1$, которое делится только на $+p, -p,$ 1, –1 называется простым. Простых чисел бесконечно много. Доказательство этого факта дал еще Евклид.

Действительно, если предположить обратное, то существует наибольшее простое число. Тогда, если к произведению всех простых чисел добавить единицу, получим число, превышающее наибольшее простое число, которое не будет делиться ни на одно из простых чисел. Противоречие и доказывает неправильность предположения.

Если для большого натурального числа n количество простых чисел обозначить, через N то можно показать, что имеется асимптотическое равенство $N = n/\ln(n)$.

На практике представляют интерес простые числа вида $2^m - 1$ и $2^{2^k} + 1$. Числа вида $2^m - 1$ для простого числа m называются *числами Мерсенна*. Числа вида $2^{2^k} + 1$ называются *числами Ферма*. На сегодняшний день известны следующие числа Ферма: 3, 5, 17, 257, 65537.

Наименьшими простыми числами являются 2, 3, 5, 7, 11, 13, ... Число 1 не является простым. Не являющееся простым число называется *составным*. Простые числа играют важную роль при формировании криптосистем с открытым ключом.

Наибольший общий делитель двух положительных целых чисел r и s обозначается НОД[r, s] и определяется как наибольшее положительное число, которое делит оба из них. Если НОД[r, s] = 1, то такие числа называются *взаимно простыми*. В дальнейшем для краткости будем использовать обозначение (r, s) = 1. Наименьшее общее крат-

ное двух положительных целых чисел r и s обозначается НОК[r, s] и определяется как наименьшее положительное число, которое делится на оба из них. Для каждого целого числа c и положительного числа d найдется единственная пара целых чисел Q (частное) и s (остаток), таких, что $c = d*Q + s$. Частное обозначается $Q = [c/d]$, остаток $s = c(\mathrm{mod}\ d)$. Вычисление остатка от сложного выражения облегчается выполнением следующих выражений:

$$(a + b)\ (\mathrm{mod}\ d) = ((a \bmod d) + (b \bmod d)) \bmod d;$$

$$(a*b)\ (\mathrm{mod}\ d) = ((a \bmod d) * (b \bmod d)) \bmod d.$$

Наибольший общий делитель двух заданных положительных чисел s и t может быть вычислен с помощью итеративного применения алгоритма деления. Эта процедура известна как алгоритм Евклида. Удобнее всего *алгоритм Евклида* записывается в матричном виде.

Введем в рассмотрение матрицу

$$A = \begin{pmatrix} 0 & 1 \\ 1 & -Q \end{pmatrix}.$$

Тогда алгоритм Евклида нахождения наибольшего делителя пары чисел (s, t), где $s > t$ сводится к последовательности шагов

$$\begin{pmatrix} s_1 \\ t_1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -Q_1 \end{pmatrix} \begin{pmatrix} s \\ t \end{pmatrix} = \begin{pmatrix} t \\ s - Q_1 t \end{pmatrix}, \qquad (2.1)$$

где остановка процесса наступает при получении нулевого остатка. Обозначим матрицу r-го шага преобразования как

$$A^r = \begin{pmatrix} 0 & 1 \\ 1 & -Q_r \end{pmatrix}.$$

Тогда окончательно имеем

$$s_n = \left(\prod_{r=1}^{n} A^r\right) \begin{pmatrix} s \\ t \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} s \\ t \end{pmatrix} = A_{11}s + A_{12}t;$$

$$\begin{pmatrix} s_2 \\ t_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -Q_2 \end{pmatrix} \begin{pmatrix} s_1 \\ t_1 \end{pmatrix};$$

$$\begin{pmatrix} s_{n-1} \\ t_{n-1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -Q_{n-1} \end{pmatrix} \begin{pmatrix} s_{n-2} \\ t_{n-2} \end{pmatrix}; \qquad (2.2)$$

$$\begin{pmatrix} s_n \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -Q_n \end{pmatrix} \begin{pmatrix} s_{n-1} \\ t_{n-1} \end{pmatrix}.$$

2.2. Решение диофантова уравнения первой степени на основании алгоритма Евклида

Рассмотрим задачу целочисленного решения уравнения вида

$$ax - my = 1, \tag{2.3}$$

где наибольший общий делитель чисел $(a, m) = 1$, $a > 0$, $m > a > 0$. Для решения этой задачи можно использовать алгоритм Евклида. Поскольку согласно (2.2),

$$1 = s_n = -A_{11}m + A_{12}a\ , \tag{2.4}$$

имеем $y = -A_{11}$, и $x = A_{12}$.

2.3. Решение сравнения первой степени на основании алгоритма Евклида

Согласно теореме Ферма — Эйлера, если $(a, m) = 1$, т. е. числа a и m взаимно просты, то

$$\alpha^{\varphi(m)} = 1, \tag{2.5}$$

где $\varphi(m)$ — функция Эйлера натурального аргумента m, равна числу положительных целых чисел меньших m и взаимно простых с m.

Для функции Эйлера справедливы следующие соотношения:

$$\varphi(1) = 1.$$

Если число р простое, то

$$\varphi(p) = p - 1;$$

$$\varphi(p^r) = p^r (p - 1).$$

Если числа a и b взаимно просты, т. е. $(a, b) = 1$, то

$$\varphi(ab) = \varphi(a)\varphi(b). \tag{2.6}$$

Для произвольного n

$$\varphi(n) = n(1-1/p_1)\,(1-1/p_2)\,\ldots\,(1-1/p_k),$$

где $p_1, p_2, p_3, \ldots, p_k$ — все простые делители числа n.

Перейдем к решению сравнения первой степени, т. е. к решению уравнения

$$a = 1 \pmod{m}, \tag{2.7}$$

где $(a, m) = 1$. Из теоремы Ферма следует, что

$$a^{\varphi(m)} = 1,$$

следовательно, решения сравнения дается выражением

$$x = a^{\varphi(m)-1} \pmod{m}. \tag{2.8}$$

Однако нахождение функции $\varphi(m)$ — это самостоятельная и достаточно сложная задача, если число m не простое. В этом случае для решения сравнения первой степени используется можно использовать алгоритм Евклида. В этом случае решается эквивалентное диофантово уравнение вида

$$ax - my = 1, \tag{2.9}$$

где

$$(a, m) = 1, a > 0, m > 0.$$

Если вычисления проводятся по mod m, то это эквивалентно решению сравнения

$$ax = 1 \pmod{m} \tag{2.10}$$

Следовательно, решение сравнения является решением эквивалентного диофантова уравнения, и $x = A_{12}$.

Вычисление по модулю большого простого числа часто используется в криптографии, поскольку задача вычисления логарифма в поле Галуа является достаточно сложной задачей, что и определяет криптографическую стойкость систем с открытым ключом.

2.4. Проверка чисел на простоту

Простое число делится только на самого себя и на единицу. Убедиться, что число простое можно, перебирая все числа в интервале от 2 до $N - 1$. Существуют, однако, более эффективные способы убедиться, что число N простое. Согласно малой теореме Ферма, если число N простое, то для любого целого $0 < a < N$, $(a, N) = 1$, выполняется соотношение

$$a^{N-1} = 1 \pmod{N}. \qquad (2.11)$$

Если это сравнение не выполняется, то число N составное. Проверку условия необходимо выполнять для всех a.

К сожалению, имеются составные числа, для которых это условие выполняется. Такие числа называются *числами Кармайкла*. Доказано, что любое число Кармайкла представимо в виде $N = p_1 p_2 \ldots p_r$, $r > 3$, где все числа p_i простые и различны, причем $N - 1$ делится на каждую разность $p_i - 1$.

Из-за существования чисел Кармайкла была предложена модификация теста, называемая тестом Миллера — Рабина, которая обходит проблему составных чисел. Если N — простое число, $N - 1 = 2^s t$, где t нечетно, то согласно малой теореме Ферма для каждого a, такого, что $(a, N) = 1$, хотя бы одна из скобок в произведении

$$(a^t - 1)(a^t + 1)(a^{2t} + 1)\cdots(a^{t2^{s-1}} + 1) = a^N - 1 \qquad (2.12)$$

делится на N.

Пусть N — нечетное составное число, $N - 1 = 2^s t$, где t нечетно. Число a называется «хорошим» для N, если нарушается одно из условий:

1) N не делится на a;

2) $a^t = 1 \pmod{N}$ или существует целое k, $0 \le k < s$, такое, что $a^{t2^k} = -1 \pmod{N}$.

Тогда, очевидно, что для простого числа N не существует «хороших» чисел a.

Алгоритм, доказывающий, что число не простое.

- Выберем случайным образом число $0 < a < N$, $(a, N) = 1$, и проверим для этого числа условия 1) и 2), Тогда:
- Если хотя бы одно из этих условий нарушается, то число N составное;
- Если выполнены оба условия, то возвращаемся к шагу 1.

В [5] показано, что вероятность не определить, что число не простое после k повторений, не превосходит 4^{-k}, т. е. убывает достаточно быстро. Самое большое простое число состоит из 12 978 189 цифр.

2.5. Алгоритм быстрого возведения в степень

Для экономии времени при программной реализации вычисление выражения $a^x \pmod p$ целесообразно производить следующим образом:

- Показатель степени x записывается в виде

$$x = x_1 + 2\,x_2 + 4\,x_3 + 8\,x_4, + \ldots,$$

где величины x_i, $i = 1, 2, \ldots$ принадлежат полю $GF(2)$ (двоичное представление показателя степени). Тогда a^x запишется в виде

$$a^x = a^{x_1}(a^2)^{x_2}(a^4)^{x_3}\ldots .$$

Здесь каждое последующее основание является квадратом предыдущего.

- Процесс возведения в степень распадается на две параллельные ветви. Одна — последовательно вычисляет квадраты, другая — перемножает их в соответствии со значениями двоичных разрядов (рис. 2.1).

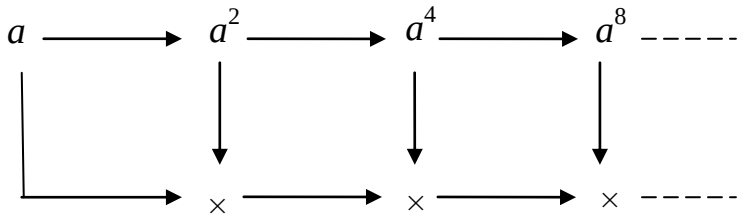

Рис. 2.1. Процесс возведения в степень

В среднем число операций, необходимое для реализации этого алгоритма, оценивается величиной 1,5 log p операций умножения (в показателе степени примерно половина нулей, следовательно, имеем

log p операций возведения в квадрат и 0,5 log p операций умножения).

2.6. Задача дискретного логарифмирования

Одним из первых способов логарифмирования является вероятностный алгоритм «встречи посередине». Логарифмирование по этому методу осуществляется следующим образом.

Для вычисления числа x, такого, что

$$a^x = b \pmod p, \tag{2.13}$$

строится база данных вида $a^y \pmod p$ для случайных чисел y, которая сортируется. Затем для случайных чисел z таких, чтобы НОД $(z, p-1) = 1$, вычисляется $b^z \pmod p$, и сравниваются результаты с базой данных. Поскольку поле конечное, после некоторого числа попыток получим

$$a^y = b^z \pmod p. \tag{2.14}$$

Тогда, возводя обе части равенства в степень $z^{-1} \pmod{p-1}$, получим

$$a^{y/z} = b \pmod p,$$

т. е.

$$x = y/z \pmod{p-1}. \tag{2.15}$$

Известен ускоренный вариант этого метода. Для вычисления логарифма число $p-1$ раскладывается на простые множители.

$$p - 1 = \prod_i p_i,$$

Тогда задача вычисления логарифма решается на циклических группах 0, 1, ..., $p_i - 1$ методом «встречи посередине», а значение x восстанавливается по китайской теореме об остатках.

2.7. Китайская теорема об остатках

Любое неотрицательное целое число, не происходящее произведение модулей, можно однозначно восстановить, если известны его вычеты по этим модулям. Этот результат был известен еще в древнем Китае и носит название китайской теоремы об остатках.

Теорема. *Для заданного множества целых положительных* m_0, m_1, m_2, ..., m_k *попарно взаимно простых чисел и множества неотрицательных целых чисел* c_0, c_1, c_2, ..., c_k, *где* $c_i < m_i$, *система имеет не более одного решения в интервале* $0 < c < \prod_{t=0}^{k} c_i$.

Каждое c_i рассчитывается по формуле $c_i = c(\mathrm{mod}\ m_i)$, где m_i — взаимно простые числа. Восстановление числа с осуществляется, согласно выражению:

$$c = \sum_{i=0}^{k} c_i N_i M_i (\mathrm{mod}\, M), \tag{2.16}$$

где

$$M = \prod_{i=0}^{k} m_i;\ M_i = M/m_i.$$

Числа M_i и N_i являются решениями уравнения

$$M_i N_i + m_i n_i = 1.$$

Здесь величины N_i и n_i определяются в соответствии с алгоритмом Евклида. Согласно китайской теореме об остатках, вычисления можно производить по множеству малых попарно взаимно простых модулей, а затем восстанавливать правильный результат в диапазоне от 0 до M.

2.8. Поле Галуа

Поле является математической структурой, в которой можно не только «складывать», «вычитать» и «умножать», но и «делить». Примерами таких полей являются поле комплексных чисел, поле вещественных чисел. Рассмотрим более подробно действия в конечном поле целых чисел. Можно показать, что имеются поля с конечным числом элементов. *Порядком поля* называется число его элементов. Поле с p элементами называется конечным полем или полем Галуа и обозначается $GF(p)$. Можно показать, что, кольцо вычетов по модулю простого числа p является полем.

Таблица 2.1

Таблица сложения в поле *GF*(5)

Операция сложения « + »	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Пример**,** в котором рассмотрим поле *GF*(5), операции сложения и умножения определяются табл. 2.1 и 2.2. В табл. 2.1 верхняя строка является первым слагаемым, крайний левый столбец **—** вторым слагаемым, а результат операции сложения считывается из таблицы на пересечении строки и столбца. Аналогично в табл. 2.2 верхняя строка является первым сомножителем, крайний левый столбец —вторым сомножителем, а результат операции умножения также считывается из таблицы на пересечении строки и столбца.

Таблица 2.2

Таблица умножения в поле *GF*(5)

Операция умножения «*»	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Рассмотрим операцию деления в этом поле. Так, (1/3)*3 = 3^{-1} *3 = 1, из таблицы умножения видно, что 3*2 = 1, следовательно,

$1/3 = 3^{-1} = 2$. Таким образом, элемент 2 является обратным для элемента 3. Каждый ненулевой элемент поля имеет обратный. Отметим, что, например,

$$2*2*2*2 = 1 \pmod 5,$$

т. е.

$$2^4 = 1.$$

Кроме того,

$$3*3*3*3 = 1 \pmod 5,$$

т. е.

$$3^4 = 1 \text{ и } 4*4*4*4 = 1 \pmod 5,$$

Другими словами

$$4^4 = 1.$$

Любой ненулевой элемент a поля $GF(5)$ в степени $p - 1$ равен единице $a^{p-1} = 1$.

Наименьшее поле $GF(2)$ содержит два элемента 0 и 1. Операции сложения и умножения определяются табл. 2.3 и 2.4.

Таблица 2.3

Таблица сложения в поле *GF*(2)

Операция сложения « + »	0	1
0	0	1
1	1	0

Таблица 2.4

Таблица умножения в поле *GF*(2)

Операция умножения «*»	0	1
0	0	0
1	0	1

2.9. Мультипликативная структура конечных полей

Если конечное поле содержит элемент α то оно должно содержать и все его степени, α, α^2, α^3, ... Так, если поле содержит мультипликативный, обратный каждому ненулевой элемент, то ему принадлежат также α^{-1}, α^{-2}, α^{-3}, ... Наименьшее из положительных чисел n, для которых $\alpha^n = 1$, называется порядком элемента α. Если порядок элемента равен n, то все элементы 1, α, α^2, ..., α^{n-1} различны. В этом случае элемент α является примитивным корнем n-й степени из единицы. При условии, что конечное поле содержит q элементов, элемент α называется примитивным элементом поля, если порядок элемента α равен $q - 1$**.** Арифметика конечных полей широко используется в криптографии. Многие криптосистемы основаны на вычислении в полях $GF(p)$, где p — большое простое число. Чтобы увеличить криптографическую стойкость шифра переходят к арифметике полей многочленов степени n, коэффициенты которых принадлежат конечному полю $GF(p^n)$.

2.10. Кольца многочленов

Для каждого поля F имеется кольцо $F(x)$**,** называемое *кольцом многочленов* на F. Многочленом над полем F называется выражение вида

$$f(x) = f_n x^n + f_{n-1} x^{n-1} + ... + f_1 x + f_0 = \sum_{i=0}^{n} f_i x^i, \qquad (2.17)$$

где символ x называется неопределенной переменной, а коэффициенты f_i принадлежат полю F. *Степень многочлена* обозначается $\deg f(x)$ и определяется как индекс старшего ненулевого коэффициента. *Приведенным многочленом* называется многочлен, старший коэффициент которого f_n равен единице. Два многочлена равны, если равны все их коэффициенты.

В кольце многочленов над заданным полем сложение и умножение определяются как обычные операции сложения и умножения многочленов:

$$f(x)+g(x)=\sum_i (f_i+g_i)x^i;$$
$$f(x)g(x)=\sum_i\sum_j f_i g_{j-i}x^i. \qquad (2.18)$$

Степень суммы определяется как наибольшая степень слагаемых. Степень произведения равна сумме степеней сомножителей. В кольце многочленов сложение, вычитание и умножение возможно всегда, а деление — не всегда. Многочлен $s(x)$ делится на многочлен $r(x)$, или $r(x)$ является делителем $s(x)$, если $s(x) = r(x)a(x)$. Ненулевой многочлен $p(x)$, делящийся только на $p(x)$ и на произвольный элемент α, поля F называется *неприводимым многочленом*. Приведенный неприводимый многочлен называется *простым многочленом*.

Например, многочлен $x^4 - 2$ является простым над полем рациональных чисел. Над полем вещественных чисел этот многочлен распадается на произведение $(x^2 - \sqrt{2})(x^2 + \sqrt{2})$. Над полем комплексных чисел каждый из сомножителей также не является простым.

Наибольший общий делитель двух многочленов $r(x)$ и $s(x)$ обозначается НОД[$r(x)$, $s(x)$] и определяется как приведенный многочлен наибольшей степени, делящийся оба этих многочлена. Если наибольший общий делитель двух многочленов равен 1, то они называются *взаимно простыми.*

Наименьшее общее кратное двух многочленов $r(x)$ и $s(x)$ обозначается НОК[$r(x)$, $s(x)$] и определяется как приведенный многочлен наименьшей степени, делящийся на оба из них.

Рассмотрим алгоритм деления для многочленов. Для каждого многочлена $c(x)$ и ненулевого многочлена $d(x)$ существует единственная пара многочленов $Q(x)$ (частное) и $s(x)$ (остаток), таких что

$$c(x) = Q(x)d(x) + s(x)$$

и

$$\deg s(x) < \deg d(x) < \deg c(x).$$

Практическое вычисление частного и остатка выполняется с помощью деления «уголком». Частное обозначается

$$Q(x) = |c(x)/d(x)|,$$

остаток

$$s(x) = c(x) \pmod{d(x)}.$$

Для суммы и произведения многочленов имеются следующие выражения:

$$[a(x) + b(x)] \pmod{d(x)} = [a(x) \pmod{d(x)} + b(x) \pmod{d(x)} \pmod{d(x)};$$

$$[a(x)b(x)] \pmod{d(x)} = [(a(x) \pmod{d(x)}) (b(x) \pmod{d(x)})] \pmod{d(x)}.$$

Рассмотрим более подробно алгоритм деления многочленов, поскольку из него вытекает важное следствие, известное под названием алгоритма Евклида для многочленов. Алгоритм деления «уголком» состоит из последовательности шагов:

$$\begin{gathered} s(x) = Q^{(1)}(x)t(x) + t^{(1)}(x); \\ t(x) = Q^{(2)}(x)t^{(1)}(x) + t^{(2)}(x); \\ t^{(1)}(x) = Q^{(3)}(x)t^{(2)}(x) + t^{(3)}(x); \\ \ldots \\ t^{(n-2)}(x) = Q^{(n)}(x)t^{(n-1)}(x) + t^{(n)}(x); \\ t^{(n-1)}(x) = Q^{(n+1)}(x)t^{(n)}(x), \end{gathered} \qquad (2.19)$$

где остановка процесса наступает при получении нулевого остатка. Можно вычислить значение многочлена $p(x)$ над полем F в любом элементе β этого поля. Для этого вместо неопределенной переменной x нужно подставить соответствующий элемент β. Элемент β называется корнем многочлена $p(x)$**,** если $p(\beta) = 0$. Легко показать, что в этом случае $(x - \beta)$ является делителем многочлена $p(x)$, т. е. $p(x) = (x - \beta)Q(x)$**.** Если степень многочлена равна n, то многочлен имеет в поле не более n корней.

Аналогично предыдущему алгоритм деления многочленов можно записать в матричном виде. Введем в рассмотрение матрицу

$$A(x) = \begin{pmatrix} 0 & 1 \\ 1 & -Q(x) \end{pmatrix}.$$

Тогда алгоритм Евклида нахождения наибольшего делителя пары чисел (s, t), где $s > t$ сводится к последовательности шагов

$$\begin{pmatrix} s_1(x) \\ t_1(x) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -Q_1(x) \end{pmatrix} \begin{pmatrix} s(x) \\ t(x) \end{pmatrix} = \begin{pmatrix} t(x) \\ s(x) - Q_1(x)t(x) \end{pmatrix};$$

$$\begin{pmatrix} s_2(x) \\ t_2(x) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -Q_2(x) \end{pmatrix} \begin{pmatrix} s_1(x) \\ t_1(x) \end{pmatrix};$$

$$\ldots \qquad (2.20)$$

$$\begin{pmatrix} s_{n-1}(x) \\ t_{n-1}(x) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -Q_{n-1}(x) \end{pmatrix} \begin{pmatrix} s_{n-2}(x) \\ t_{n-2}(x) \end{pmatrix};$$

$$\begin{pmatrix} s_n(x) \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -Q_n(x) \end{pmatrix} \begin{pmatrix} s_{n-1}(x) \\ t_{n-1}(x) \end{pmatrix},$$

где остановка процесса наступает при получении нулевого остатка.

Обозначим матрицу r-го шага преобразования через

$$A^r(x) = \begin{pmatrix} 0 & 1 \\ 1 & -Q_r(x) \end{pmatrix}.$$

Тогда окончательно

$$s_n(x) = \prod_{r=1}^{n} A^r(x) \begin{pmatrix} s(x) \\ t(x) \end{pmatrix} = \begin{pmatrix} A_{11}(x) & A_{12}(x) \\ A_{21}(x) & A_{22}(x) \end{pmatrix} \begin{pmatrix} s \\ t \end{pmatrix} = A_{11}(x)s(x) + A_{12}(x)t(x). \quad (2.21)$$

2.11. Классы вычетов в полях многочленов

Два многочлена *взаимно просты*, если их наибольший общий делитель равен единице. Классы вычетов многочленов по модулю $p(x)$ многочлена степени n, коэффициенты которого принадлежат полю $GF(p)$, образуют кольцо с конечным числом элементов. Класс вычетов многочленов по модулю $p(x)$ неприводимого многочлена степени n образуют поле Галуа $GF(p^n)$ с конечным числом элементов p^n в случае, если $p(x)$ нельзя представить в виде произведения многочленов с коэффициентами из поля $GF(p)$.

Совокупность корней многочлена является $x^{q-1} - 1$ совокупностью всех ненулевых элементов поля $GF(q)$. Каждый класс вычетов по модулю многочлена $f(x)$ степени n содержит либо 0, либо многочлен степени меньшей n. Пусть $p(x)$ — многочлен с коэффициентами

из поля F. Алгебра многочленов над полем F по модулю $p(x)$ является полем тогда, когда многочлен $p(x)$ неприводим, т. е. если $p(x)$ нельзя представить в виде произведения многочленов с коэффициентами из поля F. Например, многочлен $p(x) = x^2 + 1$ неприводим над полем вещественных чисел. Поле, образованное многочленами над полем F по модулю $p(x)$ степени k, называется *расширением поля степени k над F*. Первоначальное поле F называется *основным полем*. Поле многочленов над $GF(p)$ по модулю неприводимого многочлена степени m образуют расширение поля Галуа $GF(p^m)$ и содержит $q = p^m$ элементов. Таким образом, для любого числа q, которое является степенью простого числа, существует поле $GF(q)$ из q элементов. Можно показать [9], что любое конечное поле изоморфно некоторому полю Галуа, и отличается только выбором названий для элементов. В поле $GF(p)$ элемент $p = 0$.

2.12. Мультипликативная группа в поле многочленов

Пусть $p(x)$ — многочлен степени m с коэффициентами из поля $GF(p)$, который неприводим в этом поле, и пусть α — корень в расширении поля. Тогда $\alpha, \alpha^2, ..., \alpha^{p^m-1}$ образуют совокупность всех корней многочлена $p(x)$.

Например, поле Галуа $GF(2^4)$ из 2^4 элементов может быть образовано как поле многочленов над $GF(2)$ по модулю $x^4 + x + 1$. Пусть α обозначает класс вычетов, который содержит x. Тогда α является корнем многочлена $x^4 + x + 1$ и примитивным элементом поля. Для этого случая 15 ненулевых элементов поля приведены в табл. 2.5.

Неприводимый многочлен степени m над полем называется примитивным, если его корнем является примитивный элемент поля $GF(q^m)$.

Таблица 2.5

Форма представления элементов поля $GF(2^4)$

Степенная форма	Полиномиальная форма	Векторная форма
α^0	1	(0001)
α^1	x	(0010)
α^2	x^2	(0100)
α^3	x^3	(1000)
α^4	$x+1$	(0011)
α^5	x^2+x	(0110)
α^6	x^3+x^2	(1100)
α^7	x^3+x+1	(1011)
α^8	x^2+1	(0101)
α^9	x^3+x	(1010)
α^{10}	x^2+x+1	(0111)
α^{11}	x^3+x^2+x	(1110)
α^{12}	x^3+x^2+x+1	(1111)
α^{13}	x^3+x^2+1	(1101)
α^{14}	x^3+1	(1001)
α^{15}	1	(0001)

2.13. Китайская теорема об остатках для многочленов

В кольце многочленов над некоторым полем также имеется китайская теорема об остатках.

Теорема. *Для заданного множества взаимно простых многочленов* $m_0(x)$, $m_1(x)$, $m_2(x)$, ..., $m_k(x)$ *и множества многочленов* $c_0(x)$, $c_1(x)$, $c_2(x)$, ..., $c_k(x)$ *система сравнений*

$$c_i(x) = c(x) \pmod{m_i(x)},\ i = 0, 1, ..., k$$

имеет единственное решение

$$c(x) = \sum_{i=0}^{k} c_i(x) N_i(x) M_i(x), \bmod M(x),$$

где

$$M(x) = \prod_{i=0}^{k} m_i \; ; \; M_i(x) = M(x) / m_i(x);$$

а $M_i(x)$ и $N_i(x)$ являются решениями уравнения

$$M_i(x)N_i(x) + m_i(x)n_i(x) = 1.$$

Рассмотрим конечные поля Галуа образованные

- вычетами по модулю простого числа;
- остатками от деления по модулю неприводимого многочлена.

Такие конечные поля лежат в основе криптографических систем с открытым ключом, и знание их алгебраической структуры необходимо для понимания алгоритмов шифрования информации.

Возьмем два целых числа a и b, и пусть, например, $a = 4$, $b = 7$. При обычном сложении результат $c = a + b = 11$. Покажем, что тот же результат может быть получен при вычислении в конечных полях и восстановлении результата вычислений по китайской теореме об остатках.

Пусть выбран набор из двух взаимно простых чисел $m_0 = 3$, $m_1 = 5$, тогда $M = m_0 m_1 = 15$ и $M_0 = 5$, $M_1 = 3$. Тогда $a_0 = 1$, $a_1 = 4$, $b_0 = 2$, $b_1 = 2$, $c_0 = 2$, $c_1 = 1$. Вычисляя, согласно алгоритму Евклида имеем:

$$1 = 5N_0 + 3n_0;$$
$$5 = 1 \cdot 3 + 2;$$
$$3 = 1 \cdot 2 + 1;$$
$$2 = 1 \cdot 2 + 0;$$
$$N_0 = -1;$$
$$1 = 3N_1 + 5n_1;$$
$$N_1 = 2;$$
$$c = c_0 N_0 M_0 + c_1 N_1 M_1 = 2 \cdot (-1) \cdot 5 + 1 \cdot 2 \cdot 3 = -4 = 11 \pmod{15}.$$

Рассмотрим другой случай. Возьмем три целых числа a, b и d и пусть, например, $a = 4$, $b = 7$, $d = 5$. При обычном сложении результат $c = a + b + d = 16$. Покажем, что тот же результат может быть полу-

чен при вычислении в конечных полях и восстановлении результата вычислений по китайской теореме об остатках.

Пусть выбран набор из трех взаимно простых $m_0 = 3$, $m_1 = 4$, $m_2 = 5$ и $M = m_0 m_1 m_2 = 60$, тогда $M_0 = 20$, $M_1 = 15$, $M_2 = 12$. Следовательно $a_0 = 1$, $a_1 = 0$, $a_2 = 4$, $b_0 = 2$, $b_1 = 3$, $b_2 = 2$, $d_0 = 2$, $d_1 = 1$, $d_2 = 0$, $c_0 = 4$, $c_1 = 4$, $c_2 = 6$. Вычисляя, согласно алгоритму Евклида, имеем:

$$1 = 20N_0 + 3n_0;$$
$$20 = 6 \cdot 3 + 2;$$
$$3 = 1 \cdot 2 + 1;$$
$$2 = 1 \cdot 2 + 0;$$
$$1 = 3 - 2 = 3 - (20 - 6 \cdot 3) = -1 \cdot 20 + 7 \cdot 3;$$
$$N_0 = -1;$$
$$1 = 15N_1 + 4n_1;$$
$$15 = 3 \cdot 4 + 3;$$
$$4 = 1 \cdot 3 + 1;$$
$$3 = 3 \cdot 1 + 0;$$
$$1 = 4 - 3 = 4 - (15 - 3 \cdot 4) = -1 \cdot 15 + 4 \cdot 4;$$
$$N_1 = -1;$$
$$1 = 12N_2 + 5n_2;$$
$$12 = 2 \cdot 5 + 2;$$
$$5 = 2 \cdot 2 + 1;$$
$$2 = 2 \cdot 1 + 0;$$
$$1 = 5 - 2 \cdot 2 = 5 - 2 \cdot (12 - 2 \cdot 5) = -2 \cdot 12 + 5 \cdot 5;$$
$$N_2 = -2;$$
$$c = c_0 N_0 M_0 + c_1 N_1 M_1 + c_2 N_2 M_2 =$$
$$= 4 \cdot (-1) \cdot 20 + 4 \cdot (-1) \cdot 15 + 6 \cdot (-2) \cdot 12 = -284 = 16 \pmod{60}.$$

Таким образом, если известны значения вычетов в кольцах по малым модулям m_0, m_1, m_2, ..., то можно однозначно восстановить исходное число.

Вопросы и задания

1. Каким свойствам на множестве G должна бинарная операция, чтобы это множество называлось группой?
2. Каким свойствам на множестве R должны удовлетворять, определенные на нем операции, чтобы это множество называлось кольцом?
3. Каким свойствам на множестве F должны удовлетворять, определенные на нем операции, чтобы это множество называлось полем?
4. Дайте определение функции Эйлера.
5. Сформулируете идею решения задачи дискретного логарифмирования.
6. Дайте определение неприводимого многочлена.
7. Дайте определение простого многочлена.
8. Дайте определение кольца многочленов.
9. Как образуется поле многочленов $GF(2^n)$?
10. Сформулируйте китайскую теорему об остатках.
11. Напишите программу, реализующую алгоритм Евклида.
12. Разработайте программу решения диофантова уравнения.
13. Напишите программу решения сравнения первой степени.
14. Составьте программу вычисления функции Эйлера.
15. Напишите программу генерации простых чисел.
16. Разработайте программу решения задачи дискретного логарифмирования.
17. Напишите программу восстановления числа по китайской теореме об остатках.

Я пришел к вам как юридическое лицо к юридическому лицу. Вот пачка весом в три–четыре кило. Она продается и стоит миллион рублей, тот самый миллион, который вы из жадности не хотите мне подарить.

И.Ильф и Е.Петров

Золотой теленок

3. Криптографические системы с открытым ключом

В [10] рассмотрен принцип построения криптосистем, не требующий не только передачи ключа, но даже сохранения в тайне метода шифрования. Этот метод основан на вычислениях в поле Галуа. Такие шифры позволяют легко шифровать и дешифрировать текст, и их можно использовать многократно.

3.1. Криптографическая система без передачи ключей

Пусть абонентам A и B необходимо передать информацию по открытому каналу передачи (рис. 1.1). Для этой цели они выбирают достаточно большое простое число p, такое, что $p - 1$ разлагается на не очень большие простые множители. Если среди множителей такого числа нет кратных, то число $p - 1$ называется *евклидовым*. Каждый из абонентов независимо один от другого выбирает случайное число, натуральное, взаимно простое с числом $p - 1$. Пусть абоненты A и B случайным образом выбирают числа a, b принадлежащие полю $GF(p)$. После этого осуществляются следующие действия.

- Абонент A находит второе число α из условия

$$a\alpha = 1 \pmod{\varphi(p)}, \quad 0 < \alpha < p - 1,$$

причем $\varphi(p) = p - 1$.

- Абонент *B* находит второе число β из условия

$$b\beta = 1 (\mathrm{mod}\ \varphi(p)),\ 0 < \beta < p - 1,$$

причем $\varphi(p) = p - 1$.

Пусть абонент *A* посылает сообщение *m* абоненту *B*. Без ограничения общности, будем считать, что $0 < m < p - 1$. Процесс передачи осуществляется в несколько этапов.

- Шифруется сообщение *m* первым ключом, и находится зашифрованное сообщение m_1 согласно выражению:

$$m_1 = m^a \ (\mathrm{mod}\ p),\ 0 < m_1 < p.$$

- Сообщение m_1 отправляется абоненту *B*.
- Абонент *B*, в свою очередь, зашифровывает вновь это сообщение ключом *b* так, что

$$m_2 = m_1^{\,b} \ (\mathrm{mod}\ p),\ 0 < m_2 < p$$

и пересылает его обратно абоненту *A*.

- Абонент *A*, получив обратно свое дважды зашифрованное сообщение, шифрует его же в третий раз своим вторым ключом α согласно выражению

$$m_3 = m_2^{\,\alpha} \ (\mathrm{mod}\ p),\ 0 < m_3 < p$$

и вновь отправляет его абоненту *B*.

- Абонент *B* расшифровывает это сообщение при помощи второго ключа:

$$m_4 = m_3^{\,\beta} \ (\mathrm{mod}\ p),\ 0 < m_4 < p.$$

Докажем, что $m_4 = m$. В самом деле, из выражений для m_1, m_2, m_3 и m_4 имеем

$$m_4 = m^k \ (\mathrm{mod}\ p),$$

где

$$k = ab\alpha\beta \ (\mathrm{mod}\ (p - 1))$$

или

$$k = 1 \ (\mathrm{mod}\ \varphi\,(p)).$$

Поэтому $m_4 = m \pmod p$, а, так как каждое из них положительное и меньше p, $m_4 = m$.

3.2. Алгоритмы шифрования, построенные на основе задачи об укладке ранца

Ранцевый алгоритм, разработанный Ральфом Мерклом и Мартином Хеллманом, стал первым алгоритмом шифрования с открытым ключом широкого назначения. Проблема укладки ранца формулируется просто. Дано множество предметов с разными весами и спрашивается, можно ли положить некоторые из этих предметов в ранец так, чтобы его вес стал равен определенному значению? Более формально задача формулируется так: дан набор значений M_1, M_2, …, M_n и суммарное значение S, требуется вычислить значения b_i такие что

$$S = b_1 M_1 + b_2 M_2 + \ldots + b_n M_n.$$

Здесь b_i может быть либо нулем, либо единицей. Значение $b_i = 1$ означает, что предмет M_i кладут в рюкзак, а $b_i = 0$ — не кладут. В основе алгоритма, предложенного Мерклом и Хеллманом, лежит идея шифрования сообщения на основе решения серии задач по укладке ранца. Предметы из кучи выбираются с помощью блока открытого текста, длина которого (в битах) равна количеству коэффициентов b_i. При этом биты открытого текста соответствуют значениям b_i, а шифрованный текст является полученным суммарным весом. Пример текста, зашифрованного с помощью задачи укладки ранца, показан в табл. 3.1.

Таблица 3.1

Ранцевый алгоритм шифрования

Открытый текст	1 1 1 0 0 1	0 1 0 1 1 0	0 0 0 0 0 0	0 1 1 0 0 0
Последовательность весов	2 3 6 13 27 52	2 3 6 13 27 52	2 3 6 13 27 52	2 3 6 13 27 52
Шифрованный текст	2 + 3 + 6 + 52 = 63	3 + 13 + 2 7 = 43	0 = 0	3 + 6 = 9

Существуют две различные задачи укладки ранца; одна из них характеризуется линейным ростом трудоемкости, а другая – как принято считать, нет.

В первой задаче укладки ранца используется сверхвозрастающая последовательность весов. Сверхвозрастающей называется последовательность, в которой каждый член больше суммы всех предыдущих членов. Например, последовательность {1, 3, 6, 13, 27, 52} является сверхвозрастающей, а последовательность {1, 3, 4, 9, 15, 25} — нет. Рассмотрим решение первой задачи. Возьмем в качестве текущего полный вес, который надо получить, и сравним его с весом самого тяжелого предмета в куче. Если текущий вес меньше веса данного предмета, то предмет в ранец не кладут. Если текущий вес больше или равен весу этого предмета, то он кладется в ранец. Уменьшим текущий вес на вес положенного предмета и перейдем к следующему по весу предмету в последовательности. Будем повторять такие шаги, пока процесс не закончится. Если текущий вес уменьшится до нуля, то решение найдено. В противном случае, нет.

Например, пусть полный вес рюкзака равен 63, а последовательность весов предметов равна {2, 3, 6, 13, 27, 52}. Самый большой вес — 52, он меньше 63, поэтому предмет весом 52 кладется в рюкзак. Вычитаем 52 из 63 и получаем 11. Следующий наибольший вес в последовательности равен 27, он больше 11, поэтому предмет весом 27 в рюкзак не кладется. Следующий самый тяжелый предмет имеет вес 6, он меньше 11, поэтому предмет весом 6 также кладется в рюкзак. Вычитаем 6 из 11 и получаем 5. Следующий самый тяжелый предмет имеет вес 3, он меньше 5, поэтому предмет весом 3 в рюкзак кладется. Заканчивая этот процесс, покажем, что предмет с весом 2 кладется в рюкзак, и полный вес уменьшается до 0. Это означает, что решение найдено. Добавление одного предмета к последовательности увеличивает трудоемкость поиска решения лишь на одну операцию. Алгоритм Меркла — Хеллмана базируется на описанном выше свойстве. Если бы этот рюкзак был бы применен для

шифрования по алгоритму Меркла — Хеллмана, то открытый текст, полученный из значения шифрованного текста 63, был бы равен 111001. Последовательность весов при решении задачи укладки сверхвозрастающего ранца является закрытым ключом.

Несверхвозрастающие (нормальные) рюкзаки представляют собой вторую задачу, для решения которой быстрый алгоритм неизвестен. Единственным известным методом определения предметов, помещаемых в рюкзак, является полный перебор возможных решений, проводимый до нахождения правильной комбинации. Самый быстрый алгоритм, принимающий во внимание различные эмпирические правила, характеризуется экспоненциальной зависимостью от числа возможных предметов. При этом добавление к куче одного предмета удваивает трудоемкость поиска решения. Последовательность весов для задачи укладки нормального ранца является открытым ключом.

Меркл и Хеллман разработали способ преобразования первой задачи укладки сверхвозрастающего рюкзака во вторую задачу укладки нормального рюкзака. Для этого используется арифметика модульных операций. Создание открытого ключа по закрытому ключу осуществляется следующим образом. Для преобразования сверхвозрастающей последовательности для укладки рюкзака в нормальную возьмем сверхвозрастающую последовательность {2, 3, 6, 13, 27, 52} и умножим все значения в последовательности на число n по модулю m. Значение модуля m должно быть больше суммы всех чисел последовательности, например, 105. Множитель должен быть взаимно простым числом с модулем, например, 31. Нормальной последовательностью будет

$$2 \cdot 31 \bmod 105 = 62;$$
$$3 \cdot 31 \bmod 105 = 93;$$
$$6 \cdot 31 \bmod 105 = 81;$$
$$13 \cdot 31 \bmod 105 = 88;$$
$$27 \cdot 31 \bmod 105 = 102;$$
$$52 \cdot 31 \bmod 105 = 37.$$

Запись этой последовательности имеет вид {62, 93, 81, 88, 102, 37}. Для шифрования сообщение сначала разбивается на блоки, по размерам равные числу элементов последовательности для укладки рюкзака. Затем, считая, что единица указывает на присутствие элемента последовательности в рюкзаке, а ноль — на его отсутствие, вычисляются полные веса рюкзаков. Например, если сообщение в бинарном виде выглядит как 011000 110101 101110, то шифрованным текстом будет последовательность 174, 280, 333.

Дешифрирование происходит следующим образом. Законный абонент данного сообщения знает закрытый ключ, а именно — исходную сверхвозрастающую последовательность, а также значения n и m, использованные для превращения ее в нормальную последовательность. Для дешифрирования сообщения абонент должен сначала определить n^{-1}, такое что $n\ (n^{-1}) = 1 \pmod m$. Каждое значение шифрованного текста умножается на $n^{-1} \bmod m$, а затем разделяется на биты с помощью закрытого ключа, чтобы получить значения открытого текста. Например, если сверхвозрастающая последовательность равна {2, 3, 6, 13, 27, 52}, m равно 105, а n равно 31,то шифрованный текст имеет вид: 174, 280, 333. Следовательно, n^{-1} равно 61, поэтому значения шифрованного текста должны быть умножены на 61 по модулю 105. Тогда получим

$$174 \cdot 61 \bmod 105 = 9 = 3 + 6,$$

что соответствует 011000;

$$280 \cdot 61 \bmod 105 = 70 = 2 + 3 + 13 + 52,$$

что соответствует 110101;

$$333 \cdot 61 \bmod 105 = 48 = 2 + 6 + 13 + 27,$$

что соответствует 101110.

Следовательно, открытым текстом является битовая последовательность 011000 110101 101110.

3.3. Обмен сообщениями по алгоритму Диффи — Хелл-мана

Рассмотрим практическую реализацию обмена сообщениями с использованием вычислений в поле $GF(p)$, где p — простое число. Наиболее простой метод передать сообщение m другому абоненту, заключается в следующей последовательности шагов.

- Абонент A генерирует случайное число x, такое что $(x,\ p-1) = 1$, вычисляет $m_1 = m^x \pmod p$ и отправляет m_1 абоненту B.
- Абонент B генерирует случайное число y такое, что $(y,\ p-1) = 1$, вычисляет $m_2 = (m_1)^y \pmod p$ и отправляет m_2 абоненту A.
- Абонент A возводит полученное число в степень $z = x^{-1}$, вычисляет $m_3 = (m_2)^z \pmod p$ и отправляет m_3 абоненту B;
- Абонент B возводит полученное число в степень $r = y^{-1}$, вычисляет $m_4 = (m_3)^r \pmod p$ и получает m.

Действительно,

$$m_4 = (m_3)^r = ((m_2)^z)^r = ((m_1)^y)^z)^r) = ((m^x)^y)^z)^r) = m.$$

Этот алгоритм обычно используется для открытого распределения ключей в сеансе связи.

3.4. Обмен сообщениями по алгоритму Т. Эль-Гамаля

Алгоритм Т. Эль-Гамаля используется как для шифрования открытым ключом, так и для цифровой подписи. Алгоритм Т. Эль-Гамаля стал в США национальным стандартом цифровой подписи (DSS, Digital Signature Standard), затем с небольшими изменениями применен во всероссийском стандарте подписи ГОСТ Р43.10-94.

Пусть число p простое и α — примитивный корень степени $p-1$ из единицы являются общими для всей сети связи. Заметим, что в общем случае α может образовывать мультипликативную группу порядка $q < p$. Абонент A генерирует случайное число x_a, вычисляет

$d_a = \alpha^{x_a} (\mathrm{mod}\, p)$ и опубликовывает его. Аналогично, действует второй абонент *B*, получается следующая таблица ключей (табл. 3.2).

Таблица 3.2

Таблица ключей

Абонент	Открытый ключ	Закрытый ключ
A	d_a	x_a
B	d_b	x_b

Пусть абонент *A* посылает сообщение *m*, $0 < m < p$. Для этого абонент *A* осуществляет следующую последовательность действий.

- Генерирует случайное число *k*, вычисляет α^k (*mod p)*, затем формирует шифрованный текст, состоящий из пары чисел (c_1, c_2):

$$c_1 = \alpha^k (\mathrm{mod}\, p),\ c_2 = m(d_b)^k (\mathrm{mod}\, p)$$

и отправляет абоненту *B*.

- Для дешифрирования этого сообщения абонент *B* вычисляет

$$m_b = c_2 \cdot (c_1)^{p-1-x_b} (\mathrm{mod}\, p).$$

Покажем, что $m_b = m$. Имеем:

$$\begin{aligned} m_b &= m \cdot (d_b)^k \cdot (c_1)^{p-1-x_b} (\mathrm{mod}\, p) = \\ &= m \cdot (\alpha^{x_b})^k \cdot (\alpha^k)^{p-1-x_b} (\mathrm{mod}\, p) = \\ &= m \cdot (\alpha^k)^{p-1} = m \cdot (1^k) = m. \end{aligned}$$

3.5. Криптографическая система с открытым ключом RSA

Криптографическая система с открытым ключом RSA [2] основана на использовании того факта, что легко перемножить два больших простых числа, в то же время, крайне трудно разложить на множители их произведение. В результате произведение может быть использовано в качестве элемента открытого ключа шифрования.

Пусть абонентам A и B необходимо передать закрытые данные по открытому каналу передачи. Тогда каждый из них, независимо один от другого, выбирает два больших простых числа, находит их произведение и функцию Эйлера от этого произведения и выбирает случайное число, меньшее вычисленного значения функции Эйлера и взаимно простое с ним. Итак,

$$A: p_1, p_2, r_a = p_1 p_2, \varphi(r_a), (a, \varphi(r_a)) = 1, 0 < a < \varphi(r_a),$$

$$B: q_1, q_2, r_b = q_1 q_2, \varphi(r_b), (b, \varphi(r_b)) = 1, 0 < b < \varphi(r_b).$$

Затем абонентам A и B становится доступной таблица, которая имеет вид

$$A: r_a, a;$$

$$B: r_b, b,$$

где r_a — произведение двух простых чисел, которые известны только абоненту A; a — открытый ключ, доступный каждому, кто хочет передать сообщение абоненту A, r_b — произведение двух простых чисел, которые известны только абоненту B, b — открытый ключ, доступный каждому, кто хочет передать сообщение абоненту B.

Эта таблица может быть продолжена для любого конечного числа абонентов. Каждый из абонентов находит свой закрытый ключ путем решения сравнений

$$ax = 1 \pmod{\varphi(r_a)}; \; bx = 1 \pmod{\varphi(r_b)},$$

выбирая α и β из условий:

$$a\alpha = 1 \pmod{\varphi(r_a)}, \; 0 < \alpha < \varphi(r_a),$$

$$b\beta = 1 \pmod{\varphi(r_b)}, \; 0 < \beta < \varphi(r_b).$$

Полученные сведения приведены в табл. 3.3.

Таблица 3.3

Таблица ключей

Абонент	Открытые ключи	Секретные ключи
A	a, r_a	α
B	b, r_b	β

Рассмотрим подробнее процесс передачи сообщений. Пусть абонент A посылает сообщение $0 < m < r_b$, (при $m > r_b$ сообщение делят на отрезки, длиной менее r_b и шифруют отрезки отдельно друг за другом) абоненту B.

- Абонент A шифрует сообщение m открытым ключом абонента B, который есть в телефонной книге, вычисляет

$$m_1 = m^b \pmod{r_b},\ 0 < m_1 < r_b.$$

и посылает сообщение m_1 абоненту B.

- Абонент B дешифрирует сообщение m_1 своим секретным ключом, вычисляя

$$m_2 = m_1^{\beta} \pmod{r_b},\ 0 < m_2 < r_b.$$

Покажем, что $m_2 = m$. Действительно,

$$m_2 = m^{b\beta} \pmod{r_b}$$

и

$$b\beta = 1 \pmod{\varphi(r_b)},$$

следовательно,

$$m_2 = m \pmod{r_b},$$

но m_2 и m меньше r_b, тогда $m_2 = m$.

Криптографическая система RSA является частью многих стандартов, например стандарта X9.44. Рассмотрим возможность реализации этой системы на базе криптографической системы с открытым ключом. Криптографическая система с открытым ключом неудобна в том смысле, что получатель сообщения не знает, кто является отправителем сообщения. Этого недостатка лишена приведенная ниже система.

Пусть имеется абонент B и несколько абонентов W_1, W_2, W_3, ... Абонент B и каждый из абонентов W_i независимо друг от друга выбирают по два простых числа. Пусть P и Q — простые числа абонента B, p_i и q_i — простые числа абонентов W_i, $i = 1, 2, 3, \ldots$. Пусть абонент B произвольным образом выбирает целое число S, удовлетворяющее условиям $0 < S < \varphi(R)$, $(S, \varphi(R)) = 1$, а каждый из абонентов также случайно и независимо друг от друга выбирает число s_i удовлетво-

ряющее условиям $0 < s_i < \varphi(r_i)$, $(s_i, \varphi(r_i)) = 1$, $i = 1, 2, 3, \ldots$. После этого публикуется доступная всем таблица:

$$B : R, S;$$
$$W_1 : r_1, s_1;$$
$$W_2 : r_2, s_2;$$
$$\ldots$$
$$W_i : r_i, s_i.$$

Каждый из них — абонент B и абоненты W_i — находят свои секретные ключи T, t_i из условий

$$ST = 1 \pmod{\varphi(R)},\ 0 < T < \varphi(R),$$
$$s_i t_i = 1 \pmod{\varphi(r_i)},\ 0 < t_i < \varphi(r_i),\ i = 1, 2, 3, \ldots .$$

Предположим, абонент W_i собирается передать сообщение m абоненту B, и пусть также $r = r_i$, $t = t_i$, $s = s_i$ и $0 < r < R$. Последнее неравенство, как будет показано ниже, существенно. Будем считать, что $m < r$ и $(m, r) = 1$.

Абонент W_i шифрует сообщение m сначала своим секретным ключом:

$$m_1 = m^t \pmod{r},\ 0 < m_1 < r,$$

а потом открытым ключом абонента B:

$$m_2 = m_1^S \pmod{R},\ 0 < m_2 < R.$$

Абонент B, получив шифрованное сообщение m_2, расшифровывает распоряжение сначала своим секретным ключом:

$$m_3 = m_2^T \pmod{R},\ 0 < m_3 < R,$$

а потом открытым ключом абонента W_i :

$$m_4 = m_3^s \pmod{r},\ 0 < m_4 < r.$$

Покажем, что $m_4 = m$. Из соотношений

$$m_3 = m_2^T \pmod{R};$$
$$m_2 = m_1^S \pmod{R}$$

следует, что

$$m_3 = m_1^{ST} \pmod{R},$$

где

$$ST = 1 \pmod{\varphi(R)}.$$

По теореме Эйлера

$$m_3 = m_1^{ST} = m_1 \pmod{R} = m_1.$$

Аналогично

$$m_4 = m_3^{s} \pmod{r} = m_1^{s} \pmod{r} = m^{st} \pmod{r},\ st = 1 \pmod{\varphi(r)}.$$

Следовательно, $m_4 = m$. Таким образом, абонент B знает, что получил сообщение от абонента W_i.

3.6.Телекоммуникационные протоколы, использующие сеансовые ключи

Общепринятой криптографической техникой является шифрование каждого индивидуального обмена сообщениями отдельным ключом. Такой ключ называется сеансовым, так как он используется для единственного отдельного сеанса обмена информацией. Передача общего сеансового ключа в руки абонентов, обменивающихся информацией, является одной из задач, которые решаются в телекоммуникационных системах.

3.6.1. Обмен ключами с помощью симметричного шифрования

При обмене ключами с помощью симметричного шифрования используется протокол, в котором абоненты сети A и B, получают секретный ключ от центра распределения ключей (ЦРК). Перед началом работы сети с использованием указанного протокола ключи должны быть известны абонентам. В этой связи последовательность действий абонентов следующая

- Абонент A обращается к ЦРК и запрашивает сеансовый ключ для связи с абонентом B.
- Центр распределения ключей формирует случайный сеансовый ключ. Зашифровывается две копии ключа: одна для абонента A, а

другая — для абонента *B*. Затем с ЦРК посылается обе копии абоненту *A*.

- Абонент *A* расшифровывает свою копию сеансового ключа.
- Абонент *A* посылает абоненту *B* его копию сеансового ключа.
- Абонент *B* расшифровывает свою копию сеансового ключа.
- Абоненты *A* и *B* используют этот сеансовый ключ для безопасного обмена информацией.

Этот протокол основан на абсолютной надежности центра распределения ключей.

Рассмотрим протоколы, которые реализованы с применением симметричного шифрования. Пусть абоненты *A* и *B* владеют общим секретным ключом k_{ab}, который применяется в симметричном шифровании. Тогда для передачи нового ключа *k* от абонента *A* к абоненту *B* используется следующий алгоритм [5]:

$$A \rightarrow B: E_{kab}(k, t, B),$$

где E_{kab} ( ) — обозначение, свидетельствующее о том, что содержимое круглых скобок зашифровано на ключе *kab*, *k* — новый ключ, *t* — метка времени, *B* — идентификатор участника протокола *B*.

При односторонней аутентификация используется следующая схема протокола.

- Абонент *B* генерирует случайное число r_b и пересылает его абоненту *A*.
- Абонент *A* пересылает абоненту *B* сообщение

$$E_{kab}(k, t, r_b, B).$$

Если требуется двусторонняя аутентификация, то используется протокол следующего вида.

- Абонент *B* формирует случайное число r_b и пересылает его абоненту *A*.
- Абонент *A* генерирует случайное число r_a и пересылает абоненту *B* сообщение

$$E_{kab}(k, t, r_a, r_b, B).$$

- Абонент *B* пересылает абоненту *A* сообщение $E_k(r_a)$.

Третий шаг позволит абоненту *A* убедиться в том, имеет ли он дело с абонентом *B* и получил ли он правильное значение ключа *k*.

Последний протокол можно модифицировать таким образом, чтобы в формировании нового ключа принимали участие абоненты *A* и *B* [5]. Пример такого протокола дан ниже.

- Абонент *B* генерирует случайное число r_b и пересылает его абоненту *A*.
- Абонент *A* генерирует случайное число r_a, ключ k_a и пересылает абоненту *B* сообщение

$$E_{kab}(k_a, t, r_a, r_b, B).$$

- Абонент *B* генерирует ключ k_b и пересылает абоненту *B* сообщение

$$E_{kab}(k_b, t, r_a, r_b, A).$$

После обмена по протоколу каждый абонент по некоторой функции вычисляет общий ключ

$$k = f(k_a, k_b).$$

3.6.2. Обмен ключами, используя алгоритмы с открытыми ключами

Для согласования сеансового ключа абоненты *A* и *B* применяют алгоритмы с открытыми ключами, а затем используют этот сеансовый ключ для шифрования данных. Применяемые ключи абонентам *A* и *B* доступны в некоторой базе данных. Это значительно облегчает протокол работы сети. В этом случае этот протокол имеет следующий вид.

- Абонент *A* получает открытый ключ абонента *B* из базы данных.
- Абонент *A* генерирует случайный сеансовый ключ, зашифровывает его открытым ключом абонента *B* и посылает его абоненту *B*.
- Абонент *B* расшифровывает сообщение абонента *A* с помощью своего закрытого ключа.
- Абоненты *A* и *B* шифруют свой обмен информацией этим сеансовым ключом.

3.6.3. Передача ключей и сообщений

Рассмотрим ситуацию, когда абоненты *A* и *B* не выполняют протокол обмена ключами перед обменом сообщениями. В этом случае абонент *A* отправляет абоненту *B* сообщение без предварительного протокола обмена ключами.

- Абонент *A* генерирует случайный сеансовый ключ *K* и зашифровывает этим ключом сообщение *M*, которое представим в виде $E_K(M)$
- Абонент *A* получает открытый ключ абонента *B* из базы данных.
- Абонент *A* шифрует *K* открытым ключом абонента *B* и этот шифр имеет вид $E_B(K)$.
- Абонент *A* посылает абоненту *B* шифрованные сообщение $E_K(M)$ и сеансовый ключ E_B (*K*). Для дополнительной защиты от вскрытия абонент *A* подписывает передачу сообщения.
- Абонент *B* расшифровывает сеансовый ключ *K*, используя свой закрытый ключ.
- Абонент *B*, используя сеансовый ключ, расшифровывает сообщение от абонента *A*.

Подобная смешанная система употребляется чаще всего в телекоммуникационных системах передачи информации. Ее соединяют с цифровыми подписями, метками времени и другими протоколами обеспечения безопасности.

3.6.4. Алгоритм открытого распределения ключей Диффи — Хеллмана

Алгоритм открытого распределения ключей был изобретен в 1976 г. Абоненты *A* и *B* с помощью этого алгоритма производят генерацию секретного ключа. Рассмотрим этот алгоритм подробнее.

Абоненты A и B вместе выбирают большие простые числа n и g так, чтобы g было примитивным элементом по mod n. Абоненты A и B договариваются об их использовании по открытому каналу. Эти числа могут совместно использоваться группой абонентов. Затем выполняется следующий протокол:

- Абонент A выбирает случайное большое целое число x и посылает абоненту B сообщение

$$X = g^x \bmod n.$$

- Абонент B выбирает случайное большое целое число y и посылает абоненту A сообщение

$$Y = g^y \bmod n.$$

- Абонент A вычисляет значение

$$k = Y^x \bmod n.$$

- Абонент B вычисляет

$$k' = X^y \bmod n.$$

Можно показать, что значения k, и k' равны значению $g^{xy} \bmod n$. Несанкционированный абонент не сможет вычислить это значение, поскольку известно только n, g, X и $У$.

3.6.5. Алгоритм распределения ключей Диффи — Хеллмана с тремя и более абонентами

Протокол обмена ключами Диффи — Хеллмана можно расширить на случай с тремя и более абонентами. Рассмотрим пример трех абонентов A, B и C, которые совместное генерируют секретный ключ.

- Абонент A выбирает случайное большое целое число x и вычисляет

$$X = g^x \bmod n.$$

- Абонент B выбирает случайное большое целое число y и посылает абоненту C

$$Y = g^y \bmod n.$$

- Абонент C выбирает случайное большое целое число z и посылает абоненту A

$$Z = g^z \bmod n.$$

- Абонент A посылает абоненту B

$$Z' = Z^x \bmod n.$$

- Абонент B посылает абоненту C

$$X' = X^y \bmod n.$$

- Абонент C посылает абоненту A

$$Y' = Y^z \bmod n.$$

- Абонент A вычисляет закрытый ключ

$$k = Y'^x \bmod n.$$

- Абонент B вычисляет закрытый ключ

$$k = Z'^y \bmod n.$$

- Абонент C вычисляет закрытый ключ

$$k = X'^z \bmod n.$$

Секретный ключ k равен $g^{xyz} \bmod n$. Протокол распределения ключей можно расширить для четверых и более абонентов.

3.6.6. Алгоритм распределения ключей Хьюза

Этот вариант алгоритма Диффи — Хеллмана позволяет абоненту A генерировать ключ и послать его абоненту B.

- Абонент A выбирает случайное большое целое число x и генерирует закрытый ключ

$$k = g^x \bmod n.$$

- Абонент B выбирает случайное большое целое число y и посылает абоненту A

$$Y = g^y \bmod n.$$

- Абонент A посылает абоненту B

$$X = Y^x \bmod n.$$

- Абонент B вычисляет

$$z = y^{-1} \bmod (n - 1);$$

$$k' = X^z \bmod n.$$

Если все процедуры выполнены, то закрытый ключ $k = k'$.

Преимуществом этого протокола распределения ключей над алгоритмом Диффи — Хеллманом состоит в том, что закрытый ключ k можно вычислить заранее, до взаимодействия, и абонент A может шифровать сообщения с помощью закрытого ключа k до установления соединения абонента A с абонентом B. Абонент A может послать сообщение сразу множеству абонентов, а закрытый ключ передать позднее каждому по отдельности.

3.6.7. Протокол обмена зашифрованными ключами EKE

Протокол обмена зашифрованными ключами EKE (Encrypted Key Exchange) был разработан С. Белловином и М. Мерриттом. Он обеспечивает безопасность и проверку подлинности в компьютерных сетях, используя симметричные шифры и шифры с открытыми ключами. Общий закрытый ключ используется для шифрования генерированного случайным образом открытого ключа.

Пусть абоненты A и B имеют общий пароль P. Проверка подлинности абонентов и генерирование общего сеансового ключа K. осуществляется в соответствии со следующим протоколом.

- Абонент A случайным образом генерирует пару «открытый ключ/закрытый ключ». Он шифрует открытый ключ K' *с* помощью симметричного шифра, используя P в качестве ключа: $Ep(K')$. Абонент A посылает абоненту B сообщение

$$Ep(K').$$

- Абонент B, зная общий пароль P, расшифровывает сообщение и выделяет открытый ключ K'. Затем абонент B генерирует случайный сеансовый ключ K и шифрует его открытым ключом, который он получил от абонента A. После этого, используя общий пароль P качестве ключа он посылает абоненту A сообщение

$$Ep(E_{K'}\ (K)).$$

- Абонент A расшифровывает сообщение, получая сеансовый ключ K. Он генерирует случайную строку чисел R_A, шифрует ее с помощью сеансового ключа K и посылает абоненту B

$$E_K\ (R_A).$$

- Абонент B расшифровывает сообщение, выделяя R_A. Он генерирует другую случайную строку, R_B, шифрует обе строки сеансовым ключом K и посылает абоненту A результат

$$E_K\ (R_A, R_B).$$

- Абонент A расшифровывает сообщение, получая R_A и R_B. Если строка R_A, полученная от абонента B, является истинной, той самой строкой, которую он послал абоненту B на третьем этапе протокола, то абонент A, используя сеансовый ключ K шифрует строку R_B и посылает ее абоненту B:

$$E_K\ (R_B).$$

- Абонент B расшифровывает сообщение и получает R_B. Если строка R_B полученная от абонента A является той самой строкой, которую он послал абоненту A на четвертом этапе протокола, то считается что протокол завершен. Теперь оба абонента могут обмениваться информацией, используя сеансовый ключ K.

Весь протокол можно условно разбить на группы: с третьего по шестой этапы обеспечивает подтверждение, с третьего по пятый этапы доказывают абоненту A, что абонент B знает сеансовый ключ K, шестой этап доказывает абоненту B, что абонент A знает K.

Алгоритм EKE может быть реализован с использованием множества алгоритмов с открытыми ключами, например алгоритмы: RSA, Эль-Гамаля, Диффи — Хеллмана.

Рассмотрим реализацию протокола обмена зашифрованными ключами EKE с помощью протокола Диффи — Хеллмана. При использовании протокола Диффи — Хеллмана сеансовый ключ K генерируется автоматически. Значения g и n определяются для всех абонентов сети.

- Абонент A выбирает случайное число r_A и посылает абоненту B

$$g^{r_A} \bmod n.$$

При использовании протокола Диффи — Хеллмана абоненту A не нужно шифровать свое первое сообщение с помощью общего пароля P.

- Пользователь B выбирает случайное число r_B и вычисляет

$$K = g^{r_A r_B} \bmod n.$$

Он генерирует случайную строку r_B, затем вычисляет и посылает абоненту A сообщение зашифрованное с помощью общего пароля P:

$$E_P(g^{r_B} \bmod n),\ E_K(r_B).$$

- Абонент A расшифровывает первую половину сообщения абонента B, извлекая сообщение $g^{r_B} \bmod n$. Затем абонент A вычисляет сеансовый ключ K и использует его для шифрования R_B. Абонент A генерирует другую случайную строку R_A, шифрует обе строки ключом K и посылает результат абоненту B.

$$E_K(R_A, R_B).$$

- Абонент B расшифровывает сообщение, получая r_A, и r_B. Если полученная от абонента A строка r_B совпадает с той, которую он посылал абоненту A на втором этапе, абонент B он шифрует r_A сеансовым ключом K и посылает результат абоненту A

$$E_K(R_A).$$

- Абонент A расшифровывает сообщение, получая r_A. Если полученная от абонента B строка r_A совпадает с той, которую он посылал абоненту B на третьем этапе, протокол завершается.

Оба абонента могут обмениваться сообщениями, используя K в качестве сеансового ключа.

3.7. Удостоверение подлинности

Протокол аутентификации представляет собой криптографический протокол, в ходе выполнения, которого один абонент удостоверяется в идентичности другого абонента, использующего протокол. Протоколы аутентификации применяются достаточно часто в телекоммуникационных системах. Например, в системе мобильной связи при включении мобильной станции (мобильного телефона) происходит процедура аутентификации абонента по информации, имеющейся в SIM-карте. В любом протоколе аутентификации участвуют две стороны: абонент доказывающий (абонент *P*) и абонент проверяющий (абонент *V*). Цель абонента *V* заключается в том, чтобы подтвердить предполагаемую идентичность абонента доказывающего *P*, что он в действительно является *P*, а не кем-то иным. После выполнения протокола аутентификации абонент проверяющий должен либо принять абонента доказывающего как аутентичного, либо отвергнуть его как не соответствующего заявленной идентичности. Рассмотрим три способа аутентификации.

- «Субъект знает» — абонент доказывающий обладает некоторой информацией, которой нет у других субъектов телекоммуникационной системы (паролями, цифровыми кодами, закрытыми ключами) и знание которой он демонстрирует в протоколах аутентификации.
- «Субъект обладает» — абонент доказывающий имеет некоторый физический предмет (магнитную карту, интеллектуальную карту, генератор паролей), который необходим для его участия в протоколе аутентификации и который выполняет для него криптографические преобразования информации.
- «Субъект есть» — в протоколе проверяются некоторые признаки, характеризующие индивидуальность субъекта (биометрические признаки: отпечатки пальцев, голос, рисунок радужной оболочки глаза и др.).

Криптографические протоколы реализуют первый способ. Когда абонент *A* подключается к телекоммуникационной сети, его опознание осуществляется с помощью паролей. Абонент *A* вводит свой пароль, и в телекоммуникационной сети проверяется его правильность. Таким образом, и абоненту *A* и телекоммуникационной сети известна некоторая закрытая информация, которую телекоммуникационная сеть запрашивает всякий раз, когда абонент *A* пытается подключится.

3.7.1. Удостоверение подлинности с помощью однонаправленных функций

Для проверки паролей можно хранить, не сами значения, а хеш-функции паролей. Хеш-функция является отображением сообщения произвольной длины в строку фиксированного размера. Подробнее свойства хеш-функций рассматриваются в разд. 3.8. При этом протокол проверки паролей следующий

- Абонент *A* посылает в телекоммуникационную сеть свой пароль.
- В телекоммуникационной сети вычисляется однонаправленная хеш-функция пароля.
- В телекоммуникационной сети происходит сравнение полученного значение хеш-функция пароля с хранящимся в памяти значением хеш-функции.

Поскольку в телекоммуникационной сети не хранится таблица паролей всех абонентов, снижается угроза похищения таблицы паролей. Список паролей, обработанный однонаправленной функцией, не может быть расшифрован, так как однонаправленную функцию не удастся инвертировать для получения паролей.

3.7.2. Удостоверение подлинности с помощью открытых ключей

В телекоммуникационной сети хранятся данные об открытых ключах всех абонентов, а все абоненты хранят свои закрытые ключи. В этом случае протокол подключения имеет следующий вид.

- Из телекоммуникационной сети абоненту *A* посылается случайная строка.
- Абонент *A* шифрует эту строку своим закрытым ключом и посылает ее обратно в телекоммуникационную сеть вместе со своим именем.
- В телекоммуникационной сети из базы данных определяется открытый ключ абонента *A*, и дешифрируется сообщение.
- Если отправленная сначала и расшифрованная строки совпадают, то телекоммуникационная сеть предоставляет абоненту *A* доступ к системе.

Несанкционированный абонент не сможет воспользоваться закрытым ключом абонента *A*, следовательно, не сможет выдать себя за абонента *A*. Абонент *A* никогда не посылает в телекоммуникационную сеть свой закрытый ключ. Закрытый ключ должен быть достаточно длинным и не должен быть мнемоническим. Он будет автоматически обрабатываться аппаратурой абонента или программным обеспечением телекоммуникационной сети.

3.7.3 Формальный анализ протоколов проверки подлинности и обмена ключами

Задача формирования безопасного сеансового ключа для пары абонентов привела к задаче анализа протоколов проверки подлинности и обмена ключами. Существует четыре основных подхода к анализу криптографических протоколов

- Моделирование и проверка работы протокола с использованием языков описания и средств проверки, не разработанных специально для анализа криптографических протоколов.
- Создание экспертных систем, позволяющих конструктору протокола разрабатывать и исследовать различные сценарии.
- Выработка требований к семейству протоколов, используя понятия «знание» и «доверие».
- Разработка формальных математических методов, основанных на записи свойств криптографических систем в алгебраическом виде.

3.7.4. Протокол аутентификации с нулевым разглашением информации

Пусть абонент *A* обладает информацией *S* и должен доказать абоненту *B*, что эту информацию имеет.

Нулевое разглашение информации подразумевает, что в результате работы протокола интерактивной системы доказательства абонент *B* не увеличит свои знания об информации *S* или не сможет извлечь никакой информации о том, почему *S* истинно.

В ходе протокола абоненты *A* и *B* обмениваются сообщениями. Каждый из них может генерировать случайные числа и использовать их в своих вычислениях. В конце протокола абонент *B* должен вынести решение о том, является ли *S* истинным или ложным.

Цель абонента *A* всегда состоит в том, чтобы убедить абонента *B* в том, что *S* истинно, независимо от того, истинно ли оно на самом деле или нет. Цель абонента *B* заключается в том, чтобы вынести решение, является ли *S* истинным или ложным.

Рассмотрим теперь примеры протоколов доказательства с нулевым разглашением знания. В качестве первого протокола рассмотрим решение задачи, которая называется «Задача о пещере Али-Бабы». Имеется пещера, план которой показан на рис. 3.1. Пещера

имеет дверь с секретом между точками *C* и *D*. Каждый, кто знает кода, может открыть эту дверь и пройти из *C* в *B* или наоборот. Для всех остальных оба хода пещеры ведут в тупик.

Пусть абонент *A* знает секрет пещеры. Он хочет доказать абоненту *B* знание этого секрета, не разглашая код. Вот протокол их общения.

- Абонент *B* находится в точке *E*.
- Абонент *A* заходит в пещеру и добирается либо до точки *C*, либо до точки *D*.
- После того, как абонент *A* исчезает в пещере, абонент *B* приходит в точку *F*, не зная, в какую сторону пошел абонент *A*.
- Абонент *B* зовет абонента *A* и просит его выйти либо из левого, либо из правого коридора пещеры согласно желанию абонента *B*.
- Абонент *A* выполняет это, открывая при необходимости дверь, если он знает код.
- Абоненты *A* и *B* повторяют предыдущие шаги *n* раз.

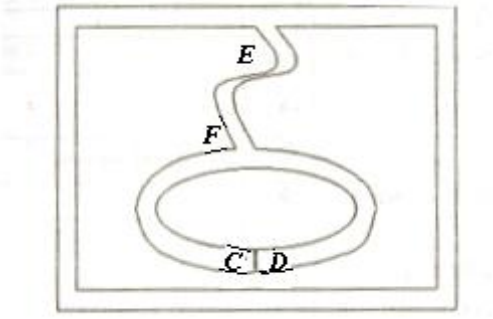

Рис. 3.1. Задача о пещере Али-Бабы

Алгоритм повторяется столько раз, сколько требуется для того, чтобы убедить абонента *B*, что абонент *A* обладает знанием *S*.

3.7.5. Протоколы аутентификации с вероятностной системой доказательств

В класс вероятностных доказательств включают: интерактивные системы доказательств, доказательства с нулевым разглашением, вероятностно-проверяемые доказательства и другие виды доказательств.

Рассмотрим протокол доказательства знания дискретного логарифма x числа X. Задаются открытые простые числа p и q, такие, что q делит $p - 1$. Задается примитивный элемент g множества $GF(p)$. Доказывающему абоненту A необходимо доказать абоненту B, что ему известна величина, такая что число x принадлежит множеству $GF(p)$, причем число $g^x = X$ — открытое.

- Абонент A выбирает случайное число r, вычисляет $g^r = M \bmod p$ и посылает его абоненту B.
- Абонент B посылает абоненту A случайное число R принадлежащее множеству $GF(p)$.
- Абонент A вычисляет число $m = r + xR \bmod q$ и посылает его абоненту B.
- Абонент B проверяет, что $g^m = X^R M \pmod p$.

Этот этап протокола называется аккредитацией. Абоненты A и B повторяют этот протокол t раз, пока абонент B не убедится, что абонент A знает число x.

Рассмотрим протокол аутентификации Шнорра с нулевым разглашением. Пусть p и q — простые числа такие, что q делит $p - 1$. В этом протоколе предлагается использовать p длиной 512 бит и q длиной 140 бит. Пусть $g \in Z_p$ таково, что $g^q = 1 \pmod p$, $g \neq 1$. Пусть $x \in Z_q$ и $y = g^x \pmod p$. Задача вычисления значения x по заданному значению y при известных значениях p, q и g называется задачей дискретного логарифмирования. Воспользуется гипотезой о вычислительной трудности задачи дискретного логарифмирования. В качестве закрытого ключа схемы аутентификации абонент A выбирает случайное число x из $\{1, \ldots, q - 1\}$. Далее абонент A вычисляет

$y = g^{-x} \pmod p$) и публикует открытый ключ y. Открытые ключи всех абонентов должны публиковаться таким образом, чтобы исключалась возможность их подмены. Запишем протокол аутентификации Шнорра.

- Абонент A выбирает случайное число k из множества {1, …, $q - 1$}, вычисляет $r = g^k \pmod p$ и посылает r абоненту B.
- Абонент B выбирает случайный запрос e из множества {1, …, $2^t - 1$}, где t — некоторый параметр, и посылает e абоненту A
- Абонент A вычисляет $s = (k + xe) \pmod q$ и посылает s абоненту B.
- Абонент B проверяет соотношение $r=g^s y^e \pmod p$, и если оно выполняется, то принимает доказательство, в противном случае отвергает его.

Протокол выполняется столько раз, сколько необходимо для доказательства знания x.

Рассмотрим алгоритм идентификации Гиллу — Кискате. Абонент A обладает строкой данных, передавая которую он должен доказать абоненту B доказать свою подлинность (именно он является абонентом A) абоненту B. Идентификация абонента A проводится по ряду признаков строки данных. Эта битовая строка обозначается J. Она аналогична открытому ключу. Другой открытой информацией является показатель степени v и модуль n, где n — произведение двух простых чисел. Закрытым ключом, которым владеет абонент A, служит число D, рассчитываемое так, чтобы

$$JD^v = 1 \pmod n.$$

Абонент A посылает абоненту B признаки J. Абонент A хочет доказать абоненту B, что это именно признаки J. Для этого нужно доказать абоненту B, что абонент A владеет закрытым ключом D. Доказательство выполняется в соответствии со следующим протоколом.

- Абонент A выбирает случайное целое число r, находящееся в диапазоне от 1 до $n - 1$. Вычисляется $T = r^v \bmod n$ и отправляется абоненту B.

- Абонент B выбирает случайное целое d, находящееся в диапазоне от 0 до $v - 1$ и посылает число d абоненту A.
- Абонент A вычисляет число $C = rD^d \bmod n$ и посылает его абоненту B.
- Абонент B вычисляет число $T' = C^v J^d \bmod n$. Если $T = T' \pmod n$, то абонент В убеждается в том, что абонент является подлинным. Данное равенство выполняется в соответствии с выражениями

$$T' = C^v J^d = (rD^d)^v J^d = r^v (D^v J)^d = r^v = T \pmod n,$$

так как

$$JD^v = 1 \pmod n.$$

3.7. Хеш-функция

Хеш-функция (функция хеширования) H представляет собой отображение, на вход которого подается сообщение переменой длины M, а выходом является строка фиксированной длины $H(M)$. Хеш-функция должна обладать следующими свойствами.

1. Хеш-функция может быть применена к аргументу любого размера.
2. Выходное значение хеш-функции имеет фиксированный размер.
3. Хеш-функцию $H(M)$ достаточно просто вычислить для любого M.
4. Для любого y с вычислительной точки зрения невозможно найти x, такое что $H(x) = y$.
5. Для любого фиксированного x с вычислительной точки зрения невозможно найти z, не равное x, такое, что $H(x) = H(z)$.

Четвертое свойство эквивалентно тому, что H является односторонней функцией. Пятое свойство гарантирует, что не может быть найдено другое сообщение, дающее то же значение. Это позволяет использовать H в качестве контрольной суммы для проверки целост-

ности передаваемого сообщения. Рассмотрим основные области использования хеш-функции.

- Защита паролей при их передаче и хранении.
- Формирование контрольных кодов MDC (Manipulation Detection Code) — кода обнаружения манипуляций с данными.
- Получение сжатого образа сообщения перед формированием электронной подписи.

Рассмотрим более подробно использование хэш-функции для формирования электронной подписи.

Пусть абоненту A необходимо послать сообщение M с электронной подписью. С этой целью абонентом A выбирается два простых числа P и Q, вычисляет $N = PQ$ и $\varphi(N)$. После этого выбирается число $0 < d < \varphi(N)$ (открытое число) и вычисляется закрытое число $c = d^{-1} \bmod \varphi(N)$.

Для формирования электронной подписи под передаваемым сообщением $M = (m_1, m_2, \ldots, m_n)$, вычисляется хеш-функция $y = H(M)$ и число $s = y^c \bmod N$, которое и является цифровой подписью под сообщением M. Абонент A передает пару сообщений (M, s). Абонент B, зная открытое число d, проверяет цифровую подпись, вычисляя хеш-функцию $y_1 = H(M)$ и $w = s^d \bmod N = y_1^{cd} \bmod N$. Подпись является подлинной, если $y_1 = y$.

3.8.1. Функция хеширования в стандарте DSS

Алгоритм безопасного хеширования (алгоритм формирования функции хеширования) принят в качестве стандарта США в 1992 г. и предназначен для использования совместно с алгоритмом цифровой подписи, определенным в стандарте DSS (Digital Signature Standard). Рассмотрим реализацию этого алгоритма подробнее.

Прежде всего, исходное сообщение дополняется символами так, чтобы его длина стала кратной 512 битам. При дополнении к сообщению добавляется единица, затем столько нулей, сколько необходимо

для получения сообщения, длина которого на 64 бита меньше, чем кратная 512, и затем добавляется 64-битовый код длины сообщения.

После дополнения полученная информационная последовательность имеет вид

$$p = p_1, p_2, ..., p_m, i = 1, 2, ..., m,$$

где длина всех блоков равна 512 битам. Используется рекуррентный алгоритм вычисления хеш-функции последовательно по мере обработки блоков.

На вход i-го основного цикла рекуррентного алгоритма хеширования, который обозначен как SHA_i, поступает i-й блок информационной последовательности p_i и результат работы предыдущего цикла SHA_{i-1}, т. е.

$$SHA_i = h(p_i, SHA_{i-1}).$$

Каждый блок p_i может быть представлен как результат конкатенации шестнадцати 32 разрядных слов (последовательного слияния разрядов каждого из 16 слов) в соответствии с выражением

$$p_i = w_1//w_2// ... //w_{16},$$

где «//»— означает знак конкатенации. Тогда набор из 16 слов полностью отражает информацию, содержащуюся в блоке p_i. Перед началом каждого цикла соответствующий блок w_1, w_2, ... , w_{16} расширяется до 80 слов по 32 разряда в каждом. Расширение происходит следующим образом. Пусть w_1,w_2, ... w_{16} исходный блок, а w_1, w_2, ... , w_{80} расширенный блок. При этом алгоритм расширения может быть записан в виде

$$w_j = w_j \text{ для } j = 1, 2, ... , 16,$$

$$w_j = \text{Rol}\,(w_{j-3} \oplus w_{j-8} \oplus w_{j-14} \oplus w_{j-16}) \text{ для } j = 17, 18, ... , 80,$$

где Rol — операция циклического сдвига на один разряд влево. Первые 16 слов остаются неизменными, а все остальные слова определяются в соответствии с последней формулой.

Инициализируются пять 32 битовых переменных следующими шестнадцатеричными постоянными числами:

A = 67452301, B = EFCDAB89, C = 98BADCFE, D = 10325476, E = C3D2E1F0.

При этом первый (стартовый) цикл рекуррентного алгоритма хеширования SHA_0 является результатом конкатенации (слияния) этих переменных, т. е.

$$SHA_0 = A//B//C//D//E.$$

Полученные пять переменных копируются в новые переменные *a, b, c, d* и *e* соответственно.

Главный цикл вычисления хеш-функции имеет следующий вид.

- Переменная t последовательно за 80 шагов принимает значения от 0 до 79 с шагом 1.
- На каждом шаге вычисляется вспомогательная переменная temp, и изменяются переменные *a, b, c, d* и *e* согласно выражениям

$$\text{temp} = (a << 5) + f_t(b, c, d) + w_t + k_t;$$
$$e = d;\ d = c;\ c = b << 30;\ b = a;\ a = \text{temp}.$$

Здесь обозначено: << — операция циклического сдвига влево; $\wedge$ — операция логического побитового «и»; $\vee$ — операция логического побитового «или»; $\neg$ — операция побитового «не»; $\oplus$— операция побитового сложения по модулю 2; k_t — шестнадцатеричные постоянные числа, определяемые по следующим формулам:

$$k_t = 5A827999,\ t = 0, 1, ..., 19;$$
$$k_t = 6ED9EBA1,\ t = 20,\ 21, ..., 39;$$
$$k_t = 8F1BBCDC,\ t = 40, 41, ..., 59;$$
$$k_t = CA62C1D6,\ t = 60, 61, ..., 79.$$

Функции $f_t(x, y, z)$ задаются следующими выражениями:

$$f_t(x, y, z) = x \wedge y \vee \neg x \wedge z;,\ t = 0, 1, ..., 19;$$
$$f_t(x, y, z) = x \oplus y \oplus z;\ t = 20, 21, ..., 39;\ t = 60, 61, ..., 79;$$
$$f_t(x, y, z) = x \wedge y \vee x \wedge z \vee y \wedge z;,\ t = 40, 41, ..., 59;$$

После окончания цикла значения *a, b, c, d* и *e* складываются по модулю 2 с пятью 32-битовыми переменными *A, B, C, D* и *E* соответственно, и осуществляется переход к обработке следующего 512-

битового блока расширенного сообщения. Выходное значение хэш-функции является конкатенацией (слиянием) значений *A*, *B*, *C*, *D* и *E*.

3.8.2. Функция хеширования в стандарте ГОСТ Р34.11-94

В этом стандарте с помощью функция хеширования информационная последовательность преобразуется в выходную последовательность разрядностью 256 бит. В соответствии со стандартом в основе данной хеш-функции лежит алгоритм блочного кодирования ГОСТ 28147-89.

Пусть p входная информационная последовательность. Последовательность p разбивается на блоки $p_1, p_2, ..., p_m$, $i = 1, 2, ..., m$, где длина всех блоков равна 256 битам. Последний неполный блок дополняется до требуемого размера. Добавляются еще два 256-разрядных блока, содержащие код длины последовательности и контрольную сумму. Тогда рекуррентный процесс вычисления хеш-фукции $H(p)$ может быть описан следующим образом:

$$\mathrm{GOST}_i = H(p_i, \mathrm{GOST}_{i-1}),$$

где GOST_i — результат i-го цикла алгоритма преобразования; GOST_0 — 256-разрядный стартовый цикл хеширования, на выбор которого ограничений не накладывается; p_i — очередной i-й блок входной информации $i = 1, 2, \ldots m$. Алгоритм вычисления функции хеширования GOST_i — состоит из трех частей:

- генерации четырех 256-битовых ключей;
- шифрующего преобразования, заключающегося в том, что происходит шифрование 64-битовых подслов слова X разрядности 256 бит, где X — очередной блок обрабатываемой открытой информации. Шифрование осуществляется на ключах K_i, где $i = 1, 2, 3, 4$ с использованием алгоритма шифрования ГОСТ 28147-89 в режиме простой замены;
- перемешивающего преобразования результата шифрования.

Рассмотрим процесс генерации ключей. Обозначим очередной 256-битовый вектор p_i через X:

$$X = (b_{256}, b_{255}, ..., b_1),$$

где b_i выбираются из множества {0, 1}, i = 1, 2, …, 256. Этот же вектор X можно представить и в других эквивалентных формах записи:

$$X = x_4 \,//\, x_3 \,//\, x_2 \,//\, x_1 = \eta_{16} \,//\, \eta_{15} \,//\, ... \,//\, \eta_1 = \xi_{32} \,//\, \xi_{31} \,//\, ... \,//\, \xi_1,$$

где x_i, i = 1, 2, 3, 4 — 64-битовые вектора; η_j, j = 1, 2, 3, ... 16 — 32-битовые вектора; ξ_k, k = 1, 2, 3, …, 32 — 8-битовые вектора, обозначение «//» соответствует операции конкатенации (слияния) битовых символов. Таким образом, 256-битовый вектор X может быть получен конкатенацией (слиянием) различных битовых символов.

Обозначим через $A(X)$ функцию вида

$$A(X) = (x_1 \oplus x_2) \,//\, x_4 \,//\, x_3 \,//\, x_2,$$

где x_i, i = 1, 2, 3, 4 — 64-битовые вектора. Введем отображение T, которое отображает вектор

$$\xi_{32} \,//\, \xi_{31} \,//\, ... \,//\, \xi_1$$

в вектор

$$\xi_{\varphi(32)} \,//\, \xi_{\varphi(31)} \,//\, ... \,//\, \xi_{\varphi(1)},$$

где функцию φ можно записать в виде

$$\varphi(i + 1 + 4(k - 1)) = 8i + k,\ i = 0, 1, 2, 3;\ k = 1, ..., 8.$$

Перестановка элементов 256-битовой последовательности выполняется по формуле $y = \varphi(x)$, где x — порядковый номер 8-битового значения в исходной последовательности; y — порядковый номер 8-битового значения в результирующей последовательности.

Для генерации ключей необходимо использовать следующие исходные данные.

- $G_0 = C_0 = \mathrm{GOST}_0$ — 256-разрядный стартовый вектор;
- Постоянные числа C_i (i = 2, 3, 4), имеющие значения

$C_2 = C_4 = 0^{256}$ (256 нулей),

$C_3 = 1^8\,0^8\,1^{16}\,0^{24}\,1^{16}\,0^8\,(0^8\,1^8)^2\,1^8\,0^8\,(0^8\,1^8)^4\,(1^8\,0^8)^4$,

где показатель степени означает число повторений соответствующей цифры. Поскольку алгоритм вычисления хеш-функции рекуррентный, для текущих значений введем обозначения G — текущее значение

хеш-функции, X — очередной блок обрабатываемой открытой информации. При вычислении ключей реализуется следующий алгоритм.

1. Присвоить значения $i = 1$, $U = G$, $V = X$.
2. Выполнить вычисление $W = U \oplus V$, $K_i = T(W)$.
3. Увеличить счетчик i на единицу $i = i + 1$.
4. Проверить условие $i = 5$. При положительном исходе перейти к седьмому шагу.
5. Выполнить вычисление

   $U = A(U) \oplus C_i$, $V = A(A(V))$, $W = U \oplus V$, $K_i = T(W)$.
6. Перейти к третьему шагу.
7. Окончание вычислений.

При преобразовании осуществляется шифрование 64-битовых подслов 256-битового слова G. Для шифрующего преобразования используется представление G в виде

$$G = x_4 // x_3 // x_2 // x_1,$$

где x_i — 64-битовый вектор, $i = 1, 2, 3, 4$ и набор ключей K_i, $i = 1, 2, 3, 4$. После выполнения шифрования, согласно шифру ГОСТ 28147-89, получаются слова

$$s_i = E_i(x_i),$$

где $i = 1, 2, 3, 4$; E_i — функция шифрования в режиме простой замены на ключе K_i. В результате получается вектор

$$S = s_4 // s_3 // s_2 // s_1 .$$

Затем осуществляется перемешивание полученной последовательности с применением регистра сдвига. Исходными данными являются слова G, X и S.

Пусть отображение ψ преобразует слово $S = \eta_{16} // \eta_{15} // ... \eta_1$ в слово $\psi(S)$ по правилу

$$\psi(S) = \eta_1 \oplus \eta_2 \oplus \eta_3 \oplus \eta_4 \oplus \eta_{13} \oplus \eta_{16} // \eta_{15} // ... // \eta_2.$$

Тогда в качестве значения функции хеширования на следующем шаге принимается слово

$$G_{step} = \psi^{61}(G \oplus \psi(X \oplus \psi^{12}(S))),$$

где ψ^{i} — i-я степень преобразования ψ, т. е. преобразование осуществляется i раз, G_{step} — значение функции хеширования на следующем шаге.

Возвращаясь к ранее введенным обозначениям, имеем

$$\text{GOST}_i = \psi^{61}((\text{GOST}_{i-1} \oplus \psi(p_i \oplus \psi^{12}(E(p_i)))).$$

Процесс повторяется до тех пор, пока не будет обработана вся информационная последовательность. Полученный на последнем шаге 256-битовый блок символов является значением функции хеширования $y = H(M) = \text{GOST}_{\text{imax}}$, где imax — номер последнего 256-битового блока.

3.8.3. Стандарт цифровой подписи ГОСТ Р34.10-94

Формирование цифровой подписи осуществляется в несколько этапов. На предварительном этап выбираются числа p, q и g такие, что p является простым числом, $2^{509} < p < 2^{512}$, либо $2^{1020} < p < 2^{1024}$; q — простым делителем числа $p - 1$, $2^{254} < q < 2^{256}$; g является числом порядка q, $1 < q < p - 1$. Эти три числа известны группе абонентов. Выбирается закрытый ключ x, $1 \le x \le q$ и вычисляется открытый ключ $y = g^x \pmod p$ для проверки подписи.

Формирование электронной подписи происходит в соответствии со следующим алгоритмом.

- Вычисление значение хеш-функции $H(M)$ от сообщения M. Если значение $H(M) \pmod q = 0$, то присваивается $H(M)$ значение $0^{255}1$, где 0^{255} является 255 кратным повторением нуля.
- Вырабатывается целое число k, $0 < k < q$. Число k снимается с датчика случайных чисел.
- Вычисляется два значения $r_1 = g^k \pmod p$ и $r = r_1 \pmod q$, так, что если $r = 0$, то осуществляется переход ко второму шагу и вырабатывается другое значение k.
- С использованием секретного ключа x абоненты вычисляют значение

$$s = (xr + kH(M)) \pmod q.$$

Если $s = 0$, то осуществляется переход ко второму шагу.

Проверка цифровой подписи возможна при наличии у абонента открытого ключа абонента, пославшего сообщение. Пусть m — полученное сообщение. Уравнение проверки будет иметь вид

$$hm_1 = H(m)^{-1} (\mathrm{mod}\ q);$$

$$r_1 = \left(g^{hm_1 s}\ y^{-r\ hm_1}\ (\mathrm{mod} p)\right)(\mathrm{mod}\ q).$$

Если $r_1 = r$, то считается, что подпись подлинная. В самом деле

$$\left(g^{hm_1 s} y^{-r\ hm_1}\ (\mathrm{mod}\ p)\right)(\mathrm{mod}\ q) = \left(g^{hm_1 s}\ g^{-xr\ hm_1}\ (\mathrm{mod}\ p)\right)(\mathrm{mod}\ q) =$$

$$= \left(g^{hm_1 (s - xr)}\ (\mathrm{mod}\ p)\right)(\mathrm{mod}\ q) = \left(g^{hm_1\ kH(m)}\ (\mathrm{mod}\ p)\right)(\mathrm{mod}\ q) = r.$$

Таким образом, можно удостовериться в подлинности цифровой подписи.

3.9. Скрытый канал

Пусть абоненты *A* и *B* могут обмениваться сообщениями через абонента *C*, но абонент *C* несанкционированно хочет получать их сообщения. Абоненты *A* и *B* мирятся с риском возможного раскрытия, иначе они вообще не смогут общаться, но им нужно согласовать свои планы. Для этого им необходимо найти способ передавать информацию, максимально защищенную от перехвата. Для этого нужно создать скрытый (виртуальный, содержащийся в самих открытых сообщениях) канал связи при передаче по каналу открытых сообщений. С помощью обмена подписанными сообщениями они передают информацию, и абонент *C*, даже если он просматривает все сообщения, не сможет обнаружить в открытых сообщениях наличие скрытой информации.

Рассмотрим идею организации скрытого канала, построенного на основе алгоритма цифровой подписи. Абонент *C* видит, как подписанные сообщения передаются туда и обратно, но реальная передаваемая информация проходит незаметно для него по скрытому каналу. В действительности, алгоритм скрытого канала в подписях не от-

личим от нормального алгоритма в подписях, по крайней мере, для абонента *C*. Он не только не может прочитать сообщение, передаваемое по скрытому каналу, но у него вообще нет представления о существовании такого канала. Рассмотрим реализацию такого алгоритма.

- Абонент *A* создает сообщение, все равно какое.
- Используя общий с абонентом *B* закрытый ключ, абонент *A* подписывает сообщение, пряча свое скрытое сообщение в подписи.
- Абонент *A* посылает подписанное сообщение абоненту *B* и абонент *C* может его прочесть.
- Абонент *C* читает сообщение и проверяет подпись. Не обнаружив ничего подозрительного, он передает подписанное сообщение абоненту *B*.
- Абонент *B* проверяет подпись под сообщением, убеждаясь, что сообщение получено от абонента *A*.
- Абонент *B* игнорирует сообщение и, используя общий с абонентом *A* закрытый ключ, извлекает скрытое сообщение.

3.10. Организация скрытого канала на основе подписи Эль-Гамаля

Рассмотрим организацию оригинальной подписи сообщения. Чтобы подписать сообщение *M* выбирается случайное число *k*, взаимно простое с $p - 1$. Затем вычисляется

$$a = g^k \bmod p,$$

и с помощью расширенного алгоритма Эвклида находится *b* из следующего уравнения:

$$M = (xa + kb) \bmod (p - 1).$$

Подписью является пара чисел *a* и *b*. Случайное значение *k* является неизвестным для несанкционированного абонента. Для проверки подписи нужно убедиться, что

$$y^a a^b \bmod p = g^M \bmod p.$$

Такой алгоритм называется подписью Эль-Гамаля . Она требует нового значения k, и это значение должно быть выбрано случайным образом.

Рассмотрим организацию скрытого канала на основе подписи Эль-Гамаля. Сначала выбирается простое число p и два случайных числа, g и r, меньшие p. Затем вычисляется

$$K = g^r \bmod p.$$

Открытым ключом служат K, g и p. Закрытым ключом является r. Помимо абонента A закрытый ключ r известен и абоненту B. Это число используется не только для подписи сообщения, но и в качестве ключа для отправки и чтения скрытого сообщения.

Чтобы послать скрытое сообщение M в сообщении M' числа M и p должны быть попарно взаимно простыми; кроме того, взаимно простыми должны быть числа M' и $p - 1$. Абонент A вычисляет

$$X = g^M \bmod p$$

и решает следующее уравнение для Y (с помощью расширенного алгоритма Эвклида):

$$M' = rX + MY \bmod (p - 1).$$

Подписью является пара чисел X и Y. Абонент C может проверить подпись Эль-Гамаля. Он убеждается, что

$$K^X X^Y = g^{M'} \bmod p$$

Абонент B может восстановить скрытое сообщение. Сначала он убеждается, что

$$(g^r)^X X^Y = g^{M'} \bmod p.$$

Если это так, то абонент B считает сообщение подлинным, затем для восстановления M он вычисляет

$$M = (Y^{-1}(M' - rX)) \bmod (p - 1).$$

Например, пусть $p = 11$, а $g = 2$. Закрытый ключ r выбирается равным 8. Это означает, что открытым ключом, который абонент C может использовать для проверки подписи, будет

$$g^r \bmod p = 2^8 \bmod 11 = 3.$$

Чтобы отправить скрытое сообщение $M = 9$, используя сообщение $M' = 5$, абонент A проверяет, что 9 и 11, а также 5 и 11 попарно взаимно просты. Он также убеждается, что взаимно просты числа 9 и 11–1 = 10. Поэтому он вычисляет

$$X = g^M (\bmod p) = 2^9 \bmod 11 = 6.$$

Затем он решает следующее уравнение для Y:

$$5 = 8 \cdot 6 + 9 \cdot Y \bmod 10.$$

Решая это уравнение, получим, что $Y = 3$, поэтому подписью служит пара чисел 6 и 3 (X и Y). Абонент B убеждается, что

$$(g^r)^X X^Y = g^{M'} \bmod p;$$

$$(2^8)^6 6^3 = 2^5 (\bmod 11),$$

поэтому он может восстановить скрытое сообщение, вычисляя

$$M = (Y^{-1}(M' - rX)) \bmod (p - 1) = 3^{-1} (5 - 8 \cdot 6) \bmod 10 = 49 \bmod 10 = 9.$$

Таким образом, непосредственно в цифровой подписи может находиться скрытая информация, которую нельзя обнаружить при верификации подписи. Эта идея организации скрытого канала может быть реализована и для других алгоритмов электронной цифровой подписи.

Вопросы и задания

1. Каким свойством обладают члены сверхвозрастающей последовательности?
2. Проверьте вручную на малых простых числах алгоритмы Диффи — Хеллмана и Т. Эль-Гамаля.
3. Для абонентов A и B в алгоритме RSA выберите просдые числа и определите открытые и закрытые ключи.
4. Напишите программу реализующую алгоритмы Диффи — Хеллмана и Хьюза открытого распределения ключей.
5. Напишите программу реализующую протокол обмена зашифрованными ключами EKE

6. Поясните основную идею протоколов доказательства с нулевым разглашением знания.

7. Напишите программу реализующую протокол доказательства знания дискретного логарифма *x* числа *X*.

8. Напишите программу реализующую алгоритм идентификации Гиллу — Кискате.

9. Дайте определение хеш-функции.

10. Напишите программу вычисления хеш-функции SHA-1.

11. Напишите программу вычисления хеш-функции ГОСТ 28147-89.

12. Напишите программу вычисления цифровой подписи по стандарту ГОСТ Р34.10-94.

13. Напишите программу организации скрытого канала на основе подписи Т. Эль-Гамаля.

Вот зрелище, достойное того, чтобы на него взглянул Бог, созерцая свое творение.

Сенека Младший

4. Защита информации в стандарте GSM

Радиосвязь по своей природе является более уязвимой для прослушивания, чем связь по проводам. Сотовый принцип впервые был предложен лабораторией Bell Labs в США. В 1979 г. в Чикаго начала работу первая сотовая сеть в диапазоне частот 800 МГц. Благодаря высокой пропускной способности, эффективности и открытым стандартам GSM был выбран в качестве международного стандарта.

В стандарте GSM (Global System for Mobile Communications) с целью повышения криптографической защиты информации используется алгоритм аутентификации, шифрование поточным шифром и алгоритм генерации ключей. Для исключения несанкционированного использования ресурсов системы связи вводятся и определяются механизмы аутентификации или проверки подлинности абонента [17].

Телефонный разговор в телекоммуникационной системе стандарта GSM передается в цифровой форме в виде последовательности кадров. Каждый кадр передаваемой информации фактически шифруется своим ключом шифрования, которое обеспечивается поточным шифром.

Алгоритм генерации ключей реализован в SIM-карте абонента (Subscriber Identification Module) и используется в механизме аутентификации абонента.

4.1. Обеспечение безопасности передачи информации в телекоммуникационных системах стандарта GSM

Единственной гарантией надежности криптографических алгоритмов является их абсолютная открытость, которая позволяет изучать алгоритмы и находить в них слабые места. Безопасность телекоммуникационной системы стандарта GSM основана на разделении степени защиты между мобильной станцией и ее базовой приемопередающей станцией. Режимы защиты информации в стандарте GSM определяются несколькими рекомендациями (табл.4.1).

Таблица 4.1

Спецификации безопасности стандарта GSM

Спецификация	Название	Описание
GSM 02.09	Аспекты секретности	Определяет характеристики безопасности, применяемые в сетях GSM.
GSM 03.21	Алгоритмы секретности	Определяет криптографические алгоритмы в системе связи
GSM 02.17	Модули подлинности абонента (*SIM*)	Определяет основные характеристики модуля подлинности абонента.

Для изучения принципов организации безопасной передачи информации в системах GSM определим, как взаимодействуют между собой все участники процесса передачи данных.

4.2 Принципы взаимодействия компонентов системы стандарта GSM

В телекоммуникационных системах передачи информации стандарта GSM для обеспечения связь на больших расстояниях использу-

ется сеть смежных радиосот [17]. В каждой соте имеется базовая приемопередающая станция BTS (Base Transceiver Station), работающая на выделенном для нее наборе радиоканалов, которые отличаются от радиоканалов, используемых в соседних сотах, чтобы избежать совпадений (рис 4.1).

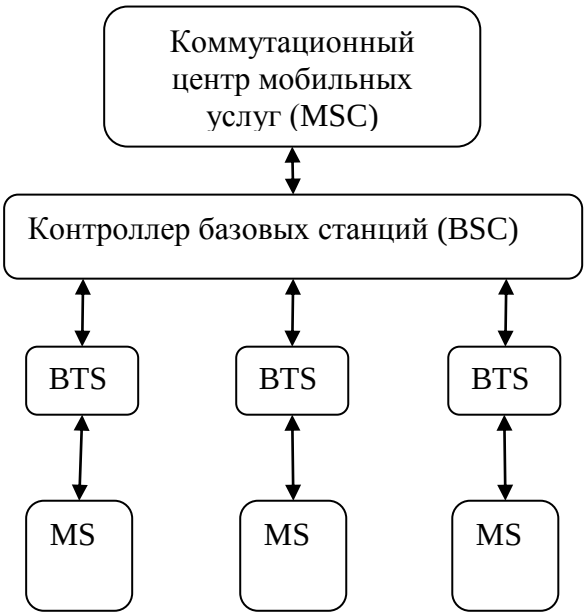

Рис. 4.1. Структура сети стандарта GSM

В структуре сети стандарта GSM основной функцией базовых приемопередающих станций BTS является обеспечение передачи и приема сообщений (информации) по радиоканалу между самой BTS и мобильной станцией MS (Mobile Station, например, телефоном). Все базовые станции логически сгруппированы и управляются контроллером базовых станций BSC (Base Station Controller) для передачи вызова при движении абонента из одной соты в другую (так называемая эстафета) и управления мощностью. Группу контроллеров BSC обслуживает коммутационный центр мобильных услуг MSC (Mobile services Switching Centre). По своему назначению он аналогичен обычному телефонному коммутатору и отвечает за идентификацию и регистрацию абонентов, маршрутизацию вызовов, за интерфейс с традиционными телефонными сетями и т. д. Он направляет вызовы в

телефонную сеть общего пользования, цифровую сеть интегрального обслуживания и другие сети частного или общего пользования, стационарные или мобильные. Коммутационный центр является связующим элементом сети стандарта GSM. Он отвечает за маршрутизацию или коммутацию вызовов от места возникновения к месту их назначения. Коммутационный центр мобильных услуг MSC действует так же, как интерфейс между сетью стандарта GSM, телефонной сетью и сетями данных. Он может быть также соединен с другими коммутационными центрами MSC той же самой сети и с другими сетями стандарта GSM.

4.3 Аутентификация

Для исключения несанкционированного использования ресурсов системы связи определяется механизм аутентификации или проверки подлинности абонента. Аутентификация определяется с помощью алгоритма *A*3. Каждому абоненту системы связи присваивается «временное удостоверение личности» или временный международный идентификационный номер пользователя TMSI (Temporary Mobile Station Indentifer). Если абоненту еще не присвоен временный номер (например, при первом включении мобильной станции), идентификация проводится через международный идентификационный номер IMSI (International Mobile Station Indentifer), который выдается каждому абоненту при его подключении к сети. После окончания процедуры аутентификации и начала режима шифрования временный идентификационный номер TMSI передается на мобильную станцию только в зашифрованном виде. Этот номер TMSI будет использоваться при всех последующих доступах к системе. Необходимая информация об абонентах хранится в базах данных оператора сети абонента.

Рассмотрим более подробно аутентификацию с помощью алгоритма А3. Процедура проверки сетью подлинности абонента реализуется следующим образом. Если мобильная станция регистрируется в

данной сети впервые (рис. 4.2), то она с помощью BTS передает на коммутационный центр MSC свой IMSI.

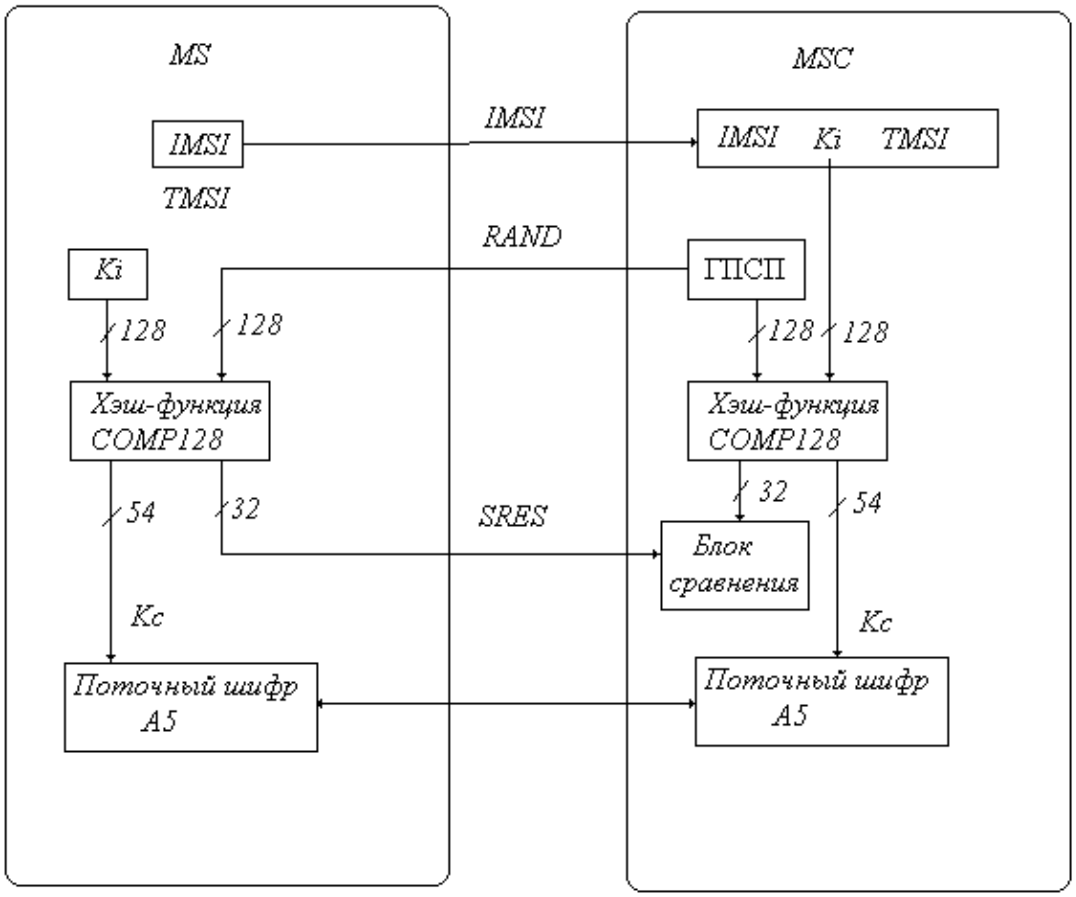

Рис. 4.2. Алгоритм аутентификации A3

В случае, если аутентификация проводится не в первый раз, то на коммутационный центр передается идентификационный номер TMSI. Коммутационный центр мобильных услуг MSC производит поиск по IMSI или TMSI в своей базе данных и определяет ключ K*i*, соответствующий данному абоненту. Коммутационный центр мобильных услуг MSC с помощью генератора псевдослучайных последовательностей (ГПСП) генерирует случайное число RAND**,** которое передается на мобильную станцию абонента.

Мобильная станция определяет значение отклика SRES с помощью вычисления хэш-функции по алгоритму COMP128, используя RAND и Ki, после чего станция отсылает этот отклик в коммутацион-

ный центр мобильных услуг. Центр MSC также определяет значение отклика, используя RAND и значение Ki из своей базы данных, и сравнивает полученное значение с откликом, принятым от мобильной станции. Если они совпали, то аутентификация прошла успешно. Далее мобильная станция и центр коммутации переходят в режим шифрования, в котором если мобильная станция аутентифицировалась впервые, то ей в зашифрованном виде передается номер TMSI.

Часть выходного значения идентификационного номера значением отклика SRES и служит для аутентификации (32 бита), а часть (54 бита) — для формирования ключа шифрования Kc (алгоритм A8), который используется алгоритмом шифрования поточным шифром A5. Фактически аутентификация и генерация ключа шифрования происходят одновременно. Алгоритмы A3 и A8 в большинстве случаев используют хеш-функцию COMP128, на вход которой подаются значения Ki и RAND. Такой подход используется как при регистрации в домашней сети, так и при регистрации во внешней сети. По причине секретности вычисление SRES и Kc происходит в рамках SIM-карты.

4.4. Алгоритмы аутентификации *A*3 и генерации ключа *A*8. Хеш-функция COMP128

Алгоритмы A3 и A8 рассматриваются вместе, так как, по сути, они представляют собой один алгоритм, который одновременно проводит аутентификацию абонента и генерацию ключа шифрования. Оба алгоритма используют одну и ту же хеш-функцию COMP128, на вход которой поступают значения RAND и Ki, а на выходе получаются значения SRES и Kc (рис 4.2).

Алгоритмы аутентификации *A*3 и генерации ключа *A*8 основаны на разделении секрета мобильной станцией и базовой станцией, т.к. обеим сторонам известно значение секретного ключа *Ki*. Рассмотрим более подробно алгоритм аутентификации *A*3, который осуществляется в несколько этапов.

- Базовая станция с помощью генератора псевдослучайных кодов формирует случайное число RAND, которое передается на мобильную станцию.
- Из 128-разрядного числа RAND и 128-разрядного секретного ключа Ki мобильная станция формирует 256-разрядное входное значение для хэш-функции COMP128. Число RAND составляет младшие биты входного значения, а секретный ключ Ki — старшие. Далее хеш-функция COMP128 формирует выходное 128-разрядное значение, но реально используются только 96 разрядов. Старшие 32 разряда являются значением отклика SRES, а младшие 54 разряда определяют значение ключа шифрования Kc. Отклик SRES пересылается базовой станции.
- Базовая станция проводит такие же операции, как и мобильная и сравнивает свое значение отклика со значением, принятым от мобильной станции. Если значения совпали, то считается, что мобильной станции известен секретный ключ Ki, и, значит, аутентификация прошла удачно. После этого можно начать шифрование информации, так как обе стороны будут использовать один ключ шифрования.

Таким образом, в результате аутентификации получается ключ шифрования Kc, который могут использовать для шифрования как мобильная, так и базовая станции.

Хеш-функция COMP128 принимает 256-разрядное значение и формирует 128-разрядное. Работа этой функции основана на пяти таблицах различного размера, а именно: $T_0[0..511]$, $T_1[0..255]$, $T_2[0..127]$, $T_3[0..63]$ и $T_4[0..31]$. Выходные значения этих таблиц имеют размер 8, 7, 6, 5 и 4 бит соответственно. Формируется входной 256-битовый вектор X, являющийся конкатенацией Ki и RAND

$$X = \mathrm{Ki} \,//\, \mathrm{RAND}. \qquad (5.1)$$

Этот же вектор X можно представить и в другой эквивалентной форме записи:

$$X = \xi_{32} \,//\, \xi_{31} \,//\, \ldots \,//\, \xi_1,$$

где ξ_k, k = 1, 2, 3, …, 32 — 8-битовые вектора. Главный цикл вычисления хэш-функции может быть описан следующим образом

- Переменная i последовательно за 5 шагов принимает значения от 0 до 4 с шагом 1, что соответствует использованию на каждом шаге другой таблицы T_i.
- На каждом шаге по i формируется внутренний цикл по вспомогательной переменной j, принимающей значения от 0 до $2^i - 1$ с шагом 1.
- На каждом шаге по j формируется еще один внутренний цикл по вспомогательной переменной k, принимающей значения от 0 до $2^{4-i} - 1$ с шагом 1.
- На каждом шаге по k вычисляются вспомогательные переменные s, t, x, y согласно выражениям

$$s = k + j\,2^{5-i};$$
$$t = s + 2^{4-i};$$
$$x = (\xi_s + 2\xi_t) \pmod{2^{9-i}};$$
$$y = (2\,\xi_s + \xi_t) \pmod{2^{9-i}};$$
$$\xi_s = T_i[x];$$
$$\xi_t = T_i[y].$$

Этот процесс повторяется 9 раз, и, поскольку таблица $T4$ имеет выходной размер 4 бита, в итоге в массиве X оказываются 32 полубайта, что соответствует 128-битовому выходному значению хэш-функция COMP128

$$X = \xi_{32} // \xi_{31} // \ldots // \xi_1,$$

где ξ_k, k = 1, 2, 3, …, 32 — 4-битовые вектора.

4.5. Шифрование поточным шифром A5

Телефонный разговор [14] в формате стандарта GSM передается в виде последовательности кадров. Каждый кадр передаваемой информации фактически шифруется своим ключом шифрования. Шифрование обеспечивается поточным шифром A5. Для шифрования каждого кадра алгоритм A5 инициализируется ключом Kc, полученным

обычно на стадиях аутентификации алгоритмом A8, и последовательным номером передаваемого кадра. Последовательный номер кадра является открытой информацией и может быть использован для расшифровывания разговора. Ключ Kc используется до следующей аутентификации абонента. Разговор шифруется только на этапе передачи информации между мобильной станцией *MS* и базовой приемопередающей станцией BTS. Для установки режима шифрования BTS передает мобильной станции команду на переход в режим шифрования. После получения команды CMC (Ciphering Mode Command) мобильная станция, используя имеющийся у нее ключ Kc, приступает к шифрованию и дешифрированию сообщений. Алгоритм A5 описан в стандарте GSM 03.20. Этот алгоритм существует в двух модификациях A5/2 и A5/1. Алгоритм A5/1 обладает повышенной стойкостью. Рассмотрим наиболее распространенный алгоритм A5/2. Генератор ПСП (псевдослучайной последовательности) A5/2 состоит из трёх регистров с линейной обратной связью [6], которые обозначим как *R*1, *R*2 и *R*3. Образующие многочлены этих регистров имеют вид:

$$x^{19} + x^{18} + x^{17} + x^{14} + 1;$$
$$x^{22} + x^{21} + 1;$$
$$x^{23} + x^{22} + x^{21} + x^{8} + 1$$

соответственно.

Выходные биты снимаются со старших разрядов этих регистров и складываются по модулю 2, образуя выходной бит генератора ПСП. В алгоритме A5/2 используется изменяемое управление тактированием. Сдвигом регистров управляет функция majority, которая имеет следующий вид:

$$\text{majority}(x_1, x_2, x_3) = x_1 \wedge x_2 \oplus x_1 \wedge x_3 \oplus x_2 \wedge x_3, \qquad (5.2)$$

где $\oplus$ — операция побитового сложения по модулю 2, $\wedge$ — операция «и». Результатом работы этой функции является один бит, который определяет, какие регистры будут тактироваться (сдвигаться), а какие нет, т. е. если бит с выхода функции совпадает со значением бита регистра на определенной позиции, то в регистре осуществляется сдвиг

на один такт. На вход функции majority значения x_1, x_2, x_3 поступают значения битов с номерами 3, 7, 10 четвертого регистра $R4$. Образующий многочлен этого регистра имеет вид

$$x^{17} + x^5 + 1,$$

причем каждый бит отвечает за тактирование одного из трех регистров $R1$, $R2$ и $R3$. Обычно на каждом этапе тактируются два из трех регистров. Регистр $R4$ тактируется на каждом шаге.

Вопросы и задания

1. Как осуществляется аутентификация в стандарте GSM?
2. Какие шифры используются в стандарте GSM?
3. Сравните криптостойкость различных шифров применяемых в стандарте GSM
4. Поясните, на каком принципе реализован генератор псевдослучайной последовательности в шифре A5.
5. Какой период повторения регистров $R1$, $R2$ и $R3$?
6. Напишите программу реализующую работу регистров сдвига, применяющихся в аутентификации в стандарте GSM.
7. Напишите программу реализующую алгоритм работы функции majority в шифре A5/2.
8. Получите алгоритм работы и напишите программу шифра A5/2.
9. Получите алгоритм работы и напишите программу шифра A5/1.

Всякая человеческая голова подобна желудку: одна переваривает входящую в оную пищу, а другая от нее засоряется.

Козьма Прутков

5. Защита информации в телекоммуникационных системах стандартов UMTS и CDMA2000

Телекоммуникационные системы, использующие стандарты UMTS (Universal Mobile Telecommunications System) относятся к третьему поколению систем мобильной связи [10], называемому поколением 3*G*. Для мобильной связи третьего поколения используется дециметровый диапазон частот (около 2 ГГц), и обеспечивается передача данных со скоростью 2 Мбит/с. Телекоммуникационные системы поколения 3G используются для видеотелефонной связи, цифрового сотового телевидения и др. Для телекоммуникационных систем разработаны два стандарта сетей поколения 3G: стандарт UMTS и стандарт CDMA2000. Стандарт UMTS распространён в основном в Европе, а стандарт CDMA2000 — в Азии и США. В принципе эти два стандарта предполагают два различных подхода к организации сетей поколения 3G. В стандарте CDMA2000 подразумевается сохранение частот и постепенный переход к новым технологиям, путём наращивания технических мощностей оператора. Стандарт UMTS является совершенно новый стандартом, в то время как разновидности стандарта CDMA, предложенные для поколения 3G, являются развитием уже эксплуатирующейся в мире технологии второго поколения.

5.1. Механизмы обеспечения безопасности в стандарте UMTS

В стандарте UMTS определены требования обеспечения конфиденциальности, которые обеспечивают:

- защиту от вторжений на линии радиодоступа, обеспечивают для абонентов конфиденциальный доступ к новым услугам систем связи 3-го поколения;
- защиту от определения посторонним абонентом информации о номере IMSI;
- надежную аутентификацию пользователей в сети, а также конфиденциальность передаваемой шифрованной информации.

Рассмотрим подробнее процесс аутентификация в стандарте UMTS. Идентификатором мобильной станции является международный идентификационный номер подвижного абонента IMSN, или временный идентификационный номер подвижного абонента TMSN, который используется при выполнении сетью процедуры аутентификации абонента.

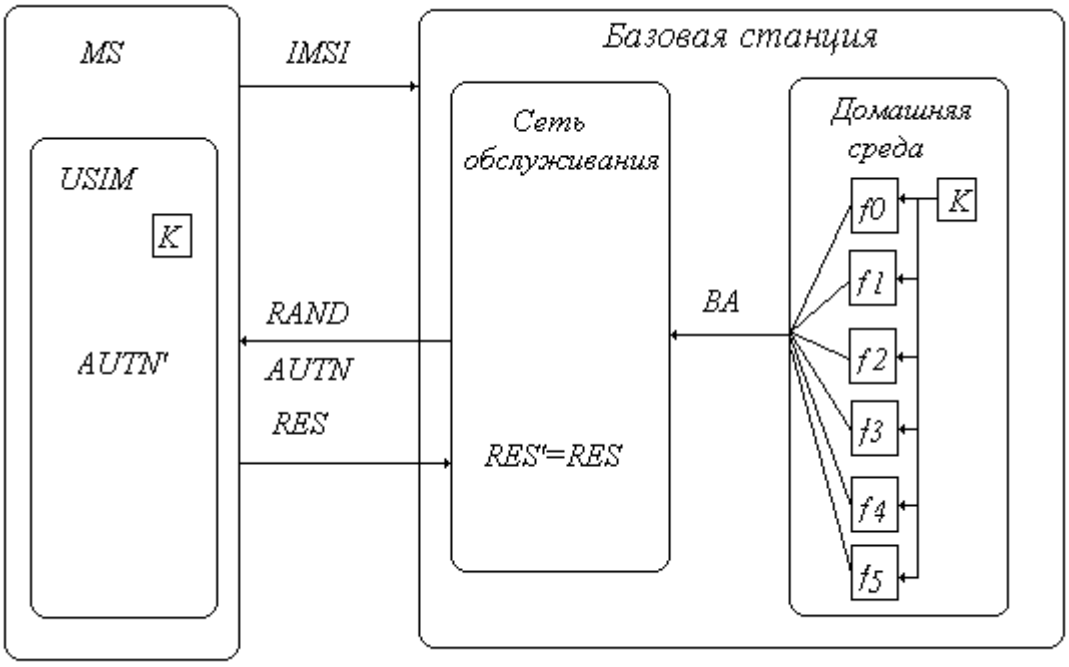

Рис. 5.1. Аутентификация в стандарте UMTS

При передаче идентификационного номера подвижного абонента IMSN или временного идентификационного номера подвижного абонента (TMSN) на базовую станцию начинается процедура аутентификации. Данная процедура выполняется в два этапа (рис. 5.1).

На первом осуществляется передача вектора аутентификации (BA) от домашней среды в сеть обслуживания. Этот вектор, содержащий необходимые для осуществления аутентификации параметры и основные криптографические ключи, должен быть надежно защищен от перехвата и модификации. Каждый вектор аутентификации содержит:

- случайное число RAND;
- ожидаемый отзыв RES;
- ключ шифрования, $K_Ш$;
- ключ целостности, $K_Ц$;
- параметр (метка) аутентификации сети AUTN.

Конфиденциальность и целостность передачи BA обеспечиваются использованием протокола MAP (Mobile Application Part). Этот протокол отвечает за секретность на прикладном уровне независимо от используемых сетевых и транспортных протоколов. Для вычисления параметров вектора аутентификации используются пять односторонних функций *f*1, *f*2, *f*3, *f*4, *f*5 (рис. 5.2.). На этом рисунке обозначено:

- SQN (Sequence Number) порядковый номер аутентификации служит для защиты от угрозы «повтор»;
- AK (Anonymity Key) ключ, используемый для шифрования SQN;
- AMF (Authentication Management Field) административное управляющее поле аутентификации. AMF может быть использовано для указания многократного шифрования;
- MAC (Message Authentication Code) код аутентификации сообщения. В данном случае

- MAC = *f*1(K, AMF, SQN, RAND), где *f*1 является алгоритмом формирования кода аутентификации сообщения, включающего K, AMF, SQN, RAND.

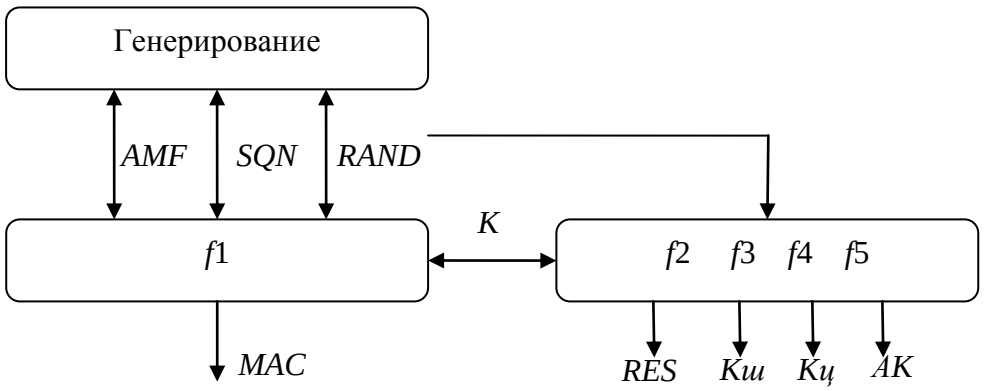

Рис. 5.2. Формирование вектора аутентификации

Вектор аутентификации ВА имеет вид:

$$BA = RAND//RES//Кш//Кц//AUTN,$$

AUTN — параметр аутентификации сети, который имеет аналитическую запись

$$AUTN = SQN \oplus AK//AMF//MAC.$$

На втором этапе аутентификации и ключевого соглашения сеть обслуживания выполняет процедуру «вызов — ответ» для взаимного установления подлинности между модулем идентичности абонента USIM (UMTS Subscriber Identity Module) и телекоммуникационной сетью. Помимо аутентификации данная процедура обеспечивает генерирование ключей шифрования и защиты целостности. Программные модули, отвечающие за выполнение процедур аутентификации и ключевого соглашения, размещены в USIM и в центре аутентификации. В них используются криптографические функции *f*0, ..., *f*5. Основные определенные для UMTS криптографические функции и алгоритмы (наряду с их назначением) приведены в табл. 5.1.

Таблица 5.1

Функции безопасности в стандарте UMTS

Функция	Назначение
*f*0	Функция генерирования случайного числа
*f*1	Функция аутентификации сети
*f*2	Функция аутентификации абонента
*f*3	Функция генерирования ключа шифрования $K_{Ш}$
*f*4	Функция генерирования ключа целостности $K_{Ц}$
*f*5	Функция генерирования ключа анонимности АК
*f*6	Алгоритм шифрования, предназначенный для обеспечения конфиденциальности данных, передаваемых между домашней средой и сетью обслуживания
*f*7	Алгоритм целостности, предназначенный для обеспечения контроля целостности данных на участке «домашняя среда — сеть обслуживания»
*f*8	Алгоритм шифрования, предназначенный для шифрования данных, передаваемых на участке «сеть обслуживания — мобильная станция»
*f*9	Алгоритм целостности, предназначенный для обеспечения контроля целостности данных на участке «сеть обслуживания — мобильная станция»

В схеме взаимной аутентификации сети обслуживания и модуля идентичности абонента USIM используется предварительно распределенный секретный ключ K длиной 128 бит, размещаемый в USIM и в центре аутентификации домашней среды. Одним из условий обеспе-

чения безопасности информации в системе является использование секретного ключа на протяжении всего срока службы USIM. Функция $f0$ генерирования случайного числа RAND локально размещается в центре аутентификации домашней среды. Немаловажно, чтобы значение RAND также не повторялось в течение срока службы USIM. В противном случае, зная значение RAND, использованное в одном из предшествующих сеансов связи, теоретически возможно восстановить отклик RES, что повлечет за собой нарушение конфиденциальности передаваемой информации.

Процедура аутентификации и ключевого соглашения поначалу инициируется сетью обслуживания, которая передает на мобильную станцию сообщение вызова, содержащее случайное число RAND и символ установления подлинности AUTN. Суть этой процедуры заключается в установлении подлинности вызова, получаемого MS, который в действительности можно сгенерировать, зная секретный ключ K. С учетом имеющихся ограничений на время установления соединения выбрана процедура аутентификации и ключевого соглашения с вызовом и ответом при использовании кодов аутентификации. Это решение позволило минимизировать вычислительные затраты, что важно, поскольку функции $f1$ и $f2$ выполняются в модуле идентичности абонента USIM. Схема взаимной аутентификации системы UMTS основана на процедуре, описанной в [11].

Получив вызов, модуль идентичности абонента USIM проверяет подлинность сети. Это осуществляется путем вычисления значения $\mathrm{AUTN}' = f(\mathrm{K}, \mathrm{SQN}, \mathrm{RAND})$. Далее в модуле идентичности абонента USIM сравниваются значения AUTN' и AUTN (последнее получено в составе вектора аутентификации). В случае их равенства подлинность сети считается установленной. Затем вычисляется ожидаемый отзыв RES с использованием функции $f2$ в модуле идентичности абонента USIM. В этом случае $\mathrm{RES} = f2(\mathrm{K}, \mathrm{RAND})$, который необходим для получения мобильной станцией доступа в сеть. Одновременно вычис-

ляются ключи шифрования $K_Ш$ и обеспечения целостности $K_Ц$ (табл. 5.1):

$$K_Ш = f2\ (K, RAND);\ K_Ц = f4\ (K, RAND).$$

После выполнения этих процедур в сети обслуживания сравниваются значения RES с RES′, которые также содержится в векторе аутентификации (рис. 5.1). Подлинность оборудования пользователя считается установленной, если RES = RES′. В процессе аутентификации и ключевого соглашения может использоваться ключ анонимности АК, генерируемый с помощью функции *f*5:

$$АК = f5\ (K, RAND).$$

Этот ключ необходим для сокрытия значения SQN, являющегося составной частью вектора аутентификации. Сокрытие SQN, усложняющее несанкционированное отслеживание местоположения абонента [10], реализуется суммированием по модулю 2 самого значения SQN со значением ключа АК (табл. 5.1). Очевидно, функция *f*5 в данном случае должна выполняться раньше, чем функция *f*1. Конфиденциальность информационного обмена в стандарте UMTS, обеспечивается на участке между мобильной станцией и базовой станцией и возлагается на функцию шифрования *f*8, описанную в [14] как симметричный потоковый шифр. При этом из базовой сети передается ключ шифрования, который после аутентификации используется базовой сетью и оборудованием абонента совместно.

В архитектуре безопасности предусматривается опция выбора из 16 различных алгоритмов шифрования. Указатель на используемый алгоритм содержится в идентификаторе алгоритма шифрования (UEA). Стандартный алгоритм обозначен UEA1, а «пустой алгоритм», соответствующий отсутствию шифрования — UEA0. Криптографическое ядро построено на базе шифра KASUMI [13], симметричного блочного шифра на основе конструкции Фейстеля (см. разд. 2). Размер шифруемого блока данных равен 64, а длина ключа шифрования 128 бит. Шифр KASUMI функционирует в режиме обратной связи, где функция *f*8 используется для генерирования блоков ключевого по-

тока, побитно суммируемых с блоками открытого текста по модулю 2. Шифр KASUMI является модификацией алгоритма MISTY1 разработанного в 1996 г. японской компанией Mitsubishi Electric, поэтому рассмотрим подробнее алгоритм MISTY1.

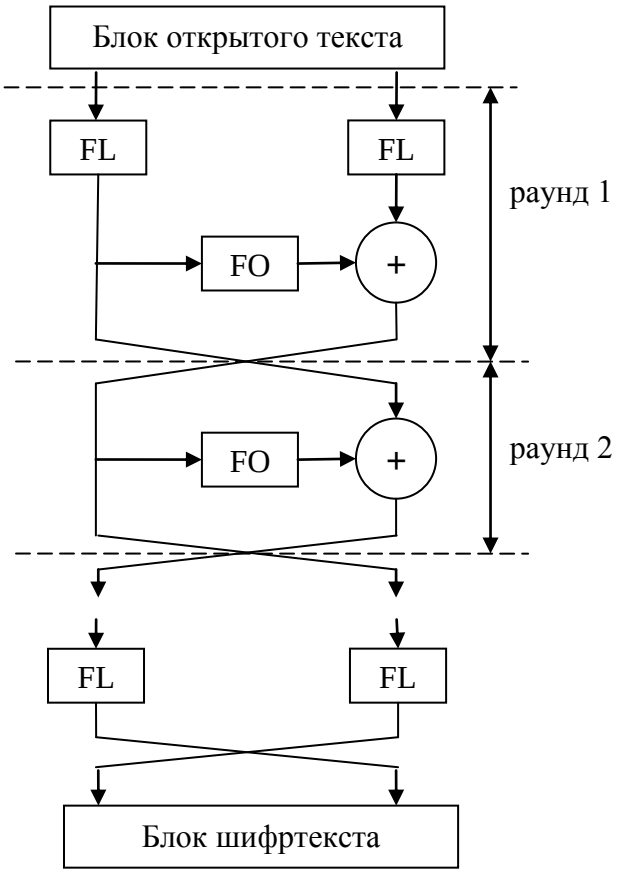

Рис. 5.3. Последовательность шифрования данных по алгоритму MISTY1

На рис. 5.3 представлена последовательность операций, выполняемых при шифровании данных по алгоритму MISTY1. Как видно, алгоритм является алгоритмом Фейстеля с различными четными и нечетными раундами. В нечетных раундах, по сравнению с четными раундами выполняется дополнительно операция FL, которая также выполняется и по завершении последнего раунда. Количество раундов равно восьми.

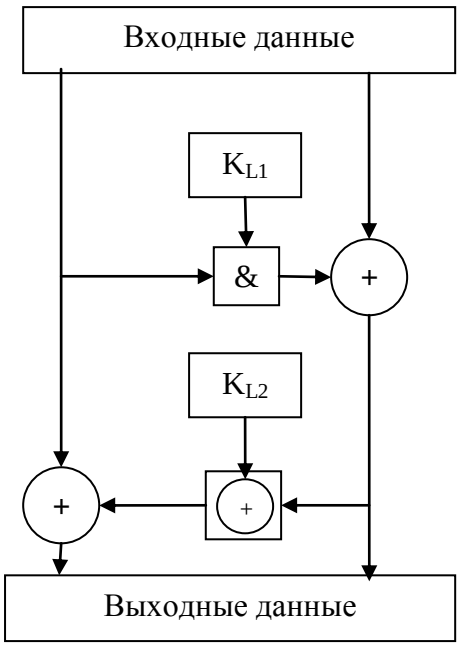

Рис. 5.4. Алгоритм операции FL

Алгоритм операции FL представлен на рис. 5.4, где обозначение «&» соответствует выполнению операции побитого логического «и», а обозначение «⊕» соответствует выполнению операции побитового сложения по модулю 2, K_{L1} и K_{L2} — ключи шифрования. Таким образом, при выполнении операции FL 32-битовое входное значение разбивается на два блока по 16 бит, над которыми выполняются операции сложения по модулю 2 и побитные логические операции, причем в логических операциях участвуют определенные фрагменты расширенного ключа шифрования для операции K_{L1} и K_{L2}. Более сложной является операция FO, алгоритм которой показан на рис. 5.5. В данной операции 32-битовое входное значение также делится на два 16 битных фрагмента, один из которых обрабатывается операцией сложения по модулю 2 с определенным фрагментом ключа для операции FO (K_{Ox}) и операцией FI, после чего значения данных в этих блоках складываются по модулю 2 и меняются местами. Фактически, схема на рис. 5.5 является трехраундовой сетью Фейстеля с дополнительным сложением по модулю 2 после финального раунда.

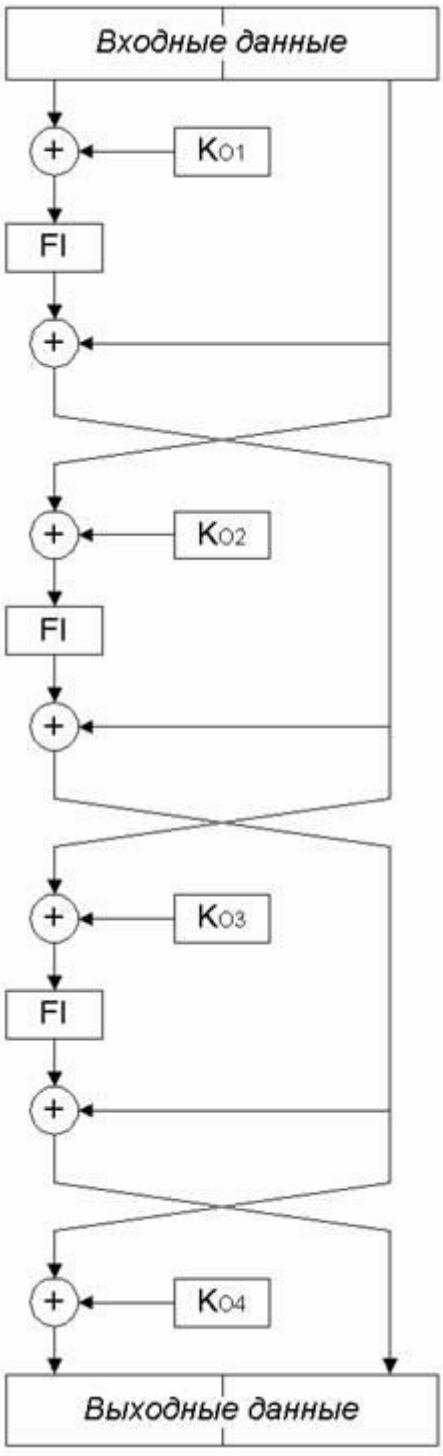

Рис. 5.5. Алгоритм операции FO

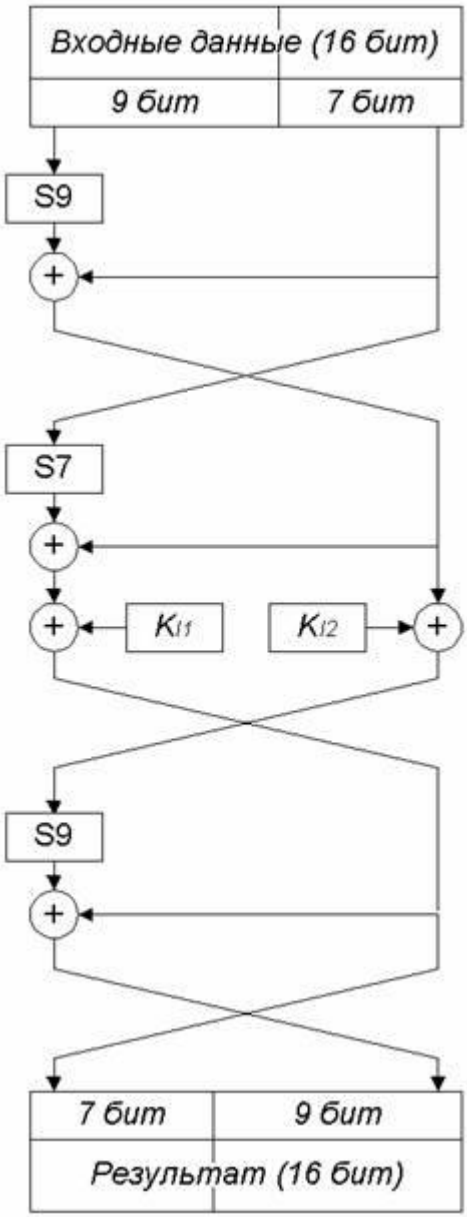

Рис. 5.6. Алгоритм операции FI

В отличие от операций FL и FO, операция FI обрабатывает 16-битные блоки входных данных (рис. 5.6). Это 3-раундовая сеть Фейстеля, обрабатывающая 7-битный и 9-битный блоки, над которыми выполняются следующие операции.

- Табличные замены *S*7 и *S*9 над 7- и 9-битным блоком соответственно.
- Сложение блоков между собой по модулю 2, причем в первом и третьем раундах 7-битный блок дополняется нулями, и ре-

зультатом сложения является 9-битный блок, а во втором раунде при сложении отбрасываются два левых бита 9-битного блока и результатом сложения является 7-битный блок.

- Сложение по модулю 2 соответствующих блоков с 7- и 9-битным фрагментами расширенного ключа шифрования K_{I1} и K_{I2}, выполняемое во втором раунде.

Размерность ключа алгоритма составляет 128 бит. Перед началом шифрования выполняется процедура вычисления фрагментов ключа, используемые в операциях FL, FO и FI. Рассмотренный вариант алгоритма MISTY1 назван алгоритмом KASUMI, является стандартом шифрования мобильной связи стандарта UMTS и реализован в функции шифрования *f*8.

Обеспечение целостности информации в телекоммуникационных системах стандарта UMTS охватывает передачу данных между мобильной станцией и базовой станцией. В архитектуре безопасности UMTS выделяют 16 алгоритмов обеспечения целостности, указанные через идентификатор алгоритма целостности (UIA). Стандартный алгоритм обозначен как UIA1. Стандартная функция целостности *f*9 также реализована на базе блочного шифра KASUMI. На вход *f*9 наряду с подлежащим защите сообщением поступает ключ целостности $K_{Ц}$ длиной 128 бит (рис. 5.7). Вычисленное значение $MAC\__{Ц}$ отправитель включает в передаваемое сообщение. На приеме принявший его абонент вычисляет $MAC\__{Ц}$. Целостность переданных данных считается подтвержденной, если вычисленный $MAC\__{Ц}$ и полученный $MAC\__{Ц}$ идентичны.

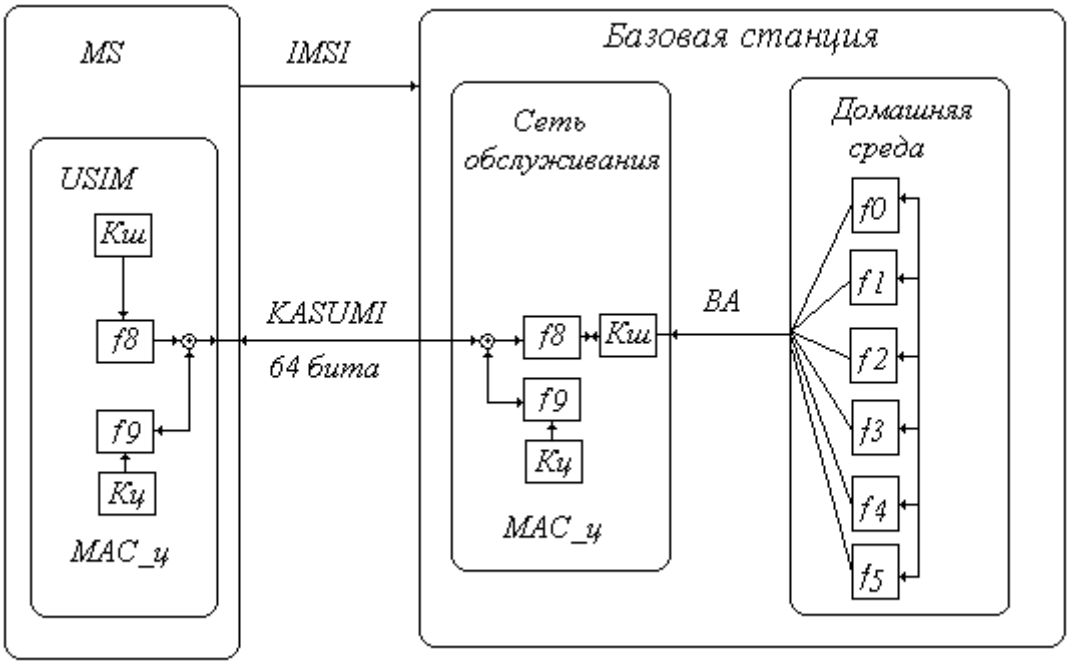

Рис. 5.7. Шифрование сообщений в стандарте UMTS

5.2. Механизмы обеспечения безопасности в стандарте CDMA2000

Обобщенная архитектура безопасности в стандарте сотовой связи CDMA2000 показана на рис. 5.8. В системах стандарта CDMA2000 предусмотрено использование как несъемного (UIM), так и съемного (R_UIM) (removable) модуля идентичности, а идентификатор оборудования пользователя носит название IMSN (TMSN).

Процедура аутентификации и ключевого соглашения в CDMA2000 подобна аналогичной процедуре UMTS с изложенными далее дополнениями и уточнениями. Используемые при этом алгоритмы определены в [11]. Процедура реализуется в два этапа. На первом этапе вектор аутентификации от домашней среды передается в сеть обслуживания. В стандарте CDMA2000 предусмотрено необязательное дополнение процедуры аутентификации и ключевого соглашения функцией *f*11 генерирования ключей аутентификации UIM

(K_{AU}) и функцией UMAC алгоритма аутентификации UIM, преобразующей с помощью K_{AU} аутентификации сообщения (рис. 5.8).

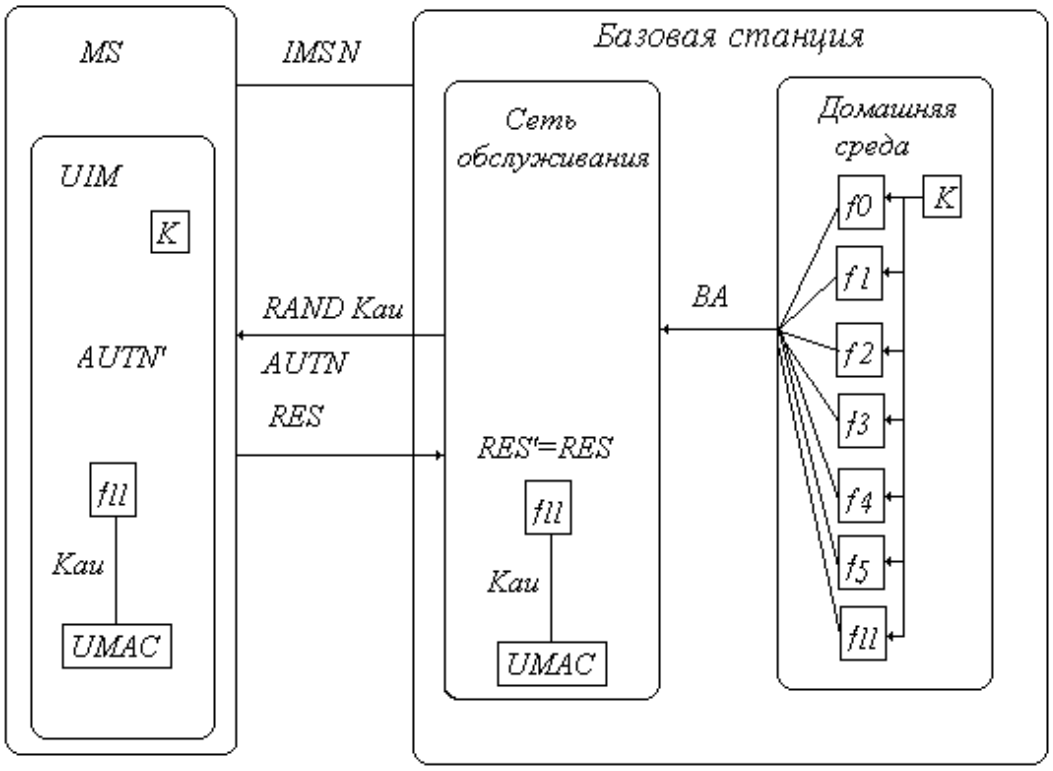

Рис. 5.8. Аутентификация в стандарте CDMA2000

В абонентских терминалах с несъемным UIM алгоритмы аутентификации и ключевого соглашения реализуются в виде стандартизированной встроенной функции, что гарантирует возможность использования мобильной станции при смене оператора. В терминалах со съемным модулем идентичности абонента R_UIM оператор может использовать собственный набор алгоритмов аутентификации и ключевого соглашения. Функции и алгоритмы безопасности, имеющиеся в CDMA2000, приведены в табл. 5.2.

Рассмотрим процесс аутентификации мобильного абонента. После получения вектора аутентификации в несъемном модуле идентичности UIM инициируется выполнение функции *f*5, генерирующей

ключ анонимности, и расшифровывается составная часть вектора аутентификации SQN. Значение SQN используется в качестве входа функции для *f*1, генерирующей параметр аутентификации сети AUTN′. После сравнения AUTN′ и AUTN с целью вычисления RES, $K_{Ш}$, $K_{Ц}$, и K_{AU} выполняются функции соответственно *f*2, *f*3, *f*4, и *f*11. После этого значение RES передается в сеть для подтверждения подлинности абонента. Значения $K_{Ш}$ и $K_{Ц}$ передаются от UIM к терминалу, где используются для обеспечения конфиденциальности и целостности информации. Первое различие двух рассматриваемых систем заключается в возможности использования в стандарте CDMA2000 как встроенного, так и съемного модуля идентичности абонента (R_UIM). Второе отличие заключается в том, что алгоритмы безопасности CDMA2000 полностью стандартизированы, в то время как в стандарте UMTS лишь частично. Третье отличие заключается в наличии алгоритма аутентификации UIM и функции генерирования ключа аутентификации UIM.

В целях обеспечения конфиденциальности информационного обмена в стандарте CDMA2000 используется стандарт шифрования AES [5] (также известный как Rijndael [7]), представляющий собой симметричный блочный шифр, зашифровывающий блоки данных длиной 128 бит под управлением 128 битового ключа. В результате шифрования формируются блоки ключевого потока, применяемые для потокового шифрования/расшифровывания данных (рис. 5.9).

Защита конфиденциальности осуществляется на участке мобильная станция базовая станция. Длина ключа шифрования равна 128 бит.

Таблица 5.2

Функции безопасности в стандарте CDMA2000

Функция	Назначение
*f*0	Функция генерирования случайного числа
*f*1	Функция аутентификации сети
*f*2	Функция аутентификации абонента
*f*3	Функция генерирования ключа шифрования $K_{Ш}$
*f*4	Функция генерирования ключа целостности $K_{Ц}$
*f*5	Функция генерирования ключа анонимности AK
*f*6	Алгоритм шифрования, предназначенный для обеспечения конфиденциальности данных, передаваемых между домашней средой и сетью обслуживания
*f*7	Алгоритм целостности, предназначенный для обеспечения контроля целостности данных на участке «домашняя среда — сеть обслуживания»
*f*8	Алгоритм шифрования, предназначенный для шифрования данных, передаваемых на участке «сеть обслуживания — мобильная станция»
*f*9	Алгоритм целостности, предназначенный для обеспечения контроля целостности данных на участке «сеть обслуживания — мобильная станция»
*f*11	Функция генерирования ключа K_{AU} аутентификации UIM
UMAC	Алгоритм аутентификации UIM
ESP AES	Алгоритм шифрования
EHMAC	Усовершенствованный алгоритм обеспечения целостности HMAC

В стандарте шифрования AES некоторые операции выполняются над байтами, которые рассматриваются как элементы поля $GF(2^8)$. В качестве неприводимого многочлена выбран многочлен $f(x) = x^8 + x^4 + x^3 + x + 1$. Примитивным элементом поля является элемент $\alpha = x + 1$. *В* $GF(2^8)$ справедливо $\alpha^{255} = 1$. Соотношения между различными формами представления элементов поля $GF(2^8)$ приведено в табл. 5.3.

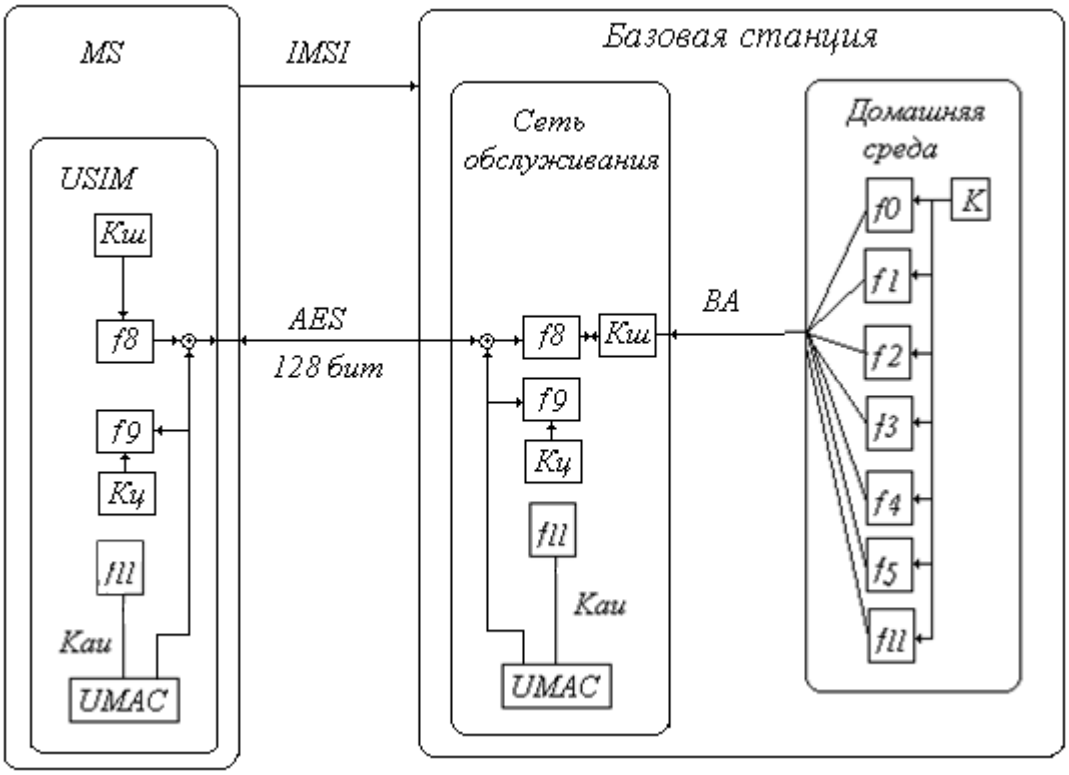

Рис. 5.9 Шифрование данных в стандарте CDMA2000

Шифр AES является итерационным блочным шифром, имеющим переменную длину блоков и различные длины ключей. Длины ключа и длины блоков могут быть равны, независимо друг от друга, величинам 128, 192 или 256 бит. В рассматриваемом стандарте используется шифр AES-128.

Таблица 5.3

Форма представления элементов поля $GF(2^4)$

Степенная форма	Полиномиальная форма	Векторная форма
α^0	1	(00000001)
α^1	$x + 1$	(00000011)
α^2	$x^2 + 1$	(00000101)
α^3	$x^3 + x^2 + x + 1$	(00001111)
….	....	…
α^{253}	$x^6 + x^4 + x + 1$	(01010010)
α^{254}	$x^7 + x^6 + x^5 + x^4 + x^2 + x$	(11110110)

В шифре AES-128 приняты следующие обозначения: Nb — число 32-битовых слов, содержащихся во входном блоке данных ($Nb = 4$), Nk — число 32-битовых слов, содержащихся в ключе шифрования ($Nk = 4$), Nr — количество раундов шифрования ($Nr = 10$). Промежуточные результаты преобразований, выполняемых в рамках криптографического алгоритма, называются состояниями (State). Состояние можно представить в виде прямоугольного массива байтов. Этот массив имеет 4 строки и 4 столбца, где каждый столбец представляет собой 32-битовое слово. Тогда обрабатываемый входной блок данных можно представит в виде четырех слов:

$$w_0 = s_{00} // s_{10} // s_{20} // s_{30};$$
$$w_1 = s_{01} // s_{11} // s_{21} // s_{31};$$
$$w_2 = s_{02} // s_{12} // s_{22} // s_{32};$$
$$w_3 = s_{03} // s_{13} // s_{23} // s_{33}.$$

Этот же блок данных можно представить в виде матрицы состояния State размера 4 × 4 байта. Ключ шифрования также представлен в виде прямоугольного массива с четырьмя строками.

Каждый раунд состоит из 4 преобразований: замена байтов, сдвиг строк, перемешивание столбцов, добавление раундового ключа.

- Замена байтов (ByteSub). Данное преобразование нелинейное и выполняется с каждым байтом состояния. Таблицы замены

S-блока являются инвертирумыми и построены из композиции двух преобразований.

- Преобразование сдвига строк (ShiftRow). Нулевая строка состояния не сдвигается. Последние 3 строки циклически сдвигаются на различное число байтов.
- Преобразование перемешивания столбцов (MixColumn). В этом преобразовании столбцы состояния рассматриваются как многочлены над $GF(2^8)$ и умножаются по модулю $x^4 + 1$.
- Добавление раундового ключа (AddRoundKey). В данной операции раундовый ключ добавляется к состоянию посредством простой поразрядной операции логического побитового сложения по модулю 2.
- Раундовый ключ формируется из ключа шифрования посредством алгоритма получения ключей (Key Schedule).

Для достижения целостности передаваемой информации в стандарте CDMA2000 вычисляется код аутентификации сообщения MAC (Message Authentication Code), отвечающий также за защиту сведений от преднамеренной модификации. Размер кода аутентификации сообщения MAC определяется важностью сообщения и при необходимости усекается к требуемому значению, но не может быть меньше 32 бит. Для сообщений приоритетной значимости взамен кода аутентификации сообщения MAC используется алгоритм аутентификации UIM (UMAC), при вычислении которого необходимо использовать не только ключ целостности $K_Ц$, но и ключ аутентификации K_{AU}. Вычисление значения UMAC может быть осуществлено только непосредственно самим модулем идентичности абонента UIM (рис. 5.9). В случае предоставления домашней средой ключа аутентификации K_{AU} в виде составной части вектора аутентификации и при готовности сети обслуживания к его приему мобильная станция определяет значение UMAC и помещает его в соответствующий пакет данных.

Вопросы и задания

1. В чем отличие механизма аутентификации в стандартах UMTS и CDMA?
2. Какой шифр используется в телекоммуникационной системе стандарта GSM?
3. Какая хеш-функция применяется в стандарте GSM?
4. Какой шифр для защиты информации используется в телекоммуникационной системе стандарта UMTS?
5. Как обеспечивается целостность передаваемых данных в стандарте UMTS?
6. Как обеспечивается целостность передаваемых данных в стандарте CDMA2000?
7. Какова длина блока шифруемых данных в телекоммуникационной системы передачи информации стандарта UMTS?
8. Какова длина блока шифруемых данных в стандарте CDMA2000?
9. Напишите программу вычислений в поле $GF(2^8)$.
10. Напишите программу деления на образующий многочлен $x^8 + x^4 + x^3 + x + 1$.
11. Напишите программу преобразования замена байтов (ByteSub).
12. Напишите программу шифра AES.
13. Разработайте программу шифра KASUMI.

6. Лабораторная работа «Криптографические методы защиты информации»

Лабораторная работа «Криптографические методы защиты информации» предназначена для практического закрепления знаний по основным разделам данного учебного пособия.

При выполнении лабораторной работы студент должен:
уметь проектировать устройства криптографической защиты информации в телекоммуникационных системах; моделировать с помощью современных программных продуктов функциональные узлы генераторов псевдослучайных последовательностей, генераторов поточных шифров на базе регистров с обратными связями, блоков формирования CRC кодов; осуществлять экспериментальные исследования криптографических методов защиты информации;

представлять пути обеспечения заданных характеристик защиты информации в телекоммуникационных системах; выбирать требуемые характеристики многочленов для обеспечения требуемой степени защиты информации в генераторах поточных шифров; а также тенденции, перспективы и проблемы развития техники защиты информации.

Выполнение лабораторной работы должно дать студентам практические навыки исследования методов защиты информации в телекоммуникационных системах; навыки статистической обработки результатов измерений и оценки точности проводимых экспериментальных исследований.

Основные экспериментальные исследования, проводимые в лабораторной работе, подкреплены методическим и теоретически материалом раздела 1 настоящего учебного пособия.

Перед выполнением лабораторной работы требуется знать материал раздела 1, уметь производить умножение и деление многочленов, иметь представление о работе регистров с линейными обратными связями, уметь проектировать генераторы псевдослучайных последовательностей с заданными криптографическими свойствами.

Цель лабораторной работы: Исследование свойств алгоритмов защиты информации на базе CRC кода и поточного шифра.

Содержание работы:

Исследование методов умножения и деления многочленов, описание которых приведено в настоящем учебном пособии (см. рис.1.20 и 1.21).

Исследование принципов работы регистров с линейными обратными связями (раздел 1.4.2).

Исследование свойств алгоритма защиты информации на базе CRC-кода, описанного в разделе 1.4.5.

Исследование статистических свойств генераторов псевдослучайных последовательностей (разделы 1.4.3 и 1.4.4).

Исследование свойств поточного шифра на базе регистра с обратными связями (раздел 1.4.6).

Лабораторное оборудование состоит из лабораторного макета, имеющего встроенный микроконтроллем и панель визуализации результатов эксперимента; персональный компьютер, необходимый для выполнения теоретических расчетов умножения и деления многочленов, а также для выбора образующего полинома при построении генераторов псевдослучайной последовательности чисел.

Описание лабораторной установки

Лабораторная установка предназначена для изучения операций умножения и деления в полях многочленов, лежащих в основе криптографической защиты информации с помощью CRC кода и исследования статистических свойств генераторов псевдослучайных чисел. На установке выполняются следующие разделы лабораторной работы:

- Умножение и деление многочленов.
- Регистры с линейными обратными связями.

- Свойства алгоритма защиты информации на базе CRC-кода
- Статистические свойства генераторов псевдослучайной последовательности.
- Свойства поточного шифра на базе регистра с обратными связями.

Вид лицевой панели представлен на рис. 6.1. В правом верхнем углу лицевой панели лабораторной установки расположен цифробуквенный дисплей, на котором отображаются пункты выполнения лабораторной работы и результаты экспериментальных исследований. В левом нижнем углу лицевой панели, в поле «Многочлен», расположены светодиоды, состояние которых отражает в режиме программирования значение двоичных коэффициентов многочлена. Нумерация двоичных коэффициентов многочлена осуществляется, начиная с младшего разряда. Например, многочлену $x^{16} + x^{12} + x^3 + x + 1$ соответствует запись в двоичном коде 10001000000001011, которая соответствует включенным светодиодам с номерами 17, 13, 4, 2, 1. Дополнительно коэффициенты многочлена выводятся в шестнадцатеричной системе на дисплей, дублируя двоичное представление. Таким образом, на дисплее будет высвечено значение 1100B. В правом нижнем углу лицевой панели расположено поле «Управление меню». Перемещение по меню осуществляется с помощью кнопок управления в этом поле. В табл. 6.1 приведено описание кнопок управления.

При включении лабораторной установки на цифробуквенном дисплее высвечивается слово «Меню». Схема основного меню «Меню» представлена на рис. 6.2, где запись x16 + x12 + x3 + x + 1 соответствует многочлену $x^{16} + x^{12} + x^3 + x + 1$.

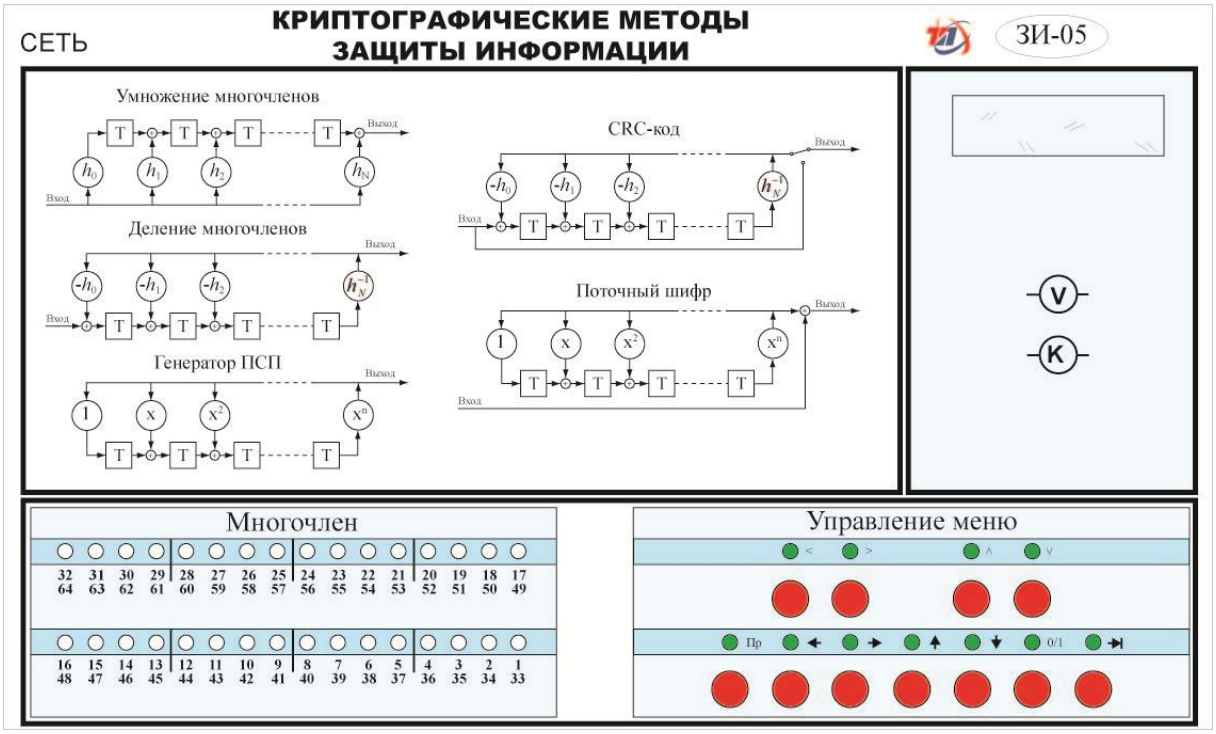

Рис. 6.1. Вид лицевой панели

Выполнение работы начинается с пункта «Поля многочленов», переход к которому осуществляется кнопкой «\/». На этом первом уровне доступны для выбора пункты меню: «Поля многочленов», «Регистр с линейными обратными связями «РСЛОС», «CRC-код», «Генератор псевдослучайной последовательности «ГПСП», «Поточный шифр». С первого уровня каждого пункта меню имеется переход на второй и последующие уровни.

Выбор выполняемого пункта осуществляется путем нажатия управляющих кнопок «>» и «<». Запись многочленов осуществляется в режиме программирования. Переход в режим программирования многочлена осуществляется с помощью кнопки «Пр», находящейся в правом нижнем поле лицевой панели. При этом над ней загорается светодиод, подтверждающий переход в режим программирования.

Таблица 6.1

Назначение управляющих кнопок

<	Перемещение по меню влево
>	Перемещение по меню вправо
/\	Перемещение по меню вверх
\/	Перемещение по меню вниз
Пр	Включение/выключение режима программирования
◄	Перемещение на позицию влево
►	Перемещение на позицию вправо
▲	Перемещение на позицию вверх (влево на 16)
▼	Перемещение на позицию вниз (вправо на 16)
0/1	Изменение значения многочлена на текущей позиции в режиме программирования
▶|	Установка нулей на всех позициях, правее текущей позиции

Программирование значений двоичных коэффициентов многочлена осуществляется с помощью кнопки «0/1». Переход к набору следующего коэффициента выполняется кнопкой «◄», причем, после набора очередного двоичного символа, результат отображается на экране дисплея.

Заметим, что на экране жидкокристаллического дисплея вместо надписи «Ваш выбор: $x^{16} + x^{12} + x^3 + x + 1$» будет выведено «Ваш выбор: x16 + x12 + x3 + x + 1».

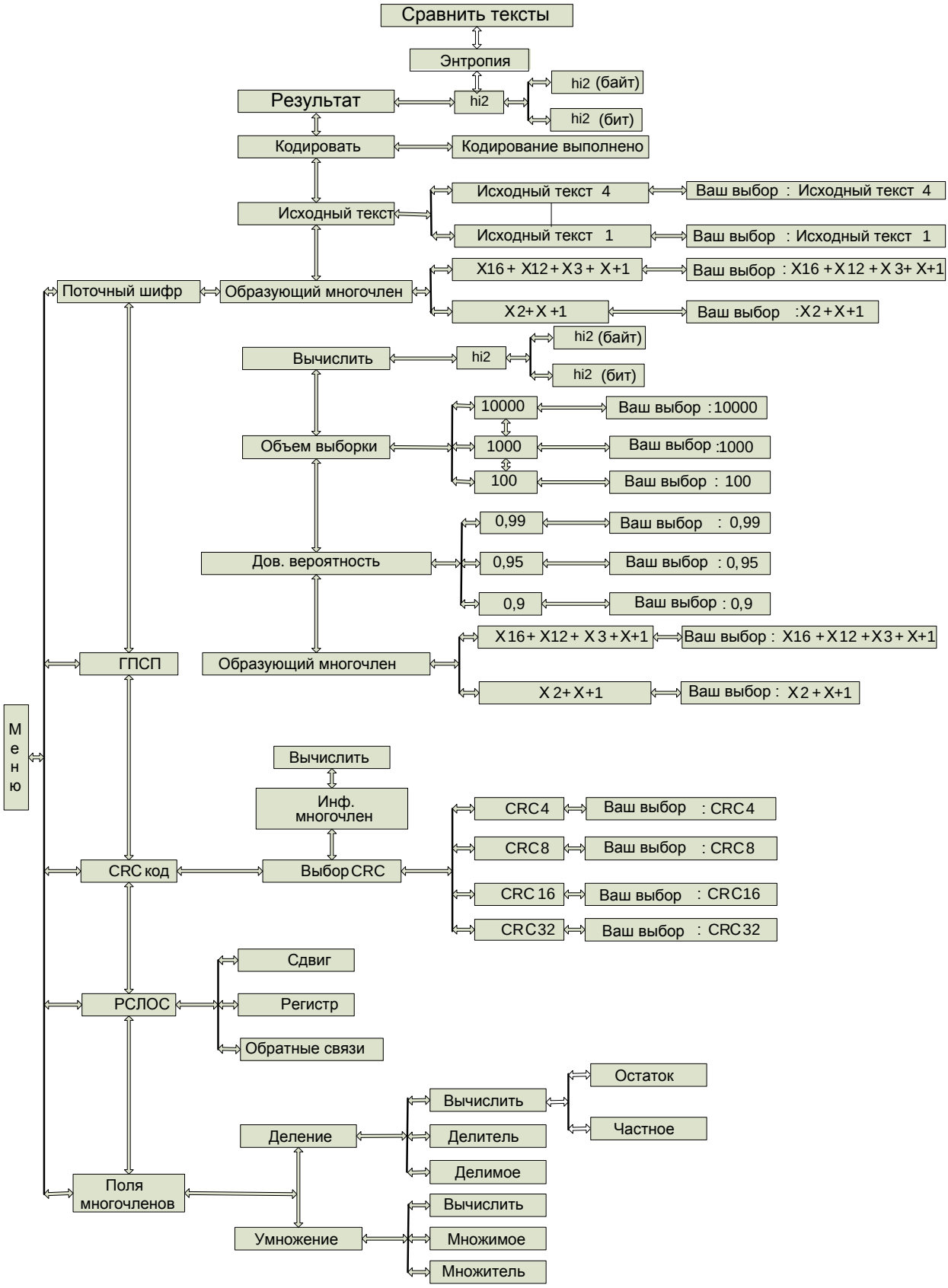

Рис. 6.2. Схема основного меню

Умножение и деление многочленов

На рис. 6.3 приведена схема передвижения по меню с помощью кнопок управления.

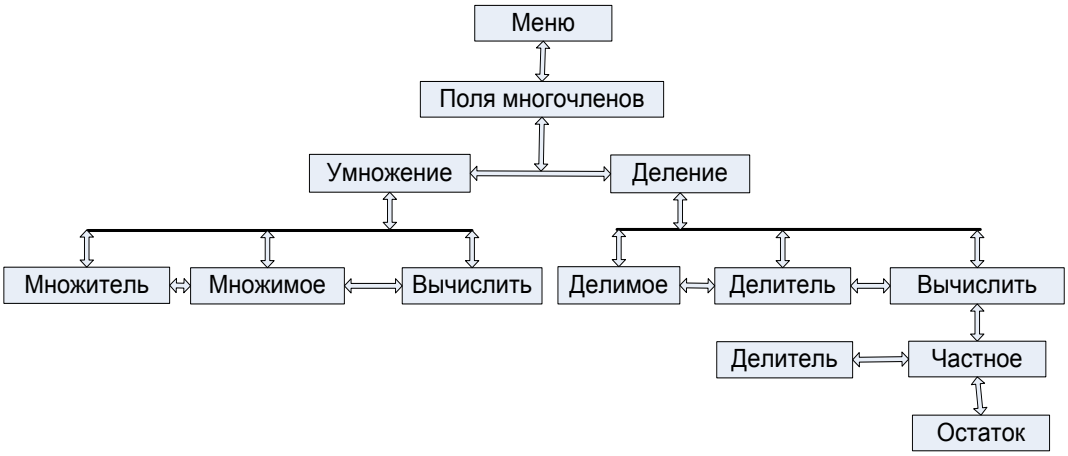

Рис. 6.3. Схема пункта меню «Умножение и деление многочленов»

Переход на следующий уровень происходит с помощью управляющей кнопки «\/». На этом уровне доступны для выбора пункты меню: «Умножение» и «Деление». Выбор осуществляется кнопками «>» и «<». Переход к выполнению подпункта «Множитель», соответствующего режиму программирования множителя, выполняется с помощью управляющей кнопки «\/». Режим программирования множителя задается кнопкой «Пр», находящейся в правом нижнем поле лицевой панели. При этом над кнопкой загорается светодиод, подтверждающий переход в режим программирования. Программирование значений двоичных коэффициентов многочлена осуществляется с помощью кнопки «0/1». Для перехода к набору следующего коэффициента необходимо использовать кнопку «◄». После набора очередного двоичного символа результат отображается на экране дисплея. При этом в левом нижнем поле лицевой панели «Многочлен» с помощью светодиодов будет представлена запись в двоичном коде, а его шестнадцатеричный эквивалент — на экране дисплея в нижней

строчке. Шестнадцатеричный эквивалент на экране дисплея появляется после набора каждого двоичного символа. При неправильном наборе одного двоичного символа можно с помощью кнопок «◄» и «►» вернуться к соответствующему разряду и повторным использованием кнопки «0/1» исправить значение набранного разряда. Кнопкой «►|» происходит установка нулей на всех позициях, правее текущей позиции. Выход из режима программирования осуществляется путем повторного нажатия на кнопку «Пр». При этом гаснет светодиод, расположенный над данной кнопкой.

Переход к подпункту «множимое» происходит с помощью кнопки «>» Для перехода в режим программирования множителя необходимо использовать кнопку «Пр», находящуюся в правом нижнем поле лицевой панели. Программирование множимого осуществляется аналогично программированию множителя. Для перехода к выполнению подпункта меню «вычислить» необходимо выйти из режима программирования, повторно нажав кнопку «Пр».

С помощью кнопки «>» необходимо добиться появления надписи «Вычислить». При этом в поле «Многочлен» на светодиодах и экране дисплея появляется результат операции.

Аналогичным образом осуществляется выполнение операции деление многочленов.

Выбрав в качестве делителя примитивный многочлен из таблицы примитивных многочленов, и несколько раз изменив значение делимого можно убедиться, что остаток от деления каждый раз соответствует одному из элементов конечной мультипликативной группе в поле $GF(2^m)$, где m — степень делителя.

Переход к выполнению следующего пункта осуществляется с помощью кнопки «/\», что соответствует перемещению по меню на один уровень вверх. Повторное нажатие кнопки «/\», соответствует перемещению по меню еще на один уровень вверх, при этом появление надписи «поля многочленов» подтверждает, что мы находимся на верхнем уровне меню.

Регистры с линейными обратными связями (РСЛОС)

Кнопкой «>» выбирается пункт меню: «Исследование регистров с линейными обратными связями», что соответствует появлению на дисплее надписи «РСЛОС». Кнопкой «\/» осуществляется переход к выполнению этого пункта. На рис. 6.4 приведена схема передвижения по меню с помощью кнопок управления.

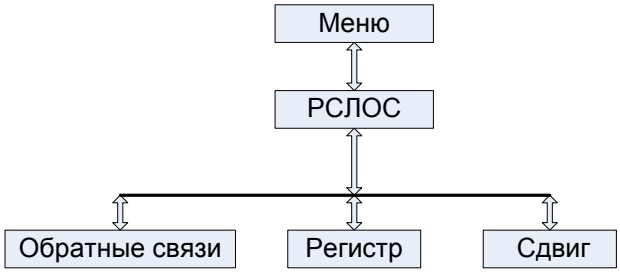

Рис. 6.4. Схема пункта меню «РСЛОС»

Из таблицы примитивных многочленов (табл. 6.2) выбирается примитивный многочлен небольшой степени (например, 4), который будет задавать вид обратной связи.

Переход в режим программирования обратных связей осуществляется кнопкой «Пр», находящейся в правом нижнем поле лицевой панели, при этом над ней загорается светодиод, подтверждающий переход в режим программирования двоичных коэффициентов многочлена, соответствующего регистру обратных связей. Программирование осуществляется аналогично пункту «Поля многочленов». Выход из режима программирования осуществляется повторным нажатием кнопки «Пр». При этом светодиод, расположенный над кнопкой, гаснет.

Переход к выполнению подпункта меню «регистр» осуществляется кнопкой «>». Для перехода в режим программирования состояния регистра необходимо использовать кнопку «Пр», находящуюся в правом нижнем поле лицевой панели. Программирование регистра

осуществляется аналогично пункту умножение и деление многочленов.

Таблица 6.2

Примитивные многочлены

Степень 2	Степень 3	Степень 4
$x^2 + x + 1$	$x^3 + x + 1$ $x^3 + x^2 + 1$	$x^4 + x + 1$ $x^4 + x^3 + 1$
Степень 5	**Степень 6**	**Степень 7**
$x^5 + x^2 + 1$ $x^5 + x^3 + 1$	$x^6 + x + 1$ $x^6 + x^4 + x^3 + x + 1$ $x^6 + x^5 + 1$	$x^7 + x + 1$ $x^7 + x^3 + 1$ $x^7 + x^3 + x^2 + x + 1$
Степень 8	**Степень 9**	**Степень 10**
$x^8 + x^4 + x^3 + x^2 + 1$ $x^8 + x^5 + x^3 + x + 1$ $x^8 + x^5 + x^3 + x^2 + 1$	$x^9 + x^4 + 1$ $x^9 + x^4 + x^3 + x + 1$ $x^9 + x^5 + 1$	$x^{10} + x^3 + 1$ $x^{10} + x^4 + x^3 + x + 1$ $x^{10} + x^5 + x^2 + x + 1$
Степень 11	**Степень 12**	**Степень 13**
$x^{11} + x^2 + 1$ $x^{11} + x^4 + x^2 + x + 1$ $x^{11} + x^5 + x^3 + x + 1$	$x^{12} + x^6 + x^4 + x + 1$ $x^{12} + x^6 + x^5 + x^3 + 1$ $x^{12} + x^6 + x^5 +$ $+x^4 + x^3 + x + 1$	$x^{13} + x^4 + x^3 + x + 1$ $x^{13} + x^5 + x^2 + x + 1$ $x^{13} + x^5 + x^4 + x^2 + 1$

Переход к выполнению подпункта меню «сдвиг» происходит с помощью кнопок «>» и «\/».

При выключенном режиме программирования, последовательным нажатием на кнопки «◄» и «►» можно осуществить пошаговый сдвиг содержимого регистра влево и вправо. При этом как на светодиодах, так и на дисплее отображается значение содержимого регистра.

Переход к выполнению следующего пункта осуществляется повторным нажатием кнопки «/\». При этом появление надписи «РСЛОС» подтвердит нахождение в верхнем уровне меню.

Свойства алгоритма защиты информации на базе CRC-кода

С помощью управляющей кнопки «>» выбирается пункт меню «CRC-код». На рис. 6.5 приведена схема передвижения по меню с помощью кнопок управления.

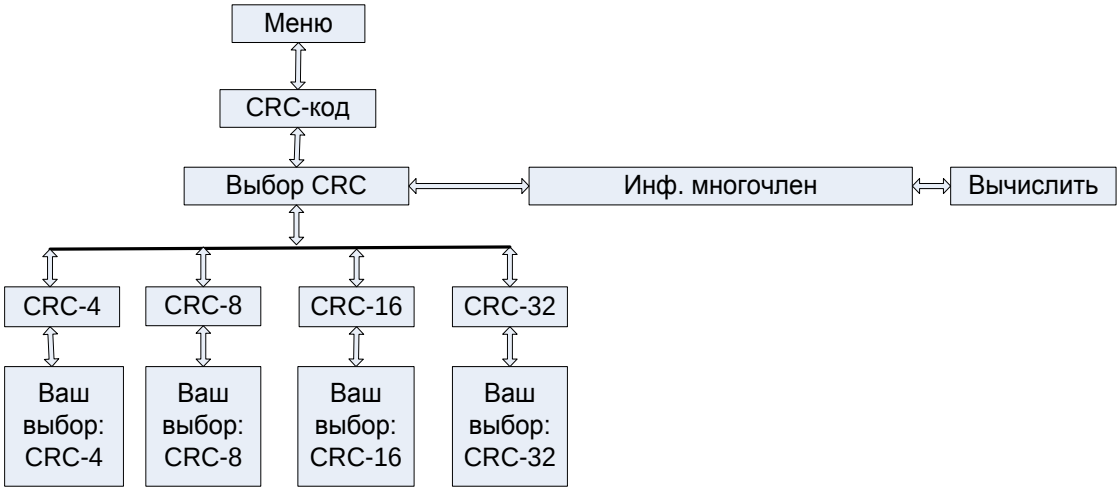

Рис. 6.5. Схема меню «CRC-код»

Кнопкой «\/» необходимо добиться появления надписи «Выбор CRC». Повторным нажатием кнопки «\/» и с помощью управляющих кнопок «>» и «<» выбирается соответствующий CRC-код (CRC-4, CRC-8, CRC-16, CRC-32). Надписи будут последовательно появляться на экране дисплея. Повторным нажатием кнопки «\/» подтверждается выбор. При этом на дисплее появляется надпись «Ваш выбор: CRC-N». Двукратное нажатие кнопки «/\» возвращает на два уровня вверх, что соответствует надписи «Выбор CRC».

Переход к пункту меню «Информационный многочлен» осуществляется кнопкой «>». Для перехода в режим программирования информационного многочлена необходимо использовать кнопку «Пр», находящуюся в правом нижнем поле лицевой панели. При этом над ней загорается светодиод, подтверждающий переход в режим про-

граммирования. Набор информационного многочлена осуществляется в соответствии с рассмотренной ранее методикой.

Переход к выполнению подпункта меню «Вычислить» осуществляется кнопкой «>». При этом на светодиодах и на дисплее высвечивается значение CRC-кода. Изменяя информационную последовательность необходимо убедиться, что при этом изменяется и значение CRC-кода.

Выполнение следующего пункта осуществляется повторным нажатием кнопки «/\». При этом появление надписи «CRC-кода» подтверждает нахождение на верхнем уровне меню.

Статистические свойства генераторов псевдослучайной последовательности

Кнопкой «>» выбирается режим «Генератор псевдослучайной последовательности», которому соответствует надпись на экране дисплея «ГПСП». На рис. 6.6 приведена схема передвижения по меню с помощью кнопок управления.

Кнопкой «\/» необходимо добиться появления надписи «Образующий многочлен». Переход в режим выбора образующего многочлена осуществляется кнопкой «\/». Выбор образующего многочлена происходит с помощью управляющих кнопок «<» и «>». Нажатие кнопки «\/» подтверждает выбор многочлена. При этом на дисплее появляется надпись, например, «Ваш выбор: x16 + x12 + x3 + x + 1», что соответствует многочлену $x^{16} + x^{12} + x^3 + x + 1$.

Переход к выполнению подпункта меню «Доверительная вероятность» осуществляется двукратным нажатием кнопки «/\», что соответствует возврату по меню на два уровня вверх. Появление надписи «Образующий многочлен» свидетельствует о возврате к предыдущему пункту меню. Кнопкой «>» необходимо добиться появления надписи «Доверительная вероятность» и с помощью кнопки «\/» перейти в режим выбора численного значения доверительной вероятности.

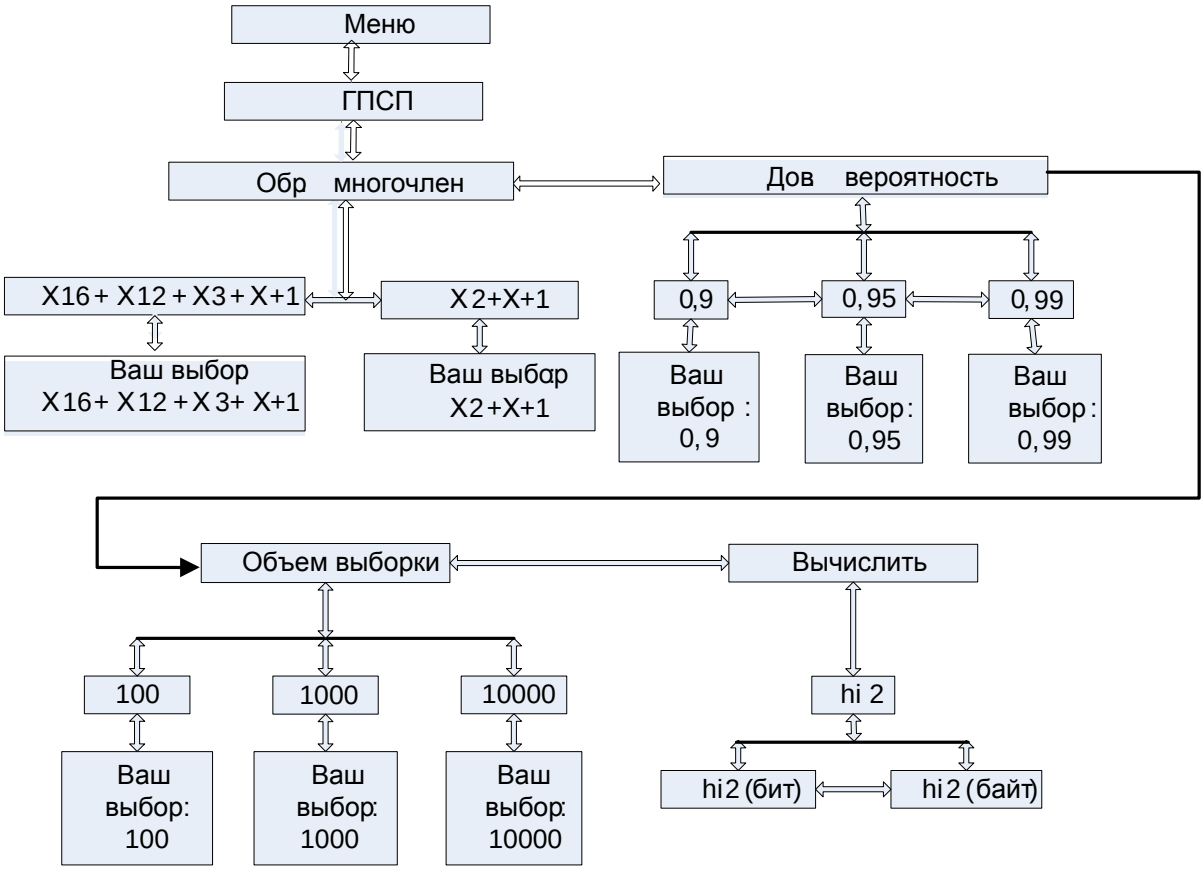

Рис. 6.6 Схема пункта меню «ГПСП»

Кнопкой «>» необходимо добиться появления требуемого значения доверительной вероятности {0,9, 0,95, 0,99} и кнопкой «\/» подтвердить выбор. При этом появляется надпись «Ваш выбор: 0.9x».

Переход к выполнению следующего пункта меню осуществляется двукратным нажатием кнопки «/\», что соответствует возврату по меню на два уровня вверх. При этом появляется надпись «Доверительная вероятность», нажатием кнопки «>» добиться появления надписи «Объем выборки» и кнопкой «\/» перейти в режим выбора значения.

С помощью кнопки «>» необходимо добиться появления требуемого значения объема выборки {100, 1000, 10000} и кнопкой «\/» подтвердить выбор. При этом появляется надпись «Ваш выбор: 1xxxx».

Переход к выполнению следующего пункта меню «Вычислить» осуществляется двукратным нажатием кнопки «/\». При этом появляется надпись «Объем выборки». Кнопкой «>» необходимо добиться появления надписи «Вычислить». После этого кнопкой «\/» необходимо добиться появления надписи «hi2» и кнопкой «\/» подтвердить переход в режим расчета по битам или байтам. Переключение между этими режимами осуществляется кнопками «>» и «<». При этом на экране дисплея появляется в верхней строке граничное значение критерия χ^2 соответствующего выбранной доверительной вероятности, а в нижней строке вычисленное значение, соответствующее выбранной доверительной вероятности.

Переход к выполнению следующего пункта осуществляется двукратным нажатием кнопки «/\», при этом появление надписи «ГПСП» подтверждает нахождение в верхнем уровне меню.

Свойства поточного шифра на базе регистра с обратными связями

Кнопкой «>» выбирается пункт меню «Поточный шифр» и кнопкой «\/» подтверждается этот выбор. На экране дисплея появляется надпись «Образующий многочлен». Кнопкой «\/» подтверждается этот выбор. На рис. 6.7 приведена схема передвижения по меню.

Кнопками «>» и «<» выбирается образующий многочлен из тех, которые последовательно появляются на дисплее. Повторное нажатие кнопки «\/» подтверждает выбор. При этом на дисплее появляется надпись «Ваш выбор: x16 + x2 + 1». Переход к выполнению подпункта меню «Исходный текст» осуществляется двукратным нажатием кнопки «/\». При этом появляется надпись «Образующий многочлен». Кнопка «>» обеспечивает появления на экране дисплея надписи «Исходный текст».

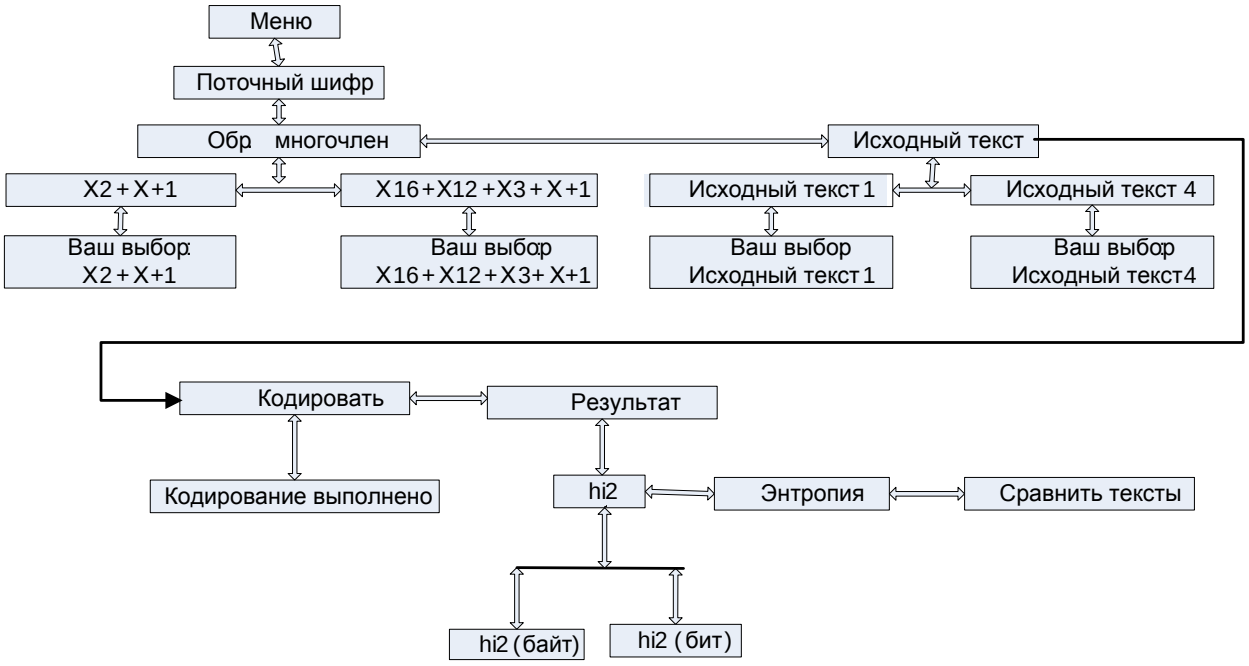

Рис. 6.7. Схема меню «Поточный шифр»

С помощью кнопки «\/» необходимо установить режим выбора конкретного текста. Выбор номера текста осуществляется кнопками «>» и «<» и кнопкой «\/» подтверждается выбор. При этом на дисплее появляется надпись «Ваш выбор: Текст № x».

Для перехода к выполнению подпункта меню «Кодировать» необходимо два раза нажать кнопку «/\» и кнопкой «>» добиться появления надписи «Кодировать». Подтверждение выбора осуществляется кнопкой «\/». При этом на дисплее высвечивается надпись «Кодирование выполнено».

После выполнения предыдущего подпункта, с помощью кнопки «/\», необходимо вернуться на один уровень вверх. При этом появляется надпись «Кодировать». Кнопкой «>» добиться появления надписи «Результат» и кнопкой «\/» подтвердить выбор.

Повторным нажатием кнопки «\/» осуществляется выбор режима отображения вычисленного значения χ^2. Кнопками «>» и «<» осуществляется выбор между расчётом по байтам или по битам.

С помощью кнопки «/\» осуществляется возврат по меню на один уровень вверх. Кнопкой «>» необходимо добиться появления

надписи «Энтропия». На верхней строке дисплея отображается в скобках значение энтропии открытого текста, а на нижней строке — значение энтропии кодированного текста.

Кнопкой «>» необходимо добиться появления надписи «Сравнить тексты». После нажатия кнопки «\/» осуществляется просмотр исходного и кодированного текста кнопками «◄», «►», «▲» и «▼». Исходный текст отображается в верхней строке, а кодированный текст — в нижней. Кнопками «◄» и «►» осуществляется сдвиг на один символ. Кнопками «▲» и «▼» осуществляется сдвиг на 16 символов.

Подготовка к выполнению лабораторной работы

Перед выполнением лабораторной работы необходимо иметь представление о правилах умножения и деления многочленов. Уметь проводить все вычисления в двоичной и шестнадцатеричной системах счисления.

Программа и порядок выполнения работы.

Ознакомившись с описанием лабораторной установки, включить установку нажатием кнопки «Сеть». После этого провести следующие измерения.

Умножение и деление многочленов.

Провести три процедуры умножения многочленов степени не ниже пятой. Множимое, множитель и результат, считываются с экрана дисплея и заносятся в табл 6.3. Запись многочленов в таблицу производится в шестнадцатеричной системе счисления. Так, например, представленная на языке СИ запись 0x51, означает, что число 51 записано в шестнадцатеричной системе счисления и соответствует двоичной записи многочлена 1010001 или $x^5 + x^3 + 1$.

Таблица 6.3

Тестовый пример умножения

Умножение многочленов		
Множимое	Множитель	Результат
0x5	0x3	0xf
0x51	0x39	0xDE9
0x5207	0x3243B	0xFD0CFAA1

Провести три процедуры деления многочленов степени не ниже пятой. Множимое, множитель и результат, считываются с экрана дисплея и заносятся в табл. 6.4.

Таблица 6.4

Тестовый пример деления

Деление многочленов			
Делимое	Делитель	Результат	
		Частное	Остаток
0x654	0x9A	0xD	0x76
0x952497	0x3289A	0x4F	0x9D01
0x32000860000	0x16161BE	0x3BACA	0x58D20C

Регистры с линейными обратными связями

Из таблицы примитивных многочленов (табл. 6.2) выбрать три примитивных многочлена, которые будут задавать вид обратной связи. Начальное состояние регистра выбирается произвольно в интервале от 1 до $2n - 1$, где n — длина регистра, равная порядку многочлена в цепи обратной связи. Результаты выполнения приводятся в отчете в виде таблицы (пример — табл. 6.5 для $n = 2$), состоящей из $2^n - 1$ строк и четырех столбцов. В первом столбце записывается количество сдвигов влево, во втором — содержимое регистра в шестнадцатерич-

ной системе. Аналогично оформляются третий и четвертый столбцы таблицы 5 для сдвига содержимого регистра вправо.

Таблица 6.5

Тестовый пример исследования состояния регистра сдвига

Обратные связи — 0x43			
Номер сдвига влево	Регистр	Номер сдвига вправо	Регистр
Начальное состояние сдвигового регистра – 0x1		Начальное состояние сдвигового регистра – 0x10	
1	0x2	1	0x8
2	0x4	2	0x4
3	0x8	3	0x2

Количество сдвигов и вправо и влево должно быть равно $2^n - 1$. При этом необходимо убедиться, что в таблице приведены все возможные состояния регистра соответствующие данному примитивному многочлену.

Свойства алгоритма защиты информации на базе CRC-кода

Выбрать два из возможных CRC-кодов (CRC-4, CRC-8, CRC-16, CRC-32). Выбираются три произвольных информационных многочлена. Для многочлена определить соответствующий ему CRC код. Результаты исследований занести в табл. 6.6.

Таблица 6.6

Вычисление CRC-кода

CRC-код	Информационный многочлен	CRC-сумма
CRC-4	0xDF1	0x1
	0xCF1	0x5
	0x3124	0xF
	0x74C00A4	0x7
CRC-8	0xDF1	0x2E
	0x3124	0x54
	0x74C00A4	0x4A

Статистические свойства генераторов псевдослучайной последовательности.

Таблица 6.7

Исследование генератора ПСП

Объем выборки = 100		
Степень образующего многочлена	hi2 (байт)	hi2 (бит)
12	257,99	0,22
16	234,96	0,16

Объем выборки = 10000		
Степень образующего многочлена	hi2 (байт)	hi2 (бит)
12	193,99	0,02
16	232,96	0,77

Выбрать два образующих многочлена разной степени. Для двух разных объемов выборки рассчитать значения χ^2, которые сравнить с пороговым значением. Определить соответствие экспериментальных данных гипотезе о равномерном распределении. Результаты вычислений занести в табл. 6.7.

Вычисленные значения χ^2, сравнить с пороговыми значениями (табл. 6.8). Если в результате измерения вычисленное значение превышает пороговое, то гипотеза о равномерном значении отклоняется.

Таблица 6.8

Пороговые значения для принятия решения о равномерности распределения

Доверительная вероятность	Пороговое значение hi2 (байт)	Пороговое значение hi2 (бит)
0.9	285	4
0.95	294	6
0.99	311	9

Свойства поточного шифра на базе регистра с обратными связями

Таблица 6.9

Результаты исследования текстовых примеров

Исходный текст №1				
Степень образующего многочлена	hi2 (байт)	hi2 (бит)	Энтропия	
8	237,97	0,49	4,39	7,10
10	257,99	0,22	4,39	7,14
12	269,96	0,77	4,39	7,20
Исходный текст №2				
Степень образующего многочлена	hi2 (байт)	hi2 (бит)	Энтропия	
8	215,99	1,52	4,47	7,21
10	233,99	0,69	4,47	7,25
12	229,99	0,02	4,47	7,24

Определить значение энтропии открытого текста и значение энтропии после выполнения процедуры шифрования поточным шиф-

ром. Для этого выбрать три образующих многочлена. Выбрать два из четырех возможных открытых текстов. Результаты анализа свести в табл. 6.9.

Содержание отчета

Наименование и цель работы.

- Таблица выполнения пункта «Умножение и деление многочленов», сопровождаемая проверочным вычислением «вручную».
- Таблица выполнения пункта «Регистры с линейными обратными связями», сопровождаемая выводами о подтверждении мультипликативной структуре поля многочленов.
- Таблица выполнения пункта «Свойства алгоритма защиты информации на базе CRC кода», сопровождаемая проверочным вычислением «вручную» и выводами о том, какие изменения в информационном многочлене обнаруживает CRC код.
- Таблица выполнения пункта «Статистические свойства генератора псевдослучайной последовательности» с выводами о результатах тестирования.
- Таблица выполнения пункта «Свойства поточного шифра на базе регистра с обратными связями» и результаты тестирования открытого и шифрованного текстов, с выводами. Кроме того, должны быть приведены начальные последовательности символов открытого и шифрованного текстов, сопровождаемые проверочным вычислением «вручную».
- Выводы по работе.

Задания к лабораторной работе

Задание 1. Выбрать два многочлена степени не ниже пятой с коэффициентами из поля GF(2), перемножить их, привести подобные и сравнить полученный результат с результатом вычислений на лабораторной установке.

Задание 2. Выбрать первый многочлен степени не ниже восьмой с коэффициентами из поля GF(2), выбрать второй многочлен степени не ниже пятой из таблицы примитивных многочленов, разделить первый многочлен на второй и сравнить полученный результат с результатом вычислений на лабораторной установке.

Задание 3. Сформировать случайный информационный многочлен с коэффициентами из поля GF(2) степени не ниже десятой и вычислить для него контрольный код CRC-16 путем деления информационного многочлена на порождающий. Сравнить полученный результат с результатом экспериментальных исследований, полученных на лабораторной установке.

Задание 4. Взять в качестве образующего многочлен степени не ниже пятой из таблицы примитивных мночленов и заполнив регистр случайным набором нулей и единиц, выписать все возможные последовательные состояния при последовательном сдвиге регистра. Сравнить полученные состояния с результатами, полученными на лабораторной установке.

Задание 5. Последовательно сосчитать с индикатора лабораторной установки один из текстов, записанных в памяти и рассчитать для него энропию. Сравнить полученный результат с результатом, полученным на лабораторной установке.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Саймон С.* Книга кодов. — М.: АСТ, 2007. — 448 с.
2. *Смарт Н.* Криптография М.: Техносфера 2005. — 528 с.
3. *Асосков А. В.* Поточные шифры. / А. В. Асосков, М. А. Иванов, А. А. Мирский, А. В. Рузин. — М.: Кудиц – Образ, 2003. — 336 с.
4. *Иванов М. А.* Теория, применение и оценка качества генераторов псевдослучайных последовательностей. / И. В. Чугунков. — М.: Кудиц – Образ, 2003. — 240 с.
5. *Шнайер Б.* Прикладная криптография. М.: Триумф, 2002. — 815 с.
6. *Макаров С. Б.* Телекоммуникационные технологии: введение в технологии GSM: учебное пособие для высш. учеб. заведений. / С. Б. Макаров, Н. В. Певцов, Е. А. Попов, М. А. Сиверс. — М.: Издательский центр «Академия», 2006. — 256 с.
7. *Рябко Б. Я.* Криптографические методы защиты информации: учебное пособие для вузов. / Б. Я Рябко, А. Н. Фионов. — М.: Горячая Линия — Телеком, 2005. — 229 с.
8. *Нечаев В. И.* Элементы криптографии (Основы теории защиты информации): Учебное пособие для университетов и педагогических вузов. — М.: Высшая школа 1999. — 109 с.
9. *Баричев С. Г.* Основы современной криптографии. Учебное пособие для высших учебных заведений. / С. Г. Баричев, В. В. Гончаров, Р. Е. Серов. — М.: Горячая линия — Телеком, 2001. — 175 с.
10. *Волков А. Н.* UMTS. Стандарт сотовой связи третьего поколения. / А. Н. Волков, А. Е. Рыжков, М. А. Сиверс. — СПб.: Изд-во «Линк», 2008. — 224 с.

Ветров Юрий Викторович
Макаров Сергей Борисович

Криптографические методы защиты информации в телекоммуникационных системах

Учебное пособие

Лицензия ЛР № 020593 от 07 09 97
Налоговая льгота – Общероссийский классификатор продукции
ОК 005-93, т 2, 953005 – учебная литература

Подготовлено к печати Формат 80x64/16 Печать цифровая
Усл. печ. л. 10,94 Уч.–изд. л. Тираж экз. Заказ

Отпечатано с готового оригинал-макета, предоставленного авторами
в Цифровом типографическом центре
Издательства Политехнического университета
195251, Санкт-Петербург, Политехническая ул., 29
Тел. (812) 550-40-14
Тел./факс. (812) 297-547-76