Под общ. ред. В.В.Ященко

ВВЕДЕНИЕ В КРИПТОГРАФИЮ

Авторский коллектив: В. В. Ященко (редактор, глава 1), Н. П. Вар-новский (главы 2, 3), Ю. В. Нестеренко (глава 4), Г. А. Кабатянский (глава 5), П. Н. Девянин, В. Г. Проскурин, А. В. Черемушкин (глава 6), П. А. Гырдымов, А. Ю. Зубов, А. В. Зязин, В. Н. Овчинников (глава 7).

В книге впервые на русском языке дается систематическое изложение научных основ криптографии от простейших примеров и основных понятий до современных криптографических конструкций. Понимание принципов криптографии стало для многих потребностью в связи с широким распространением криптографических средств обеспечения информационной безопасности. Поэтому книга может быть полезна массовому читателю.

Книга рассчитана на студентов-математиков и специалистов по информационной безопасности.

Содержание

Содержание

Предисловие ко второму изданию

В настоящем втором издании исправлены опечатки и неточности, замеченные в первом издании.

сентябрь 1999 г. *В. Ященко*

Предисловие к первому изданию

Криптография — наука о шифрах — долгое время была засекречена, так как применялась, в основном, для защиты государственных и военных секретов. В настоящее время методы и средства криптографии используются для обеспечения информационной безопасности не только государства, но и частных лиц, и организаций. Дело здесь совсем не обязательно в секретах. Слишком много различных сведений «гуляет» по всему свету в цифровом виде. И над этими сведениями «висят» угрозы недружественного ознакомления, накопления, подмены, фальсификации и т. п. Наиболее надежные методы защиты от таких угроз дает именно криптография.

Пока криптографические алгоритмы для рядового потребителя — тайна за семью печатями, хотя многим уже приходилось пользоваться некоторыми криптографическими средствами: шифрование электронной почты, интеллектуальные банковские карточки и др. Естественно, что при этом основной вопрос для пользователя — обеспечивает ли данное криптографическое средство надежную защиту. Но даже правильно сформулировать этот элементарный вопрос непросто. От какого противника защищаемся? Какие возможности у этого противника? Какие цели он может преследовать? Как измерять надежность защиты? Список таких вопросов можно продолжить. Для ответа на них пользователю необходимы знания основных понятий криптографии.

Популярное изложение научных основ криптографии (речь идет только о «негосударственной» криптографии; разделы криптографии, связанные с государственной безопасностью, должны оставаться секретными) — цель настоящей книги. Ее можно использовать и в качестве учебного пособия. На русском языке аналогичных книг пока нет. Материалы ряда глав публиковались авторами ранее в других изданиях (глава 1 — в книге С. А. Дориченко, В. В. Ященко, «25 этюдов о шифрах», М.: ТЕИС, 1994; главы 1,2,4,5 — в журнале «Математическое просвещение», третья серия, выпуск 2, М.: МЦНМО, 1998; глава 7 — в газете «Информатика» (еженедельное приложение к газете «Первое сентября»), № 4, январь 1998). При подготовке настоящего издания эти материалы были переработаны и дополнены.

Изложение материала рассчитано на читателя с математическим складом ума. В основном главы не зависят друг от друга (это достигнуто за счет некоторых повторов) и их можно читать в произвольном порядке. Главу 1 — вводную — рекомендуется прочитать всем, поскольку в ней на

популярном уровне разъясняются все основные понятия современной криптографии: шифр, ключ, стойкость, электронная цифровая подпись, криптографический протокол и др. В других главах часть материала повторяется, но уже более углубленно. В главах 2, 3, 4, 5 используются некоторые сведения из высшей математики, известные ученикам математических классов и студентам. Глава 6 ориентирована на знатоков компьютерных технологий. Глава 7 содержит материалы олимпиад по криптографии для школьников и поэтому для ее чтения никаких знаний, выходящих за пределы школьной программы, не требуется.

Предупреждение: криптографические средства и программные продукты, упоминаемые в книге, используются только для иллюстрации общих криптографических идей; авторы не ставили своей целью давать оценки или сравнивать имеющиеся на рынке криптографические средства.

Криптография была поставлена на научную основу во многом благодаря работам выдающегося американского ученого Клода Шеннона. Его доклад «Математическая теория криптографии» был подготовлен в секретном варианте в 1945 г., рассекречен и опубликован в 1948 г., переведен на русский язык в 1963 г. Поскольку «Работы по теории информации и кибернетике» (1963 г.) К. Шеннона стали библиографической редкостью, мы включили в приложение основную часть статьи К. Шеннона «Теория связи в секретных системах». Эту основополагающую работу рекомендуется прочитать всем интересующимся криптографией.

Для профессионального понимания криптографических алгоритмов и умения оценивать их сильные и слабые стороны необходима уже серьезная математическая подготовка (на уровне математических факультетов университетов). Это объясняется тем, что современная криптография основана на глубоких результатах таких разделов математики, как теория сложности вычислений, теория чисел, алгебра, теория информации и др. Желающим серьезно изучать криптографию можно порекомендовать обзорную монографию «Криптография в банковском деле» Анохина М. И., Варновского Н. П., Сидельникова В. М., Ященко В. В., М.: МИФИ, 1997.

октябрь 1998 г. *В. Ященко*

Глава 1

Основные понятия криптографии

1. Введение

Как передать нужную информацию нужному адресату в тайне от других? Каждый из читателей в разное время и с разными целями наверняка пытался решить для себя эту практическую задачу (для удобства дальнейших ссылок назовем ее «задача ТП», т. е. задача *Тайной Передачи*). Выбрав подходящее решение, он, скорее всего, повторил изобретение одного из способов скрытой передачи информации, которым уже не одна тысяча лет.

Размышляя над задачей ТП, нетрудно прийти к выводу, что есть три возможности.

1. Создать абсолютно надежный, недоступный для других канал связи между абонентами.

2. Использовать общедоступный канал связи, но скрыть сам факт передачи информации.

3. Использовать общедоступный канал связи, но передавать по нему нужную информацию в так преобразованном виде, чтобы восстановить ее мог только адресат.

Прокомментируем эти три возможности.

1. При современном уровне развития науки и техники сделать такой канал связи между удаленными абонентами для неоднократной передачи больших объемов информации практически нереально.

2. Разработкой средств и методов скрытия факта передачи сообщения занимается *стеганография*.

Первые следы стеганографических методов теряются в глубокой древности. Например, известен такой способ скрытия письменного сообщения: голову раба брили, на коже головы писали сообщение и после отрастания волос раба отправляли к адресату.

Из детективных произведений хорошо известны различные способы тайнописи между строк обычного, незащищаемого текста: от молока до сложных химических реактивов с последующей обработкой.

Также из детективов известен метод «*микроточки*»: сообщение записывается с помощью современной техники на очень маленький носитель

(микроточку), который пересылается с обычным письмом, например, под маркой или где-нибудь в другом, заранее обусловленном месте.

В настоящее время в связи с широким распространением компьютеров известно много тонких методов «запрятывания» защищаемой информации внутри больших объемов информации, хранящейся в компьютере. Наглядный пример запрятывания текстового файла в графический можно найти в Интернете[1]; он же приведен в журнале «Компьютерра», №48 (225) от 1 декабря 1997 г., на стр. 62. (Следует отметить, что авторы статьи в журнале ошибочно относят стеганографию к криптографии. Конечно, с помощью стеганографии можно прятать и предварительно зашифрованные тексты, но, вообще говоря, стеганография и криптография — принципиально различные направления в теории и практике защиты информации.)

3. Разработкой методов преобразования (*шифрования*) информации с целью ее защиты от незаконных пользователей занимается *криптография.* Такие методы и способы преобразования информации называются *шифрами.*

Шифрование (*зашифрование*) — процесс применения шифра к защищаемой информации, т. е. преобразование защищаемой информации (*открытого текста*) в шифрованное сообщение (*шифртекст, криптограмму*) с помощью определенных правил, содержащихся в шифре.

Дешифрование — процесс, обратный шифрованию, т. е. преобразование шифрованного сообщения в защищаемую информацию с помощью определенных правил, содержащихся в шифре.

Криптография — прикладная наука, она использует самые последние достижения фундаментальных наук и, в первую очередь, математики. С другой стороны, все конкретные задачи криптографии существенно зависят от уровня развития техники и технологии, от применяемых средств связи и способов передачи информации.

2. Предмет криптографии

Что же является предметом криптографии? Для ответа на этот вопрос вернемся к задаче ТП, чтобы уточнить ситуацию и используемые понятия.

Прежде всего заметим, что эта задача возникает только для информации, которая нуждается в защите. Обычно в таких случаях говорят, что информация содержит тайну или является *защищаемой, приватной, конфиденциальной, секретной.* Для наиболее типичных, часто встречающихся ситуаций такого типа введены даже специальные понятия:

– государственная тайна;

– военная тайна;

---

[1] `http://www.geocities.com/SiliconValley/Vista/6001/`

– коммерческая тайна;
– юридическая тайна;
– врачебная тайна и т. д.

Далее мы будем говорить о защищаемой информации, имея в виду следующие признаки такой информации:

– имеется какой-то определенный круг *законных пользователей*, которые имеют право владеть этой информацией;

– имеются *незаконные пользователи*, которые стремятся овладеть этой информацией с тем, чтобы обратить ее себе во благо, а законным пользователям во вред.

Для простоты мы вначале ограничимся рассмотрением только одной *угрозы* — угрозы разглашения информации. Существуют и другие угрозы для защищаемой информации со стороны незаконных пользователей: подмена, имитация и др. О них мы поговорим ниже.

Теперь мы можем изобразить ситуацию, в которой возникает задача ТП, следующей схемой (см. рис. 1).

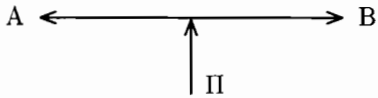


Рис. 1.

Здесь A и B — удаленные законные пользователи защищаемой информации; они хотят обмениваться информацией по общедоступному каналу связи. П — незаконный пользователь (*противник*), который может перехватывать передаваемые по каналу связи сообщения и пытаться извлечь из них интересующую его информацию. Эту формальную схему можно считать моделью типичной ситуации, в которой применяются криптографические методы защиты информации.

Отметим, что исторически в криптографии закрепились некоторые военные слова (противник, атака на шифр и др.) Они наиболее точно отражают смысл соответствующих криптографических понятий. Вместе с тем широко известная военная терминология, основанная на понятии кода (военно-морские коды, коды Генерального штаба, кодовые книги, кодобозначения и т. п.), уже не применяется в теоретической криптографии. Дело в том, что за последние десятилетия сформировалась *теория кодирования* — большое научное направление, которое разрабатывает и изучает методы защиты информации от случайных искажений в каналах связи. И если ранее термины кодирование и шифрование употреблялись как синонимы, то теперь это недопустимо. Так, например, очень распространенное выражение «кодирование — разновидность шифрования» становится просто неправильным.

Криптография занимается методами преобразования информации, которые бы не позволили противнику извлечь ее из перехватываемых

сообщений. При этом по каналу связи передается уже не сама защищаемая информация, а результат ее преобразования с помощью шифра, и для противника возникает сложная задача *вскрытия шифра*.

Вскрытие (*взламывание*) *шифра* — процесс получения защищаемой информации из шифрованного сообщения без знания примененного шифра.

Однако помимо перехвата и вскрытия шифра противник может пытаться получить защищаемую информацию многими другими способами. Наиболее известным из таких способов является агентурный, когда противник каким-либо путем склоняет к сотрудничеству одного из законных пользователей и с помощью этого агента получает доступ к защищаемой информации. В такой ситуации криптография бессильна.

Противник может пытаться не получить, а уничтожить или модифицировать защищаемую информацию в процессе ее передачи. Это — совсем другой тип угроз для информации, отличный от перехвата и вскрытия шифра. Для защиты от таких угроз разрабатываются свои специфические методы.

Следовательно, на пути от одного законного пользователя к другому информация должна защищаться различными способами, противостоящими различным угрозам. Возникает ситуация цепи из разнотипных звеньев, которая защищает информацию. Естественно, противник будет стремиться найти самое слабое звено, чтобы с наименьшими затратами добраться до информации. А значит, и законные пользователи должны учитывать это обстоятельство в своей стратегии защиты: бессмысленно делать какое-то звено очень прочным, если есть заведомо более слабые звенья («принцип равнопрочности защиты»).

Не следует забывать и еще об одной важной проблеме: проблеме соотношения цены информации, затрат на ее защиту и затрат на ее добывание. При современном уровне развития техники сами средства связи, а также разработка средств перехвата информации из них и средств защиты информации требуют очень больших затрат. Прежде чем защищать информацию, задайте себе два вопроса:

1) является ли она для противника более ценной, чем стоимость атаки;

2) является ли она для вас более ценной, чем стоимость защиты.

Именно перечисленные соображения и являются решающими при выборе подходящих средств защиты: физических, стеганографических, криптографических и др.

Некоторые понятия криптографии удобно иллюстрировать историческими примерами, поэтому сделаем небольшое историческое отступление.

Долгое время занятие криптографией было уделом чудаков-одиночек. Среди них были одаренные ученые, дипломаты, священнослужители. Извест-

ны случаи, когда криптография считалась даже черной магией. Этот период развития криптографии как искусства длился с незапамятных времен до начала XX века, когда появились первые шифровальные машины. Понимание математического характера решаемых криптографией задач пришло только в середине XX века — после работ выдающегося американского ученого К. Шеннона.

История криптографии связана с большим количеством дипломатических и военных тайн и поэтому окутана туманом легенд. Наиболее полная книга по истории криптографии содержит более тысячи страниц. Она опубликована в 1967 году и на русский язык не переведена[1]. На русском языке недавно вышел в свет фундаментальный труд по истории криптографии в России[2].

Свой след в истории криптографии оставили многие хорошо известные исторические личности. Приведем несколько наиболее ярких примеров. Первые сведения об использовании шифров в военном деле связаны с именем спартанского полководца Лисандра (шифр «Сцитала»). Цезарь использовал в переписке шифр, который вошел в историю как «шифр Цезаря». В древней Греции был изобретен вид шифра, который в дальнейшем стал называться «квадрат Полития». Одну из первых книг по криптографии написал аббат И. Трителий (1462–1516), живший в Германии. В 1566 году известный математик Д. Кардано опубликовал работу с описанием изобретенной им системы шифрования («решетка Кардано»). Франция XVI века оставила в истории криптографии шифры короля Генриха IV и Ришелье. В упомянутой книге Т. А. Соболевой подробно описано много российских шифров, в том числе и «цифирная азбука» 1700 года, автором которой был Петр Великий. (Некоторые примеры из книги приведены на форзаце.)

Некоторые сведения о свойствах шифров и их применении можно найти и в художественной литературе, особенно в приключенческой, детективной и военной. Хорошее подробное объяснение особенностей одного из простейших шифров — *шифра замены* и методов его вскрытия содержится в двух известных рассказах: «Золотой жук» Э. По и «Пляшущие человечки» А. Конан Дойла.

Рассмотрим более подробно два примера.

Шифр «Сцитала». Этот шифр известен со времен войны Спарты против Афин в V веке до н.э. Для его реализации использовалась сцитала — жезл, имеющий форму цилиндра. На сциталу виток к витку наматывалась узкая папирусная лента (без просветов и нахлестов), а затем на этой ленте вдоль оси сциталы записывался открытый текст. Лента разматывалась и получалось (для непосвященных), что поперек ленты в беспорядке написаны какие-то буквы. Затем лента отправлялась адресату. Адресат брал такую же сциталу, таким же образом наматывал на нее полученную ленту и читал сообщение вдоль оси сциталы.

Отметим, что в этом шифре преобразование открытого текста в шифрованный заключается в определенной перестановке букв открытого текста.

---

[1] *Kahn David.* Codebreakers. The story of Secret Writing. New York: Macmillan, 1967.

[2] *Соболева Т. А.* Тайнопись в истории России (История криптографической службы России XVIII – начала XX в.). М., 1994.

Поэтому класс шифров, к которым относится и шифр «Сцитала», называется *шифрами перестановки.*

Шифр Цезаря. Этот шифр реализует следующее преобразование открытого текста: каждая буква открытого текста заменяется третьей после нее буквой в алфавите, который считается написанным по кругу, т. е. после буквы «я» следует буква «а». Отметим, что Цезарь заменял букву третьей после нее буквой, но можно заменять и какой-нибудь другой. Главное, чтобы тот, кому посылается шифрованное сообщение, знал эту величину сдвига. Класс шифров, к которым относится и шифр Цезаря, называется *шифрами замены.*

Из предыдущего изложения понятно, что придумывание хорошего шифра — дело трудоемкое. Поэтому желательно увеличить «время жизни» хорошего шифра и использовать его для шифрования как можно большего количества сообщений. Но при этом возникает опасность, что противник уже разгадал (вскрыл) шифр и читает защищаемую информацию. Если же в шифре есть сменный ключ, то, заменив ключ, можно сделать так, что разработанные противником методы уже не дают эффекта.

Под *ключом* в криптографии понимают сменный элемент шифра, который применяется для шифрования конкретного сообщения. Например, в шифре «Сцитала» ключом является диаметр сциталы, а в шифрах типа шифра Цезаря ключом является величина сдвига букв шифртекста относительно букв открытого текста.

Описанные соображения привели к тому, что безопасность защищаемой информации стала определяться в первую очередь ключом. Сам шифр, шифрмашина или принцип шифрования стали считать известными противнику и доступными для предварительного изучения, но в них появился неизвестный для противника ключ, от которого существенно зависят применяемые преобразования информации. Теперь законные пользователи, прежде чем обмениваться шифрованными сообщениями, должны тайно от противника обменяться ключами или установить одинаковый ключ на обоих концах канала связи. А для противника появилась новая задача — определить ключ, после чего можно легко прочитать зашифрованные на этом ключе сообщения.

Вернемся к формальному описанию основного объекта криптографии (рис. 1, стр. 9). Теперь в него необходимо внести существенное изменение — добавить недоступный для противника секретный канал связи для обмена ключами (см. рис. 2). Создать такой канал связи вполне реально, поскольку нагрузка на него, вообще говоря, небольшая.

Отметим теперь, что не существует единого шифра, подходящего для всех случаев. Выбор способа шифрования зависит от особенностей информации, ее ценности и возможностей владельцев по защите своей информации. Прежде всего подчеркнем большое разно-

Рис. 2.

образие видов защищаемой информации: документальная, телефонная, телевизионная, компьютерная и т. д. Каждый вид информации имеет свои специфические особенности, и эти особенности сильно влияют на выбор методов шифрования информации. Большое значение имеют объемы и требуемая скорость передачи шифрованной информации. Выбор вида шифра и его параметров существенно зависит от характера защищаемых секретов или тайны. Некоторые тайны (например, государственные, военные и др.) должны сохраняться десятилетиями, а некоторые (например, биржевые) — уже через несколько часов можно разгласить. Необходимо учитывать также и возможности того противника, от которого защищается данная информация. Одно дело — противостоять одиночке или даже банде уголовников, а другое дело — мощной государственной структуре.

Способность шифра противостоять всевозможным атакам на него называют *стойкостью шифра.*

Под *атакой на шифр* понимают попытку вскрытия этого шифра.

Понятие стойкости шифра является центральным для криптографии. Хотя качественно понять его довольно легко, но получение строгих доказуемых оценок стойкости для каждого конкретного шифра — проблема нерешенная. Это объясняется тем, что до сих пор нет необходимых для решения такой проблемы математических результатов. (Мы вернемся к обсуждению этого вопроса ниже.) Поэтому стойкость конкретного шифра оценивается только путем всевозможных попыток его вскрытия и зависит от квалификации *криптоаналитиков*, атакующих шифр. Такую процедуру иногда называют *проверкой стойкости.*

Важным подготовительным этапом для проверки стойкости шифра является продумывание различных предполагаемых возможностей, с помощью которых противник может атаковать шифр. Появление таких возможностей у противника обычно не зависит от криптографии, это является некоторой внешней подсказкой и существенно влияет на стойкость шифра. Поэтому оценки стойкости шифра всегда содержат те предположения о целях и возможностях противника, в условиях которых эти оценки получены.

Прежде всего, как это уже отмечалось выше, обычно считается, что противник знает сам шифр и имеет возможности для его предварительного изучения. Противник также знает некоторые характеристики открытых текстов, например, общую тематику сообщений, их стиль, некоторые стандарты, форматы и т. д.

Из более специфических приведем еще три примера возможностей противника:

– противник может перехватывать все шифрованные сообщения, но не имеет соответствующих им открытых текстов;

– противник может перехватывать все шифрованные сообщения и добывать соответствующие им открытые тексты;

– противник имеет доступ к шифру (но не к ключам!) и поэтому может зашифровывать и дешифровывать любую информацию.

На протяжении многих веков среди специалистов не утихали споры о стойкости шифров и о возможности построения абсолютно стойкого шифра. Приведем три характерных высказывания на этот счет.

Английский математик Чарльз Беббидж (XIX в.):
«Всякий человек, даже если он не знаком с техникой вскрытия шифров, твердо считает, что сможет изобрести абсолютно стойкий шифр, и чем более умен и образован этот человек, тем более твердо это убеждение. Я сам разделял эту уверенность в течение многих лет.»

«Отец кибернетики» Норберт Винер:
«Любой шифр может быть вскрыт, если только в этом есть настоятельная необходимость и информация, которую предполагается получить, стоит затраченных средств, усилий и времени...»

Автор шифра PGP Ф. Зиммерманн («Компьютерра», №48 от 1.12.1997, стр. 45–46):
«Каждый, кто думает, что изобрел непробиваемую схему шифрования, — или невероятно редкий гений, или просто наивен и неопытен...»
«Каждый программист воображает себя криптографом, что ведет к распространению исключительно плохого криптообеспечения...»

В заключение данного раздела сделаем еще одно замечание — о терминологии. В последнее время наряду со словом «криптография» часто встречается и слово «криптология», но соотношение между ними не всегда понимается правильно. Сейчас происходит окончательное формирование этих научных дисциплин, уточняются их предмет и задачи.

Криптология — наука, состоящая из двух ветвей: криптографии и криптоанализа.

Криптография — наука о способах преобразования (шифрования) информации с целью ее защиты от незаконных пользователей.

Криптоанализ — наука (и практика ее применения) о методах и способах вскрытия шифров.

Соотношение криптографии и криптоанализа очевидно: криптография — защита, т. е. разработка шифров, а криптоанализ — нападение, т. е. атака на шифры. Однако эти две дисциплины связаны друг с другом, и не бывает хороших криптографов, не владеющих методами криптоанализа.

3. Математические основы

Большое влияние на развитие криптографии оказали появившиеся в середине XX века работы американского математика Клода Шеннона. В этих работах были заложены основы теории информации, а также был разработан математический аппарат для исследований во многих областях науки, связанных с информацией. Более того, принято считать, что теория информации как наука родилась в 1948 году после публикации работы К. Шеннона «Математическая теория связи»[1].

В своей работе «Теория связи в секретных системах» Клод Шеннон обобщил накопленный до него опыт разработки шифров[2]. Оказалось, что даже в очень сложных шифрах в качестве типичных компонентов можно выделить такие простые шифры как *шифры замены, шифры перестановки* или их сочетания.

Шифр замены является простейшим, наиболее популярным шифром. Типичными примерами являются шифр Цезаря, «цифирная азбука» Петра Великого и «пляшущие человечки» А. Конан Дойла. Как видно из самого названия, шифр замены осуществляет преобразование замены букв или других «частей» открытого текста на аналогичные «части» шифрованного текста. Легко дать математическое описание шифра замены. Пусть X и Y — два алфавита (открытого и шифрованного текстов соответственно), состоящие из одинакового числа символов. Пусть также $g\colon X \to Y$ — взаимнооднозначное отображение X в Y. Тогда шифр замены действует так: открытый текст $x_1x_2\ldots x_n$ преобразуется в шифрованный текст $g(x_1)g(x_2)\ldots g(x_n)$.

Шифр перестановки, как видно из названия, осуществляет преобразование перестановки букв в открытом тексте. Типичным примером шифра перестановки является шифр «Сцитала». Обычно открытый текст разбивается на отрезки равной длины и каждый отрезок шифруется независимо. Пусть, например, длина отрезков равна n и σ — взаимнооднозначное отображение множества $\{1, 2, \ldots, n\}$ в себя. Тогда шифр перестановки действует так: отрезок открытого текста $x_1 \ldots x_n$ преобразуется в отрезок шифрованного текста $x_{\sigma(1)} \ldots x_{\sigma(n)}$.

---

[1] *Shannon C. E.* A mathematical theory of communication // Bell System Techn. J. V. 27, №3, 1948. P. 379–423; V. 27, №4, 1948. P. 623–656.

[2] См. Приложение.

Важнейшим для развития криптографии был результат К. Шеннона о существовании и единственности абсолютно стойкого шифра. Единственным таким шифром является какая-нибудь форма так называемой *ленты однократного использования*, в которой открытый текст «объединяется» с полностью случайным ключом такой же длины.

Этот результат был доказан К. Шенноном с помощью разработанного им теоретико-информационного метода исследования шифров. Мы не будем здесь останавливаться на этом подробно, заинтересованному читателю рекомендуем изучить работу К. Шеннона[1].

Обсудим особенности строения абсолютно стойкого шифра и возможности его практического использования. Типичным и наиболее простым примером реализации абсолютно стойкого шифра является шифр Вернама, который осуществляет побитовое сложение n-битового открытого текста и n-битового ключа:

$$y_i = x_i \oplus k_i, \quad i = 1, \ldots, n.$$

Здесь $x_1 \ldots x_n$ — открытый текст, $k_1, \ldots, k_n$ — ключ, $y_1 \ldots y_n$ — шифрованный текст.

Подчеркнем, что для абсолютной стойкости существенным является каждое из следующих требований к ленте однократного использования:

1) полная случайность (равновероятность) ключа (это, в частности, означает, что ключ нельзя вырабатывать с помощью какого-либо детерминированного устройства);

2) равенство длины ключа и длины открытого текста;

3) однократность использования ключа.

В случае нарушения хотя бы одного из этих условий шифр перестает быть абсолютно стойким и появляются принципиальные возможности для его вскрытия (хотя они могут быть трудно реализуемыми).

Но, оказывается, именно эти условия и делают абсолютно стойкий шифр очень дорогим и непрактичным. Прежде чем пользоваться таким шифром, мы должны обеспечить всех абонентов достаточным запасом случайных ключей и исключить возможность их повторного применения. А это сделать необычайно трудно и дорого.

Как отмечал Д. Кан: «Проблема создания, регистрации, распространения и отмены ключей может показаться не слишком сложной тому, кто не имеет опыта передачи сообщений по каналам военной связи, но в военное время объем передаваемых сообщений ставит в тупик даже профессиональных связистов. За сутки могут быть зашифрованы сотни тысяч слов. Создание миллионов ключевых знаков потребовало бы огромных финансовых издержек и было бы сопряжено с большими затратами времени. Так как каждый текст должен иметь свой собственный, единственный и неповторимый ключ,

[1] См. Приложение.

применение идеальной системы потребовало бы передачи по крайней мере такого количества знаков, которое эквивалентно всему объему передаваемой военной информации.»

В силу указанных причин абсолютно стойкие шифры применяются только в сетях связи с небольшим объемом передаваемой информации, обычно это сети для передачи особо важной государственной информации.

Теперь уже понятно, что чаще всего для защиты своей информации законные пользователи вынуждены применять неабсолютно стойкие шифры. Такие шифры, по крайней мере теоретически, могут быть вскрыты. Вопрос только в том, хватит ли у противника сил, средств и времени для разработки и реализации соответствующих алгоритмов. Обычно эту мысль выражают так: противник с неограниченными ресурсами может вскрыть любой неабсолютно стойкий шифр.

Как же должен действовать в этой ситуации законный пользователь, выбирая для себя шифр? Лучше всего, конечно, было бы доказать, что никакой противник не может вскрыть выбранный шифр, скажем, за 10 лет и тем самым получить теоретическую оценку стойкости. К сожалению, математическая теория еще не дает нужных теорем — они относятся к нерешенной *проблеме нижних оценок вычислительной сложности задач.*

Поэтому у пользователя остается единственный путь — получение практических оценок стойкости. Этот путь состоит из следующих этапов:

– понять и четко сформулировать, от какого противника мы собираемся защищать информацию; необходимо уяснить, что именно противник знает или сможет узнать о системе шифра, а также какие силы и средства он сможет применить для его вскрытия;

– мысленно стать в положение противника и пытаться с его позиций атаковать шифр, т. е. разрабатывать различные алгоритмы вскрытия шифра; при этом необходимо в максимальной мере обеспечить моделирование сил, средств и возможностей противника;

– наилучший из разработанных алгоритмов использовать для практической оценки стойкости шифра.

Здесь полезно для иллюстрации упомянуть о двух простейших методах вскрытия шифра: случайное угадывание ключа (он срабатывает с маленькой вероятностью, зато имеет маленькую сложность) и перебор всех подряд ключей вплоть до нахождения истинного (он срабатывает всегда, зато имеет очень большую сложность). Отметим также, что не всегда нужна атака на ключ: для некоторых шифров можно сразу, даже не зная ключа, восстанавливать открытый текст по шифрованному.

4. Новые направления

В 1983 году в книге «Коды и математика» М. Н. Аршинова и Л. Е. Садовского (библиотечка «Квант») было написано: «Приемов тайнописи — великое множество, и, скорее всего, это та область, где уже нет нужды придумывать что-нибудь существенно новое.» Однако это было очередное большое заблуждение относительно криптографии. Еще в 1976 году была опубликована работа молодых американских математиков У. Диффи и М. Э. Хеллмана «Новые направления в криптографии»[1], которая не только существенно изменила криптографию, но и привела к появлению и бурному развитию новых направлений в математике. Центральным понятием «новой криптографии» является понятие односторонней функции (подробнее об этом см. главу 2).

Односторонней называется функция $F\colon X \to Y$, обладающая двумя свойствами:

а) существует полиномиальный алгоритм вычисления значений $F(x)$;

б) не существует полиномиального алгоритма *инвертирования* функции F (т. е. решения уравнения $F(x) = y$ относительно x, точное определение см. на стр. 30).

Отметим, что односторонняя функция существенно отличается от функций, привычных со школьной скамьи, из-за ограничений на сложность ее вычисления и инвертирования. Вопрос о существовании односторонних функций пока открыт.

Еще одним новым понятием является понятие *функции с секретом*. Иногда еще употребляется термин *функция с ловушкой*. *Функцией с секретом* K называется функция $F_K\colon X \to Y$, зависящая от параметра K и обладающая тремя свойствами:

а) существует полиномиальный алгоритм вычисления значения $F_K(x)$ для любых K и x;

б) не существует полиномиального алгоритма инвертирования F_K при неизвестном K;

в) существует полиномиальный алгоритм инвертирования F_K при известном K.

Про существование функций с секретом можно сказать то же самое, что сказано про односторонние функции. Для практических целей криптографии было построено несколько функций, которые могут оказаться функциями с секретом. Для них свойство б) пока строго не доказано, но считается, что задача инвертирования эквивалентна некоторой давно изучаемой трудной математической задаче. Наиболее известной и популярной из них является теоретико-числовая функция, на которой построен шифр RSA (подробнее об этом см. главу 4).

---

[1] *Диффи У., Хеллман М. Э.* Защищенность и имитостойкость. Введение в криптографию // ТИИЭР. Т. 67, №3, 1979.

Применение функций с секретом в криптографии позволяет:

1) организовать обмен шифрованными сообщениями с использованием только открытых каналов связи, т. е. отказаться от секретных каналов связи для предварительного обмена ключами;

2) включить в задачу вскрытия шифра трудную математическую задачу и тем самым повысить обоснованность стойкости шифра;

3) решать новые криптографические задачи, отличные от шифрования (*электронная цифровая подпись* и др.).

Опишем, например, как можно реализовать п. 1). Пользователь A, который хочет получать шифрованные сообщения, должен выбрать какую-нибудь функцию F_K с секретом K. Он сообщает всем заинтересованным (например, публикует) описание функции F_K в качестве своего алгоритма шифрования. Но при этом значение секрета K он никому не сообщает и держит в секрете. Если теперь пользователь B хочет послать пользователю A защищаемую информацию $x \in X$, то он вычисляет $y = F_K(x)$ и посылает y по открытому каналу пользователю A. Поскольку A для своего секрета K умеет инвертировать F_K, то он вычисляет x по полученному y. Никто другой не знает K и поэтому в силу свойства б) функции с секретом не сможет за полиномиальное время по известному шифрованному сообщению $F_K(x)$ вычислить защищаемую информацию x.

Описанную систему называют *криптосистемой с открытым ключом*, поскольку алгоритм шифрования F_K является общедоступным или открытым. В последнее время такие криптосистемы еще называют *асимметричными*, поскольку в них есть асимметрия в алгоритмах: алгоритмы шифрования и дешифрования различны. В отличие от таких систем традиционные шифры называют *симметричными*: в них ключ для шифрования и дешифрования один и тот же. Для асимметричных систем алгоритм шифрования общеизвестен, но восстановить по нему алгоритм дешифрования за полиномиальное время невозможно.

Описанную выше идею Диффи и Хеллман предложили использовать также для электронной цифровой подписи сообщений, которую невозможно подделать за полиномиальное время. Пусть пользователю A необходимо подписать сообщение x. Он, зная секрет K, находит такое y, что $F_K(y) = x$, и вместе с сообщением x посылает y пользователю B в качестве своей цифровой подписи. Пользователь B хранит y в качестве доказательства того, что A подписал сообщение x.

Сообщение, подписанное цифровой подписью, можно представлять себе как пару (x, y), где x — сообщение, y — решение уравнения $F_K(y) = x$, $F_K\colon X \to Y$ — функция с секретом, известная всем взаимодействующим абонентам. Из определения функции F_K очевидны следующие полезные свойства цифровой подписи:

1) подписать сообщение x, т. е. решить уравнение $F_K(y) = x$, может только абонент — обладатель данного секрета K; другими словами, подделать подпись невозможно;

2) проверить подлинность подписи может любой абонент, знающий открытый ключ, т. е. саму функцию F_K;

3) при возникновении споров отказаться от подписи невозможно в силу ее неподделываемости;

4) подписанные сообщения (x, y) можно, не опасаясь ущерба, пересылать по любым каналам связи.

Кроме принципа построения криптосистемы с открытым ключом, Диффи и Хеллман в той же работе предложили еще одну новую идею — *открытое распределение ключей.* Они задались вопросом: можно ли организовать такую процедуру взаимодействия абонентов A и B по открытым каналам связи, чтобы решить следующие задачи:

1) вначале у A и B нет никакой общей секретной информации, но в конце процедуры такая общая секретная информация (общий ключ) у A и B появляется, т. е. вырабатывается;

2) пассивный противник, который перехватывает все передачи информации и знает, что хотят получить A и B, тем не менее не может восстановить выработанный общий ключ A и B.

Диффи и Хеллман предложили решать эти задачи с помощью функции

$$F(x) = \alpha^x \pmod p,$$

где p — большое простое число, x — произвольное натуральное число, α — некоторый *примитивный элемент* поля $GF(p)$. Общепризнанно, что инвертирование функции $\alpha^x \pmod p$, т. е. дискретное логарифмирование, является трудной математической задачей. (Подробнее см. главу 4.)

Сама процедура или, как принято говорить, *протокол выработки общего ключа* описывается следующим образом.

Абоненты A и B независимо друг от друга случайно выбирают по одному натуральному числу — скажем x_A и x_B. Эти элементы они держат в секрете. Далее каждый из них вычисляет новый элемент:

$$y_A = \alpha^{x_A} \pmod p, \quad y_B = \alpha^{x_B} \pmod p.$$

(Числа p и α считаются общедоступными.) Потом они обмениваются этими элементами по каналу связи. Теперь абонент A, получив y_B и зная свой секретный элемент x_A, вычисляет новый элемент:

$$y_B^{x_A} \pmod p = (\alpha^{x_B})^{x_A} \pmod p.$$

Аналогично поступает абонент B:

$$y_A^{x_B} \pmod p = (\alpha^{x_A})^{x_B} \pmod p.$$

Тем самым у A и B появился общий элемент поля, равный $\alpha^{x_A x_B}$. Этот элемент и объявляется общим ключом A и B.

Из описания протокола видно, что противник знает $p, \alpha, \alpha^{x_A}, \alpha^{x_B}$, не знает x_A и x_B и хочет узнать $\alpha^{x_A x_B}$. В настоящее время нет алгоритмов действий противника, более эффективных, чем дискретное логарифмирование, а это — трудная математическая задача.

Успехи, достигнутые в разработке схем цифровой подписи и открытого распределения ключей, позволили применить эти идеи также и к другим задачам взаимодействия удаленных абонентов. Так возникло большое новое направление теоретической криптографии — криптографические протоколы (подробнее см. главу 3).

Объектом изучения теории криптографических протоколов являются удаленные абоненты, взаимодействующие, как правило, по открытым каналам связи. Целью взаимодействия абонентов является решение какой-то задачи. Имеется также противник, который преследует собственные цели. При этом противник в разных задачах может иметь разные возможности: например, может взаимодействовать с абонентами от имени других абонентов или вмешиваться в обмены информацией между абонентами и т. д. Противником может даже оказаться один из абонентов или несколько абонентов, вступивших в сговор.

Приведем еще несколько примеров задач, решаемых удаленными абонентами. (Читателю рекомендуем по своему вкусу самостоятельно придумать еще какие-нибудь примеры.)

1. Взаимодействуют два не доверяющих друг другу абонента. Они хотят подписать контракт. Это надо сделать так, чтобы не допустить следующую ситуацию: один из абонентов получил подпись другого, а сам не подписался.

Протокол решения этой задачи принято называть *протоколом подписания контракта.*

2. Взаимодействуют два не доверяющих друг другу абонента. Они хотят бросить жребий с помощью монеты. Это надо сделать так, чтобы абонент, подбрасывающий монету, не мог изменить результат подбрасывания после получения догадки от абонента, угадывающего этот результат.

Протокол решения этой задачи принято называть *протоколом подбрасывания монеты.*

Опишем один из простейших протоколов подбрасывания монеты по телефону (так называемая схема Блюма-Микали). Для его реализации у абонентов A и B должна быть односторонняя функция $f\colon X \to Y$, удовлетворяющая следующим условиям:

1) X — множество целых чисел, которое содержит одинаковое количество четных и нечетных чисел;

2) любые числа $x_1, x_2 \in X$, имеющие один образ $f(x_1) = f(x_2)$, имеют одну четность;

3) по заданному образу $f(x)$ «трудно» вычислить четность неизвестного аргумента x.

Роль подбрасывания монеты играет случайный и равновероятный выбор элемента $x \in X$, а роль орла и решки — четность и нечетность x соответственно. Пусть A — абонент, подбрасывающий монету, а B — абонент, угадывающий результат. Протокол состоит из следующих шагов:

1) A выбирает x («подбрасывает монету»), зашифровывает x, т. е. вычисляет $y = f(x)$, и посылает y абоненту B;

2) B получает y, пытается угадать четность x и посылает свою догадку абоненту A;

3) A получает догадку от B и сообщает B, угадал ли он, посылая ему выбранное число x;

4) B проверяет, не обманывает ли A, вычисляя значение $f(x)$ и сравнивая его с полученным на втором шаге значением y.

3. Взаимодействуют два абонента A и B (типичный пример: A — клиент банка, B — банк). Абонент A хочет доказать абоненту B, что он именно A, а не противник.

Протокол решения этой задачи принято называть *протоколом идентификации абонента.*

4. Взаимодействуют несколько удаленных абонентов, получивших приказы из одного центра. Часть абонентов, включая центр, могут быть противниками. Необходимо выработать единую стратегию действий, выигрышную для абонентов.

Эту задачу принято называть задачей о византийских генералах, а протокол ее решения — *протоколом византийского соглашения.*

Опишем пример, которому эта задача обязана своим названием. Византия. Ночь перед великой битвой. Византийская армия состоит из n легионов, каждый из которых подчиняется своему генералу. Кроме того, у армии есть главнокомандующий, который руководит генералами. Однако империя находится в упадке и до одной трети генералов, включая главнокомандующего, могут быть предателями. В течение ночи каждый из генералов получает от главнокомандующего приказ о действиях на утро, причем возможны два варианта приказа: «атаковать» или «отступать». Если все честные генералы атакуют, то они побеждают. Если все они отступают, то им удается сохранить армию. Но если часть из них атакует, а часть отступает, то они терпят поражение. Если главнокомандующий окажется предателем, то он может дать разным генералам разные приказы, поэтому приказы главнокомандующего не стоит выполнять беспрекословно. Если каждый генерал будет действовать независимо от остальных, результаты могут оказаться плачевными. Очевидно, что генералы нуждаются в обмене информацией друг с другом (относительно полученных приказов) с тем, чтобы прийти к соглашению.

Осмысление различных протоколов и методов их построения привело в 1985–1986 г.г. к появлению двух плодотворных математических моделей — *интерактивной системы доказательства* и *доказательства с нулевым разглашением*. Математические исследования этих новых объектов позволили доказать много утверждений, весьма полезных при разработке криптографических протоколов (подробнее об этом см. главу 2).

Под интерактивной системой доказательства (P, V, S) понимают протокол взаимодействия двух абонентов: P (доказывающий) и V (проверяющий). Абонент P хочет доказать V, что утверждение S истинно. При этом абонент V самостоятельно, без помощи P, не может проверить утверждение S (поэтому V и называется проверяющим). Абонент P может быть и противником, который хочет доказать V, что утверждение S истинно, хотя оно ложно. Протокол может состоять из многих *раундов* обмена сообщениями между P и V и должен удовлетворять двум условиям:

1) *полнота* — если S действительно истинно, то абонент P убедит абонента V признать это;

2) *корректность* — если S ложно, то абонент P вряд ли убедит абонента V, что S истинно.

Здесь словами «вряд ли» мы для простоты заменили точную математическую формулировку.

Подчеркнем, что в определении системы (P, V, S) не допускалось, что V может быть противником. А если V оказался противником, который хочет «выведать» у P какую-нибудь новую полезную для себя информацию об утверждении S? В этом случае P, естественно, может не хотеть, чтобы это случилось в результате работы протокола (P, V, S). Протокол (P, V, S), решающий такую задачу, называется доказательством с нулевым разглашением и должен удовлетворять, кроме условий 1) и 2), еще и следующему условию:

3) *нулевое разглашение* — в результате работы протокола (P, V, S) абонент V не увеличит свои знания об утверждении S или, другими словами, не сможет извлечь никакой информации о том, почему S истинно.

5. Заключение

За последние годы криптография и криптографические методы все шире входят в нашу жизнь и даже быт. Вот несколько примеров. Отправляя Email, мы в некоторых случаях отвечаем на вопрос меню: «Нужен ли режим зашифрования?» Владелец интеллектуальной банковской карточки, обращаясь через терминал к банку, вначале выполняет

криптографический протокол аутентификации карточки. Пользователи сети Интернет наверняка знакомы с дискуссиями вокруг возможного принятия стандарта цифровой подписи для тех страниц, которые содержат «критическую» информацию (юридическую, прайс-листы и др.). С недавних пор пользователи сетей стали указывать после своей фамилии наряду с уже привычным «Email ... » и менее привычное — «Отпечаток открытого ключа ... ».

С каждым днем таких примеров становится все больше. Именно новые практические приложения криптографии и являются одним из источников ее развития.

Глава 2

Криптография и теория сложности

Основное внимание в настоящей главе мы уделяем разъяснению важнейших идей, связанных с применением теоретико-сложностного подхода в криптографии. Изложение по необходимости недостаточно формальное — для математической криптографии типичны многостраничные определения. Предполагается знакомство читателя с основами теории сложности вычислений: понятиями машины Тьюринга, классов P и NP (см. [2]), а также с главой 1 настоящей книги.

1. Введение

В теоретической криптографии существуют два основных подхода к определению стойкости криптосистем и криптографических протоколов (в дальнейшем мы будем также использовать общий термин — криптографические схемы): теоретико-информационный и теоретико-сложностной. Теоретико-информационный подход предполагает, что противник, атакующий криптографическую схему, не имеет даже теоретической возможности получить информацию, достаточную для осуществления своих целей. Классическим примером здесь может служить шифр Вернама с одноразовыми ключами, абсолютно стойкий против пассивного противника.

Подавляющее большинство используемых на практике криптографических схем не обладает столь высокой стойкостью. Более того, обычно бывает несложно указать алгоритм, который выполняет стоящую перед противником задачу, но не практически, а в принципе. Рассмотрим следующий пример.

Пример 1 (Криптосистема с открытым ключом). *Криптосистема с открытым ключом* полностью определяется тремя алгоритмами: генерации ключей, шифрования и дешифрования. Алгоритм генерации ключей G общедоступен; всякий желающий может подать ему на вход случайную строку r надлежащей длины и получить пару ключей (K_1, K_2). Открытый ключ K_1 публикуется, а секретный ключ K_2 и случайная строка r хранятся в секрете. Алгоритмы шифрования E_{K_1} и

дешифрования D_{K_2} таковы, что если (K_1, K_2) — пара ключей, сгенерированная алгоритмом G, то $D_{K_2}(E_{K_1}(m)) = m$ для любого открытого текста m. Для простоты изложения предполагаем, что открытый текст и криптограмма имеют одинаковую длину n. Кроме того, считаем, что открытый текст, криптограмма и оба ключа являются строками в двоичном алфавите.

Предположим теперь, что противник атакует эту криптосистему. Ему известен открытый ключ K_1, но неизвестен соответствующий секретный ключ K_2. Противник перехватил криптограмму d и пытается найти сообщение m, где $d = E_{K_1}(m)$. Поскольку алгоритм шифрования общеизвестен, противник может просто последовательно перебрать все возможные сообщения длины n, вычислить для каждого такого сообщения m_i криптограмму $d_i = E_{K_1}(m_i)$ и сравнить d_i с d. То сообщение, для которого $d_i = d$, и будет искомым открытым текстом. Если повезет, то открытый текст будет найден достаточно быстро. В худшем же случае перебор будет выполнен за время порядка $2^n T(n)$, где $T(n)$ — время, требуемое для вычисления функции E_{K_1} от сообщений длины n. Если сообщения имеют длину порядка 1000 битов, то такой перебор неосуществим на практике ни на каких самых мощных компьютерах.

Мы рассмотрели лишь один из возможных способов атаки на криптосистему и простейший алгоритм поиска открытого текста, называемый обычно алгоритмом полного перебора. Используется также и другое название: «метод грубой силы». Другой простейший алгоритм поиска открытого текста — угадывание. Этот очевидный алгоритм требует небольших вычислений, но срабатывает с пренебрежимо малой вероятностью (при больших длинах текстов). На самом деле противник может пытаться атаковать криптосистему различными способами и использовать различные, более изощренные алгоритмы поиска открытого текста. Естественно считать криптосистему стойкой, если любой такой алгоритм требует практически неосуществимого объема вычислений или срабатывает с пренебрежимо малой вероятностью. (При этом противник может использовать не только детерминированные, но и вероятностные алгоритмы.) Это и есть теоретико-сложностной подход к определению стойкости. Для его реализации в отношении того или иного типа криптографических схем необходимо выполнить следующее:

1) дать формальное определение схемы данного типа;

2) дать формальное определение стойкости схемы;

3) доказать стойкость конкретной конструкции схемы данного типа.

Здесь сразу же возникает ряд проблем.

Во-первых, в криптографических схемах, разумеется, всегда используются фиксированные значения параметров. Например, крипто-

системы разрабатываются для ключей длины, скажем, в 256 или 512 байтов. Для применения же теоретико-сложностного подхода необходимо, чтобы задача, вычислительную сложность которой предполагается использовать, была массовой. Поэтому в теоретической криптографии рассматриваются математические модели криптографических схем. Эти модели зависят от некоторого параметра, называемого параметром безопасности, который может принимать сколь угодно большие значения (обычно для простоты предполагается, что параметр безопасности может пробегать весь натуральный ряд).

Во-вторых, определение стойкости криптографической схемы зависит от той задачи, которая стоит перед противником, и от того, какая информация о схеме ему доступна. Поэтому стойкость схем приходится определять и исследовать отдельно для каждого предположения о противнике.

В-третьих, необходимо уточнить, какой объем вычислений можно считать «практически неосуществимым». Из сказанного выше следует, что эта величина не может быть просто константой, она должна быть представлена функцией от растущего параметра безопасности. В соответствии с тезисом Эдмондса алгоритм считается эффективным, если время его выполнения ограничено некоторым полиномом от длины входного слова (в нашем случае — от параметра безопасности). В противном случае говорят, что вычисления по данному алгоритму практически неосуществимы. Заметим также, что сами криптографические схемы должны быть эффективными, т. е. все вычисления, предписанные той или иной схемой, должны выполняться за полиномиальное время.

В-четвертых, необходимо определить, какую вероятность можно считать пренебрежимо малой. В криптографии принято считать таковой любую вероятность, которая для любого полинома p и для всех достаточно больших n не превосходит $1/p(n)$, где n — параметр безопасности.

Итак, при наличии всех указанных выше определений, проблема обоснования стойкости криптографической схемы свелась к доказательству отсутствия полиномиального алгоритма, который решает задачу, стоящую перед противником. Но здесь возникает еще одно и весьма серьезное препятствие: современное состояние теории сложности вычислений не позволяет доказывать сверхполиномиальные нижние оценки сложности для конкретных задач рассматриваемого класса. Из этого следует, что на данный момент стойкость криптографических схем может быть установлена лишь с привлечением каких-либо недоказанных предположений. Поэтому основное направление исследований состоит в поиске наиболее слабых достаточных условий (в идеале — необходимых и достаточных) для существования стойких схем каждого из типов.

В основном, рассматриваются предположения двух типов — общие (или теоретико-сложностные) и теоретико-числовые, т. е. предположения о сложности конкретных теоретико-числовых задач. Все эти предположения в литературе обычно называются криптографическими.

Ниже мы кратко рассмотрим несколько интересных математических объектов, возникших на стыке теории сложности и криптографии. Более подробный обзор по этим вопросам можно найти в книге [1].

2. Криптография и гипотеза P≠NP

Как правило, знакомство математиков-неспециалистов с теорией сложности вычислений ограничивается классами P и NP и знаменитой гипотезой P≠NP.

Напомним вкратце необходимые сведения из теории сложности вычислений. Пусть Σ^* — множество всех конечных строк в двоичном алфавите $\Sigma = \{0, 1\}$. Подмножества $L \subseteq \Sigma^*$ в теории сложности принято называть языками. Говорят, что машина Тьюринга M работает за полиномиальное время (или просто, что она полиномиальна), если существует полином p такой, что на любом входном слове длины n машина M останавливается после выполнения не более, чем $p(n)$ операций. Машина Тьюринга M распознает (другой термин — принимает) язык L, если на всяком входном слове $x \in L$ машина M останавливается в принимающем состоянии, а на всяком слове $x \notin L$ — в отвергающем. Класс P — это класс всех языков, распознаваемых машинами Тьюринга, работающими за полиномиальное время. Функция $f : \Sigma^* \to \Sigma^*$ вычислима за полиномиальное время, если существует полиномиальная машина Тьюринга такая, что если на вход ей подано слово $x \in \Sigma^*$, то в момент останова на ленте будет записано значение $f(x)$. Язык L принадлежит классу NP, если существуют предикат $P(x, y) : \Sigma^* \times \Sigma^* \to \{0, 1\}$, вычислимый за полиномиальное время, и полином p такие, что $L = \{x | \exists y P(x, y) \& |y| \leqslant p(|x|)\}$. Таким образом, язык L принадлежит NP, если для всякого слова из L длины n можно угадать некоторую строку полиномиальной от n длины и затем с помощью предиката P убедиться в правильности догадки. Ясно, что $P \subseteq NP$. Является ли это включение строгим — одна из самых известных нерешенных задач математики. Большинство специалистов считают, что оно строгое (так называемая гипотеза P≠NP). В классе NP выделен подкласс максимально сложных языков, называемых NP-полными: любой NP-полный язык распознаваем за полиномиальное время тогда и только тогда, когда P=NP.

Для дальнейшего нам потребуется еще понятие вероятностной машины Тьюринга. В обычных машинах Тьюринга (их называют детерминированными, чтобы отличить от вероятностных) новое состояние, в которое машина переходит на очередном шаге, полностью определяется текущим состоянием и тем символом, который обозревает головка на ленте. В вероятностных машинах новое состояние может зависеть еще и от случайной величины, которая принимает значения 0 и 1 с вероятностью 1/2 каждое. Альтернативно,

можно считать, что вероятностная машина Тьюринга имеет дополнительную случайную ленту, на которой записана бесконечная двоичная случайная строка. Случайная лента может читаться только в одном направлении и переход в новое состояние может зависеть от символа, обозреваемого на этой ленте.

Рассмотрим теперь следующий естественный вопрос: не является ли гипотеза P≠NP необходимым и достаточным условием для существования стойких криптографических схем?

Необходимость, и в самом деле, во многих случаях почти очевидна. Вернемся к примеру 1. Определим следующий язык

$L = \{(K_1, d, i) |$ существует сообщение m такое, что $E_{K_1}(m) = d$ и его i-ый бит равен $1\}$.

Ясно, что $L \in$ NP: вместо описанного во введении полного перебора можно просто угадать открытый текст m и проверить за полиномиальное время, что $E_{K_1}(m) = d$ и i-ый бит m равен 1. Если да, то входное слово (K_1, d, i) принимается, в противном случае — отвергается.

В предположении P=NP существует детерминированный полиномиальный алгоритм, распознающий язык L. Зная K_1 и d, с помощью этого алгоритма можно последовательно, по биту, вычислить открытый текст m. Тем самым криптосистема нестойкая.

Тот же подход: угадать секретный ключ и проверить (за полиномиальное время) правильность догадки, применим в принципе и к другим криптографическим схемам. Однако, в некоторых случаях возникают технические трудности, связанные с тем, что по информации, которая имеется у противника, искомая величина (открытый текст, секретный ключ и т. п.) восстанавливается неоднозначно.

Что же касается вопроса о достаточности предположения P≠NP, то здесь напрашивается следующий подход: выбрать какую-либо NP-полную задачу и построить на ее основе криптографическую схему, задача взлома которой (т. е. задача, стоящая перед противником) была бы NP-полной. Такие попытки предпринимались в начале 80-х годов, в особенности в отношении криптосистем с открытым ключом, но к успеху не привели. Результатом всех этих попыток стало осознание следующего факта: даже если P≠NP, то любая NP-полная задача может оказаться трудной лишь на некоторой бесконечной последовательности входных слов. Иными словами, в определение класса NP заложена мера сложности «в худшем случае». Для стойкости же криптографической схемы необходимо, чтобы задача противника была сложной «почти всюду». Таким образом, стало ясно, что для криптографической стойкости необходимо существенно более сильное предположение, чем P≠NP. А именно, предположение о существовании односторонних функций.

3. Односторонние функции

Говоря неформально, односторонняя функция — это эффективно вычислимая функция, для задачи инвертирования которой не существует эффективных алгоритмов. Под инвертированием понимается массовая задача нахождения по заданному значению функции одного (любого) значения из прообраза (заметим, что обратная функция, вообще говоря, может не существовать).

Поскольку понятие односторонней функции — центральное в математической криптографии, ниже мы даем его формальное определение.

Пусть $\Sigma^n = \{0,1\}^n$ — множество всех двоичных строк длины n. Под функцией f мы понимаем семейство $\{f_n\}$, где $f_n : \Sigma^n \to \Sigma^m$, $m = m(n)$. Для простоты изложения мы предполагаем, что n пробегает весь натуральный ряд и что каждая из функций f_n всюду определена.

Функция f называется *честной*, если существует полином q такой, что $n \leqslant q(m(n))$ для всех n.

Определение 1. Честная функция f называется *односторонней*, если

1. Существует полиномиальный алгоритм, который для всякого x вычисляет $f(x)$.

2. Для любой полиномиальной вероятностной машины Тьюринга A выполнено следующее. Пусть строка x выбрана наудачу из множества Σ^n (обозначается $x \in_R \Sigma^n$). Тогда для любого полинома p и всех достаточно больших n

$$Pr\{f(A(f(x))) = f(x)\} < 1/p(n).$$

Вероятность здесь определяется случайным выбором строки x и случайными величинами, которые A использует в своей работе.

Условие 2 качественно означает следующее. Любая полиномиальная вероятностная машина Тьюринга A может по данному y найти x из уравнения $f(x) = y$ лишь с пренебрежимо малой вероятностью.

Заметим, что требование честности нельзя опустить. Поскольку длина входного слова $f(x)$ машины A равна m, ей может не хватить полиномиального (от m) времени просто на выписывание строки x, если f слишком сильно «сжимает» входные значения.

Ясно, что из предположения о существовании односторонних функций следует, что P≠NP. Однако, не исключена следующая ситуация: P≠NP, но односторонних функций нет.

Существование односторонних функций является необходимым условием для стойкости многих типов криптографических схем. В некоторых случаях этот факт устанавливается достаточно просто. Обратимся опять к примеру 1. Рассмотрим функцию f такую, что $f(r) = K_1$. Она вычислима с помощью алгоритма G за полиномиальное время. Покажем, что если f — не односторонняя функция, то криптосистема

нестойкая. Предположим, что существует полиномиальный вероятностный алгоритм A, который инвертирует f с вероятностью по крайней мере $1/p(n)$ для некоторого полинома p. Здесь n — длина ключа K_1. Противник может подать на вход алгоритму A ключ K_1 и получить с указанной вероятностью некоторое значение r' из прообраза. Далее противник подает r' на вход алгоритма G и получает пару ключей (K_1, K_2'). Хотя K_2' не обязательно совпадает с K_2, тем не менее, по определению криптосистемы $D_{K_2'}(E_{K_1}(m)) = m$ для любого открытого текста m. Поскольку K_2' найден с вероятностью по крайней мере $1/p(n)$, которая в криптографии не считается пренебрежимо малой, криптосистема нестойкая.

Для других криптографических схем подобный результат доказывается не столь просто. В работе Импальяццо и Луби [7] доказана необходимость односторонних функций для существования целого ряда стойких криптографических схем.

Из всего сказанного следует, что предположение о существовании односторонних функций является самым слабым криптографическим предположением, которое может оказаться достаточным для доказательства существования стойких криптографических схем различных типов. На выяснение того, является ли это условие и в самом деле достаточным, направлены значительные усилия специалистов. Трудность задачи построения криптографических схем из односторонних функций можно пояснить на следующем примере. Пусть f — односторонняя функция и нам требуется построить *криптосистему с секретным ключом*. В такой криптосистеме имеется только один ключ — секретный, который известен и отправителю, и получателю шифрованного сообщения. Алгоритмы шифрования E_K и дешифрования D_K оба зависят от этого секретного ключа K и таковы, что $D_K(E_K(m)) = m$ для любого открытого текста m. Ясно, что если криптограмму d сообщения m вычислять как $d = f(m)$, то противник, перехвативший d, может вычислить m лишь с пренебрежимо малой вероятностью. Но во-первых, непонятно, каким образом сможет восстановить сообщение m из криптограммы законный получатель? Во-вторых, из того, что функция f односторонняя следует лишь, что противник не может вычислить все сообщение целиком. А это — весьма низкий уровень стойкости. Желательно, чтобы противник, знающий криптограмму d, не мог вычислить ни одного бита открытого текста.

На настоящий момент доказано, что существование односторонних функций является необходимым и достаточным условием для существования стойких криптосистем с секретным ключом, а также стойких криптографических протоколов нескольких типов, включая протоколы электронной подписи. С другой стороны, имеется результат

Импальяццо и Рудиха [9], который является достаточно сильным аргументом в пользу того, что для некоторых типов криптографических схем (включая протоколы распределения ключей типа Диффи-Хеллмана) требуются более сильные предположения, чем предположение о существовании односторонних функций. К сожалению, этот результат слишком сложный, чтобы его можно было разъяснить в настоящей главе.

4. Псевдослучайные генераторы

Существенный недостаток шифра Вернама состоит в том, что ключи одноразовые. Можно ли избавиться от этого недостатка за счет некоторого снижения стойкости? Один из способов решения этой проблемы состоит в следующем. Отправитель и получатель имеют общий секретный ключ K длины n и с помощью некоторого достаточно эффективного алгоритма g генерируют из него последовательность $r = g(K)$ длины $q(n)$, где q — некоторый полином. Такая криптосистема (обозначим ее Cr) позволяет шифровать сообщение m (или совокупность сообщений) длиной до $q(n)$ битов по формуле $d = r \oplus m$, где $\oplus$ — поразрядное сложение битовых строк по модулю 2. Дешифрование выполняется по формуле $m = d \oplus r$. Из результатов Шеннона вытекает, что такая криптосистема не является абсолютно стойкой, т. е. стойкой против любого противника (в чем, впрочем, нетрудно убедиться и непосредственно). Но что будет, если требуется защищаться только от полиномиально ограниченного противника, который может атаковать криптосистему лишь с помощью полиномиальных вероятностных алгоритмов? Каким условиям должны удовлетворять последовательность r и алгоритм g, чтобы криптосистема Cr была стойкой? Поиски ответов на эти вопросы привели к появлению понятия псевдослучайного генератора, которое было введено Блюмом и Микали [3].

Пусть $g : \{0,1\}^n \to \{0,1\}^{q(n)}$ — функция, вычислимая за полиномиальное (от n) время. Такая функция называется генератором. Интуитивно, генератор g является псевдослучайным, если порождаемые им последовательности неотличимы никаким полиномиальным вероятностным алгоритмом от случайных последовательностей той же длины $q(n)$. Формально этот объект определяется следующим образом.

Пусть A — полиномиальная вероятностная машина Тьюринга, которая получает на входе двоичные строки длины $q(n)$ и выдает в результате своей работы один бит. Пусть

$$P_1(A,n) = Pr\{A(r) = 1 | r \in_R \{0,1\}^{q(n)}\}.$$

Вероятность здесь определяется случайным выбором строки r и случайными величинами, которые A использует в своей работе. Пусть

$$P_2(A,n) = Pr\{A(g(s)) = 1 | s \in_R \{0,1\}^n\}.$$

Эта вероятность определяется случайным выбором строки s и случайными величинами, которые A использует в своей работе. Подчеркнем, что функция g вычисляется детерминированным алгоритмом.

Определение 2. Генератор g называется *криптографически стойким псевдослучайным генератором*, если для любой полиномиальной вероятностной машины Тьюринга A, для любого полинома p и всех достаточно больших n

$$|P_1(A,n) - P_2(A,n)| < 1/p(n).$$

Всюду ниже мы для краткости будем называть криптографически стойкие псевдослучайные генераторы просто псевдослучайными генераторами. Такое сокращение является общепринятым в криптографической литературе.

Нетрудно убедиться, что для существования псевдослучайных генераторов необходимо существование односторонних функций. В самом деле, сама функция g должна быть односторонней. Доказательство этого простого факта мы оставляем читателю в качестве упражнения. Вопрос о том, является ли существование односторонних функций одновременно и достаточным условием, долгое время оставался открытым. В 1982 г. Яо [10] построил псевдослучайный генератор, исходя из предположения о существовании *односторонних перестановок*, т. е. сохраняющих длину взаимнооднозначных односторонних функций. За этим последовала серия работ, в которых достаточное условие все более и более ослаблялось, пока наконец в 1989–1990 гг. Импальяццо, Левин и Луби [8] и Хостад [6] не получили следующий окончательный результат.

Теорема 1. *Псевдослучайные генераторы существуют тогда и только тогда, когда существуют односторонние функции.*

Псевдослучайные генераторы находят применение не только в криптографии, но и в теории сложности, и в других областях дискретной математики. Обсуждение этих приложений выходит за рамки настоящей главы. Здесь же в качестве иллюстрации мы рассмотрим описанную в начале данного раздела криптосистему Cr, использующую псевдослучайный генератор в качестве алгоритма g. Прежде всего, нам необходимо дать определение стойкости криптосистемы с секретным ключом.

Пусть E_K — алгоритм шифрования криптосистемы с секретным ключом. Обозначим результат его работы $d = E_K(m)$, здесь K — секретный ключ длиной n битов, а m — открытый текст длиной $q(n)$

битов. Через m_i обозначается i-ый бит открытого текста. Пусть A — полиномиальная вероятностная машина Тьюринга, которая получает на вход криптограмму d и выдает пару (i, σ), где $i \in \{1, \ldots, q(n)\}$, $\sigma \in \{0, 1\}$. Интуитивно, криптосистема является стойкой, если никакая машина Тьюринга A не может вычислить ни один бит открытого текста с вероятностью успеха, существенно большей, чем при простом угадывании.

Определение 3. Криптосистема называется стойкой, если для любой полиномиальной вероятностной машины Тьюринга A, для любого полинома p и всех достаточно больших n

$$Pr\{A(d) = (i, \sigma) \ \& \ \sigma = m_i \mid K \in_R \{0,1\}^n,\ m \in_R \{0,1\}^{q(n)}\} < \frac{1}{2} + \frac{1}{p(n)}.$$

Эта вероятность (всюду ниже для краткости мы ее обозначаем просто Pr) определяется случайным выбором секретного ключа K, случайным выбором открытого текста m из множества всех двоичных строк длины $q(n)$ и случайными величинами, которые A использует в своей работе.

Покажем, что криптосистема Cr с псевдослучайным генератором в качестве g является стойкой в смысле данного определения. Предположим противное, т. е. что существуют полиномиальный вероятностный алгоритм A и полином p такие, что $Pr \geqslant 1/2 + 1/p(n)$ для бесконечно многих n. Рассмотрим алгоритм B, который получает на входе двоичную строку r длины $q(n)$, выбирает $m \in_R \{0,1\}^{q(n)}$, вычисляет $d = m \oplus r$ и вызывает A как подпрограмму, подавая ей на вход строку d. Получив от A пару (i, σ), B проверяет, действительно ли $m_i = \sigma$ и если да, то выдает 1, в противном случае — 0, и останавливается. Легко видеть, что B работает за полиномиальное (от n) время. Убедимся, что алгоритм B отличает псевдослучайные строки, порожденные генератором g, от случайных строк длины $q(n)$. В самом деле, если строки r, поступающие на вход B, являются случайными, то d — криптограмма шифра Вернама и, согласно теореме Шеннона, $Pr = 1/2$. Если строки r порождены генератором g, то криптограммы d имеют такое же распределение вероятностей, как в криптосистеме Cr, и, согласно предположению, $Pr \geqslant 1/2 + 1/p(n)$ для бесконечно многих n. Полученное противоречие с определением псевдослучайного генератора доказывает утверждение о стойкости криптосистемы Cr.

Разумеется, стойкость криптосистемы с секретным ключом можно определять различным образом. Например, можно рассматривать стойкость против атаки с выбором открытого текста: противник может предварительно выбрать полиномиальное количество открытых текстов и получить их криптограммы, после чего он получает ту криптограмму, по которой ему требуется вычислить хотя бы

один бит соответствующего открытого текста. Нетрудно убедиться, что криптосистема Cr с псевдослучайным генератором в качестве g является стойкой и против атаки с выбором открытого текста.

Таким образом, мы убедились, что с помощью псевдослучайных генераторов можно строить стойкие криптосистемы. Основное направление исследований в данной области — поиск методов построения эффективных псевдослучайных генераторов на основе различных криптографических предположений. Показателем эффективности здесь служит количество операций, затрачиваемых на вычисление каждого очередного бита псевдослучайной последовательности.

5. Доказательства с нулевым разглашением

Предположим, что Алиса знает доказательство некоторой теоремы и желает убедить Боба в том, что теорема верна. Конечно, Алиса может просто передать доказательство Бобу на проверку. Но тогда впоследствии Боб сможет сам, без помощи Алисы, доказывать третьим лицам эту теорему. А может ли Алиса убедить Боба так, чтобы он не получил при этом никакой информации, которая помогла бы ему восстановить доказательство теоремы? Этим двум, казалось бы взаимно исключающим требованиям, удовлетворяют протоколы доказательства с нулевым разглашением. Последнее понятие было введено Гольдвассер, Микали и Ракоффом в 1985 г. [4].

Рассматривается следующая модель протокола. В распоряжении Алисы и Боба имеются вероятностные машины Тьюринга **P** и **V** соответственно. Вычислительные ресурсы, которые может использовать Алиса, неограничены, в то время как машина **V** работает за полиномиальное время. Машины **P** и **V** имеют общую коммуникационную ленту для обмена сообщениями. После записи сообщения на коммуникационную ленту машина переходит в состояние ожидания и выходит из него, как только на ленту будет записано ответное сообщение. Машины **P** и **V** имеют также общую входную ленту, на которую записано входное слово x. Утверждение, которое доказывает Алиса, суть «$x \in L$», где L — некоторый фиксированный (известный и Алисе, и Бобу) язык. Чтобы избежать тривиальности, язык L должен быть трудным (например, NP-полным), иначе Боб сможет самостоятельно проверить, что $x \in L$. По существу, протокол доказательства состоит в том, что Боб, используя случайность, выбирает некоторые вопросы, задает их Алисе и проверяет правильность ответов. Выполнение протокола завершается, когда машина **V** останавливается, при этом она выдает 1, если доказательство принято, и 0 — в противном случае.

Пусть A и B — две интерактивные, т. е. взаимодействующие через общую коммуникационную ленту, вероятностные машины Тьюринга. Через $[B(x), A(x)]$ обозначается случайная величина — выходное слово машины A, когда A и B работают на входном слове x. Через $|x|$ обозначается длина слова x.

Определение 4. *Интерактивным доказательством для языка* L называется пара интерактивных машин Тьюринга $(\mathbf{P}, \mathbf{V})$ такая, что выполняются следующие два условия.

1. (Полнота). Для всех $x \in L$

$$Pr\{[\mathbf{P}(x), \mathbf{V}(x)] = 1\} = 1.$$

2. (Корректность). Для любой машины Тьюринга $\mathbf{P}^*$, для любого полинома p и для всех $x \notin L$ достаточно большой длины

$$Pr\{[\mathbf{P}^*(x), \mathbf{V}(x)] = 1\} < 1/p(|x|).$$

Полнота означает, что если входное слово принадлежит языку L и оба участника, и Алиса, и Боб, следуют протоколу, то доказательство будет всегда принято. Требование корректности защищает Боба от нечестной Алисы, которая пытается обмануть его, «доказывая» ложное утверждение. При этом Алиса может каким угодно образом отклоняться от действий, предписанных протоколом, т. е. вместо машины Тьюринга $\mathbf{P}$ использовать любую другую машину $\mathbf{P}^*$. Требуется, чтобы вероятность обмана была в любом случае пренебрежимо малой.

Определение 5. Интерактивный протокол доказательства для языка L называется *доказательством с абсолютно нулевым разглашением*, если, кроме условий 1 и 2, выполнено еще и следующее условие.

3. (Свойство нулевого разглашения). Для любой полиномиальной вероятностной машины Тьюринга $\mathbf{V}^*$ существует вероятностная машина Тьюринга $\mathbf{M}_{\mathbf{V}^*}$, работающая за полиномиальное в среднем время, и такая, что для всех $x \in L$

$$\mathbf{M}_{\mathbf{V}^*}(x) = [\mathbf{P}(x), \mathbf{V}^*(x)].$$

Машина $\mathbf{M}_{\mathbf{V}^*}$ называется моделирующей машиной для $\mathbf{V}^*$. Предполагается, что математическое ожидание времени ее работы ограничено полиномом от длины x. Это означает, что в принципе $\mathbf{M}_{\mathbf{V}^*}$ может, в зависимости от того, какие значения примут используемые в ее работе случайные переменные, работать достаточно долго. Но вероятность того, что время ее работы превысит некоторую полиномиальную границу, мала. Для каждой машины $\mathbf{V}^*$ строится своя моделирующая машина; последняя может использовать $\mathbf{V}^*$ как подпрограмму. Через $\mathbf{M}_{\mathbf{V}^*}(x)$ обозначается случайная величина — выходное слово машины $\mathbf{M}_{\mathbf{V}^*}$, когда на входе она получает слово x.

Свойство нулевого разглашения защищает Алису от нечестного Боба, который, произвольно отклоняясь от действий, предписанных протоколом (используя $\mathbf{V}^*$ вместо $\mathbf{V}$), пытается извлечь из его выполнения дополнительную информацию. Условие 3 означает, что Боб может при этом получить только такую информацию, которую он смог бы вычислить и самостоятельно (без выполнения протокола) за полиномиальное время.

Приведем в качестве примера протокол доказательства с абсолютно нулевым разглашением для языка ИЗОМОРФИЗМ ГРАФОВ из работы Гольдрайха, Микали и Вигдерсона [5]. Входным словом является пара графов $G_1 = (U, E_1)$ и $G_0 = (U, E_0)$. Здесь U — множество вершин, которое можно отождествить с множеством натуральных чисел $\{1, \ldots, n\}$, E_1 и E_0 — множества ребер такие, что $|E_1| = |E_0| = m$. Графы G_1 и G_0 называются изоморфными, если существует перестановка φ на множестве U такая, что $(u, v) \in E_0$ тогда и только тогда, когда $(\varphi(u), \varphi(v)) \in E_1$ (обозначается $G_1 = \varphi G_0$). Задача распознавания изоморфизма графов — хорошо известная математическая задача, для которой на данный момент не известно полиномиальных алгоритмов. С другой стороны, неизвестно, является ли эта задача NP-полной, хотя есть веские основания предполагать, что не является.

Протокол IG

Пусть φ — изоморфизм между G_1 и G_0. Следующие четыре шага выполняются в цикле m раз, каждый раз с независимыми случайными величинами.

1. $\mathbf{P}$ выбирает случайную перестановку π на множестве U, вычисляет $H = \pi G_1$ и посылает этот граф $\mathbf{V}$.
2. $\mathbf{V}$ выбирает случайный бит α и посылает его $\mathbf{P}$.
3. Если $\alpha = 1$, то $\mathbf{P}$ посылает $\mathbf{V}$ перестановку π, в противном случае — перестановку $\pi \circ \varphi$.
4. Если перестановка, полученная $\mathbf{V}$, не является изоморфизмом между G_α и H, то $\mathbf{V}$ останавливается и отвергает доказательство. В противном случае выполнение протокола продолжается.

Если проверки п.4 дали положительный результат во всех m циклах, то $\mathbf{V}$ принимает доказательство.

Заметим, что если в протоколе IG машина $\mathbf{P}$ получает изоморфизм φ в качестве дополнительного входного слова, то ей для выполнения протокола не требуются неограниченные вычислительные ресурсы. Более того, в этом случае $\mathbf{P}$ может быть полиномиальной вероятностной машиной Тьюринга.

Теорема 2 ([5]). *Протокол IG является доказательством с абсолютно нулевым разглашением для языка ИЗОМОРФИЗМ ГРАФОВ.*

Полнота протокола IG очевидна.

Для доказательства корректности достаточно заметить, что бит α, который $\mathbf{V}$ выбирает на шаге 2, указывает $\mathbf{P}$, для какого из графов — G_0 или G_1 — требуется продемонстрировать изоморфизм с графом H. Если G_0 и G_1 не изоморфны, то H может быть изоморфен, в лучшем случае, одному из них. Поэтому проверка п. 4 даст положительный результат с вероятностью $\leqslant 1/2$ в одном цикле и с вероятностью $\leqslant 1/2^m$ во всех m циклах.

Доказательство свойства нулевого разглашения значительно сложнее. Поэтому мы воспроизводим только основную идею. Прежде всего, заметим, что основная задача машины $\mathbf{V}^*$ — получить максимально возможную информацию об изоморфизме между G_0 и G_1. Естественно предположить, что она, в отличие от $\mathbf{V}$, будет выдавать в качестве выходного слова не один бит, а всю полученную в результате выполнения протокола информацию, включая содержимое своей случайной ленты, графы H и перестановки, полученные соответственно на шагах 1 и 3 протокола IG. Моделирующая машина $\mathbf{M}_{\mathbf{V}^*}$ должна уметь строить такие же случайные строки, графы и перестановки, не зная при этом изоморфизм φ! Поэтому $\mathbf{M}_{\mathbf{V}^*}$ пытается угадать тот бит α, который будет запросом машины $\mathbf{V}^*$ на шаге 2. Для этого $\mathbf{M}_{\mathbf{V}^*}$ выбирает случайный бит β, случайную перестановку ψ и вычисляет $H = \psi G_\beta$. Далее $\mathbf{M}_{\mathbf{V}^*}$ запоминает состояние машины $\mathbf{V}^*$ (включая содержимое случайной ленты) и вызывает ее как подпрограмму, подавая ей на вход граф H. Ответом машины $\mathbf{V}^*$ будет некоторый бит α. Если $\alpha = \beta$, то моделирование в данном цикле завершено успешно, поскольку $\mathbf{M}_{\mathbf{V}^*}$ может продемонстрировать требуемый изоморфизм. Если же $\alpha \neq \beta$, то $\mathbf{M}_{\mathbf{V}^*}$ восстанавливает ранее сохраненное состояние машины $\mathbf{V}^*$ и повторяет попытку.

Если в определении свойства нулевого разглашения заменить равенство случайных величин $\mathbf{M}_{\mathbf{V}^*}(x)$ и $[\mathbf{P}(x), \mathbf{V}^*(x)]$ требованием, чтобы их распределения вероятностей «почти не отличались», то получится другая разновидность доказательств — *доказательства со статистически нулевым разглашением.*

Еще один тип — *доказательства с вычислительно нулевым разглашением.* В этом случае требуется, чтобы моделирующая машина создавала распределение вероятностей, которое неотличимо от $[\mathbf{P}(x), \mathbf{V}^*(x)]$ никаким полиномиальным вероятностным алгоритмом (неотличимость здесь определяется аналогично тому, как это делалось в определении псевдослучайного генератора).

Подчеркнем особо, что во всех трех определениях нулевого разглашения условия накладываются на действия моделирующей машины только на тех словах, которые принадлежат языку.

Помимо интереса к доказательствам с нулевым разглашением как к нетривиальному математическому объекту, они исследуются также и в связи с практическими приложениями. Наиболее естественный и важный тип таких приложений — протоколы аутентификации (см. главу 3). С помощью такого протокола Алиса может доказать Бобу свою аутентичность.

Предположим, например, что Алиса — это интеллектуальная банковская карточка, в которой реализован алгоритм **P**, а Боб — это компьютер банка, выполняющий программу **V**. Прежде чем начать выполнение каких-либо банковских операций, банк должен убедиться в подлинности карточки и идентифицировать ее владельца, или, говоря на языке криптографии, карточка должна пройти аутентификацию. В принципе для этой цели можно использовать приведенный выше протокол IG. В этом случае в памяти банковского компьютера хранится пара графов (G_0, G_1), сопоставленная Алисе, а на интеллектуальной карточке — та же пара графов и изоморфизм φ. Предполагается, что, кроме Алисы, этот изоморфизм никто не знает (кроме, быть может, Боба) и поэтому с помощью протокола IG карточка доказывает свою аутентичность. При этом свойство полноты означает, что карточка наверняка докажет свою аутентичность. Свойство корректности защищает интересы банка от злоумышленника, который, не являясь клиентом банка, пытается пройти аутентификацию, используя фальшивую карточку. Свойство нулевого разглашения защищает клиента от злоумышленника, который, подслушав одно или более выполнений протокола аутентификации данной карточки, пытается пройти аутентификацию под именем Алисы. Конечно, в данном случае бессмысленно доказывать, что пара графов (G_0, G_1) принадлежит языку ИЗОМОРФИЗМ ГРАФОВ, поскольку она заведомо выбирается из этого языка. Вместо этого Алиса доказывает, что она знает изоморфизм φ. Интерактивные доказательства такого типа называются *доказательствами знания*.

Для практического применения очень важным свойством протокола IG, как и других протоколов доказательства знания, является то, что алгоритм **P**, получивший в качестве дополнительного входа изоморфизм φ, работает за полиномиальное время. Вместо протокола IG можно использовать, вообще говоря, любое другое доказательство с нулевым разглашением, в котором алгоритм **P** обладает этим свойством. Но для реальных приложений протокол IG, как и большинство подобных протоколов, не эффективен: большое количество циклов, слишком длинные сообщения и т. д. Поиск более эффективных доказуемо стойких протоколов — одно из основных направлений исследований в данной области.

Литература к главе 2

[1] *Анохин М. И., Варновский Н. П., Сидельников В. М., Ященко В. В.* Криптография в банковском деле. М.: МИФИ, 1997.

[2] *Гэри М., Джонсон Д.* Вычислительные машины и трудно решаемые задачи. М.: Мир, 1982.

[3] *Blum M., Micali S.* How to generate crytographically strong sequences of pseudo-random bits // SIAM J. Comput. **V. 13**, No 4, 1984. P. 850–864.

[4] *Goldwasser S., Micali S., Rackoff C.* The knowledge complexity of interactive proof systems // SIAM J. Comput. **V. 18**, No 1, 1989. P. 186–208.

[5] *Goldreich O., Micali S., Wigderson A.* Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems // J. ACM. **V. 38**, No 3, 1991. P. 691–729.

[6] *Håstad J.* Pseudo-random generators under uniform assumptions // Proc. 22nd Annu. ACM Symp. on Theory of Computing. 1990. P. 395–404.

[7] *Impagliazzo R., Luby M.* One-way functions are essential for complexity based cryptography // Proc. 30th Annu. Symp. on Found. of Comput. Sci. 1989. P. 230–235.

[8] *Impagliazzo R. , Levin L., Luby M.* Pseudo-random generation from one-way functions // Proc. 21st Annu. ACM Symp. on Theory of Computing. 1989. P. 12–24.

[9] *Impagliazzo R., Rudich S.* Limits on the provable consequences of one-way permutations // Proc. 21st Annu. ACM Symp. on Theory of Computing. 1989. P. 44–61.

[10] *Yao A.C.* Theory and applications of trapdoor functions // Proc. 23rd Annu. Symp. on Found. of Comput. Sci. 1982. P. 80–91.

Глава 3

Криптографические протоколы

1. Введение

Математическая криптография возникла как наука о шифровании информации, т. е. как наука о криптосистемах. В классической шенноновской модели системы секретной связи имеются два полностью доверяющих друг другу участника, которым необходимо передавать между собой информацию, не предназначенную для третьих лиц. Такая информация называется конфиденциальной или секретной. Возникает задача обеспечения конфиденциальности, т. е. защиты секретной информации от внешнего противника. Эта задача, по крайней мере исторически, — первая задача криптографии. Она традиционно решается с помощью криптосистем.

Представим теперь себе следующую ситуацию. Имеются два абонента A и B сети связи, скажем, компьютерной сети. B — это банк, в котором у A имеется счет и A хочет переслать B по сети в электронной форме платежное поручение перевести, например, 10 фантиков со своего счета на счет другого клиента C. Нужна ли в данном случае криптографическая защита? Предлагаем читателю самостоятельно поразмышлять и убедиться, что такая защита и в самом деле необходима. Но здесь следует отметить следующий очень важный момент: у A и B нет никакой конфиденциальной информации. В самом деле, клиенты пересылают банку в качестве сообщений платежные поручения, содержание которых стандартно и общеизвестно. Для банка важно убедиться в том, что данное сообщение действительно исходит от A, а последнему, в свою очередь, необходимо, чтобы никто не мог изменить сумму, указанную в платежном поручении, или просто послать поддельное поручение от его имени. Иными словами, требуется гарантия поступления сообщений из достоверного источника и в неискаженном виде. Такая гарантия называется обеспечением целостности информации и составляет вторую задачу криптографии.

Нетрудно видеть, что при пересылке платежных поручений в электронной форме возникает еще и совершенно иной тип угроз безопасности клиентов: всякий, кто перехватит сообщение от A к B, узнает, что C получил от A 10 фантиков. А что будет, если эта информация попадет в руки мафии? Возможно, кто-то из читателей скажет, что здесь как раз и требуется конфиденциальность. И будет неправ! На самом деле клиентам необходимо нечто, аналогичное свойству анонимности обычных бумажных денег. Хотя каждая бумажная купюра имеет уникальный номер, определить, кто ее использовал и в каких платежах, практически невозможно. Аналог этого свойства в криптографии называется неотслеживаемостью. Обеспечение неотслеживаемости — третья задача криптографии.

Если задача обеспечения конфиденциальности решается с помощью криптосистем, то для обеспечения целостности и неотслеживаемости разрабатываются криптографические протоколы. Имеются и другие отличия криптографических протоколов от криптосистем, из которых можно выделить следующие:

– протоколы могут быть интерактивными, т. е. подразумевать многораундовый обмен сообщениями между участниками;

– в протоколе может быть более двух участников;

– участники протокола, вообще говоря, не доверяют друг другу. Поэтому криптографические протоколы должны защищать их участников не только от внешнего противника, но и от нечестных действий партнеров.

К сожалению, понятие криптографического протокола, по-видимому, невозможно формализовать. То же относится и к задачам обеспечения целостности и неотслеживаемости. Под протоколом (не обязательно криптографическим) обычно понимают распределенный алгоритм, т. е. совокупность алгоритмов для каждого из участников, плюс спецификации форматов сообщений, пересылаемых между участниками, плюс спецификации синхронизации действий участников, плюс описание действий при возникновении сбоев. На последний элемент этого списка следует обратить особое внимание, поскольку его часто упускают из виду, а некорректный повторный пуск может полностью разрушить безопасность участников даже в стойком криптографическом протоколе.

Криптографические протоколы — сравнительно молодая отрасль математической криптографии. Первые протоколы появились около 20 лет назад. С тех пор эта отрасль бурно развивалась, и на настоящий момент имеется уже не менее двух десятков различных типов криптографических протоколов. Все эти типы можно условно разделить на две группы: прикладные протоколы и примитивные. Прикладной протокол решает конкретную задачу, которая возникает (или может воз-

никнуть) на практике. Примитивные же протоколы используются как своеобразные «строительные блоки» при разработке прикладных протоколов.

За последнее десятилетие криптографические протоколы превратились в основной объект исследований в теоретической криптографии. Например, на крупнейших ежегодных международных криптографических конференциях Crypto и EUROCRYPT бóльшая часть докладов посвящена именно протоколам. Безусловно, такая ситуация может быть всего лишь отражением преобладающих интересов исследователей. Но для вывода о превращении криптографических протоколов в основной объект криптографических исследований имеются и объективные основания. Как можно понять из приведенного выше примера, в банковских платежных системах в наши дни вместо платежных поручений на бумаге используется их электронная форма. Выгоды от такой замены настолько ощутимы, что, по-видимому, банки от нее уже никогда не откажутся, какие бы технические и криптографические (связанные с обеспечением целостности) трудности при этом не возникали. Но платежные поручения — лишь один из многочисленных типов документов, находящихся в обороте в сфере бизнеса. А ведь существуют еще документы, с которыми работают государственные органы и общественные организации, юридические документы и т. д. В последние годы в развитых странах отчетливо прослеживается тенденция перевода всего документооборота в электронную форму. Обсуждение всех выгод и последствий такого шага выходит за рамки тематики данной главы. Для нас важно отметить, что поскольку переход на электронные документы представляется неизбежным, возникает необходимость обеспечения, в каждом конкретном случае, целостности и неотслеживаемости, т. е. разработки соответствующих криптографических протоколов.

Сказанное выше опровергает следующее расхожее представление: поскольку-де криптографы научились конструировать криптосистемы, которые в течение длительного времени выдерживают все атаки, математические исследования в криптографии в наши дни представляют в основном лишь академический интерес. Не касаясь проблем, связанных с криптосистемами, укажем лишь, что в исследованиях многих типов криптографических протоколов сделаны только первые шаги и еще многие математические проблемы предстоит решить, прежде чем криптографические протоколы войдут в повсеместное использование.

Цель данной главы — познакомить читателя с некоторыми типами криптографических протоколов и обрисовать круг математических задач, возникающих при исследовании их стойкости. При этом предполагается знакомство читателя с главами 1, 2 и 4.

2. Целостность. Протоколы аутентификации и электронной подписи

"Воротится коза, постучится в дверь и запоет:

— Козлятушки, ребятушки!

Отопритеся, отворитеся!

Ваша мать пришла — молока принесла.

Козлятки отопрут дверь и впустят мать ...

Волк подслушал как поет коза. Вот раз коза ушла, волк подбежал к избушке и закричал толстым голосом:

— Вы детушки! Вы козлятушки!

Отопритеся, отворитеся,

Ваша мать пришла, молока принесла.

Козлята ему отвечают:

— Слышим, слышим — да не матушкин это голосок! ...

Волку делать нечего. Пошел он в кузницу и велел себе горло перековать, чтобы петь тонюсеньким голосом ...

Только коза ушла, волк опять шасть к избушке, постучался и начал причитывать тонюсеньким голосом:

— Козлятушки, ребятушки!

Отопритеся, отворитеся!

Ваша мать пришла — молока принесла.

Козлята отворили дверь, волк кинулся в избу и всех козлят съел."

«Волк и семеро козлят». Русская народная сказка.

Как уже отмечалось во введении, понятие целостности информации, по-видимому, не допускает математической формализации. В данном разделе мы рассмотрим методы обеспечения целостности на примере двух наиболее важных и распространенных типов криптографических протоколов — схем аутентификации и электронной подписи.

Назначение и суть протоколов аутентификации (называемых также протоколами идентификации) легко понять на следующем примере. Представим себе информационную систему, которая работает в компьютерной сети и обеспечивает доступ к некоторым данным. У администратора системы имеется список всех ее пользователей вместе с сопоставленным каждому из них набором полномочий, на основе которых осуществляется разграничение доступа к ресурсам системы. Ресурсами могут быть, например, некоторые фрагменты информации, а также функции, выполняемые системой. Одним пользователям может быть разрешено читать одну часть информации, другим — другую ее часть, а третьим — еще и вносить в нее изменения. В данном контексте под обеспечением целостности понимается предотвращение доступа к

системе лиц, не являющихся ее пользователями, а также предотвращение доступа пользователей к тем ресурсам, на которые у них нет полномочий. Наиболее распространенный метод разграничения доступа, парольная защита, имеет массу недостатков. Их обсуждение стало общим местом для текстов по компьютерной безопасности, поэтому мы сразу перейдем к криптографической постановке задачи.

В протоколе имеются два участника — Алиса, которая должна доказать свою аутентичность, и Боб, который эту аутентичность должен проверить. У Алисы имеются два ключа — общедоступный открытый K_1 и секретный K_2. Фактически, Алисе нужно доказать, что она знает K_2, и сделать это таким образом, чтобы это доказательство можно было проверить, зная только K_1.

Задача аутентификации уже обсуждалась в главе 2. Там же были сформулированы основные требования, которым должен удовлетворять стойкий протокол аутентификации. Напомним, что для удовлетворения этих требований достаточно, чтобы протокол аутентификации был доказательством с нулевым разглашением. В главе 2 приведен протокол доказательства с абсолютно нулевым разглашением для задачи ИЗОМОРФИЗМ ГРАФОВ. Но этот протокол имеет неприемлемо большое с практической точки зрения количество раундов обмена сообщениями между Алисой и Бобом.

Ниже мы приводим протокол Шнорра [1], один из наиболее эффективных практических протоколов аутентификации. Для его описания нам потребуются некоторые обозначения, которые будут использоваться и в последующих разделах данной главы.

Пусть p и q — простые числа такие, что q делит $p-1$. Шнорр предлагает [1] использовать p длины порядка 512 битов и q — длины порядка 140 битов. Пусть $g \in Z_p$ таково, что $g^q = 1 \mod p$, $g \neq 1$. Пусть $x \in_R Z_q$ и $y = g^x \mod p$. Задача вычисления значения x по заданному значению y при известных p, q и g называется задачей дискретного логарифмирования (см. главу 4). Для задачи дискретного логарифмирования на данный момент не известно эффективных алгоритмов. Поэтому в криптографии широко используется гипотеза о вычислительной трудности задачи дискретного логарифмирования. Сформулируем ее более строго. Пусть n — растущий целочисленный параметр, число p выбирается из множества всех простых чисел длины n таких, что $p-1$ имеет простой делитель длины не меньше n^ε для некоторой константы $\varepsilon > 0$, q — из множества всех таких простых делителей числа $p-1$, g — из множества всех чисел g таких, что $g^q = 1 \mod p$, а $x \in Z_q$. Тогда функция $f(x,p,q,g) = (g^x \mod p, p, q, g)$ — односторонняя (см. главу 2). Рекомендации, данные Шнорром относительно длин чисел p и q, можно трактовать следующим образом. На тот момент (1989 г.) инвертирование функции f считалось практически невыполнимым уже

для p и q длины порядка 512 и 140 битов соответственно. Здесь, однако, следует учитывать, что прогресс в области вычислительной техники и в алгоритмической теории чисел (см. главу 4) может привести к необходимости пересмотра этих величин.

В качестве секретного ключа схемы аутентификации Алиса выбирает случайное число x из $\{1, \ldots, q-1\}$. Далее Алиса вычисляет $y = g^{-x} \mod p$ и публикует открытый ключ y. Открытые ключи всех участников схемы должны публиковаться таким образом, чтобы исключалась возможность их подмены (такое хранилище ключей называется общедоступным сертифицированным справочником). Эта проблема, называемая часто проблемой аутентичности открытых ключей, составляет отдельный предмет исследований в криптографии и в данной главе не рассматривается.

Схема аутентификации Шнорра

1. Алиса выбирает случайное число k из множества $\{1, \ldots, q-1\}$, вычисляет $r = g^k \mod p$ и посылает r Бобу.
2. Боб выбирает случайный запрос e из множества $\{0, \ldots, 2^t - 1\}$, где t — некоторый параметр, и посылает e Алисе.
3. Алиса вычисляет $s = k + xe \mod q$ и посылает s Бобу.
4. Боб проверяет соотношение $r = g^s y^e \mod p$ и, если оно выполняется, принимает доказательство, в противном случае — отвергает.

Отметим одно существенное отличие этого протокола от протокола доказательства для задачи ИЗОМОРФИЗМ ГРАФОВ, приведенного в главе 2: последний протокол многораундовый, в нем количество раундов, т. е. посылок сообщений от Алисы Бобу и обратно, возрастает с ростом размерности задачи (количества вершин графа). А в протоколе Шнорра количество раундов равно трем, вне зависимости от значений других параметров. Поэтому с практической точки зрения протокол Шнорра является значительно более эффективным.

Первое из требований к стойкости протоколов аутентификации, корректность, означает, что противник, знающий только открытый ключ y, может пройти аутентификацию лишь с пренебрежимо малой вероятностью. Несложный анализ показывает, что корректность протокола Шнорра зависит от выбранного значения параметра t. В самом деле, если t невелико, то противник имеет хорошие шансы просто угадать тот запрос e, который он получит от Боба на шаге 2. Пусть для простоты $t = 1$. Тогда противник, не знающий секретного ключа x, может действовать следующим образом. Подбросив монету, он выбирает равновероятным образом одно из значений 0 или 1. Обозначим его через e'. Далее противник выбирает произвольное s из $\{0, \ldots, q-1\}$,

вычисляет $r = g^s y^{e'} \mod p$ и посылает r Бобу. Ясно, что запрос e, полученный от Боба на шаге 2, совпадет с e' с вероятностью $1/2$, и именно с такой вероятностью противник пройдет аутентификацию.

Если же значение t достаточно велико, то шансы угадать запрос e малы. Шнорр [1] рекомендует $t = 72$. Разумеется, вероятность 2^{-72}, а именно такой будет вероятность простого угадывания, можно считать пренебрежимо малой. Но что будет, если противник атакует схему, используя более изощренные методы? Задача теоретической криптографии в том и состоит, чтобы исследовать стойкость криптографических схем против любых (эффективных) атак противника.

Если в схеме Шнорра Алиса является противником, то на шаге 1 вместо действий, предписанных протоколом, она может выбирать r произвольным (но эффективным) образом. Иными словами, Алиса использует некоторый полиномиальный вероятностный алгоритм, который для каждого конкретного значения r определяет вероятность его выбора.

Пусть r — некоторое значение, которое Алиса передала Бобу на шаге 1. Предположим, что нам удалось найти два запроса $e_1, e_2 \in \{0, \ldots, 2^t - 1\}$, $e_1 \neq e_2$, такие, что Алиса может для каждого из них найти соответствующие значения s, для которых проверка на шаге 4 даст положительный результат. Обозначим эти значения s через s_1 и s_2 соответственно. Мы имеем:

$$r = g^{s_1} y^{e_1} \mod p,$$
$$r = g^{s_2} y^{e_2} \mod p.$$

Отсюда

$$g^{s_1} y^{e_1} = g^{s_2} y^{e_2} \mod p,$$

или

$$g^{s_1 - s_2} = y^{e_2 - e_1} \mod p.$$

Поскольку $e_1 \neq e_2$, существует $(e_2 - e_1)^{-1} \mod q$ и, следовательно, $(s_1 - s_2)(e_2 - e_1)^{-1}$ — дискретный логарифм y, т. е. $(s_1 - s_2)(e_2 - e_1)^{-1} = x \mod q$.

Таким образом, либо запросы $e_1, e_2, e_1 \neq e_2$, такие, что Алиса может ответить надлежащим образом на оба из них (при одном и том же r) на шаге 3 протокола, встречаются «достаточно редко», и это означает, что атака Алисы успешна лишь с пренебрежимо малой вероятностью. Либо такие значения попадаются «достаточно часто», и тогда тот алгоритм, который применяет Алиса, можно использовать для вычисления дискретных логарифмов.

Эта неформально изложенная идея была использована Шнорром [1] для доказательства полиномиальной сводимости задачи дискретного логарифмирования к задаче, стоящей перед пассивным противником,

т. е. таким, который пытается пройти аутентификацию, зная лишь открытый ключ. Иными словами, доказано, что в предположении трудности задачи дискретного логарифмирования схема аутентификации Шнорра является стойкой против пассивного противника, т. е. корректной.

Отметим, что вопреки весьма распространенному мнению, этот результат, как и большинство подобных результатов в теоретической криптографии, не является асимптотическим: если задача дискретного логарифмирования трудна для чисел (p и q) данной длины, то схема Шнорра является корректной при использовании чисел той же длины.

Активный противник может провести некоторое количество сеансов выполнения протокола в качестве проверяющего с честным доказывающим (или подслушать такие выполнения) и после этого попытаться атаковать схему аутентификации. Для стойкости против активного противника достаточно, чтобы протокол аутентификации был доказательством с нулевым разглашением. Однако свойство нулевого разглашения для схемы Шнорра до сих пор никому доказать не удалось. Более того, на данный момент известен единственный метод доказательства свойства нулевого разглашения — так называемый метод «черного ящика». В этом методе моделирующая машина использует алгоритм проверяющего (машину Тьюринга V^* — в обозначениях из главы 2) лишь в качестве оракула, т. е. не анализируя сам этот алгоритм, подает ему на вход любые значения по своему выбору и получает соответствующие выходные значения. Гольдрайх и Кравчик доказали [2], что трехраундовые доказательства с нулевым разглашением, в которых последнее свойство устанавливается методом «черного ящика», существуют лишь в тривиальном случае, т. е. когда проверяющий может самостоятельно, без всякой помощи доказывающего, проверить истинность утверждаемого. В отношении схемы Шнорра из этого результата следует, что либо существует эффективный алгоритм дискретного логарифмирования, либо свойство нулевого разглашения этого протокола не может быть доказано методом «черного ящика». Вопрос о существовании доказательств с нулевым разглашением, для которых свойство нулевого разглашения не может быть доказано методом «черного ящика», остается открытым.

Нетрудно показать, что схема Шнорра обладает несколько более слабым свойством — свойством нулевого разглашения относительно честного проверяющего. В этом случае достаточно построить моделирующую машину только для честного проверяющего, который на шаге 2 и в самом деле выбирает случайный запрос e из множества $\{0, \ldots, 2^t - 1\}$.

Вся информация, которую получает Боб в результате выполнения протокола — это тройка чисел (r, e, s) такая, что $r = g^s y^e \mod p$ и $r \neq 1$. Обозначим множество всех таких троек через Ω. В этой тройке число e выбирается случайным образом из $\{0, \ldots, 2^t - 1\}$, а $r = g^k \mod p$, где k — случайное число из множества $\{1, \ldots, q-1\}$. Ясно, что при таком выборе k значение r будет распределено равновероятным образом среди всех отличных от единицы элементов группы, порожденной g. Значение s вычисляется согласно шагу 3 протокола и определяется величинами e и r однозначно. Таким образом, распределение вероятностей на множестве Ω определяется случайным выбором чисел e и k.

Моделирующая машина должна создать на множестве Ω такое же распределение вероятностей как в протоколе. Но при этом она не может использовать формулу из шага 3, поскольку в нее входит секретный ключ x. Вместо этого моделирующая машина выбирает случайные e и s из $\{0, \ldots, 2^t - 1\}$ и $\{0, \ldots, q-1\}$ соответственно и вычисляет $r = g^s y^e \mod p$. Если $r = 1$, то попытка неудачна и выбираются новые e и s. В противном случае моделирующая машина выдает тройку (r, e, s). Напомним, что моделирующая машина строится только для честного доказывающего. Из этого, в частности, следует, что открытый ключ y принадлежит группе, порожденной g, т. е. существует число $x \in \{1, \ldots, q-1\}$ такое, что $g^{-x} = y \mod p$. Поэтому при любых e и s число $r = g^s y^e = g^s g^{-xe} = g^{s-xe} \mod p$ также принадлежит этой группе. Остается лишь заметить, что поскольку моделирующая машина выбирает число s случайным образом и независимо от значения e, величина $s - xe \mod q$ — случайный элемент множества $\{0, \ldots, q-1\}$, также не зависящий от e. Поэтому число r, сгенерированное моделирующей машиной, является случайным элементом группы, порожденной g, не зависящим от e, т. е. имеет такое же распределение, как в протоколе. Иными словами, одно и то же распределение вероятностей на множестве Ω можно создать двумя способами: либо выбрать случайное k (а, следовательно, и r) и вычислить $s = k + xe \mod q$, либо выбрать случайное s и получить $r = g^{s-xe} \mod p$.

Свойства нулевого разглашения относительно честного проверяющего может оказаться достаточно, если схема аутентификации используется, например, для контроля за доступом в охраняемое помещение. В этом случае Алиса — это пропуск, выполненный в виде интеллектуальной карточки, а Боб — компьютер охраны. В такой ситуации главная задача — обеспечить корректность схемы аутентификации, а защищаться от нечестного проверяющего бессмысленно. Что же касается свойства нулевого разглашения относительно честного проверяющего, то оно представляется далеко не лишним, так как позволяет

обезопаситься от противника, который может попытаться подслушивать сеансы выполнения протокола с целью изготовления фальшивого пропуска.

Пример протокола аутентификации приведен в эпиграфе к данному разделу. В этом протоколе коза выступает в качестве доказывающего, а семеро козлят — в качестве проверяющего. Протокол призван обеспечивать целостность. В данном случае — целостность семерых козлят. В эпиграфе описана и атака на протокол. Противником выступает волк, и этот противник активный: сначала он подслушивает выполнение протокола, потом пытается сам пройти аутентификацию в качестве доказывающего (козы) и при этом накапливает и анализирует получаемую информацию. Протокол оказался нестойким против активного противника. Надо было обратиться за консультацией к профессиональным криптографам и использовать, например, схему Шнорра.

Прежде чем завершить разговор о протоколах аутентификации необходимо подчеркнуть, что никакое математически строго доказанное свойство криптографического протокола не может гарантировать его безопасности во всех случаях жизни. В самом деле, даже протокол доказательства с нулевым разглашением не защищает от следующей атаки на схему аутентификации, известной в криптографической литературе под названием «мафиозная угроза». В данном сценарии имеются четыре участника: A, B, C и D. Предположим, что A зашел в кафе выпить чашечку кофе и расплачивается с помощью кредитной карточки. Для выполнения любой банковской операции карточка должна идентифицировать себя, т. е. выполнить протокол аутентификации. Но владелец кафе, B, — мафиози, сообщник которого C в тот же самый момент находится в магазине ювелира D и пытается купить бриллиант, также с помощью кредитной карточки. При этом C «представляется» как A, а D просит доказать это с помощью протокола аутентификации. Устройство считывания для карточек у B и карточка у C — это специально изготовленные приемо-передающие устройства, которые лишь пересылают сообщения между A и D. В результате A, обмениваясь сообщениями с B, на самом деле идентифицирует себя для D.

Мы здесь не затрагиваем вопроса о практической осуществимости «мафиозной угрозы». Мы привели ее просто как пример, показывающий, что только математического обоснования стойкости того или иного криптографического протокола недостаточно. Для практического применения конкретного протокола требуются еще усилия специалистов в области информационной безопасности по анализу условий его применения.

Теперь мы переходим к обсуждению другого типа протоколов аутентификации — к протоколам электронной подписи. Они обеспечивают аутентификацию сообщений, т. е. гарантируют, что сообщения

поступают от достоверного отправителя и в неискаженном виде. Более того, целостность здесь понимается в расширенном смысле: получатель сообщения не только убеждается в его достоверности, но и получает электронную подпись, которую в дальнейшем может использовать как доказательство достоверности сообщения третьим лицам (арбитру) в том случае, если отправитель впоследствии попытается отказаться от своей подписи. Здесь имеется полная аналогия обычной подписи на бумаге, за исключением того, что под арбитром обычно понимается технический эксперт, который дает заключение о подлинности электронной подписи, т. е. выполняет функцию, аналогичную функции графолога в случае обычной подписи. Назначение схем электронной подписи мы уже обсуждали во введении на примере платежных поручений.

Все протоколы электронной подписи, разрабатываемые для практического применения, являются неинтерактивными (т. е. весь обмен сообщениями состоит в передаче отправителем получателю подписанного сообщения). Это вызвано, прежде всего, требованиями к эффективности.

Как обычно, математическая формализация рассматривает схему электронной подписи как бесконечное семейство схем, зависящих от целочисленного параметра n, $n > 0$, называемого параметром безопасности.

Схема электронной подписи — это тройка алгоритмов (G, S, V), где

G — полиномиальный вероятностный алгоритм генерации ключей. На входе 1^n алгоритм G выдает пару (K_1, K_2), где K_1 — открытый ключ схемы подписи, а K_2 — соответствующий ему секретный ключ. Ключ K_1 помещается в общедоступный сертифицированный справочник;

S — полиномиальный вероятностный алгоритм генерации подписей. На входе (m, K_2), где m — сообщение, алгоритм S выдает подпись s для сообщения m;

V — полиномиальный алгоритм проверки подписи. Если $V(K_1, m, s) = 1$, то подпись s для сообщения m принимается. В этом случае говорят, что s — корректная подпись для сообщения m. Если $V(K_1, m, s) = 0$, то подпись s отвергается. Если подпись s для сообщения m сгенерирована на ключе K_2 с помощью алгоритма S, то всегда должно быть $V(K_1, m, s) = 1$.

Здесь K_1, K_2, m и s — двоичные строки, длины которых ограничены некоторыми фиксированными полиномами от n.

Для определения стойкости схемы электронной подписи необходимо принять некоторые предположения о противнике. Последний может пытаться подделывать подписи, зная только открытый ключ схемы. Такой противник называется пассивным и является самым слабым из всех возможных противников, т. к. открытый ключ схемы подписи всегда

общедоступен. Разумеется, пассивный противник, как и любой другой, знает описание схемы подписи, поскольку предполагается, что алгоритмы G, S и V общедоступны.

Более изобретательный противник может попытаться собрать некоторую дополнительную информацию о схеме подписи, набрав некоторое количество пар (сообщение, подпись). Здесь имеются два основных варианта. Атака с известными сообщениями состоит в том, что противник перехватывает некоторое (ограниченное фиксированным полиномом от n) количество подписанных сообщений. При этом противник никак не влияет на выбор этих сообщений. По существу это — разновидность пассивного противника, так как подписанные сообщения обычно пересылаются по общедоступному каналу связи (в некоторых работах по теоретической криптографии принимается математическая модель, в которой подписанные сообщения просто публикуются).

Еще более сильный противник может сам выбирать сообщения и каким-то образом получать корректные подписи для них. Такая атака называется атакой с выбором сообщений, а соответствующий противник — активным. Вопрос о практической осуществимости подобной атаки выходит за рамки математической модели и в данной главе не рассматривается. Отметим лишь, что, по-видимому, никакими организационными мерами нельзя полностью исключить возможность атаки с выбором сообщений и здесь, как и всюду в теоретической криптографии, делается предположение в пользу противника. Это значит, что обычно исследуется стойкость схем электронной подписи в присутствии самого сильного противника, осуществляющего атаку с выбором сообщений.

Помимо атаки необходимо также определить ту угрозу безопасности схемы электронной подписи, против которой мы желаем защищаться. Самая сильная угроза — полное раскрытие, т. е. вычисление противником секретного ключа K_2. Ясно, что если противник может осуществить подобную угрозу, то схема нестойкая. С другой стороны, нетрудно понять, что стойкости против полного раскрытия недостаточно для того, чтобы считать схему подписи стойкой на практике. Достаточно рассмотреть следующую вырожденную схему. Алгоритм G выбирает случайный секретный ключ K_2 и вычисляет $K_1 = f(K_2)$, где f — некоторая односторонняя функция. Подпись для сообщения m алгоритм S вычисляет в виде $s = g(m)$, где g — некоторая легко вычислимая функция. Функция g публикуется как часть открытого ключа, и проверка подписи состоит в вычислении $g(m)$ и сравнении подписи s с этим значением. Ясно, что такая схема будет стойкой против полного раскрытия, поскольку для вычисления секретного ключа K_2 требуется инвертировать одностороннюю функцию f. Но всякий желающий может вычислить подпись для любого сообщения. Этот пример поучи-

телен еще и потому, что в криптографической литературе встречаются статьи, в которых анализ стойкости схем электронной подписи подменяется рассуждениями о сложности задачи полного раскрытия.

На другом полюсе находится самая слабая из угроз — угроза экзистенциональной подделки. Такая угроза осуществима, если после проведения атаки на схему электронной подписи противник может подделать подпись хотя бы для одного, пусть даже бессмысленного, сообщения.

Стойкость схемы электронной подписи определяется относительно пары (атака, угроза). Наиболее стойкими являются схемы, стойкие против самой слабой из угроз на основе самой сильной из атак, т. е. против экзистенциональной подделки на основе атаки с выбором сообщений. Опишем эти атаку и угрозу на несколько более строгом уровне. Систематическое изложение атак и угроз для схем электронной подписи см. в [3].

Под противником мы понимаем вероятностную машину Тьюринга A, которая получает на входе 1^n и K_1 и завершает свою работу за полиномиальное (от n) время. Машина A имеет доступ к алгоритму S генерации подписей (знающему секретный ключ K_2) как к оракулу. Тем самым A может выбрать сообщения $m_1, \ldots, m_l$, где $l \leqslant p(n)$ для некоторого полинома p, и получить для них корректные подписи $s_1, \ldots, s_l$ соответственно. При оценке времени работы машины A каждое обращение к оракулу считается одной командой. Можно считать, что к моменту выбора сообщения m_i машина A уже знает подписи $s_1, \ldots, s_{i-1}$ для всех предшествующих сообщений. Такая атака называется адаптивной. После этого машина A должна найти пару (m, s), где $m \neq m_i$ для всех $i = 1, \ldots, l$ такую, что s — корректная подпись для сообщения m. Обозначим через $A(1^n, K_1) \to (m, s)$ событие, состоящее в том, что A находит такую пару. Схема электронной подписи (G, S, V) называется стойкой против экзистенциональной подделки на основе (адаптивной) атаки с выбором сообщений, если для любой машины Тьюринга A указанного выше вида, для любого полинома P и для всех достаточно больших n

$$Pr\{A(1^n, K_1) \to (m, s)\} < 1/P(n).$$

Вероятность здесь определяется случайными величинами алгоритмов G, S и A.

Несложные рассуждения, аналогичные тем, которые приведены в главе 2, показывают, что для существования схем электронной подписи, стойких против экзистенциональной подделки на основе атаки с выбором сообщений, необходимо существование односторонних функций. Одно из замечательных достижений теоретической криптографии состоит в доказательстве того, что это условие одновременно является и

достаточным. Поскольку этот результат весьма нетривиален, ниже мы лишь схематично излагаем основные идеи.

Рассмотрим вначале схему электронной подписи Лампорта [4]. Пусть $f : \Sigma^n \to \Sigma^n$, где $\Sigma = \{0, 1\}$, — односторонняя функция и пусть Σ^r — множество, из которого выбираются подписываемые сообщения (т. е. каждое сообщение $m = m_1 \ldots m_r$ рассматривается как r-битовая строка). Алгоритм вычисления функции f считается общедоступным.

Алгоритм генерации ключей выбирает $2r$ случайных строк из Σ^n: $x_1^0, x_1^1, x_2^0, x_2^1, \ldots, x_r^0, x_r^1$. Совокупность этих $2r$ строк составляет секретный ключ схемы подписи. Далее алгоритм вычисляет

$$y_1^0 = f(x_1^0),\ y_1^1 = f(x_1^1), \ldots, y_r^0 = f(x_r^0),\ y_r^1 = f(x_r^1).$$

Совокупность строк $y_1^0, y_1^1, \ldots, y_r^0, y_r^1$ публикуется как открытый ключ схемы.

Подписью для сообщения $m = m_1 \ldots m_r$ служит последовательность строк $(x_1^{m_1}, \ldots, x_r^{m_r})$. Иными словами, для i-го бита сообщения m открывается одно из значений x_i^0 или x_i^1 в зависимости от того, чему равен i-ый бит m_i — нулю или единице. Проверка подписи $s = (s_1, \ldots, s_r)$ очевидна: для каждого $i = 1, \ldots, r$ вычисляется значение $f(s_i)$ и сравнивается с соответствующей строкой $y_i^{m_i}$ из открытого ключа.

Схема Лампорта «одноразовая», она позволяет подписать одно r-битовое сообщение, после чего надо сгенерировать новые ключи и заменить открытый ключ в сертифицированном справочнике. Но с другой стороны, эта схема уже обладает тем свойством, которого мы добиваемся: если f — односторонняя функция, то схема стойкая. В самом деле, пусть $Y = (y_1^0, y_1^1, \ldots, y_r^0, y_r^1)$ — открытый ключ, а (m, s) — сообщение и корректная подпись для него, вычисленная на секретном ключе $X = (x_1^0, x_1^1, \ldots, x_r^0, x_r^1)$, соответствующем открытому ключу Y. Предположим, что противник может за полиномиальное время с вероятностью по крайней мере $1/P(n)$ для некоторого полинома P найти пару (m', s'), где $m' \neq m$, а s' — корректная подпись для m'. Пусть i_0 — номер бита, в котором отличаются сообщения m и m', и предположим для определенности, что $m_{i_0} = 0$, а $m'_{i_0} = 1$. Тогда в подписи s отсутствует значение $x_{i_0}^1$, а в s' должно присутствовать какое-либо значение x такое, что $f(x) = y_{i_0}^1 = f(x_{i_0}^1)$. Следовательно, алгоритм A, которым пользуется противник для вычисления s', должен уметь инвертировать функцию f на этом значении y_{i_0}. Тогда следующий алгоритм будет противоречить предположению о том, что функция f односторонняя. Пусть u выбрано наудачу из Σ^n и $v = f(u)$ задано нам в качестве входного значения. Выбираем случайное j из множества $\{1, \ldots, 2r\}$. Для всех $i = 1, \ldots, 2r$, $i \neq j$, генерируем соответствующие компоненты ключей X и Y с помощью алгоритма G, а вместо j-го элемента ключа Y подставляем v. Далее вызываем алгоритм A, подавая ему на вход ключ Y. Когда A за-

просит подпись для некоторого сообщения m, проверяем, требуется ли для подписания этого сообщения прообраз значения v. Если да, то попытка неудачна (вероятность этого 1/2). В противном случае создаем подпись s для m и передаем ее алгоритму A.

Если A успешно подделывает подпись для некоторого сообщения m', $m' \neq m$, то проверяем, не содержит ли эта подпись прообраз значения v. Если да, то выдаем этот прообраз.

Ясно, что в случае успеха описанный алгоритм инвертирует функцию f. Очевидно также, что этот алгоритм работает за полиномиальное время. Оценим теперь вероятность успеха. Прежде всего заметим, что алгоритм A получает на вход только открытый ключ Y и распределение вероятностей на множестве всех возможных значений Y такое же как в том случае, когда A атакует схему подписи. Поэтому мы угадаем i_0 и m'_{i_0} (т. е. произойдет событие $j = 2 \cdot i_0 + m'_{i_0}$) с вероятностью $1/2r$. Общая вероятность успеха будет по крайней мере $\dfrac{1}{4rP(n)}$, что не меньше, чем $1/Q(n)$ для некоторого полинома Q (напомним, что по определению r не превосходит некоторого полинома от параметра безопасности n).

В качестве своеобразного криптографического курьеза укажем, что если в схеме Лампорта разрешить подписывать любые сообщения, длина которых не превосходит r, то из подписи для сообщения m можно легко создать подпись для любого его префикса. Например, из подписи для сообщения, представляющего собой знаменитую фразу «казнить нельзя, помиловать» можно изготовить подпись для сообщения «казнить». Конечно, в данном случае можно предложить простые и эффективные контрмеры. Например, можно потребовать, чтобы короткие сообщения дополнялись справа двоичными нулями до длины r. Но этот курьез служит хорошей иллюстрацией следующего общего принципа: всякое утверждение о стойкости какой-либо криптографической схемы требует точной спецификации значений всех ее параметров и зачастую даже незначительное отступление от установленных значений полностью разрушает стойкость схемы.

Теперь осталось только преобразовать схему Лампорта в «многоразовую». Идея такого преобразования достаточно очевидна: в момент подписания очередного сообщения m нужно выбрать новые секретный и открытый ключи X и Y и отправить получателю тройку (m, Y, s), где s — подпись для сообщения $m\|Y$. В результате возникает цепочка открытых ключей $Y_1, Y_2, \ldots, Y_j, \ldots$ и соответствующих им подписей $s_2, \ldots, s_j, \ldots$. Здесь Y_1 — открытый ключ, хранящийся в сертифицированном справочнике, а s_j — подпись для ключа Y_j, сгенерированная с помощью секретного ключа X_{j-1}, соответствующего открытому

ключу Y_{j-1}. Таким образом, каждый открытый ключ Y_j будет аутентифицирован посредством цепочки подписей $s_2, \ldots, s_j$ и, следовательно, может использоваться совместно с секретным ключом X_j для генерации и проверки подписей для сообщений. Проблема, однако, в том, что с помощью одной пары ключей (X, Y) можно подписать только r битов сообщения, а длина одного только нового открытого ключа равна $2rn$ битам.

Для описания способа решения этой проблемы нам потребуется новое понятие — понятие криптографической хэш-функции. Хэш-функции — весьма любопытный криптографический примитив, заслуживающий отдельного разговора. Здесь же мы лишь кратко обсудим это понятие. Говоря неформально, криптографическая хэш-функция — это легко вычислимая функция, сжимающая входные значения, для которой не существует эффективного алгоритма поиска коллизий. Коллизией для функции h называется пара значений x, y, $x \neq y$, такая, что $h(x) = h(y)$. Поскольку хэш-функция сжимает входные значения, коллизии заведомо существуют и их можно найти полным перебором. Но, по определению, никакой полиномиальный алгоритм не найдет коллизий у криптографической хэш-функции.

Следующее определение семейства односторонних хэш-функций было дано Наором и Юнгом [4]. Пусть n — целочисленный параметр, $n > 0$ и H_n — множество функций $h : \Sigma^n \to \Sigma^k$, где $k = k(n) < n$. При этом предполагается, что $n < p(k)$ для некоторого полинома p. Далее, каждая функция $h \in H_n$ имеет описание $\bar{h}$ и существует полиномиальный вероятностный алгоритм, который на входе 1^n выбирает равновероятным образом из множества всех описаний функций из H_n. Предполагается также, что существует полиномиальный алгоритм, который, получив на вход описание $\bar{h}$ функции из H_n и $x \in \Sigma^n$, выдает значение $h(x)$. Под противником, который пытается отыскать коллизии, понимается полиномиальный вероятностный алгоритм B, работающий в два этапа. Сначала он получает на вход 1^n и должен выдать какое-нибудь значение x из Σ^n. После этого B получает $\bar{h}$, выбранное наудачу из множества всех описаний функций из H_n. Семейство $\{H_n\}$ называется семейством односторонних хэш-функций, если для любого такого алгоритма B, для любого полинома P и для всех достаточно больших n

$$Pr\{B(\bar{h}) = y : y \in \Sigma^n, y \neq x \& h(x) = h(y)\} < 1/P(n).$$

Предположим, что существует семейство односторонних хэш-функций $\{H_{2nr}\}$, где $H_{2nr} = \{h : \Sigma^{2nr} \to \Sigma^k, k < n\}$. Как показали Наор и Юнг [4], с помощью такого семейства схему Лампорта можно превратить в «многоразовую» следующим образом. Подписывающий выбирает случайное описание $\bar{h}$ функции $h \in H_{2nr}$ и помещает это описание в

сертифицированный справочник вместе с открытым ключом Y_1. Подписываемые сообщения имеют длину $n-k$. Когда подписывается первое сообщение, генерируется новая пара ключей (X_2, Y_2) и вычисляется значение $h(Y_2)$. Далее, с помощью первых $n-k$ элементов ключа X_1 подписывается сообщение, а с помощью последних k элементов — значение $h(Y_2)$. После этого сообщение, новый открытый ключ Y_2 и подписи посылаются получателю. Аналогичным образом подписываются все последующие сообщения. Для проверки подписи нужно либо проверять всю цепочку, начиная с Y_1, либо помнить последний аутентифицированный подписывающим открытый ключ.

Анализ этой схемы (подробности см. в [4]) показывает, что для подделки подписей противник должен уметь либо подменять открытый ключ Y_j, не меняя значения $h(Y_j)$, а значит, находить коллизии для функции h, либо инвертировать функцию f.

Остается еще понять, почему указанная возможность находить коллизии противоречит определению семейства односторонних хэш-функций. Ведь противник заранее знает описание $\bar{h}$ и только потом получает значение, для которого требуется найти коллизию; а значит, задача противника при атаке на схему подписи кажется более легкой, чем та, которая указана в определении семейства односторонних хэш-функций. Но дело в том, что подписывающий выбирает $\bar{h}$ и все X_j равновероятным образом и независимо друг от друга. Следовательно, и все Y_j будут независимы от $\bar{h}$. Поэтому противник, атакующий хэш-функцию, может сам выбрать значение Y таким же образом, как это делает подписывающий, и после этого получить случайное описание $\bar{h}$. Легко понять, что вероятность успеха атаки при этом не изменится.

Наор и Юнг [4] построили семейство односторонних хэш-функций, исходя из предположения о существовании односторонней перестановки. Ромпель [5] усилил этот результат, доказав, что достаточно (любой) односторонней функции. Тем самым установлено, что схемы электронной подписи, стойкие против экзистенциальной подделки на основе атаки с выбором сообщений, существуют тогда и только тогда, когда существуют односторонние функции.

Этот фундаментальный результат можно интерпретировать следующим образом. Если существуют схемы подписи, стойкие в каком-либо, даже весьма слабом, смысле, то существуют и схемы, стойкие против экзистенциальной подделки на основе атаки с выбором сообщений, т. е. схемы, стойкие в самом сильном смысле. В самом деле, как уже отмечалось выше, если схему подписи можно считать в каком-либо смысле стойкой, то она должна быть стойкой против полного раскрытия, а из этого уже следует существование односторонних функций.

К сожалению, эта схема подписи слишком неэффективна, чтобы ее можно было использовать на практике. Во-первых, подписи имеют

слишком большую длину. На каждый подписываемый бит требуется n битов подписи. Во-вторых, получатель должен хранить всю цепочку открытых ключей, начиная с Y_1, и всю последовательность подписанных сообщений для предъявления их арбитру в случае возникновения споров о подлинности подписей. С помощью некоторых ухищрений можно несколько сократить длины цепочек и уменьшить размер подписи, но полностью избавиться от этих недостатков не удается. Вопрос о возможности построения практически эффективных схем электронной подписи, исходя из одного только предположения о существования односторонней функции, остается открытым.

Рассмотрим теперь пример практической схемы электронной подписи. Вернемся к схеме аутентификации Шнорра. В этом протоколе интерактивность требуется только для того, чтобы получить от проверяющего случайный запрос e. Поэтому если бы у доказывающего был надежный источник случайности, пользующийся доверием проверяющего, то протокол можно было бы сделать неинтерактивным. Фиат и Шамир [6] предложили способ преобразования протокола аутентификации в схему электронной подписи путем замены случайного запроса неким «суррогатом». А именно, пусть m — подписываемое сообщение и h — криптографическая хэш-функция. Тогда вместо обращения к проверяющему (он же — получатель сообщения) доказывающий (он же — подписывающий) вычисляет величину $h(m)$ и использует ее в качестве запроса e. Этот метод универсален, так как применим к широкому классу протоколов аутентификации. Опишем теперь получаемую в результате такого преобразования схему электронной подписи Шнорра [1]. Открытый и секретный ключи подписывающего генерируются в этой схеме таким же образом, как в схеме аутентификации Шнорра. Открытый ключ помещается в общедоступный сертифицированный справочник.

Схема электронной подписи Шнорра

1. Подписывающий выбирает случайное число $k \in \{1, \ldots, q-1\}$ и вычисляет $r = g^k \mod p$.
2. Подписывающий вычисляет $e = h(r, m)$, где m — подписываемое сообщение.
3. Подписывающий вычисляет $s = k + xe \mod q$ и посылает сообщение m с подписью (e, s) получателю.
4. Получатель вычисляет $r' = g^s y^e \mod p$ и проверяет, выполняется ли равенство $e = h(r', m)$. Если да, то подпись принимается, в противном случае — отвергается.

Предполагается, что хэш-функция h отображает пары значений (r, m) в множество $\{0, \ldots, 2^t - 1\}$.

Легко проверить, что для подписи, сгенерированной согласно протоколу, проверка п. 4 всегда будет выполнена.

Стойкость схемы Шнорра в значительной степени зависит от свойств функции h. Если противник умеет отыскивать коллизии специального вида, т. е. по заданной паре (r, m) находить другое сообщение m', $m' \neq m$, такое, что $h(r, m) = h(r, m')$, то он может осуществлять экзистенциональную подделку подписей. Для этого достаточно перехватить сообщение m и подпись (e, s) для него, а также найти коллизию указанного вида. Тогда пара (e, s) будет также подписью и для сообщения m'.

Хэш-функции являются неотъемлемой частью конструкции схем электронной подписи. Это является следствием необходимости подписывать сообщения различной длины. Конечно, длинные сообщения можно разбивать на блоки, имеющие требуемую для схемы подписи длину, и подписывать каждый блок. Но это решение неэффективно. На практике используются хэш-функции, которые преобразуют сообщения произвольной длины в хэш-значения требуемой длины. Ясно, что такая хэш-функция должна быть в каком-то смысле стойкой против попыток найти коллизии. Но поскольку практические хэш-функции конструируются для конкретных длин хэш-значений (скажем, 256 битов), формализовать это требование не удается.

В отличие от протоколов аутентификации, для практических схем электронной подписи не известно методов доказательства стойкости. Стойкие схемы подписи не могут быть доказательствами с нулевым разглашением. Это легко понять, если вспомнить, что определение нулевого разглашения требует существования моделирующей машины, которая, не зная секретного ключа, создает для всех величин, наблюдаемых проверяющим, распределение вероятностей, неотличимое от того, которое возникает при выполнении протокола. Но все, что видит проверяющий (он же — получатель) в процессе выполнения протокола, это — сообщения с подписями. Следовательно, моделирующая машина, если она существует, может подделывать подписи, так как создаваемые ею «подписи» должны быть неотличимы от подлинных, в частности, и для алгоритма проверки подписей.

Стойкость схем электронной подписи против пассивного противника, который, зная только открытый ключ, пытается подделывать подписи, может быть доказана в так называемой модели со случайным оракулом. В этой модели подписывающий и проверяющий вместо вычисления функции h обращаются к оракулу, который для каждого входного значения α выбирает случайное выходное значение β и выдает его в качестве ответа. При этом пара (α, β) запоминается и в случае повторного обращения с входным значением α оракул снова выдаст значение β. Как заметили Фиат и Шамир [6], изложенная выше идея

доказательства корректности схем аутентификации применима в данной модели для доказательства стойкости схем подписи против пассивного противника. Этот результат справедлив для широкого класса схем подписи, включающего схему Шнорра. Фактически это означает, что схемы подписи являются стойкими (против указанного выше пассивного противника), если хэш-функция ведет себя, как случайная функция. Это утверждение является по-существу единственным результатом теоретической криптографии, касающимся стойкости практических схем электронной подписи, если не считать работы [8], где то же самое доказано в предположении, что хэш-функция ведет себя, как функция дешифрования стойкой, в некотором смысле, криптосистемы.

3. Неотслеживаемость. Электронные деньги

"И он сделает то, что всем — малым и великим, богатым и нищим, свободным и рабам — положено будет начертание на правую руку их или на чело их, И что никому нельзя будет ни покупать, ни продавать, кроме того, кто имеет это начертание, или имя зверя, или число имени его.
Здесь мудрость. Кто имеет ум, тот сочти число зверя, ибо это число человеческое; число его шестьсот шестьдесят шесть."

Откровение святого Иоанна Богослова, глава 13.

Лет пятнадцать назад, а может быть и больше, жители некоторых районов Москвы обнаруживали в своих почтовых ящиках необычные послания. На листочках, вырванных из ученических тетрадей, не слишком грамотные люди старательно переписывали один и тот же текст, повествующий о том, что где-то в Брюсселе (не иначе, как в штаб-квартире НАТО) появился конфьютер (сохраняем терминологию авторов) под названием «Зверь» и с номером 666. И этот конфьютер якобы предначертал каждому смертному определенное число, без которого не то что покупать или продавать, но даже шагу ступить нельзя будет. В заключение, разумеется, предрекался скорый конец света.

Безусловно, это сочинение, как и все ему подобные, не заслуживало бы никакого внимания, если бы его авторы, сами того не ведая, не нащупали весьма серьезную угрозу, которую несет компьютеризация правам и свободам личности.

Поскольку данный раздел посвящен электронным деньгам, рассмотрим эту угрозу на следующем простом примере. Столь популярная ныне во всем мире кредитная карточка представляет собой носитель

информации, который при каждом платеже полностью идентифицирует своего владельца. И если владелец карточки использует ее для покупки билетов на транспорт, то можно отследить все его поездки, что в цивилизованном обществе без санкции прокурора недопустимо. Аналогичным образом, для каждого владельца кредитных карточек возможно собирать информацию о том, какие товары и где он покупает, какими услугами пользуется, какие культурно-зрелищные мероприятия посещает и т. д.

Дальше — больше. Организация компьютерного доступа к хранилищам информации и перевод документов в электронную форму создадут предпосылки для ведения досье, отражающих весь круг интересов каждого из граждан. Этот перечень угроз правам и свободам личности, безусловно, можно продолжить[1]. Резюмируя, можно сказать, что компьютеризация создает беспрецедентные возможности для организации тотальной слежки. Угроза эта тем более серьезная, что она до сих пор еще не осознана даже многими специалистами.

Для предотвращения подобной угрозы необходима система контроля за доступом к ресурсам, которая удовлетворяет двум, казалось бы, взаимно исключающим требованиям. Во-первых, всякий желающий должен иметь возможность обратиться к этой системе анонимно, а во-вторых, при этом все же доказать свое право на доступ к тому или иному ресурсу. Обычные бумажные деньги обеспечивают оба этих свойства. Если ресурсом, например, является некоторый товар, то наличие у покупателя достаточного количества купюр является доказательством его права на доступ к ресурсу. С другой стороны, хотя каждая бумажная купюра и имеет уникальный номер, отслеживать купюры по номерам практически невозможно. Кредитные карточки удовлетворяют только второму требованию.

Всюду ниже под электронными деньгами мы будем понимать электронные платежные средства, обеспечивающие неотслеживаемость. Понятие неотслеживаемости, также как и целостности, по-видимому, не может быть формализовано и будет поясняться на конкретных примерах протоколов.

Для работы с электронными деньгами разрабатываются специальные криптографические протоколы, называемые протоколами электронных платежей. В таком протоколе задействованы три участника, которых мы будем называть: банк, покупатель и продавец. Покупатель и продавец, каждый, имеют счет в банке, и покупатель желает заплатить продавцу за товар или услугу.

---

[1] Когда данный раздел уже был написан, вышел из печати журнал «Компьютерра» №20 от 20 мая 1998 г., в котором эти угрозы обсуждаются более подробно. Замечательно, что у одного из авторов возникли те же ассоциации со словами из Апокалипсиса.

В платежной системе используются три основные транзакции:

– *снятие со счета;*
– *платеж;*
– *депозит.*

В транзакции снятия со счета покупатель получает подписанную банком электронную банкноту на затребованную сумму. При этом счет покупателя уменьшается на эту сумму. В транзакции платежа покупатель передает банкноту продавцу и указывает сумму платежа. Продавец, в свою очередь, передает эту информацию банку, который проверяет подлинность банкноты. Если банкнота подлинная, банк проверяет, не была ли она потрачена ранее. Если нет, то банк заносит банкноту в специальный регистр, зачисляет требуемую сумму на счет продавца, уведомляет продавца об этом, и, если достоинство банкноты выше, чем сумма платежа, возвращает покупателю «сдачу» (через продавца). С помощью транзакции депозита, покупатель может положить «сдачу» на свой счет в банке.

Безопасность банка основывается на невозможности подделать его подпись для создания фальшивой банкноты, или, более общим образом, на невозможности, получив набор подлинных электронных банкнот, подделать подпись еще хотя бы для одной банкноты. Для неотслеживаемости покупателя необходимо, чтобы банк, получив банкноту в транзакции платежа, не мог установить, кому она была выдана. То же относится и к «сдаче». Это, казалось бы, парадоксальное требование удовлетворяется с помощью схемы так называемой затемненной (слепой) подписи: в транзакции снятия со счета банк подписывает не банкноту, а некоторую «абракадабру», из которой покупатель восстанавливает подписанную банкноту. Таким образом, неотслеживаемость гарантируется тем, что банк просто не знает, что именно он подписал.

Рассмотрим простейший вариант платежной системы, в которой используется затемненная подпись, соответствующая схеме подписи RSA. Последняя основана на тех же принципах, что и криптосистема RSA (см. главу 4). Подписывающий, в нашем случае — банк, выбирает два секретных простых числа p и q достаточно большой длины и публикует их произведение $N = pq$. Пусть e и d, где $ed = 1 \mod \varphi(N)$, — соответственно открытый и секретный ключи криптосистемы RSA. Генерация подписи в схеме электронной подписи RSA состоит в применении к сообщению m функции дешифрования криптосистемы RSA: $s = m^d \mod N$. Для проверки подписи нужно применить к ней функцию шифрования. Если $s^e = m \mod N$, то s — корректная подпись для сообщения m.

Итак, банк выбирает и публикует числа N и e, а также некоторую одностороннюю функцию $f : Z_N \to Z_N$, назначение которой станет

ясно из дальнейшего. Пара ключей (e, d) используется банком исключительно для создания электронных банкнот, т. е. устанавливается соглашение о том, что электронной подписи, сгенерированной на ключе d, соответствует электронная банкнота достоинством, скажем, в 1 фантик.

В транзакции снятия со счета покупатель выбирает случайное число $n \in Z_N$ и вычисляет $f(n)$. Ему нужно получить подпись банка на этой банкноте, т.е. значение $f(n)^d$. Но просто послать значение $f(n)$ банку покупатель не может, поскольку для снятия денег со счета он должен идентифицировать себя. Поэтому, если банк получает $f(n)$, он в дальнейшем всегда узнает данную банкноту и неотслеживаемость будет потеряна. Решение проблемы состоит в использовании затемненной подписи: покупатель выбирает случайное число $r \in Z_N$, $r \neq 0$, вычисляет $f(n)r^e \mod N$ и посылает это значение банку. Множитель r^e часто называют затемняющим множителем. Банк вычисляет значение $f(n)^d \cdot r \mod N$ и возвращает его покупателю. Покупатель легко «снимает» затемняющий множитель и получает подписанную банкноту $(n, f(n)^d \mod N)$.

В транзакции платежа покупатель передает продавцу электронную банкноту $(n, f(n)^d \mod N)$. В принципе, продавец может проверить подлинность любой банкноты (n, s) самостоятельно. Для этого достаточно вычислить $f(n)$ и проверить, что $f(n) = s^e \mod N$. Но дело в том, что электронные банкноты, как и любую другую информацию, представленную в электронной форме, легко копировать. Поэтому нечестный покупатель может заплатить одной и той же электронной банкнотой многократно. Для предотвращения подобного злоупотребления продавец передает банкноту на проверку банку. Банк проверяет по специальному регистру, не была ли эта банкнота потрачена ранее, и если нет, то зачисляет 1 фантик на счет продавца и уведомляет его об этом.

Безопасность банка в этой системе электронных платежей основывается на вере в стойкость схемы электронной подписи RSA. Применение функции f в этой конструкции необходимо ввиду известного свойства мультипликативности схемы RSA: если s_1 и s_2 — подписи для m_1 и m_2 соответственно, то $s_1 s_2 = m_1^d m_2^d \mod N$ — подпись для $m_1 m_2$. Поэтому, если бы в системе электронных платежей использовались банкноты вида $(n, n^d \mod N)$, то из двух подлинных банкнот всегда можно было бы изготовить третью. Неотслеживаемость клиентов в данной системе абсолютна. Все, что остается у банка от транзакции снятия со счета, — это значение $f(n)^d \cdot r \mod N$, которое благодаря затемняющему множителю r представляет собой просто случайное число из Z_N. Поэтому у банка нет никакой информации о том, какую именно банкноту он выдал данному клиент .

В этом примере банк выдает банкноты только достоинством в 1 фантик и все платежи должны быть кратны этой величине. Оказывается, можно реализовать и более гибкую систему. Рассмотрим следующую систему электронных платежей из работы Шаума [9]. Здесь уместно отметить, что все основные идеи, связанные с понятием неотслеживаемости, электронными деньгами и со схемами затемненной подписи, принадлежат этому голландскому математику.

Система из работы [9] также основана на схеме электронной подписи RSA. Допуская некоторую вольность в обозначениях будем писать $n^{1/t} \mod N$ вместо $n^d \mod N$, где $dt = 1 \mod \varphi(N)$, и называть эту величину корнем t-ой степени из n. Как и выше, $f : Z_N \to Z_N$ — некоторая односторонняя функция, которую выбирает и публикует банк.

Устанавливается соглашение, согласно которому корню степени, равной i-му нечетному простому числу, соответствует номинал в 2^{i-1} фантиков. Т.е. предъявитель пары $(n, f(n)^{1/3} \bmod N)$ является владельцем электронной банкноты достоинством в 1 фантик. Если в этой паре вместо корня кубического присутствует корень 7-ой степени, то банкнота имеет достоинство 4 фантика, а если 21-ой степени — то 5 фантиков. Иными словами, для банкноты достоинством S фантиков необходим корень степени, равной произведению всех простых чисел, соответствующих единицам в двоичном представлении числа S.

Все банкноты, выдаваемые банком, имеют одинаковое достоинство. Для простоты изложения будем, как и в [9], предполагать, что оно равно 15 фантикам. Тогда подпись банка на банкноте, это — корень h-ой степени, где $h = 3 \cdot 5 \cdot 7 \cdot 11$. Для этой схемы нужен также еще дополнительный модуль RSA N_1, который используется в работе с так называемой копилкой (см. ниже). Этот модуль выбирается и публикуется таким же образом, как и модуль N.

Транзакция снятия со счета выполняется таким же образом, как описано выше. В результате покупатель получает электронную банкноту $(n_1, f(n_1)^{1/h} \mod N)$.

Предположим теперь, что покупатель желает заплатить продавцу 5 фантиков. Для этого он вычисляет $f(n_1)^{1/3\cdot 7} \bmod N$, просто возводя полученную банкноту в 55-ую степень, и создает копилку, выбирая случайное значение j и вычисляя $f(j)s_1^{5\cdot 11} \mod N_1$. Здесь опять $s_1^{5\cdot 11}$ — затемняющий множитель. Транзакция платежа начинается с пересылки значений n_1, $f(n_1)^{1/3\cdot 7} \mod N$, $f(j)s_1^{5\cdot 11} \mod N_1$, а также суммы платежа (5 фантиков) продавцу. Продавец, в свою очередь, передает всю эту информацию банку. Банк легко проверяет, что пара $(n_1, f(n_1)^{1/3\cdot 7})$ представляет собой подлинную банкноту достоинством 5 фантиков. Он проверяет по специальному регистру, не была ли банкнота с номером n_1 потрачена ранее. Если нет, записывает в регистр вновь полученную банкноту, увеличивает счет продавца на 5 фантиков и посылает про-

давцу уведомление о завершении транзакции платежа, а также «сдачу» (10 фантиков) покупателю, возвращаемую через копилку: $f(j)^{1/5\cdot 11}s_1 \mod N_1$.

В транзакции депозита покупатель посылает банку копилку $(j, f(j)^{1/5\cdot 11})$. Банк проверяет ее таким же образом, как и банкноту в транзакции платежа, и если копилка с номером j подлинная и ранее не использовалась в транзакции депозита, то зачисляет сумму 10 фантиков на счет покупателя.

Если все платежи, осуществляемые покупателями, делаются на максимальную сумму (15 фантиков), то схема обеспечивает безусловную (или теоретико-информационную) неотслеживаемость покупателя: выдавая затемненную подпись, банк не получает никакой информации о номере подписываемой банкноты.

Необходимость депозита полученной от банка «сдачи» нарушает неотслеживаемость: банк запоминает все платежи, а значит, и все «сдачи», и при выполнении транзакции депозита может «вычислить» клиента, если последний выполнил платеж на уникальную или достаточно редко встречающуюся сумму. Эта проблема частично может быть решена за счет многократного использования копилки в транзакциях платежа.

Предположим, что покупатель получил в банке вторую банкноту с номером n_2 и желает заплатить тому же или другому продавцу сумму в 3 фантика. Тогда в транзакции платежа он может использовать копилку со «сдачей», оставшейся после первого платежа, и послать продавцу n_2, $f(n_2)^{1/3\cdot 5} \mod N$, $f(j)^{1/5\cdot 11}s_2^{7\cdot 11} \mod N_1$. Платеж выполняется таким же образом, как было описано выше, и в результате покупатель получит копилку $f(j)^{1/5\cdot 11\cdot 7\cdot 11} \mod N_1$.

В транзакции депозита покупатель кладет накопленную в копилке сумму на свой счет в банке. Для этого он посылает банку значения j, $f(j)^{1/5\cdot 7\cdot 11\cdot 11} \mod N_1$ и указывает сумму. Банк проверяет копилку так же, как банкноту, т. е. устанавливает наличие всех корней с объявленными покупателем кратностями, а также проверяет, что копилка с номером j не использовалась ранее ни в одной транзакции депозита. При выполнении всех этих условий банк зачисляет сумму, находящуюся в копилке, на счет покупателя.

Если количество клиентов в платежной системе достаточно велико и если каждый из них использует в транзакциях платежа одну и ту же копилку до тех пор, пока накапливаемая в ней сумма не превысит определенный предел (скажем, 100 фантиков), а после этого сразу же выполняет депозит, то шансы банка отследить действия кого-либо из клиентов представляются практически ничтожными.

Оба рассмотренных примера относятся к классу так называемых централизованных систем, отличительная черта которых — необходимость участия банка во всех транзакциях платежа. С точки зрения

эффективности значительно привлекательнее выглядят автономные системы электронных платежей, в которых продавец самостоятельно, без обращения к банку, проверяет подлинность полученных от покупателя электронных денег. Последние, чтобы отличить автономные системы от централизованных, будем называть электронной монетой (заметим, что общепринятой терминологии здесь нет).

Как уже отмечалось выше, без обращения к банку в каждой транзакции платежа невозможно предотвратить повторную трату одной и той же электронной монеты. Вместо этого автономные системы обеспечивают идентификацию нарушителя *post factum*. Конструкции автономных систем электронных платежей достаточно сложны (см., например, [10], [11]), поэтому здесь мы описываем лишь их основную идею, да и то в самых общих чертах. Наше описание предполагает использование схемы аутентификации Шнорра, но можно использовать и любую другую схему, обладающую всеми необходимыми свойствами.

Каждый клиент банка выбирает секретный ключ x, содержащий идентификатор этого клиента, и затем вычисляет открытый ключ $y = g^x \mod p$. В транзакции снятия со счета клиент выбирает случайное значение k и вычисляет $r = g^k \mod p$. Электронная монета состоит из некоторой строки, содержащей y и r, и подписи банка для этой строки. Основная трудность здесь состоит в том, что банк должен подписать монету затемненной подписью, но при этом убедиться в том, что монета имеет требуемую структуру. Один из способов решения этой проблемы заинтересованный читатель может найти в работе Якоби [10].

В транзакции платежа продавец сначала проверяет подпись банка, и, если она корректна, выбирает случайный запрос e, как в схеме аутентификации Шнорра, и посылает e покупателю. Последний вычисляет $s = k + ex \mod q$ и посылает s продавцу. Продавец проверяет правильность ответа, используя те значения r и y, которые содержатся в монете.

В транзакции депозита продавец посылает банку электронную монету, а также e и s. Если банк обнаруживает, что данная монета уже была потрачена ранее, то у него оказываются две различные пары (e, s) и (e', s'), удовлетворяющие проверочному соотношению в схеме Шнорра при одних и тех же y и r. Как было показано в предыдущем разделе, этого достаточно для того, чтобы банк смог вычислить секретный ключ x, а значит, и идентифицировать нарушителя.

Замечательное свойство автономных систем электронных платежей состоит в том, что они, с одной стороны, обеспечивают неотслеживаемость честных клиентов, а с другой — позволяют однозначно иденти-

фицировать нарушителей. Но в таких системах банк идет на определенный риск, поскольку в момент обнаружения повторной траты электронной монеты на счету нарушителя может не оказаться суммы, достаточной для покрытия перерасхода. Несколько более подробно эта проблема обсуждается в [12].

В большинстве автономных систем электронные монеты могут использоваться лишь в одном платеже, после чего необходимо выполнить депозит. Если монета может использоваться во многих платежах, без промежуточных депозитов, то такая монета называется переводимой. Если бы переводимые монеты могли находиться в обращении достаточно долго, то это обеспечило бы практическую неотслеживаемость клиентов. Но с другой стороны, стало бы значительно сложнее обнаруживать повторные траты одной и той же монеты. Еще один недостаток в том, что длина переводимой монеты возрастает с каждым ее переводом от клиента к клиенту. С интуитивной точки зрения это представляется естественным, поскольку монета должна содержать информацию, позволяющую банку идентифицировать нарушителя, потратившего монету дважды. Поэтому каждый клиент, через которого проходит монета, должен оставить на ней свои «отпечатки пальцев». Шаум и Педерсен [13] доказали, что возрастание длины переводимой монеты и в самом деле неизбежно.

4. Протоколы типа «подбрасывание монеты по телефону»

"— Хорошо, дайте же сюда деньги.

— На что-ж деньги? У меня вот они в руке! Как только напишете расписку, в ту же минуту их возьмете.

— Да позвольте, как же мне писать расписку? Прежде нужно видеть деньги.

Чичиков выпустил из рук бумажки Собакевичу, который, приблизившись к столу и накрывши их пальцами левой руки, другою написал на лоскутке бумаги, что задаток двадцать пять рублей государственными ассигнациями за проданные души получен сполна."

Н. В. Гоголь. «Мертвые души», глава 5.

В данном разделе мы кратко обсудим те типы криптографических протоколов, в которых два участника должны обменяться некоторой информацией. Но участники не доверяют друг другу и каждый из них может оказаться обманщиком. Поэтому, если один из участников по

неосторожности «выпустит информацию из рук» преждевременно, то в обмен он может получить совсем не то, о чем договаривались, или вообще не получить ничего: проблемы здесь те же, что и в «протоколе» обмена расписки на ассигнации у Чичикова и Собакевича.

Из всех криптографических протоколов данного типа, пожалуй, наиболее наглядным, и к тому же достаточно простым, является протокол подбрасывания монеты. Предположим, что двум участникам, Алисе и Бобу, необходимо бросить жребий. В случае, когда они оба физически находятся в одном и том же месте, задачу можно решить с помощью обычной процедуры подбрасывания монеты. Если кто-либо из участников не доверяет монете, можно использовать другие источники случайности. Правда, создание надежных источников случайности — весьма непростая задача, но она уже относится к математической статистике, а не к криптографии.

Если же Алиса и Боб удалены друг от друга и могут общаться лишь по каналу связи, то задача о жребии, на первый взгляд, кажется неразрешимой. В самом деле, если, следуя обычной процедуре подбрасывания монеты, первый ход делает Алиса, которая выбирает один из возможных вариантов — «орел» или «решка», то Боб всегда может объявить тот исход, который ему выгоден.

Тем не менее, эта задача была решена Блюмом [14]. Любопытно, что даже в заголовке своей работы Блюм охарактеризовал предложенный им метод как метод «решения нерешаемых задач».

Легко понять, что задача о жребии решается очень просто, если существует надежный агент — третья сторона, которая пользуется полным доверием и Алисы, и Боба, и которая имеет конфиденциальные (закрытые) каналы связи с обоими участниками. В этом случае Боб и Алиса выбирают случайные биты b и c соответственно и посылают их в тайне друг от друга агенту. Последний ждет, пока не поступят оба бита, и после этого публикует b, c и $d = b \oplus c$ — исход подбрасывания монеты.

В отсутствие надежного агента срабатывает идея, которую проще всего понять на следующей «физической» реализации. Боб выбирает случайный бит b, записывает его на листе бумаги, запирает этот лист в ящике, оставляя ключ от замка у себя, и посылает ящик Алисе. Предполагается, что, не имея ключа, Алиса не может добраться до содержимого ящика. Получив ящик, Алиса выбирает случайный бит c и посылает его Бобу. В ответ Боб посылает Алисе ключ от ящика. Исходом подбрасывания монеты будет опять-таки бит $d = b \oplus c$.

Ниже мы излагаем криптографическую реализацию той же идеи, основанную на задаче дискретного логарифмирования, и используем при этом обозначения из раздела 2.

Протокол подбрасывания монеты

1. Алиса выбирает случайное число x из Z_q, вычисляет $y = g^x \mod p$ и посылает y Бобу.

2. Боб выбирает случайный бит b, случайное число k из Z_q, вычисляет $r = y^b g^k \mod p$ и посылает r Алисе.

3. Алиса выбирает случайный бит c и посылает его Бобу.

4. Боб посылает Алисе b и k.

5. Алиса проверяет, выполняется ли сравнение $r = y^b g^k \mod p$. Если да, то результатом выполнения протокола будет бит $d = b \oplus c$.

Значение r — это криптографический аналог того ящика, о котором шла речь в описании физической реализации. В самом деле, из значения r Алиса не может извлечь никакой информации о бите b. Поскольку k выбирается случайным образом из Z_q, значение r в обоих случаях, при $b = 0$ и $b = 1$, является случайным элементом группы, порожденной g, и поэтому не несет никакой информации о значении бита b (разумеется, Алиса может попытаться обманывать, выбирая значение y, не принадлежащее группе, порожденной g; однако Боб легко обнаружит такой обман, проверяя сравнение $y^q = 1 \mod p$).

С другой стороны, Боб может обманывать, т. е. открывать значение бита b по своему желанию и как 0, и как 1, но только в том случае, если он умеет вычислять дискретные логарифмы. Это вытекает из следующих соображений. Поскольку, как уже отмечалось выше, можно считать, что значение r заведомо принадлежит группе, порожденной g, существует единственное число $\alpha \in Z_q$ такое, что $r = g^\alpha \mod p$. Для того, чтобы открыть значение $b = 0$, Боб должен послать Алисе на шаге 4 число α, а для того, чтобы открыть значение $b = 1$, — число $k = \alpha - x \mod q$. Отсюда $x = \alpha - k \mod q$. Пусть M — полиномиальная вероятностная машина Тьюринга, которую Боб использует для осуществления такого обмана. Тогда следующий алгоритм будет вычислять дискретные логарифмы.

1. Передаем входное значение y машине M.

2. Получив в ответ значение r, запоминаем состояние машины M.

3. Выбираем случайный бит c и передаем его M.

4. Получив от M значения b и k, запоминаем эти величины, восстанавливаем ранее запомненное состояние машины M и переходим на шаг 3.

5. Как только среди пар (b, k) найдутся $(0, k_1)$ и $(1, k_2)$, вычисляем $x = k_1 - k_2 \mod q$ — дискретный логарифм величины y.

На основе этих соображений можно построить доказательство следующего утверждения: в предположении трудности задачи дискретного логарифмирования приведенный выше протокол подбрасывания монеты является стойким. Заметим, что для данного протокола достаточно весьма слабой формы этого предположения. Поскольку Алиса может прекратить выполнение протокола, если с момента передачи значения y Бобу до момента получения от него значений b и k проходит более, скажем, 30 секунд, достаточно предположить, что задача дискретного логарифмирования не может быть решена за такое время.

Если из приведенного выше протокола подбрасывания монеты вычленить шаги 1, 2 и 4, то получим так называемый протокол привязки к биту (bit commitment). Шаги 1 и 2 в таком протоколе называются этапом привязки, а шаг 4 — этапом открытия бита. В этом протоколе для значения r, в которое упаковывается бит b, (аналог ящика в физической реализации) обычно используется термин блоб (blob), для Алисы — получатель, а для Боба — отправитель. Говоря неформально, от протокола привязки к биту требуется такая конструкция блоба, которая обеспечивает одновременное выполнение следующих двух требований:

1) после выполнения этапа привязки получатель не может самостоятельно определить, какой бит упакован в блоб;

2) на этапе открытия бита отправитель может открыть любой блоб либо только как 0, либо только как 1.

В нашем случае требование 1) выполняется безусловно, т. е. вне зависимости от того, какими вычислительными ресурсами обладает получатель, он не может самостоятельно узнать, какой бит находится в блобе. В таких случаях говорят, что протокол гарантирует безусловную безопасность отправителя. В то же время безопасность получателя основывается на недоказанном предположении о вычислительной трудности задачи дискретного логарифмирования. Такая асимметрия типична для многих типов криптографических протоколов. Она имеется, например, в доказательствах с вычислительно нулевым разглашением. Но существует и другой тип протоколов привязки к биту, в которых уже безопасность получателя безусловна, а безопасность отправителя основывается на недоказанных предположениях. Для многих типов криптографических протоколов с указанной выше асимметрией построены такого рода двойственные протоколы.

Протокол привязки к биту — один из основных типов примитивных криптографических протоколов. Он находит многочисленные при-

менения в криптографии. В качестве иллюстрации рассмотрим способ повышения эффективности доказательств с нулевым разглашением на примере рассмотренного в главе 2 протокола для задачи ИЗОМОРФИЗМ ГРАФОВ. В этом протоколе главная проблема — большое количество раундов, растущее пропорционально размеру графа. Достаточно естественная идея — выполнить все эти последовательные раунды параллельно. На первом шаге **P** выбирает m случайных перестановок $\pi_1, \ldots, \pi_m$, вычисляет $H_1 = \pi_1 G_1, \ldots, H_m = \pi_m G_1$ и посылает все эти m графов **V**. На втором шаге **V** выбирает m случайных битов $\alpha_1, \ldots, \alpha_m$ и посылает их **P**, а на третьем **P** формирует все m требуемых перестановок и посылает их **V**.

Но будет ли такой протокол доказательством с нулевым разглашением? Прежде всего заметим, что этот протокол трехраундовый, а как показывает уже упоминавшийся в разделе 2 результат Гольдрайха и Кравчика, трехраундовых доказательств с нулевым разглашением скорее всего не существует. Тот метод, который использовался в главе 2 для построения моделирующей машины, срабатывал, поскольку при последовательном выполнении протокола моделирующая машина могла угадать на каждом раунде запрос α_i проверяющего с вероятностью $1/2$. В параллельном же варианте вероятность угадывания равна $1/2^m$, т. е. пренебрежимо мала. Кроме того, проверяющий формирует свои запросы $\alpha_1, \ldots, \alpha_m$, уже получив от **P** все графы $H_1, \ldots, H_m$, и может выбирать их (запросы) зависящими достаточно сложным образом от всех этих графов. Это также представляется непреодолимым препятствием при попытке построения моделирующей машины.

Зависимость $\alpha_1, \ldots, \alpha_m$ от $H_1, \ldots, H_m$ можно предотвратить следующим образом. Проверяющий **V** выбирает свои запросы в самом начале выполнения протокола, еще до того, как увидит $H_1, \ldots, H_m$. Каждый бит α_i упаковывается в блоб r_i, и **V** посылает все блобы $r_1, \ldots, r_m$ доказывающему. Только после этого **P** посылает **V** все графы $H_1, \ldots, H_m$. В ответ **V** открывает блобы, а **P**, получив $\alpha_1, \ldots, \alpha_m$, формирует требуемые перестановки и посылает их **V**.

В результате количество раундов в протоколе возросло, но осталось константой (достаточно 5 раундов). При использовании протокола привязки к биту, обеспечивающего безусловную безопасность отправителя, даже обладающий неограниченными вычислительными возможностями доказывающий не может извлечь из блобов $r_1, \ldots, r_m$ никакой информации о запросах $\alpha_1, \ldots, \alpha_m$. Поэтому корректность протокола сохраняется.

Итак, второе из указанных выше препятствий устранено. А как быть с первым? Оказывается, моделирующей машине совсем необязательно угадывать запросы $\mathbf{V}^*$. Следующая замечательная идея мно-

гократно использовалась в работах, посвященных криптографическим протоколам (см. [15]). Моделирующая машина M_{V^*}, получив от $\mathbf{V}^*$ блобы, запоминает состояние машины $\mathbf{V}^*$, выбирает графы $H'_1, \ldots, H'_m$, как случайные перестановки графа G_0, и передает их $\mathbf{V}^*$. В ответ $\mathbf{V}^*$ открывает блобы, и M_{V^*} получает запросы $\alpha_1, \ldots, \alpha_m$. После этого моделирующая машина формирует графы $H_1, \ldots, H_m$ как ответы на запросы $\alpha_1, \ldots, \alpha_m$, восстанавливает запомненное состояние машины $\mathbf{V}^*$ и передает ей $H_1, \ldots, H_m$. Поскольку протокол привязки к биту предполагается стойким, машина M_{V^*} вновь получит те же самые биты $\alpha_1, \ldots, \alpha_m$ и успешно завершит моделирование, выдавая $\alpha_1, \ldots, \alpha_m$, $H_1, \ldots, H_m$, требуемые перестановки, а также блобы $r_1, \ldots, r_m$ и те сообщения, которые были выданы машиной $\mathbf{V}^*$ для их открытия.

Предположим, что то предположение, на котором основывается стойкость протокола привязки к биту, стало доказанным фактом. Например, удалось доказать, что не существует полиномиальных алгоритмов для задачи дискретного логарифмирования. Даже в этом случае полиномиально ограниченный отправитель в приведенном выше протоколе привязки к биту может угадать x хоть и с малой, но ненулевой вероятностью, и обмануть получателя. Из-за этого рассмотренный нами параллельный вариант протокола будет доказательством лишь со статистически нулевым разглашением. А исходный последовательный вариант (см. главу 2) обладал свойством абсолютно нулевого разглашения. В работе Беллара, Микали и Островски [15], где был предложен описанный выше способ преобразования последовательных протоколов в параллельные, используется специальная конструкция блобов для задачи ИЗОМОРФИЗМ ГРАФОВ, сохраняющая свойство абсолютно нулевого разглашения.

5. Еще раз о разделении секрета

В работе [16] со ссылкой на книгу «Gent und seine schönheiten» (Thill–Verlag, Brüssel, 1990) описывается следующий исторический пример. В XIII–XIV веках в г. Генте была построена ратушная башня. В «секрете» (secreet), самом надежном помещении башни, хранились уставы и привилегии, имевшие важное значение. Помещение имело две двери, каждая с тремя замками, ключи от которых находились во владении различных цехов. Документы хранились в шкафу, который, в свою очередь, также запирался на три ключа. Один из этих ключей хранился у фогта, а два других — у главного шеффена.

Мы привели этот пример в разделе, посвященном протоколам разделения секрета, главным образом для того, чтобы показать, что хотя криптографические протоколы — сравнительно молодая отрасль криптографии, задачи, которые решаются с помощью протоколов, возникли очень давно и имеют свою историю.

Почему «еще раз о разделении секрета»? В главе 5 разделение секрета рассматривается в основном как математическая, прежде всего комбинаторная, задача. Здесь же мы его обсуждаем как криптографический протокол. При этом предполагается, что читатель знаком с главой 5.

В протоколе разделения секрета имеются n участников $P_1, \ldots, P_n$, которых мы будем называть процессорами, и один выделенный участник D, называемый дилером (или, иногда, лидером). Протокол состоит из двух фаз.

На фазе разделения секрета дилер, знающий некоторый секрет s, генерирует n долей секрета $s_1, \ldots, s_n$ и посылает s_i процессору P_i по защищенному каналу связи. На фазе восстановления секрета любое подмножество из не менее чем $t+1$ процессоров, где t — параметр протокола, однозначно восстанавливает секрет, обмениваясь сообщениями по защищенным каналам связи. А любое подмножество из не более чем t процессоров не может восстановить секрет (что означают слова «не может восстановить» будет пояснено ниже).

Как и в других типах криптографических протоколов, в протоколе разделения секрета участники, вообще говоря, не доверяют друг другу и каждый из них может оказаться противником. В том числе и дилер. Можно ли обеспечить какую-либо защиту честных участников даже и в этом случае? Безусловно, нечестный дилер может просто саботировать выполнение протокола. Но если дилер пытается обмануть более хитрым способом, то от этого, оказывается, можно защититься следующим образом. Фаза разделения секрета начинается с того, что дилер публикует секрет s в «зашифрованном» виде (точнее было бы сказать — выполняет привязку к строке s, по аналогии с привязкой к биту). С помощью этой информации каждый процессор P_i может проверить, что значение s_i, полученное им от дилера, действительно является долей секрета s. Такой протокол называется протоколом проверяемого разделения секрета. В обычных схемах разделения секрета рассматривается пассивный противник, а именно, противником являются не более чем t участников, которые, объединив свои доли, пытаются получить какую-либо информацию о значении секрета. На фазе восстановления секрета в протоколе проверяемого разделения секрета действует активный противник: нечестные участники могут преследовать цель сорвать восстановление значения s честными участниками, посылая им вместо своих долей секрета любую другую

информацию. От протокола требуется, чтобы честные участники, если их по крайней мере $t+1$, всегда правильно восстанавливали значение s.

Рассмотрим конструкцию протокола проверяемого разделения секрета из работы [17]. Конструкция основана на задаче дискретного логарифмирования.

В соответствии со схемой Шамира дилер выбирает случа́йный полином $Q(x) = a_0 + a_1x + \cdots + a_tx^t$ степени t, где $a_0 = s$, вычисляет $r_i = g^{a_i} \mod p$ $(i = 0, 1, \ldots, t)$ и публикует $r_0, \ldots, r_t$. После этого для всякого $j = 1, \ldots, n$ дилер вычисляет $s_j = Q(j)$ и посылает это значение процессору P_j по защищенному каналу. Процессор P_j, проверяя равенство

$$g^{s_j} = r_0 \cdot (r_1)^j \cdot \cdots \cdot (r_t)^{j^t} \mod p,$$

убеждается, что s_j — доля секрета s. В самом деле,

$$r_0 \cdot (r_1)^j \cdot \cdots \cdot (r_t)^{j^t} = g^{a_0} \cdot g^{a_1 j} \cdot \cdots \cdot g^{a_t j^t} = g^{a_0 + a_1 j + \cdots + a_t j^t} = g^{Q(j)} \mod p.$$

Конструкцию протокола для фазы восстановления секрета рассмотрим в наиболее простом случае, когда дилер честный. На этой фазе каждый процессор P_j посылает каждому другому процессору P_i свою долю s_j. Всякий честный участник P_i, получив некоторое значение s_j от P_j, проверяет это значение, как описано выше, и отбрасывает все доли s_j, не прошедшие проверку. Поскольку честных участников не менее $t+1$, P_i получит по крайней мере $t+1$ правильных долей секрета. Используя алгоритм восстановления секрета из схемы Шамира, P_i восстановит значение s.

В отличие от схем разделения секрета, рассматриваемых в главе 5, стойкость данного протокола основывается на предположении о вычислительной трудности задачи дискретного логарифмирования. Поэтому, если в обычных схемах разделения секрета требуется, чтобы любое подмножество участников, не составляющее кворума, не получало никакой информации о секрете, то во многих схемах проверяемого разделения секрета такое подмножество лишь «не может восстановить» секрет. В том смысле, что для его восстановления требуется решить некоторую гипотетически трудную вычислительную задачу. В рассмотренном выше примере всякий участник мог бы узнать секрет s, если бы он умел вычислять дискретные логарифмы.

Одно из возможных приложений схем разделения секрета — организация хранения криптографических ключей. Свойство проверяемости представляется далеко не лишним для таких приложений. Но круг приложений схем проверяемого разделения секрета существенно шире.

Предположим, что с помощью описанной выше схемы разделены два секрета s_1 и s_2 и что оба эти секреты являются числами. Теперь представим себе ситуацию, что после этого потребовалась разделить секрет $s = s_1 + s_2$. Конечно, это может сделать дилер с помощью того же протокола. А могут ли процессоры выполнить то же самое без участия дилера?

Пусть $Q_1(x) = a_0 + a_1x + \cdots + a_tx^t$ и $Q_2(x) = b_0 + b_1x + \cdots + b_tx^t$ — полиномы, которые использовались для разделения секретов s_1 и s_2 соответственно. Пусть $r_i^1 = g^{a_i} \bmod p$ и $r_i^2 = g^{b_i} \bmod p$ для $i = 0, \ldots, t$. Для любого $j = 1, \ldots, n$ пусть $s_j^1 = Q_1(j)$ и $s_j^2 = Q_2(j)$ — доли секретов s_1 и s_2, полученные процессором P_j. Ясно, что $Q(x) = Q_1(x) + Q_2(x)$ — также полином степени t и $Q(0) = s$.

Поэтому каждый процессор P_j может вычислить долю s_j секрета s просто по формуле $s_j = s_j^1 + s_j^2$. Эти доли проверяемы с помощью значений $r_i = r_i^1 \cdot r_i^2 \mod p$.

Рабин и Бен-Ор показали [18], что выполняя такого рода вычисления над долями секретов, процессоры могут вычислить любую функцию над конечным полем «проверяемым образом». Этот результат относится к области протоколов конфиденциального вычисления (secure multi party computation). Типичная задача здесь такая. Требуется вычислить значение функции f на некотором наборе значений аргументов $y_1, \ldots, y_m$. С помощью схемы проверяемого разделения секрета вычисляются доли $x_1, \ldots, x_n$ этих значений. В начале выполнения протокола доля x_i известна процессору P_i и только ему. Протокол должен обеспечивать вычисление значения $f(x_1, \ldots, x_n) = f(y_1, \ldots, y_m)$ таким образом, чтобы для некоторого параметра t:

1) в результате выполнения протокола любое подмножество из не более чем t процессоров не получало никакой информации о значениях x_i других процессоров (кроме той, которая следует из известных им долей и значения функции $f(x_1, \ldots, x_n)$);

2) при любых действиях нечестных участников остальные участники вычисляют правильное значение $f(x_1, \ldots, x_n)$, если только количество нечестных участников не превосходит t.

Протоколы конфиденциального вычисления, ввиду своей общности, представляют несомненный теоретический интерес. Кроме того, многие типы прикладных криптографических протоколов (например, протоколы голосования) по существу являются частными случаями протоколов конфиденциального вычисления.

При различных предположениях о процессорах и о сети связи доказан ряд теорем следующего типа: если t не превосходит некоторого порогового значения (зависящего от n и от предположений), то всякая вычислимая функция имеет протокол конфиденциального вычисления.

6. Поиграем в «кубики». Протоколы голосования

“ ... всеобщее голосование бессмысленно.

... Вы, вероятно, согласитесь со мной, что гениальные люди встречаются редко, не правда ли? Но будем щедры и допустим, что во Франции их сейчас имеется человек пять. Прибавим, с такой же щедростью, двести высокоталантливых людей, тысячу других, тоже талантливых, каждый в своей области, и десять тысяч человек так или иначе выдающихся. Вот вам генеральный штаб в одиннадцать тысяч двести пять умов. За ним идет армия посредственности, за которой следует вся масса дурачья. А так как посредственность и дураки всегда составляют огромное большинство, то немыслимо представить, чтобы они могли избрать разумное правительство.”

Г. де Мопассан. «Обед и несколько мыслей».

В этом разделе мы займемся бессмысленными вещами, а именно, обсудим, как помочь «массе дурачья» избрать «неразумное правительство», т. е. рассмотрим протоколы голосования.

У читателя, ознакомившегося с предшествующими разделами, могло сложиться впечатление, что криптографические протоколы — в общем-то не такая уж и сложная вещь. Но дело в том, что до сих пор мы выбирали именно такие протоколы, конструкции которых, на наш взгляд, наиболее просто понять при первом знакомстве с предметом. К тому же изложение велось на полуформальном уровне, чтобы основные идеи не затуманивались обилием технических деталей. Большинство же типов прикладных криптографических протоколов достаточно сложны, и хорошим примером здесь служат протоколы голосования.

Как отмечалось во введении, примитивные криптографические протоколы используются как своеобразные «строительные блоки» или «кубики», из которых складывается прикладной протокол. В данном разделе мы рассмотрим не столько конструкцию каждого отдельного «кубика», сколько процесс сборки.

Пусть в голосовании участвуют l избирателей $V_1, \ldots, V_l$, которые являются абонентами компьютерной сети и подают свои голоса в электронной форме. Предположим для простоты, что голосование имеет два исхода: «за» и «против», которые будут представляться как 1 и -1 соответственно. Из всех возможных требований к протоколу голосования выделим пока два основных:

1) голосование должно быть тайным;

2) должна быть обеспечена правильность подсчета голосов.

Как уже отмечалось в предыдущем разделе, протоколы голосования можно рассматривать как частный случай протоколов конфиденциального вычисления. В начальный момент у каждого участника V_i есть секретное значение $b_i \in \{-1, 1\}$ — его голос, — и требуется вычислить функцию $f(b_1 \ldots, b_l) = \sum_{i=1}^{l} b_i$. Протокол конфиденциального вычисления удовлетворяет двум указанным требованиям, если только доля нечестных участников не слишком велика. У такого решения есть одно замечательное достоинство — в протоколе участвуют только избиратели, т. е. не требуется никакого центрального органа, который пользовался бы доверием голосующих. Но есть и весьма серьезный недостаток. Протоколы конфиденциального вычисления настолько сложны (с точки зрения количества вычислений, выполняемых каждым участником, и количества пересылаемой информации), что уже при сравнительно небольших l они практически невыполнимы.

Остается второй путь — создание центра подсчета голосов (в дальнейшем для краткости будем называть его просто центр). Сначала предположим, что центр честный и пользуется безусловным доверием всех избирателей. В такой ситуации напрашивается следующее решение. Центр выбирает секретный x и открытый y — ключи некоторой криптосистемы с открытым ключом — и публикует y. Каждый избиратель V_i посылает центру сообщение, содержащее идентификатор этого избирателя и его голос b_i, зашифрованный на ключе y. Центр проверяет соответствие поданных бюллетеней спискам избирателей, расшифровывает бюллетени и отбрасывает недействительные (в которых голоса отличны от -1 и 1), подсчитывает и публикует итог.

Уже в этой простой схеме есть «подводный камень». Если каждый избиратель просто шифрует свой бит b_i на ключе y, то возможных криптограмм всего две и ни о какой анонимности голосов речи быть не может. Можно шифровать строку, которая состоит из бита b_i, дополненного, например, справа случайной строкой. Это накладывает дополнительные требования на криптосистему: старший бит открытого текста должен быть трудным, т. е. задача его вычисления по криптограмме должна быть эквивалентна (в смысле полиномиальной сводимости) задаче вычисления всего открытого текста. Такие криптосистемы существуют, но лучше использовать криптосистему вероятностного шифрования (см. [19]), в ней криптограмма сообщения m на ключе k вычисляется с помощью рандомизированного алгоритма: $c = E_k(m, r)$, где r — случайная строка. Это означает, что у каждого сообщения существует, вообще говоря, экспоненциально много криптограмм, вычисленных на одном и том же ключе. Но дешифрование при этом всегда однозначно! Криптосистемы вероятностного шифрования были введены в работе Гольдвассер и Микали [19], где при некоторых предположениях доказано существование криптосистем такого типа, обладающих

так называемой семантической стойкостью. Это — своего рода аналог шенноновской абсолютной стойкости, но относительно противника, работающего за полиномиальное время.

Мы рассмотрим в качестве примера один из вариантов криптосистемы Эль-Гамаля [20], основанной на задаче дискретного логарифмирования. В обозначениях из раздела 2 пусть G_q — подгруппа Z_p^*, порожденная g. Для сообщения $m \in G_q$ выбирается $\alpha \in_R Z_q$ и вычисляется криптограмма (a, b), где $a = g^\alpha \mod p$, $b = y^\alpha m \mod p$. Получатель, знающий секретный ключ x, вычисляет

$$b/a^x = y^\alpha m/(g^\alpha)^x = y^\alpha m/g^{x\alpha} = y^\alpha m/y^\alpha = m \mod p.$$

Вернемся к протоколу голосования. Пусть h — еще один порождающий группы G_q. Тогда для $b \in \{-1, 1\}$ бюллетень вычисляется в виде $(g^\alpha, y^\alpha h^b)$. После применения алгоритма дешифрования центр получит значение $h^b \mod p$, после чего бит b можно извлечь, просто подставляя оба значения 1 и -1.

В такой схеме голосование, по существу, не может быть тайным, поскольку центр знает, как голосовал каждый избиратель. Но с правильностью подсчета голосов ситуация иная. Предположим, что для проведения голосования создано табло — хранилище информации, в котором для каждого избирателя выделена отдельная строка. Эта строка содержит, например, полные паспортные данные избирателя, и в эту строку он помещает свой бюллетень. Предполагается, что табло доступно на чтение всем участникам голосования, а также сторонним наблюдателям. По истечении срока подачи голосов табло «закрывается», т. е. фиксируется его состояние. После этого выделяется некоторое время, в течение которого каждый избиратель проверяет содержимое своей строки на табло. Все претензии разбираются, при необходимости вносятся соответствующие изменения и, когда все избиратели удовлетворены, содержимое табло фиксируется окончательно.

После этого центр вычисляет $z = \sum_{i=1}^{l} b_i$ и публикует итог голосования z. Пусть $(g^{\alpha_i}, y^{\alpha_i} h^{b_i})$ — бюллетень избирателя V_i. Поскольку все бюллетени находятся на табло, любой избиратель, а также всякий сторонний наблюдатель, может вычислить

$$(\prod_{i-1}^{l} g^{\alpha_i} \mod p, \prod_{i=1}^{l} y^{\alpha_i} h^{b_i} \mod p).$$

Обозначим $A = \prod_{i=1}^{l} g^{\alpha_i} \mod p$, $B = \prod_{i=1}^{l} y^{\alpha_i} \mod p$. Если центр правильно подсчитал голоса, то должно выполняться равенство $h^z = \prod_{i=1}^{l} h^{b_i} \mod p$. Поэтому, если вторую из вычисленных выше величин поделить на h^z, то должно получиться значение B. Пусть $B' = (\prod_{i=1}^{l} y^{\alpha_i} h^{b_i})/h^z \mod p$. Проблема в том, что проверяющий не

знает значение B и не может самостоятельно выяснить, верно ли, что $B' = B$. Но нетрудно проверить, что должно выполняться сравнение $B = A^x \mod p$. Поэтому проверяющий может потребовать от центра доказательство следующего факта: дискретный логарифм B' по основанию A равен дискретному логарифму y по основанию g. Мы приводим предназначенный для этой цели протокол Шаума и Педерсена [21], цитируя его по работе [22].

Протокол Шаума и Педерсена

1. Доказывающий выбирает $k \in_R Z_q$, вычисляет $(\beta, \gamma) = (g^k \mod p, A^k \mod p)$ и посылает (β, γ) проверяющему.
2. Проверяющий выбирает запрос $e \in_R Z_q$ и посылает e доказывающему.
3. Доказывающий вычисляет $s = k + ex \mod q$ и посылает s проверяющему.
4. Проверяющий убеждается, что $g^s = \beta y^e \mod p$ и $A^s = \gamma B^e \mod p$, и принимает доказательство. В противном случае, если хотя бы одно из этих сравнений не выполняется, — отвергает.

Читатель, который ознакомился со вторым разделом данной главы, без труда обнаружит сходство этого протокола со схемой аутентификации Шнорра. Такому читателю будет полезно попытаться самостоятельно провести анализ стойкости данного протокола.

В принципе центр может доказать утверждение $B' = A \mod p$ каждому желающему. Неудобство в том, что протокол интерактивный. Но используя тот же прием, которым схема аутентификации Шнорра преобразуется в схему электронной подписи (см. раздел 2), можно и этот протокол сделать неинтерактивным. В таком случае центр может просто опубликовать неинтерактивное доказательство вместе с итогом z.

Конечно, предположение о том, что все избиратели доверяют одному центру подсчета голосов, весьма далеко от реальности. Можно создать n центров $C_1, \ldots, C_n$. Тогда предположение о том, что по крайней мере t центров из n честные, где, например, $t \geqslant 2n/3$, выглядит более реалистичным.

В этом случае центры совместно выбирают и публикуют три случайных порождающих g, y и h группы G_q. Бюллетень избирателя V_i формируется так же, как и в предыдущем варианте: $(g^{\alpha_i}, y^{\alpha_i} h^{b_i})$. Но теперь центры не в состоянии сами дешифровать эту криптограмму. Вместо этого каждый из них вычисляет $(\prod_{i=1}^{l} g^{\alpha_i}, \prod_{i=1}^{l} y^{\alpha_i} h^{b_i})$. Избиратель V_i с помощью описанной в предыдущем разделе схемы проверяемого разделения секрета создает доли $\alpha_{i_1}, \ldots, \alpha_{i_n}$ секретного значения

α_i и передает долю α_{i_j} центру C_j. Далее, выполняя вычисления над долями (см. раздел 5), центры вычисляют $\delta = \sum_{i=1}^{l} \alpha_i \mod p$. Если при этом хотя бы t центров честные, то остальные не смогут восстановить ни одно из значений α_i. Полученный результат δ проверяется: должно выполняться сравнение $g^\delta = \prod_{i=1}^{l} g^{\alpha_i} \mod p$. Если оно выполнено, то значение δ публикуется, и этого значения достаточно для вычисления итога голосования. В самом деле, $(\prod_{i=1}^{l} y^{\alpha_i} h^{b_i})/y^\delta = \prod_{i=1}^{l} h^{b_i} \mod p$. Правда, для того чтобы получить итог z, необходимо вычислить дискретный логарифм полученной величины. Но поскольку абсолютное значение z невелико (заведомо не превосходит количества избирателей l) это можно сделать простым перебором.

В этой схеме возникает новая проблема — необходимо обеспечить протокол, с помощью которого центры совместно выбирают порождающие g, y и h, причем так, что ни один из них не является известной степенью другого. Эту проблему можно решить следующим способом. Пусть G — вероятностный алгоритм, который генерирует три случайных порождающих. Его можно рассматривать как детерминированный алгоритм, который получает на входе помимо чисел p и q еще и случайную строку r требуемой длины. В разделе 4 был описан протокол подбрасывания монеты, который очевидным образом обобщается на случай n участников. С помощью такого обобщенного протокола центры могут по биту сгенерировать строку r (существуют и более эффективные схемы). Если хотя бы один из центров честный, то r — случайная строка. По окончании выполнения протокола центры публикуют r, g, y и h. Поскольку алгоритм G предполагается общеизвестным, всякий желающий может проверить, что g, y и h получены с помощью строки r.

Все проблемы решены? Отнюдь! Если в случае одного центра последний расшифровывал все бюллетени и проверял их правильность, отбрасывая недействительные, то в случае n центров нечестный избиратель может попытаться с помощью недействительного бюллетеня сорвать выборы или исказить их итог. Эту проблему можно решить, если потребовать, чтобы вместе с бюллетенем каждый избиратель публиковал доказательство, что бюллетень построен правильно. Другими словами, требуется протокол, с помощью которого для данного бюллетеня (A_i, B_i) избиратель V_i доказывал, что он знает $\alpha_i \in Z_q$ и $b_i \in \{-1, 1\}$ такие, что $A_i = g^{\alpha_i} \mod p$, $B_i = y^{\alpha_i} h^{b_i} \mod p$. При этом протокол не должен давать проверяющему никакой полезной для него информации о значениях α_i и b_i. Пример такого протокола дан в работе [22]; мы его не воспроизводим здесь ввиду громоздкости.

Теперь, наконец, все? Опять нет! ... Впрочем, поставим на этом точку. Надеемся, что читатель уже убедился в том, что протоколы голосования — нетривиальный объект и всевозможные его технические

аспекты можно обсуждать еще очень долго. К тому же мы рассматривали только два основных требования к протоколам голосования — анонимность и правильность подсчета голосов. А ведь существуют и другие. Причем новые требования могут возникать уже в процессе анализа протоколов голосования, когда обнаруживается, что последние обладают некоторыми неожиданными свойствами, которых нет в ныне общепринятых неэлектронных системах выборов. Например, при использовании описанной выше конструкции бюллетеня избиратель V_i знает значения α_i и b_i, которые он впоследствии может использовать для доказательства, что голосовал данным определенным образом (например, «за», а не «против»). Это создает возможности покупки голосов и требует разработки соответствующей криптографической защиты.

Дальнейшие подробности протоколов голосования см. в работах [22], [23], из которых заимствованы изложенные выше идеи построения таких протоколов.

И последнее, о чем хотелось бы сказать в этом разделе. Всюду выше мы обсуждали (неформально) лишь стойкость различных криптографических примитивов, используемых в протоколе голосования, по отдельности. Но при композиции стойких криптографических протоколов может получиться нестойкий. Исследование стойкости композиций протоколов — еще одна, и зачастую весьма непростая, задача теоретической криптографии.

7. За пределами стандартных предположений. Конфиденциальная передача сообщений

"Ты умеешь считать? — спросила Белая Королева.— Сколько будет один плюс один плюс один плюс один плюс один плюс один плюс один плюс один плюс один плюс один?

— Я не знаю, — ответила Алиса. — Я сбилась со счета.

Она не умеет считать,— сказала Черная Королева."

Л. Кэрролл. «Алиса в зазеркалье».

Криптографические протоколы в своем большинстве относятся к одной из двух категорий. В первую входят протоколы, обладающие теоретико-информационной или абсолютной стойкостью. Вторая категория — это те протоколы, стойкость которых основывается на вычислительной сложности каких-либо математических задач. Основывается в том смысле, что либо стойкость протокола доказана при некотором конкретном предположении, либо справедливость предположения необходима для стойкости, т. е. противник, научившийся решать

соответствующую задачу эффективно, заведомо «взламывает» протокол.

В теоретической криптографии, как и во всякой научной дисциплине, представлено все многообразие направлений научного поиска. Одно из таких направлений — разработка криптографических схем, стойких при каких-либо нестандартных предположениях. Работ, посвященных этой тематике, немного, но они есть. И в этом заключительном разделе мы рассмотрим пример криптографического протокола, доказуемо стойкого при нестандартном предположении.

Задача конфиденциальной передачи сообщений состоит в следующем. Имеются два участника, Алиса и Боб, которые являются абонентами сети связи. Участники соединены n проводами, по каждому из которых можно пересылать сообщения в обе стороны, независимо от того, что происходит с другими проводами. Никакой общей секретной информации у Алисы и Боба изначально нет. У Алисы имеется конфиденциальное сообщение m, и задача состоит в том, чтобы его конфиденциальным же образом передать Бобу. Против участников действует активный противник, который может полностью контролировать не более t проводов. Полный контроль означает, что противник перехватывает все сообщения, передаваемые по данному проводу, и может заменять их любыми другими сообщениями.

Если положить $n = 1$ и рассматривать пассивного противника, который только подслушивает, то получим классическую модель секретной связи, введенную еще Шенноном. Задача конфиденциальной передачи сообщений в этой модели всегда решалась с помощью криптосистем. Но выделение криптосистем в отдельную категорию — это, скорее, дань традиции. На самом деле требуется протокол конфиденциальной передачи сообщений, при построении которого используются такие криптографические примитивы, как алгоритмы шифрования и дешифрования, протокол распределения ключей и т. д. Например, в случае криптосистемы с открытым ключом примитивом может быть и протокол электронной подписи, если тот орган (центр доверия), которому поручено вести сертифицированный справочник, заверяет все открытые ключи, хранящиеся в этом справочнике, своей подписью.

В предположении $n \geqslant 2t + 1$ задача конфиденциальной передачи сообщений может быть решена с помощью следующего простого протокола.

Прежде всего заметим, что при данном предположении Алиса и Боб имеют абсолютно надежный открытый канал связи. Если, например, Алисе необходимо послать сообщение x Бобу, то она посылает x по каждому из проводов, а Боб выбирает из всех полученных значений то, которое появилось по крайней мере $t + 1$ раз.

Далее, пусть q — большое простое число, $q > n$. Алиса выбирает случайный полином $Q(x)$ степени t над Z_q. Пусть $P = Q(0)$. Идея состоит в том, чтобы передать P Бобу в качестве одноразового ключа для шифра Вернама. При этом нужно обеспечить такую передачу, чтобы противник не мог узнать ничего о значении P. Для этого Алиса использует пороговую схему разделения секрета, т. е. посылает значение $Q(j)$ по j-му проводу. Пусть r_j, $j = 1 \ldots, n$ — значение, которое Боб получил по j-му проводу. Если все n пар (j, r_j) интерполируются полиномом степени t, то передача успешна и Боб может вычислить ключ P. Далее Алиса и Боб общаются по описанному выше открытому каналу. Если Боб получил ключ P, то он уведомляет об этом Алису специальным сообщением. Алиса вычисляет и посылает Бобу криптограмму $z = P + m \mod q$. Боб дешифрует криптограмму и получает сообщение m. Если пары (j, r_j) не интерполируются полиномом степени t, то Боб посылает все эти пары Алисе, которая обнаружит хотя бы для одного j, что $r_j \neq Q(j)$. Ясно, что в этом случае провод контролируется противником. Алиса посылает Бобу список всех таких номеров j, и соответствующие провода исключаются из работы. После этого Алиса и Боб повторяют весь протокол, используя оставшиеся провода. Ясно, что после не более t повторений передача ключа будет успешной.

Данный протокол — простейший вариант протокола конфиденциальной передачи сообщений из работы Долева и др. [24]. В этой работе предложен значительно более эффективный протокол, который при том же предположении $n \geqslant 2t+1$ является доказуемо стойким против более сильного противника.

Если противник пассивный, т. е. он лишь подслушивает не более t проводов, то задача конфиденциальной передачи сообщений решается совсем просто. Мы предлагаем читателю в качестве несложного упражнения самостоятельно сконструировать соответствующий протокол при $n > t$.

Протокол конфиденциальной передачи сообщений является доказуемо стойким в предположении, что противнику не хватает ресурсов, чтобы контролировать хотя бы половину проводов, которыми соединены Алиса и Боб. Этот результат наводит на следующие размышления. Так называемый здравый смысл подсказывает, что для построения системы секретной связи целесообразно создать закрытую сеть связи, т. е. такую, доступ к которой сильно ограничен. Но, по-видимому, действительно надежное решение следует искать в прямо противоположном направлении: объединить все сети связи в единую открытую сеть с очень большим количеством соединений между каждой парой абонентов. Конечно, это — всего лишь теоретические рассуждения, относящиеся к математической модели реальных систем. Но сколько

раз еще результаты, полученные в математической криптографии, как и в любых других научных дисциплинах, заставят нас усомниться в очевидности тех решений, которые подсказываются здравым смыслом.

8. Вместо заключения

Математическая теория криптографических протоколов развивается совместными усилиями ученых различных стран. Среди авторов работ, посвященных протоколам, встречаются имена математиков США и Израиля, Канады и Голландии, Италии и Японии, Франции и Германии, Дании и Венгрии. Этот список можно продолжить. И лишь наша замечательная математическая школа практически никаких заслуг в этой области не имеет. Будем надеяться, что в недалеком будущем ситуация изменится и не без участия кого-либо из наших читателей.

Литература к главе 3

[1] *Schnorr C. P.* Efficient identification and signatures for smart cards // Proc. Crypto'89, Lect. Notes in Comput. Sci. **V. 435**, 1990. P. 239–252.

[2] *Goldreich O., Krawczyk H.* On the composition of zero-knowledge proof systems // SIAM J. Comput. **V. 25**, No 1, 1996. P. 169–192.

[3] *Goldwasser S., Micali S., Rivest R.* A secure digital signature scheme // SIAM J. Comput. **V. 17**, No 2, 1988. P. 281–308.

[4] *Naor M., Yung M.* Universal one-way hash functions and their cryptographic applications // Proc. 21st Annu. ACM Symp. on Theory of Computing. 1989. P. 33–43.

[5] *Rompel J.* One-way functions are necessary and sufficient for secure signatures // Proc. 22nd Annu. ACM Symp. on Theory of Computing. 1990. P. 387–394.

[6] *Fiat A., Shamir A.* How to prove yourself: practical solutions to identification and signature problems // Proc. Crypto'86, Lect. Notes in Comput. Sci. **V. 263**, 1987. P. 186–194.

[7] *Feige U., Fiat A., Shamir A.* Zero-knowledge proofs of identity // J. of Cryptology, **V. 1**, No 2, 1988. P. 77–94.

[8] *Варновский Н. П.* О стойкости схем электронной подписи с аппаратной поддержкой. Технический отчет. Лаборатория МГУ по математическим проблемам криптографии, 1997.

[9] *Chaum D.* Online cash checks // Proc. EUROCRYPT'89, Lect. Notes in Comput. Sci., **V. 434**, 1990. P. 288–293.

[10] *Yacobi Y.* Efficient electronic money // Proc. ASIACRYPT'94, Lect. Notes in Comput. Sci., **V. 739**, 1994. P. 131–139.

[11] *Brands S.* Untraceable off-line cash in wallets with observers // Proc. Crypto'93, Lect. Notes in Comput. Sci., **V. 773**, 1994. P. 302–318.

[12] *Анохин М. И., Варновский Н. П., Сидельников В. М., Ященко В. В.* Криптография в банковском деле. М.: МИФИ, 1997.

[13] *Chaum D., Pedersen T. P.* Transferred cash grows in size // Proc. EUROCRYPT'92, Lect. Notes in Comput. Sci., **V. 658**, 1993. P. 390–407.

[14] *Blum M.* Coin flipping by telephone: A protocol for solving impossible problems // Proc. 24th IEEE Comp. Conf., 1982. P. 133–137; reprinted in SIGACT News, **V. 15**, No 1, 1983. P. 23–27.

[15] *Bellare M., Micali S., Ostrovsky R.* Perfect zero-knowledge in constant rounds // Proc. 22nd Annu. ACM Symp. on Theory of Computing. 1990. P. 482–493.

[16] *Kersten A. G.* Shared secret Schemes aus geometrisches sicht. Mitteilungen mathem. Seminar Giessen, Heft 208, 1992.

[17] *Feldman P.* A practical scheme for non-interactive verifiable secret sharing // Proc. 28th Annu. Symp. on Found. of Comput. Sci. 1987. P. 427–437.

[18] *Rabin T., Ben-Or M.* Verifiable secret sharing and multiparty protocols with honest majority // Proc. 21st Annu. ACM Symp. on Theory of Computing. 1989. P. 73–85.

[19] *Goldwasser S., Micali S.* Probabilistic encryption // J. of Computer and System Sciences, **V. 28**, No 2, 1984. P. 270–299.

[20] *El Gamal T.* A public-key cryptosystem and a signature scheme based on discrete logarithms // IEEE Trans. Inf. Theory, **IT-31**, No 4, 1985. P. 469–472.

[21] *Chaum D., Pedersen T. P.* Wallet databases with observers // Proc. Crypto'92, Lect. Notes in Comput. Sci., **V. 740**, 1993. P. 89–105.

[22] *Cramer R., Gennaro R., Schoenmakers B.* A secure and optimally efficient multi-authority election scheme // Proc. EUROCRYPT'97, Lect. Notes in Comput. Sci., **V. 1233**, 1997. P. 103–118.

[23] *Cramer R., Franklin M., Schoenmakers B., Yung M.* Multi-authority secret ballot elections with linear work // Proc. EUROCRYPT'96, Lect. Notes in Comput. Sci., **V. 1070**, 1996. P. 72–83.

[24] *Dolev D., Dwork C., Waarts O., Yung M.* Perfectly secure message transmission // Proc. 31st Annu. Symp. on Found. of Comput. Sci. 1990. P. 36–45.

Глава 4

Алгоритмические проблемы теории чисел

1. Введение

Вопрос «как сосчитать?» всегда сопутствовал теоретико-числовым исследованиям. Труды Евклида и Диофанта, Ферма и Эйлера, Гаусса, Чебышева и Эрмита содержат остроумные и весьма эффективные алгоритмы решения диофантовых уравнений, выяснения разрешимости сравнений, построения больших по тем временам простых чисел, нахождения наилучших приближений и т.д. Без преувеличения можно сказать, что вся теория чисел пронизана алгоритмами. В последние два десятилетия, благодаря в первую очередь запросам криптографии и широкому распространению ЭВМ, исследования по алгоритмическим вопросам теории чисел переживают период бурного и весьма плодотворного развития. Мы кратко затронем здесь лишь те алгоритмические аспекты теории чисел, которые связаны с криптографическими применениями.

Вычислительные машины и электронные средства связи проникли практически во все сферы человеческой деятельности. Немыслима без них и современная криптография. Шифрование и дешифрование текстов можно представлять себе как процессы переработки целых чисел при помощи ЭВМ, а способы, которыми выполняются эти операции, как некоторые функции, определенные на множестве целых чисел. Все это делает естественным появление в криптографии методов теории чисел. Кроме того, стойкость ряда современных криптосистем обосновывается только сложностью некоторых теоретико-числовых задач (см. [22]).

Но возможности ЭВМ имеют определенные границы. Приходится разбивать длинную цифровую последовательность на блоки ограниченной длины и шифровать каждый такой блок отдельно. Мы будем считать в дальнейшем, что все шифруемые целые числа неотрицательны и по величине меньше некоторого заданного (скажем, техническими ограничениями) числа m. Таким же условиям будут удовлетворять и числа, получаемые в процессе шифрования. Это позволяет считать

и те, и другие числа элементами кольца вычетов $\mathbb{Z}/m\mathbb{Z}$. Шифрующая функция при этом может рассматриваться как взаимнооднозначное отображение колец вычетов

$$f : \mathbb{Z}/m\mathbb{Z} \to \mathbb{Z}/m\mathbb{Z},$$

а число $f(x)$ представляет собой сообщение x в зашифрованном виде.

Простейший шифр такого рода — шифр замены, соответствует отображению $f : x \to x+k \pmod m$ при некотором фиксированном целом k. Подобный шифр использовал еще Юлий Цезарь. Конечно, не каждое отображение f подходит для целей надежного сокрытия информации (подробнее об этом см. главу 1).

В 1978 г., см. [1], американцы Р. Ривест, А. Шамир и Л. Адлеман (R.L.Rivest, A.Shamir, L.Adleman) предложили пример функции f, обладающей рядом замечательных достоинств. На ее основе была построена реально используемая система шифрования, получившая название по первым буквам имен авторов — система RSA. Эта функция такова, что

а) существует достаточно быстрый алгоритм вычисления значений $f(x)$;

б) существует достаточно быстрый алгоритм вычисления значений обратной функции $f^{-1}(x)$;

в) функция $f(x)$ обладает некоторым «секретом», знание которого позволяет быстро вычислять значения $f^{-1}(x)$; в противном же случае вычисление $f^{-1}(x)$ становится трудно разрешимой в вычислительном отношении задачей, требующей для своего решения столь много времени, что по его прошествии зашифрованная информация перестает представлять интерес для лиц, использующих отображение f в качестве шифра.

Подробнее об отображениях такого сорта и возможностях их использования в криптографии рассказано в главах 1, 2.

Еще до выхода из печати статьи [1] копия доклада в Массачусетсском Технологическом институте, посвященного системе RSA, была послана известному популяризатору математики М. Гарднеру, который в 1977 г. в журнале Scientific American опубликовал статью [2], посвященную этой системе шифрования. В русском переводе заглавие статьи Гарднера звучит так: *Новый вид шифра, на расшифровку которого потребуются миллионы лет.* Именно статья [2] сыграла важнейшую роль в распространении информации об RSA, привлекла к криптографии внимание широких кругов неспециалистов и фактически способствовала бурному прогрессу этой области, произошедшему в последовавшие 20 лет.

2. Система шифрования RSA

В дальнейшем мы будем предполагать, что читатель знаком с элементарными фактами теории чисел. Тех же, кто хотел бы ознакомиться с ними или напомнить себе эти факты, мы отсылаем к книге [3].

Пусть m и e натуральные числа. Функция f, реализующая схему RSA, устроена следующим образом

$$f : x \to x^e \pmod{m}. \tag{1}$$

Для дешифрования сообщения $a = f(x)$ достаточно решить сравнение

$$x^e \equiv a \pmod{m}. \tag{2}$$

При некоторых условиях на m и e это сравнение имеет единственное решение x.

Для того, чтобы описать эти условия и объяснить, как можно найти решение, нам потребуется одна теоретико-числовая функция, так называемая функция Эйлера. Эта функция натурального аргумента m обозначается $\varphi(m)$ и равняется количеству целых чисел на отрезке от 1 до m, взаимно простых с m. Так $\varphi(1) = 1$ и $\varphi(p^r) = p^{r-1}(p-1)$ для любого простого числа p и натурального r. Кроме того, $\varphi(ab) = \varphi(a)\varphi(b)$ для любых натуральных взаимно простых a и b. Эти свойства позволяют легко вычислить значение $\varphi(m)$, если известно разложение числа m на простые сомножители.

Если показатель степени e в сравнении (2) взаимно прост с $\varphi(m)$, то сравнение (2) имеет единственное решение. Для того, чтобы найти его, определим целое число d, удовлетворяющее условиям

$$de \equiv 1 \pmod{\varphi(m)}, \quad 1 \leqslant d < \varphi(m). \tag{3}$$

Такое число существует, поскольку $(e, \varphi(m)) = 1$, и притом единственно. Здесь и далее символом (a, b) будет обозначаться наибольший общий делитель чисел a и b. Классическая теорема Эйлера, см. [3], утверждает, что для каждого числа x, взаимно простого с m, выполняется сравнение $x^{\varphi(m)} \equiv 1 \pmod{m}$ и, следовательно,

$$a^d \equiv x^{de} \equiv x \pmod{m}. \tag{4}$$

Таким образом, в предположении $(a, m) = 1$, единственное решение сравнения (2) может быть найдено в виде

$$x \equiv a^d \pmod{m}. \tag{5}$$

Если дополнительно предположить, что число m состоит из различных простых сомножителей, то сравнение (5) будет выполняться и без предположения $(a, m) = 1$. Действительно, обозначим $r = (a, m)$ и $s = m/r$. Тогда $\varphi(m)$ делится на $\varphi(s)$, а из (2) следует, что $(x, s) = 1$. Подобно (4), теперь легко находим $x \equiv a^d \pmod{s}$. А кроме того, имеем $x \equiv 0 \equiv a^d \pmod{r}$. Получившиеся сравнения в силу $(r, s) = 1$ дают нам (5).

Функция (1), принятая в системе RSA, может быть вычислена достаточно быстро. Как это сделать, мы обсудим чуть ниже. Пока отметим лишь, что обратная к $f(x)$ функция $f^{-1}: x \to x^d \pmod m$ вычисляется по тем же правилам, что и $f(x)$, лишь с заменой показателя степени e на d. Таким образом, для функции (1) будут выполнены указанные выше свойства а) и б).

Для вычисления функции (1) достаточно знать лишь числа e и m. Именно они составляют открытый ключ для шифрования. А вот для вычисления обратной функции требуется знать число d, оно и является «секретом», о котором речь идет в пункте в). Казалось бы, ничего не стоит, зная число m, разложить его на простые сомножители, вычислить затем с помощью известных правил значение $\varphi(m)$ и, наконец, с помощью (3) определить нужное число d. Все шаги этого вычисления могут быть реализованы достаточно быстро, за исключением первого. Именно разложение числа m на простые множители и составляет наиболее трудоемкую часть вычислений. В теории чисел несмотря на многолетнюю ее историю и на очень интенсивные поиски в течение последних 20 лет, эффективный алгоритм разложения натуральных чисел на множители так и не найден. Конечно, можно, перебирая все простые числа до $\sqrt{m}$, и, деля на них m, найти требуемое разложение. Но, учитывая, что количество простых в этом промежутке, асимптотически равно $2\sqrt{m} \cdot (\ln m)^{-1}$, см. [5], гл. 5, находим, что при m, записываемом 100 десятичными цифрами, найдется не менее $4 \cdot 10^{42}$ простых чисел, на которые придется делить m при разложении его на множители. Очень грубые прикидки показывают, что компьютеру, выполняющему миллион делений в секунду, для разложения числа $m > 10^{99}$ таким способом на простые сомножители потребуется не менее, чем 10^{35} лет. Известны и более эффективные способы разложения целых чисел на множители, чем простой перебор простых делителей, но и они работают очень медленно. Таким образом, название статьи М. Гарднера вполне оправдано.

Авторы схемы RSA предложили выбирать число m в виде произведения двух простых множителей p и q, примерно одинаковых по величине. Так как

$$\varphi(m) = \varphi(pq) = (p-1)(q-1), \tag{6}$$

то единственное условие на выбор показателя степени e в отображении (1) есть

$$(e, p-1) = (e, q-1) = 1. \tag{7}$$

Итак, лицо, заинтересованное в организации шифрованной переписки с помощью схемы RSA, выбирает два достаточно больших простых числа p и q. Перемножая их, оно находит число $m = pq$. Затем выбирается число e, удовлетворяющее условиям (7), вычисляется с помощью (6) число $\varphi(m)$ и с помощью (3) — число d. Числа m и e публикуются,

число d остается секретным. Теперь любой может отправлять зашифрованные с помощью (1) сообщения организатору этой системы, а организатор легко сможет дешифровывать их с помощью (5).

Для иллюстрации своего метода Ривест, Шамир и Адлеман зашифровали таким способом некоторую английскую фразу. Сначала она стандартным образом (a=01, b=02, ... , z=26, пробел=00) была записана в виде целого числа x, а затем зашифрована с помощью отображения (1) при

$$m = 1143816257578888676692577997614661201021829672124236256256184293$$
$$5706935245733897830597123563958705058989075147599290026879543541$$

и $e = 9007$. Эти два числа были опубликованы, причем дополнительно сообщалось, что $m = pq$, где p и q — простые числа, записываемые соответственно 64 и 65 десятичными знаками. Первому, кто дешифрует соответствующее сообщение

$$f(x) = 9686961375462206147714092225435588290575999112457431987469512093$$
$$0816298225145708356931476622883989628013391990551829945157815154,$$

была обещана награда в 100$.

Эта история завершилась спустя 17 лет в 1994 г., см. [5], когда D. Atkins, M. Graff, A. K. Lenstra и P. C. Leyland сообщили о дешифровке фразы, предложенной в [1]. Она[1)] была вынесена в заголовок статьи [5], а соответствующие числа p и q оказались равными

$$p = 3490529510847650949147849619903898133417764638493387843990820577,$$
$$q = 3276913299326670954996198819083446141317764296799294253979828853 3.$$

Интересующиеся могут найти детали вычислений в работе [5]. Здесь же мы отметим, что этот замечательный результат (разложение на множители 129-значного десятичного числа) был достигнут благодаря использованию алгоритма разложения чисел на множители, называемого методом квадратичного решета. Выполнение вычислений потребовало колоссальных ресурсов. В работе, возглавлявшейся четырьмя авторами проекта, и продолжавшейся после предварительной теоретической подготовки примерно 220 дней, на добровольных началах участвовало около 600 человек и примерно 1600 компьютеров, объединенных сетью Internet. Наконец, отметим, что премия в 100$ была передана в Free Software Foundation.

---

1) ***The magic words are squeamish ossifrage.*** Приведем перевод двух последних слов, входящих в эту, по всей видимости, бессмысленную фразу:
squeamish — брезгливый, привередливый, обидчивый;
ossifrage — скопа.

Описанная выше схема RSA ставит ряд вопросов, которые мы и попробуем обсудить ниже. Например, как проводить вычисления с большими числами, ведь стандартное математическое обеспечение не позволяет перемножать числа размером по 65 десятичных знаков? Как вычислять огромные степени больших чисел? Что значит быстрый алгоритм вычисления и что такое сложная вычислительная задача? Где взять большие простые числа? Как, например, построить простое число в 65 десятичных знаков? Существуют ли другие способы решения сравнения (2)? Ведь, если можно найти решение (2), не вычисляя секретный показатель d или не разлагая число m на простые сомножители, да еще сделать это достаточно быстро, вся система RSA разваливается. Наверное, читателю могут прийти в голову и другие вопросы.

Начнем с конца. За 17 лет, прошедших между публикациями работ [1] и [5], никто так и не смог дешифровать предложенную авторами RSA фразу. Конечно, это всего лишь косвенное подтверждение стойкости системы RSA, но все же достаточно убедительное. Ниже мы обсудим теоретические проблемы, возникающие при решении полиномиальных сравнений.

Мы не будем обсуждать, как выполнять арифметические действия с большими целыми числами, рекомендуем читателю обратиться к замечательной книжке Д. Кнута [6, гл. 4]. Заметим только, что большое число всегда можно разбить на меньшие блоки, с которыми компьютер может оперировать так же, как мы оперируем с цифрами, когда проводим вычисления вручную на бумаге. Конечно, для этого нужны специальные программы. Созданы и получили достаточно широкое распространение даже специальные языки программирования для вычислений с большими числами. Укажем здесь два из них — PARI и UBASIC. Эти языки свободно распространяются. Информацию о том, как их получить в пользование, можно найти в книге [17].

3. Сложность теоретико-числовых алгоритмов

Сложность алгоритмов теории чисел обычно принято измерять количеством арифметических операций (сложений, вычитаний, умножений и делений с остатком), необходимых для выполнения всех действий, предписанных алгоритмом. Впрочем, это определение не учитывает величины чисел, участвующих в вычислениях. Ясно, что перемножить два стозначных числа значительно сложнее, чем два однозначных, хотя при этом и в том, и в другом случае выполняется лишь одна арифметическая операция. Поэтому иногда учитывают еще и величину чисел, сводя дело к так называемым битовым операциям, т. е. оценивая количество необходимых операций с цифрами 0 и 1, в двоичной записи чисел. Это зависит от рассматриваемой задачи, от целей автора и т.д.

На первый взгляд странным также кажется, что операции умножения и деления приравниваются по сложности к операциям сложения и вычитания. Житейский опыт подсказывает, что умножать числа значительно сложнее, чем складывать их. В действительности же, вычисления можно организовать так, что на умножение или деление больших чисел понадобится не намного больше битовых операций, чем на сложение. В книге [7] описывается алгоритм Шёнхаге – Штрассена, основанный на так называемом быстром преобразовании Фурье, и требующий $O(n \ln n \ln\ln n)$ битовых операций для умножения двух n-разрядных двоичных чисел. Таким же количеством битовых операций можно обойтись при выполнении деления с остатком двух двоичных чисел, записываемых не более, чем n цифрами. Для сравнения отметим, что сложение n-разрядных двоичных чисел требует $O(n)$ битовых операций.

Говоря в этой статье о сложности алгоритмов, мы будем иметь в виду количество арифметических операций. При построении эффективных алгоритмов и обсуждении верхних оценок сложности обычно хватает интуитивных понятий той области математики, которой принадлежит алгоритм. Формализация же этих понятий требуется лишь тогда, когда речь идет об отсутствии алгоритма или доказательстве нижних оценок сложности. Более детальное и формальное обсуждение этих вопросов см. в главе 2.

Приведем теперь примеры достаточно быстрых алгоритмов с оценками их сложности. Здесь и в дальнейшем мы не будем придерживаться формального описания алгоритмов, стараясь в первую очередь объяснить смысл выполняемых действий.

Следующий алгоритм вычисляет $a^d \pmod m$ за $O(\ln m)$ арифметических операций. При этом, конечно, предполагается, что натуральные числа a и d не превосходят по величине m.

1. Алгоритм вычисления $a^d \pmod m$

1. Представим d в двоичной системе счисления $d = d_0 2^r + \cdots \cdots + d_{r-1} 2 + d_r$, где d_i, цифры в двоичном представлении, равны 0 или 1, $d_0 = 1$.
2. Положим $a_0 = a$ и затем для $i = 1, \ldots, r$ вычислим
$$a_i \equiv a_{i-1}^2 \cdot a^{d_i} \pmod m.$$
3. a_r есть искомый вычет $a^d \pmod m$.

Справедливость этого алгоритма вытекает из сравнения
$$a_i \equiv a^{d_0 2^i + \cdots + d_i} \pmod m,$$
легко доказываемого индукцией по i.

Так как каждое вычисление на шаге 2 требует не более трех умножений по модулю m и этот шаг выполняется $r \leqslant \log_2 m$ раз, то сложность алгоритма может быть оценена величиной $O(\ln m)$.

Второй алгоритм — это классический алгоритм Евклида вычисления наибольшего общего делителя целых чисел. Мы предполагаем заданными два натуральных числа a и b и вычисляем их наибольший общий делитель (a, b).

2. Алгоритм Евклида

1. Вычислим r — остаток от деления числа a на b, $a = bq + r$, $0 \leqslant r < b$.
2. Если $r = 0$, то b есть искомое число.
3. Если $r \neq 0$, то заменим пару чисел $\langle a, b\rangle$ парой $\langle b, r\rangle$ и перейдем к шагу 1.

Не останавливаясь на объяснении, почему алгоритм действительно находит (a, b), докажем некоторую оценку его сложности.

Теорема 3. *При вычислении наибольшего общего делителя (a, b) с помощью алгоритма Евклида будет выполнено не более $5p$ операций деления с остатком, где p есть количество цифр в десятичной записи меньшего из чисел a и b.*

Доказательство. Положим $r_0 = a > b$ и определим $r_1, r_2, \ldots, r_n$ — последовательность делителей, появляющихся в процессе выполнения шага 1 алгоритма Евклида. Тогда

$$r_1 = b, \ldots, \quad 0 \leqslant r_{i+1} < r_i, \quad i = 0, 1, \ldots n-1.$$

Пусть также $u_0 = 1$, $u_1 = 1$, $u_{k+1} = u_k + u_{k-1}$, $k \geqslant 1$, — последовательность Фибоначчи. Индукцией по i от $i = n-1$ до $i = 0$ легко доказывается неравенство $r_{i+1} \geqslant u_{n-i}$. А так как $u_n \geqslant 10^{(n-1)/5}$, то имеем неравенства $10^p > b = r_1 \geqslant u_n \geqslant 10^{(n-1)/5}$ и $n < 5p + 1$.

Немного подправив алгоритм Евклида, можно достаточно быстро решать сравнения $ax \equiv 1 \pmod b$ при условии, что $(a, b) = 1$. Эта задача равносильна поиску целых решений уравнения $ax + by = 1$.

3. Алгоритм решения уравнения $ax + by = 1$

0. Определим матрицу $E = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.
1. Вычислим r — остаток от деления числа a на b, $a = bq + r$, $0 \leqslant r < b$.
2. Если $r = 0$, то второй столбец матрицы E дает вектор $\begin{pmatrix} x \\ y \end{pmatrix}$ решений уравнения.
3. Если $r \neq 0$, то заменим матрицу E матрицей $E \cdot \begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix}$.
4. Заменим пару чисел $\langle a, b\rangle$ парой $\langle b, r\rangle$ и перейдем к шагу 1.

Если обозначить через E_k матрицу E, возникающую в процессе работы алгоритма перед шагом 2 после k делений с остатком (шаг 1),

то в обозначениях из доказательства теоремы 1 в этот момент выполняется векторное равенство $\langle a, b\rangle \cdot E_k = \langle r_{k-1}, r_k\rangle$. Его легко доказать индукцией по k. Поскольку числа a и b взаимно просты, имеем $r_n = 1$, и это доказывает, что алгоритм действительно дает решение уравнения $ax + by = 1$. Буквой n мы обозначили количество делений с остатком, которое в точности такое же, как и в алгоритме Евклида.

Три приведенных выше алгоритма относятся к разряду так называемых полиномиальных алгоритмов. Это название носят алгоритмы, сложность которых оценивается сверху степенным образом в зависимости от длины записи входящих чисел (см. подробности в главе 2). Если наибольшее из чисел, подаваемых на вход алгоритма, не превосходит m, то сложность алгоритмов этого типа оценивается величиной $O(\ln^c m)$, где c — некоторая абсолютная постоянная. Во всех приведенных выше примерах $c = 1$.

Полиномиальные алгоритмы в теории чисел — большая редкость. Да и оценки сложности алгоритмов чаще всего опираются на какие-либо не доказанные, но правдоподобные гипотезы, обычно относящиеся к аналитической теории чисел.

Для некоторых задач эффективные алгоритмы вообще не известны. Иногда в таких случаях все же можно предложить последовательность действий, которая, «если повезет», быстро приводит к требуемому результату. Существует класс так называемых вероятностных алгоритмов, которые дают правильный результат, но имеют вероятностную оценку времени работы. Обычно работа этих алгоритмов зависит от одного или нескольких параметров. В худшем случае они работают достаточно долго. Но удачный выбор параметра определяет быстрое завершение работы. Такие алгоритмы, если множество «хороших» значений параметров велико, на практике работают достаточно эффективно, хотя и не имеют хороших оценок сложности.

Мы будем иногда использовать слова детерминированный алгоритм, чтобы отличать алгоритмы в обычном смысле от вероятностных алгоритмов.

Как пример, рассмотрим вероятностный алгоритм, позволяющий эффективно находить решения полиномиальных сравнений по простому модулю. Пусть p — простое число, которое предполагается большим, и $f(x) \in \mathbb{Z}[x]$ — многочлен, степень которого предполагается ограниченной. Задача состоит в отыскании решений сравнения

$$f(x) \equiv 0 \pmod{p}. \tag{8}$$

Например, речь может идти о решении квадратичных сравнений, если степень многочлена $f(x)$ равна 2. Другими словами, мы должны отыскать в поле $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$ все элементы, удовлетворяющие уравнению $f(x) = 0$.

Согласно малой теореме Ферма, все элементы поля $\mathbb{F}_p$ являются однократными корнями многочлена $x^p - x$. Поэтому, вычислив наибольший общий делитель $d(x) = (x^p - x, f(x))$, мы найдем многочлен $d(x)$, множество корней которого в поле $\mathbb{F}_p$ совпадает с множеством корней многочлена $f(x)$, причем все эти корни однократны. Если окажется, что многочлен $d(x)$ имеет нулевую степень, т. е. лежит в поле $\mathbb{F}_p$, это будет означать, что сравнение (8) не имеет решений.

Для вычисления многочлена $d(x)$ удобно сначала вычислить многочлен $c(x) \equiv x^p \pmod{f(x)}$, пользуясь алгоритмом, подобным описанному выше алгоритму возведения в степень (напомним, что число p предполагается большим). А затем с помощью аналога алгоритма Евклида вычислить $d(x) = (c(x) - x, f(x))$. Все это выполняется за полиномиальное количество арифметических операций.

Таким образом, обсуждая далее задачу нахождения решений сравнения (8), мы можем предполагать, что в кольце многочленов $\mathbb{F}_p[x]$ справедливо равенство

$$f(x) = (x - a_1) \cdot \ldots \cdot (x - a_n), \quad a_i \in \mathbb{F}_p, a_i \neq a_j.$$

4. Алгоритм нахождения делителей многочлена $f(x)$ в кольце $\mathbb{F}_p[x]$

1. Выберем каким-либо способом элемент $\delta \in \mathbb{F}_p$.
2. Вычислим наибольший общий делитель
$$g(x) = (f(x), (x + \delta)^{\frac{p-1}{2}} - 1).$$
3. Если многочлен $g(x)$ окажется собственным делителем $f(x)$, то многочлен $f(x)$ распадется на два множителя и с каждым из них независимо нужно будет проделать все операции, предписываемые настоящим алгоритмом для многочлена $f(x)$.
4. Если окажется, что $g(x) = 1$ или $g(x) = f(x)$, следует перейти к шагу 1 и, выбрав новое значение δ, продолжить выполнение алгоритма.

Количество операций на шаге 2 оценивается величиной $O(\ln p)$, если вычисления проводить так, как это указывалось выше при нахождении $d(x)$. Выясним теперь, сколь долго придется выбирать числа δ, пока на шаге 2 не будет найден собственный делитель $f(x)$.

Количество решений уравнения $(t + a_1)^{\frac{p-1}{2}} = (t + a_2)^{\frac{p-1}{2}}$ в поле $\mathbb{F}_p$ не превосходит $\frac{p-3}{2}$. Это означает, что подмножество $D \subset \mathbb{F}_p$, состоящее из элементов δ, удовлетворяющих условиям

$$(\delta + a_1)^{\frac{p-1}{2}} \neq (\delta + a_2)^{\frac{p-1}{2}}, \quad \delta \neq -a_1,\ \delta \neq -a_2,$$

состоит не менее, чем из $\frac{p-1}{2}$ элементов. Учитывая теперь, что каждый ненулевой элемент $b \in \mathbb{F}_p$ удовлетворяет одному из равенств $b^{\frac{p-1}{2}} = 1$,

либо $b^{\frac{p-1}{2}} = -1$, заключаем, что для $\delta \in D$ одно из чисел a_1, a_2 будет корнем многочлена $(x+\delta)^{\frac{p-1}{2}} - 1$, а другое — нет. Для таких элементов δ многочлен $g(x)$, определенный на шаге 2 алгоритма, будет собственным делителем многочлена $f(x)$.

Итак, существует не менее $\frac{p-1}{2}$ «удачных» выборов элемента δ, при которых на шаге 2 алгоритма многочлен $f(x)$ распадется на два собственных множителя. Следовательно, при «случайном» выборе элемента $\delta \in \mathbb{F}_p$, вероятность того, что многочлен не разложится на множители после k повторений шагов алгоритма 1–4, не превосходит 2^{-k}. Вероятность с ростом k убывает очень быстро. И действительно, на практике этот алгоритм работает достаточно эффективно.

Заметим, что при оценке вероятности мы использовали только два корня многочлена $f(x)$. При $n > 2$ эта вероятность, конечно, еще меньше. Более тонкий анализ с использованием оценок А. Вейля для сумм характеров показывает, что вероятность для многочлена $f(x)$ не распасться на множители при однократном проходе шагов алгоритма 1–4, не превосходит $2^{-n} + O(p^{-1/2})$. Здесь постоянная в $O(\cdot)$ зависит от n. Детали доказательства см. в [24]. В настоящее время известно элементарное доказательство оценки А. Вейля (см. [9]).

В книге [6] описывается принадлежащий Берлекэмпу детерминированный алгоритм решения сравнения (8), требующий $O(pn^3)$ арифметических операций. Он практически бесполезен при больших p, а вот при маленьких p и не очень больших n он работает не очень долго.

Если в сравнении (8) заменить простой модуль p составным модулем m, то задача нахождения решений соответствующего сравнения становится намного более сложной. Известные алгоритмы ее решения основаны на сведении сравнения к совокупности сравнений (8) по простым модулям — делителям m, и, следовательно, они требуют разложения числа m на простые сомножители, что, как уже указывалось, является достаточно трудоемкой задачей.

4. Как отличить составное число от простого

Существует довольно эффективный способ убедиться, что заданное число является составным, не разлагая это число на множители. Согласно малой теореме Ферма, если число N простое, то для любого целого a, не делящегося на N, выполняется сравнение

$$a^{N-1} \equiv 1 \pmod{N}. \tag{9}$$

Если же при каком-то a это сравнение нарушается, можно утверждать, что N — составное. Проверка (9) не требует больших вычислений, это следует из алгоритма 1. Вопрос только в том, как найти для составного N целое число a, не удовлетворяющее (9). Можно, например, пы-

таться найти необходимое число a, испытывая все целые числа подряд, начиная с 2. Или попробовать выбирать эти числа случайным образом на отрезке $1 < a < N$.

К сожалению, такой подход не всегда дает то, что хотелось бы. Имеются составные числа N, обладающие свойством (9) для любого целого a с условием $(a, N) = 1$. Такие числа называются числами Кармайкла. Рассмотрим, например, число $561 = 3 \cdot 11 \cdot 17$. Так как 560 делится на каждое из чисел 2, 10, 16, то с помощью малой теоремы Ферма легко проверить, что 561 есть число Кармайкла. Можно доказать (Carmichael, 1912), что любое из чисел Кармайкла имеет вид $N = p_1 \cdot \ldots \cdot p_r$, $r \geqslant 3$, где все простые p_i различны, причем $N - 1$ делится на каждую разность $p_i - 1$. Лишь недавно, см. [10], была решена проблема о бесконечности множества таких чисел.

В 1976 г. Миллер предложил заменить проверку (9) проверкой несколько иного условия. Детали последующего изложения можно найти в [8]. Если N — простое число, $N - 1 = 2^s \cdot t$, где t нечетно, то согласно малой теореме Ферма для каждого a с условием $(a, N) = 1$ хотя бы одна из скобок в произведении

$$(a^t - 1)(a^t + 1)(a^{2t} + 1) \cdot \ldots \cdot (a^{2^{s-1}t} + 1) = a^{N-1} - 1$$

делится на N. Обращение этого свойства можно использовать, чтобы отличать составные числа от простых.

Пусть N — нечетное составное число, $N - 1 = 2^s \cdot t$, где t нечетно. Назовем целое число a, $1 < a < N$, «хорошим» для N, если нарушается одно из двух условий:

α) N не делится на a;

β) $a^t \equiv 1 \pmod N$ или существует целое k, $0 \leqslant k < s$, такое, что

$$a^{2^k t} \equiv -1 \pmod N.$$

Из сказанного ранее следует, что для простого числа N не существует хороших чисел a. Если же N составное число, то, как доказал Рабин, их существует не менее $\frac{3}{4}(N - 1)$.

Теперь можно построить вероятностный алгоритм, отличающий составные числа от простых.

5. Алгоритм, доказывающий непростоту числа

1. Выберем случайным образом число a, $1 < a < N$, и проверим для этого числа указанные выше свойства α) и β).
2. Если хотя бы одно из них нарушается, то число N составное.
3. Если выполнены оба условия α) и β), возвращаемся к шагу 1.

Из сказанного выше следует, что составное число не будет определено как составное после однократного выполнения шагов 1–3 с вероятностью не большей 4^{-1}. А вероятность не определить его после k повторений не превосходит 4^{-k}, т. е. убывает очень быстро.

Миллер предложил детерминированный алгоритм определения составных чисел, имеющий сложность $O(\ln^3 N)$, однако справедливость его результата зависит от недоказанной в настоящее время расширенной гипотезы Римана. Согласно этому алгоритму достаточно проверить условия α) и β) для всех целых чисел a, $2 \leqslant a \leqslant 70 \ln^2 N$. Если при каком-нибудь a из указанного промежутка нарушается одно из условий α) или β), число N составное. В противном случае оно будет простым или степенью простого числа. Последняя возможность, конечно, легко проверяется.

Напомним некоторые понятия, см. [4], необходимые для формулировки расширенной гипотезы Римана. Они понадобятся нам и в дальнейшем. Пусть $m \geqslant 2$ — целое число. Функция $\chi : \mathbb{Z} \to \mathbb{C}$ называется характером Дирихле по модулю m, или просто характером, если эта функция периодична с периодом m, отлична от нуля только на числах, взаимно простых с m, и мультипликативна, т. е. для любых целых u, v выполняется равенство $\chi(uv) = \chi(u)\chi(v)$. Для каждого m существует ровно $\varphi(m)$ характеров Дирихле. Они образуют группу по умножению. Единичным элементом этой группы является так называемый главный характер χ_0, равный 1 на всех числах, взаимно простых с m, и 0 на остальных целых числах. Порядком характера называется его порядок как элемента мультипликативной группы характеров.

С каждым характером связана так называемая L-функция Дирихле — функция комплексного переменного s, определенная рядом $L(s,\chi) = \sum\limits_{n=1}^{\infty} \dfrac{\chi(n)}{n^s}$. Сумма этого ряда аналитична в области $\operatorname{Re} s > 1$ и может быть аналитически продолжена на всю комплексную плоскость. Следующее соотношение $L(s,\chi_0) = \zeta(s) \prod\limits_{p|m} (1 - p^{-s})$ связывает L-функцию, отвечающую главному характеру, с дзета-функцией Римана $\zeta(s) = \sum\limits_{n=0}^{\infty} \dfrac{1}{n^s}$. Расширенная гипотеза Римана утверждает, что комплексные нули всех L-функций Дирихле, расположенные в полосе $0 < \operatorname{Re} s < 1$, лежат на прямой $\operatorname{Re} s = \dfrac{1}{2}$. В настоящее время не доказана даже простейшая форма этой гипотезы — классическая гипотеза Римана, утверждающая такой же факт о нулях дзета-функции.

В 1952 г. Анкени с помощью расширенной гипотезы Римана доказал, что для каждого простого числа q существует квадратичный невычет a, удовлетворяющий неравенствам $2 \leqslant a \leqslant 70 \ln^2 q$. Константа 70 была сосчитана позднее. Именно это утверждение и лежит в основе алгоритма Миллера. В 1957 г. Берджесс доказал существование такого

невычета без использования расширенной гипотезы Римана, но с худшей оценкой $2 \leqslant a \leqslant q^{\frac{1}{4\sqrt{e}}} + \varepsilon$, справедливой при любом положительном ε и q, большем некоторой границы, зависящей от ε.

Алгоритм Миллера принципиально отличается от алгоритма 5, так как полученное с его помощью утверждение о том, что число N — составное, опирается на недоказанную расширенную гипотезу Римана и потому может быть неверным. В то время как вероятностный алгоритм 5 дает совершенно правильный ответ для составных чисел. Несмотря на отсутствие оценок сложности, на практике он работает вполне удовлетворительно.

5. Как строить большие простые числа

Мы не будем описывать здесь историю этой задачи, рекомендуем обратиться к книге [6] и обзорам [8, 9]. Конечно же, большие простые числа можно строить сравнительно быстро. При этом можно обеспечить их случайное распределение в заданном диапазоне величин. В противном случае теряла бы всякий практический смысл система шифрования RSA. Наиболее эффективным средством построения простых чисел является несколько модифицированная малая теорема Ферма.

Теорема 4. *Пусть N, S — нечетные натуральные числа, $N-1 = S \cdot R$, причем для каждого простого делителя q числа S существует целое число a такое, что*

$$a^{N-1} \equiv 1 \pmod{N}, \quad (a^{\frac{N-1}{q}} - 1, N) = 1. \tag{10}$$

Тогда каждый простой делитель p числа N удовлетворяет сравнению

$$p \equiv 1 \pmod{2S}.$$

Доказательство. Пусть p — простой делитель числа N, а q — некоторый делитель S. Из условий (10) следует, что в поле вычетов $\mathbb{F}_p$ справедливы соотношения

$$a^{N-1} = 1, \ a^{\frac{N-1}{q}} \neq 1, \ a^{p-1} = 1. \tag{11}$$

Обозначим буквой r порядок элемента a в мультипликативной группе поля $\mathbb{F}_p$. Первые два из соотношений (11) означают, что q входит в разложение на простые множители числа r в степени такой же, как и в разложение $N-1$, а последнее — что $p-1$ делится на r. Таким образом, каждый простой делитель числа S входит в разложение $p-1$ в степени не меньшей, чем в S, так что $p-1$ делится на S. Кроме того, $p-1$ четно. Теорема 2 доказана.

Следствие. *Если выполнены условия теоремы 2 и $R \leqslant 4S+2$, то N — простое число.*

Действительно, пусть N равняется произведению не менее двух простых чисел. Каждое из них, согласно утверждению теоремы 2, не меньше, чем $2S+1$. Но тогда $(2S+1)^2 \leqslant N = SR+1 \leqslant 4S^2+2S+1$. Противоречие и доказывает следствие.

Покажем теперь, как с помощью последнего утверждения, имея большое простое число S, можно построить существенно большее простое число N. Выберем для этого случайным образом четное число R на промежутке $S \leqslant R \leqslant 4S+2$ и положим $N = SR+1$. Затем проверим число N на отсутствие малых простых делителей, разделив его на малые простые числа; испытаем N некоторое количество раз с помощью алгоритма 5. Если при этом выяснится, что N — составное число, следует выбрать новое значение R и опять повторить вычисления. Так следует делать до тех пор, пока не будет найдено число N, выдержавшее испытания алгоритмом 5 достаточно много раз. В этом случае появляется надежда на то, что N — простое число, и следует попытаться доказать простоту с помощью тестов теоремы 2.

Для этого можно случайным образом выбирать число a, $1 < a < N$, и проверять для него выполнимость соотношений

$$a^{N-1} \equiv 1 \pmod N, \quad (a^R - 1, N) = 1. \tag{12}$$

Если при выбранном a эти соотношения выполняются, то, согласно следствию из теоремы 2, можно утверждать, что число N простое. Если же эти условия нарушаются, нужно выбрать другое значение a и повторять эти операции до тех пор, пока такое число не будет обнаружено.

Предположим, что построенное число N действительно является простым. Зададимся вопросом, сколь долго придется перебирать числа a, пока не будет найдено такое, для которого будут выполнены условия (12). Заметим, что для простого числа N первое условие (12), согласно малой теореме Ферма, будет выполняться всегда. Те же числа a, для которых нарушается второе условие (12), удовлетворяют сравнению $a^R \equiv 1 \pmod N$. Как известно, уравнение $x^R = 1$ в поле вычетов $\mathbb{F}_N$ имеет не более R решений. Одно из них $x = 1$. Поэтому на промежутке $1 < a < N$ имеется не более $R-1$ чисел, для которых не выполняются условия (12). Это означает, что, выбирая случайным образом числа a на промежутке $1 < a < N$, при простом N можно с вероятностью большей, чем $1 - O(S^{-1})$, найти число a, для которого будут выполнены условия теоремы 2, и тем доказать, что N действительно является простым числом.

Заметим, что построенное таким способом простое число N будет удовлетворять неравенству $N > S^2$, т. е. будет записываться вдвое большим количеством цифр, чем исходное простое число S. Заменив теперь число S на найденное простое число N и повторив с этим но-

вым S все указанные выше действия, можно построить еще большее простое число. Начав с какого-нибудь простого числа, скажем, записанного 10 десятичными цифрами (простоту его можно проверить, например, делением на маленькие табличные простые числа), и повторив указанную процедуру достаточное число раз, можно построить простые числа нужной величины.

Обсудим теперь некоторые теоретические вопросы, возникающие в связи с нахождением простых чисел вида $N = SR + 1$, где числа R и S удовлетворяют неравенствам $S \leqslant R \leqslant 4S + 2$. Прежде всего, согласно теореме Дирихле, доказанной еще в 1839 г., прогрессия $2Sn + 1$, $n = 1, 2, 3, \ldots$ содержит бесконечное количество простых чисел. Нас интересуют простые числа, лежащие недалеко от начала прогрессии. Оценка наименьшего простого числа в арифметической прогрессии была получена в 1944 г. Ю. В. Линником. Соответствующая теорема утверждает, что наименьшее простое число в арифметической прогрессии $2Sn+1$ не превосходит S^C, где C — некоторая достаточно большая абсолютная постоянная. В предположении справедливости расширенной гипотезы Римана можно доказать, [11, стр. 272], что наименьшее такое простое число не превосходит $c(\varepsilon) \cdot S^{2+\varepsilon}$ при любом положительном ε.

Таким образом, в настоящее время никаких теоретических гарантий для существования простого числа $N = SR + 1$, $S \leqslant R \leqslant 4S + 2$ не существует. Тем не менее, опыт вычислений на ЭВМ показывает, что простые числа в арифметической прогрессии встречаются достаточно близко к ее началу. Упомянем в этой связи гипотезу о существовании бесконечного количества простых чисел q с условием, что число $2q + 1$ также простое, т. е. простым является уже первый член прогрессии.

Очень важен в связи с описываемым методом построения простых чисел также вопрос о расстоянии между соседними простыми числами в арифметической прогрессии. Ведь убедившись, что при некотором R число $N = SR+1$ составное, можно следующее значение R взять равным $R+2$ и действовать так далее, пока не будет найдено простое число N. И если расстояние между соседними простыми числами в прогрессии велико, нет надежды быстро построить нужное число N. Перебор чисел R до того момента, как мы наткнемся на простое число N окажется слишком долгим. В более простом вопросе о расстоянии между соседними простыми числами p_n и p_{n+1} в натуральном ряде доказано лишь, что $p_{n+1} - p_n = O\left(p_n^{\frac{38}{61}+\varepsilon}\right)$, что, конечно, не очень хорошо для наших целей. Вместе с тем существует так называемая гипотеза Крамера (1936 г.), что $p_{n+1} - p_n = O(\ln^2 p_n)$, дающая вполне приемлемую оценку. Примерно такой же результат следует и из расширенной гипотезы Римана. Вычисления на ЭВМ показывают, что простые

числа в арифметических прогрессиях расположены достаточно плотно.

В качестве итога обсуждения в этом разделе подчеркнем следующее: если принять на веру, что наименьшее простое число, а также расстояние между соседними простыми числами в прогрессии $2Sn + 1$ при $S \leqslant n \leqslant 4S + 2$ оцениваются величиной $O(\ln^2 S)$, то описанная схема построения больших простых чисел имеет полиномиальную оценку сложности. Кроме того, несмотря на отсутствие теоретических оценок времени работы алгоритмов, отыскивающих простые числа в арифметических прогрессиях со сравнительно большой разностью, на практике эти алгоритмы работают вполне удовлетворительно. На обычном персональном компьютере без особых затрат времени строятся таким способом простые числа порядка 10^{300}.

Конечно, способ конструирования простых чисел для использования в схеме RSA должен быть массовым, а сами простые числа должны быть в каком-то смысле хорошо распределенными. Это вносит ряд дополнительных осложнений в работу алгоритмов. Впрочем, описанная схема допускает массу вариаций. Все эти вопросы рассматриваются в статье [12].

Наконец, отметим, что существуют методы построения больших простых чисел, использующие не только простые делители $N - 1$, но и делители чисел $N + 1$, $N^2 + 1$, $N^2 \pm N + 1$. В основе их лежит использование последовательностей целых чисел, удовлетворяющих линейным рекуррентным уравнениям различных порядков. Отметим, что последовательность a^n, члены которой присутствуют в формулировке малой теоремы Ферма, составляет решение рекуррентного уравнения первого порядка $u_{n+1} = au_n$, $u_0 = 1$.

6. Как проверить большое число на простоту

Есть некоторое отличие в постановках задач предыдущего и настоящего разделов. Когда мы строим простое число N, мы обладаем некоторой дополнительной информацией о нем, возникающей в процессе построения. Например, такой информацией является знание простых делителей числа $N - 1$. Эта информация иногда облегчает доказательство простоты N.

В этом разделе мы предполагаем лишь, что нам задано некоторое число N, например, выбранное случайным образом на каком-то промежутке, и требуется установить его простоту, или доказать, что оно является составным. Эту задачу за полиномиальное количество операций решает указанный в п. 4 алгоритм Миллера. Однако, справедливость полученного с его помощью утверждения зависит от недоказанной расширенной гипотезы Римана. Если число N выдержало ис-

пытания алгоритмом 5 для 100 различных значений параметра a, то, по-видимому, можно утверждать, что оно является простым с вероятностью большей, чем $1 - 4^{-100}$. Эта вероятность очень близка к единице, однако все же оставляет некоторую тень сомнения на простоте числа N. В дальнейшем в этом разделе мы будем считать, что заданное число N является простым, а нам требуется лишь доказать это.

В настоящее время известны детерминированные алгоритмы различной сложности для доказательства простоты чисел. Мы остановимся подробнее на одном из них, предложенном в 1983 г. в совместной работе Адлемана, Померанца и Рамели [13]. Для доказательства простоты или непростоты числа N этот алгоритм требует $(\ln N)^{c \ln\ln\ln N}$ арифметических операций. Здесь c — некоторая положительная абсолютная постоянная. Функция $\ln\ln\ln N$ хоть и медленно, но все же возрастает с ростом N, поэтому алгоритм не является полиномиальным. Но все же его практические реализации позволяют достаточно быстро тестировать числа на простоту. Существенные усовершенствования и упрощения в первоначальный вариант алгоритма были внесены в работах Х. Ленстры и А. Коена [14, 15]. Мы будем называть описываемый ниже алгоритм алгоритмом Адлемана – Ленстры.

В основе алгоритма лежит использование сравнений типа малой теоремы Ферма, но в кольцах целых чисел круговых полей, т. е. полей, порожденных над полем $\mathbb{Q}$ числами $\zeta_p = e^{2\pi i/p}$ — корнями из 1. Пусть q — простое нечетное число и c — первообразный корень по модулю q, т. е. образующий элемент мультипликативной группы поля $\mathbb{F}_q$, которая циклична. Для каждого целого числа x, не делящегося на q, можно определить его индекс, $\operatorname{ind}_q x \in \mathbb{Z}/(q-1)\mathbb{Z}$, называемый также *дискретным логарифмом*, с помощью сравнения $x \equiv c^{\operatorname{ind}_q x} \pmod q$. Рассмотрим далее два простых числа p, q с условием, что $q-1$ делится на p, но не делится на p^2.

Следующая функция, определенная на множестве целых чисел,

$$\chi(x) = \begin{cases} 0, \text{ если } & q|x, \\ \zeta_p^{\operatorname{ind}_q x}, \text{ если } & (x,q) = 1 \end{cases}$$

является характером по модулю q и порядок этого характера равен p. Сумма

$$\tau(\chi) = -\sum_{x=1}^{q-1} \chi(x)\zeta_q^x \in \mathbb{Z}[\zeta_p, \zeta_q]$$

называется суммой Гаусса. Формулируемая ниже теорема 3 представляет собой аналог малой теоремы Ферма, используемый в алгоритме Адлемана – Ленстры.

Теорема 5. *Пусть N — нечетное простое число, $(N, pq) = 1$. Тогда в кольце $\mathbb{Z}[\zeta_p, \zeta_q]$ выполняется сравнение*

$$\tau(\chi)^N \equiv \chi(N)^{-N} \cdot \tau(\chi^N) \pmod{NZ[\zeta_p, \zeta_q]}.$$

Если при каких-либо числах p, q сравнение из теоремы 3 нарушается, можно утверждать, что N составное число. В противном случае, если сравнение выполняется, оно дает некоторую информацию о возможных простых делителях числа N. Собрав такую информацию для различных p, q, в конце концов удается установить, что N имеет лишь один простой делитель и является простым.

В случае $p = 2$ легко проверить, что сравнение из теоремы 3 равносильно хорошо известному в элементарной теории чисел сравнению

$$q^{\frac{N-1}{2}} \equiv \left(\frac{q}{N}\right) \pmod{N}, \tag{13}$$

где $\left(\frac{q}{N}\right)$ — так называемый символ Якоби. Хорошо известно также, что последнее сравнение выполняется не только для простых q, но и для любых целых q, взаимно простых с N. Заметим также, что для вычисления символа Якоби существует быстрый алгоритм, основанный на законе взаимности Гаусса и, в некотором смысле, подобный алгоритму Евклида вычисления наибольшего общего делителя. Следующий пример показывает, каким образом выполнимость нескольких сравнений типа (13) дает некоторую информацию о возможных простых делителях числа N.

Пример 1 (Х. Ленстра). Пусть N — натуральное число, $(N, 6) = 1$, для которого выполнены сравнения

$$a^{\frac{N-1}{2}} \equiv \left(\frac{a}{N}\right) \pmod{N}, \quad \text{при } a = -1, 2, 3, \tag{14}$$

а кроме того с некоторым целым числом b имеем

$$b^{\frac{N-1}{2}} \equiv -1 \pmod{N}. \tag{15}$$

Как уже указывалось, при простом N сравнения (14) выполняются для любого a, взаимно простого с N, а сравнение (15) означает, что b есть первообразный корень по модулю N. Количество первообразных корней равно $\varphi(N-1)$, т. е. достаточно велико. Таким образом, число b с условием (15) при простом N может быть найдено достаточно быстро с помощью случайного выбора и последующей проверки (15).

Докажем, что из выполнимости (14–15) следует, что каждый делитель r числа N удовлетворяет одному из сравнений

$$r \equiv 1 \pmod{24} \text{ или } r \equiv N \pmod{24}. \tag{16}$$

Не уменьшая общности, можно считать, что r — простое число. Введем теперь обозначения $N - 1 = u \cdot 2^k$, $r - 1 = v \cdot 2^m$, где u и v — нечетные

числа. Из (15) и сравнения $b^{r-1} \equiv 1 \pmod r$ следует, что $m \geqslant k$. Далее, согласно (14), выполняются следующие сравнения

$$\left(\frac{a}{N}\right) = \left(\frac{a}{N}\right)^v \equiv a^{uv2^{k-1}} \pmod r, \qquad \left(\frac{a}{r}\right) = \left(\frac{a}{r}\right)^u \equiv a^{uv2^{m-1}} \pmod r,$$

означающие (в силу того, что символ Якоби может равняться лишь -1 или $+1$), что

$$\left(\frac{a}{N}\right)^{2^{m-k}} = \left(\frac{a}{r}\right).$$

При $m > k$ это равенство означает, что $\left(\frac{a}{r}\right) = 1$ при $a = -1, 2, 3$, и, следовательно, $r \equiv 1 \pmod{24}$. Если же $m = k$, то имеем $\left(\frac{a}{N}\right) = \left(\frac{a}{r}\right)$ и $r \equiv N \pmod{24}$. Этим (16) доказано.

Информация такого рода получается и в случае произвольных простых чисел p и q с указанными выше свойствами.

Опишем (очень грубо) схему алгоритма Адлемана – Ленстры для проверки простоты N:

1) выбираются различные простые числа $p_1, \ldots, p_k$ и различные простые нечетные $q_1, \ldots, q_s$ такие, что

а) для каждого j все простые делители числа $q_j - 1$ содержатся среди $p_1, \ldots, p_k$ и $q_j - 1$ не делятся на квадрат простого числа,

б) $S = 2q_1 \cdot \ldots \cdot q_s > \sqrt{N}$;

2) для каждой пары выбранных чисел p, q проводятся тесты, подобные сравнению из теоремы 3. Если N не удовлетворяет какому-либо из этих тестов — оно составное. В противном случае

3) определяется не очень большое множество чисел, с которыми только и могут быть сравнимы простые делители N. А именно, каждый простой делитель r числа N должен удовлетворять сравнению вида

$$r \equiv N^j \pmod S, \quad 0 \leqslant j < T = p_1 \cdot \ldots \cdot p_k;$$

4) проверяется, содержит ли найденное множество делители N. Если при этом делители не обнаружены, утверждается, что N — простое число.

Если число N составное, оно обязательно имеет простой делитель r, меньший $\sqrt{N} < S$, который сам содержится среди возможных остатков. Именно на этом свойстве основано применение пункта 4) алгоритма.

Пример 2. Если выбрать следующие множества простых чисел

$$\{p\} = \{2, 3, 5, 7\} \text{ и } \{q\} = \{3, 7, 11, 31, 43, 71, 211\},$$

то таким способом удается проверять простоту чисел $N < 8{,}5 \cdot 10^{19}$.

Отметим, что в работе [13] для тестирования использовались не сравнения теоремы 3, а закон взаимности для степенных вычетов и так называемые суммы Якоби. Сумма Якоби

$$J(\chi_1, \chi_2) = -\sum_{x=2}^{q-1} \chi_1(x)\chi_2(1-x)$$

определяется для двух характеров χ_1, χ_2 по модулю q. Если характеры имеют порядок p, то соответствующая сумма Якоби принадлежит кольцу $\mathbb{Z}[\zeta_p]$. Поскольку числа p, участвующие в алгоритме, сравнительно невелики, то вычисления с суммами Якоби производятся в полях существенно меньшей степени, чем вычисления с суммами Гаусса. Это главная причина, по которой суммы Якоби предпочтительнее для вычислений. При $\chi_1\chi_2 \neq \chi_0$ выполняется классическое соотношение

$$J(\chi_1, \chi_2) = \frac{\tau(\chi_1) \cdot \tau(\chi_2)}{\tau(\chi_1 \cdot \chi_2)},$$

связывающее суммы Гаусса с суммами Якоби и позволяющее переписать сравнение теоремы 3 в терминах сумм Якоби (см. [16]). Так, при $p = 3$ и $q = 7$ соответствующее сравнение, справедливое для простых N, отличных от 2, 3, 7, принимает вид

$$(-3\zeta - 2)^{\left[\frac{N}{3}\right]} \cdot (3\zeta + 1)^{\left[\frac{2N}{3}\right]} \equiv \xi \pmod{N\mathbb{Z}[\zeta]},$$

где $\zeta = e^{2\pi i/3}$ и ξ — некоторый корень кубический из 1.

В 1984 г. в работе [15] было внесено существенное усовершенствование в алгоритм, позволившее освободиться от требования неделимости чисел $q-1$ на квадраты простых чисел. В результате, например, выбрав число $T = 2^4 \cdot 3^2 \cdot 5 \cdot 7 = 5040$ и взяв S равным произведению простых чисел q с условием, что T делится на $q-1$, получим $S > 1{,}5 \cdot 10^{52}$, что позволяет доказывать простоту чисел N, записываемых сотней десятичных знаков. При этом вычисления будут проводиться в полях, порожденных корнями из 1 степеней 16, 9, 5 и 7.

Мой персональный компьютер с процессором Pentium-150, пользуясь реализацией этого алгоритма на языке UBASIC, доказал простоту записываемого 65 десятичными знаками, большего из простых чисел в примере Ривеста, Шамира и Адлемана (см. раздел 1) за 8 секунд. Сравнение этих 8 секунд и 17 лет, потребовавшихся для разложения на множители предложенного в примере числа, конечно, впечатляет.

Отметим, что оценка сложности этого алгоритма представляет собой трудную задачу аналитической теории чисел. Как уже указывалось, количество операций оценивается величиной $(\ln N)^{c \ln\ln\ln N}$. Однако соответствующие числа S и T, возникающие в процессе доказательства, не могут быть явно указаны в зависимости от N. Доказано лишь

существование чисел S и T, для которых достигается оценка. Впрочем, есть вероятностный вариант алгоритма, доказывающий простоту простого числа N с вероятностью большей $1 - 2^{-k}$ за $O(k(\ln N)^{c \ln\ln\ln N})$ арифметических операций. А в предположении расширенной гипотезы Римана эта оценка сложности может быть получена при эффективно указанных S и T.

7. Как раскладывают составные числа на множители

Мы лишь кратко коснемся этой темы, отсылая читателей к книгам [6, 16, 17]. Среди многих алгоритмов разложения мы выберем ту линию развития, которая привела к разложению числа, предложенного RSA.

Поиском эффективных способов разложения целых чисел на множители занимаются уже очень давно. Эта задача интересовала выдающихся ученых в области теории чисел. Вероятно Ферма был первый, кто предложил представить разлагаемое число N в виде разности квадратов $N = x^2 - y^2$, а затем, вычисляя $(N, x - y)$, попытаться найти нетривиальный делитель N. Он же предложил и способ, позволяющий найти требуемое представление. Если разлагаемое число имеет два не очень отличающиеся по величине множителя, этот способ позволяет определить их быстрее, чем простой перебор делителей. Лежандр обратил внимание на то, что при таком подходе достаточно получить сравнение

$$x^2 \equiv y^2 \pmod{N}. \tag{17}$$

Конечно, не каждая пара чисел, удовлетворяющих ему, позволяет разложить N на множители. Эйлер и Гаусс предложили некоторые способы нахождения чисел, связанных соотношением (17). Лежандр использовал для этой цели непрерывные дроби.

Напомним, что каждому иррациональному числу ξ может быть поставлена в соответствие бесконечная последовательность целых чисел $[b_0;\, b_1,\, b_2, \ldots]$, называемая его непрерывной дробью. Это сопоставление строится следующим образом

$$x_0 = \xi,\ b_i = [x_i],\ x_{i+1} = \frac{1}{x_i - b_i},\ i = 0,\, 1,\, 2, \ldots.$$

Лежандр доказал, что непрерывная дробь квадратичной иррациональности периодична. Если раскладывать в непрерывную дробь число $\xi = \sqrt{N}$, то возникающие в процессе разложения числа x_i имеют вид $x_i = \dfrac{\sqrt{N} + P_i}{Q_i}$ с целыми P_i, Q_i, причем всегда $0 \leqslant P_i < \sqrt{N}$,

$0 < Q_i < 2\sqrt{N}$. С каждой непрерывной дробью можно связать последовательность рациональных чисел, так называемых подходящих дробей, $\dfrac{A_i}{B_i}$, $i \geqslant 0$, вычисляемых по правилам

$$A_{i+1} = b_{i+1}A_i + A_{i-1},\ B_{i+1} = b_{i+1}B_i + B_{i-1},\ i \geqslant 0,$$
$$A_0 = b_0,\ B_0 = A_{-1} = 1,\ B_{-1} = 0$$

и стремящихся к разлагаемому числу. Если в непрерывную дробь разлагается число $\xi = \sqrt{N}$, то справедливо соотношение

$$A_{i-1}^2 - NB_{i-1}^2 = (-1)^i Q_i, \tag{18}$$

из которого следует

$$A_{i-1}^2 \equiv (-1)^i Q_i \pmod N. \tag{19}$$

Заметим, что длина периода разложения в непрерывную дробь числа $\xi = \sqrt{N}$ может быть большой и достигать величин порядка $\sqrt{N}$.

В 1971 г. Шенкс предложил использовать сравнения (19) для конструирования чисел, удовлетворяющих (17). Если вычисления проводить до тех пор, пока при четном i не получится $Q_i = R^2$ при некотором целом R, то пара чисел $\langle A_{i-1}, R\rangle$ будет удовлетворять (17) и с ее помощью можно надеяться получить разложение N на простые множители.

В 1975 г. Моррисон и Бриллхарт стали перемножать сравнения (19) при различных i с тем, чтобы таким способом получить квадрат целого числа в правой части. Этот метод, метод непрерывных дробей, позволил впервые разложить на множители седьмое число Ферма $F_7 = 2^{128}+1$. Для реализации алгоритма выбирается так называемая база множителей $\{p_1, p_2, \ldots, p_s\}$. В нее входят ограниченные по величине некоторым параметром простые числа такие, что $\left(\dfrac{N}{p_i}\right) = 1$. Последнее условие связано с тем, что, согласно (18), в разложение на простые множители чисел Q_i могут входить лишь те простые, для которых N является квадратичным вычетом.

На первом этапе алгоритма каждое очередное число Q_i делится на все числа $p_1, p_2, \ldots, p_s$ и, если оно не разлагается полностью в произведение степеней этих простых, то отбрасывается. Иначе получается разложение

$$(-1)^i Q_i = (-1)^{a_0} \prod_{j=1}^{s} p_j^{a_j}. \tag{20}$$

Этому номеру i сопоставляется вектор $(a_0, a_1, \ldots, a_s)$ (вектор показателей). Затем вычисляется следующее значение Q_{i+1}, и с ним проделывается в точности такая же процедура.

Эти вычисления проводятся до тех пор, пока не будет построено $s+2$ вектора показателей. В получившейся матрице показателей, оче-

видно, можно подобрать вектора-строки так, что их сумма будет вектором с четными координатами $2(b_0, b_1, \ldots b_s)$. Если Δ — множество номеров векторов, вошедших в эту сумму, то, как легко проверить с помощью (19), имеет место сравнение

$$\left(\prod_{i\in\Delta} A_{i-1}\right)^2 \equiv \left(\prod_{j=1}^{s} p_j^{b_j}\right)^2 \pmod N.$$

Если с помощью этого сравнения не удается разложить N на множители, разложение в непрерывную дробь продолжается, продолжается набор векторов показателей и т. д.

В этот алгоритм был внесен ряд усовершенствований: вместо $\sqrt{N}$ можно раскладывать в непрерывную дробь число $\sqrt{kN}$, где маленький множитель k подбирается так, чтобы в базу множителей вошли все малые простые; была предложена так называемая стратегия раннего обрыва и т. д. Сложность этого алгоритма была оценена в 1982 г. величиной $O(\exp(\sqrt{1{,}5\cdot\ln N\cdot\ln\ln N}))$. При выводе этой оценки использовался ряд правдоподобных, но не доказанных гипотез о распределении простых чисел. Получившаяся в оценке функция растет медленнее любой степенной функции. Алгоритмы, сложность которых оценивается подобным образом, получили название субэкспоненциальных (в зависимости от $\ln N$).

В 1982 г. Померанцом был предложен еще один субэкспоненциальный алгоритм — алгоритм квадратичного решета. Его сложность оценивается такой же функцией, как и в методе непрерывных дробей, но вместо константы 1,5 получена лучшая — 9/8. Обозначим $m = \left[\sqrt{N}\right]$, $Q(x) = (x+m)^2 - N$ и выберем ту же базу множителей, что и в методе непрерывных дробей. При малых целых значениях x величина $Q(x)$ будет сравнительно невелика. Следующий шаг объясняет название алгоритма — квадратичное решето. Вместо того, чтобы перебирать числа x и раскладывать соответствующие значения $Q(x)$ на множители, алгоритм сразу отсеивает негодные значения x, оставляя лишь те, для которых $Q(x)$ имеет делители среди элементов базы множителей.

Задав некоторую границу B, для каждого простого числа p, входящего в базу множителей, и каждого показателя степени a, с условием $p^a \leqslant B$ находим решения x квадратичного сравнения $Q(x) \equiv 0 \pmod{p^a}$. Множество решений обозначим буквой Λ. Итак, для каждого $x \in \Lambda$ найдется элемент базы множителей, а может быть и не один, входящий в некоторой степени в разложение на простые сомножители числа $Q(x)$. Те числа x, при которых значения $Q(x)$ оказываются полностью разложенными, дают нам вектор показателей, как и в алгоритме непрерывных дробей. Если таких векторов окажется достаточно много,

с ними можно проделать те же операции, что и в алгоритме непрерывных дробей.

Мы кратко описали здесь лишь основную идею алгоритма. Помимо этого, используется много других дополнительных соображений и различных технических приемов. Например, аналог соотношения (20) имеет вид

$$Q(x) = q_1 q_2 (-1)^{a_0} \prod_{j=1}^{s} p_j^{a_j} \pmod N. \tag{21}$$

В нем допускается наличие двух дополнительных больших простых множителей $B_1 < q_i < B_2$. Эти множители впоследствии при перемножении значений $Q(x)$ исключаются.

Некоторые детали реализации алгоритма можно найти в работе [6]. Отметим здесь только, что на множители раскладывалось число $5N$, база множителей состояла из -1 и 524338 простых чисел, меньших, чем $B_1 = 16333609$. При этом было использовано $B_2 = 2^{30}$. В результате просеивания получилось 112011 соотношений вида (21) без множителей q_i, 1431337 соотношений с одним таким множителем и 6881138 соотношений с двумя множителями. Именно на поиск всех этих соотношений понадобились 220 дней и большое количество работавших параллельно компьютеров. На втором шаге алгоритма, когда из соотношений (21) комбинировались четные векторы показателей степеней, приходилось работать с матрицами, размеры которых измерялись сотнями тысяч битов. Этот второй шаг потребовал 45 часов работы. Уже четвертый вектор с четными показателями привел к искомому разложению на множители.

8. Дискретное логарифмирование

Пусть p — нечетное простое число. Еще Эйлер знал, что мультипликативная группа кольца $\mathbb{Z}/p\mathbb{Z}$ циклична, т. е. существуют такие целые числа a, что сравнение

$$a^x \equiv b \pmod p \tag{22}$$

разрешимо относительно x при любом $b \in \mathbb{Z}$, не делящемся на p. Числа a с этим свойством называются первообразными корнями, и количество их равно $\varphi(p-1)$, где φ — функция Эйлера. Целое x, удовлетворяющее сравнению (22), называется индексом или дискретным логарифмом числа b.

В параграфе 2 мы описали алгоритм, позволяющий по заданному числу x достаточно быстро вычислять $a^x \pmod p$. Обратная же операция — вычисление по заданному b его дискретного логарифма, вообще говоря, является очень сложной в вычислительном отношении задачей.

Именно это свойство дискретного логарифма и используется в его многочисленных криптографических применениях (см. главу 1). Наиболее быстрые (из известных) алгоритмы решения этой задачи, основанные на так называемом методе решета числового поля, требуют выполнения $\exp(c(\ln p)^{1/3}(\ln\ln p)^{2/3})$ арифметических операций (см. [25]), где c — некоторая положительная постоянная. Это сравнимо со сложностью наиболее быстрых алгоритмов разложения чисел на множители. Конечно, указанная оценка сложности получена при условии справедливости ряда достаточно правдоподобных гипотез.

Говоря о сложности задачи дискретного логарифмирования, мы имели в виду «общий случай». Ведь и большое целое число легко может быть разложено на простые сомножители, если все эти сомножители не очень велики. Известен алгоритм, позволяющий быстро решать задачу дискретного логарифмирования, если $p-1$ есть произведение малых простых чисел.

Пусть q — простое число, делящее $p-1$. Обозначим $c \equiv a^{\frac{p-1}{q}} \pmod p$, тогда классы вычетов $1, c, c^2, \ldots, c^{q-1}$ все различны и образуют полное множество решений уравнения $x^q = 1$ в поле $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$. Если q не велико и целое число d удовлетворяет сравнению $x^q \equiv 1 \pmod p$, то показатель k, $0 \leqslant k < q$, для которого выполняется $d \equiv c^k \pmod p$, легко может быть найден, например, с помощью перебора. Именно на этом свойстве основан упомянутый выше алгоритм.

Допустим, что $p-1 = q^k h$, $(q,h) = 1$. Алгоритм последовательно строит целые числа u_j, $j = 0, 1, \ldots, k$, для которых выполняется сравнение

$$\left(b^h a^{-hu_j}\right)^{q^{k-j}} \equiv 1 \pmod p. \tag{23}$$

Так как выполняется сравнение $b^{hq^k} \equiv 1 \pmod p$, то найдется целое число u_0, для которого $b^{hq^{q-1}} \equiv c^{u_0} \pmod p$. При таком выборе сравнение (23) с $j = 0$, очевидно, выполняется. Предположим, что найдено число u_j, удовлетворяющее сравнению (23). Тогда определим t с помощью сравнения

$$\left(b^h a^{-hu_j}\right)^{q^{k-j-1}} \equiv c^t \pmod p, \tag{24}$$

и положим $u_{j+1} = u_j + tq^j$. Имеют место сравнения

$$\left(b^h a^{-hu_{j+1}}\right)^{q^{k-j-1}} \equiv c^t a^{-thq^{k-1}} \equiv 1 \pmod p, \tag{25}$$

означающие справедливость (23) при $j+1$.

При $j = k$ сравнение (23) означает в силу (22), что $a^{(x-u_k)h} \equiv$ $\equiv 1 \pmod p$. Целое число a есть первообразный корень по модулю p, поэтому имеем $(x-u_k)h \equiv 0 \pmod{p-1}$ и

$$x \equiv u_k \pmod{q^k}.$$

Если $p-1 = q_1^{k_1} \cdot \ldots \cdot q_s^{k_s}$, где все простые числа q_j малы, то указанная процедура позволяет найти вычеты $x \pmod{q_i^{k_i}}$, $i = 1, \ldots, s$, и, с помощью китайской теоремы об остатках, вычет $x \pmod{p-1}$, т. е. решить сравнение (22).

В случае обычных логарифмов в поле действительных чисел имеется специальное основание $e = 2{,}171828\ldots$, позволяющее достаточно быстро вычислять логарифмы с произвольной точностью. Например, это можно делать с помощью быстро сходящегося ряда

$$\ln \frac{1+x}{1-x} = 2\left(x + \frac{x^3}{3} + \frac{x^5}{5} + \ldots\right), \ |x| < 1. \tag{26}$$

Логарифмы по произвольному основанию c могут быть вычислены с помощью тождества

$$\log_c x = \frac{\ln x}{\ln c}. \tag{27}$$

В случае дискретных логарифмов нет основания, по которому логарифмы вычислялись бы столь же быстро, как натуральные в поле действительных чисел. Вместе с тем, последняя формула, связывающая логарифмы с различными основаниями, остается справедливой и позволяет выбирать основание удобным способом. Единственное условие для этого состоит в том, чтобы логарифм нового основания $\operatorname{Log} c$ был взаимно прост с $p-1$. Тогда в формуле (27) возможно деление по модулю $p-1$. Заметим, что это условие будет выполнено, если и только если c — первообразный корень. Из расширенной гипотезы Римана следует, что наименьший первообразный корень по модулю p ограничен величиной $O(\log^6 p)$. Поэтому в дальнейшем для простоты изложения мы будем предполагать, что основание a в (22) невелико, именно $a = O(\log^6 p)$.

Так как поле $\mathbb{F}_p$ неполно, вычисление дискретных логарифмов не может использовать предельный переход и основано на иных принципах. Прежде всего, нужный дискретный логарифм $\operatorname{Log} b$ вычисляется не сам по себе, а вместе с совокупностью логарифмов ряда других чисел. Заметим, что всякое сравнение вида

$$q_1^{k_1} \cdot \ldots \cdot q_s^{k_s} \equiv q_1^{m_1} \cdot \ldots \cdot q_s^{m_s} \pmod{p}, \tag{28}$$

где $q_i, k_i, m_i \in \mathbb{Z}$, приводит к соотношению между логарифмами

$$(k_1 - m_1)\operatorname{Log} q_1 + \cdots + (k_s - m_s)\operatorname{Log} q_s \equiv 0 \pmod{p-1}. \tag{29}$$

А если выполняются сравнения

$$a \equiv q_1^{r_1} \cdot \ldots \cdot q_s^{r_s} \pmod{p-1} \qquad b \equiv q_1^{x_1} \cdot \ldots \cdot q_s^{x_s} \pmod{p},$$

то

$$r_1 \operatorname{Log} q_1 + \cdots + r_s \operatorname{Log} q_s \equiv 1 \pmod{p-1}, \tag{30}$$

и

$$\operatorname{Log} b \equiv x_1 \operatorname{Log} q_1 + \cdots + x_s \operatorname{Log} q_s \pmod{p-1}. \tag{31}$$

Имея достаточно много векторов $k_1, \ldots, k_s$, $m_1, \ldots, m_s$ с условием (28), можно найти решение соответствующей системы сравнений (29), (30). Если эта система имеет единственное решение, то им как раз и будет набор логарифмов $\operatorname{Log} q_1, \ldots, \operatorname{Log} q_s$. Затем с помощью (31) можно найти $\operatorname{Log} b$.

Мы опишем ниже реализацию этой идеи, взятую из работы [18]. Эвристические соображения позволили авторам [18] утверждать, что предложенный ими алгоритм требует $L^{1+\varepsilon}$, где $L = \exp(\sqrt{\ln p \cdot \ln \ln p})$, арифметических операций для вычисления $\operatorname{Log} b$.

Положим

$$H = [\sqrt{p}] + 1, \quad J = H^2 - q.$$

Тогда $0 < J < 2\sqrt{p} + 1$, и, как легко проверить, для любой пары целых чисел c_1, c_2 выполняется сравнение

$$(H + c_1)(H + c_2) \equiv J + (c_1 + c_2)H + c_1 c_2 \pmod p. \tag{32}$$

Если числа c_i не очень велики, скажем $c_i \leqslant L^{1/2+\varepsilon}$ при некотором $\varepsilon > 0$, то правая часть сравнения (32) не превосходит $p^{1/2+\varepsilon/2}$. Можно доказать, что случайно выбранное натуральное число $x < p^{1/2+\varepsilon/2}$ раскладывается в произведение простых чисел, меньших $L^{1/2}$, с вероятностью большей, чем $L^{-1/2-\varepsilon/2}$.

Обозначим через $S = \{q_1, \ldots, q_s\}$ совокупность всех простых чисел $q < L^{1/2}$, а также всех целых чисел вида $H + c$ при $0 < c < L^{1/2+\varepsilon}$. Тогда $s = O(L^{1/2+\varepsilon})$. Будем теперь перебирать случайным образом числа и для каждой такой пары пытаться разложить на множители соответствующее выражение из правой части (32). Для разложения можно воспользоваться, например, делением на все простые числа, меньшие чем $L^{1/2}$. Перебрав все $\frac{1}{2}(L^{1/2+\varepsilon})^2 = O(L^{1+2\varepsilon})$ указанных пар c_1, c_2, мы найдем, как это следует из указанных выше вероятностных соображений, не менее

$$L^{-1/2-\varepsilon/2} \cdot O(L^{1+2\varepsilon}) = O(L^{1/2+3\varepsilon/2}) \tag{33}$$

пар, для которых правая часть сравнения (32) полностью раскладывается на простые сомножители, меньшие $L^{1/2}$. Сравнение (32), таким образом, принимает вид (28). Так строится система уравнений типа (29).

Напомним, что число a, согласно нашему предположению, существенно меньше, чем $L^{1/2}$. Поэтому оно раскладывается в произведение простых чисел, входящих в множество $\{q_1, \ldots, q_s\}$, и это приводит к сравнению (30).

Заметим, что количество (33) найденных сравнений типа (29) превосходит число s. Следовательно, построенная система неоднородных линейных сравнений относительно $\operatorname{Log} q_i$ содержит сравнений больше,

чем неизвестных. Конечно, множество ее решений может при этом быть бесконечным. Одна из правдоподобных гипотез состоит в том, что система все- таки имеет единственное решение, и, решив ее, можно определить дискретные логарифмы всех чисел q_i. На этом завершается первый этап работы алгоритма из [18].

Как было отмечено, каждое из чисел, стоящих в правой части сравнения (32), не превосходит $p^{1/2+\varepsilon/2}$. Поэтому оно раскладывается в произведение не более $O(\ln p)$ простых сомножителей и, следовательно, каждое из сравнений (29) построенной системы содержит лишь $O(\ln p)$ отличных от нуля коэффициентов. Матрица системы сравнений будет разреженной, что позволяет применять для ее решения специальные методы с меньшей оценкой сложности, чем обычный гауссов метод исключения переменных.

Вместо перебора всех допустимых значений c_i в [18] предлагается использовать так называемое решето, отбрасывающее все пары этих чисел, для которых правая часть (32) заведомо не раскладывается в произведение малых простых сомножителей. Для каждого c_1 и каждой малой простой степени $q' < L^{1/2}$ можно найти все решения $c_2 < L^{1/2}$ линейного сравнения

$$J + (c_1 + c_2)H + c_1c_2 \equiv 0 \pmod{q'}.$$

Организованная правильным образом, эта процедура одновременно отбирает все нужные пары чисел c_1, c_2 и дает разложение на простые сомножители правых частей сравнений (32).

Итак, после первого этапа работы алгоритма в нашем распоряжении оказываются дискретные логарифмы всех чисел из множества S. Второй этап алгоритма сводит поиск дискретного логарифма числа b к поиску логарифмов некоторого множества чисел u, не превосходящих по величине L^2. Выбирая случайным образом число w не более $L^{1/4}$ раз, можно, как показывают вероятностные соображения, найти такое w, что вычет $a^w b \pmod p$ раскладывается в произведение простых чисел, меньших L^2. Пусть

$$a^w b \equiv \prod_{i=1}^{s} q_i^{y_i} \prod_{j=1}^{t} u_j^{z_j} \pmod p$$

такое разложение, где $u_1, \ldots, u_t$ — некоторые простые числа с условием $L^{1/2} < u < L^2$. На поиск этого сравнения потребуется $O(L^{1/2})$ арифметических операций. В результате вычисление дискретного логарифма числа b сводится к вычислению t дискретных логарифмов для чисел u_j, $1 \leqslant j \leqslant t$ среднего размера.

Наконец, на последнем этапе производится вычисление логарифмов всех чисел u_j. Пусть u — простое число из интервала $L^{1/2} < u < L^2$.

Обозначим

$$G = \left[\frac{\sqrt{p}}{u}\right], \quad I = HGu - p.$$

Для любых целых чисел $c_1, c_2 < L^{1/2+\varepsilon}$ выполняется сравнение

$$(H + c_1)(H + c_2)u \equiv I + (c_1G + c_2H + c_1c_2)u \pmod{p}. \qquad (34)$$

Отметим, что правая часть этого сравнения не превосходит $p^{1/2}L^{5/2+\varepsilon}$. Просеивая все числа c_1, c_2 из указанного интервала, можно найти такие, что числа $G + c_2$ и правая часть сравнения (34) состоят из простых сомножителей, не превосходящих $L^{1/2}$. Тогда сравнение (34) позволяет вычислить $\operatorname{Log} u$. Вычисление $\operatorname{Log} b$ при известных уже значениях $\operatorname{Log} q_i$ требует $L^{1/2+\varepsilon}$ арифметических операций.

Существуют и другие способы построения соотношений (28). В [23] для этого используются вычисления в полях алгебраических чисел. В качестве множителей в соотношениях типа (28) используются не только простые числа, но и простые идеалы с небольшой нормой.

Задача вычисления дискретных логарифмов может рассматриваться также и в полях $\mathbb{F}_{p^n}$, состоящих из p^n элементов, в мультипликативных группах классов вычетов $(\mathbb{Z}/m\mathbb{Z})^*$, в группах точек эллиптических кривых и вообще в произвольных группах. С литературой по этому вопросу можно ознакомиться по работе [19].

9. Заключение

Мы затронули в этой главе лишь небольшую часть вопросов, связанных с теоретико-числовыми алгоритмами и оценками их сложности. Мы не описывали перспективные исследования, связанные с распространением алгоритмов решета на поля алгебраических чисел (решето числового поля), и использование их для разложения целых чисел на множители или решения задачи дискретного логарифмирования, см. [20]. Именно с помощью этих алгоритмов достигнуты теоретические оценки сложности разложения на множители

$$\exp(c(\ln N)^{1/3}(\ln\ln N)^{2/3}).$$

Не были затронуты эллиптические кривые, т. е. определенные с точностью до обратимого множителя пропорциональности множества точек

$$E_{a,b} = \{(x, y, z) \in (\mathbb{Z}/m\mathbb{Z})^3 | y^2z = x^3 + axz^2 + bz^3\},$$

обладающие групповой структурой. С их помощью удалось построить весьма эффективные алгоритмы разложения чисел на множители и проверки целых чисел на простоту. В отличие от мультипликативной группы $(\mathbb{Z}/m\mathbb{Z})^*$ порядок группы $E_{a,b}$ при одном и том же m меняется в зависимости от целых параметров a, b. Это оказывается весьма существенным, например, при разложении чисел m на множители. Мы

отсылаем читателей за подробностями использования эллиптических кривых к статье [21].

Литература к главе 4

[1] *Rivest R. L., Shamir A., Adleman L.* A method for obtaining digital signatures and public key cryptosystems // Commun. ACM. V.21, No 2, 1978. P. 120–126.

[2] *Gardner M.* A new kind of cipher that would take millions of years to break // Sci. Amer. 1977. P. 120–124.

[3] *Виноградов И. М.* Основы теории чисел. М.: Наука, 1972.

[4] *Карацуба А. А.* Основы аналитической теории чисел. М.: Наука, 1983 г.

[5] *Atkins D., Graff M., Lenstra A. K. and Leyland P. C.* The magic words are squeamish ossifrage // ASIACRYPT–94, Lect. Notes in Comput. Sci. V. 917. Springer, 1995.

[6] *Кнут Д.* Искусство программирования на ЭВМ. Т.2: Получисленные алгоритмы. М.: Мир, 1977.

[7] *Ахо А., Хопкрофт Дж., Ульман Дж.* Построение и анализ вычислительных алгоритмов. М.: Мир, 1979.

[8] *Williams H. C.* Primality testing on a computer // Ars Combin., 5, 1978. P. 127–185. (Русский перевод: Кибернетический сборник, вып. 23, 1986. С. 51–99.)

[9] *Василенко О. Н.* Современные способы проверки простоты чисел // Кибернетический сборник, вып. 25, 1988. С. 162–188.

[10] *Alford W. R., Granville A., Pomerance C.* There are infinitely many Carmichael numbers // Ann. Math. 140, 1994. P. 703–722.

[11] *Прахар К.* Распределение простых чисел. М.: Мир, 1967.

[12] *Plaisted D. A.* Fast verification, testing, and generation of large primes // Theor. Comp. Sci. 9, 1979. P. 1–16.

[13] *Adleman L. M., Pomerance C., Rumely R. S.* On distinguishing prime numbers from composite numbers // Annals of Math. 117, 1983. P. 173–206.

[14] *Lenstra H. W. (jr.)* Primality testing algorithms (after Adleman, Rumely and Williams) // Lecture Notes in Math. V. 901, 1981. P. 243–257.

[15] *Cohen H., Lenstra H. W. (jr.)* Primality testing and Jacobi sums // Math. of Comput. V. 42, №165, 1984. P. 297–330.

[16] *Riesel H.* Prime numbers and computer methods for factorization. Birkhauser, 1985.

[17] *Cohen H.* A course in computational algebraic number theory. GraduateTexts in Math. V. 138. New York, Springer, 1993.

[18] *Coppersmith D., Odlyzko A. M., Schroeppel R.* Discrete logarithms in $GF(p)$ // Algorithmica. V. 1, 1986. P. 1–15.

[19] *McCurley K. S.* The discrete logarithm problem // Proc. of Symp. in Appl. Math. V. 42, 1990. P. 49–74.

[20] *Lenstra A. K., Lenstra H. W., Manasse M. S., Pollard J. M.* The number field sieve // Proc. 22nd Ann. ACM Symp. on Theory of Computing. Baltimore, May 14–16, 1990. P. 564–572.

[21] *Lenstra H. W. (jr.)* Elliptic curves and number-theoretic algorithms // ICM86. P. 99–120. (Русский перевод: Международный конгресс математиков в Беркли, М.: Мир, 1991, С. 164–193.)

[22] *Koblitz N.* A Course in Number Theory and Cryptography. 2nd ed. Springer, 1994.

[23] *Lenstra A. K., Lenstra H. W. (jr.)* The Development of the Number Field Sieve. Lect. Notes in Math. V. 1554. Springer, 1993.

[24] *Ben-Or M.* Probabilistic algorithms in finite fields. Proc. 22 IEEE Symp. Found. Comp. Sci, 1981. P. 394–398.

[25] *Gordon D.M.* Discrete logarithms in $GF(p)$, using the number field sieve. SIAM J. Disc. Math. V.6, №1, 1993. P. 124–138.

Глава 5

Математика разделения секрета

1. Введение

Рассмотрим следующую, в наше время вполне реальную ситуацию. Два совладельца драгоценности хотят положить ее на хранение в сейф. Сейф современный, с цифровым замком на 16 цифр. Так как совладельцы не доверяют друг другу, то они хотят закрыть сейф таким образом, чтобы они могли открыть его вместе, но никак не порознь. Для этого они приглашают третье лицо, называемое дилером, которому они оба доверяют (например, потому что оно не получит больше доступ к сейфу). Дилер случайно выбирает 16 цифр в качестве «ключа», чтобы закрыть сейф, и затем сообщает первому совладельцу втайне от второго первые 8 цифр «ключа», а второму совладельцу втайне от первого — последние 8 цифр «ключа». Такой способ представляется с точки здравого смысла оптимальным, ведь каждый из совладельцев получил «полключа» и что может быть лучше?! Недостатком данного примера является то, что любой из совладельцев, оставшись наедине с сейфом, может за пару минут найти недостающие «полключа» с помощью несложного устройства, перебирающего ключи со скоростью 1 МГц. Кажется, что единственный выход — в увеличении размера «ключа», скажем, вдвое. Но есть другой, математический выход, опровергающий (в данном случае — к счастью) соображения здравого смысла. А именно, дилер независимо выбирает две случайные последовательности по 16 цифр в каждой, сообщает каждому из совладельцев втайне от другого «его» последовательность, а в качестве «ключа», чтобы закрыть сейф, использует последовательность, полученную сложением по модулю 10 соответствующих цифр двух выбранных последовательностей. Довольно очевидно (и ниже мы это докажем), что для каждого из совладельцев все 10^{16} возможных «ключей» одинаково вероятны и остается только перебирать их, что потребует в среднем более полутора лет для устройства, перебирающего ключи со скоростью 100 МГц.

И с математической, и с практической точки зрения неинтересно останавливаться на случае двух участников и следует рассмотреть

общую ситуацию. Неформально говоря, «схема, разделяющая секрет» (СРС) позволяет «распределить» секрет между n участниками таким образом, чтобы заранее заданные разрешенные множества участников могли однозначно восстановить секрет (совокупность этих множеств называется структурой доступа), а неразрешенные — не получали никакой дополнительной к имеющейся априорной информации о возможном значении секрета. СРС с последним свойством называются совершенными (и только они, как правило, рассматриваются в этой статье).

История СРС начинается с 1979 года, когда эта проблема была поставлена и во многом решена Г. Блейкли [1] и А. Шамиром [2] для случая пороговых (n, k)-СРС (т. е. разрешенными множествами являются любые множества из k или более элементов). Особый интерес вызвали так называемые идеальные СРС, т. е. такие, где «размер» информации, предоставляемой участнику, не больше «размера» секрета (а меньше, как было показано, он и не может быть). Оказалось [3], что любой такой СРС соответствует матроид (определение, что это такое, см. в п. 4) и, следовательно, не для любой структуры доступа возможно идеальное разделение секрета. С другой стороны, было показано, что для любого набора разрешенных множеств можно построить совершенную СРС, однако известные построения весьма «неэкономны». В данной статье рассматриваются алгебро-геометрические и комбинаторные задачи, возникающие при математическом анализе «схем, разделяющих секрет». Вот пример одной из таких задач.

Будем говорить, что семейство подпространств $\{L_0, \ldots, L_n\}$ конечномерного векторного пространства L над полем K удовлетворяет свойству «все или ничего», если для любого множества $A \subset \{1, \ldots, n\}$ линейная оболочка подпространств $\{L_a : a \in A\}$ либо содержит подпространство L_0 целиком, либо пересекается с ним только по вектору $\mathbf{0}$. В п. 3 мы увидим, что такое семейство задает «линейную» СРС, у которой множество $A \subset \{1, \ldots, n\}$ является разрешенным, если и только если линейная оболочка подпространств $\{L_a : a \in A\}$ содержит подпространство L_0 целиком. В связи с этим понятием возникает ряд вопросов. Например, если поле K конечно ($|K| = q$) и все подпространства $\{L_0, \ldots, L_n\}$ одномерны, то каково максимально возможное число участников n для линейных пороговых (n, k)-СРС ($k > 1$)? Иначе говоря, каково максимально возможное число векторов $\{h_0, \ldots, h_n\}$ таких, что любые k векторов, содержащие вектор h_0, линейно независимы, а любые $k+1$ векторов, содержащие вектор h_0, линейно зависимы. Оказывается, что это свойство эквивалентно следующему, на первый взгляд более сильному, свойству: любые k векторов линейно независимы, а любые $k + 1$ — линейно зависимы. Такие системы векторов изучались в геометрии как N-множества ($N = n + 1$) в конечной проективной геометрии $PG(k - 1, q)$, в комбинаторике как ортогональные таблицы

силы k и индекса $\lambda = 1$, в теории кодирования как проверочные матрицы МДР кодов (подробнее см. [4]). В п. 3 мы приведем известную конструкцию таких множеств с $N = q + 1$, а довольно старая гипотеза состоит в том, что это и есть максимально возможное N, за исключением двух случаев: случая $q < k$, когда $N = k + 1$, и случая $q = 2^m$, $k = 3$ или $k = q - 1$, когда $N = q + 2$.

2. Разделение секрета для произвольных структур доступа

Начнем с формальной математической модели. Имеется $n+1$ множество $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_n$ и (совместное) распределение вероятностей P на их декартовом произведении $\mathcal{S} = \mathcal{S}_0 \times \cdots \times \mathcal{S}_n$. Соответствующие случайные величины обозначаются через S_i. Имеется также некоторое множество Γ подмножеств множества $\{1, \ldots, n\}$, называемое структурой доступа.

Определение 1. Пара $(P, \mathcal{S})$ называется *совершенной вероятностной СРС*, реализующей структуру доступа Γ, если

$$P(S_0 = c_0 | S_i = c_i, i \in A) \in \{0, 1\} \text{ для } A \in \Gamma, \tag{1}$$

$$P(S_0 = c_0 | S_i = c_i, i \in A) = P(S_0 = c_0) \text{ для } A \notin \Gamma. \tag{2}$$

Это определение можно истолковать следующим образом. Имеется множество $\mathcal{S}_0$ всех возможных секретов, из которого секрет s_0 выбирается с вероятностью $p(s_0)$, и имеется СРС, которая «распределяет» секрет s_0 между n участниками, посылая «проекции» $s_1, \ldots, s_n$ секрета с вероятностью $P_{s_0}(s_1, \ldots, s_n)$. Отметим, что i-й участник получает свою «проекцию» $s_i \in \mathcal{S}_i$ и не имеет информации о значениях других «проекций», однако знает все множества $\mathcal{S}_i$, а также оба распределения вероятностей $p(s_0)$ и $P_{s_0}(s_1, \ldots, s_n)$. Эти два распределения могут быть эквивалентно заменены на одно: $P(s_0, s_1, \ldots, s_n) = p(s_0) P_{s_0}(s_1, \ldots, s_n)$, что и было сделано выше. Цель СРС, как указывалось во введении, состоит в том, чтобы:

а) участники из разрешенного множества A (т. е. $A \in \Gamma$) вместе могли бы однозначно восстановить значение секрета — это отражено в свойстве (1);

б) участники, образующие неразрешенное множество A ($A \notin \Gamma$), не могли бы получить дополнительную информацию об s_0, т. е., чтобы вероятность того, что значение секрета $S_0 = c_0$, не зависела от значений «проекций» S_i при $i \in A$ — это свойство (2).

Замечание о терминологии. В англоязычной литературе для обозначения «порции» информации, посылаемой участнику СРС, были введены термины «share» (А. Шамир) и «shadow» (Г. Блейкли). Первый

термин оказался наиболее популярным и автор долго боролся с соблазном привлечь массового читателя, постоянно используя в качестве его перевода слово «акция». Неадекватная (во всех смыслах) замена «акции» на «проекцию» может быть несколько оправдана следующим примером.

Пример 1. Множество $\mathcal{S}_0$ всех возможных секретов состоит из 0, 1 и 2, «представленных» соответственно: шаром; кубом, ребра которого параллельны осям координат; цилиндром, образующие которого параллельны оси Z. При этом диаметры шара и основания цилиндра, и длины ребра куба и образующей цилиндра, равны. Первый участник получает в качестве своей «доли» секрета его проекцию на плоскость XY, а второй — на плоскость XZ. Ясно, что вместе они однозначно восстановят секрет, а порознь — не могут. Однако, эта СРС не является совершенной, так как любой из участников получает информацию о секрете, оставляя только два значения секрета как возможные при данной проекции (например, если проекция — квадрат, то шар невозможен).

Еще одно замечание. Элемент (участник) $x \in \{1, \ldots, n\}$ называется несущественным (относительно Γ), если для любого неразрешенного множества A множество $A \cup x$ также неразрешенное. Очевидно, что несущественные участники настолько несущественны для разделения секрета, что им просто не нужно посылать никакой информации. Поэтому далее, без ограничения общности, рассматриваются только такие структуры доступа Γ, для которых все элементы являются существенными. Кроме того, естественно предполагать, что Γ является монотонной структурой, т. е. из $A \subset B, A \in \Gamma$ следует $B \in \Gamma$.

Пример 2. Рассмотрим простейшую структуру доступа — (n, n)-пороговую схему, т. е. все участники вместе могут восстановить секрет, а любое подмножество участников не может получить дополнительной информации о секрете. Будем строить идеальную СРС, выбирая и секрет, и его проекции из группы Z_q вычетов по модулю q, т. е. $\mathcal{S}_0 = \mathcal{S}_1 = \ldots = \mathcal{S}_n = Z_q$. Дилер генерирует $n-1$ независимых равномерно распределенных на Z_q случайных величин x_i и посылает i-му участнику $(i = 1, \ldots, n-1)$ его «проекцию» $s_i = x_i$, а n-му участнику посылает $s_n = s_0 - (s_1 + \cdots + s_{n-1})$. Кажущееся «неравноправие» n-ого участника тут же исчезает, если мы выпишем распределение $P_{s_0}(s_1, \ldots, s_n)$, которое очевидно равно $1/q^{n-1}$, если $s_0 = s_1 + \cdots + s_n$, и равно 0 — в остальных случаях. Теперь легко проверяется и свойство (2), означающее в данном случае независимость случайной величины S_0 от случайных величин $\{S_i : i \in A\}$ при любом собственном подмножестве A.

Данное выше определение СРС, оперирующее словами «распределение вероятностей», ниже переведено, почти без потери общности, на комбинаторный язык, который представляется автору более простым для понимания. Произвольная $M \times (n+1)$-матрица V, строки которой

имеют вид $\mathbf{v} = (v_0, v_1, \ldots, v_n)$, где $v_i \in \mathcal{S}_i$, называется матрицей комбинаторной СРС, а ее строки — «правилами» распределения секрета. Для заданного значения секрета s_0 дилер СРС случайно и равновероятно выбирает строку $\mathbf{v}$ из тех строк матрицы V, для которых значение нулевой координаты равно s_0.

Определение 2. Матрица V задает *совершенную комбинаторную СРС*, реализующую структуру доступа Γ, если, во-первых, для любого множества $A \in \Gamma$ нулевая координата любой строки матрицы V однозначно определяется значениями ее координат из множества A, и, во-вторых, для любого множества $A \notin \Gamma$ и любых заданных значений координат из множества A число строк матрицы V с данным значением α нулевой координаты не зависит от α.

Сопоставим совершенной вероятностной СРС, задаваемой парой $(P, \mathcal{S})$, матрицу V, состоящую из строк $s \in \mathcal{S}$, таких что $P(s) > 0$. Заметим, что если в определении 1 положить все ненулевые значения P одинаковыми, а условия (1) и (2) переформулировать на комбинаторном языке, то получится определение 2. Это комбинаторное определение несколько обобщается, если допустить в матрице V повторяющиеся строки, что эквивалентно вероятностному определению 1, когда значения вероятностей $P(s)$ — рациональные числа.

Пример 2 (продолжение). Переформулируем данную выше конструкцию (n, n)-пороговой СРС на комбинаторном языке. Строками матрицы V являются все векторы $\mathbf{s}$ такие, что $-s_0 + s_1 + \cdots + s_n = 0$. Очевидно, что матрица V задает совершенную комбинаторную СРС для $\Gamma = \{1, \ldots, n\}$, так как для любого собственного подмножества $A \subset \{1, \ldots, n\}$ и любых заданных значений координат из множества A число строк матрицы V с данным значением нулевой координаты равно $q^{n-1-|A|}$.

Удивительно, но простой схемы примера 2 оказывается достаточно, чтобы из нее, как из кирпичиков, построить совершенную СРС для произвольной структуры доступа. А именно, для всех разрешенных множеств, т. е. для $A \in \Gamma$, независимо реализуем описанную только что пороговую $(|A|, |A|)$-СРС, послав тем самым i-му участнику столько «проекций» s_i^A, скольким разрешенным множествам он принадлежит. Это словесное описание несложно перевести на комбинаторный язык свойств матрицы V и убедиться, что эта СРС совершенна. Как это часто бывает, «совершенная» не значит «экономная», и у данной СРС размер «проекции» оказывается, как правило, во много раз больше, чем размер секрета. Эту схему можно сделать более экономной, так как достаточно реализовать пороговые $(|A|, |A|)$-СРС только для минимальных разрешенных множеств A, т. е. для $A \in \Gamma_{\min}$, где $\Gamma_{\min}$ — совокупность минимальных (относительно включения) множеств из Γ. Тем

не менее, для пороговой $(n, n/2)$-СРС размер «проекции» (измеренный, например, в битах) будет в $C_n^{n/2} \sim 2^n/\sqrt{2\pi n}$ раз больше размера секрета (это наихудший случай для рассматриваемой конструкции). С другой стороны, как мы убедимся чуть позже, любая пороговая структура доступа может быть реализована идеально, т. е. при совпадающих размерах «проекции» и секрета. Поэтому естественно возникает вопрос о том, каково максимально возможное превышение размера «проекции» над размером секрета для наихудшей структуры доступа при наилучшей реализации. Формально, $R(n) = \max R(\Gamma)$, где max берется по всем структурам доступа Γ на n участниках, а $R(\Gamma) = \min \max \frac{\log|S_i|}{\log|S_0|}$, где min берется по всем СРС, реализующим данную структуру доступа Γ, а max — по $i = 1, \ldots, n$. Приведенная конструкция показывает, что $R(n) \leqslant C_n^{n/2}$. С другой стороны, как было доказано лишь недавно [5], $R(n) \geqslant n/\log n$. Такая огромная «щель» между верхней и нижней оценкой дает, по нашему мнению, достаточный простор для исследований (автор предполагает, что $R(n)$ растет экспоненциально от n).

3. Линейное разделение секрета.

Начнем с предложенной А. Шамиром [2] элегантной схемы разделения секрета для пороговых структур доступа. Пусть $K = GF(q)$ конечное поле из q элементов (например, $q = p$ — простое число и $K = Z_p$) и $q > n$. Сопоставим участникам n различных ненулевых элементов поля $\{a_1, \ldots, a_n\}$ и положим $a_0 = 0$. При распределении секрета s_0 дилер СРС генерирует $k-1$ независимых равномерно распределенных на $GF(q)$ случайных величин f_j $(j = 1, \ldots, k-1)$ и посылает i-му участнику $(i = 1, \ldots, n)$ «его» значение $s_i = f(a_i)$ многочлена $f(x) = f_0 + f_1 x + \cdots + f_{k-1} x^{k-1}$, где $f_0 = s_0$. Поскольку любой многочлен степени $k-1$ однозначно восстанавливается по его значениям в произвольных k точках (например, по интерполяционной формуле Лагранжа), то любые k участников вместе могут восстановить многочлен $f(x)$ и, следовательно, найти значение секрета как $s_0 = f(0)$. По этой же причине для любых $k-1$ участников, любых полученных ими значений проекций s_i и любого значения секрета s_0 существует ровно один «соответствующий» им многочлен, т. е. такой, что $s_i = f(a_i)$ и $s_0 = f(0)$. Следовательно, эта схема является совершенной в соответствии с определением 2. «Линейность» данной схемы становится ясна, если записать «разделение секрета» в векторно-матричном виде:

$$\mathbf{s} = \mathbf{f}H, \tag{3}$$

где $\mathbf{s} = (s_0, \ldots, s_n), \mathbf{f} = (f_0, \ldots, f_{k-1})$, $k \times (n+1)$-матрица $H = (h_{ij}) = (a_i^{j-1})$ и $h_{00} = 1$. Заметим, что любые k столбцов этой матрицы

линейно независимы, а максимально возможное число столбцов матрицы H равно q, и чтобы добиться обещанного в п. 1 значения $q+1$ надо добавить столбец, соответствующий точке «бесконечность».

Упражнение. *Придумайте сами, как это сделать.*

Возьмем в (3) в качестве H произвольную $r \times (n+1)$-матрицу с элементами из поля K. Получаемую СРС будем называть одномерной линейной СРС. Она является совершенной комбинаторной СРС со структурой доступа Γ, состоящей из множеств A таких, что вектор $\mathbf{h}_0$ представим в виде линейной комбинации векторов $\{\mathbf{h}_j : j \in A\}$, где $\mathbf{h}_j$ это j-ый столбец матрицы H. Строками матрицы V, соответствующей данной СРС являются, как видно из (3), линейные комбинации строк матрицы H. Перепишем (3) в следующем виде

$$s_j = (\mathbf{f}, \mathbf{h}_j) \text{ для } j = 0, 1, \ldots, n,$$

где $(\mathbf{f}, \mathbf{h}_j)$ — скалярное произведение векторов $\mathbf{f}$ и $\mathbf{h}_j$. Если $A \in \Gamma$, т. е. если $\mathbf{h}_0 = \sum \lambda_j \mathbf{h}_j$, то

$$s_0 = (\mathbf{f}, \mathbf{h}_0) = (\mathbf{f}, \sum \lambda_j \mathbf{h}_j) = \sum \lambda_j (\mathbf{f}, \mathbf{h}_j) = \sum \lambda_j s_j$$

и, следовательно, значение секрета однозначно находится по его «проекциям». Рассмотрим теперь случай, когда вектор $\mathbf{h}_0$ не представим в виде линейной комбинации векторов $\{\mathbf{h}_j : j \in A\}$. Нам нужно показать, что в этом случае для любых заданных значений координат из множества A число строк матрицы V с данным значением нулевой координаты не зависит от этого значения. В этом нетрудно убедиться, рассмотрев (3) как систему линейных уравнений относительно неизвестных f_i и воспользовавшись тем, что система совместна тогда и только тогда, когда ранг матрицы коэффициентов равен рангу расширенной матрицы, а число решений у совместных систем одинаково и равно числу решений однородной системы.

Указание. Рассмотрите две системы: без «нулевого» уравнения (т. е. со свободным членом) и с ним. Так как вектор $\mathbf{h}_0$ не представим в виде линейной комбинации векторов $\{\mathbf{h}_j : j \in A\}$, то ранг матрицы коэффициентов второй системы на 1 больше ранга матрицы коэффициентов первой системы. Отсюда немедленно следует, что если первая система совместна, то совместна и вторая при любом s_0.

Эта конструкция подводит нас к определению общей линейной СРС. Пусть секрет и его «проекции» представляются как конечномерные векторы $\mathbf{s}_i = (s_i^1, \ldots, s_i^{m_i})$ и генерируются по формуле $\mathbf{s}_i = \mathbf{f}H_i$, где H_i — некоторые $r \times m_i$-матрицы. Сопоставим каждой матрице H_i линейное пространство L_i ее столбцов (т. е. состоящее из всех линейных комбинаций вектор-столбцов матрицы H_i). Несложные рассуждения, аналогичные приведенным выше для одномерного случая (все $m_i = 1$), показывают, что данная конструкция дает совершенную СРС тогда и

только тогда, когда семейство линейных подпространств $\{L_0, \ldots, L_n\}$ конечномерного векторного пространства K^r удовлетворяет упомянутому во введении свойству «все или ничего». При этом множество A является разрешенным ($A \in \Gamma$), если и только если линейная оболочка подпространств $\{L_a : a \in A\}$ содержит подпространство L_0 целиком. С другой стороны, множество A является неразрешенным ($A \notin \Gamma$), если и только если линейная оболочка подпространств $\{L_a : a \in A\}$ пересекается с подпространством L_0 только по вектору $\mathbf{0}$. Отметим, что если бы для некоторого A пересечение L_0 и линейной оболочки $\{L_a : a \in A\}$ было нетривиальным, то участники A не могли бы восстановить секрет однозначно, но получали бы некоторую информацию о нем, т. е. схема не была бы совершенной.

Пример 3. Рассмотрим следующую структуру доступа для случая четырех участников, задаваемую $\Gamma_{\min} = \{\{1,2\},\{2,3\},\{3,4\}\}$. Она известна как первый построенный пример структуры доступа, для которой не существует идеальной реализации. Более того, было доказано, что для любой ее совершенной реализации $R(\Gamma) \geqslant 3/2$. С другой стороны, непосредственная проверка показывает, что выбор матриц $H_0, H_1, \ldots, H_4$, приведенных в табл. 1, дает совершенную линейную СРС с $R = 3/2$, реализующую эту структуру, которая, следовательно, является и оптимальной (наиболее экономной) СРС.

Таблица 1.

$$H_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, H_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, H_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, H_3 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, H_4 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

4. Идеальное разделение секрета и матроиды

Начнем с определения идеальных СРС. Для этого вернемся к комбинаторному определению совершенной СРС. Следующее определение совершенной СРС [3] является даже более общим, чем вероятностное определение 1, поскольку условие (2) заменено в нем на более слабое.

Для произвольного множества $B \subseteq \{0, 1, \ldots, n\}$ обозначим через V_B $M \times |B|$-матрицу, полученную из матрицы V удалением столбцов,

номера которых не принадлежат множеству B. Пусть $||W||$ обозначает число различных строк в матрице W.

Определение 3. Матрица V задает БД-совершенную СРС, реализующую структуру доступа Γ, если

$$||V_{A\cup 0}|| = ||V_A|| \times ||V_0||^{\delta_\Gamma(A)}, \tag{4}$$

где $\delta_\Gamma(A) = 0$, если $A \in \Gamma$, и $\delta_\Gamma(A) = 1$ в противном случае.

Это определение отличается от определений 1 и 2 тем, что на неразрешенные множества A накладывается довольно слабое условие, а именно, если множество строк V с данными значениями координат из множества A непусто, то все возможные значения секрета встречаются в нулевой координате этих строк (без требований «одинаково часто» как в комбинаторном определении 2 или же «с априорной вероятностью» как в вероятностном определении 1). Легко видеть, что матрица любой совершенной вероятностной СРС задает БД-совершенную СРС, но обратное неверно.

Для произвольной комбинаторной СРС, задаваемой матрицей V, определим на множествах $A \subseteq \{0, 1, \ldots, n\}$ функцию $h(A) = \log_q ||V_A||$, где $q = |\mathcal{S}_0|$. Легко проверить, что $\max\{h(A), h(B)\} \leqslant h(A \cup B) \leqslant h(A) + h(B)$ для любых множеств A и B, а условие (4) может быть переписано в виде

$$h_q(V_{A\cup 0}) = h_q(V_A) + \delta_\Gamma(A) h_q(V_0),$$

Лемма. *Для любой БД-совершенной СРС если $A \notin \Gamma$ и $\{A \cup i\} \in \Gamma$, то $h(i) \geqslant h(0)$.*

Доказательство. По условиям леммы $h(A \cup 0) = h(A) + h(0)$ и $h(A \cup i \cup 0) = h(A \cup i)$. Следовательно,

$$h(A) + h(i) \geqslant h(A \cup i) = h(A \cup i \cup 0) \geqslant h(A \cup 0) = h(A) + h(0). \qquad \blacksquare$$

Так как мы предполагаем, что все точки $i \in \{1, \ldots, n\}$ существенные, т. е. для любого i найдется подмножество A такое, что $A \notin \Gamma$ и $\{A \cup i\} \in \Gamma$, то из леммы вытекает

Следствие. *Для любой БД-совершенной СРС $|\mathcal{S}_i| \geqslant |\mathcal{S}_0|$ для всех $i = 1, \ldots, n$.*

Следствие означает, как мы и предупреждали в начале статьи, что для совершенных СРС «размер» проекции не может быть меньше «размера» секрета. Поэтому БД-совершенная СРС называется идеальной, если $|\mathcal{S}_i| = |\mathcal{S}_0|$ для всех $i = 1, \ldots, n$.

Замечание. Неравенство $|\mathcal{S}_i| \geqslant |\mathcal{S}_0|$ справедливо и для совершенных вероятностных СРС, поскольку их матрицы задают БД-совершенные СРС.

Естественный вопрос состоит в том, для каких структур доступа Γ существуют реализующие их идеальные (вероятностные или комби-

наторные) СРС. Как уже отмечалось во введении, наилучший на сегодняшний день ответ использует слово «матроид». Напомним определение матроидов и некоторые их основные свойства (см. [6]).

Матроидом называется конечное множество X и семейство I его подмножеств, называемых независимыми (остальные множества называются зависимыми), если выполнены следующие свойства:

$$\varnothing \in I; \tag{5.1}$$

$$\text{Если } A \in I \text{ и } B \subset A, \text{ то } B \in I; \tag{5.2}$$

$$\text{Если } A, B \in I \text{ и } |A| = |B| + 1,$$
$$\text{то существует } a \in A \backslash B \text{ такое, что } a \cup B \in I. \tag{5.3}$$

Пример 4. Множество X — это множество векторов в некотором линейном векторном пространстве, а независимые подмножества — это линейно независимые подмножества векторов.

Собственно с этого примера и началась теория матроидов, вначале как попытка дать аксиоматическое определение линейной независимости векторов через «внутренние свойства», т. е. не апеллируя к понятию вектора. К счастью, попытка не удалась, так как нашлись матроиды, не представимые как линейные (т. е. как системы векторов), а сама теория матроидов разрослась далеко за пределы «линейной алгебры» (см. [6]).

Пример 5 (матроид Вамоса). Рассмотрим следующее множество: $X = \{1, 2, 3, 4, 5, 6, 7, 8\}$ и положим $a = \{1, 2\}$, $b = \{3, 4\}$, $c = \{5, 6\}$ и $d = \{7, 8\}$. Матроид Вамоса определяется как матроид, в котором множества $a \cup c$, $a \cup d$, $b \cup c$, $b \cup d$, $c \cup d$, а также все подмножества из пяти или более элементов являются зависимыми. Известно, что этот матроид не является линейным.

Матроид также можно определить через так называемую ранговую функцию $r(A)$ матроида, определяемую как максимальная мощность независимого подмножества $B \subseteq A$. Очевидно, что независимые множества (и только они) задаются условием $r(A) = |A|$. Ранговая функция матроида обладает свойствами

$$r(A) \in Z,\ r(\varnothing) = 0; \tag{6.1}$$

$$r(A) \leqslant r(A \cup b) \leqslant r(A) + 1; \tag{6.2}$$

$$\text{Если } r(A \cup b) = r(A \cup c) = r(A), \text{ то } r(A \cup b \cup c) = r(A). \tag{6.3}$$

Обратно, пусть некоторая функция $r(A)$ обладает свойствами (6). Назовем независимыми те множества A, для которых $r(A) = |A|$. Тогда эти множества задают матроид, а функция r является его ранговой функцией. Возможно также определить матроид через минимальные зависимые множества, называемые циклами. Матроид называется связным, если для любых двух его точек существует содержащий их цикл.

Теперь мы можем сформулировать основной результат.

Теорема ([3]). *Для любой БД-совершенной идеальной СРС, реализующей структуру доступа* Γ, *независимые множества, определяемые условием* $\log_{|\mathcal{S}_0|} ||V_A|| = |A|$, *задают связный матроид на множестве* $\{0, 1, \ldots, n\}$. *Все циклы этого матроида, содержащие точку* 0, *имеют вид* $0 \cup A$, *где* $A \in \Gamma_{min}$.

Главным в доказательстве теоремы является «проверка» целочисленности функции $h(A)$. В самом деле, $h(\cdot)$ очевидно обладает остальными свойствами (6) и, следовательно, при условии целочисленности является ранговой функцией и задает матроид. Доказательство этой теоремы и несколько более общих утверждений можно найти в [7].

Отметим, что из второй части утверждения теоремы следует, что разным идеальным СРС, реализующим данную структуру доступа Γ, всегда соответствует один и тот же матроид, поскольку матроид однозначно определяется всеми циклами, проходящими через фиксированную точку (см. [6]). Тем самым, каждой идеально реализуемой структуре доступа соответствует однозначно определенный матроид.

В связи с теоремой возникает несколько естественных вопросов. Прежде всего, не порождают ли идеальные СРС все матроиды? Нет, например, матроид Вамоса не может быть получен как матроид идеальной СРС [8]. С другой стороны, линейные матроиды есть ни что иное как рассмотренные в п. 3 идеальные одномерные линейные СРС. В связи с этим возникает вопрос о существовании структуры доступа Γ, которую невозможно реализовать в виде идеальной одномерной линейной СРС, но можно в виде идеальной многомерной линейной СРС. Недавно такой пример был построен [9], и, значит, мы можем говорить о многомерных линейных матроидах как классе матроидов более общем, чем линейные.

Итак, идеальных СРС больше, чем линейных матроидов, но меньше, чем всех матроидов. Уточнить, «насколько больше», представляется довольно сложной задачей. В частности, существует ли идеально реализуемая структура доступа Γ, которую невозможно реализовать как идеальную линейную многомерную СРС?

Литература к главе 5

[1] *Blakley G. R.* Safeguarding cryptographic keys // Proc. AFIPS 1979 National Computer Conference. V. 48. N. Y., 1979. P. 313–317.

[2] *Shamir A. How to Share a Secret* // Comm. ACM. V. 22, No 1, 1979. P. 612–613.

[3] *Brickell E. F., Davenport D. M.* On the classification of Ideal Secret Sharing Schemes. // J. Cryptology. V. 4, 1991. P. 123–134.

[4] *Мак-Вильямс Ф. Дж., Слоэн Н. Дж. А. Теория кодов, исправляющих ошибки.* М.: Связь, 1979.

[5] *Csirmaz L.* The size of a share must be large // J. Cryptology. V. 10, No 4, 1997. P. 223–232.

[6] *Welsh D. J. A.* Matroid Theory. Academic Press, 1976.

[7] *Блейкли Г. Р., Кабатянский Г. А.* Обобщенные идеальные схемы, разделяющие секрет, и матроиды // Проблемы передачи информации. Т. 33, вып. 3, 1997. С. 102–110.

[8] *Seymour P. O.* On Secret-Sharing Matroids. // J. Comb. Theory. Ser. B. V. 56, 1992. P. 69–73.

[9] *Ashihmin A., Simonis J.* Almost Affine codes. // Designs, codes and cryptography.

Глава 6

Компьютер и криптография

“ ... задача воеводства совсем не в том состоит, чтобы достигать какого-то мечтательного благополучия, а в том, чтобы исстари заведенный порядок (хотя бы и не благополучный) от повреждений оберегать и ограждать.”

М. Е. Салтыков-Щедрин

1. Вместо введения

Для чего криптографии нужен компьютер?

Криптография — одна из древнейших наук. О ней вспоминают всегда, когда возникает необходимость скрывать тайны. Но одним карандашом и бумагой пользоваться достаточно трудоемко и обременительно. И с древнейших времен человек пытается использовать различные орудия, позволяющие облегчить труд шифровальщика. После многовекового использования веревочек, жезлов, полосок, ленточек, планшетов, трафаретов, дисков и т. д. и т. п., криптография в начале XX века поставила себе на службу массу механических, пневматических, электрических, а затем и электронных устройств. В 40-х годах она придумала для своих нужд первые электронные вычислители. Появление компьютеров также поначалу было воспринято именно как появление мощнейшего помощника для людей, занимающихся разработкой и анализом алгоритмов шифрования. Кстати, первый компьютер с названием «Colossus», в создании которого участвовал математик А. Тьюринг, был разработан в Англии именно для решения задач дешифрования германской шифрмашины «Enigma» в самом начале Второй мировой войны. Но компьютеры не облегчили изнуряющий труд криптографа, а только привнесли массу новых проблем. Появились новые виды информации, требующей закрытия, новые области применения криптографических методов, а самое главное — существенно возросли возможности противника по раскрытию применявшихся ранее шифров.

Первые три десятилетия после своего появления компьютеры оправдывали свое название «вычислитель» и представляли собой инструмент для создания новых и взлома старых шифров.

В восьмидесятые годы произошла неизбежная переоценка ценностей. Из подсобного средства, вычислителя, компьютер стал центральным звеном для множества различных самостоятельных систем, выполняющих самые разнообразные функции. Это — информационные системы, системы связи и системы управления, системы автоматического проектирования, автоматизированные системы практически для всех областей человеческой деятельности, включая производственную, военную, финансовую, экономическую, медицинскую, и многие другие. Нетрудно догадаться, что и в области криптографии компьютер занял центральное место, взяв на себя большинство функций традиционной криптографической деятельности, включая реализацию криптографических алгоритмов, проверку их качества, генерацию и распределение ключей, автоматизацию работы по анализу перехвата и раскрытию шифров и т. д.

Наконец, сейчас, в конце девяностых годов в связи с появлением глобальных сетей, мы начинаем осознавать неизбежность вхождения в мировое информационное пространство, часто называемое киберпространством, которое стремительно развивается по своим законам, засасывая миллионы новых и новых обитателей. Проваливаясь в него, мы забываем о существовании времени и пространства, теряем грань между реальностью и вымыслом. Самые фантастические идеи находят в нем простое и естественное воплощение, действительное становится виртуальным, а виртуальное действительным. Универсальность, полная совместимость и взаимосвязанность удаленных друг от друга компьютеров позволяют, с одной стороны, реализовать принципиально новые криптографические идеи, а с другой, существенно расширить области применения криптографии.

Вместе с тем, восхищаясь возможностями компьютеров, не надо забывать о простых вещах. Как справедливо заметил американский математик Нил Коблиц — автор книги «Курс по теории чисел и криптографии», — выступая на семинаре в Московском университете, системы с открытыми ключами созданы для людей, которые очень сильно не доверяют друг другу, но при этом бесконечно доверяют своим компьютерам.

Для чего компьютеру нужна криптография?

Пока компьютеры были большими, их обслуживали специально подготовленные инженеры-программисты, системщики, аппаратчики, информационщики, а также операторы и технический персонал. Доступ к хранящейся и обрабатываемой в них информации имел также

ограниченный круг людей. Поэтому проблемы защиты информации в основном сводились к повышению надежности работы, дублированию критичной информации и организационным мерам. Позднее было осознано, что наряду с безопасностью данных огромную роль играет также безопасность программного обеспечения и аппаратных средств. Поэтому стали говорить о компьютерной безопасности. Наконец, с появлением автоматизированных систем обработки данных (представляющих собой неразрывное целое из объединенных в сеть компьютеров, средств телекоммуникаций, информационных технологий и распределенных информационных массивов) стали больше говорить об информационной безопасности системы в целом, понимая под этим состояние защищенности всех процессов обработки, хранения и передачи информации в системе.

Естественно, что криптография заняла в этой области подобающее ей место, предоставив массу алгоритмов для закрытия хранимой и передаваемой конфиденциальной информации. Как ни странно, но даже для информационных систем, обрабатывающих только открытую и общедоступную информацию, также не удалось обойтись без криптографических решений. Действительно, можно ли доверять полученным от такой системы данным, если они могут быть легко изменены, модифицированы или даже уничтожены кем-то из пользователей. Например, кому нужна информационная система с недостоверной информацией, или система принятия решений, работа которой основана на случайной или подтасованной информации? Именно криптография предоставляет незаменимый набор средств для обеспечения безопасности работы системы, такие как электронная цифровая подпись, протоколы идентификации и аутентификации абонентов, коды аутентификации сообщений и многое другое.

Вместе с тем, как это обычно бывает, при практической реализации даже самых хороших теоретических результатов возникают «маленькие» трудности. Чтобы получить представление о том, что это за трудности, проанализируем процесс создания программы для шифрования файлов.

2. Немного теории

Что надо знать перед написанием программы шифрования

С появлением персональных компьютеров криптография приобрела совершенно новое лицо. Хотя компьютеры привнесли немало нового и в криптографию, все же самое радикальное новшество заключается в том, что о криптографии стали говорить открыто и на каждом углу. Где есть компьютеры, там обязательно появляются вопросы о защите

информации. А где начинают задумываться о защите информации — там неизбежно вспоминают о криптографии.

Криптография — очень таинственная и хитрая область знаний. Как правило, о ней можно было прочитать очень мало. Сейчас же ситуация изменилась, и о ней пишут много. Парадокс в том, что чем больше о ней пишут, тем меньше в ней понимаешь. Точнее, тем больше понимаешь, что ты в ней ничего не понимаешь.

Не секрет, что каждый, кто хоть немного поработал на компьютере, не раз задумывался о защите своих файлов. Неважно от кого. Просто каждый из нас не любит, когда в его делах кто-то копается. А при работе с персональным компьютером (ПК) постоянно возникает ситуация, когда он находится в безраздельной власти, в общем-то, посторонних вам людей. К соблазну попробовать зашифровать файлы, директории или диски постоянно подталкивают нас и многочисленные программы со встроенными функциями шифрования: нортоновский Diskreet, Secretdisk, различные архиваторы, редакторы и т. д. Поэтому без преувеличения можно сказать, что с криптографией сталкивался каждый пользователь ПК.

На более серьезном уровне приходится строить отношения с криптографией тем, кто имеет свое дело и использует ПК или локальную сеть ПК в своей организации или фирме. Здесь уже потери от небрежного обращения со своей информацией могут привести к непоправимым последствиям. Не будем утомлять читателя примерами из жизни, поскольку каждый наверняка читал об этом в газетах или слышал от друзей.

Дело не в примерах, а в сути дела. А суть в том, что без криптографии решить задачи сохранения ваших тайн практически невозможно. Не будем тратить время на доказательство этого тезиса. Интуитивно это понимают все, хотя у непрофессионалов и возникает иногда некоторое чувство внутреннего сопротивления.

Итак, вы поняли, что вам надо воспользоваться какой-либо программой для шифрования. И вот здесь-то сразу появляются непреодолимые трудности. То, что они непреодолимые, вы понимаете чуть позднее. Для того, чтобы выбрать для себя нужную программу, надо научиться оценивать ее качество и уметь сравнивать такие программы между собой. А вот здесь-то никаких критериев вам никто и никогда не расскажет, да и в книгах вы ничего путного не прочитаете. А верить продавцу с его навязчивой и крикливой рекламой в данном случае полностью абсурдно, так как продавцом может оказаться ваш конкурент со всеми вытекающими отсюда последствиями. Верить можно только той программе, которую вы написали сами. Если вы с этим не согласны, значит у вас еще все впереди. Ваши иллюзии еще не скоро рассеются.

Итак, приступим к написанию программы шифрования файлов. Поначалу создание такой программы кажется вполне доступной задачей, с которой может справиться любой человек, изучивший какой-нибудь язык программирования. Нет ничего проще. Вы выбираете самый надежный из известных вам алгоритм шифрования, например описанный в ГОСТ 28147, садитесь за компьютер, и через некоторое время у вас появляется работающая программа. Хотя она работает не очень быстро и интерфейс у нее не очень удобный, она вас вполне устраивает и вы ей полностью доверяете, поскольку это ваша программа.

Но со временем у вас опять появляется червь сомнения. Вы опять начинаете задумываться о надежности вашей защиты и понимаете, что, несмотря на свой богатый опыт в области криптографии, вы в сущности ничего в этом не понимаете. Единственное, что согревает душу в такой ситуации, так это то, что вы теперь ясно осознаете необходимость серьезного подхода к решению задачи. А это уже очень много. С этого момента и начинается ваша работа по освоению криптографии.

Какой алгоритм выбрать?

Мы разговаривали со многими криптографами, и все они начинали с объяснения, что такое шифр Цезаря. Он прост и удобен в обращении, но имеет один существенный недостаток — раскрыть его может каждый школьник. Поэтому не будем на нем останавливаться и сразу перейдем к шифру Виженера, либо Вернама, либо любому другому, в котором шифрование сводится к наложению одной последовательности на другую. Как признают все криптографы, при одноразовом использовании такого типа шифров дешифровать сообщение невозможно.

Простейший способ построить программу шифрования — реализовать генератор случайной последовательности, знаки которой можно последовательно складывать со знаками исходного сообщения. Для наложения, конечно, нужна не любая последовательность, а именно случайная. Что это такое? На первый взгляд нет проблем. Различных генераторов известно великое множество. Какой из них выбрать? Ответ очевиден — тот, который вырабатывает последовательность, наиболее близкую к случайной.

А теперь попробуйте дать определение случайной последовательности. Для эксперимента попросите кого-нибудь это сделать, и он наверняка скажет вам, что это такая последовательность, в которой каждый элемент случайно, независимо от других элементов и равновероятно может принимать каждое из возможных значений, или что-нибудь вроде «это последовательность, являющаяся реализацией схемы независимых испытаний», или — «полученная в результате выборки из множества всех последовательностей с равномерным распределением».

Недостаток использования такого рода определений для целей криптографии становится очевидным при взгляде на последовательность

$$0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.$$

Она является одной из реализаций схемы последовательных независимых испытаний и ни в чем не уступает любой другой последовательности, например,

$$1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1,$$

так как вероятности появления их на выходе удовлетворяющего таким определениям случайного датчика одинаковы. С точки зрения теории, разницы нет никакой, но попробуйте их использовать для шифрования.

Таким образом, наша случайная последовательность должна быть не просто получена по той или иной вероятностной схеме, но еще и быть похожей на случайную (ведь никому не придет в голову назвать первую из этих последовательностей случайной). Но кто может ответить, что значит «похожей»? На что похожа, например, последовательность

3876504335317997501469391035319109708663589625180623 0
2982289092672371151411524515556647925609871796831049 6
8360539125133039103105418470259112815585875597000563 5
6937703949226241396723616837470247248135048208451745 4
3990212200528238143667515252273?

Попробуйте отличить ее от случайной последовательности. Вместе с тем, специалист по теории чисел скажет, что это десятичное представление простого числа $2^{829} + 1$. А теперь подумайте, как эта последовательность будет выглядеть в двоичной записи.

Каковы возможные критерии похожести? Легко сформулировать условия похожести одной последовательности на другую. Но как сформулировать, что значит «последовательность похожа на случайную последовательность»? Эта проблема не имеет однозначного ответа. Есть много подходов к определению понятия «похожести», но каждый из них страдает односторонностью. Поэтому от выбранного вами подхода во многом зависит качество шифрования.

Удобно ли носить большую связку ключей?

Перед вами программа для шифрования. Что является секретным в этой программе? Сама программа? Исходный текст? Описание алгоритма? Ответ, безусловно, зависит от условий ее применения. Конечно, лучше всего сделать так, чтобы ваш конкурент не имел ни того, ни другого, ни третьего. Поэтому обычно подобные программы защищают и от копирования, и от дизассемблирования, и от работы под отладчиком и т. д. Если она защищена грамотным специалистом, то пройдет

не один месяц, пока защита будет снята. Тем не менее, программа, как правило, используется длительное время, а хакеров сейчас великое множество. Поэтому лучше сразу исходить из того, что все связанное с программой секрета на представляет.

Что еще представляет интерес для вашего конкурента? Закрытая программой информация? От нее никто не откажется, но она мало что дает. Если алгоритм шифрования выбран надежный, то закрытая информация не представляет секрета, так как она непосредственно предназначена для хранения и передачи в открытом виде. Поэтому при отсутствии следов не уничтоженной открытой информации можно быть спокойным.

Остаются ключи. Имея ключи, ваши конкуренты откроют все закрытые двери. Это главный секрет во всем процессе шифрования, и, можете не сомневаться, что основные усилия вашего конкурента будут направлены именно на подкуп персонала, имеющего доступ к ключам.

Но как вводить ключ в программу и где хранить ключи? Если программу использовать интенсивно, то ключей нужно очень много. Где их брать и как заменять один на другой?

Скупой рыцарь хранил свои сокровища в сундуках, которые были закрыты с помощью замков, ключи от которых он постоянно носил с собой. Он правильно делал, что никогда не расставался с ключами. Но поскольку он был очень богат, то ключей у него было очень много, и носить такую связку ему, наверно, было очень неудобно. Вообще-то, он мог завести еще один сундук, куда можно было бы сложить все ключи. Тогда бы ему потребовалось носить всего один ключ. Именно эта идея используется в ключевых системах с главным ключом, когда все ключи, кроме главного, хранятся в компьютере на винчестере в зашифрованном виде. Хотя такой способ хранения ключей несколько усложняет процедуру доступа к зашифрованной информации, преимущества здесь неоспоримы. Действительно, можете ли вы представить себе человека, который помнит несколько сотен ключей?

Таким образом, задача сводится к обеспечению надежного хранения всего одного ключа. Правда, при этом ценность этого ключа резко возрастает и носить его в кармане становится очень опасным. Не только потому, что наверняка найдутся люди, которые будут испытывать непреодолимое желание взять у вас его, хотя бы на время; но еще и потому, что если он потеряется или испортится, получив механическое повреждение, то вы уже никогда не сможете ничего восстановить из той информации, которую вы хранили в своем компьютере. С ключами надо быть очень осторожным.

В 1976 году американские математики Диффи и Хеллман предложили перейти к системам с открытым ключом. Давайте издадим книгу-

справочник с открытыми ключами всех корреспондентов и поместим ее на общедоступном сервере. Это полностью снимет проблему с хранением большого числа ключей.

Нетрудно догадаться, сколь неоднозначный и шокирующий эффект это предложение произвело на криптографическую общественность. Но уж больно заманчивы были те возможности, которые предоставляла такая система. Как показали авторы идеи, такая система дает возможность подписывать сообщения электронной цифровой подписью, которая выполняет роль обычной подписи под документом. Таким образом, сразу снимался тяжкий груз по производству, запоминанию, хранению, распределению и уничтожению огромного количества секретных ключей, да к тому же открывались новые возможности и сферы применения.

Более того, плодотворность идеи выразилась в проявлении большого интереса к ней у значительной части крупных математиков. Ведь для реализации такой системы нужны были труднорешаемые математические задачи. А кто, кроме самих математиков, может придумать такие задачи!

Посмотрим, например, как организуется выработка ключей в широко распространенном протоколе SSH (Secure Shell). Данный протокол обеспечивает аутентификацию и закрытие коммуникаций при взаимодействии UNIX машин. Для выработки ключей в нем используется протокол, являющийся модификацией хорошо известного протокола обмена ключами, предложенного Диффи и Хеллманом. Пусть p — большое простое число и $q = (p-1)/2$. Пусть g — число, имеющее порядок равный q по модулю p. Оно является образующим элементом мультипликативной группы порядка q. Помимо операции возведения в степень в протоколе используются алгоритмы вычисления функции хеширования SHA (secure hash algorithm) и цифровой подписи DSS (digital signature standard), принятые в США в качестве стандартов. Протокол состоит в следующем (см. [6]).

1. Клиент C генерирует случайное число x, $1 < x < q$, вычисляет значение $e = g^x \pmod p$ и отправляет сообщение «e» серверу S.

2. Сервер S генерирует случайное число y, $1 < y < q$, вычисляет значение $f = g^y \pmod p$, вычисляет значение ключа $K = e^y \pmod p$, вычисляет проверочное значение $H = hash(V, K_s, e, f, K)$ (здесь $hash$ — функция хеширования на основе алгоритма SHA, V — некоторая строка, содержащая идентифицирующую информацию о клиенте и сервере, K_s — открытый ключ сервера), вычисляет значение цифровой подписи m под H на своем секретном ключе в соответствии с алгоритмом DSS, а затем отправляет сообщение «(K_s, H, m, f)» клиенту C.

3. Клиент C проверяет, действительно ли K_s является ключом сервера. Если да, то вычисляет значение ключа $K = f^x \pmod p$, проверяет правильность значения вектора H. Наконец, проверяет подпись под этим значением.

В этом протоколе простое число p строится из известного каждому школьнику числа π по формуле

$$p = 2^{1024} - 2^{960} - 1 + 2^{64} \left\lfloor 2^{894}\pi + 12903 \right\rfloor,$$

и равно

179769313486231590770839156793787453197860296048756011706444
423684197180216158519368947833795864925541502180565485980503
646440548199239100050792877003355816639229553136239076508735
759914822574862575007425302077447712589550957937778424442426
617334727629299387668709205606050270810842907692932019128194
467627007.

Его шестнадцатиричная запись имеет «менее случайный» вид

```
FFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1
29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD
EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245
E485B576 625E7EC6 F44C42E9 A637ED6B 0BFF5CB6 F406B7ED
EE386BFB 5A899FA5 AE9F2411 7C4B1FE6 49286651 ECE65381
FFFFFFFF FFFFFFFF.
```

Данное число построено Ричардом Шреппелем из университета штата Аризона, а его свойства описаны в работе [7].

В качестве образующего элемента для этого простого числа взято число $g = 2$. Возведение в степень такого числа выполняется достаточно быстро, так как умножение на 2 — это простой сдвиг двоичной записи числа в сторону старших разрядов.

49783156431138-я попытка

Теперь встанем на сторону нападающей стороны и попробуем написать программу для нахождения неизвестного ключа. Пусть у вас есть исходный и закрытый файлы. Если это не так, то надо постараться получить хотя бы общее представление о том, что представляет собой исходный файл: исполняемый код, текст программы на каком-либо языке программирования, текст в кодировке ASCII, ANSI, KOI-8 и т. п. Если даже это неизвестно, то возможность определения правильного ключа зачастую вообще становится проблематичной. Но не будем о грустном. Пусть у вас есть оба файла. Воспользуемся написанной ранее программой шифрования и добавим к ней блоки генерации ключей и сравнения полученного зашифрованного текста с имеющимся открытым. Прежде

чем запустить на исполнение программу, подумаем, сколько времени она может работать.

Сначала найдем среднее число знаков текста, которое требуется для отбраковки случайного ключа. Предположим, что наша программа последовательно сравнивает биты текста, полученного шифрованием с помощью случайно выбранного ключа, с битами имеющегося закрытого файла. Если реализованный в программе шифр обладает хорошими свойствами и дает шифртекст, похожий на случайную последовательность, то по первому биту будет забракована приблизительно половина ключей, по второму — еще четверть, и т. д. В итоге мы получим, что среднее число знаков для отбраковки оценивается бесконечной суммой

$$\sum_{i=1}^{\infty} \frac{i}{2^i}.$$

Математики знают, что эта сумма равна двум. Поэтому, как это ни странно, потребуется в среднем всего два бита текста для отбраковки ложного случайного ключа. Это один из первых парадоксов криптографического анализа. А ведь для того, чтобы убедиться в том, что ключ найден правильно, надо расшифровать весь текст.

Данный результат обнадеживает своей неожиданностью. Поэтому сразу переходим ко второй части задачи и найдем, сколько потребуется в среднем перебрать вариантов ключа до появления искомого. Считаем, что все ключи равноправны. Перенумеруем их и начнем по очереди проверять с помощью нашей программы. На первый взгляд кажется, что от выбора порядка нумерации ключей зависит очень много. Действительно, если у вас есть дар предвидения и вы выбрали такой порядок, что искомый ключ имеет первый номер, то программа сработает уже на первом шаге. Но не будем верить в чудо и найдем среднее число шагов для случайно выбранного порядка. Оно равно

$$\sum_{i=1}^{n} \frac{ik_i}{n!},$$

где k_i — число различных упорядочений множества из n ключей, у которых истинный ключ стоит на месте с номером i. Так как легко подсчитать, что $k_i = (n-1)!$, то наша сумма равна

$$\sum_{i=1}^{n} \frac{ik_i}{n!} = \sum_{i=1}^{n} \frac{i}{n} = \frac{n(n+1)}{2n} = \frac{(n+1)}{2}.$$

Итак, в среднем придется перебрать чуть больше половины всех ключей.

Теперь вернемся от математики к суровой действительности. Пусть вы написали великолепную программу, которая проверяет за одну

секунду один миллион вариантов ключа. Тогда за час программа переберет 3600 000 000 ключей, за сутки — 86 400 000 000 ключей, а за год — более 30 000 000 000 000. Короче, для перебора 2^{56} ключей шифратора DES вашей программе потребуется в среднем чуть более 1500 лет. Вдохновляющий результат, не правда ли? А теперь подсчитайте, сколько лет потребуется вашей программе для нахождения неизвестного ключа у отечественного алгоритма шифрования ГОСТ 28147, в котором число ключей равно 2^{256}.

Подобные оценки помогут вам избежать излишеств и правильно выбрать длину ключа для вашей программы.

А теперь перейдем к решению практических вопросов.

3. Как зашифровать файл?

Руки прочь от моих файлов!

Представьте себе: вы приходите домой после трудного дня и в прихожей вас встречает ваш младший брат, громко декламируя ваши стихи, посвященные любимой девушке из соседнего класса, которые вы вчера набрали на своем компьютере. И не только декламирует, а еще и комментирует, и из его комментариев следует, что Пушкина из вас ну никак не получится.

Вы, конечно, можете объяснить брату, что он не прав, подкрепив свои слова парой подзатыльников и другими весомыми аргументами. Это называется *административные меры защиты.*

Вы можете поставить пароль на включение компьютера. Как думаете, сколько времени продержится такая защита? Правильно. Дня три-четыре. Потом ваш брат замучает вас просьбами разрешить ему поиграть в WarCraft, и вам придется поделиться с ним паролем.

Было бы хорошо сделать так, чтобы ваш брат мог бы беспрепятственно играть в любимые игры, а доступ к чужим файлам был бы для него закрыт. Это называется разграничение доступа.

Как же организовать разграничение доступа на своем компьютере? Если бы у вас была установлена не Windows 95, а Windows NT или UNIX, вы легко могли бы ограничить доступ к своим файлам. В UNIX вам пришлось бы набрать команду `chmod` с нужными параметрами, а в Windows NT хватило бы нескольких движений мышью. Но у вас стоит Windows 95, а Windows 95 разграничение доступа не поддерживает. Так что же вам делать? Неужели нельзя защитить свои файлы от других пользователей?

Можно. Тут нам на помощь приходит криптография. Если вы не хотите, чтобы другие читали ваш файл, этот файл нужно зашифровать. Пусть его теперь читают все кому не лень. Все равно, если кто-

то, кроме вас, прочитает этот файл, он увидит не содержимое файла, а шифртекст. Конечно, используемый шифр должен быть достаточно стойким.

Как вводить ключ?

Как вы собираетесь вводить ключ в программу шифрования? Проще всего — с клавиатуры. Именно этот вариант реализован в большинстве готовых программ, и знакомство с ними начинается с вежливого приглашения: «enter your password: ... ». Только имейте в виду, что пароль, вводимый с клавиатуры, можно подсмотреть. Если программа шифрования при вводе пароля отображает его на экране, такой программой лучше не пользоваться. Когда при зашифровании файла вы вводите пароль, он тоже не должен отображаться на экране.

А что случится, если при вводе пароля вы случайно нажали не ту клавишу? Вы думаете, что ввели один пароль, а на самом деле ввели другой. Пробуете расшифровать файл, а программа говорит: «Пароль неправильный». Чтобы такого не было, обычно программы шифрования при вводе пароля для зашифрования файла просят ввести пароль дважды. Если пользователь в первый раз ввел один пароль, а во второй раз — другой, значит, по крайней мере один раз он ошибся. А если оба раза пользователь ввел одно и то же, значит, все нормально. Вряд ли пользователь дважды ошибется одинаково.

Как проверять правильность ключа?

Кстати, а как программа при расшифровании файла определяет, что пароль неправильный? По-разному. Некоторые программы вообще не проверяют правильность пароля. В этом случае, если вы ввели неправильный пароль, файл как бы расшифруется, но вы увидите совсем не то, что зашифровали. Это неудобно. Предположим, вы зашифровали файл мегабайт в 50 с помощью алгоритма ГОСТ. Сколько времени он будет расшифроваться? Если у вас дома стоит обычный Pentium, то, по крайней мере, минуту. А скорее всего, минуты три. Вы все это время сидите, ждете, а потом оказывается, что зря ждали — пароль-то ввели неправильный. А если вы exe-файл неправильно расшифровали, а потом запустили на выполнение? Скорее всего, придется давить Reset. Так что лучше, когда перед расшифрованием программа проверяет правильность пароля.

Остается вопрос: как проверять правильность пароля? Можно просто вписать пароль в начало зашифрованного файла перед шифртекстом. При расшифровании вы вводите пароль, программа читает начало зашифрованного файла и сравнивает то, что вы ввели, и то, что в файле. Если совпало, значит, пароль правильный. А если не

совпало — неправильный. Просто и удобно. Только что произойдет, если кто-нибудь другой случайно просмотрит зашифрованный файл, например, с помощью Norton Commander? Весь файл — сплошная абракадабра, а в начале файла — осмысленное слово. Не нужно быть семи пядей во лбу, чтобы догадаться, что это и есть пароль. Так что нужно придумывать что-то другое.

Очевидно, эталон пароля, который хранится в зашифрованном файле, тоже надо зашифровать. Только как? Проще всего шифровать пароль по той же схеме, что и текст исходного файла. Если шифр стойкий (а иначе его и не стоит использовать), пароль будет закрыт надежно. А что брать в качестве ключа? Можно взять константу. Тогда в каждом зашифрованном файле эталон пароля будет зашифрован на одном и том же ключе. Так, например, делает встроенная система шифрования файлов электронной почты Sprint Mail. Только там зашифрованный пароль хранится не в начале зашифрованного файла, а в конце. Но что будет, если кто-то узнает ключ, на котором шифруются все пароли? Он сможет расшифровывать все файлы, которые вы зашифруете. И не важно, что пароли разные — злоумышленник возьмет нужный пароль прямо из файла, который хочет прочесть.

Вы, скорее всего, подумали: а откуда злоумышленник узнает ключ, на котором шифруются эталоны паролей? Этот ключ встроен в программу шифрования, его никто не знает, даже вы его не знаете. Тем не менее, если злоумышленник достаточно квалифицирован, если он умеет пользоваться дизассемблером и отладчиком, этот ключ он узнает без труда. Обычно на решение такой задачи уходит всего несколько часов. Как это можно сделать — тема отдельной статьи. Пока поверьте на слово — имея только exe-файл, определенные навыки и много свободного времени, можно разобраться в том, что делает программа, до мельчайших подробностей.

Лучше шифровать эталон пароля сам на себе. Взять пароль в качестве открытого текста и взять тот же пароль в качестве ключа. При расшифровании файла, когда вы вводите пароль, программа пытается расшифровать только начало файла. Если в результате получилась строка, совпадающая с паролем, значит, пароль правильный, и можно расшифровывать файл дальше. А если получилось что-то другое, значит, пароль неправильный.

Некоторые программы шифрования шифруют пароли иначе. Берется какая-то строка, всегда одна и та же, шифруется на пароле и записывается в зашифрованный файл. Diskreet, например, шифрует на пароле строку «ABCDEFGHENRIXYZ» (эта строка завершается нулевым байтом, как принято в языке C). Когда Diskreet проверяет пароль, он берет начало зашифрованного файла (точнее, байты с 16-го по 31-й) и расшифровывает их на пароле, который ввел пользователь. Если после расши-

фрования получилось «ABCDEFGHENRIXYZ» — пароль правильный, если не получилось — неправильный.

На первый взгляд кажется, что эта схема хуже, чем предыдущая. Действительно, в предыдущем случае злоумышленнику неизвестны ни открытый текст, ни ключ, а в последнем случае неизвестен только ключ, а открытый текст известен. Но если вы еще помните, что было написано в начале этой главы, то, наверное, уже поняли, что последняя схема ничуть не слабее предыдущей. Если шифр стойкий, то ключ шифрования невозможно получить за приемлемое время, даже если известны и открытый текст, и шифртекст.

Какой должен быть пароль?

Какой длины должен быть пароль, чтобы защита была стойкой? Число различных вариантов пароля должно быть не меньше числа различных ключей. Если вы шифруете файл с помощью алгоритма ГОСТ, а для пароля используете только строчные английские буквы и хотите, чтобы стойкость защиты была не ниже стойкости ГОСТа, то длина пароля должна быть не меньше, чем $\log_{26} 2^{256} = 54{,}46298970967$. (Здесь 2^{256} — число различных ключей ГОСТ, а 26 — число различных английских букв.) Так что вам придется придумывать 55-буквенный пароль. К тому же учтите, что если вы используете в качестве пароля не хаотичную последовательность букв, а осмысленную фразу, то нужно сделать поправку на избыточность языка. Если в пароль входят не только строчные буквы, но и заглавные, то для обеспечения необходимого числа ключей ГОСТ достаточно 51 символа. Только имейте в виду, что некоторые программы шифрования, получив пароль, преобразуют все его буквы к одному регистру. Например, Diskreet делает все английские буквы, входящие в пароль, заглавными. Вы можете использовать в пароле русские буквы, но будьте осторожны! Не все программы шифрования корректно работают с русскими паролями. Таблица 1 поможет вам оценить требуемую длину пароля в различных ситуациях.

Имейте в виду, что данные в этой таблице относятся к тому случаю, когда в качестве пароля берется равномерно распределенная случайная последовательность символов. Если в качестве пароля вы используете только осмысленные слова и фразы, количество возможных вариантов пароля будет гораздо меньше. Если в качестве пароля используется длинная фраза русского языка, то, как показывают теоретико-информационные исследования, количество возможных вариантов будет равно не 33^n, где n — число символов во фразе, а всего лишь 2^n (эта приближенная оценка верна только для больших n). Так что в этом случае для достижения стойкости DES придется брать пароль длиной 56 символов, а для достижения стойкости ГОСТ — 256 символов.

Таблица 1.

Алфавит	Мощность алфавита	Количество вариантов 6-символьного пароля	Длина пароля для достижения	
			стойкости DES $=2^{56}=$ $=7{,}21\cdot10^{16}$	стойкости ГОСТ $=2^{256}=$ $=1{,}16\cdot10^{77}$
строчные английские буквы	26	$3{,}09\cdot10^{8}$	12	55
строчные русские буквы	33	$1{,}29\cdot10^{9}$	12	51
строчные и заглавные английские буквы	52	$1{,}97\cdot10^{10}$	10	45
строчные и заглавные английские буквы и цифры	62	$5{,}68\cdot10^{10}$	10	43
строчные и заглавные русские буквы	66	$8{,}27\cdot10^{10}$	10	43
строчные и заглавные русские буквы и цифры	76	$1{,}93\cdot10^{11}$	9	41
строчные и заглавные английские буквы, цифры и знаки препинания	94	$6{,}90\cdot10^{11}$	9	40
строчные и заглавные русские буквы, цифры и знаки препинания	108	$1{,}59\cdot10^{12}$	9	38
все алфавитно-цифровые символы русифицированной клавиатуры	160	$1{,}68\cdot10^{13}$	8	35

Так стоит ли вообще использовать для шифрования пароль, вводимый с клавиатуры? Для ответа на этот вопрос надо определить для себя — от кого вы собираетесь защищать информацию. Если ваш противник умеет только подбирать пароль с клавиатуры, то в качестве пароля лучше всего взять осмысленное слово длиной 6–8 символов. Главное, чтобы злоумышленнику было трудно догадаться до этого слова. При этом надо помнить, что нельзя использовать в качестве пароля:

- свое имя (фамилию, отчество, прозвище, ...);
- свою дату рождения (номер телефона, номер паспорта, ...);
- имя того файла, который вам надо зашифровать;
- и все другие пароли, которые легко угадать.

Если ваш противник, от которого вы защищаете информацию, умеет программировать, то он может написать программу, которая будет подбирать пароль автоматически. Даже если противник не умеет программировать, он может взять такую программу из Internet — там таких программ много. В этом случае пароль не должен быть осмысленным словом. В современном английском языке обычно употребляется всего около 100 000 слов, в русском — чуть больше. Перебрать 100 000 паролей можно очень быстро. Если в качестве шифра используется DES, на процессоре Pentium можно перебрать все английские слова-пароли всего за несколько секунд.

Даже если пароль представляет собой слово, которого нет в словаре, его все равно можно легко угадать. Вы уже знаете, что порядок букв в словах и фразах естественного языка подчиняется определенным статистическим закономерностям. Например, в русском языке комбинация букв ий встречается часто, а оь — никогда. Для большинства естественных языков статистика встречаемости символов документирована. Если программа перебора вначале подбирает наиболее вероятные пароли, а менее вероятные оставляет на потом, то перебор сокращается в десятки и сотни раз. Один из авторов видел, как подобная программа подобрала пароль natenok на компьютере с процессором 386DX–40 всего за 10 минут. Общая сложность перебора была равна $8{,}03 \cdot 10^9$. Вот другие известные авторам случаи удачного подбора паролей:

Сложность перебора[1)]	Время подбора	Тип процессора
$2{,}08 \cdot 10^{11}$	15 минут	486DX/4–100
$5{,}68 \cdot 10^{10}$	8 часов	Pentium–120

В первом случае пароль представлял собой два английских слова, записанных подряд без пробела. Одно из них было трехбуквенным,

[1)] Имеется в виду сложность тотального перебора. Поскольку реально проводился оптимизированный перебор, его сложность была гораздо меньше. Этим и объясняется несоответствие сложности перебора и времени подбора.

другое — пятибуквенным. Во втором случае пароль состоял из трех строчных английских букв, двух заглавных английских букв и одной цифры. Этот пароль был абсолютно бессмысленным. Выводы делайте сами.

Лучший результат по подбору ключа был достигнут в 1997 году, когда в сети Internet был дешифрован файл, зашифрованный с помощью DES. В подборе ключа участвовали десятки тысяч пользователей Internet. Все множество ключей было разбито на непересекающиеся подмножества и каждый перебирал ключи из своего подмножества. Перебор длился несколько недель. Руководил работой добровольной «виртуальной» бригады взломщиков со своего сервера программист Рокки Версер — автор программы, перебирающей ключи. Общая сложность перебора составляла $7{,}21 \cdot 10^{16}$, но ключ был найден после перебора всего 25% ключей. При этом «расколол» сообщение компьютер с процессором Pentium/90 с 16 Мбайт оперативной памяти. Оно гласило: «Надежная криптография делает мир безопасным».

А можно ли обойтись без пароля?

Как видите, применять пароль в качестве ключа шифрования не так удобно, как кажется на первый взгляд. Либо защита будет нестойкой, либо пароль трудно запомнить и ввести без ошибок. Если вы хотите построить по-настоящему надежную защиту, вместо пароля нужно использовать что-то другое.

Можно, например, хранить ключ на дискете[1]. Вы создаете каким-то образом (каким — обсудим позже) случайный, равновероятный и достаточно длинный ключ и записываете его на дискету. Когда программа шифрования запрашивает ключ, вы вводите ключ не с клавиатуры, а с дискеты. Вы просто вставляете дискету в дисковод, а программа считывает оттуда ключ и зашифровывает файл на этом ключе. При расшифровании файла программа просит «Вставьте ключевую дискету в дисковод». Вы вставляете дискету в дисковод, программа считывает оттуда ключ, проверяет его правильность и, если ключ правильный, расшифровывает файл.

Конечно, программа шифрования должна уметь работать с ключевыми дискетами.

Ключевую дискету нужно хранить в месте, недоступном для злоумышленников. Если кто-то воспользуется вашей ключевой дискетой, то он сможет прочесть все, что вы зашифровали с ее помощью. Не теряйте ключевые дискеты! Если вы потеряете такую дискету, то тем

[1] Или на электронном ключе Touch Memory, или на пластиковой карте. Только в этом случае нужно, чтобы в вашем компьюьере было соответствующее устройство ввода.

самым вы потеряете все данные, которые вы зашифровали с ее помощью.

Удобно ли пользоваться ключевыми дискетами? Нет. Но за надежность защиты приходится платить. Вообще, это общая закономерность — *чем надежнее система защиты, тем труднее с ней работать*. И наоборот, чем система защиты удобнее в обращении, тем она слабее. Конечно, разрабатывая систему защиты, можно наделать ошибок, и система получится и ненадежной, и неудобной. Но если система разрабатывалась на совесть, приведенное выше утверждение почти всегда верно.

Можно ли как-нибудь защитить ключ, хранящийся на дискете? Конечно! Его тоже можно зашифровать. И не нужно для этого придумывать сложный шифр — вполне хватит самого примитивного шифра наподобие простой замены. Почему? Да потому, что ключ шифрования — текст случайный и равновероятный. На чем основан метод дешифрования шифра простой замены? На том, что открытый текст — это осмысленный текст. А в осмысленном тексте обязательно присутствуют статистические закономерности. Но в ключе никаких закономерностей нет — ключ случаен и равновероятен!

Что использовать в качестве ключа, на котором шифруется ключ? Можно использовать пароль, но, как вы уже знаете, это не очень надежно. Можно завести вторую ключевую дискету, на которой хранить ключ, необходимый для доступа к первой. Но тогда придется заводить третью дискету, чтобы защитить вторую, а потом заводить четвертую, чтобы защитить третью ...

Так как же быть? Подумайте сами. Один из возможных ответов вы найдете в конце главы.

Где взять ключи?

Каким должен быть ключ шифрования? Случайным и равновероятным. А как получить случайную и равновероятную последовательность символов? Правильно, с помощью генератора случайных чисел. Написать свой генератор «случайных» чисел очень просто. Хорошие по статистическим свойствам последовательности получаются по формуле *линейного конгруэнтного метода*:

$$r_{n+1} = ar_n + b \pmod m,$$

где r_i — i-й член псевдослучайной последовательности; a, b, m — некоторые целые числа.

Качество псевдослучайной последовательности зависит от выбора чисел a, b и m. Эти числа обязательно должны быть взаимно просты. Есть и другие правила выбора этих коэффициентов, о них можно прочитать в [1]. В Diskreet, например, используется следующий генератор

псевдослучайной последовательности:

$$r_{n+1} = 214013 r_n + 2531011 \pmod{2^{32}}.$$

Как видите, формула для получения очередного «случайного» числа *рекурсивна* — каждый член последовательности зависит от предыдущего. Возникает вопрос: откуда берется первый член? Обычно в качестве r_1 берут текущее время с точностью до тика таймера (0,054945 сек.). Если для генерации ключа используется линейный конгруэнтный метод, ключом является последовательность чисел $(r_1, r_2, \ldots, r_N)$, где

$$N = \frac{(\text{длина ключа})}{(\text{длина } r_i \text{ в байтах})}.$$

Предположим, что с помощью линейного конгруэнтного метода сгенерирована последовательность $(r_1, r_2, \ldots, r_{128})$, где каждое r_i есть короткое целое число (16 бит). Созданный ключ представляет собой случайную равновероятную последовательность длиной 256 байт. Оценим, сколько различных вариантов ключа можно получить по данной схеме.

Зафиксируем значение r_1 . Какие значения может принимать r_2? Только одно значение. Если r_1 фиксировано, то значение r_2 определено однозначно:

$$r_2 = ar_1 + b \pmod{m}.$$

Значение r_3 тоже определено однозначно. Оно равно

$$r_3 = ar_2 + b \pmod{m} = a(ar_1 + b \pmod{m}) + b \pmod{m}.$$

Таким образом, значение r_1 однозначно определяет значения всех следующих членов последовательности. Получается, что различных последовательностей $(r_1, r_2, \ldots, r_{128})$ в точности столько же, сколько различных значений r_1. В нашем примере r_1 — короткое целое число, принимающее значения от 0 до $2^{16} - 1 = 65535$. Оказывается, что стойкость ключевой системы (число различных вариантов ключа) равна не 2^{256}, а всего лишь 2^{16}, что в $1{,}77 \cdot 10^{72}$ раз меньше!

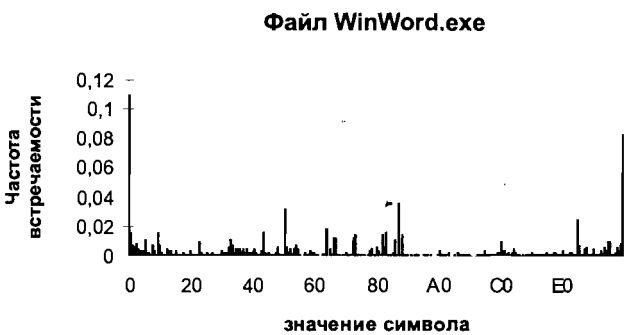
Получается, что псевдослучайные последовательности в качестве ключей использовать нельзя. А что можно?

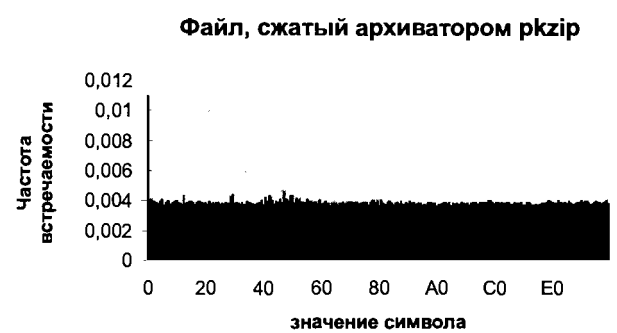
В чем слабость псевдослучайных последовательностей? В том, что они псевдослучайны. Первый член последовательности однозначно определяет остальные. Чтобы ключ был по-настоящему случайным и равновероятным, последовательность должна быть не псевдослучайной, а истинно случайной.

Где взять истинно случайную последовательность?

На первый взгляд, эта задача очень проста. Можно, например, вычислить случайный адрес памяти и взять оттуда данные. Или можно вычислить случайный номер сектора на диске и взять данные оттуда.

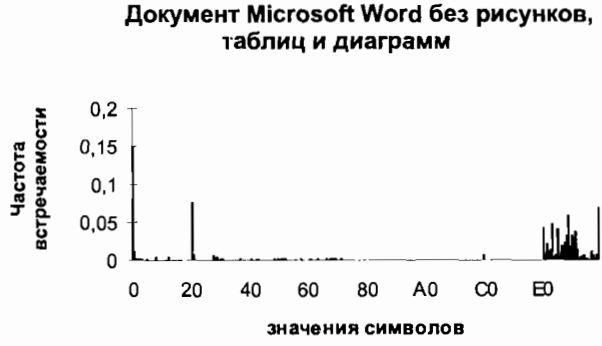
Эти данные, бесспорно, будут случайными. Но какое распределение будут иметь члены полученной случайной последовательности? Неизвестно. Если полученные случайные данные представляют собой фрагмент текстового файла, то распределение символов, представляемых байтами файла, будет одно, если это фрагмент машинного кода — совсем другое. Одно можно сказать точно — это распределение почти никогда не будет равномерным. Ниже приведены гистограммы встречаемости символов для различных видов информации.





Как видите, из всех приведенных гистограмм только последняя более-менее похожа на гистограмму равномерного распределения. Однако, если приглядеться повнимательнее, в ней можно заметить пик вдоль оси Y (впрочем, этот пик нетипичен для архивов pkzip — он объясняется тем, что выбранный архив включал в себя очень много коротких файлов).

Конечно, если распределение исходной последовательности задано априорно, путем несложного преобразования эту последовательность можно превратить в распределенную равномерно. Но подсчитать распределение символов для информации, хранимой в памяти и на дисках компьютера, не удается даже в том случае, если известно, откуда эта информация взялась. Посмотрите на гистограмму встречаемости символов в тексте этой главы. А теперь посмотрите на следующую гистограмму.



Как видите, распределения отличаются довольно сильно. Получается, что если в документе есть картинки, исходное распределение одно, а если нет — совсем другое.

Кроме того, в осмысленных текстах (в том числе и в текстах exe-файлов, баз данных и т. д.) обязательно присутствуют статистические закономерности более высокого порядка. Если построить гистограмму встречаемости биграмм и триграмм, она будет еще менее похожа на горизонтальную линию (что должно иметь место при равномерном распределении), чем гистограмма встречаемости отдельных символов. Например, в exe-файле, скомпилированном с помощью Visual C++, машинный код подавляющего большинства функций (подпрограмм) начинается байтами

55 8B EC 83 EC[1)]

[1)] Машинные команды

```
push ebp
mov ebp,esp
sub esp, ...
```

и заканчивается байтами

8B E5 5D C3[1].

Поэтому последовательность байтов, полученных из случайного места оперативной памяти или диска, *нельзя считать случайной*. В такой последовательности наверняка есть внутренние статистические зависимости.

Так где же взять случайную последовательность?

Возможно, у вас уже появилась мысль, что истинно случайную последовательность программным путем не получить. Это не так. Существует, по крайней мере, один способ, позволяющий получить истинно случайную последовательность байтов. Основная идея этого способа заключается в том, чтобы привлечь к процессу выработки случайной последовательности самого пользователя.

Пользователь работает с компьютером. Он двигает мышь, нажимает клавиши на клавиатуре. Попробуем взять в качестве элементов случайной последовательности интервалы между последовательными нажатиями клавиш. Какая получится последовательность?

Случайная? Несомненно. Равномерно распределенная? Вряд ли. Какое у нее будет распределение? Трудно сказать. Если пользователь набирает текст, получится одно распределение, если работает с Norton Commander — другое, если играет в Tetris — третье. Будут ли соседние элементы последовательности зависеть друг от друга? Скорее всего, да.

Кажется, что никаких преимуществ по сравнению с предыдущим методом нет. Однако в отличие от предыдущего метода в данном случае все перечисленные проблемы можно решить.

Для начала посмотрим, можно ли написать программу генерации ключа так, чтобы распределение исходной случайной величины было известно. Исходная случайная величина у нас — это продолжительность интервала между нажатиями клавиш. В разных ситуациях эта величина распределена по-разному. Выберем ситуацию, когда распределение этой случайной величины легко посчитать. Пусть, например, пользователь в процессе выработки ключа играет в Tetris. При игре в Tetris частота нажатий на клавиши мало зависит от пользователя, другими словами, все пользователи, играя в Tetris, нажимают на клавиши примерно одинаково часто (конечно, кроме пользователей, которые играют в Tetris первый раз в жизни).

---

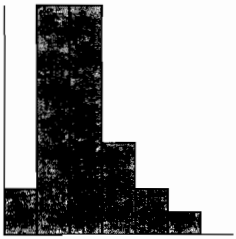
[1] Машинные команды

```
mov ebp,esp
pop ebp
ret
```

Распределение интервалов времени между последовательными нажатиями на клавиши несложно рассчитать. Для этого нужно прежде всего написать резидентную программу, которая перехватывала бы прерывание 16h, отвечающее за работу с клавиатурой, при каждом вызове прерывания получала бы текущее значение таймера, сравнивала его с предыдущим полученным значением, вычисляла разность и записывала полученное число в файл[1]. Затем нужно запустить эту программу и, пока она работает, поиграть некоторое время в Tetris. Проведите этот эксперимент в качестве упражнения.

Посмотрите на рисунок. Распределение нашей случайной величины будет выглядеть примерно так:

Распределение явно неравномерное, но это и не важно. Важно то, что распределение случайной величины при разных испытаниях примерно одинаково. Преобразовать это распределение в равномерное совсем несложно.



Остается решить последнюю проблему. Таймер компьютера тикает с частотой 18,2 раза в секунду, т. е. один тик занимает примерно 55 миллисекунд. Когда пользователь нажимает клавиши подряд, не думая, или просто держит клавишу нажатой, интервалы между последовательными нажатиями составляют 100–200 миллисекунд. Получается, что первый пик на приведенном графике на самом деле выглядит примерно так, как показано на рисунке.

[1] Некоторые версии Tetris используют для работы с клавиатурой прерывание 9. В этом случае написать такую программу немного сложнее.

А это нехорошо — в полученной случайной последовательности числа 2 и 3 будут встречаться гораздо чаще, чем любые другие. После преобразования к равномерному распределению значения этих часто встречающихся чисел изменятся, но сам факт наличия двух-трех значений, на которые приходится львиная доля наблюдений, останется.

Получается, что нам мешает дискретность таймера персонального компьютера. Как от нее избавиться? Подумайте сами. Один из возможных ответов вы найдете в конце главы.

4. Поучимся на чужих ошибках

Удобно, красиво, но ...

Рассмотрим примеры использования криптографических алгоритмов в широко распространенных программных продуктах.

В последнее время проблема защиты информации перестала быть головной болью только государственных структур, над нею начинают задумываться многие обычные пользователи ПК. Идя навстречу их пожеланиям, многие производители программ стали включать в свои продукты функции защиты данных. Однако в большинстве случаев разработчики таких программ не ставят своей целью использовать в них сколько-нибудь стойкие алгоритмы. Они считают основной своей задачей предоставить пользователям возможность защитить информацию либо от случайного несанкционированного доступа, либо от неквалифицированного взломщика. Программные продукты, о которых речь пойдет ниже, широко известны. Они, скорее, маскируют информацию, чем реализуют алгоритмы надежного криптографического закрытия.

Многие из вас, наверное, пользовались для редактирования документов программным продуктом Microsoft Word. Эта программа предоставляет пользователю широкий спектр возможностей для работы с документами, в том числе возможность сохранения информации в файлах в различном формате.

Если вы посмотрите в меню сохранения документов программы Word, то в параметрах этой операции обнаружите возможность указать пароль для доступа к документу. То есть для его открытия и дальнейшей работы с ним пользователь должен ввести пароль. Что происходит с документом, если в соответствующем поле параметров ввести пароль?

Для ответа на этот вопрос достаточно посмотреть на два документа, желательно идентичных по содержанию, но сохраненных с паролем и без него. В редакторе они будут выглядеть совершенно одинаково. Однако, откроем эти документы (они имеют расширение .doc) какой-нибудь программой просмотра файлов.

Мы увидим, что файл, представляющий документ в формате Microsoft Word, имеет сложную структуру. Он состоит из заголовка и нескольких разделов, которые описывают текст, а также содержат данные о работе пользователя с документом и служебную информацию. В одном из разделов файла, соответствующем документу без пароля, мы можем увидеть сам открытый текст. При этом в том же разделе файла с паролем мы обнаружим уже случайную последовательность символов. Оказывается, пароль использовался не только для разрешения доступа к документу при его открытии, но и являлся ключом некоторой криптографической схемы, зашифровавшей текст.

Очевидно, что если не зашифровывать текст, то сама идея использовать пароль была бы бесполезной. Любой пользователь сначала мог бы «вытащить руками» из файла большую часть текста, а потом перенести его в Word. В то же время, выбранная в Microsoft Word схема шифрования информации остановит только начинающего хакера [4]. Рассмотрим ее подробнее.

Из пароля пользователя Word вырабатывает массив длиной 16 байт, который назовем гаммой (`gamma[0..15]`). Далее, каждый байт открытого текста (`plain_text[i]`) последовательно складывается побитно (`XOR`) с байтом гаммы, в результате получаются знаки шифрованного текста

$$(\texttt{cipher\_text[i]}),$$

которые мы можем видеть в файле с паролем. То есть шифрование производится согласно формуле:

$$\texttt{cipher\_text[i]} := \texttt{plain\_text[i]}\ \texttt{XOR}\ \texttt{gamma}\,[\texttt{i} \bmod 16],$$

где mod 16 — операция получения остатка от целочисленного деления на 16.

Перед нами типичный пример криптографической схемы гаммирования короткой гаммой. Так как каждый шестнадцатый символ шифрованного текста получается прибавлением к символу открытого текста одного и того же значения гаммы, можно считать, что мы имеем дело с 16-ю простыми заменами. Для каждой из шестнадцати позиций символа в тексте подсчитаем таблицу частот его значений, после чего выберем в каждой из них значения символа, встретившегося чаще других.

Заметим, что самый частый символ в документе Word — это пробел (его значение в кодировке `ASCII` есть `0x20`). В этом легко убедиться, просматривая документ в шестнадцатиричном формате. Следовательно, самым частым символам в таблице частот соответствуют зашифрованные пробелы, и, складывая побитно значения этих символов с `0x20`, мы получим все 16 знаков гаммы. Далее, зная гамму, расшифровываем весь текст. Не правда ли, просто!

На эту очевидную слабость многие сразу обратили внимание. Поэтому фирма Microsoft для последних версий текстового процессора Microsoft Word, начиная с Word 97, полностью изменила алгоритм шифрования файлов, встроив в него хорошо известные алгоритмы шифрования RC4 и хеширования MD5.

Теперь посмотрим, как защищаются пароли пользователя в операционных системах (ОС) Microsoft Windows 95 первых версий (до OSR 2).

Большинство современных сетевых ОС являются многопользовательскими, это и Novell NetWare, и Microsoft Windows NT, и т. д. Для разграничения доступа пользователей к своим ресурсам эти ОС требуют от последних доказать свою подлинность. Делается это с помощью пароля, который известен и ОС, и пользователю. Ясно, насколько важно системе для обеспечения ее безопасности надежно и недоступно для постороннего доступа хранить информацию о паролях пользователей.

ОС Microsoft Windows 95 не является многопользовательской и не предоставляет возможность пользователям разделять свои ресурсы. Тем не менее, для удобства работы она запрашивает у пользователя при входе в систему его имя и пароль. Но если он ничего не ответит (нажмет кнопку «Cancel»), ОС все равно разрешит ему работать дальше. Для чего же тогда запрашивается пароль?

Дело в том, что ПК может работать в локальной вычислительной сети (ЛВС), где ему доступны ресурсы или серверы, для обращения к которым требуются пароли, причем, возможно, различные. Чтобы пользователю не нужно было их все запоминать, ОС Microsoft Windows 95 запоминает пароли для доступа к ресурсам ЛВС в специальном файле с именем «`имя_пользователя.pwl`». В этом файле данные шифруются на том самом пароле, который система запрашивает у пользователя при его входе в систему. Если пароль введен правильно, то в дальнейшем ОС сама подставляет соответствующий пароль при запросе пользователя на доступ к ресурсам или серверам ЛВС.

Данные в .pwl файлах шифруются следующим образом [5]. Из пароля пользователя по алгоритму шифрования RC4 вырабатывается гамма. Каждый пароль на доступ к соответствующему ресурсу вместе с некоторой служебной информацией суммируется побитно с полученной гаммой. То есть каждый раз при шифровании используется одна и та же гамма. Если учесть, что .pwl файл содержит зашифрованную запись, начинающуюся с имени пользователя, дополненного до 20 символов пробелами, то задача вскрытия пароля становится элементарной. Получив первые 20 знаков гаммы, мы можем прочитать любой сохраненный в файле пароль (учитывая то обстоятельство, что редко когда используются пароли длиной более 10 символов).

Следует отметить, что сам по себе алгоритм RC4 довольно сложный, и в данном случае использовались слабости не самого алгоритма,

а схемы его применения, а именно многократное использование одной гаммы.

Сколько дырок в вычислительных сетях?

При современном уровне развития компьютерных и информационных технологий даже обычный домашний ПК уже не мыслится отдельно от всего компьютерного киберпространства. Проникновение вычислительных сетей всюду, где есть компьютеры, стремление самих пользователей объединяться вынесло на передний план лозунг компании Sun Microsystem «Сеть — это компьютер». Разработчики современного программного обеспечения также стали ориентироваться на использование сетевых технологий и обеспечение пользователей удобными средствами для работы с распределенными ресурсами и удаленными источниками информации.

Многие из вас пользовались ресурсами глобальной сети Internet, кто-то имел возможность работать в локальных вычислительных сетях (ЛВС). Наверное, при этом вы задавали себе вопросы:

– как защищается информация, передаваемая по открытым каналам связи;

– можно ли, перехватывая данные, которыми обмениваются компьютеры в вычислительной сети, получить информацию о действиях работающих на них пользователей;

– можно ли вмешаться в протокол взаимодействия компьютеров;

– как защитить свои данные при их передаче по каналам локальных или глобальных сетей.

Мы не будем пытаться дать исчерпывающие ответы на поставленные вопросы, в этом случае мы вышли бы далеко за рамки данной главы. Попробуем просто посмотреть на примере существующих сейчас популярных систем, как эти вопросы решаются на практике.

Во-первых, как защищается информация, передаваемая по каналам ЛВС и в глобальных сетях? Для большинства распространенных сетевых ОС можно ответить — никак!

Информация между компьютерами передается в открытом виде по специальным коммуникационным протоколам. В этом легко убедиться, воспользовавшись специальными программами перехвата и анализа данных сетевого информационного обмена.

Такой программой является широко распространенный программный пакет LANalyzer for Windows фирмы Novell. Для его использования не требуется каких-то специальных навыков работы в ЛВС. Для начала перехвата информации достаточно нажать кнопку «Start», а для просмотра пойманных пакетов кнопку «View». Дополнительные возможно-

сти программы, например настройки фильтров перехвата, можно изучить и использовать в дальнейшем, по ходу дела.

Просмотрев улов, вы обнаружите, что каждый пакет имеет весьма сложную структуру, представляющую иерархию вложенных друг в друга сетевых протоколов. Но главное не в этом. Попробуйте перехватить пакеты, когда с одного компьютера на другой копируется текстовый файл. Без особого труда вы найдете файл в перехвате. Если он маленький, то он размещен в одном пакете, если большой, то в нескольких.

Например, пусть копируется файл `help.txt` с содержанием:

```
Help, I need somebody,
Help, not just anybody,
Help, you know I need someone, help.
```

Этот файл может выглядеть при передаче по каналам ЛВС так (пакет декодирован программой LANalyzer):

```
ip: ====================== Internet Protocol ======================
      Station:127.0.0.1 ---->127.0.0.2
      Protocol: TCP
      Version: 4
      Header Length (32 bit words): 5
      Precedence: Routine
             Normal Delay, Normal Throughput, Normal Reliability
      Total length: 194
      Identification: 12292
      Fragmentation not allowed, Last fragment
      Fragment Offset: 0
      Time to Live: 128 seconds
      Checksum: 0xB689(Valid)
  tcp: ================ Transmission Control Protocol ================
      Source Port: 1091
      Destination Port: NETBIOS-SSN
      Sequence Number: 20624641
      Acknowledgement Number: 849305
      Data Offset (32-bit words): 5
      Window: 7473
      Control Bits: Acknowledgement Field is Valid (ACK)
             Push Function Requested (PSH)
      Checksum: 0xCB85(Valid)
      Urgent Pointer: 0
Data:
   0: 00 00 00 96 FF 53 4D 42 2F 00 00 00 00 18 03 80 |.....SMB/.......
  10: 00 00 00 00 00 00 00 00 00 00 00 00 00 08 FE CA |................
  20: 00 08 00 06 0E FF 00 00 00 00 08 00 00 00 00 FF |................
  30: FF FF FF 00 00 57 00 00 00 57 00 3F 00 00 00 00 |.....W...W.?....
  40: 00 57 00 48 65 6C 70 2C 20 49 20 6E 65 65 64 20 |.W.Help, I need
  50: 73 6F 6D 65 62 6F 64 79 2C 0D 0A 48 65 6C 70 2C |somebody,..Help,
  60: 20 6E 6F 74 20 6A 75 73 74 20 61 6E 79 62 6F 64 | not just anybod
  70: 79 2C 0D 0A 48 65 6C 70 2C 20 79 6F 75 20 6B 6E |y,..Help, you kn
  80: 6F 77 20 49 20 6E 65 65 64 20 73 6F 6D 65 6F 6E |ow I need someon
  90: 65 2C 20 68 65 6C 70 2E 0D 0A                   |e, help...
```

Таким образом, если злоумышленник нашел возможность подключиться к ЛВС и установил необходимое программное обеспечение, то он без особого труда может собирать информацию о работе пользователей сети, запускаемых программных продуктах, содержании разрабатываемых документов и т. п.

Кроме того, злоумышленник иногда может вмешиваться в работу пользователей сети. В частности, некоторые атаки на сетевые ОС используют слабости протоколов идентификации получателей и отправителей информации в ЛВС. Дело в том, что каждый пакет, передаваемый по каналам ЛВС, снабжается электронными сетевыми адресами компьютера-отправителя пакета и компьютера, которому этот пакет предназначен. Это как адреса на почтовых конвертах.

В начале 90-х годов широко использовалась сетевая ОС Novell NetWare ver 3.11. Пользуясь слабостью реализованных в этой системе сетевых протоколов, злоумышленник с помощью специальной программы, подменяющей сетевые адреса, мог выдать свой компьютер за компьютер, на котором работает другой пользователь, в том числе Супервизор [2]. После чего мог давать системе любые запросы на использование ее ресурсов, заводить новых пользователей, устанавливать их права доступа. Данный способ подмены сетевых адресов получил название «Голландская атака» и широко описан в литературе.

Более того, при работе с Novell NetWare ver 3.11, пользуясь описанными выше слабостями, можно было создавать в ЛВС ложные серверы (компьютеры, где находится информация о пользователях и основные ресурсы сети) и направлять запросы других пользователей через свой компьютер.

Рассмотрим еще один важный вопрос, связанный с безопасностью сетевых ОС. В большинстве случаев компьютеры ЛВС объединяются в группы, которые в различных системах называются по разному: подсеть, домен, рабочая группа и т. д. Однако часто эти группы имеют некоторую общую структуру. В группе выделяется один (иногда несколько) мощный компьютер, называемый сервером или контроллером домена, который отвечает за общую безопасность. Остальные компьютеры (рабочие станции) предназначаются для работы пользователей сети.

Сервер несет ответственность за разрешение санкционированного доступа к ресурсам всей группы, например, к хранящимся на нем файлам или сетевым принтерам. Он отвечает и за допуск в сеть зарегистрированных пользователей.

Если пользователь хочет воспользоваться ресурсами компьютеров группы, он должен отправить на сервер информацию, подтверждающую свою подлинность. Как уже говорилось выше, чаще всего это делается с помощью пароля. Пользователь на рабочей станции вводит свои

имя и пароль, которые отправляются на сервер для проверки. Сервер сравнивает полученные данные с хранящимся у него эталонным паролем пользователя и по результатам проверки принимает решение о доступе пользователя к ресурсам группы.

Так поступают серверы большинства современных операционных систем. Но мы знаем, что информация, передаваемая по каналам локальных вычислительных сетей, может быть легко перехвачена злоумышленником. Следовательно, пароль пользователя должен передаваться с рабочей станции на сервер таким образом, что, даже перехватив его, им нельзя было воспользоваться. Здесь на помощь приходит криптография.

Разберем это на примере сетевой ОС Novell NetWare ver 3.11.

Заметим, что в версии 2.12 этой системы пароль пользователя вообще передавался в открытом виде. Данное обстоятельство делало все усилия по организации разграничения доступа к информации совершенно бесполезными.

В версии 3.11 был использован протокол передачи информации о пароле пользователя, состоящий из следующих шагов.

Шаг 1. Пользователь рабочей станции в ответ на запрос системы вводит свое имя и пароль.

Шаг 2. Рабочая станция преобразует пароль пользователя, вычисляя массив данных, называемый образом или сверткой пароля.

Шаг 3. Рабочая станция запрашивает у сервера разовый ключ (случайный массив данных, используемый однократно, для одного входа пользователя в систему).

Шаг 4. С использованием разового ключа и образа пароля пользователя рабочая станция вычисляет разовый билет на вход в систему, который отправляет на сервер для проверки.

Шаг 5. Сервер осуществляет проверку билета и направляет рабочей станции сообщение о ее результате. Если проверка прошла успешно, то пользователь получает разрешение на доступ в систему.

При реализации данного протокола решался ряд задач по обеспечению его безопасности, а именно:

– сделать максимально трудным подбор пароля пользователя злоумышленником, перехватившим в ЛВС разовый ключ и билет;

– добиться невозможности повторного использования билета для получения доступа к ресурсам системы.

Как видно из описания протокола, вторая задача была успешно решена. При каждом запросе пользователя на вход в систему билет вычислялся с участием разового ключа, а значит в дальнейшем он не повторялся и не мог быть использован еще раз.

Для решения первой задачи в качестве базового элемента протокола использовался криптографический алгоритм, участвующий в преобразовании пароля в его образ на *Шаге 2* и в вычислении разового билета на *Шаге 4*.

Алгоритм перерабатывает массив данных длиной 32 байта, его схема приведена ниже (см. рис. 1). На ней:

Array — массив данных из 32 элементов (байтов), после каждого шага работы схемы заполнение массива сдвигается на один элемент влево, а в *Array*[31] записывается результат побитного сложения;

Tab — фиксированная таблица из 32 элементов (байтов), после каждого шага работы схемы заполнение таблицы циклически сдвигается на один элемент влево;

Elem — один элемент памяти (байт);

XOR — операция побитного сложения байтов;

+ и − — операции сложения и вычитания, соответственно, байтов по модулю 256;

(mod 32) — операция получения остатка от деления на 32, результат этой операции используется как индекс для выборки элемента массива *Array*.

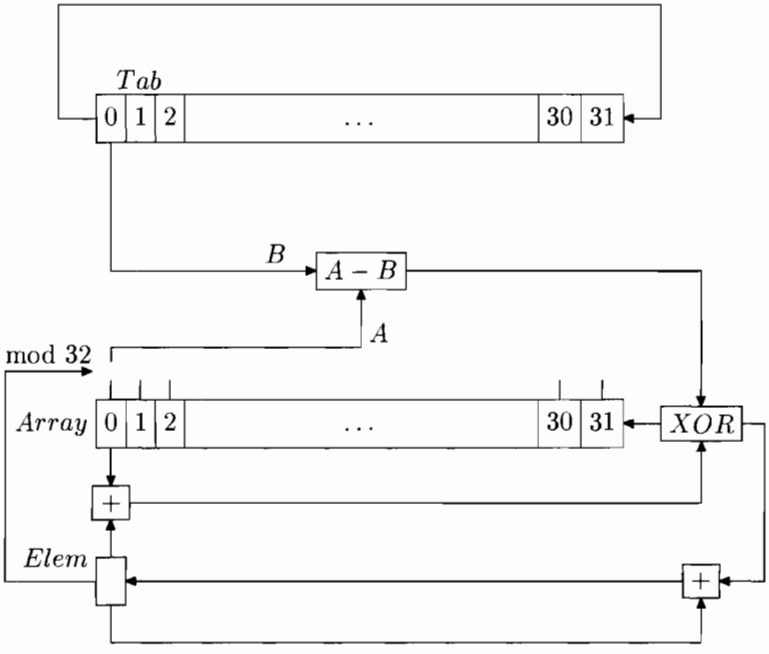


Рис. 1. *Схема криптографического преобразования, используемого ОС Novell NetWare.*

Схема работает 64 такта.

Несмотря на кажущуюся сложность схемы, оказалось, что она содержит ряд недостатков, позволяющих, при некоторых благоприятных условиях, подбирать пароль пользователя [2]. Например, злоумышленник, перехватив разовый ключ и билет, мог попытаться в лоб подобрать пароль пользователя. При этом он мог использовать словари наиболее часто встречающихся паролей, которые любой из вас может найти в Internet. Конечно, пароль не любого пользователя удастся таким образом подобрать. Но это слабое утешение.

Какой же может быть найден выход для пользователей, желающих надежно защитить свою информацию при ее передаче по открытым каналам вычислительных сетей?

Ответ здесь один — использование программ шифрования, реализующих надежные криптографические алгоритмы.

Здесь может быть предложено два основных подхода по организации системы шифрования:

– предварительное шифрование информации перед передачей ее по сети;

– прозрачное шифрование пакетов сетевого информационного обмена.

Первый подход наиболее удобен в глобальных информационных сетях, например, Internet. Суть его в том, что пользователи, предварительно обменявшись ключами, шифруют свои данные, а потом передают их по каналам сети стандартными средствами файлового обмена. Выгоды этого подхода очевидны, он позволяет пользователям, работающим с различными ОС, без особых затрат обмениваться шифрованными данными.

Так работают пользователи программы Pretty Good Private (PGP), разработанной Филиппом Зиммерманом в начале 90-х годов и широко распространенной во всем мире. PGP позволяет вырабатывать индивидуальные ключи пользователей, безопасно ими обмениваться и шифровать данные. В нем реализованы алгоритм блочного шифрования IDEA и схема открытого распределения ключей RSA.

Если вы не доверяете PGP (и это, по-видимому, правильно), то можете разработать свои собственные программы шифрования, взяв за основу только его общую схему.

Для работы в ЛВС эффективнее использовать второй подход. Связано это с тем, что в локальных сетях происходит достаточно интенсивный информационный обмен, и поэтому предварительное шифрование оказывается неудобным и занимает много времени.

Прозрачное шифрование пакетов сетевого информационного обмена обеспечивает защиту не только данных файлов пользователей, но и служебной информации, передаваемой по каналам ЛВС, о чем

пользователь может и не подозревать. Например, в этом случае будут шифроваться данные о пароле пользователя, передаваемые на сервер для проверки, данные о запросах пользователя на доступ к ресурсам системы, а также данные, посылаемые на печать.

Однако, данный подход значительно сложнее для реализации. Больше того, в настоящее время среди распространенных на рынке сетевых ОС нет ни одной, которая бы предоставляла пользователям такую возможность. Скорее всего, появление доступных систем прозрачного шифрования пакетов — дело ближайшего будущего.

В глобальных сетях дела с этим обстоят значительно лучше. Уже достаточно давно получили широкое распространение программные продукты, предоставляющие пользователям возможность прозрачного шифрования данных, передаваемых по сети. Например, Netscape Navigator, занимающий 75% рынка программ, предназначенных для работы с World Wide Web (WWW) сети Internet и проведения через Internet расчетов по кредитным карточкам, включает в себя криптографическую подсистему.

Netscape используют миллионы людей по всему миру, тем не менее, в этой программе были найдены существенные «дыры». Ранние версии Netscape содержали изъяны в двух основных элементах криптографической подсистемы [5]:

- собственно в самом алгоритме шифрования;
- алгоритме генерации ключей.

Первый изъян состоял в том, что для шифрования данных сетевого информационного обмена в программе Netscape в варианте, предназначенном для экспорта из США (а именно им большинство и пользуется), реализован алгоритм гаммирования RC4 [6] с ключом *40 бит*! Возможности современной вычислительной техники таковы, что ключ такой длины можно определить простым перебором в течении нескольких дней. Какое-то время среди пользователей сети Internet даже развернулось нечто вроде соревнования — кто быстрее сможет найти ключ.

В июле 1994 года ключ шифрования был восстановлен за 8 дней с использованием объединенных вычислительных ресурсов 120 рабочих станций и двух параллельных суперкомпьютеров.

В августе 1995 года данная задача была решена за то же время с использованием около 100 компьютеров.

В декабре 1995 года один суперкомпьютер подобрал ключ за 7 дней.

Вторая «дыра» в системе защиты была выявлена у программы Netscape (версия 1.2) двумя студентами Калифорнийского университета в сентябре 1995 года. Суть ее в том, что были обнаружены существенные слабости в алгоритме генерации ключей шифрования. Ключ

должен являться случайным числом. В Netscape для его вычисления был реализован алгоритм генерации «случайных» чисел, основанный на показании внутреннего таймера и значениях сетевых адресов. В результате анализа ими был предложен алгоритм подбора ключа в течение всего одной минуты, а программа, реализующая данный алгоритм, была опубликована в Internet.

5. Вместо заключения

Полезные советы

1. «Бойтесь данайцев, дары приносящих». Не верьте тому, кто заверяет вас, что он способен надежно защитить ваши секреты. Он либо безнадежный глупец, либо ваш конкурент. Специалист в любой ситуации будет высказывать сомнения в надежности предлагаемой им защиты и уж, по крайней мере, укажет вам ее возможные слабости.

2. Не будьте самоуверенны и не питайте иллюзий. Не рассчитывайте полностью на свой интеллект и сообразительность. Со временем вас осенит, и ваше остроумие подскажет вам, в чем слабости в предложенной вами конструкции велосипеда.

3. Как можно меньше рекламы. Не хвастайтесь своим алгоритмом и вообще, старайтесь как можно меньше показывать другим свою программу. Следуйте афоризму Дона Хуана: «Не расхваливайте в кругу друзей достоинств своей супруги. Один из них наверняка захочет в этом убедиться самостоятельно».

4. Не уподобляйтесь ученым. Даже если вам удалось доказать теорему о том, что задача вскрытия вашей системы защиты алгоритмически неразрешима, не расслабляйтесь. Попробуйте понять, почему не выполняются условия доказанной вами теоремы.

5. Защита защиты. Создав наилучшую систему защиты, проверьте, защищена ли она.

Ответ на первый вопрос.

Надо так написать программу шифрования, чтобы после N неудачных попыток ввода пароля на доступ к ключевой дискете ключ с дискеты стирался. Если число N слишком велико, злоумышленник, возможно, сумеет подобрать пароль. А если N слишком мало, может случиться так, что вы сами случайно испортите свою ключевую дискету. Целесообразно выбрать значение N в интервале $5 \leqslant N \leqslant 100$. Пароль на доступ к дискете не обязательно делать очень длинным и сложным. Главное, чтобы его нельзя было угадать за N попыток.

Ответ на второй вопрос.

Надо считать случайной величиной не интервал времени между последовательными нажатиями клавиш, а сумму n последовательных интервалов. Другими словами, новый член добавляется в случайную последовательность не при каждом нажатии пользователем клавиши, а при каждом n-ом нажатии. Число n можно взять в диапазоне $5 \leqslant n \leqslant 10$. Если n слишком мало, дискретность сохранится, а если слишком велико — ключ будет вырабатываться слишком медленно. Конечно, обе поставленные задачи можно решить и другими способами.

Литература к главе 6

[1] *Д. Кнут.* Искусство программирования для ЭВМ. Т. 2, М.: Мир, 1977.

[2] Теория и практика обеспечения информационной безопасности. Под редакцией П. Д. Зегжды. М.: Издательство Агентства «Яхтсмен», 1996. 192 с.

[3] *Р. Сайка.* 56-разрядный код DES «расколот» на персональном компьютере. Computerworld Россия, 29 июля 1997.

[4] `http://www.softseek.com/Business_and_Productivity/Microsoft_Office/Word_Add_Ons/Review_14603_index.html`

[5] `http://oliver.efri.hr/~crv/security/bugs/NT`

[6] *T. Ylonen, T. Kivinen, M. Saarinen.* SSH Transport Layer Protocol. Internet-Draft, November 1997. `draft-ietf-secsh-transport-03.txt.`

[7] *Orman H.*, The Oakley Key Determination Protocol. Version 1, TR97-92. Department of Computer Science Technical Report, University of Arizona.

Глава 7

Олимпиады по криптографии для школьников

Предисловие

С 1991 г. Институт криптографии, связи и информатики Академии ФСБ Российской Федерации проводит ежегодные олимпиады по криптографии и математике для школьников г. Москвы и Подмосковья. Год от года растет популярность этих олимпиад, о чем свидетельствует, например, то, что в последней из них приняло участие более пятисот школьников 9–11 классов. Олимпиады по криптографии и математике вызывают интерес у школьников необычностью своего жанра. Кроме того, призерам предоставляются льготы при поступлении в Институт.

Школьники часто спрашивают, с какой литературой по криптографии им следует познакомиться, чтобы успешно выступить на олимпиаде. Никаких специальных знаний для решения задач не требуется — в этом вы убедитесь, ознакомившись с задачами, которые приводятся в данной главе. Вместе с тем, мы не можем отрицать, что предварительное знакомство с криптографией полезно хотя бы чисто психологически, поскольку «внешний вид» задач может показаться необычным. Многие задачи нашей олимпиады — криптографические. Часть задач имеет криптографическую окраску, но их суть — математическая. Отдельные задачи — чисто математические.

При подведении итогов каждой олимпиады мы знакомим участников с общедоступными книгами по криптографии, которых в последние годы появилось достаточно много. Однако эти книги либо слишком сложны для школьников, либо поверхностны или недостаточно полны, либо малодоступны. Поэтому авторы настоящей главы поставили перед собой две основные цели: во-первых, предложить элементарное введение в криптографию, используя при этом чудесные детективные сюжеты известных произведений Ж. Верна, А. Конан Дойла, Э. По, В. Каверина, связанные с зашифрованными сообщениями; во-вторых, привести условия задач всех наших олимпиад с ответами и решениями.

1. Введение

Если вы хотите передать свое текстовое сообщение (последовательность символов некоторого алфавита) адресату так, чтобы оно осталось тайным для посторонних лиц, то у вас есть, по крайней мере, две возможности. Вы можете попытаться скрыть сам факт передачи текста, то есть прибегнуть к методам стеганографии, в арсенале которой — симпатические (невидимые) чернила, микроточки и тому подобные средства. Другая возможность заключается в попытке скрыть смысл сообщения от посторонних лиц, случайно или намеренно познакомившихся с передаваемым текстом. В этом случае вы можете прибегнуть к методам криптографии. Термин «криптография» происходит от двух греческих слов: «криптос» — тайна и «графейн» — писать, и означает тайнопись. «Тайнопись» как раз и подразумевает, что вы скрываете смысл своего сообщения.

Сообщение, которое вы хотите передать адресату, будем называть открытым сообщением. Например, в задаче 2.5 (раздел «Условия задач») одним из открытых сообщений является фраза:

КОРАБЛИ ОТХОДЯТ ВЕЧЕРОМ

Для сохранения сообщения в тайне оно преобразуется криптографическими методами и только после этого передается адресату. Преобразованное сообщение будем называть шифрованным сообщением (или зашифрованным сообщением). Другое название зашифрованного сообщения — криптограмма (или шифртекст). В задаче 2.5 зашифрованное сообщение выглядит так:

ЮПЯТБНЩМСДТЛЖГПСГХСЦЦ

Зашифрованное сообщение не обязательно должно быть последовательностью букв, как в указанной выше задаче. Часто зашифрованное сообщение может представлять собой последовательность цифр или специальных знаков (например, «пляшущих человечков»).

Процесс преобразования открытого сообщения в шифрованное будем называть *шифрованием* или *зашифрованием*. Адресату заранее сообщается, как из шифрованного сообщения получить открытое. Этот процесс получения исходного сообщения называют *дешифрованием или расшифрованием*.

При выборе правила шифрования надо стремиться к тому, чтобы посторонние лица, не знающие правила расшифрования, не смогли восстановить по криптограмме открытое сообщение. В этом случае вы скроете смысл сообщения и обеспечите «тайнопись».

Для удобства дальнейшего изложения обозначим буквой A — открытое сообщение, B — шифрованное сообщение, f — правило шифрования, g — правило расшифрования. В этом случае зашифрование открытого сообщения A в шифрованное сообщение B можно записать в виде

$f(A) = B$. Обратное преобразование (то есть получение открытого сообщения A путем расшифрования B) запишется в виде соотношения $g(B) = A$.

Правило зашифрования f не может быть произвольным. Оно должно быть таким, чтобы по шифртексту B с помощью правила расшифрования g можно было однозначно восстановить открытое сообщение A. Однотипные правила зашифрования можно объединить в классы. Внутри класса правила различаются между собой по значениям некоторого параметра, которое может быть числом, таблицей и т. д. В криптографии конкретное значение такого параметра обычно называют *ключом*. По сути дела, ключ выбирает конкретное правило зашифрования из данного класса правил.

Зачем понадобилось вводить понятие ключа? Есть, по крайней мере, два обстоятельства, которые позволяют понять необходимость этого. Во-первых, обычно шифрование производится с использованием специальных устройств. У вас должна быть возможность изменять значение параметров устройства, чтобы зашифрованное сообщение не смогли расшифровать даже лица, имеющие точно такое же устройство, но не знающие выбранного вами значения параметра. Во-вторых, многократное использование одного и того же правила зашифрования f для зашифрования открытых текстов создает предпосылки для получения открытых сообщений по шифрованным без знания правила расшифрования g. Поэтому необходимо своевременно менять правило зашифрования.

Используя понятие ключа, процесс зашифрования можно описать в виде соотношения

$$f_\alpha(A) = B,$$

в котором α — выбранный ключ, известный отправителю и адресату.

Для каждого ключа α шифрпреобразование f_α должно быть обратимым, то есть должно существовать обратное преобразование g_α, которое при выбранном ключе α однозначно определяет открытое сообщение A по шифрованному сообщению B:

$$g_\alpha(B) = A.$$

Совокупность преобразований f_α и набор ключей, которым они соответствуют, будем называть *шифром*.

Среди всех шифров можно выделить два больших класса: шифры перестановки и шифры замены.

Шифрами перестановки называются такие шифры, преобразования из которых приводят к изменению только порядка следования символов исходного сообщения. Примером преобразования, которое может содержаться в шифре перестановки, является следующее правило. Каждая буква исходного сообщения, стоящая в тексте на позиции с

четным номером, меняется местами с предшествующей ей буквой. В этом случае ясно, что и исходное, и шифрованное сообщение состоят из одних и тех же букв.

Шифрами замены называются такие шифры, преобразования из которых приводят к замене каждого символа открытого сообщения на другие символы — шифробозначения, причем порядок следования шифробозначений совпадает с порядком следования соответствующих им символов открытого сообщения. В качестве примера преобразования, которое может содержаться в шифре замены, приведем такое правило. Каждая буква исходного сообщения заменяется на ее порядковый номер в алфавите. В этом случае исходный буквенный текст преобразуется в числовой.

Как правило, в задачах олимпиад шифр известен, а использованный ключ — нет. Для определения исходного текста по шифрованному при неизвестном ключе возможны два подхода: первый — определить ключ и затем найти исходное сообщение расшифрованием; второй — найти исходное сообщение без определения ключа. Получение открытого сообщения по шифрованному без заранее известного ключа называется *вскрытием шифра*, в отличие от расшифрования — когда ключ известен. Во многих случаях вскрытие шифра возможно, что и демонстрируют победители наших олимпиад.

Под стойкостью шифра, как правило, понимается способность противостоять попыткам провести его вскрытие. При анализе шифра обычно исходят из принципа, сформулированного голландцем Огустом Керкгоффсом (1835–1903). Согласно этому принципу при вскрытии криптограммы противнику известно о шифре все, кроме используемого ключа. Одной из естественных характеристик шифра является число его возможных ключей. Ведь вскрытие шифра можно осуществлять перебором всех возможных его ключей. Мы уже говорили, что в приводимых ниже задачах олимпиад, как правило, шифр известен, но неизвестен выбранный ключ, что соответствует принципу Керкгоффса. Так, в задаче 4.4 все дело сводится к перебору 24 различных вариантов ключа, из которых только один дает читаемый текст. Поэтому многие участники олимпиады смогли восстановить сообщение на латинском языке, даже не зная этого языка.

Подчас смешивают два понятия: *шифрование* и *кодирование*. Мы уже договорились, что для шифрования надо знать шифр и секретный ключ. При кодировании нет ничего секретного, есть только определенная замена букв или слов на заранее определенные символы. Методы кодирования направлены не на то, чтобы скрыть открытое сообщение, а на то, чтобы представить его в более удобном виде для передачи по техническим средствам связи, для уменьшения длины сообщения и т.д. В принципе, кодирование, конечно же, можно рассматривать как

шифр замены, для которого «набор» возможных ключей состоит только из одного ключа (например, буква «а» в азбуке Морзе всегда кодируется знаками • — и это не является секретом).

В настоящее время для защиты информации широко используются электронные шифровальные устройства. Важной характеристикой таких устройств является не только стойкость реализуемого шифра, но и высокая скорость осуществления процессов шифрования и расшифрования. Для создания и обеспечения грамотной эксплуатации такой техники широко используются достижения современной криптографии, в основе которой лежат математика, информатика, физика, электроника и другие науки.

Современная криптография бурно развивается. В ней появляются новые направления. Так, с 1976 года развивается «открытая криптография». Ее отличительной особенностью является разделение ключей для зашифрования и расшифрования. При этом ключ для зашифрования не требуется делать секретным, более того, он может быть общедоступным и содержаться в телефонном справочнике вместе с фамилией и адресом его владельца. Подробнее об этом и других современных задачах криптографии можно прочитать в главах 1, 2, 3 этой книги.

Наряду с термином «криптография» в литературе встречается термин «криптология», также происходящий от греческих корней, означающих «тайный» и «слово». Этот термин используется для обозначения всей области секретной связи. Криптологию делят на две части: криптографию и криптоанализ. Криптограф пытается найти методы обеспечения секретности сообщений, криптоаналитик пытается при неизвестном ключе выполнить обратную задачу. При этом часто говорят, что криптоаналитик вскрыл шифр, хотя чаще он вскрывает ключ заранее известного шифра.

2. Шифры замены

Наиболее известными и часто используемыми шифрами являются шифры замены. Они характеризуются тем, что отдельные части сообщения (буквы, слова, ...) заменяются на какие-либо другие буквы, числа, символы и т. д. При этом замена осуществляется так, чтобы потом по шифрованному сообщению можно было однозначно восстановить передаваемое сообщение.

Пусть, например, зашифровывается сообщение на русском языке и при этом замене подлежит каждая буква сообщения. Формально в этом случае шифр замены можно описать следующим образом. Для каждой буквы α исходного алфавита строится некоторое множество символов M_α так, что множества M_α и M_β попарно не пересекаются при $\alpha \neq \beta$, то есть любые два различные множества не содержат

одинаковых элементов. Множество M_α называется множеством шифрообозначений для буквы α.

Таблица

а	*б*	*в*	...	*я*
$M_а$	$M_б$	$M_в$	...	$M_я$

(1)

является ключом шифра замены. Зная ее, можно осуществить как зашифрование, так и расшифрование.

При зашифровании каждая буква α открытого сообщения, начиная с первой, заменяется любым символом из множества M_α. Если в сообщении содержится несколько букв α, то каждая из них заменяется на любой символ из M_α. За счет этого с помощью одного ключа (1) можно получить различные варианты зашифрованного сообщения для одного и того же открытого сообщения. Например, если ключом является таблица

а	б	в	г	д	е	ж	з	и	к	л	м	н	о	п	р
21	37	14	22	01	24	62	73	46	23	12	08	27	53	35	04
40	26	63	47	31	83	88	30	02	91	72	32	77	68	60	44
10	03	71	82	15	70	11	55	90	69	38	61	54	09	84	45

с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
20	13	59	25	75	43	19	29	06	65	74	48	36	28	16
52	39	07	49	33	85	58	80	50	34	17	56	78	64	41
89	67	93	76	18	51	87	66	81	92	42	79	86	05	57

то сообщение «я знаком с шифрами замены» может быть зашифровано, например, любым из следующих трех способов:

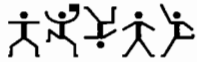
16 55 54 10 69 09 61 89 29 90 49 44 10 08 02 73 21 32 83 54 74
41 55 77 10 23 68 08 20 66 90 76 44 21 61 90 55 21 61 83 54 42
57 30 27 10 91 68 32 20 80 02 49 45 40 32 46 55 40 08 83 27 42

Так как множества $M_а$, $M_б$, $M_в$, ..., $M_я$ попарно не пересекаются, то по каждому символу шифрованного сообщения можно однозначно определить, какому множеству он принадлежит, и, следовательно, какую букву открытого сообщения он заменяет. Поэтому расшифрование возможно и открытое сообщение определяется единственным образом.

Часто M_α состоит из одного элемента. Например, в романе Ж. Верна «Путешествие к центру Земли» в руки профессора Лиденброка попадает пергамент с рукописью из знаков рунического письма. Каждое множество M_α состоит из одного элемента. Элемент каждого множества выбирается из набора символов вида

(2)

В рассказе А. Конан Дойла «Пляшущие человечки» каждый символ изображает пляшущего человечка в самых различных позах



(3)

На первый взгляд кажется, что чем хитрее символы, тем труднее вскрыть сообщение, не имея ключа. Это, конечно, не так. Если каждому символу однозначно сопоставить какую-либо букву или число, то легко перейти к зашифрованному сообщению из букв или чисел. В романе Ж. Верна «Путешествие к центру Земли» каждый рунический знак был заменен на соответствующую букву немецкого языка, что облегчило восстановление открытого сообщения. С точки зрения криптографов использование различных сложных символов не усложняет шифра. Однако, если зашифрованное сообщение состоит из букв или цифр, то вскрывать такое сообщение удобнее.

Рассмотрим некоторые примеры шифров замены. Пусть каждое множество M_α состоит из одной буквы. Например,

а	б	в	г	д	е	ж	з	и	к	л	м	н	о	п	р
г	л	ь	п	д	р	а	м	ц	в	э	ъ	х	о	б	н

с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
с	ж	я	и	ю	к	щ	ф	е	у	ы	ч	ш	т	а

(4)

Такой шифр называется шифром простой однобуквенной замены. По ключу (4) удобно проводить зашифрование и расшифрование: при зашифровании каждая буква открытого текста заменяется на соответствующую букву из второй строки (а на г и т. д.) При расшифровании, наоборот, г заменяется на а и т. д. При шифровании и расшифровании надо помнить вторую строчку в (4), то есть ключ.

Запомнить произвольный порядок букв алфавита достаточно сложно. Поэтому всегда пытались придумать какое-либо правило, по которому можно просто восстановить вторую строчку в (4).

Одним из первых шифров, известных из истории, был так называемый шифр Цезаря, для которого вторая строка в (4) является последовательностью, записанной в алфавитном порядке, но начинающейся не с буквы а:

а	б	в	...	ь	э	ю	я
г	д	е	...	я	а	б	в

(5)

В одной из задач (задача 4.4) используется шифр Цезаря. Запомнить ключ в этом случае просто — надо знать первую букву второй строки (4) (последовательность букв в алфавите предполагается известной). Однако такой шифр обладает большим недостатком. Число различных ключей равно числу букв в алфавите. Перебрав эти варианты, можно

однозначно восстановить открытое сообщение, так как при правильном выборе ключа получится «осмысленный» текст. В других случаях обычно получается «нечитаемый» текст. Задача 4.4 именно на это и рассчитана. Несмотря на то, что используется фраза на латинском языке, которого школьники не знают, многие участники олимпиады смогли указать открытое сообщение.

Другим примером шифра замены может служить лозунговый шифр. Здесь запоминание ключевой последовательности основано на лозунге — легко запоминаемом слове. Например, выберем слово-лозунг «учебник» и заполним вторую строку таблицы по следующему правилу: сначала выписываем слово-лозунг, а затем выписываем в алфавитном порядке буквы алфавита, не вошедшие в слово-лозунг. Вторая строка в (4) примет вид

у ч е б н и к а в г д ж з л м о

п р с т ф х ц ш щ ъ ы ь э ю я

В данном случае число вариантов ключа существенно больше числа букв алфавита.

Рассмотренные шифры имеют одну слабость. Если в открытом сообщении часто встречается какая-либо буква, то в шифрованном сообщении часто будет встречаться соответствующий ей символ или буква. Поэтому при вскрытии шифра замены обычно стараются наиболее часто встречающимся символам шифрованного сообщения поставить в соответствие буквы открытого сообщения с наибольшей предполагаемой частотой появления. Если шифрованное сообщение достаточно большое, то этот путь приводит к успеху, даже если вы не знаете ключа.

Кроме частоты появления букв, могут быть использованы другие обстоятельства, помогающие раскрыть сообщение. Например, может быть известна разбивка на слова, как в задаче 4.2, и расставлены знаки препинания. Рассматривая небольшое число возможных вариантов замены для предлогов и союзов, можно попытаться определить часть ключа. В этой задаче существенно используется, какие гласные или согласные могут быть удвоенными: «нн», «ее», «ии» и др.

При анализе шифрованного сообщения следует исходить из того, что число различных вариантов для части определяемого ключа не такое уж большое, если вы находитесь на правильном пути. В противном случае либо вы получите противоречие, либо число вариантов ключа будет сильно возрастать. Обычно, начиная с некоторого момента определение открытого сообщения становится делом техники. Так, в задаче 4.2, если вы определили «денно и нощно», то дальнейшее определение открытого текста не представляет труда.

Вообще-то можно сказать, что вскрытие шифров замены является искусством и достаточно трудно формализовать этот процесс.

Популярные у школьников криптограммы (типа рассмотренной в задаче 1.5) по сути дела являются шифром замены с ключом

0	1	2	3	4	5	6	7	8	9
ш	и	ф	р	з	а	м	е	н	ы

в котором каждой цифре ставится в соответствие буква. При этом должны соблюдаться правила арифметики. Эти правила значительно облегчают определение открытого текста, так же, как правила синтаксиса и орфографии в задаче 4.2 облегчают нахождение четверостишия В. Высоцкого.

Любые особенности текста, которые могут быть вам известны, — ваши помощники. Например, в задаче 5.2 прямо сказано, что в тексте есть выражения «зпт», «тчк», как часто бывает в реальных телеграммах. И эта подсказка — путь к решению задачи.

Шифрование даже относительно небольших текстов на одном ключе для рассмотренных шифров замены создает условия для вскрытия открытых сообщений. Поэтому такие шифры пытались усовершенствовать. Одно из направлений — построение шифров разнозначной замены, когда каждой букве ставится в соответствие один или два символа. (Простейшим примером является шифр, определяемый в задаче 4.2.) Например,

а	б	в	г	д	е	ж	з	и	к	л	м	н	о	п	р
73	74	51	65	2	68	59	1	60	52	75	61	8	66	58	3

с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
69	64	53	54	9	62	71	4	67	56	72	63	55	70	57

Если шифрованное сообщение написано без пробелов между символами, то появляется дополнительная трудность при разбиении шифрованного сообщения на отдельные символы и слова.

Другое направление создания шифров замены состоит в том, чтобы множества шифробозначений M_α содержали более одного элемента. Такие шифры получили название шифров многозначной замены. Они позволяют скрыть истинную частоту букв открытого сообщения, что существенно затрудняет вскрытие этих шифров. Главная трудность, которая возникает при использовании таких шифров, заключается в запоминании ключа. Надо запомнить не одну строчку, а для каждой буквы алфавита α — множество ее шифробозначений M_α. Как правило, элементами множеств M_α являются числа. Из художественной литературы и кинофильмов про разведчиков вам известно, что во время второй мировой войны часто использовались так называемые книжные шифры. Множество шифробозначений для каждой буквы определяется всеми пятизначными наборами цифр, в каждом из которых первые две цифры указывают номер страницы, третья цифра — номер строки, четвертая и пятая цифры — номер места данной буквы в указанной

строке. Поэтому при поимке разведчика всегда пытались найти книгу, которая могла быть использована им в качестве ключа.

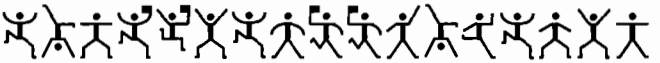
Мы не останавливаемся здесь на более сложных методах построения шифров замены. Приведенных примеров достаточно, чтобы оценить многообразие таких шифров. Но все они имеют серьезный недостаток — на одном ключе нельзя шифровать достаточно длинные сообщения. Поэтому, как правило, шифры замены используются в комбинации с другими шифрами. Чаще всего — с шифрами перестановки, о которых вы прочитаете в следующем разделе.

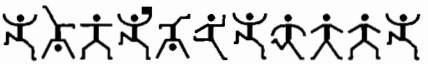
В заключение, следуя героям известных литературных произведений, вскроем некоторые шифры замены. Обратите внимание на то, какие неожиданные обстоятельства при этом используются. Действительно, вскрытие шифров — искусство.

А. Конан Дойл, «Пляшущие человечки»

В этом рассказе Холмсу необходимо было прочитать тексты пяти записок:

I. 

II. 

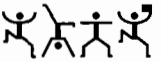
III. 

IV. 

V. 

Первая записка была так коротка, что дала возможность Холмсу сделать всего лишь одно правдоподобное предположение, оказавшееся впоследствии правильным. По-видимому, флаги употребляются лишь для того, чтобы отмечать концы отдельных слов. Больше ничего по первой записке установить было нельзя. Четвертая записка, по всей видимости, содержала всего одно слово, так как в ней не было флагов.

Вторая и третья записки начинались, несомненно, с одного и того же слова из четырех букв. Вот это слово:



Оно кончается той же буквой, какой и начинается. Счастливая мысль: письма обычно начинаются с имени того, кому письмо адресовано. Человек, писавший миссис Кьюбит эти послания, был, безусловно, близко

с ней знаком. Вполне естественно, что он называет ее просто по имени. А зовут ее Илси. Таким образом, Холмсу стали известны три буквы: И, Л и С.

В двух записках их автор обращается к миссис Кьюбит по имени и, видимо, чего-то требует от нее. Не хочет ли он, чтобы она пришла куда-нибудь, где он мог с ней поговорить? Холмс обратился ко второму слову третьей записки. В нем 7 букв, из которых третья и последняя — И. Холмс предположил, что слово это — ПРИХОДИ, и сразу оказался обладателем еще 5 букв: П, Р, Х, О, Д.

Тогда он обратился к четвертой записи, которая появилась на двери сарая. Холмс предположил, что она является ответом и что написала ее миссис Кьюбит. Подставив в текст уже известные буквы, он получил: –И–О–Д–. Что же могла миссис Кьюбит ответить на просьбу прийти? Внезапно Холмс догадался: НИКОГДА

Возвратившись к первой записке, Холмс получил:

– –Д–С– А– СЛ–НИ

Он предположил, что четвертое слово — СЛЕНИ Это — фамилия, чрезвычайно распространенная в Америке. Коротенькое слово из двух букв, стоящее перед фамилией, по всей вероятности, имя. Какое же имя может состоять из двух букв? В Америке весьма распространено имя Аб. Теперь остается установить только первое слово фразы; оно состоит всего из одной буквы, и отгадать его нетрудно: это — местоимение Я.

Далее Холмс восстанавливает содержание второй записки:

```
ИЛСИ  Я  –И––  –  –ЛРИД–А
          *  o  o        *
```

Здесь указаны границы слов, а снизу одинаковыми символами отмечены одинаковые буквы. Четвертое слово состоит из одной буквы (по-видимому, это союз или предлог). Буквы О и И уже определены, С, А и К — тоже. Остаются следующие возможности: это — либо В, либо У. Вряд ли это — В, так как в этом случае получилось бы «нечитаемое» третье слово –И–В. Поэтому, скорее всего — это предлог У. Небольшой перебор незадействованных букв дает правдоподобную гипотезу о значении третьего слова: ЖИВУ. Скорее всего, последнее слово (–ЛРИДЖА) — мужское имя, в котором неизвестная буква — Э. Поэтому вторая записка гласит: ИЛСИ Я ЖИВУ У ЭЛРИДЖА

Холмс послал телеграмму в нью-йоркское полицейское управление с запросом о том, кто такой Аб Слени. Поступил ответ: «Самый опасный бандит в Чикаго».

Сразу после этого появилась последняя (5-я) записка, в которой не хватало трех букв: ИЛСИ ГО–ОВЬСЯ К С–ЕР–И, из которой сразу определяются буквы М и Т:

ИЛСИ ГОТОВЬСЯ К СМЕРТИ

Шестая записка была направлена Холмсом преступнику:

Э. По, «Золотой жук»

Найден пергамент с текстом криптограммы. Для удобства пронумеруем по порядку все символы этого текста:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
5	3	#	#	+	3	0	5	)	)	6	*	;	4	8	2	6	)	4
20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36		
#	•	)	4	#	)	;	8	0	6	*	;	4	8	+	8	□		
37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52			
6	0	)	)	8	5	;	;	]	8	*	;	:	#	*	8			
53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69		
+	8	3	(	8	8	)	5	*	+	;	4	6	(	;	8	8		
70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86		
*	9	6	*	?	;	8	)	*	#	(	;	4	8	5	)	;		
87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102			
5	*	+	2	:	*	#	(	;	4	9	5	6	*	2	(			
103	104	105	106	107	108	109	110	111	112	113	114	115						
5	*	=	4	)	8	□	8	*	;	4	0	6						
116	117	118	119	120	121	122	123	124	125	126	127	128						
9	2	8	5	)	;	)	6	+	8	)	4	#						
129	130	131	132	133	134	135	136	137	138	139	140	141	142					
#	;	1	(	#	9	;	4	8	0	8	1	;	8					
143	144	145	146	147	148	149	150	151	152	153	154	155	156					
:	8	#	1	;	4	8	+	8	5	;	4	)	4					
157	158	159	160	161	162	163	164	165	166	167	168	169						
8	5	+	5	2	8	8	0	6	*	8	1	(						
170	171	172	173	174	175	176	177	178	179	180	181	182						
#	9	;	4	8	;	(	8	8	;	4	(	#						
183	184	185	186	187	188	189	190	191	192	193	194	195						
?	3	4	;	4	8	)	4	#	;	1	6	1						
196	197	198	199	200	201	202	203	204										
;	:	1	8	8	;	#	?	;										

Кроме того, на пергаменте изображены череп и козленок. Главный герой рассказа рассуждал следующим образом. По английски козленок — kid; череп связан с капитаном Киддом, по английски — kidd. Козленок

был нарисован на пергаменте в том месте, где ставится подпись. Изображение черепа в противоположном по диагонали углу наводило на мысль о печати или гербе. Капитан Кидд владел несметным богатством. Кидд, насколько мы можем судить о нем, не сумел бы составить истинно сложную криптограмму. По-видимому, это была простая замена. Возникает только вопрос о языке, на котором был написан текст. В данном случае трудностей с определением языка не было: подпись давала разгадку. Игра слов kid и kidd возможна лишь в английском языке.

Текст криптограммы идет в сплошную строку. Задача была бы намного проще, если бы отдельные слова были отделены пробелами. Тогда можно было бы начать с анализа и сличения более коротких слов, и как только нашлось бы слово из одной буквы (например, местоимение «я» или союз «и» — для русского языка), начало было бы положено. Но просветов в строке не было.

Приходится подсчитывать частоты одинаковых символов, чтобы узнать, какие из них чаще, а какие реже встречаются в криптограмме. В результате получилась таблица частот всех символов:

8	*;*	*4*	*)*	*#*	***	*5*	*6*	*(*	*+*	*1*	*0*	*2*	*9*	*:*	*3*	*?*	□	•	*]*	=
34	27	19	16	15	14	12	11	9	8	7	6	5	5	4	4	3	2	1	1	1

В английской письменной речи самая частая буква — e. Далее идут в нисходящем порядке: a, o, i, d, h, n, r, s, t, u, y, c, f, g, l, m, w, b, k, p, q, x, z. Буква e, однако, настолько часто встречается, что трудно построить фразу, в которой она не занимала бы господствующего положения. Итак, уже сразу у нас в руках путеводная нить. Составленная таблица, вообще говоря, может быть очень полезна, но в данном случае она понадобилась лишь в начале работы.

Поскольку символ *8* встречается чаще других, примем его за букву e английского алфавита. Для проверки этой гипотезы взглянем, встречается ли этот символ дважды подряд, так как в английском языке буква e часто удваивается, например, в словах meet, fleet, speed, seen, seed, been, agree, и т. д. Хотя криптограмма невелика, пара *88* стоит в нем пять раз.

Самое частое слово в английском языке — определенный артикль the. Посмотрим, не повторяется ли у нас сочетание из трех символов, расположенных в одинаковой последовательности и оканчивающихся символом *8*. Если такое найдется, то это будет, по всей вероятности, the. Приглядевшись, находим семь раз сочетание из трех символов *;48*. Итак, мы имеем право предположить, что символ *;* — это буква t, а *4* — h; вместе с тем подтверждается, что *8* — это действительно e. Мы сделали важный шаг вперед.

То, что мы расшифровали целое слово, потому так существенно, что позволяет найти границы некоторых других слов. Для примера возьмем

предпоследнее из сочетаний этого рода *;48* (позиции 172–174). Идущий сразу за *8* символ *;* будет, как видно, начальной буквой нового слова. Выпишем, начиная с него, 6 символов подряд. Только один из них нам незнаком. Обозначим известные символы буквами и оставим свободное место для неизвестного символа (обозначим его точкой) t.eeth, ни одно слово, начинающееся с t и состоящее из 6 букв, не имеет в английском языке окончания th. В этом легко убедиться, подставляя на свободное место все буквы по очереди. Попробуем отбросить две последние буквы и получим t.ee, для заполнения свободного места можно снова взяться за алфавит. Единственно верным прочтением этого слова будет tree (дерево). В таком случае мы узнаем еще одну букву — r, она обозначена символом *(* и мы можем прочитать два слова подряд the tree, в дальнейшем эта гипотеза может либо подтвердиться, либо привести к некоторому «нечитаемому» фрагменту. В последнем случае следует попытаться восстановить либо слово t.e, либо t.eet, либо слово, целиком включающее в себя t.eeth

Развиваем успех. Немного далее (186–188) находим уже знакомое нам сочетание *;48*. Примем его опять за границу нового слова и выпишем целый отрывок, начиная с двух расшифрованных нами слов. Получаем такую запись:

the tree *;4(#?34* the

Заменим уже известные символы буквами:

the tree thr*#?3*h the

а неизвестные — точками:

the tree thr...h the

Нет никакого сомнения, что неясное слово — through (через). Это открытие дает нам еще три буквы — o, u и g, обозначенные в криптограмме символами *#* *?* и *3*.

Надписывая над уже определенными символами криптограммы их значения, находим вблизи от ее начала (позиции 54–58) группу символов *83(88*, которая читается так egree, это, конечно, слово degree (градус) без первой буквы. Теперь мы знаем, что буква d обозначена символом *+*. Вслед за словом degree через 4 символа встречаем группу *;46(;88**. Заменим известные символы буквами, а неизвестные — точками th.rtee, по-видимому, перед нами слово thirteen (тринадцать). К известным нам буквам прибавились i и n, обозначенные в криптограмме символами *6* и ***.

Криптограмма начинается так: *53##+*. Подставляя буквы и точки, получаем .good, недостающая буква, конечно, a, и, значит, два первых слова будут читаться так: a good (хороший). Определены следующие 11 символов:

5	*+*	*8*	*3*	*4*	*6*	***	*#*	*(*	*;*	*?*
a	d	e	g	h	i	n	o	r	t	u

На этом анализ Э. По заканчивается. Дальнейшую работу проделаем самостоятельно.

Четвертый по частоте (16 вхождений) символ *)* еще не определен. Возвратимся к диаграмме встречаемости букв английского языка. Среди первого десятка букв этой диаграммы у нас не встретилась лишь буква s. Она — первый претендент на значение символа *)*. Эта гипотеза подтверждается тем, что вряд ли *)* обозначает гласную букву, так как в таком случае мы получили бы «нечитаемые» фрагменты

6	7	8	9	10	11	12
g	.	a	*)*	*)*	i	n

или

37	38	39	40	41	42
i	.	*)*	*)*	e	a

То, что символ *)* — это буква s, легко проверяется на участке криптограммы с 60-й по 89-ю позиции .and thirteen .inutes north east and Поэтому полагаем, что символ *)* — это s. Попутно определилось значение символа ***9***, это — m.

Перебирая возможные значения символа ***0***, стоящего на позициях 7 и 28 криптограммы, убеждаемся в том, что единственно возможным его значением может быть лишь буква l (glass — стекло, hostel — общежитие, гостиница или трактир).

Определяем, далее, значение символа □ как v по фрагменту текста в позициях 107–113.

Теперь на участке текста с 22-й по 70-ю позиции остались неопределенными лишь значения символов *]* и *:*, встретившихся по одному разу. Очевидно, что символ *]* — это w, а символ *:* — это y. Теперь на участке текста с 172-й по 204-ю позиции не выявлено лишь значение символа ***1***, которое, как нетрудно заметить, может быть лишь буквой f.

Символ ***2***, стоящий на позициях 117 и 90, очевидно, заменяет букву b.

Осталось определить лишь значения символов • и =. Небольшой перебор еще неустановленных букв показывает, что символ = — это c, а символ • может обозначать одну из букв k, p, q, x или z. Обратившись к словарю, находим единственное подходящее окончание p слова bishop (епископ, слон).

Таким образом, однозначно определились значения всех 21 символов, встречающихся в криптограмме. Получился следующий открытый текст:

«A good glass in the bishop's hostel in the devil's seat twenty one degrees and thirteen minutes northeast and by north main branch seventh limb east side shoot from the left eye of the death's head a bee line from the tree through the shot fifty feet out».

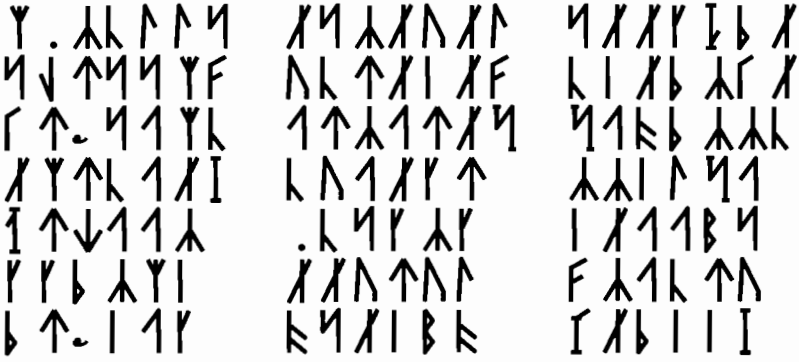
В переводе на русский язык: «Хорошее стекло в трактире епископа на чертовом стуле двадцать один градус и тринадцать минут северо-северо-восток главный сук седьмая ветвь восточная сторона стреляй из левого глаза мертвой головы прямая от дерева через выстрел на пятьдесят футов».

Восстановленная простая замена:

A	B	C	D	E	F	G	H	I	L	M	N	O	P	R	S	T	U	V	W	Y
5	*2*	=	+	*8*	*1*	*3*	*4*	*6*	*0*	*9*	*	#	•	(	)	;	?	□	]	:

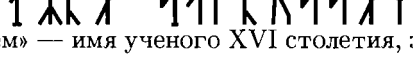
Ж. Верн, «Путешествие к центру Земли»

В руки профессора Лиденброка попадает пергамент со следующей рукописью:



«Это — рунические письмена; знаки эти совершенно похожи на знаки манускрипта Снорре. Но ... что же они означают? — спрашивает профессор, — ... Ведь это все же древнеисландский язык, — бормотал он себе под нос». Изучение рукописи привело профессора к выводу о том, что это зашифрованное сообщение. Для его прочтения профессор решил заменить буквы сообщения их аналогами в современном немецком алфавите: «А теперь я буду диктовать тебе, — говорит он своему помощнику, — буквы нашего алфавита, соответствующие каждому из этих исландских знаков». Он называл одну букву за другой, и таким образом последовательно составлялась таблица непостижимых слов:

m . r n l l s	e s r e u e l	s e e c J d e
s g t s s m f	u n t e i e f	n i e d r k e
k t , s a m n	a t r a t e S	S a o d r r n
e m t n a e l	n u a e c t	r r i l S a
A t v a a r	. n s c r c	i e a a b s
c c d r m i	e e u t u l	f r a n t u
d t , i a c	o s e i b o	K e d i i l

Можно было предположить, что таинственная запись сделана одним из обладателей книги, в которой находился пергамент. Не оставил ли он своего имени на какой-нибудь странице? На обороте второй страницы профессор обнаружил что-то вроде пятна, похожего на чернильную кляксу. Воспользовавшись лупой, он различил несколько наполовину стертых знаков, которые можно было восстановить. Получилась запись  которая читалась как «арне сакнуссем» — имя ученого XVI столетия, знаменитого алхимика!

Далее профессор рассуждал так: «Документ содержит 132 буквы, 79 согласных и 53 гласных. Приблизительно такое же соотношение существует в южных языках, в то время как наречия севера бесконечно богаче согласными. Следовательно, мы имеем дело с одним из южных языков.» « ... Сакнуссем, — продолжал профессор, — был ученый человек; поэтому раз он писал не на родном языке, то, разумеется, должен был отдавать предпочтение языку, общепринятому среди образованных умов XVI века, а именно — латинскому. Если я ошибаюсь, то можно будет испробовать испанский, французский, итальянский, греческий или еврейский. Но ученые XVI столетия писали обычно по-латински. Таким образом, я вправе признать не подлежащим сомнению, что это — латынь.»

«Всмотримся хорошенько, — сказал он, снова взяв исписанный листок. — Вот ряд из 132 букв, расположенных крайне беспорядочно. Вот слова, в которых встречаются только согласные, как, например, первое m.rnlls; в других, напротив, преобладают гласные, например, в пятом unteief, или в предпоследнем — oseibo. Очевидно, что эта группировка не случайна; она произведена автоматически, при помощи неизвестного нам соотношения, которое определило последовательность этих букв. Я считаю несомненным, что первоначальная фраза была написана правильно, но затем по какому-то принципу, который надо найти, подверглась преобразованию. Тот, кто владел бы ключом этого шифра, свободно прочел бы ее. Но что это за ключ?»

«При желании затемнить смысл фразы первое, что приходит на ум, как мне кажется, это написать слова в вертикальном направлении, а не в горизонтальном». Проверяя эту гипотезу, он начал диктовать, называя сначала первые буквы каждого слова, потом вторые; он диктовал буквы в таком порядке:

m e s s u n k a S e n r A . i c e f d o K . s e g n i t t a m u r t n e c e
r t s e r r e t t e , r o t a i v s a d u a , e d n e c s e d s a d n e l a k
a r t n i i i l u J s i r a t r a c S a r b m u t a b i l e d m e k m e r e t
a r c s i l u c o l s l e f f e n S n l

С полученным текстом у профессора долго ничего не выходило. Это почти привело его в отчаяние. Однако « ... совершенно машинально я

стал обмахиваться этим листком бумаги, так что лицевая и оборотная стороны листка попеременно представали перед моими глазами. ... Каково же было мое изумление, когда вдруг мне показалось, что передо мной промелькнули знакомые, совершенно ясные слова, латинские слова: craterem, terrestre!» Дело в том, что читать этот текст нужно было не слева направо, как обычно, а наоборот! Таким образом, случай помог профессору найти ключ к решению задачи. Документ гласил следующее:

«In Sneffels Ioculis craterem kem delibat umbra Scartaris Julii intra calendas descende, audas viator, et terrestre centrum attinges. Kod feci. Arne Saknussem».

В переводе это означало: «Спустись в кратер Екуль Снайфедльс, который тень Скартариса ласкает перед июльскими календами, отважный странник, и ты достигнешь центра Земли. Это я совершил. Арне Сакнуссем».

3. Шифры перестановки

Шифр, преобразования из которого изменяют только порядок следования символов исходного текста, но не изменяют их самих, называется шифром перестановки (ШП).

Рассмотрим преобразование из ШП, предназначенное для зашифрования сообщения длиной n символов. Его можно представить с помощью таблицы

$$\begin{pmatrix} 1 & 2 & \dots & n \\ i_1 & i_2 & \dots & i_n \end{pmatrix}, \tag{6}$$

где i_1 — номер места шифртекста, на которое попадает первая буква исходного сообщения при выбранном преобразовании, i_2 — номер места для второй буквы и т. д. В верхней строке таблицы выписаны по порядку числа от 1 до n, а в нижней — те же числа, но в произвольном порядке. Такая таблица называется подстановкой степени n.

Зная подстановку, задающую преобразование, можно осуществить как зашифрование, так и расшифрование текста. Например, если для преобразования используется подстановка

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 5 & 2 & 3 & 1 & 4 & 6 \end{pmatrix}$$

и в соответствии с ней зашифровывается слово МОСКВА, то получится КОСВМА. Попробуйте расшифровать сообщение НЧЕИУК, полученное в результате преобразования с помощью указанной выше подстановки.

В качестве упражнения читателю предлагается самостоятельно выписать подстановки, задающие преобразования в описанных ниже трех

примерах шифров перестановки. Ответы помещены в конце раздела.

Читатель, знакомый с методом математической индукции, может легко убедиться в том, что существует $1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$ (обозначается $n!$, читается «n факториал») вариантов заполнения нижней строки таблицы (6). Таким образом, число различных преобразований шифра перестановки, предназначенного для зашифрования сообщений длины n, меньше либо равно $n!$ (заметим, что в это число входит и вариант преобразования, оставляющий все символы на своих местах!).

С увеличением числа n значение $n!$ растет очень быстро. Приведем таблицу значений $n!$ для первых 10 натуральных чисел:

n	1	2	3	4	5	6	7	8	9	10
$n!$	1	2	6	24	120	720	5040	40320	362880	3628800

При больших n для приближенного вычисления $n!$ можно пользоваться известной формулой Стирлинга

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n,$$

где $e = 2,718281828\ldots$.

Примером ШП, предназначенного для зашифрования сообщений длины n, является шифр, в котором в качестве множества ключей взято множество всех подстановок степени n, а соответствующие им преобразования шифра задаются, как было описано выше. Число ключей такого шифра равно $n!$.

Для использования на практике такой шифр не удобен, так как при больших значениях n приходится работать с длинными таблицами.

Широкое распространение получили шифры перестановки, использующие некоторую геометрическую фигуру. Преобразования из этого шифра состоят в том, что в фигуру исходный текст вписывается по ходу одного «маршрута», а затем по ходу другого выписывается с нее. Такой шифр называют маршрутной перестановкой. Например, можно вписывать исходное сообщение в прямоугольную таблицу, выбрав такой маршрут: по горизонтали, начиная с левого верхнего угла поочередно слева направо и справа налево. Выписывать же сообщение будем по другому маршруту: по вертикали, начиная с верхнего правого угла и двигаясь поочередно сверху вниз и снизу вверх.

Зашифруем, например, указанным способом фразу:

ПРИМЕРМАРШРУТНОЙПЕРЕСТАНОВКИ

используя прямоугольник размера 4×7:

П	Р	И	М	Е	Р	М
Н	Т	У	Р	Ш	Р	А
О	Й	П	Е	Р	Е	С
И	К	В	О	Н	А	Т

Зашифрованная фраза выглядит так:

МАСТАЕРРЕШРНОЕРМИУПВКЙТРПНОИ

Теоретически маршруты могут быть значительно более изощренными, однако запутанность маршрутов усложняет использование таких шифров.

Ниже приводятся описания трех разновидностей шифров перестановки, встречавшихся в задачах олимпиад.

Шифр «Сцитала». Одним из самых первых шифровальных приспособлений был жезл («Сцитала»), применявшийся еще во времена войны Спарты против Афин в V веке до н. э. Это был цилиндр, на который виток к витку наматывалась узкая папирусная лента (без просветов и нахлестов), а затем на этой ленте вдоль его оси записывался необходимый для передачи текст. Лента сматывалась с цилиндра и отправлялась адресату, который, имея цилиндр точно такого же диаметра, наматывал ленту на него и прочитывал сообщение. Ясно, что такой способ шифрования осуществляет перестановку местами букв сообщения.

Шифр «Сцитала», как видно из решения задачи 2.1, реализует не более n перестановок (n, по прежнему, — длина сообщения). Действительно, этот шифр, как нетрудно видеть, эквивалентен следующему шифру маршрутной перестановки: в таблицу, состоящую из m столбцов, построчно записывают сообщение, после чего выписывают буквы по столбцам. Число задействованных столбцов таблицы не может превосходить длины сообщения.

Имеются еще и чисто физические ограничения, накладываемые реализацией шифра «Сцитала». Естественно предположить, что диаметр жезла не должен превосходить 10 сантиметров. При высоте строки в 1 сантиметр на одном витке такого жезла уместится не более 32 букв ($10\pi < 32$). Таким образом, число перестановок, реализуемых «Сциталой», вряд ли превосходит 32.

Шифр «Поворотная решетка». Для использования шифра, называемого поворотной решеткой, изготавливается трафарет из прямоугольного листа клетчатой бумаги размера $2m \times 2k$ клеток. В трафарете вырезано mk клеток так, что при наложении его на чистый лист бумаги того же размера четырьмя возможными способами его вырезы полностью покрывают всю площадь листа.

Буквы сообщения последовательно вписываются в вырезы трафарета (по строкам, в каждой строке слева направо) при каждом из четырех его возможных положений в заранее установленном порядке.

Поясним процесс шифрования на примере. Пусть в качестве ключа используется решетка 6×10, приведенная на рис. 1.

Зашифруем с ее помощью текст

ШИФРРЕШЕТКАЯВЛЯЕТСЯЧАСТНЫМСЛУЧАЕМШИФРАМАРШРУТНОЙПЕРЕСТАНОВКИ

Наложив решетку на лист бумаги, вписываем первые 15 (по числу вы-

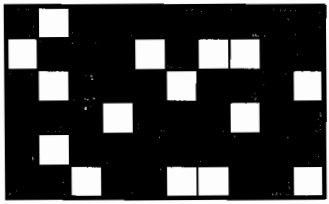


Рис. 1.

резов) букв сообщения: ШИФРРЕШЕТКАЯВЛЯ . . . . Сняв решетку, мы увидим текст, представленный на рис. 2. Поворачиваем решетку на 180°. В окошечках появятся новые, еще не заполненные клетки. Вписываем в них следующие 15 букв. Получится запись, приведенная на рис. 3. Затем переворачиваем решетку на другую сторону и зашифровываем остаток текста аналогичным образом (рис. 4, 5).

	Ш								
И				Ф		Р	Р		
	Е				Ш				Е
			Т				К		
	А								
		Я			В	Л			Я

Рис. 2.

Е	Ш		Т	С			Я		
И				Ф		Р	Р	Ч	
	Е	А			Ш	С			Е
Т			Т	Н			К	Ы	
	А	М	С		Л				У
		Я			В	Л		Ч	Я

Рис. 3.

Е	Ш	А	Т	С	Е	М	Я		Ш
И	И			Ф		Р	Р	Ч	
	Е	А	Ф		Ш	С	Р		Е
Т	А		Т	Н	М		К	Ы	А
Р	А	М	С	Ш	Л	Р	У		У
	Т	Я			В	Л		Ч	Я

Рис. 4.

Е	Ш	А	Т	С	Е	М	Я	Н	Ш
И	И	О	Й	Ф	П	Р	Р	Ч	Е
Р	Е	А	Ф	Е	Ш	С	Р	С	Е
Т	А	Т	Т	Н	М	А	К	Ы	А
Р	А	М	С	Ш	Л	Р	У	Н	У
О	Т	Я	В	К	В	Л	И	Ч	Я

Рис. 5.

Получатель сообщения, имеющий точно такую же решетку, без труда прочтет исходный текст, наложив решетку на шифртекст по порядку четырьмя способами.

Можно доказать, что число возможных трафаретов, то есть количество ключей шифра «решетка», составляет $T = 4^{mk}$ (см. задачу 1.1). Этот шифр предназначен для сообщений длины $n = 4mk$. Число всех перестановок в тексте такой длины составит $(4mk)!$, что во много раз

больше числа T. Однако, уже при размере трафарета 8×8 число возможных решеток превосходит 4 миллиарда.

Широко распространена разновидность шифра маршрутной перестановки, называемая **«шифром вертикальной перестановки» (ШВП)**. В нем снова используется прямоугольник, в который сообщение вписывается обычным способом (по строкам слева направо). Выписываются буквы по вертикали, а столбцы при этом берутся в порядке, определяемом ключом. Пусть, например, этот ключ таков: (5,4,1,7,2,6,3), и с его помощью надо зашифровать сообщение:

ВОТПРИМЕРШИФРАВЕРТИКАЛЬНОЙПЕРЕСТАНОВКИ

Впишем сообщение в прямоугольник, столбцы которого пронумерованы в соответствии с ключом:

5	1	4	7	2	6	3
В	О	Т	П	Р	И	М
Е	Р	Ш	И	Ф	Р	А
В	Е	Р	Т	И	К	А
Л	Ь	Н	О	Й	П	Е
Р	Е	С	Т	А	Н	О
В	К	И	–	–	–	–

Теперь, выбирая столбцы в порядке, заданном ключом, и выписывая последовательно буквы каждого из них сверху вниз, получаем такую криптограмму:

ОРЕЬЕКРФИЙА–МААЕО–ТШРНСИВЕВЛРВИРКПН–ПИТОТ–

Число ключей ШВП не более $m!$, где m — число столбцов таблицы. Как правило, m гораздо меньше, чем длина текста n (сообщение укладывается в несколько строк по m букв), а, значит, и $m!$ много меньше $n!$.

Пользуясь приведенной выше формулой Стирлинга при больших m и n, попытайтесь оценить, во сколько раз число возможных перестановок ШВП с m столбцами меньше числа всех перестановок на тексте длины n, кратном m.

В случае, когда ключ ШВП не рекомендуется записывать, его можно извлекать из какого-то легко запоминающегося слова или предложения. Для этого существует много способов. Наиболее распространенный состоит в том, чтобы приписывать буквам числа в соответствии с обычным алфавитным порядком букв. Например, пусть ключевым словом будет ПЕРЕСТАНОВКА. Присутствующая в нем буква А получает номер 1. Если какая-то буква входит несколько раз, то ее появления нумеруются последовательно слева направо. Поэтому второе вхождение буквы А получает номер 2. Поскольку буквы Б в этом слове нет, то буква В получает номер 3 и так далее. Процесс продолжается до тех

пор, пока все буквы не получат номера. Таким образом, мы получаем следующий ключ:

П	Е	Р	Е	С	Т	А	Н	О	В	К	А
9	4	10	5	11	12	1	7	8	3	6	2

Перейдем к вопросу о методах вскрытия шифров перестановки. Проблема, возникающая при восстановлении сообщения, зашифрованного ШП, состоит не только в том, что число возможных ключей велико даже при небольших длинах текста. Если и удастся перебрать все допустимые варианты перестановок, не всегда ясно, какой из этих вариантов истинный. Например, пусть требуется восстановить исходный текст по криптограмме АОГР, и нам ничего не известно, кроме того, что применялся шифр перестановки. Какой вариант «осмысленного» исходного текста признать истинным: ГОРА или РОГА? А может быть АРГО? Приведем пример еще более запутанной ситуации. Пусть требуется восстановить сообщение по криптограмме

АА НИНК–ТЕОМЛ,З.ЬЬЗИВТЛП–ЬЯО

полученной шифром перестановки. Возможны, как минимум, два варианта исходного сообщения:

КАЗНИТЬ,-НЕЛЬЗЯ-ПОМИЛОВАТЬ. и
КАЗНИТЬ-НЕЛЬЗЯ,-ПОМИЛОВАТЬ.

Эти варианты имеют прямо противоположный смысл и в имеющихся условиях у нас нет возможности определить, какой из вариантов истинный.

Иногда, за счет особенностей реализации шифра, удается получить информацию об использованном преобразовании (перестановке). Рассмотрим шифр «Сцитала» из задачи 2.1. Выше уже рассматривался вопрос о количестве перестановок, реализуемых «Сциталой». Их оказалось не более 32. Это число невелико, поэтому можно осуществить перебор всех вариантов. При достаточной длине сообщения, мы, скорее всего, получим единственный читаемый вариант текста. Однако, используя информацию о расположении линий, оставленных шифровальщиком, удается определить диаметр стержня, а значит, и возникающую перестановку букв (см. задачу 2.1).

В рассмотренном примере шифровальщик по неосторожности оставил на папирусе следы, позволяющие нам легко прочитать сообщение. Возможны и другие ситуации, когда не очень «грамотное» использование шифра облегчает вскрытие переписки.

В задаче 5.2 содержится пример текста, зашифрованного ШВП. По условию пробелы между словами при записи текста в таблицу опускались. Поэтому заключаем, что все столбцы, содержащие пробел в последней строке, должны стоять в конце текста. Таким образом, возникает разбиение столбцов на две группы (содержащие 6 букв, и

содержащие 5 букв). Для завершения восстановления исходного текста достаточно найти порядок следования столбцов в каждой из групп в отдельности, что гораздо проще.

Аналогичная ситуация возникает и при «неполном» использовании шифра «решетка» (см. задачу 4.1). Пусть имеется решетка размера $m \times r$, и зашифрованное с ее помощью сообщение длины $mr - k$, не содержащее пробелов. Незаполненные k мест в решетке при условии, что $k \leqslant mr/4$, соответствуют вырезам в четвертом положении решетки. На основе такой информации, происходит резкое уменьшение числа допустимых решеток (их будет $4^{mr/4-k}$). Читателю предлагается самостоятельно подсчитать число допустимых решеток при $k > mr/4$.

На примере решения задачи 5.2 продемонстрируем еще один подход к вскрытию шифров вертикальной перестановки — лингвистический. Он основан на том, что в естественных языках некоторые комбинации букв встречаются очень часто, другие — гораздо реже, а многие вообще не встречаются (например — «ыьъ»).

Будем подбирать порядок следования столбцов друг за другом так, чтобы во всех строках этих столбцов получались «читаемые» отрезки текста. В приведенном решении задачи восстановление текста начинается с подбора цепочки из трех столбцов первой группы, содержащей в последней строке сочетание ТЧК, так как естественно предположить, что сообщение заканчивается точкой. Далее подбираются столбцы, продолжающие участки текста в других строках, и т. д.

Сочетание лингвистического метода с учетом дополнительной информации довольно быстро может привести к вскрытию сообщения.

В заключение рассказа о шифрах перестановки приведем историю с зашифрованным автографом А. С. Пушкина, описанную в романе В. Каверина «Исполнение желаний».

Главный герой романа — студент-историк Н. Трубачевский, — занимавшийся работой в архиве своего учителя — академика Бауэра С. И., — нашел в одном из секретных ящиков пушкинского бюро фрагмент недописанной X главы «Евгения Онегина». Это был перегнутый вдвое полулист плотной голубоватой бумаги с водяным знаком 1829 года. На листе было написано следующее.

1. Властитель слабый и лукавый
2. Его мы очень смирным знали
3. Гроза двенадцатого года
4. Но Бог помог — стал ропот ниже
5. И чем жирнее, тем тяжеле
6. Авось, о Шиболет народный
7. Авось, аренды забывая
8. Сей муж судьбы, сей странник бранный

1. Нечаянно пригретый славой
2. Орла двуглавого щипали
3. Остервенение народа
4. Мы очутилися в Париже
5. Скажи, зачем ты в самом деле
6. Но стихоплет великородный
7. Авось по манью Николая
8. Сей всадник, папою венчанный

9. Тряслися грозно Пиринеи	*9. Безрукий князь друзьям Мореи*
10. Я всех уйму с моим народом	*10. А про себя и в ус не дует*
11. Потешный полк Петра Титана	*11. Предавших некогда тирана*
12. Россия присмирела снова	*12. Но искра пламени иного*
13. У них свои бывали сходки	*13. Они за рюмкой русской водки*
14. Витийством резким знамениты	*14. У беспокойного Никиты*
15. Друг Марса, Вакха и Венеры	*15. Свои решительные меры*
16. Так было над Невою льдистой	*16. Блестит над каменкой тенистой*
17. Плешивый щеголь, враг труда	*17. Над нами царствовал тогда*
18. Когда не наши повара	*18. У Бонапартова шатра*
19. Настала — кто тут нам помог?	*19. Барклай, зима иль русский бог?*
20. И скоро силою вещей	*20. А русский царь главой царей*
21. О русский глупый наш народ	*21. .....*
22. Тебе б я оду посвятил	*22. Меня уже предупредил*
23. Ханжа запрется в монастырь	*23. Семействам возвратит Сибирь*
24. Пред кем унизились цари	*24. Исчезнувший как тень зари*
25. Волкан Неаполя пылал	*25. Из Кишинева уж мигал*
26. Наш царь в конгрессе говорил	*26. Ты александровский холоп (?)*
27. Дружина старых усачей	*27. Свирепой шайке палачей*
28. И пуще царь пошел кутить	*28. Уже издавна, может быть*
29. Они за чашею вина	*29. .....*
30. Сбирались члены сей семьи	*30. У осторожного Ильи*
31. Тут Лунин дерзко предлагал	*31. И вдохновенно бормотал*
32. Но там, где ранее весна	*32. И над холмами Тульчина*

Без особых усилий Трубачевский прочитал рукопись, и ничего не понял. Он переписал ее, получилась бессвязная чепуха, в которой одна строка, едва начавшая мысль, перебивается другой, а та — третьей, еще более бессмысленной и бессвязной. Он попробовал разбить рукопись на строфы, — опять не получилось. Стал искать рифмы, — как будто и рифм не было, хотя на белый стих все это мало похоже. Просчитал строку — четырехстопный ямб, размер, которым написан «Евгений Онегин».

Трубачевский с азартом взялся за рукопись, пытался читать ее, пропуская по одной строке, потом по две, по три, надеясь случайно угадать тайную последовательность, в которой были записаны строки. У него ничего не получалось. Тогда он стал читать третью строку вслед за первой, пятую за третьей, восьмую за пятой, предположив, что пропуски должны увеличиваться в арифметической прогрессии. Все то же! Отчаявшись, он бросил эту затею. Однако, она не давала ему покоя ни на лекции, ни в трамвае ... Как шахматист, играющий в уме, он не только знал наизусть каждую строчку, он видел ее в десяти комбинациях сразу.

Прошло время. Однажды, когда он смотрел на светлые пятна окон подходящего к перрону поезда, каким-то внутренним зрением он

увидел перед собой всю рукопись — и с такой необыкновенной отчетливостью, как это бывает только во сне.

Сможете ли вы прочитать эти стихи? Ответ вы найдете в романе В. Каверина.

Ответы к упражнению.

1) Шифр маршрутной перестановки

1	2	3	4	5	6	7	8	9	10	11	12	13	14
25	24	17	16	9	8	1	2	7	10	15	18	23	26

15	16	17	18	19	20	21	22	23	24	25	26	27	28
27	22	19	14	11	6	3	4	5	12	13	20	21	28

2) Шифр «решетка»

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	11	15	17	18	22	26	30	34	38	42	53	56	57	60

16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	4	5	8	19	23	27	31	35	39	43	44	46	50	59

31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
3	6	7	10	12	24	28	32	36	40	41	45	47	48	52

46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
9	13	14	16	20	21	25	29	33	37	49	51	54	55	58

3) ШВП

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
23	1	17	34	7	29	12	24	2	18	35	8	30	13	25

16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
3	19	36	9	31	14	26	4	20	37	10	32	15	27	5

31	32	33	34	35	36	37	38
21	38	11	33	16	28	6	22

4. Многоалфавитные шифры замены с периодическим ключом

Рассмотрим 30-буквенный алфавит русского языка:

А Б В Г Д Е Ж З И К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Э Ю Я.

В этом алфавите отсутствуют буквы Ё, Й и Ъ, что практически не ограничивает возможностей по составлению открытых сообщений на русском языке. В самом деле, замена буквы Ё на букву Е, буквы Й — на букву И, а буквы Ъ — на букву Ь позволяет понять смысл открытого сообщения, написанного с использованием этого алфавита.

В алфавите любого естественного языка буквы следуют друг за другом в определенном порядке. Это дает возможность присвоить каждой букве алфавита ее естественный порядковый номер. Так, в приведенном алфавите букве А присваивается порядковый номер 1, букве О — порядковый номер 14, а букве Ы — порядковый номер 27. Если в открытом сообщении каждую букву заменить ее естественным порядковым номером в рассматриваемом алфавите, то преобразование числового сообщения в буквенное позволяет однозначно восстановить исходное открытое сообщение. Например, числовое сообщение 1 11 20 1 3 9 18 преобразуется в буквенное сообщение: АЛФАВИТ.

Дополним естественный порядок букв в алфавите. Будем считать, что за последней буквой алфавита следует его первая буква. Такой порядок букв достигается, если расположить их на окружности в естественном порядке по часовой стрелке. При таком расположении можно каждой из букв присвоить порядковый номер относительно любой буквы алфавита. Такой номер назовем относительным порядковым номером. Заметим, что если число букв в алфавите равно z, то относительный порядковый номер данной буквы может принимать все значения от 0 до $(z-1)$ в зависимости от буквы, относительно которой он вычисляется. Для примера рассмотрим исходный 30-буквенный алфавит русского языка, расположенный на окружности (см. рис.).

В этом случае порядковый номер буквы А относительно буквы А равен 0, относительно буквы Я он уже равен 1 и так далее, относительно буквы Б порядковый номер А равен 29. Значения относительных порядковых номеров букв алфавита из z букв совпадают со значениями всевозможных остатков от деления целых чисел на натуральное число z. Убедитесь в том, что порядковый номер какой-либо буквы алфавита относительно другой буквы равен остатку от деления разности их естественных порядковых номеров на число букв в алфавите.

Обозначим символами:

$D(N_1, N_2)$ — порядковый номер буквы с естественным порядковым номером N_1 относительно буквы с естественным порядковым номером N_2;

$r_m(N)$ — остаток от деления целого числа N на натуральное число m.

При этом справедливо равенство $D(N_1, N_2) = r_z(N_1 - N_2)$, где z — число букв в алфавите.

Для удобства обозначим $N_1 \boxminus N_2 = r_z(N_1 - N_2)$, $N_1 \boxplus N_2 = r_z(N_1 + N_2)$. Тогда имеют место равенства:

$$D(N_1, N_2) = N_1 \boxminus N_2, \tag{7}$$

$$N_1 = N_2 \boxplus D(N_1, N_2). \tag{8}$$

Формула (8) непосредственно получается из (7) и ее можно использовать для замены буквы с естественным порядковым номером N_2 на букву с естественным порядковым номером N_1. Число $D(N_1, N_2)$ называется знаком гаммы.

Для уяснения введенных обозначений читателю предлагается самостоятельно решить следующие задачи.

1. Докажите, что для любых целых N_1, N_2 и любого натурального z справедливо равенство: $D(N_1, N_2) = N_1 - N_2 - \left[\dfrac{N_1 - N_2}{z}\right] \cdot z$, где $[X]$ — целая часть числа X (наибольшее целое число, не превосходящее числа X).

2. Докажите равенство (8) и равенство:

$$N_2 = N_1 \boxminus D(N_1, N_2). \tag{9}$$

Для зашифрования некоторого открытого сообщения, состоящего из N букв, с помощью указанной замены требуется N знаков гаммы: по одному на каждую букву сообщения. Последовательность знаков гаммы, необходимая для зашифрования открытого сообщения, является ключом данного шифра.

Если последовательность знаков гаммы имеет небольшой (по сравнению с длиной открытого текста) период, то соответствующий шифр называется шифром замены с периодическим ключом. Ключом такого шифра, по существу, является отрезок гаммы, равный по длине периоду.

Число отрезков некоторой длины T, состоящих из чисел от 0 до $(z-1)$ равно z^T, так как на каждой из T позиций отрезка может быть любое из z чисел (независимо от чисел, находящихся на других позициях). Для наглядности приведем значения z^T при $z = 30$ в зависимости от значений T:

T	1	2	3	4	5	6	7
30^T	30	900	27000	810000	24300000	$0,729 \cdot 10^9$	$0,2187 \cdot 10^{11}$

T	8	9	10
30^T	$0,6561 \cdot 10^{12}$	$0,19683 \cdot 10^{14}$	$0,59049 \cdot 10^{15}$

Как видно из приведенной таблицы, число ключей рассматриваемого шифра замены с ключом периода 10, достаточно внушительно и составляет уже сотни триллионов. Это обстоятельство делает практически невозможным вскрытие шифра методом перебора всех его ключей даже при меньших значениях периода гаммы.

Для рассматриваемого шифра характерно то, что буквы открытого текста, зашифрованные одним и тем же знаком гаммы, по сути, зашифрованы одним и тем же шифром простой замены. Например, ключевая таблица этого шифра простой замены при знаке гаммы, равном 1, имеет вид:

АБВГДЕЖЗИКЛМНОПРСТУФХЦЧШЩЬЫЭЮЯ
БВГДЕЖЗИКЛМНОПРСТУФХЦЧШЩЬЫЭЮЯА

Вторую строку этой ключевой таблицы называют алфавитом шифрования, соответствующим данному знаку гаммы. Поскольку в рассматриваемом шифре возможны все значения гаммы от 0 до 29, то данный шифр можно рассматривать как 30-алфавитный шифр замены. Если каждому из этих алфавитов поставить в соответствие его первую букву, то каждый знак гаммы можно заменить этой буквой. В этом случае ключ рассматриваемого шифра можно взаимнооднозначно заменить соответствующим словом в этом же алфавите. Такой многоалфавитный шифр замены был описан в 1585 году французом Блезом де Виженером в его «Трактате о шифрах»:

ABCDEFGHIJKLMNOPQRSTUVWXYZ
BCDEFGHIJKLMNOPQRSTUVWXYZA
CDEFGHIJKLMNOPQRSTUVWXYZAB
DEFGHIJKLMNOPQRSTUVWXYZABC
EFGHIJKLMNOPQRSTUVWXYZABCD
FGHIJKLMNOPQRSTUVWXYZABCDE
GHIJKLMNOPQRSTUVWXYZABCDEF
HIJKLMNOPQRSTUVWXYZABCDEFG
IJKLMNOPQRSTUVWXYZABCDEFGH
JKLMNOPQRSTUVWXYZABCDEFGHI
KLMNOPQRSTUVWXYZABCDEFGHIJ
LMNOPQRSTUVWXYZABCDEFGHIJK
MNOPQRSTUVWXYZABCDEFGHIJKL
NOPQRSTUVWXYZABCDEFGHIJKLM
OPQRSTUVWXYZABCDEFGHIJKLMN
PQRSTUVWXYZABCDEFGHIJKLMNO
QRSTUVWXYZABCDEFGHIJKLMNOP
RSTUVWXYZABCDEFGHIJKLMNOPQ
STUVWXYZABCDEFGHIJKLMNOPQR
TUVWXYZABCDEFGHIJKLMNOPQRS
UVWXYZABCDEFGHIJKLMNOPQRST
VWXYZABCDEFGHIJKLMNOPQRSTU
WXYZABCDEFGHIJKLMNOPQRSTUV
XYZABCDEFGHIJKLMNOPQRSTUVW
YZABCDEFGHIJKLMNOPQRSTUVWX
ZABCDEFGHIJKLMNOPQRSTUVWXY

Все алфавиты шифрования относительно латинского алфавита были сведены им в таблицу, получившую впоследствии название ее автора. Выше приведена таблица Виженера для современного латинского алфавита, она состоит из списка 26 алфавитов шифрования. Способ зашифрования с помощью таблицы Виженера заключается в том, что первый из алфавитов соответствует алфавиту открытого текста, а букве ключевого слова соответствует алфавит шифрования из данного списка, начинающийся с этой буквы. Буква шифрованного текста находится в алфавите шифрования на месте, соответствующем данной букве открытого текста. Простота построения таблицы Виженера делает эту систему привлекательной для практического использования. Рассмотрим пример вскрытия многоалфавитного шифра замены с периодическим ключом, содержащийся в рассказе Жюля Верна «Жангада». Вот текст, который был получен с помощью такого типа шифра:

С Г У Ч П В Э Л Л З Й Р Т Е П Н Л Н Ф Г И Н Б О Р Г Й У
Г Л Ч Д К О Т Х Ж Г У У М З Д Х Р Ъ С Г С Ю Д Т П Ъ А Р
В Й Г Г И Щ В Ч Э Е Ц С Т У Ж В С Е В Х А Х Я Ф Б Ь Б Е
Т Ф З С Э Ф Т Х Ж З Б З Ъ Г Ф Б Щ И Х Х Р И П Ж Т З В Т
Ж Й Т Г О Й Б Н Т Ф Ф Е О И Х Т Т Е Г И И О К З П Т Ф Л
Е У Г С Ф И П Т Ь М О Ф О К С Х М Г Б Т Ж Ф Ы Г У Ч О Ю
Н Ф Н Ш З Г Э Л Л Ш Р У Д Е Н К О Л Г Г Н С Б К С С Е У
П Н Ф Ц Е Е Е Г Г С Ж Н О Е Ы И О Н Р С И Т К Ц Ь Е Д Б
У Б Т Е Т Л О Т Б Ф Ц С Б Ю Й П М П З Т Ж П Т У Ф К Д Г

Догадавшись, что ключом является натуральное число, персонаж «Жангады», судья Жаррикес, объясняет сыну обвиняемого Маноэлю, как был зашифрован документ: —«Давайте возьмем фразу, все равно какую, ну хотя бы вот эту:

У СУДЬИ ЖАРРИКЕСА ПРОНИЦАТЕЛЬНЫЙ УМ

А теперь я возьму наудачу какое-нибудь число, чтобы сделать из этой фразы криптограмму. Предположим, что число состоит из трех цифр, например, 4, 2 и 3. Я подписываю число 423 под строчкой так, чтобы под каждой буквой стояла цифра, и повторяю число, пока не дойду до конца фразы. Вот что получится:

У С У Д Ь И Ж А Р Р И К Е С А П Р О Н И Ц А Т Е Л Ь Н Ы Й У М
4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4

Будем заменять каждую букву нашей фразы той буквой, которая стоит после нее в алфавите

АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ

на месте, указанном цифрой. Например, если под буквой А стоит цифра 3, вы отсчитываете три буквы и заменяете ее буквой Г. Если буква

находится в конце алфавита и к ней нельзя прибавить нужного числа букв, тогда отсчитывают недостающие буквы с начала алфавита.

Доведем до конца начатую криптограмму, построенную на числе 423, и исходная фраза заменится следующей:

ЧУЦИЮЛКВУФКНЙУГУТССКЩДФИПЮРЯЛЦР

Но как найти числовой ключ? Подсчет, проведенный Жаррикесом, показывает, что поиск ключа перебором всех возможных чисел, состоящих не более чем из 10 цифр, потребует более трехсот лет. Судья пытается наудачу отгадать заветное число. Наступает день казни. Обвиняемого Жоама Дакосту ведут на виселицу ...

Но все заканчивается благополучно. Помог счастливый случай. Другу Жоама удается узнать, что автора криптограммы звали Ортега. Поставив буквы О, Р, Т, Е, Г, А над последними шестью буквами документа и подсчитав, на сколько эти буквы по алфавиту сдвинуты относительно букв криптограммы, судья, наконец, находит ключ к документу:

исходное сообщение	О	Р	Т	Е	Г	А
шифрованное сообщение	Т	У	Ф	К	Д	Г
относительный сдвиг букв	4	3	2	5	1	3

Г. А. Гуревич в статье «Криптограмма Жюля Верна» (журнал «Квант» №9, 1985 г.) обращает внимание на то, что судья прошел практически весь путь до отгадки. Будучи уверенным, что в документе упоминается имя Жоама Дакосты, судья строит предположение: «Если бы строчки были разделены на слова, то мы могли бы выделить слова, состоящие из семи букв, как и фамилия Дакоста, и, опробуя их одно за другим, может быть и отыскали бы число, являющееся ключом криптограммы». Маноэль, в свою очередь, поняв основную идею судьи, предлагает опробовать возможные расположения слова ДАКОСТА в исходном тексте. Поскольку текст состоит из 252 букв, то достаточно опробовать не более 246 вариантов. В один прекрасный момент, записав над фрагментом ЙБНТФФЕ слово ДАКОСТА, мы определили бы последовательность цифр 5134325. Естественно предположить, что последняя цифра 5 — начало следующего периода:

исходное сообщение	...	Д	А	К	О	С	Т	А	...
шифрованное сообщение	...	Й	Б	Н	Т	Ф	Ф	Е	...
относительный сдвиг букв	...	5	1	3	4	3	2	5	...

Вместо ключа 432513 мы нашли его циклическую перестановку 513432, что ни в коей мере не мешает расшифрованию текста. Для этого достаточно для каждой буквы шифрованного текста определить букву, относительно которой данная буква сдвинута на величину соответствующей цифры ключа:

С Г У Ч П В Э Л Л З Й Р Т Е П Н Л Н Ф Г И Н Б О Р
4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4
Н А С Т О Я Щ И Й В И Н О В Н И К К Р А Ж И А Л М

Г Й У Г Л Ч Д К О Т Х Ж Г У У М З Д Х Р Ъ С Г С Ю
3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3
А З О В И У Б И Й С Т В А С О Л Д А Т О Х Р А Н Ы

Д Т П Ъ А Р В Й Г Г И Щ В Ч Э Е Ц С Т У Ж В С Е В
2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2
В Н О Ч Ь Н А Д В А Д Ц А Т Ь В Т О Р О Е Я Н В А

Х А Х Я Ф Б Ь Б Е Т Ф З С Э Ф Т Х Ж З Б З Ъ Г Ф Б
5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5
Р Я Т Ы С Я Ч А В О С Е М Ь С О Т Д В А Д Ц А Т Ь

Щ И Х Х Р И П Ж Т З В Т Ж Й Т Г О Й Б Н Т Ф Ф Е О
1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1
Ш Е С Т О Г О Г О Д А Н Е Ж О А М Д А К О С Т А Н

И Х Т Т Е Г И И О К З П Т Ф Л Е У Г С Ф И П Т Ь М
3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3
Е С П Р А В Е Д Л И В О П Р И Г О В О Р Е Н Н Ы Й

О Ф О К С Х М Г Б Т Ж Ф Ы Г У Ч О Ю Н Ф Н Ш З Г Э
4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4
К С М Е Р Т И А Я Н Е С Ч А С Т Н Ы Й С Л У Ж А Щ

Л Л Ш Р У Д Е Н К О Л Г Г Н С Б К С С Е П У Н Ф Ц
3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 1 5 3 4 3
И Й У П Р А В Л Е Н И Я А Л М А З Н О Г О О К Р У

Е Е Е Г Г С Ж Н О Е Ы И О Н Р С И Т К Ц Ь Е Д Б У
2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2
Г А Д А Я О Д И Н В Ч Е М И П О Д П И С Ы В А Ю С

Б Т Е Т Л О Т Б Ф Ц С Б Ю Й П М П З Т Ж П Т У Ф К
5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5 1 3 4 3 2 5
Ь С В О И М Н А С Т О Я Щ И М И М Е Н Е М О Р Т Е

Д Г
1 3
Г А

Итак, первая идея состоит в использовании вероятного слова, то есть слова, которое с большой вероятностью может содержаться в данном открытом тексте. Речь идет в том числе и о словах, часто встречающихся в любых открытых текстах. К ним, например, относятся такие слова как КОТОРЫЙ, ТОГДА, ЧТО, ЕСЛИ, приставки ПРИ, ПРЕ, ПОД и т. п.

Вторая идея основана на том, что буквы открытого сообщения находятся в открытом тексте на вполне определенных позициях. Если разность номеров их позиций окажется кратной периоду гаммы, то стоящие на этих позициях буквы будут зашифрованы одним и тем же знаком гаммы. Это означает, что определенные части открытого текста окажутся зашифрованными шифром простой замены. Эту идею можно использовать для определения периода ключа многоалфавитного шифра замены.

Для определения периода гаммы могут быть применены два способа. Первый из них известен как тест Казизки, второй способ использует так называемый индекс совпадения.

Тест Казизки был описан в 1863 году Фридрихом Казизки. Он основан на следующем наблюдении: два одинаковых отрезка открытого текста будут соответствовать двум одинаковым отрезкам шифрованного текста, если разность номеров позиций их начал кратна периоду гаммы. Следовательно, если мы обнаружим два одинаковых отрезка шифрованного текста, состоящих по крайней мере из трех букв, то с большой вероятностью им соответствуют одинаковые отрезки открытого текста (случайное совпадение маловероятно). Тест Казизки, по сути, заключается в том, что в шифрованном тексте надо найти пары одинаковых отрезков, вычислить разности номеров позиций их начал и определить общие делители найденных разностей. Как правило, один из этих общих делителей равен периоду гаммы.

Для уточнения значения периода гаммы может быть использован индекс совпадения, предложенный в 1920 году Уильямом Фридманом. Для последовательности букв индекс совпадения представляет собой число, равное количеству всех пар номеров позиций последовательности, на которых находятся одинаковые буквы, деленному на общее количество всех пар номеров позиций этой последовательности, т. е. среднему числу пар, состоящих из одинаковых букв. Примечательно то, что при зашифровании последовательности с помощью шифра простой замены указанное число не меняется.

Для иллюстрации этого подхода рассмотрим тот же самый шифрованный текст, записанный в виде последовательности столбцов, содержащих по шесть подряд идущих букв текста в каждом (подряд идущие буквы текста располагаются в столбцах сверху вниз):

СЭТФРЧЖД С А И Ц С Я Т Т Ъ Х Т Т Т Х И Ф Ф О М Ы Н Э Д Г С Ф Г Ы И Д Т Ц М Т
Г Л Е Г Г Д Г Х Ю Р Щ С Е Ф Ф Х Г Х З Г Ф Т О Л И Ф Г Г Ф Л Е Г С Ц С И Т Б Л С П У
У Л П И Й К У Р Д В В Т В Б З Ж Ф Р В О Ф Т К Е П О Б У Н Л Н Н Е Е Ж О К У О Б З Ф
Ч З Н Н У О У Ъ Т Й Ч У Х Ь С З Б И Т Й Е Е З У Т К Т Ч Ш Ш К С У Е Н Н Ц Б Т Ю Т К
П Й Л Б Г Т М С П Г Э Ж А Б Э Б Щ П Ж Б О Г П Г Ь С Ж О З Р О Б П Е О Р Ь Т Б Й Ж Д
В Р Н О Л Х З Г Ъ Г Е В Х Е Ф З И Ж Й Н И И Т С М Х Ф Ю Г У Л К Н Г Е С Е Е Ф П П Г

Составим для каждой из 6 получившихся строк соответствующий ей набор частот встречаемости букв в каждой из них:

	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
1 строка	1	0	0	2	3	0	1	0	3	0	0	0	2	1	1	0
2 строка	0	1	0	9	1	3	0	1	2	0	0	4	0	0	1	1
3 строка	0	3	4	0	1	3	2	2	1	1	3	2	0	3	4	2
4 строка	0	2	0	0	0	4	0	3	1	2	3	0	0	4	1	0
5 строка	1	6	0	4	1	1	4	1	0	2	0	0	1	0	4	5
6 строка	0	0	2	5	0	5	1	2	3	1	1	2	1	3	1	2

	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
1 строка	1	4	8	0	4	2	2	1	0	0	1	2	0	2	0	1
2 строка	1	4	2	1	5	3	1	0	0	1	0	0	0	0	1	0
3 строка	2	0	2	4	3	0	0	0	0	0	0	0	0	0	0	0
4 строка	0	2	6	4	0	1	1	3	2	0	1	0	1	0	1	0
5 строка	2	2	2	0	0	0	0	0	0	1	0	0	2	2	0	0
6 строка	1	2	1	1	3	3	0	0	0	0	1	0	0	0	1	0

По этой таблице частот встречаемости букв вычислим для каждой строки соответствующий ей индекс совпадения:

Номер строки	1	2	3	4	5	6
Индекс совпадения	0,060	0,077	0,045	0,053	0,057	0,057

Для всего шифрованного текста индекс совпадения равен 0,040, что заметно меньше, чем индекс совпадения для каждой из указанных строк. Это является хорошим подтверждением гипотезы о длине периода гаммы.

Другие идеи подходов к вскрытию рассматриваемых шифров основаны на тех или иных особенностях их построения и использования (см. решения задач 1.2, 2.2, 2.5, 2.6, 3.4, 3.5, 4.6).

5. Условия задач олимпиад по математике и криптографии

Ниже приводятся задачи семи олимпиад по криптографии и математике. Нумерация задач двойная: первая цифра — номер олимпиады, вторая — номер задачи в олимпиаде. Для решения задач не требуется специальных знаний. Все необходимые определения даны в условиях. Задачи рассчитаны на учащихся 9, 10 и 11 классов.

1.1. Ключом шифра, называемого «поворотная решетка», является трафарет, изготовленный из квадратного листа клетчатой бумаги размера $n \times n$ (n — четно). Некоторые из клеток вырезаются. Одна из сто-

рон трафарета помечена. При наложении этого трафарета на чистый лист бумаги четырьмя возможными способами (помеченной стороной вверх, вправо, вниз, влево) его вырезы полностью покрывают всю площадь квадрата, причем каждая клетка оказывается под вырезом ровно один раз.

Буквы сообщения, имеющего длину n^2, последовательно вписываются в вырезы трафарета, сначала наложенного на чистый лист бумаги помеченной стороной вверх. После заполнения всех вырезов трафарета буквами сообщения трафарет располагается в следующем положении и т. д. После снятия трафарета на листе бумаги оказывается зашифрованное сообщение.

Найдите число различных ключей для произвольного четного числа n.

1.2. В адрес олимпиады пришло зашифрованное сообщение:

Ф В М Е Ж Т И В Ф Ю

Найдите исходное сообщение, если известно, что шифрпреобразование заключалось в следующем. Пусть x_1, x_2 — корни трехчлена $x^2 + 3x + 1$. К порядковому номеру каждой буквы в стандартном русском алфавите (33 буквы) прибавлялось значение многочлена $f(x) = x^6 + 3x^5 + x^4 + x^3 + 4x^2 + 4x + 3$, вычисленное либо при $x = x_1$, либо при $x = x_2$ (в неизвестном нам порядке), а затем полученное число заменялось соответствующей ему буквой.

1.3. Для передачи информации от резидента Гарриваса в Нагонии только что внедренному разведчику был установлен следующий порядок.

Все сообщения резидента определены заранее и пронумерованы числами 1, 2, 3, ... Разведчик, обладающий феноменальной памятью, полностью запомнил соответствие между сообщениями и их номерами. Теперь для того, чтобы передать информацию разведчику, достаточно было сообщить ему лишь соответствующее число.

Для передачи числа в условленном месте оставлялась равная этому числу денежная сумма.

На момент разработки операции в Нагонии имели хождение денежные купюры достоинством 1,3,7 и 10 бут (бут — денежная единица Нагонии). Однако в результате денежной реформы купюры достоинством 1 и 3 бут были изъяты из обращения.

Выясните, начиная с какого номера можно передать разведчику любое сообщение, пользуясь только оставшимися в обращении купюрами.

1.4. Сколько существует упорядоченных пар натуральных чисел a и b, для которых известны их наибольший общий делитель $d = 6$ и их наименьшее общее кратное $m = 6930$. Сформулируйте ответ и в общем

случае, используя канонические разложения d и m на простые множители.

1.5. Дана криптограмма:

$$\begin{array}{ccccc} \text{ФН} & \times & \text{Ы} & = & \text{ФАФ} \\ + & & \times & & - \\ \text{ЕЕ} & + & \text{Е} & = & \text{НЗ} \\ = & & = & & = \\ \text{ИША} & + & \text{МР} & = & \text{ИМН} \end{array}$$

Восстановите цифровые значения букв, при которых справедливы все указанные равенства, если разным буквам соответствуют различные цифры. Расставьте буквы в порядке возрастания их цифровых значений и получите искомый текст.

1.6. Одна фирма предложила устройство для автоматической проверки пароля. Паролем может быть любой непустой упорядоченный набор букв в алфавите $\{a, b, c\}$. Будем обозначать такие наборы большими латинскими буквами. Устройство перерабатывает введенный в него набор P в набор $Q = \varphi(P)$. Отображение φ держится в секрете, однако про него известно, что оно определено не для каждого набора букв и обладает следующими свойствами. Для любого набора букв P

1) $\varphi(aP) = P$;

2) $\varphi(bP) = \varphi(P)a\varphi(P)$;

3) набор $\varphi(cP)$ получается из набора $\varphi(P)$ выписыванием букв в обратном порядке.

Устройство признает предъявленный пароль верным, если $\varphi(P)=P$. Например, трехбуквенный набор bab является верным паролем, так как $\varphi(bab) = \varphi(ab)a\varphi(ab) = bab$. Подберите верный пароль, состоящий более чем из трех букв.

2.1. В древнем шифре, известном под названием «Сцитала», использовалась полоска папируса, которая наматывалась на круглый стержень виток к витку без просветов и нахлестов. Далее, при горизонтальном положении стержня, на папирус построчно записывался текст сообщения. После этого полоска папируса с записанным на ней текстом посылалась адресату, имеющему точно такой же стержень, что позволяло ему прочитать сообщение.

В наш адрес поступило сообщение, зашифрованное с помощью шифра «Сцитала». Однако ее автор, заботясь о том, чтобы строчки были ровные, во время письма проводил горизонтальные линии, которые остались на полоске в виде черточек между буквами. Угол наклона этих черточек к краю ленты равен α, ширина полоски равна d, а ширина каждой строки равна h. Укажите, как, пользуясь имеющимися данными, прочитать текст.

2.2. Исходное цифровое сообщение коммерсант шифрует и передает. Для этого он делит последовательность цифр исходного сообщения на группы по пять цифр в каждой и после двух последовательных групп приписывает еще две последние цифры суммы чисел, изображенных этими двумя группами. Затем к каждой цифре полученной последовательности он прибавляет соответствующий по номеру член некоторой целочисленной арифметической прогрессии, заменяя результат сложения остатком от деления его на 10.

Найдите исходное цифровое сообщение по шифрованному сообщению:

4 2 3 4 6 1 4 0 5 3 1 3

2.3. Рассмотрим преобразование цифрового текста, в котором каждая цифра заменяется остатком от деления значения многочлена $F(x) = b(x^3 + 7x^2 + 3x + a)$ на число 10, где a, b — фиксированные натуральные числа.

Выясните, при каких значениях a, b указанное преобразование может быть шифрпреобразованием (то есть допускает однозначное расшифрование).

2.4. При установке кодового замка каждой из 26 латинских букв, расположенных на его клавиатуре, сопоставляется произвольное натуральное число, известное лишь обладателю замка. Разным буквам сопоставляются не обязательно разные числа. После набора произвольной комбинации попарно различных букв происходит суммирование числовых значений, соответствующих набранным буквам. Замок открывается, если сумма делится на 26.

Докажите, что для любых числовых значений букв существует комбинация, открывающая замок.

2.5. Сообщение, записанное в алфавите

АБВГДЕЖЗИКЛМНОПРСТУФХЦЧШЩЬЫЭЮЯ

зашифровывается при помощи последовательности букв этого же алфавита. Длина последовательности равна длине сообщения. Шифрование каждой буквы исходного сообщения состоит в сложении ее порядкового номера в алфавите с порядковым номером соответствующей буквы шифрующей последовательности и замене такой суммы на букву алфавита, порядковый номер которой имеет тот же остаток от деления на 30, что и эта сумма.

Восстановите два исходных сообщения, каждое из которых содержит слово КОРАБЛИ, если результат их зашифрования при помощи одной и той же шифрующей последовательности известен:

ЮПТЦАРГШАЛЖЖЕВЦЩЫРВУУ и ЮПЯТБНЩМСДТЛЖГПСГХСЦЦ

2.6. Буквы русского алфавита занумерованы в соответствии с таблицей:

А	Б	В	Г	Д	Е	Ж	З	И	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ы	Э	Ю	Я
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30

Для зашифрования сообщения, состоящего из n букв, выбирается ключ K — некоторая последовательность из n букв приведенного выше алфавита. Зашифрование каждой буквы сообщения состоит в сложении ее номера в таблице с номером соответствующей буквы ключевой последовательности и замене полученной суммы на букву алфавита, номер которой имеет тот же остаток от деления на 30, что и эта сумма.

Прочтите шифрованное сообщение: РБЬНТСИТСРРЕЗОХ, если известно, что шифрующая последовательность не содержала никаких букв, кроме А, Б и В.

3.1. Установите, можно ли создать проводную телефонную сеть связи, состоящую из 993 абонентов, каждый из которых был бы связан ровно с 99 другими.

3.2. Шифрпреобразование простой замены в алфавите $A = \{a_1, a_2, \ldots \ldots, a_n\}$, состоящем из n различных букв, заключается в замене каждой буквы шифруемого текста буквой того же алфавита, причем разные буквы заменяются разными. Ключом шифра простой замены называется таблица, в которой указано, какой буквой надо заменить каждую букву алфавита A. Если слово СРОЧНО зашифровать простой заменой с помощью ключа:

А	Б	В	Г	Д	Е	Ж	З	И	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ы	Э	Ю	Я
Ч	Я	Ю	Э	Ы	Ь	Щ	Ш	Ц	Х	Ф	У	Б	Д	Т	З	В	Р	П	М	Л	К	А	И	О	Ж	Е	С	Г	Н

то получится слово ВЗДАБД. Зашифровав полученное слово с помощью того же ключа еще раз, получим слово ЮШЫЧЯЫ. Сколько всего различных слов можно получить, если указанный процесс шифрования продолжать неограниченно?

3.3. Сообщение, зашифрованное в пункте А шифром простой замены в алфавите из букв русского языка и знака пробела (–) между словами, передается в пункт Б отрезками по 12 символов. При передаче очередного отрезка сначала передаются символы, стоящие на четных местах в порядке возрастания их номеров, начиная со второго, а затем — символы, стоящие на нечетных местах (также в порядке возрастания их номеров), начиная с первого. В пункте В полученное шифрованное сообщение дополнительно шифруется с помощью некоторого другого шифра простой замены в том же алфавите, а затем таким же образом, как и из пункта А, передается в пункт В. По перехваченным в пункте В отрезкам:

С	О	–	Г	Ж	Т	П	Н	Б	Л	Ж	О
Р	С	Т	К	Д	К	С	П	Х	Е	У	Б
–	Е	–	П	Ф	П	У	Б	–	Ю	О	Б
С	П	–	Е	О	К	Ж	У	У	Л	Ж	Л
С	М	Ц	Х	Б	Э	К	Г	О	Щ	П	Ы
У	Л	К	Л	–	И	К	Н	Т	Л	Ж	Г

восстановите исходное сообщение, зная, что в одном из переданных отрезков зашифровано слово КРИПТОГРАФИЯ.

3.4. Дана последовательность чисел $C_1, C_2, \ldots, C_n, \ldots$ в которой C_n есть последняя цифра числа n^n. Докажите, что эта последовательность периодическая и ее наименьший период равен 20.

3.5. Исходное сообщение, состоящее из букв русского алфавита и знака пробела (–) между словами, преобразуется в цифровое сообщение заменой каждого его символа парой цифр согласно следующей таблице:

А	Б	В	Г	Д	Е	Ж	З	И	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ы	Э	Ю	Я	–
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

Для зашифрования полученного цифрового сообщения используется отрезок последовательности из задачи 3.4, начинающийся с некоторого члена C_k. При зашифровании каждая цифра сообщения складывается с соответствующей цифрой отрезка и заменяется последней цифрой полученной суммы. Восстановите сообщение:

2339867216458160670617315588

3.6. Равносторонний треугольник ABC разбит на четыре части так, как показано на рисунке, где M и N — середины сторон AB и BC соответственно. Известно, что $PK \perp MQ$ и $NL \perp MQ$. В каком отношении точки P и Q делят сторону AC, если известно, что из этих частей можно составить квадрат?

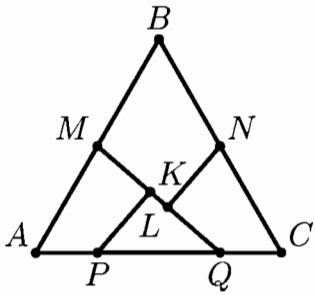


Рис. 6.

4.1. Ключом шифра, называемого «решеткой», является прямоугольный трафарет размера 6×10 клеток. В трафарете вырезаны 15 клеток

так, что при наложении его на прямоугольный лист бумаги размера 6×10 клеток четырьмя возможными способами его вырезы полностью покрывают всю площадь листа.

Буквы сообщения (без пропусков) последовательно вписываются в вырезы трафарета (по строкам, в каждой строке слева направо) при каждом из четырех его возможных положений. Прочтите исходный текст, если после зашифрования на листе бумаги оказался следующий текст (на русском языке):

Р	П	Т	Е	Ш	А	В	Е	С	Л
О	Я	Т	А	Л	–	Ь	З	Т	–
–	У	К	Т	–	Я	А	Ь	–	С
Н	П	–	Ь	Е	У	–	Ш	Л	С
Т	И	Ь	З	Ы	Я	Е	М	–	О
–	Е	Ф	–	–	Р	О	–	С	М

4.2. Криптограмма

12 2 24 5 3 21 6 29 28 2 20 18 20 21 5 10 27 17 2 11 2 16 —
19 2 27 5 8 29 12 31 22 2 16, 19 2 19 5 17 29 8 29 6 29 16:
8 2 19 19 29 10 19 29 14 19 29 29 19 10 2 24 2 11 2 16
10 14 18 21 17 2 20 2 28 29 16 21 29 28 6 29 16.

получена заменой букв на числа (от 1 до 32) так, что разным буквам соответствуют разные числа. Отдельные слова разделены несколькими пробелами, буквы — одним пробелом, знаки препинания сохранены. Буквы «е» и «ё» не различаются. Прочтите четверостишие В. Высоцкого.

4.3. «Шифровальный диск» используется для зашифрования числовых сообщений. Он состоит из неподвижного диска и соосно вращающегося на нем диска меньшего диаметра. На обоих дисках нанесены цифры от 0 до 9, которые расположены в вершинах правильных 10-угольников, вписанных в диски.

Цифра X на неподвижном диске зашифровывается в цифру Y подвижного диска, лежащую на том же радиусе, что и X.

Для построения вписанного 10-угольника без транспортира надо уметь строить угол в $36°$. Попытайтесь вычислить с точностью до 0,1 значение какой-либо тригонометрической функции такого угла без таблиц и калькулятора.

4.4. Зашифрование фразы на латинском языке осуществлено в два этапа. На первом этапе каждая буква текста заменяется на следующую в алфавитном порядке (последняя Z заменяется на первую A). На втором этапе применяется шифр простой замены с неизвестным ключом. Его применение заключается в замене каждой буквы шифруемого текста буквой того же алфавита, при этом разные буквы заменяются

разными буквами. Ключом такого шифра является таблица, в которой указано, какой буквой надо заменить каждую букву алфавита.

По данному шифртексту

OSZJX FXRE YOQJSZ RAYFJ

восстановите открытое сообщение, если известно, что для использованного (неизвестного) ключа результат шифрования не зависит от порядка выполнения указанных этапов для любого открытого сообщения. Пробелы в тексте разделяют слова.

Латинский алфавит состоит из следующих 24 букв:

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

4.5. Для проверки телетайпа, печатающего буквами русского алфавита

АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ

передан набор из 9 слов, содержащий все 33 буквы алфавита. В результате неисправности телетайпа на приемном конце получены слова

ГЪЙ АЭЕ БПРК ЕЖЩЮ НМЬЧ СЫЛЗ ШДУ ЦХОТ ЯФВИ

Восстановите исходный текст, если известно, что характер неисправности таков, что каждая буква заменяется буквой, отстоящей от нее в указанном алфавите не дальше, чем на две буквы. Например, буква Б может перейти в одну из букв {А, Б, В, Г}.

4.6. Исходное сообщение из букв русского алфавита преобразуется в числовое сообщение заменой каждой его буквы числом по следующей таблице:

А	Б	В	Г	Д	Е	Ж	З	И	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ы	Э	Ю	Я
00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29

Для зашифрования полученного числового сообщения используется шифрующий отрезок последовательности $A_1, A_2, \ldots$ подходящей длины, начинающийся с A_{100}.

При зашифровании каждое число числового сообщения складывается с соответствующим числом шифрующего отрезка. Затем вычисляется остаток от деления полученной суммы на 30, который по данной таблице заменяется буквой. Восстановите сообщение КЕНЗЭРЕ, если шифрующий отрезок взят из последовательности, у которой $A_1 = 3$ и $A_{k+1} = A_k + 3(k^2 + k + 1)$ для любого натурального k.

4.7. Чтобы запомнить периодически меняющийся пароль в ЭВМ, математики придумали следующий способ. При известном числе a (например, номере месяца в году), пароль представляет собой первые шесть цифр наименьшего решения уравнения

$$a(x^2 - 1) = \sqrt{1 + \frac{x}{a}}.$$

(Число меньшей значности дополняется справа необходимым числом нулей.)

Решите такое уравнение при произвольном $a > 0$.

5.1. Комбинация (x, y, z) трех натуральных чисел, лежащих в диапазоне от 10 до 20 включительно, является отпирающей для кодового замка, если выполнено соотношение $F(x, y, z) = 99$. Найдите все отпирающие комбинации для замка с

$$F(x, y, z) = 3x^2 - y^2 - 7z.$$

5.2. Сообщение было построчно записано в таблицу, имеющую 20 столбцов. При этом в каждую клетку таблицы записывалось по одной букве сообщения, пробелы между словами были опущены, а знаки препинания заменены на условные комбинации: точка — ТЧК, запятая — ЗПТ. Затем столбцы таблицы были некоторым образом переставлены, в результате чего был получен текст:

Я Н Л В К Р А Д О Е Т Е Р Г О М И З Я Е
Й Л Т А Л Ф Ы И П Е У И О О Г Е Д Б О Р
Ч Р Д Ч И Е С М О Н Д К Х И Н Т И К Е О
Н У Л А Е Р Е Б Ы Ы Е Е З И О Н Н Ы Ч Д
Ы Т Д О Е М П П Т Щ В А Н И П Т Я З С Л
И К С И – Т Ч Н О – – Е – Л У Л – Т – Ж

Прочтите исходное сообщение.

5.3. Из точки O внутри треугольника ABC на его стороны AB, BC, AC опущены перпендикуляры OP, OQ, OR. Докажите, что $OA + OB + OC \geqslant 2(OP + OQ + OR)$.

5.4. Зашифрование сообщения состоит в замене букв исходного текста на пары цифр в соответствии с некоторой (известной только отправителю и получателю) таблицей, в которой разным буквам алфавита соответствуют разные пары цифр. Криптографу дали задание восстановить зашифрованный текст. В каком случае ему будет легче выполнить задание: если известно, что первое слово второй строки — «термометр» или что первое слово третьей строки — «ремонт»? Обоснуйте свой ответ. (Предполагается, что таблица зашифрования криптографу неизвестна).

5.5. Решите уравнение:

$$\sqrt{3x+1}\sqrt{3x+71} - (7 + \sqrt{2x-1})\sqrt{2x + 14\sqrt{2x-1} + 118} = 0.$$

5.6. При передаче сообщений используется некоторый шифр. Пусть известно, что каждому из трех шифрованных текстов

ЙМЫВОТСЬЛКЪГВЦАЯЯ
УКМАПОЧСРКЩВЗАХ
ШМФЭОГЧСЙЪКФЬВЫЕАКК

соответствовало исходное сообщение МОСКВА. Попробуйте расшифровать три текста

ТПЕОИРВНТМОЛАРГЕИАНВИЛЕДНМТААГТДЬТКУБЧКГЕИШНЕИАЯРЯ
ЛСИЕМГОРТКРОМИТВАВКНОПКРАСЕОГНАЬЕП
РТПАИОМВСВТИЕОБПРОЕННИГЬКЕЕАМТАЛВТДЬСОУМЧШСЕОНШЬИАЯК

при условии, что двум из них соответствует одно и то же сообщение. Сообщениями являются известные крылатые фразы.

6.1. В системе связи, состоящей из 1997 абонентов, каждый абонент связан ровно с N другими. Определите все возможные значения N.

6.2. Квадратная таблица размером 1997×1997 заполнена натуральными числами от 1 до 1997 так, что в каждой строке присутствуют все числа от 1 до 1997. Найдите сумму чисел, стоящих на диагонали, которая соединяет левый верхний и правый нижний углы таблицы, если заполнение таблицы симметрично относительно этой диагонали.

6.3. Текст

М И М О П Р А С Т Е Т И Р А С И С П Д
И С А Ф Е И И Б О Е Т К Ж Р Г Л Е О Л О
И Ш И С А Н Н С Й С А О О Л Т Л Е Я Т У
И Ц В Ы И П И Я Д П И Щ П Ь П С Е Ю Я Я

получен из исходного сообщения перестановкой его букв. Текст

У Щ Ф М Ш П Д Р Е Ц Ч Е Ш Ю Ш Ч Д А К Е
Ч М Д В К Ш Б Е Е Ч Д Ф Э П Й Щ Г Ш Ф Щ
Ц Е Ю Щ Ф П М Е Ч П М Е Р Щ М Е О Ф Ч Щ
Х Е Ш Р Т Г Д И Ф Р С Я Ы Л К Д Ф Ф Е Е

получен из того же исходного сообщения заменой каждой буквы на другую букву так, что разные буквы заменены разными, а одинаковые — одинаковыми. Восстановите исходное сообщение.

6.4. На каждой из трех осей установлено по одной вращающейся шестеренке и неподвижной стрелке. Шестеренки соединены последовательно. На первой шестеренке 33 зубца, на второй — 10, на третьей — 7. На каждом зубце первой шестеренки по часовой стрелке написано по одной букве русского языка в алфавитном порядке:

А Б В Г Д Е Ё Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я

На зубцах второй и третьей шестеренки в порядке возрастания по часовой стрелке написаны цифры от 0 до 9 и от 0 до 6 соответственно. Когда стрелка первой оси указывает на букву, стрелки двух других осей указывают на цифры.

Буквы сообщения шифруются последовательно. Зашифрование производится вращением первой шестеренки против часовой стрелки до первого попадания шифруемой буквы под стрелку. В этот момент последовательно выписываются цифры, на которые указывают вторая и

третья стрелки. В начале шифрования стрелка 1-го колеса указывала на букву А, а стрелки 2-го и 3-го колес — на цифру 0.

а) зашифруйте слово О Л И М П И А Д А;

б) расшифруйте сообщение 2 4 8 0 9 2 8 3 9 1 1 2 1 1.

6.5. Цифры от 1 до 9 расположены на окружности в некотором неизвестном порядке. При зашифровании цифрового сообщения каждая отличная от 0 цифра заменяется на соседнюю с ней цифру на окружности по часовой стрелке, а при расшифровании — на соседнюю с ней цифру на окружности против часовой стрелки. Цифра 0 остается без изменения в обоих случаях.

Укажите условия, при которых порядок цифр на данной окружности можно однозначно восстановить по двум цифровым текстам — результатам расшифрования и зашифрования одного и того же цифрового текста с помощью данной окружности.

6.6. Докажите, что для каждого простого числа p последовательность $a_1, a_2, a_3, \ldots$ является периодической с периодом 2, если a_n равно остатку от деления числа p^{n+2} на 24 при всех $n \geqslant 1$.

6.7. Найдите все значения параметра a, при которых уравнение

$$\underbrace{\Big|\ldots\Big|}_{1996\ \text{раз}}\,x-\underbrace{a\Big|-a\Big|-\ldots-}_{1996\ \text{раз}}\Big| = 1996.$$

имеет ровно 1997 различных решений.

7.1. Какое наименьшее число соединений требуется для организации проводной сети связи из 10 узлов, чтобы при выходе из строя любых двух узлов связи сохранялась возможность передачи информации между любыми двумя оставшимися (хотя бы по цепочке через другие узлы)?

7.2. В компьютерной сети используются пароли, состоящие из цифр. Чтобы избежать хищения паролей, их хранят на диске в зашифрованном виде. При необходимости использования происходит однозначное расшифрование соответствующего пароля. Зашифрование пароля происходит посимвольно одним и тем же преобразованием. Первая цифра остается без изменения, а результат зашифрования каждой следующей цифры зависит только от нее и от предыдущей цифры.

Известен список зашифрованных паролей:

4249188780319, 4245133784397, 5393511, 428540012393,
4262271910365, 4252370031465, 4245133784735

и два пароля 4208212275831, 4242592823026, имеющиеся в зашифрованном виде в этом списке. Можно ли определить какие-либо другие пароли? Если да, то восстановите их.

7.3. В результате перестановки букв сообщения получена криптограмма:

БТИПЧЬЛОЯЧЫЬТОТПУНТНОНЗЛЖАЧОЬОТУНИУХНИППОЛОЬЧОЕЛОЛС

Прочтите исходное сообщение, если известно, что оно было разбито на отрезки одинаковой длины r, в каждом из которых буквы переставлены одинаково по следующему правилу. Буква отрезка, имеющая порядковый номер x ($x = 1, 2, \ldots, r$), в соответствующем отрезке криптограммы имеет порядковый номер $f(x) = ax \oplus b$, где a и b — некоторые натуральные числа, $ax \oplus b$ равно остатку от деления суммы $ax + b$ на r, если остаток не равен нулю, и равно r, если остаток равен нулю.

7.4. Знаменитый математик Леонард Эйлер в 1759 г. нашел замкнутый маршрут обхода всех клеток шахматной доски ходом коня ровно по одному разу. Прочтите текст, вписанный в клетки шахматной доски по такому маршруту (см. рис. 7). Начало текста в а4.

Д	Л	Р	И	Л	П	Н	Б
У	К	А	О	Т	У	С	Т
О	О	О	А	Н	О	И	Р
Т	Б	Г	К	Т	Т	У	К
К	О	Е	О	Р	А	В	О
К	Д	Г	П	В	Л	Е	Т
Т	А	Н	Р	М	А	П	О
Е	А	О	В	И	Д	У	Л

Рис. 7.

7.5. При $a > 0$, $b > 0$, $c > 0$ докажите неравенство:

$$a^3 + b^3 + c^3 + 6abc > \frac{1}{4}(a + b + c)^3.$$

7.6. Для рисования на большой прямоугольной доске используется мел с квадратным сечением со стороной 1 см. При движении мела стороны сечения всегда параллельны краям доски. Как начертить выпуклый многоугольник площадью 1 м^2 с наименьшей площадью границы (площадь границы не входит в площадь многоугольника)?

7.7. Цифры $0, 1, \ldots, 9$ разбиты на несколько непересекающихся групп. Из цифр каждой группы составляются всевозможные числа, для записи каждого из которых все цифры группы используются ровно один раз (учитываются и записи, начинающиеся с нуля). Все полученные числа расположили в порядке возрастания и k-ому числу поставили в соответствие k-ую букву алфавита

АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ

Оказалось, что каждой букве соответствует число и каждому числу соответствует некоторая буква. Шифрование сообщения осуществляется заменой каждой буквы соответствующим ей числом. Если ненулевое число начинается с нуля, то при шифровании этот нуль не выписывается. Восстановите сообщение 873146507381 и укажите таблицу замены букв числами.

6. Указания и решения

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

1.1. Все клетки квадрата размера $n \times n$ разобьем на непересекающиеся группы по четыре клетки в каждой. Отнесем клетки к одной и той же группе, если при каждом повороте квадрата до его самосовмещения они перемещаются на места клеток этой же группы. На рисунке показано такое разбиение на группы всех клеток квадрата 6×6, причем клетки одной группы помечены одной и той же цифрой. Всего таких групп будет $n^2/4$ (целое, так как n — четное число). При наложении трафарета на квадрат ровно одна клетка из каждой группы окажется под его вырезами. Каждому трафарету поставим в соответствие упорядоченный набор всех клеток из таких групп, оказавшихся под вырезами трафарета при наложении его на квадрат помеченной стороной вверх. Такое соответствие является взаимнооднозначным, поскольку каждому ключу будет однозначно соответствовать упорядоченный набор из $n^2/4$ клеток (по одной из каждой группы), вырезанных в трафарете, и наоборот. Всего таких наборов $4^{n^2/4}$. В самом деле, существует ровно четыре различных варианта выбора клетки из каждой группы независимо от выбранных клеток из других таких групп. Таким образом, число различных ключей шифра «поворотная решетка» при четных значениях n равно $4^{n^2/4}$.

1.2. Легко видеть, что $f(x) = (x^2 + 3x + 1)(x^4 + x + 1) + 2$. Отсюда $f(x_1) = f(x_2) = 2$, где x_1, x_2 — корни многочлена $x^2 + 3x + 1$. Получаем

Буква ш.с.	Ф	В	М	Е	Ж	Т	И	В	Ф	Ю
Номер	22	3	14	7	8	20	10	3	22	32

Номер	20	1	12	5	6	18	8	1	20	30
Буква о.с.	Т	А	К	Д	Е	Р	Ж	А	Т	Ь

Ответ: ТАКДЕРЖАТЬ

1.3. *Ответ*: начиная с 54.

1.4. Разложим числа m и d на простые множители: $d = 6 = 2 \cdot 3$; $m = 6930 = 2 \cdot 3 \cdot 3 \cdot 5 \cdot 7 \cdot 11$. Обозначим буквой t число m/d, равное произведению $3 \cdot 5 \cdot 7 \cdot 11$. Найдем все его делители q вида: $q = 3^x 5^y 7^z 11^u$, где числа x, y, z и u принимают только значения 0 и 1. Тогда, как нетрудно видеть, числа q и t/q окажутся взаимно простыми. Полагая $a = dq$ и $b = dt/q$, получим все искомые пары (a, b). В самом деле, в указанных выше условиях наибольший общий делитель такой пары равен d, а ее наименьшее общее кратное равно $dqt/q = dt = dm/d = m$. Таким

образом, искомое число упорядоченных пар совпадает с числом всех делителей q вида: $3^x5^y7^z11^u$, которое равно числу всех упорядоченных наборов длины 4 и состоящих только из 0 и 1. Число всех таких наборов равно $2^4 = 16$, так как для каждого места в наборах существует ровно 2 варианта его значений независимо от значений на других местах. В общем случае число m/d представляется в виде $m/d = p^ir^j \dots s^h$, где $p, r, \dots, s$ — различные простые числа, а $i, j, \dots, h$ — натуральные числа. Число всех делителей вида: $q = p^xr^y \dots s^z$, где числа $x, y, \dots, z$ принимают только по два значения (0 и соответствующий натуральный показатель степени в представлении числа m/d), равно 2^k, где k — число всех простых делителей числа m/d. Если число различных простых множителей в каноническом разложении числа m/d равно k, то число различных упорядоченных пар (a, b) равно 2^k.

Ответ: 16 пар (пары (a, b) и (b, a) разные). В общем случае число упорядоченных пар равно 2^k, где k — число всех простых делителей m/d.

1.5. Из последней строчки легко заметить, что Ш=0. Тогда из первого столбца находим, что И=1. Затем из последнего столбца находим Ф=2. Итак,

$$\begin{array}{ccccc} 2\text{Н} & \times & \text{Ы} & = & 2\text{А}2 \\ + & & \times & & - \\ \text{ЕЕ} & + & \text{Е} & = & \text{Н}3 \\ = & & = & & = \\ 10\text{А} & + & \text{МР} & = & 1\text{МН} \end{array}$$

Из средней строки ясно, что Н>Е. Из первого столбца находим Е=7. Из средней строки можно вычислить значения Н и З: Н=8 и З=4. Получим

$$\begin{array}{ccccc} 28 & \times & \text{Ы} & = & 2\text{А}2 \\ + & & \times & & - \\ 77 & + & 7 & = & 84 \\ = & & = & & = \\ 10\text{А} & + & \text{МР} & = & 1\text{М}8 \end{array}$$

Далее, последовательно вычисляем значения: А=5,Ы=9, М=6, Р=3. Расставим буквы в порядке возрастания их цифровых значений и получим текст ШИФРЗАМЕНЫ

Ответ: ШИФРЗАМЕНЫ

1.6. *Ответ*: например, *cbcacbc*.

Обозначим $\overline{\varphi(P)}$ — набор $\varphi(P)$, выписанный в обратном порядке.

$$\varphi(cbcacbc) = \overline{\varphi(bcacbc)} = \overline{\varphi(cacbc)a\varphi(cacbc)} =$$
$$= \overline{\overline{\varphi(acbc)}a\overline{\varphi(acbc)}} = \overline{\overline{cbc}a\overline{cbc}} = \overline{cbcacbc} = cbcacbc.$$

В общем случае можно показать, что множество искомых наборов состоит из слов вида:

$$P = \begin{cases} cb\underbrace{c\ldots c}_{k\text{ раз}}acb\underbrace{c\ldots c}_{k\text{ раз}} & k \text{ — нечетное;} \\ b\underbrace{c\ldots c}_{k\text{ раз}}ab\underbrace{c\ldots c}_{k\text{ раз}} & k \text{ — четное.} \end{cases}$$

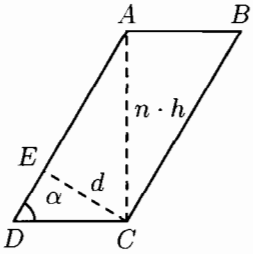


Рис. 8.

2.1. Рассмотрим один виток ленты на развертке цилиндра (разрез по горизонтальной линии). По условию высота CE, опущенная на сторону AD, равна d. Угол DAC равен $(90-\alpha)^\circ$. Отсюда AC равно $d/\cos\alpha$. Так как высота строки равна h, то всего на одном витке $n = d/(h\cdot\cos\alpha)$ букв.

Ответ: чтобы прочитать текст, надо разрезать ленту на участки по $n = d/(h\cdot\cos\alpha)$ букв и сложить их рядом.

2.2. Согласно условию, исходное сообщение состоит из двух пятерок цифр: $A_1A_2A_3A_4A_5$ и $B_1B_2B_3B_4B_5$. Пусть C_1C_2 — последние две цифры суммы чисел, изображенных этими пятерками. Через $a\oplus b$ обозначим последнюю цифру суммы чисел a и b. Пусть D обозначает цифру переноса (цифру десятков) суммы (A_5+B_5). По условию имеем, что $A_5\oplus B_5 = C_2$ и $(A_4\oplus B_4)\oplus D = C_1$.

Пусть Γ_1 — первый член, а X — разность арифметической прогрессии, которую коммерсант использовал при шифровании. Тогда из условия получаем:

$$A_1 \oplus \Gamma_1 = 4, \tag{1}$$
$$A_2 \oplus (\Gamma_1 + X) = 2, \tag{2}$$
$$A_3 \oplus (\Gamma_1 + 2X) = 3, \tag{3}$$
$$A_4 \oplus (\Gamma_1 + 3X) = 4, \tag{4}$$
$$A_5 \oplus (\Gamma_1 + 4X) = 6, \tag{5}$$
$$B_1 \oplus (\Gamma_1 + 5X) = 1, \tag{6}$$
$$B_2 \oplus (\Gamma_1 + 6X) = 4, \tag{7}$$
$$B_3 \oplus (\Gamma_1 + 7X) = 0, \tag{8}$$
$$B_4 \oplus (\Gamma_1 + 8X) = 5, \tag{9}$$
$$B_5 \oplus (\Gamma_1 + 9X) = 3, \tag{10}$$
$$((A_4 \oplus B_4) \oplus D) \oplus (\Gamma_1 + 10X) = 1, \tag{11}$$
$$(A_5 \oplus B_5) \oplus (\Gamma_1 + 11X) = 3. \tag{12}$$

Обозначим символом $A \equiv B$ равенство остатков от деления на 10 чисел A и B. Тогда записи $A\oplus B = C$ и $(A+B)\equiv C$ имеют одинаковый

смысл. Если $A \equiv B$ и $C \equiv D$, то $A+B \equiv C+D$, $A-B \equiv C-D$. Всегда $A \equiv A$, так как остаток от деления единствен.

Из соотношений (4), (5), (9) и (10) находим соответственно:

$$A_4 \equiv 4 - (\Gamma_1 + 3X), \tag{13}$$

$$A_5 \equiv 6 - (\Gamma_1 + 4X), \tag{14}$$

$$B_4 \equiv 5 - (\Gamma_1 + 8X), \tag{15}$$

$$B_5 \equiv 3 - (\Gamma_1 + 9X). \tag{16}$$

Подставляя эти значения в равенства (11) и (12), получим следующие равенства: $9 + D - \Gamma - X \equiv 1$ и $9 - \Gamma - 2X \equiv 3$. Отсюда следует, что

$$X \equiv (-2 - D), \tag{17}$$

$$\Gamma_1 \equiv 2D. \tag{18}$$

Подставив X из (17) и Γ_1 из (18) в (1), (2),(3), (13), (14), (6), (7), (8), (15), (16), найдем выражения для цифр исходного сообщения:

$$A_1 \equiv 4 - 2D, A_2 \equiv 4 - D, A_3 \equiv 7, A_4 \equiv D, A_5 \equiv 4 + 2D,$$
$$B_1 \equiv 1 + 3D, B_2 \equiv 6 + 4D, B_3 \equiv 4 + 5D, B_4 \equiv 1 + 6D,$$
$$B_5 \equiv 1 + 7D.$$

Найденные выражения дают два варианта исходных сообщений:

4470416411 (при $D = 0$),
2371640978 (при $D = 1$).

2.3. *Ответ*: a — любое, b — не должно делиться на 2 и на 5.

Указание. Обозначим через $f(x)$ — остаток от деления значения многочлена $F(x)$ на 10. Для однозначного расшифрования необходимо и достаточно, чтобы разным значениям x соответствовали разные значения $f(x)$. Поэтому $f(0)$, $f(1)$, ... , $f(9)$ принимают все значения от 0 до 9. Найдем эти значения:

$$f(0) = r_{10}(b(a+0)) \qquad f(1) = r_{10}(b(a+1))$$
$$f(2) = r_{10}(b(a+2)) \qquad f(3) = r_{10}(b(a+9))$$
$$f(4) = r_{10}(b(a+8)) \qquad f(5) = r_{10}(b(a+5))$$
$$f(6) = r_{10}(b(a+6)) \qquad f(7) = r_{10}(b(a+7))$$
$$f(8) = r_{10}(b(a+4)) \qquad f(9) = r_{10}(b(a+3)),$$

где $r_{10}(y)$ — остаток от деления числа y на 10.

Отсюда, пользуясь свойствами остатков, замечаем, что b должно быть нечетным (иначе $f(x)$ будут только четные числа) и b не должно делиться на 5 (иначе $f(x)$ будут только 0 и 5). Непосредственной проверкой можно убедиться, что при любом a и при всех b, удовлетворяющим приведенным условиям, гарантируется однозначность расшифрования.

2.4. Обозначим через $S(n)$ остаток от деления на 26 суммы чисел, которые соответствуют первым n буквам алфавита ($n = 1, 2, \ldots, 26$) $0 \leqslant S(n) \leqslant 25$.

Если среди чисел $S(1), S(2), \ldots, S(26)$ есть нуль: $S(t) = 0$, то искомой ключевой комбинацией является цепочка первых t букв алфавита.

Если среди чисел $S(1), S(2), \ldots, S(26)$ нет нуля, то обязательно найдутся два одинаковых числа: $S(k) = S(m)$ (считаем, что $k < m$). Тогда искомой ключевой комбинацией является участок алфавита, начинающийся с $(k+1)$-й и заканчивающийся m-й буквой.

2.5. Если две буквы с порядковыми номерами T_1 и T_2 зашифрованы в буквы с порядковыми номерами C_1 и C_2 с помощью одной и той же буквы, то остатки от деления чисел $(C_1 - T_1)$ и $(C_2 - T_2)$ на 30 равны между собой и совпадают с порядковым номером шифрующей буквы (порядковым номером буквы *Я* удобно считать число 0). Тогда, с учетом соглашения о порядковом номере буквы *Я*, справедливо, что T_1 равен остатку от деления числа $(T_2 + (C_1 - C_2))$ на 30, а, вместе с тем, T_2 равен остатку от деления числа $(T_1 + (C_2 - C_1))$ на 30. Если каждое из выражений в скобках заменить соответствующим остатком от деления на 30, то упомянутая связь не нарушится.

Представим в виде набора порядковых номеров известные шифрованные сообщения (обозначим их соответственно ш. с. 1 и ш. с. 2) и слово КОРАБЛИ:

слово	К	О	Р	А	Б	Л	И
T	10	14	16	1	2	11	9

ш.с.1	Ю	П	Т	Ц	А	Р	Г	Ш	А	Л	Ж	Ж	Е	В	Ц	Щ	Ы	Р	В	У	У
C_1	29	15	18	22	1	16	4	24	1	11	7	7	6	3	22	25	27	16	3	19	19

ш.с.2	Ю	П	Я	Т	Б	Н	Щ	М	С	Д	Т	Л	Ж	Г	П	С	Г	Х	С	Ц	Ц
C_2	29	15	0	18	2	13	25	12	17	5	18	11	7	4	15	17	4	21	17	22	22

Возможны 15 вариантов (номер варианта обозначим буквой k) расположения слова КОРАБЛИ в каждом из двух исходных сообщений (и. с. 1, и. с. 2).

Вначале для каждого из 15 вариантов расположения слова КОРАБЛИ в и. с. 1 найдем соответствующий участок и. с. 2. Имеем:

$C_2 - C_1$	0	0	12	26	1	27	21	18	16	24	11	4	1	1	23	22	7	5	14	3	3

T_1	10	14	16	1	2	11	9
T_2	T_{21}	T_{22}	T_{23}	T_{24}	T_{25}	T_{26}	T_{27}

Поэтому для участка и. с. 2 получаем следующие 15 вариантов:

k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
T_{21}	10	10	22	6	11	7	1	28	26	4	21	14	11	11	3
T_{22}	14	26	10	15	11	5	2	0	8	25	18	15	15	7	6
T_{23}	28	12	17	13	7	4	2	10	27	20	17	17	9	8	23
T_{24}	27	2	28	22	19	17	25	12	5	2	2	24	23	8	6
T_{25}	3	29	23	20	18	26	13	6	3	3	25	24	9	7	16
T_{26}	28	2	29	27	5	22	15	12	12	4	3	18	16	25	14
T_{27}	0	27	25	3	20	13	10	10	2	1	16	14	23	12	12

Теперь для каждого из 15 вариантов расположения слова КОРАБЛИ в и. с. 2 найдем соответствующий участок и. с. 1. Имеем:

$C_1 - C_2$	0	0	18	4	29	3	9	12	14	6	19	26	29	29	7	8	23	25	16	27	27

T_2	10	14	16	1	2	11	9
T_1	T_{11}	T_{12}	T_{13}	T_{14}	T_{15}	T_{16}	T_{17}

Поэтому для участка и. с. 1 получаем следующие 15 вариантов:

k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
T_{11}	10	10	28	14	9	13	19	22	24	16	29	6	9	9	17
T_{12}	14	2	18	13	17	23	26	28	20	3	10	13	13	21	22
T_{13}	4	20	15	19	25	28	0	22	5	12	15	15	23	24	9
T_{14}	5	0	4	10	13	15	7	20	27	0	0	8	9	24	26
T_{15}	1	5	11	14	16	8	21	28	1	1	9	10	25	27	18
T_{16}	14	20	23	25	17	0	7	10	10	18	19	4	6	27	8
T_{17}	18	21	23	15	28	5	8	8	16	17	2	4	25	6	6

Заменим порядковые номера в найденных вариантах участков и. с. 1 и и. с. 2 на буквы русского алфавита. Получаем следующие таблицы:

k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
участок и.с.2	К	К	Ц	Е	Л	Ж	А	Э	Ь	Г	Х	О	Л	Л	В
	О	Ь	К	П	Л	Д	Б	Я	З	Щ	Т	П	П	Ж	Е
	Э	М	С	Н	Ж	Г	Б	К	Ы	Ф	С	С	И	З	Ч
	Ы	Б	Э	Ц	У	С	Щ	М	Д	Б	Б	Ш	Ч	З	Е
	В	Ю	Ч	Ф	Т	Ь	Н	Е	В	В	Щ	Ш	И	Ж	Р
	Э	Б	Ю	Ы	Д	Ц	П	М	М	Г	В	Т	Р	Щ	О
	Я	Ы	Щ	В	Ф	Н	К	К	Б	А	Р	О	Ч	М	М

k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	К	К	Э	О	И	Н	У	Ц	Ш	Р	Ю	Е	И	И	С
	О	Б	Т	Н	С	Ч	Ь	Э	Ф	В	К	Н	Н	Х	Ц
участок	Г	Ф	П	У	Щ	Э	Я	Ц	Д	М	П	П	Ч	Ш	И
и.с.1	Д	Я	Г	К	Н	П	Ж	Ф	Ы	Я	Я	З	И	Ш	Ь
	А	Д	Л	О	Р	З	Х	Э	А	А	И	К	Щ	Ы	Т
	О	Ф	Ч	Щ	С	Я	Ж	К	К	Т	У	Г	Е	Ы	З
	Т	Х	Ч	П	Э	Д	З	З	Р	С	Б	Г	Щ	Е	Е

Из таблиц видно, что осмысленными являются варианты:

и.с.1 = К О Г Д А О Т . . . . . К О Р А Б Л И
и.с.2 = К О Р А Б Л И . . . . . В Е Ч Е Р О М

Естественно предположить, что в первом исходном сообщении речь идет об отплытии кораблей. Предположив, что неизвестным участком первого исходного сообщения является подходящая по смыслу часть слова ОТПЛЫВАЮТ, находим неизвестную часть второго исходного сообщения: слово ОТХОДЯТ.

2.6. Каждую букву шифрованного сообщения расшифруем в трех вариантах, предполагая последовательно, что соответствующая буква шифрующей последовательности есть буква А, Б или буква В:

шифрованное сообщение	Р	Б	Ь	Н	П	Т	С	И	Т	С	Р	Р	Е	З	О	Х
вариант А	П	А	Щ	М	О	С	Р	З	С	Р	П	П	Д	Ж	Н	Ф
вариант Б	О	Я	Ш	Л	Н	Р	П	Ж	Р	П	О	О	Г	Е	М	У
вариант В	Н	Ю	Ч	К	М	П	О	Е	П	О	Н	Н	В	Д	Л	Т

Выбирая из каждой колонки полученной таблицы ровно по одной букве, находим осмысленное сообщение НАШКОРРЕСПОНДЕНТ, которое и является искомым.

Замечание. Из полученной таблицы можно было найти такое исходное сообщение как

НАШ МОРОЗ ПОПОВ ЕМУ

которое представляется не менее осмысленным, чем приведенное выше. А если предположить одно искажение в шифрованном сообщении (скажем, в качестве 11-й буквы была бы принята не буква Р, а буква П), то, наряду с правильным вариантом, можно получить и такой:

НАШ МОРОЗ ПОМОГ ЕМУ

Число всех различных вариантов исходных сообщений без ограничений на осмысленность равно 3^{16} или 43046721, т. е. более 40 миллионов!

3.1. Если каждый из 993 абонентов связан с 99 абонентами, то для этого потребуется $993 \cdot 99/2$ линий связи, которое не может быть целым числом.

Ответ: нельзя.

3.2. Несложно заметить, что рассматриваемый шифр обладает тем свойством, что при зашифровании разные буквы заменяются разными. Следовательно, при зашифровании разных слов получаются разные слова. С другой стороны, одинаковые буквы заменяются на одинаковые независимо от цикла шифрования, так как используется один и тот же ключ. Следовательно, при зашифровании одинаковых слов получаются одинаковые слова. Таким образом, число различных слов, которые можно получить в указанном процессе шифрования с начальным словом СРОЧНО, совпадает с наименьшим номером цикла шифрования, дающем это начальное слово.

Так как буква С повторяется в каждом цикле шифрования, номер которого кратен 5, а буквы Р, О, Ч, Н — в каждом цикле, номера которых кратны 13, 7, 2 и 3 соответственно, то слово СРОЧНО появится впервые в цикле с номером, равным $НОК(2,3,5,7,13) = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 13 = 2730$.

Ответ: 2730.

3.3. Если символы одного отрезка занумеровать последовательно числами от 1 до 12, то после передачи его из А в Б символы расположатся в порядке (2,4,6,8,10,12,1,3,5,7,9,11), а после передачи этого отрезка (замена символов не меняет порядка) из Б в В — в порядке (4,8,12,3,7,11,2,6,10,1,5,9). Переставим символы перехваченных отрезков в соответствии с их номерами до передачи из пункта А. Получим отрезки вида:

Л	П	Г	С	Ж	Н	Ж	О	О	Б	Т	–
Е	С	К	Р	У	П	Д	С	Б	Х	К	Т
Ю	У	П	–	О	Б	Ф	Е	Б	–	П	–
Л	Ж	Е	С	Ж	У	О	П	Л	У	К	–
Щ	К	Х	С	П	Г	Б	М	Ы	О	Э	Ц
Л	К	Л	У	Ж	Н	–	Л	Г	Т	И	К

Поскольку в пунктах А и Б одинаковые буквы заменялись одинаковыми, а разные — разными, то найденные отрезки можно рассматривать как замену одинаковых символов исходного текста одинаковыми, а разных — разными. Сравнивая места одинаковых букв слова

КРИПТОГРАФИЯ и места одинаковых символов в отрезках, находим, что слово КРИПТОГРАФИЯ зашифровано во втором отрезке. Это дает возможность найти исходное сообщение, используя гипотезы о частых буквах русского языка и смысле исходного сообщения.

Ответ:

С	О	В	Р	Е	М	Е	Н	Н	А	Я	–
К	Р	И	П	Т	О	Г	Р	А	Ф	И	Я
Э	Т	О	–	Н	А	У	К	А	–	О	–
С	Е	К	Р	Е	Т	Н	О	С	Т	И	–
Ш	И	Ф	Р	О	В	А	Л	Ь	Н	Ы	Х
С	И	С	Т	Е	М	–	С	В	Я	З	И

3.4. Докажем, что 20 является периодом рассматриваемой последовательности. Заметим, что у двух натуральных чисел a и b совпадают цифры единиц тогда и только тогда, когда их разность делится на 10. Таким образом, мы достигнем цели, если докажем, что разность $(n+20)^{n+20} - n^n$ делится на 10 для всех натуральных значений n. Исходя из того, что $p^k - q^k$ делится на $(p-q)$, получаем, что $(n+20)^{n+20} - n^{n+20}$ делится на $((n+20)-20) = 20$. Кроме того, $n^{n+20} - n^n = n^n(n^{20}-1) = n^n((n^4)^5 - 1)$ делится на $n(n^4-1)$ для всех $n > 1$. Вместе с тем,

$$n(n^4-1) = n(n-1)(n+1)(n^2+1) = n(n-1)((n+2)(n-2)+5) =$$
$$= (n-2)(n-1)n(n+1)(n+2) + 5(n-1)n(n+1),$$

где каждое из слагаемых делится на 2 (так как содержит произведение $n(n+1)$) и делится на 5 (поскольку первое слагаемое есть произведение пяти последовательных чисел, а второе содержит множитель 5). Следовательно, $n^{n+20} - n^n$ делится на 10. Число

$$(n+20)^{n+20} - n^n = \left((n+20)^{n+20} - n^{n+20}\right) + (n^{n+20} - n^n)$$

делится на 10, так как каждое из слагаемых делится на 10.

Проверим, что 20 является наименьшим периодом. Выписывая первые 20 значений последовательности $C_1, C_2, \ldots$

1 4 7 6 5 3 6 9 0 1 6 3 6 5 6 7 4 9 0

легко убедиться, что она не имеет периода меньшей длины.

3.5. Для того, чтобы найти исходное сообщение, найдем сначала цифровое сообщение, полученное из него с помощью таблицы замены. Согласно этой таблице на нечетных местах цифрового образа исходного сообщения могут быть только цифры 0, 1, 2 и 3. Последовательно рассматривая эти значения для каждого нечетного места цифрового сообщения с использованием соответствующей цифры шифрованного сообщения, найдем соответствующие варианты значений цифр шифрующего отрезка. Для этого вычислим остатки от деления разностей цифр шифрованного и варианта цифрового сообщений:

порядковый номер места k	1	3	5	7	9	11	13	15	17	19	21	23	25	27
шифрованное сообщение S_k	2	3	8	7	1	4	8	6	6	0	1	3	5	8
вариант 0 для Γ_k	2	3	8	7	1	4	8	6	6	0	1	3	5	8
вариант 1 для Γ_k	1	2	7	6	0	3	7	5	5	9	0	2	4	7
вариант 2 для Γ_k	0	1	6	5	9	2	6	4	4	8	9	1	3	6
вариант 3 для Γ_k	9	0	5	4	8	1	5	3	3	7	8	0	2	5

По задаче 3.4 последовательность, из которой выбран шифрующий отрезок, является периодической с периодом 20. Из таблицы вариантов значений цифр шифрующего отрезка видим, что 5-я его цифра может быть равна 5, 6, 7 или 8, а его 25-я цифра — 2, 3, 4 или 5. Отсюда получаем, что $\Gamma_5 = \Gamma_{25} = 5$. На периоде последовательности, из которой выбран шифрующий отрезок, есть две цифры 5: C_5 и C_{15}. Поэтому рассмотрим два случая. Если $\Gamma_5 = C_5$, то $\Gamma_7 = C_7 = 3$. Это противоречит таблице вариантов значений цифр шифрующего отрезка, в которой Γ_7 может быть равна 4, 5, 6 или 7. Если же $\Gamma_5 = C_{15}$, то соответствующий шифрующий отрезок: 16365674901476563690163656 74 хорошо согласуется с таблицей вариантов значений его цифр. Вычитая цифры найденного отрезка из соответствующих цифр шифрованного сообщения и заменяя разности их остатками от деления на 10, получим по таблице замены пар цифр на буквы исходное сообщение:

шифрованное сообщение	23	39	86	72	16	45	81	60	67	06	17	31	55	88
шифрующий отрезок	16	36	56	74	90	14	76	56	36	90	16	36	56	74
цифровое сообщение	17	03	30	08	26	31	15	14	31	16	01	05	09	14
исходное сообщение	С	В	Я	З	Ь	–	П	О	–	Р	А	Д	И	О

3.6. Обозначения понятны из рис. 9.

1) MK_1P_1B центрально симметричен $MKPA$ относительно M.

2) NL_1Q_1B центрально симметричен $NLQC$ относительно N.

3) $P_1K_2Q_1 = PKQ$ (параллельный перенос).

4) $LK_1K_2L_1$ — квадрат.

5) $MT \perp AC$, $NS \perp AC$.

6) $PMT = QNS$ ($MT = NS$, $PM = QN$, $\angle T = \angle S = 90°$).

7) Без ограничения общности $AB = BC = CA = 1$.

8) $PT = QS = x$, $AP = \dfrac{1}{4} \mp x$, $PQ = \dfrac{1}{2}$, $QC = \dfrac{1}{4} \pm x$.

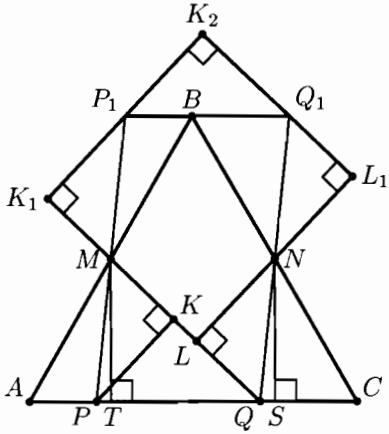


Рис. 9.

9) $PMK = NQL$ ($PM{=}QN$, $\angle M{=}\angle Q$, $\angle K{=}\angle L{=}90^\circ$) $\Rightarrow MK = QL$.

10) $MQ = ML + LQ = ML + MK = ML + K_1M = K_1L = y$.

11) Площадь $ABC = \dfrac{\sqrt{3}}{4}$ равна площади $LK_1K_2L_1 = y^2$, $y = \dfrac{\sqrt[4]{3}}{2}$.

12) $MT = \dfrac{\sqrt{3}}{4}$ (половина высоты ABC).

13) $QT = PQ - PT = \dfrac{1}{2} \mp x$.

14) $MQ^2 = MT^2 + QT^2$ (теорема Пифагора), т. е.

$$\left(\frac{\sqrt[4]{3}}{2}\right)^2 = \left(\frac{\sqrt{3}}{4}\right)^2 + (\frac{1}{2} \mp x)^2 \overset{(x<1/2)}{\Longleftrightarrow} \sqrt{\frac{\sqrt{3}}{4} - \frac{3}{16}} = \left|\frac{1}{2} - x\right| \Longleftrightarrow$$

$$\Longleftrightarrow x = \frac{1}{2} - \frac{1}{4}\sqrt{4\sqrt{3} - 3}.$$

15) $AP : PQ : QC = \dfrac{1}{4}\left(\sqrt{4\sqrt{3}-3} - 1\right) : \dfrac{1}{2} : \dfrac{1}{4}\left(3 - (\sqrt{4\sqrt{3}-3})\right) =$

$$= \left(\sqrt{4\sqrt{3}-3} - 1\right) : 2 : \left(3 - \sqrt{4\sqrt{3}-3}\right).$$

Замечание: Точки P и Q можно построить с помощью циркуля и линейки. Подумайте, как это можно сделать.

Ответ: $AP : PQ : QC = \left(\sqrt{4\sqrt{3}-3} - 1\right) : 2 : \left(3 - \sqrt{4\sqrt{3}-3}\right)$.

4.1. Исходный текст состоит из 48 букв, следовательно, при зашифровании было использовано три положения решетки полностью и еще три буквы вписаны в четвертом положении. Значит, незаполненные 12 клеток совпадают с вырезами решетки в четвертом положении. Так как

текст вписывается последовательно, то неизвестные нам три выреза могут располагаться только в первой строке таблицы и первых пяти клетках второй строки (до первого известного выреза). Считаем, что трафарет лежит в четвертом положении. Учитывая, что в одну клетку листа нельзя вписать две буквы, получаем, что вырезы могут быть только в отмеченных знаком «?» местах трафарета («*» — места известных вырезов):

	?							?	
		?	?		*				*
*				*				*	
		*				*			
								*	
*		*	*				*		

Очевидно, что из отмеченных в первой строке двух клеток вырезается только одна (так как они совмещаются поворотом). Получаем два возможных варианта решетки (либо первый «?», либо второй «?» в первой строке). Читаемый текст получается при втором варианте.

Ответ:
ПОЛЬЗУЯСЬШИФРОМРЕШЕТКАНЕЛЬЗЯОСТАВЛЯТЬПУСТЫЕМЕСТА

4.2. Один из вариантов решения состоит из следующих этапов.

1. 19=н из второй строки («19,2 19,5»).
2. 29=о из третьей строки («29,н,10») и 10=а или 10=и.
3. 14=щ из «но,14,но».
4. 8=д, 2=е, 10=и из «денно и нощно».

Получили текст:

12е24 5 3 21 6о 28е20 18 20 21 5и 27 17е 11е 16 —
не 27 5 до 12 31 22е 16, не н 5 17о до 6о 16:
денно и нощно они е24е 11е 16
ищ 18 21 17е 20е 28о 16 21о 28 6о 16.

5. 5=а и 27=з из второй строки.
6. 17=в 6=п 16=й — последнее слово второй строки — водопой.

Получили текст:

12е24а 3 21 по 28е20 18 20 21аи зв е 11ей —
не за до 12 31 22ей, не на водопой:
денно и нощно они е24е 11ей
ищ 18 21 ве 20е 28ой 21о 28 пой.

7. 21=т 18=у 28=л 20=с из последней строки «ищут веселой толпой».

8. 11=р из «з в е 11 е й» первой строки.

Итак,

12е24а3т по лесу стаи зверей —
не за до123122ей, не на водопой:
денно и нощно они е24ерей
ищут веселой толпой.

9. 24=г из «егерей».

10. 12=б 3=ю из «бегают».

11. 31=ы 22=ч из «добычей».

Ответ: Бегают по лесу стаи зверей —
Не за добычей, не на водопой:
Денно и нощно они егерей
Ищут веселой толпой.

4.3. *Ответ*: $\cos 36^\circ = (1+\sqrt{5})/4 \approx 0{,}8$.

4.4. Занумеруем буквы латинского алфавита последовательно числами от 1 до 24. Пусть x — некоторое число от 1 до 24, а $f(x)$ — число, в которое переходит x на втором этапе. Тогда перестановочность этапов можно записать в следующем виде:

$$f(x+1) = f(x) + 1, \quad \text{т. е.} \quad f(x+1) - f(x) = 1.$$

Это означает, что соседние числа x и $x+1$ на втором этапе переходят в соседние же числа $f(x)$ и $f(x+1)$, т. е. второй этап — тоже сдвиг. Последовательное применение двух сдвигов — очевидно тоже сдвиг и остается рассмотреть 24 варианта различных сдвигов. Читаемый текст определяется однозначно. Осложнения, связанные с переходом Z в A, устраняются либо переходом к остаткам при делении на 24, либо выписыванием после буквы Z второй раз алфавита AB ... Z.

Ответ: INTER ARMA SILENT MUSAE
(´интер ´арма си́лент му́зэ —
когда гремит оружие, музы молчат).

4.5. Составим возможные варианты переданных букв:

ГЪЙ	АЭЕ	БПРК	ЕЖЩЮ	НМЬЧ	СЫЗЛ	ШДУ	ЦХОТ	ЯФВИ
БШЗ	АЫВ	АНОИ	ГЕЧЬ	ЛКЪХ	ПЩЕЙ	ЦВС	ФУМР	ЭТАЖ
ВЩИ	БЬЕ	БОПЙ	ДЁШЭ	МЛЫЦ	РЪЖК	ЧГТ	ХФНС	ЮУБЗ
ГЪЙ	ВЭЁ	ВПРК	ЕЖЩЮ	НМЬЧ	СЬЗЛ	ШДУ	ЦХОТ	ЯФВИ
ДЫК	ЮЖ	ГРСЛ	ЁЗЪЯ	ОНЭШ	ТЬИМ	ЩЕФ	ЧЦПУ	ХГЙ
ЕЬЛ	ЯЗ	СТМ	ЖИЫ	ПОЮЩ	УЭЙН	ЪЁХ	ШЧРФ	ЦДК

Выбирая вторую и последнюю группу букв (где есть короткие колонки букв), определяем слова, им соответствующие: ВЯЗ, ЭТАЖ. В исходных словах 33 буквы, поэтому буквы В, Я, З, Э, Т, А, Ж уже использованы и их можно вычеркнуть из всех колонок:

ГЪЙ	АЭЕ	БПРК	ЕЖЩЮ	НМЬЧ	СЫЗЛ	ШДУ	ЦХОТ	ЯФВИ
БШ		НОИ	ГЕЧЬ	ЛКЪХ	ПЩЕЙ	Ц С	ФУМР	ЭТАЖ
ЩИ		БОПЙ	ДЁШ	МЛЫЦ	РЪ К	ЧГ	ХФНС	
ГЪЙ	В	ПРК	Е Щ	НМЬЧ	СЬ Л	ШДУ	ЦХО	
		ГРСЛ	Ё Ъ	ОН	ЬИМ	ЩЕФ	ЧЦПУ	
ЕЬЛ	ЯЗ	С М	ИЫ	ПО	У ЙН	ЪЁХ	ШЧРФ	

Из нескольких вариантов, например, в третьей группе:

ГНОЙ ГНОМ ГРОМ

выбираем варианты так, чтобы каждая буква использовалась один раз. Продолжая таким образом, получим ответ.

Ответ:

БЫК ВЯЗ ГНОЙ ДИЧЬ ПЛЮЩ СЪЁМ ЦЕХ ШУРФ ЭТАЖ

4.6. Заметим, что $A_{k+1} - A_k = (k+1)^3 - k^3 + 2$ для всех натуральных k. Складывая почленно эти равенства при $k = 1, 2, \ldots, (n-1)$, получим $A_n - A_1 = n^3 - 3 + 2n$. По условию $A_1 = 3$. Следовательно, справедливо соотношение $A_n = n^3 + 2n$.

Ясно, что при расшифровании так же, как и при зашифровании, вместо чисел A_{100}, A_{101}, A_{102}, A_{103}, A_{104}, A_{105}, A_{106} можно воспользоваться их остатками от деления на 30. Так как для каждого целого неотрицательного i

$$(100+i)^3 + 2(100+i) = i^3 + 2i + 30z,$$

где z — некоторое целое число, то получаем следующие остатки при делении чисел $A_{100}, \ldots, A_{106}$ на 30:

A_{100}	A_{101}	A_{102}	A_{103}	A_{104}	A_{105}	A_{106}
0	3	12	3	12	15	18

Заключительный этап представлен в таблице:

шифрованное сообщение	К	Е	Н	З	Э	Р	Е
числовое шифрованное сообщение	9	5	12	7	27	15	5
шифрующий отрезок	0	3	12	3	12	15	18
числовое исходное сообщение	9	2	0	4	15	0	17
исходное сообщение	К	В	А	Д	Р	А	Т

4.7. *Ответ*:

$$x = \frac{1+\sqrt{4a^2+1}}{2a} \text{ при } 0 < a < 1;$$

$$x_1 = \frac{1+\sqrt{4a^2+1}}{2a},\ x_2 = \frac{-\sqrt{4a^2-3}-1}{2a} \text{ при } a \geqslant 1.$$

5.1. *Указание.* Найдите допустимые варианты для остатков от деления неизвестных x и y на 7. Таких вариантов будет восемь. Учитывая принадлежность неизвестных к заданному диапазону, найдите допустимые варианты для (x, y) (19 вариантов). Для каждой пары (x, y) найдите z. В диапазон 10 . . 20 попадают только три решения: (12,16,11), (13,17,17), (13,18,12).

5.2. Так как при записывании сообщения в таблицу пробелы опускались, можно сделать вывод, что столбцы, содержащие пробел в последней клетке, до перестановки стояли в конце таблицы. Таким образом, столбцы можно разбить на две группы, как показано на рис. 10. При этом для получения исходного текста потребуется переставлять столбцы только внутри групп.

Я	Н	Л	В	Р	А	Л	О	Е	Г	О	М	З	Е
Й	Л	Т	А	Ф	Ы	И	П	И	О	Г	Е	Б	Р
Ч	Р	Д	Ч	Е	С	М	О	К	И	Н	Т	К	О
Н	У	Л	А	Р	Е	Б	Ы	Е	И	О	Н	Ы	Д
Ы	Т	Д	О	М	П	П	Т	А	И	П	Т	З	Л
И	К	С	И	Т	Ч	Н	О	Е	Л	У	Л	Т	Ж

К	Е	Т	Р	И	Я
Л	Е	У	О	Д	О
И	Н	Д	Х	И	Е
Е	Ы	Е	З	Н	Ч
Е	Щ	В	Н	Я	С
–	–	–	–	–	–

Рис. 10.

Естественно предположить, что сообщение оканчивалось точкой. Поэтому на третьем с конца месте в первой группе должен быть столбец, оканчивающийся на Т, на втором — на Ч, на последнем — на К. Получаем два варианта (рис. 10), из которых первый является явно «нечитаемым».

Р	А	Н
Ф	Ы	Л
Е	С	Р
Р	Е	У
М	П	Т
Т	Ч	К

З	А	Н
Б	Ы	Л
К	С	Р
Ы	Е	У
З	П	Т
Т	Ч	К

Я	Л	В	Р	Л	О	Е	Г	О	М	Е	
Й	Т	А	Ф	И	П	И	О	Г	Е	Р	
Ч	Д	Ч	Е	М	О	К	И	Н	Т	О	
Н	Л	А	Р	Б	Ы	Е	И	О	Н	Д	
Ы	Д	О	М	П	Т	А	И	П	Т	Л	
И	С	И	Т	Н	О	Е	Л	У	Л	Ж	

Рис. 11.

З	А	Н	Я	Т	И	Е	К	Р
Б	Ы	Л	О	У	Д	Е	Л	О
К	С	Р	Е	Д	И	Н	И	Х
Ы	Е	У	Ч	Е	Н	Ы	Е	З
З	П	Т	С	В	Я	Щ	Е	Н
Т	Ч	К	-	-	-	-	-	-

Рис. 12.

Таким образом, удалось зафиксировать последние три столбца первой

группы. Переставляя столбцы второй группы, ищем «читаемые» продолжения зафиксированных столбцов (рис. 11). Действуя далее аналогичным образом с оставшимися столбцами первой группы, достаточно легко получаем исходное сообщение.

Ответ:

Д	О	Л	Г	О	Е	В	Р	Е	М	Я	З	А	Н	Я	Т	И	Е	К	Р
И	П	Т	О	Г	Р	А	Ф	И	Е	Й	Б	Ы	Л	О	У	Д	Е	Л	О
М	О	Д	И	Н	О	Ч	Е	К	Т	Ч	К	С	Р	Е	Д	И	Н	И	Х
Б	Ы	Л	И	О	Д	А	Р	Е	Н	Н	Ы	Е	У	Ч	Е	Н	Ы	Е	З
П	Т	Д	И	П	Л	О	М	А	Т	Ы	З	П	Т	С	В	Я	Щ	Е	Н
Н	О	С	Л	У	Ж	И	Т	Е	Л	И	Т	Ч	К						

5.4. Во втором случае известны пары цифр, которыми шифруются буквы «р», «е», «м», «о», «н», «т», а в первом — пары цифр для тех же букв, за исключением буквы «н».

Ответ: во втором случае легче.

5.5. *Ответ*: 481.

5.6. Можно заметить, что последовательность букв МОСКВА входит как подпоследовательность в каждый из шифртекстов первой тройки:

й МывОт СьлКъгВц Аяя
укМапОч Ср Кщ Вз Ах
ш МфэОгчСйъКфьВыеАкк

На основе этого наблюдения можно предположить, что шифрование заключается в следующем. В каждый промежуток между буквами исходного сообщения (начало и конец также считаются промежутками) вставляются одна либо две буквы в соответствии с известным только отправителю и получателю ключом.

Очевидно, что первая буква сообщения должна попасть на 2-е или 3-е место шифрованного текста. Сравнивая буквы, стоящие на указанных местах в подлежащих расшифрованию криптограммах, делаем вывод, что одно и то же исходное сообщение соответствует первому и третьему шифртексту и что первая буква этого сообщения — П.

Рассуждая далее аналогичным образом, заключаем, что второй буквой повторяющегося сообщения является О (сопоставили ОИ из 1-й криптограммы и ИО из 3-й) и так далее. В итоге получим, что первой и третьей криптограмме соответствует исходное сообщение

ПОВТОРЕНИЕМАТЬУЧЕНИЯ

Теперь расшифруем вторую криптограмму. Первой буквой сообщения могут быть только С или И. Далее, подбирая к каждой из них возможные варианты последующих букв и вычеркивая заведомо «нечитаемые» цепочки букв, получим:

СЕ, СМ, ИМ, ИГ

СЕГ, ~~сео~~, СМО, СМР, ИМО, ИМР, ИГР, ~~игт~~

~~сегр, сегт~~, СМОТ, СМОК, ~~смрк, смрр~~, ИМОТ, ИМОК,
~~имрк, имрр, игрк, игрр~~
СМОТР, СМОТО, СМОКО, ~~смокм~~, ИМОТР, ИМОТО, ИМОКО, ~~имокм~~
СМОТРМ, СМОТРИ,
~~смотои, смотот, смокои, смокот, имотрм, имотри~~,
~~имотои, имотот, имокои, имокот~~
СМОТРИВ, СМОТРИА
СМОТРИВВ, СМОТРИВК, СМОТРИАК, СМОТРИАН и так далее.

В итоге получим исходное сообщение СМОТРИВКОРЕНЬ.

Ответ: 1,3 — ПОВТОРЕНИЕМАТЬУЧЕНИЯ
2 — СМОТРИВКОРЕНЬ

5.7. Обратив внимание на то, что некоторые символы в тексте условий задач пятой олимпиады набраны выделенным шрифтом, и выписав эти символы в порядке их следования, получаем текст:

задачасемьпояснитекаквынаилитекстзадачи

6.1. Так как каждый из 1997 абонентов связан ровно с N другими, то общее число направлений связи равно $1997N$. Отсюда общее число связанных пар абонентов равно $1997N/2$, так как каждая связанная пара имеет ровно 2 направления связи. Поскольку число $1997N/2$ должно быть целым, а число 1997 — нечетное, то число N должно быть четным.

Докажем, что для каждого $N = 2T$ существует система связи из 1997 абонентов, в которой каждый связан ровно с N другими. В самом деле, расположив всех абонентов на окружности и связав каждого из них с T ближайшими к нему по часовой стрелке и с ближайшими к нему против часовой стрелки, получим пример такой сети связи.

6.2. Покажем, что на диагонали присутствуют все числа от 1 до 1997. Пусть число $a \in \{1, \ldots, 1997\}$ не стоит на диагонали. Тогда, в силу симметрии таблицы, число a встречается четное количество раз. С другой стороны, так как число a по одному разу встречается в каждой строке, всего в таблице чисел a нечетное количество (1997). Получили противоречие.

Всего на диагонали 1997 клеток, поэтому каждое число из множества $\{1, \ldots, 1997\}$ встретится на диагонали ровно по одному разу. Вычисляя сумму арифметической прогрессии, находим ответ.

Ответ: 1995003.

6.3. *Ответ*: ШЕСТАЯОЛИМПИАДАПОКРИПТОГРАФИИПОСВЯЩЕНАСЕМИДЕСЯТИПЯТИЛЕТИЮСПЕЦИАЛЬНОЙСЛУЖБЫРОССИИ

Указание. Пусть некоторая буква α при зашифровании первым способом заменялась на букву β. Тогда количество повторов буквы β в первой криптограмме будет равно числу повторов буквы α во второй криптограмме.

6.4. а) Определим моменты остановок после начала шифрования. Для этого каждой букве русского алфавита припишем ее порядковый номер: А — 0, Б — 1, и т. д. Тогда буквам из шифруемого слова будут соответствовать номера: О — 15, Л — 12, И — 9, М — 13, П —16, А — 0, Д — 4. Моменты остановок будем указывать числом одношаговых (на один зубец) поворотов *I* колеса до соответствующей остановки.

№остановки	1	2	3	4	5	6	7	8	9
Буква *I* колеса	О	Л	И	М	П	И	А	Д	А
Число одношаговых поворотов от начала до остановки	15	45	75	79	82	108	132	136	165
Цифра *II* колеса	5	5	5	1	8	2	8	4	5
Цифра *III* колеса	1	2	5	2	5	3	6	3	4

Искомый шифртекст: 515355128523864354

б) Пусть t_k — количество одношаговых поворотов *I* колеса от начала до остановки с номером k, $k = 1, 2, \ldots$,

a_k — цифра, на которую указывает стрелка *II* колеса в момент остановки с номером k,

b_k — цифра *III* колеса, на которую указывает стрелка *III* колеса в момент остановки с номером k.

Тогда, учитывая, что начальное положение стрелок соответствует букве А на первом колесе и 0 на *II* и *III* колесах, справедливы равенства

$$t_k = 10m_k - a_k, \quad k = 1, 2, \ldots \tag{1}$$

$$t_k = 7n_k + b_k, \quad k = 1, 2, \ldots \tag{2}$$

для подходящих неотрицательных целых чисел m_k и n_k.

Заметим, что $1 = 7 \cdot 3 - 10 \cdot 2$. Отсюда справедливы равенства

$$a_k = 7 \cdot (3a_k) - 10 \cdot (2a_k), k = 1, 2, \ldots$$

$$b_k = 7 \cdot (3b_k) - 10 \cdot (2b_k), k = 1, 2, \ldots$$

Подставляя эти значения в равенства (1) и (2), получим

$$t_k = 10(m_k + 2a_k) - 7(3a_k), k = 1, 2, \ldots$$

$$t_k = 7(n_k + 3b_k) - 10(2b_k), k = 1, 2, \ldots$$

Следовательно,

$$10(m_k + 2a_k) - 7(3a_k) = 7(n_k + 3b_k) - 10(2b_k), k = 1, 2, \ldots$$

Правая и левая части делятся на 70, то есть имеют вид $70s_k$ для подходящего неотрицательного целого s_k. Поэтому

$$m_k = 7s_k - 2(a_k + b_k), k = 1, 2, \ldots$$

$$n_k = 10s_k - 3(a_k + b_k), k = 1, 2, \ldots$$

Подставляя m_k в (1), получим

$$t_k = 70s_k - 21a_k - 20b_k, k = 1, 2, \ldots.$$

Учитывая условие $0 < t_1 < t_2 < \cdots < t_7$ и то, что остановка колес происходит в момент первого появления шифруемой буквы под стрелкой I колеса, имеем

k	1	2	3	4	5	6	7
a_k	2	8	9	8	9	1	1
b_k	4	0	2	3	1	2	1
$-(21a_k + 20b_k)$	-122	-168	-229	-228	-209	-61	-41
t_k	18	42	51	52	71	79	99
Буквы	С	И	С	Т	Е	М	А

6.5. *Указание.* Рассмотрим некоторую расстановку ненулевых цифр на окружности. Упорядоченную пару (a, b) соседних цифр на этой окружности назовем 1-соседней, если b является соседней с a по часовой стрелке. Пару (a, c) назовем 2-соседней, если существует цифра b, для которой пары (a, b) и (b, c) являются 1-соседними.

Каждой расстановке ненулевых цифр на окружности однозначно соответствует цепочка 1-соседних пар вида: $(1, a_1)$, (a_1, a_2), (a_2, a_3), $\ldots$, (a_7, a_8), $(a_8, 1)$, которой, в свою очередь, однозначно соответствует цепочка 2-соседних пар вида:

$$(1, a_2), (a_2, a_4), (a_4, a_6), (a_6, a_8), (a_8, a_1)(a_1, a_3)(a_3, a_5)(a_5, a_7)(a_7, 1), \quad (*)$$

где $a_2, a_3, \ldots, a_8 \in \{2, \ldots, 9\}$ и $a_i \neq a_j$ при $i \neq j$.

Если из цепочки $(*)$ удалить любую пару, то по оставшимся парам она восстанавливается однозначно.

Если из цепочки $(*)$ удалить две соседние пары, то она также восстанавливается однозначно.

Удаление из $(*)$ любых трех пар приводит к неоднозначности восстановления цепочки $(*)$. В этом можно убедиться, рассмотрев следующие фрагменты цепочки вида $(*)$:

$(a, b)(b, c)(c, d)$ и $(a, c)(c, b)(b, d)$, $(a, b, c, d$ — различные цифры),

$(a, b)\_(c, d)(d, e)$ и $(a, d)(d, b)\_(c, e)$, $(a, b, c, d, e$ — различные цифры),

$(a, b)\_(c, d)\_(e, f)$ и $(a, d)(e, b)\_(c, f)$, $(a, b, c, d, e, f$ — различные цифры).

Таким образом, при наличии двух указанных в условии задачи цифровых текстов нам будут известны некоторые 2-соседние пары, в которых первая цифра берется из первой криптограммы, а вторая — из второй. Поэтому с учетом вышесказанного получаем условие однозначного восстановления порядка расстановки цифр на данной окружности.

Ответ: для однозначного восстановления расстановки цифр на окружности необходимо и достаточно, чтобы в одном из цифровых текстов было не менее 7 ненулевых цифр (это соответствует удалению из цепочки 2-соседних пар вида (*) не более двух из них).

6.6. Последовательность остатков от деления чисел $a_1, a_2, \ldots$ на 24 — периодическая с периодом 2, так как для любого натурального n справедливо:

$$a_{n+2} - a_n = p^{n+4} - p^{n+2} = \begin{cases} 24 \cdot 2^{n-1}, & \text{при } p = 2 \\ p^{n+1}(p^3 - p), & \text{при } p \geqslant 3 \end{cases}.$$

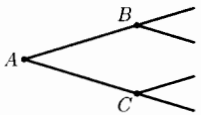
Кроме того, $p^3 - p = (p-1)p(p+1)$ кратно 24, то есть остатки у a_{n+2} и a_n равны.

6.7. *Ответ*: $a = 1996$; все решения имеют вид $\pm 3992k + 1996$, $k = 0, 1, \ldots, 998$.

Указание. При $a \leqslant 0$ рассматриваемое уравнение равносильно $|x - a| - 1995a = 1996$, которое имеет не более двух решений.

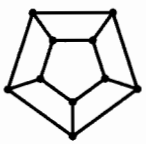
При $a > 0$ из графика функции в левой части уравнения видно, что если $1996 \in (0, a)$, число решений будет четным, поэтому не может быть равно 1997. Если $1996 \in (a, +\infty)$, то уравнение имеет ровно 2 решения. Если же $a = 1996$, то уравнение имеет ровно 1997 решений.

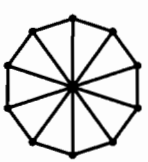
7.1. Для того, чтобы сохранилась связь при выходе из строя любых двух узлов, необходимо, чтобы в каждый узел входило не менее трех линий связи. Ситуация



недопустима, ибо при выходе из строя узлов B и C узел A становится недоступным. Значит, всего линий должно быть не менее $\dfrac{10 \times 3}{2} = 15$.

Вот два примера, удовлетворяющие условиям задачи с 15-ю линиями связи:





Приведем доказательство для первого примера. Если вышли из строя два узла на одном пятиугольнике, то связь сохранится через другие пятиугольники. Если вышли из строя по одному узлу на разных пятиугольниках, то связь сохранится по линиям, соединяющим эти пятиугольники.

Ответ: 15.

7.2. Процедура зашифрования может быть полностью описана квадратной таблицей 10×10. На пересечении строки с номером i и столбца с номером j записываем цифру, в которую при зашифровании переходит цифра j, если она стоит в пароле после цифры i. Из однозначности расшифрования следует, что в каждой строке каждая цифра встречается ровно один раз.

Обозначим через $ш_1, ш_2, \ldots, ш_7$ и $о_1, о_2$ зашифрованные пароли и два известных пароля в порядке, определяемом условием задачи. Процедура зашифрования сохраняет длину, поэтому $ш_3$ и $ш_4$ не могут соответствовать ни $о_1$, ни $о_2$. Предположив, что $ш_1$ соотвествует $о_1$, получим часть таблицы, в которой в одной строке две одинаковые цифры. Это означает, что предположение неверно. Составляя таблицы, убеждаемся, что $о_2$ не шифруется ни в $ш_6$, ни в $ш_7$, ни в $ш_5$. В результате таких рассуждений остается только один вариант перехода $о_1 - ш_2$, $о_2 - ш_5$. Заполнение таблицы будет следующим:

	0	1	2	3	4	5	6	7	8	9
0									5	
1			3							
2	4	3	7					8		
3		7								
4			2							
5									3	
6										
7						4				
8			1	9						
9										

	0	1	2	3	4	5	6	7	8	9
0			6						5	
1			3							
2	4	3	7	0	6	2	5	8	9	
3	3	7								
4			2							
5									3	7
6										
7						4				
8			1	9						
9			1							

Очевидно, что в строке с номером 2 в последней клетке стоит 1. Знание этой таблицы позволяет однозначно расшифровать $ш_3$: получится 5830829. Пароли, соответствующие $ш_1$, $ш_4$, $ш_6$, $ш_7$, восстанавливаются не полностью.

Ответ: полностью можно расшифровать только 5393511, получится 5830829.

7.3. Сообщение состоит из $3 \times 17 = 51$ буквы. Поэтому $r = 3$ или $r = 17$ (при $r = 1$ и $r = 51$ — получается нечитаемый текст). При $r = 3$ не получается осмысленного текста при всех шести возможных вариантах перестановки букв ($a = 1, 2$, $b = 0, 1, 2$). Рассмотрим случай $r = 17$:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Б	Т	И	П	Ч	Ь	Л	О	Я	Ч	Ы	Ь	Т	О	Т	П	У
Н	Т	Н	О	Н	З	Л	Ж	А	Ч	О	Ь	О	Т	У	Н	И
У	Х	Н	И	П	П	О	Л	О	Ь	Ч	О	Е	Л	О	Л	С

Соседние буквы при перестановке переходят в буквы, отстоящие друг от друга на одинаковое расстояние: буква на x-м месте переходит на место, определяемое остатком от деления $ax + b$ на 17, а буква на $(x+1)$-м месте — на место, определяемое остатком от деления $(ax+b)+a$ на 17. Это верно для любого x. Поэтому есть всего 16 вариантов переходов соседних букв (исходный текст нечитаем), которые определяют однозначно переходы всех остальных букв. Перебирая их, получаем нечитаемые тексты во всех случаях, кроме одного, который дает текст:

Ч	И	Т	Ь	П	Я	Т	Ь	Ч	Т	О	Б	Ы	П	О	Л	У
Ч	Н	О	З	Н	А	Т	Ь	Н	У	Ж	Н	О	О	Т	Л	И
Ь	Н	Е	П	Л	О	Х	О	П	О	Л	У	Ч	И	Л	О	С

Из трех вариантов начала текста легко определяется истинный вариант.

Ответ:

ЧТОБЫПОЛУЧИТЬПЯТЬНУЖНООТЛИЧНОЗНАТЬПОЛУЧИЛОСЬНЕПЛОХО

7.4. Последовательность обхода доски показана на рисунке:

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

Ответ:

Кавалергардов век недолог
И потому так сладок он.
Труба трубит, откинут полог ...

7.5. Из однородности всех членов следует, что неравенство эквивалентно неравенству $a^3 + b^3 + c^3 + 6abc > 1/4$ при условии $a + b + c = 1$, $a > 0$, $b > 0$, $c > 0$.

Пусть c — минимальное из чисел a, b, c $(0 < c \leqslant 1/3)$ и $a = x$. Тогда

$$A = a^3 + b^3 + c^3 + 6abc - \frac{1}{4} =$$

$$= x^3 + (1 - c - x)^3 + c^3 + 6x(1 - c - x)c - \frac{1}{4} =$$

$$= 3(1-3c)x^2 - 3(1-c)(1-3c)x + (1-c)^3 + c^3 - \frac{1}{4}.$$

Находим минимум квадратного трехчлена с параметром c и положительным коэффициентом при x^2. Минимум достигается в точке $x = (1-c)/2$, при этом значение A будет положительным.

7.6. Если мелом с квадратным сечением нарисовать на доске отрезок прямой так, чтобы стороны сечения были параллельны краям доски, то площадь полученной линии будет равна площади ступенчатой линии с такими же концами (см. рис. 13).

Если на доске нарисовать некоторый (выпуклый) многоугольник, то найдутся такие граничные «точки» этого многоугольника, которые являются ближайшими к одному из краев доски. Площадь границы прямоугольника, содержащей все такие «точки», равна площади границы нарисованного выпуклого многоугольника (см. рис. 14).

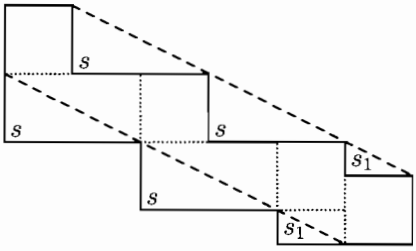


Рис. 13.

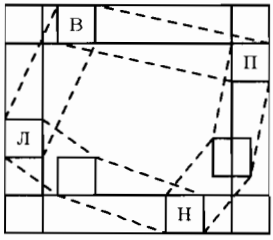


Рис. 14.

Такой прямоугольник назовем окаймляющим. Ясно, что площадь окаймляющего прямоугольника не меньше площади соответствующего многоугольника. Значит, для любого многоугольника данной площади найдется прямоугольник такой же площади, но с площадью границы не большей, чем площадь границы исходного многоугольника.

Если многоугольник со сторонами a и b имеет площадь 10000 см^2, то площадь его границы равна

$$2a + 2b + 4 = 2a + \frac{20000}{a} + 4 = 2(\sqrt{a} - \frac{100}{\sqrt{a}})^2 + 404.$$

Минимум достигается в случае, когда возводимое в квадрат выражение равно 0. В этом случае $a = 100$, что влечет $b = 100$. Таким образом, наименьшую площадь границы, равную 404 см^2, имеет квадрат со стороной 1 м.

Ответ: квадрат со стороной 1 м; площадь его границы — 404 см^2.

7.7. Если группа цифр, из которой образуются числа, состоит из k цифр, то существует ровно $k!$ различных чисел, для записи которых

используются все цифры группы ровно по одному разу. Группу из k цифр будем обозначать G_k.

Поскольку в сообщении отсутствуют цифры 2 и 9, эти цифры образуют либо две группы по одной цифре, либо одну группу из двух цифр. В обоих случаях эти цифры могут быть использованы для зашифрования ровно двух букв алфавита.

Так как $31 = 1! + 3! + 4!$, то $\{1, 3, 4, 5, 6, 7, 8, 0\} = G_1 \cup G_3 \cup G_4$.

Если $G_1 \neq \{1\}$, то из сообщения находим:
а) $G_4 = \{1, 3, 7, 8\}$, $G_3 = \{0, 5, 6\}$, $G_1 = \{4\}$ либо
б) $G_4 = \{1, 3, 7, 8\}$, $G_3 = \{4, 5, 6\}$, $G_1 = \{0\}$.

	Случай а	Случай б		Случай а	Случай б		Случай а	Случай б
А	2 (4)	0	К	1738	1738	Х	7183	7183
Б	4 (29)	2 (29)	Л	1783	1783	Ц	7318	7318
В	9 (56)	9 (92)	М	1837	1837	Ч	7381	7381
Г	56 (65)	456	Н	1873	1873	Ш	7813	7813
Д	65 (92)	465	О	3178	3178	Щ	7831	7831
Е	506	546	П	3187	3187	Ъ	8137	8137
Ё	605	564	Р	3718	3718	Ы	8173	8173
Ж	650	645	С	3781	3781	Ь	8317	8317
З	650	654	Т	3817	3817	Э	8371	8371
И	1378	1378	У	3871	3871	Ю	8713	8713
Й	1387	1387	Ф	7138	7138	Я	8731	8731

Сообщение после расшифрования имеет вид: а) ЯАЗЧ или б) ЯДАЧ, т. е. не читается.

Если $G_1 = \{1\}$, то из сообщения находим $G_3 = \{3,\ 7,\ 8\}$, $G_4 = \{0, 4, 5, 6\}$. В этом случае таблица замены букв числами имеет вид:

А	1	Ё	465	Л	783	С	4560	Ч	5460	Э	6450
Б	2(29)	Ж	546	М	837	Т	4605	Ш	5604	Ю	6504
В	9(92)	З	564	Н	873	У	4650	Щ	5640	Я	6540
Г	378	И	645	О	4056	Ф	5046	Ъ	6045		
Д	387	Й	654	П	4065	Х	5064	Ы	6054		
Е	456	К	738	Р	4506	Ц	5406	Ь	6405		

Сообщение легко прочитать: НАУКА.

Литература к главе 7

[1] *М. Гарднер.* От мозаик Пенроуза к надежным шифрам. М.: Мир, 1993.

[2] *У. Болл, Г. Кокстер.* Математические эссе и развлечения. М.: Мир, 1986.

[3] *С. А. Дориченко, В. В. Ященко.* 25 этюдов о шифрах. М.: ТЕИС, 1994.

[4] *В. Жельников.* Криптография от папируса до компьютера. М.: ABF, 1996.

[5] *Г. Фролов.* Тайны тайнописи. М., 1992.

[6] *Ч. Уэзерелл.* Этюды для программистов. М.: Мир, 1982.

[7] *А. Саломаа.* Криптография с открытым ключом. М.: Мир, 1995.

[8] *Т. А. Соболева.* Тайнопись в истории России (История криптографической службы России XVIII – начала XX в.). М., 1994.

[9] *Б. Анин, А. Петрович.* Радиошпионаж. М.: Международные отношения, 1996.

[10] *Г. А. Гуревич.* Криптограмма Жюля Верна. // Квант, №9, 1985.

[11] *В. Каверин.* Собрание сочинений в 6 т., т. 2. «Исполнение желаний» С. 211–552. М.: Художественная литература, 1964.

[12] *Э. По.* Стихотворения. Проза («Золотой жук» С. 433–462). М.: Художественная литература, 1976.

[13] *А. Конан Дойл.* Записки о Шерлоке Холмсе. «Пляшущие человечки» С. 249–275. М.: Правда, 1983.

[14] *Ж. Верн.* Собрание сочинений в 12 т. «Путешествие к центру Земли» С. 7–225. М.: Художественная литература, 1995.

[15] *Ж. Верн.* Жангада. Библиотечка приключений. Т. 9. М.: Детская литература, 1967.

Приложение

Отрывок из статьи К. Шеннона «Теория связи в секретных системах[1)]»

Материал, изложенный в данной статье, первоначально составлял содержание секретного доклада «Математическая теория криптографии», датированного 1 сентября 1945 г., который в настоящее время[2)] рассекречен.

1. Введение и краткое содержание

Вопросы криптографии и секретных систем открывают возможность для интересных применений теории связи. В настоящей статье развивается теория секретных систем. Изложение ведется в теоретическом плане и имеет своей целью дополнить положения, приводимые в обычных работах по криптографии. В этих работах детально изучаются многие стандартные типы кодов и шифров, а также способы их расшифровки. Мы будем иметь дело с общей математической структурой и свойствами секретных систем.

Наше изложение будет ограничено в нескольких отношениях. Во-первых, имеются три общих типа секретных систем: 1) системы маскировки, которые включают применение таких методов, как невидимые чернила, представление сообщения в форме безобидного текста или маскировки криптограммы, и другие методы, при помощи которых факт наличия сообщения скрывается от противника; 2) тайные системы (например, инвертирование речи), в которых для раскрытия сообщения требуется специальное оборудование; 3) «собственно» секретные системы, где смысл сообщения скрывается при помощи шифра, кода и т. д., но само существование сообщения не скрывается и предполагается, что противник обладает любым специальным оборудованием, необходимым для перехвата и записи переданных сигналов. Здесь будет

[1)] Печатается по изданию: К. Шеннон «Работы по теории информации и кибернетике», М., ИЛ, 1963, с. 333–369 (перевод В. Ф. Писаренко).

[2)] 1949 год — *прим. ред.*

рассмотрен только третий тип систем, так как системы маскировки представляют в основном психологическую проблему, а тайные системы — техническую проблему.

Во-вторых, наше изложение будет ограничено случаем дискретной информации, где сообщение, которое должно быть зашифровано, состоит из последовательных дискретных символов, каждый из которых выбран из некоторого конечного множества. Эти символы могут быть буквами или словами некоторого языка, амплитудными уровнями «квантованной» речи или видеосигнала и т. д., но главное ударение будет сделано на случае букв.

Статья делится на три части. Резюмируем теперь кратко основные результаты исследования. В первой части излагается основная математическая структура секретных систем. В теории связи считается, что язык может рассматриваться как некоторый вероятностный процесс, который создает дискретную последовательность символов в соответствии с некоторой системой вероятностей. С каждым языком связан некоторый параметр D, который можно назвать избыточностью этого языка. Избыточность измеряет в некотором смысле, насколько может быть уменьшена длина некоторого текста в данном языке без потери какой-либо части информации. Простой пример: так как в словах английского языка за буквой q всегда следует только буква u, то u может быть без ущерба опущена. Значительные сокращения в английском языке можно осуществить, используя его статистическую структуру, частую повторяемость определенных букв или слов, и т. д. Избыточность играет центральную роль в изучении секретных систем.

Секретная система определяется абстрактно как некоторое множество отображений одного пространства (множества возможных сообщений) в другое пространство (множество возможных криптограмм). Каждое конкретное отображение из этого множества соответствует способу шифрования при помощи конкретного ключа.

Предполагается, что отображения являются взаимнооднозначными, так что если известен ключ, то в результате процесса расшифровки возможен лишь единственный ответ.

Предполагается далее, что каждому ключу (и, следовательно, каждому отображению) соответствует некоторая априорная вероятность – вероятность выбрать этот ключ. Аналогично каждому возможному сообщению соответствует априорная вероятность, определяемая задающим сообщение вероятностным процессом. Эти вероятности различных ключей и сообщений являются фактически априорными вероятностями для шифровальщика противника и характеризуют его априорные знания относительно интересующей его проблемы.

Чтобы использовать такую секретную систему, сначала выбирается некоторый ключ и посылается в точку приема. Выбор ключа определяет конкретное отображение из множества отображений, образующих систему. Затем выбирается сообщение и с помощью отображения, соответствующего выбранному ключу, из этого сообщения формируется криптограмма. Эта криптограмма передается в точку приема по некоторому каналу и может быть перехвачена противником. На приемном конце с помощью отображения, обратного выбранному, из криптограммы восстанавливают первоначальное сообщение.

Если противник перехватит криптограмму, он может с ее помощью сосчитать апостериорные вероятности различных возможных сообщений и ключей, которые могли быть использованы для составления такой криптограммы. Это множество апостериорных вероятностей образует его сведения о ключах и сообщениях после перехвата. «Сведения», таким образом, представляют собой некоторое множество предположений, которым приписаны вероятности. Вычисление апостериорных вероятностей является общей задачей расшифровки.

Проиллюстрируем эти понятия простым примером. В шифре простой подстановки со случайным ключом имеется 26! отображений, соответствующих 26! способам, которыми мы можем заменить 26 различных букв. Все эти способы равновозможны, и поэтому каждый имеет априорную вероятность 1/26! Если такой шифр применяется к «нормативному английскому языку» и предполагается, что шифровальщик противника не знает ничего об источнике сообщений, кроме того, что он создает английский текст, то априорными вероятностями различных сообщений из N букв являются просто их относительные частоты в нормативном английском тексте.

Если противник перехватил такую криптограмму из N букв, его апостериорные вероятности изменятся. Если N достаточно велико (скажем, 50 букв), имеется обычно единственное сообщение с апостериорной вероятностью, близкой к единице, в то время как все другие сообщения имеют суммарную вероятность, близкую к нулю. Таким образом, имеется, по существу, единственное «решение» такой криптограммы. Для меньших N (скажем, $N = 15$) обычно найдется много сообщений и ключей, вероятности которых сравнимы, и не найдется ни одного сообщения и ключа с вероятностью, близкой к единице. В этом случае «решение» криптограммы неоднозначно.

В результате рассмотрения секретных систем, которые могут быть представлены как совокупность отображений одного множества элементов в другое, возникают две естественные операции комбинирования, производящие из двух данных систем третью. Первая операция

комбинирования называется операцией «умножения» (произведением) и соответствует зашифровке сообщения с помощью системы R с последующей зашифровкой полученной криптограммы с помощью системы S, причем ключи R и S выбираются независимо. Полный результат этой операции представляет собой секретную систему, отображения которой состоят из всех произведений (в обычном смысле произведений отображений) отображений из S на отображения из R. Вероятности результирующих отображений являются произведениями вероятностей двух исходных отображений.

Вторая операция комбинирования является «взвешенным сложением»:

$$T = pR + qS, \quad p + q = 1.$$

Она представляет собой следующее. Сначала делается предварительный выбор, какая из систем R или S будет использоваться, причем система R выбирается с вероятностью p, а система S с вероятностью q. После этого выбранная система используется описанным выше способом.

Будет показано, что секретные системы с этими двумя операциями комбинирования образуют, по существу, «линейную ассоциативную алгебру» с единицей, — алгебраический объект, подробно изучавшийся математиками.

Среди многих возможных секретных систем имеется один тип с многочисленными особыми свойствами. Этот тип назовем «чистой» системой. Система является чистой, если все ключи равновероятны и если для любых трех отображений T_i, T_j, T_k из множества отображений данной системы произведение

$$T_i T_j^{-1} T_k$$

также является отображением из этого множества. То есть зашифровка, расшифровка и снова зашифровка с любыми тремя ключами должна быть эквивалентна зашифровке с некоторым ключом.

Можно показать, что для чистого шифра все ключи по существу эквивалентны — все они приводят к тому же самому множеству апостериорных вероятностей. Больше того, каждой криптограмме соответствует некоторое множество сообщений («остаточный класс»), из которых могла бы получиться эта криптограмма, а апостериорные вероятности сообщений в этом классе пропорциональны априорным вероятностям. Вся информация, которую противник получил бы в результате перехвата криптограммы, заключается в установлении остаточного класса. Многие из обычных шифров являются чистыми системами, в том числе простая подстановка со случайным ключом. В этом случае остаточный класс состоит из всех сообщений с таким

же набором буквенных повторений, как в перехваченной криптограмме.

По определению, две системы R и S являются «подобными», если существует фиксированное отображение A (имеющее обратное A^{-1}) такое, что

$$R = AS.$$

Если R и S подобны, то между получающимися в результате применения этих систем множествами криптограмм можно установить взаимнооднозначное соответствие, приводящее к тем же самым апостериорным вероятностям. Такие две системы аналитически записываются одинаково.

Во второй части статьи рассматривается проблема «теоретической секретности». Насколько легко некоторая система поддается раскрытию при условии, что для анализа перехваченной криптограммы противник располагает неограниченным количеством времени и специалистов? Эта проблема тесно связана с вопросами связи при наличии шумов, и понятия энтропии и неопределенности, введенные в теории связи, находят прямое применение в этом разделе криптографии.

«Совершенная секретность» определяется следующими требованиями к системе. Требуется, чтобы апостериорные вероятности различных сообщений, полученные после перехвата противником данной криптограммы, были бы в точности равны априорным вероятностям тех же сообщений до перехвата. Покажем, что «совершенная секретность» возможна, но требует в случае конечного числа сообщений того же самого числа возможных ключей. Если считать, что сообщение создается с данной «скоростью» R (понятие скорости будет определено позже), то ключ должен создаваться с той же самой или с большей скоростью.

Если используется секретная система с конечным ключом и перехвачены N букв криптограммы, то для противника будет существовать определенное множество сообщений с определенными вероятностями, которые могли бы создать эту криптограмму. С увеличением N это множество обычно сужается до тех пор, пока в конце концов не получится единственного «решения» криптограммы: одно сообщение с вероятностью, близкой к единице, а все остальные с вероятностями, практически равными нулю. В работе определяется величина $H(N)$, названная *ненадежностью*. Эта величина измеряет (в статистическом смысле), насколько близка средняя криптограмма из N букв к единственному решению, т. е. насколько неточно известно противнику истинное сообщение после перехвата криптограммы из N букв. Далее выводятся различные свойства ненадежности, например: ненадежность ключа

не возрастает с ростом N. Эта ненадежность является теоретическим показателем секретности — теоретическим, поскольку она позволяет противнику дешифрировать криптограмму лишь в том случае, если он обладает неограниченным запасом времени.

В этой же части определяется функция $H(N)$ для некоторых идеализированных типов шифров, называемых *случайными шифрами*. С некоторыми видоизменениями эта функция может быть применена ко многим случаям, представляющим практический интерес. Это дает способ приближенного вычисления количества материала, который требуется перехватить, чтобы получить решение секретной системы.

Из подобного анализа следует, что для обычных языков и обычных типов шифров (но не кодов) это «расстояние единственности» равно приблизительно $H(K)/D$. Здесь $H(K)$ — число, измеряющее «объем» пространства ключей. Если все ключи априори равновероятны, то $H(K)$ равно логарифму числа возможных ключей. Вводимое число D — это избыточность языка. Оно измеряет количество «статистических ограничений», налагаемых языком. Для простой подстановки со случайным ключом наше $H(K)$ равно $\log_{10} 26!$ или приблизительно 20, а D (в десятичных единицах на букву) для английского языка равно приблизительно 0,7. Таким образом, единственность решения достигается приблизительно при 30 буквах.

Для некоторых «языков» можно построить такие секретные системы с конечным ключом, в которых неопределенность не стремится к нулю при $N \to \infty$. В этом случае противник не получит единственного решения такого шифра, сколько бы материала он не перехватил, и у него будет оставаться много альтернатив с довольно большими вероятностями. Такие системы назовем *идеальными системами*. В любом языке можно аппроксимировать такую ситуацию, т. е. отсрочить приближение $H(N)$ к нулю до сколь угодно больших N. Однако такие системы имеют много недостатков, таких как сложность и чувствительность к ошибкам при передаче криптограммы.

Третья часть статьи посвящена «практической секретности». Две системы с одинаковым объемом ключа могут быть обе разрешимы единственным образом, когда перехвачено N букв, но они могут значительно отличаться по количеству времени и усилий, затрачиваемых для получения решения. На основе анализа основных недостатков секретных систем предлагаются методы построения систем, для решения которых требуются большие затраты времени и сил. Наконец, рассматривается проблема несовместимости различных желательных качеств секретных систем.

Часть I
МАТЕМАТИЧЕСКАЯ СТРУКТУРА СЕКРЕТНЫХ СИСТЕМ

2. Секретные системы

Чтобы приступить к математическому анализу криптографии, необходимо ввести удовлетворительную идеализацию и определить математически приемлемым способом, что будет пониматься под термином секретная система. Схематическая структура секретной системы показана на рис. 1.

На передающем конце имеются два источника информации — источник сообщений и источник ключей. Источник ключей отбирает

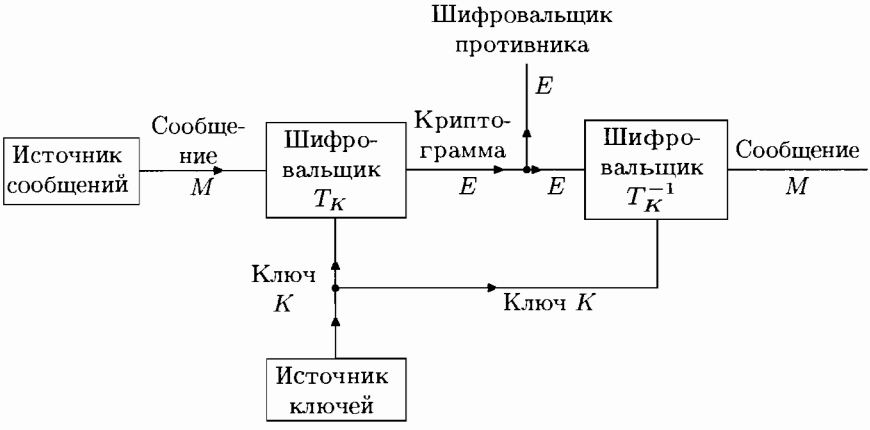


Рис. 1. *Схема общей секретной системы.*

конкретный ключ среди всех возможных ключей данной системы. Этот ключ передается некоторым способом на приемный конец, причем предполагается, что его нельзя перехватить (например, ключ передается посыльным). Источник сообщений формирует некоторое сообщение (незашифрованное), которое затем зашифровывается, и готовая криптограмма передается на приемный конец, причем криптограмма может быть перехвачена (например, пересылается по радио). На приемном конце шифровальщик с помощью ключа по криптограмме восстанавливает исходное сообщение.

Очевидно, шифровальщик на передающем конце выполняет некоторую функциональную операцию. Если M — сообщение, K — ключ и E — зашифрованное сообщение (криптограмма), то имеем

$$E = f(M, K),$$

т. е. E является функцией от M и K. Удобнее, однако, понимать E не как функцию двух переменных, а как (однопараметрическое) семейство операций или отображений, и записывать его в виде:

$$E = T_i M.$$

Отображение T_i, примененное к сообщению M, дает криптограмму E. Индекс i соответствует конкретному используемому ключу.

Вообще мы будем предполагать, что имеется лишь конечное число возможных ключей, каждому из которых соответствует вероятность p_i. Таким образом, источник ключей является статистическим процессом, или устройством, которое выбирает одно из множества отображений $T_1, \ldots, T_m$ с вероятностями $p_1, \ldots p_m$ соответственно. Будем также предполагать, что число возможных сообщений конечно и эти сообщения $M_1, \ldots M_n$ имеют априорные вероятности $q_1, \ldots, q_n$. Например, возможными сообщениями могли бы быть всевозможные последовательности английских букв, включающих по N букв каждая, а соответствующими вероятностями тогда были бы относительные частоты появления таких последовательностей в нормативном английском тексте.

Должна иметься возможность восстанавливать M на приемном конце, когда известны E и K. Поэтому отображение T_i из нашего семейства должно иметь единственное обратное отображение T_i^{-1}, так что $T_i T_i^{-1} = I$, где I — тождественное отображение. Таким образом:

$$M = T_i^{-1} E.$$

Во всяком случае, это обратное отображение T_i^{-1} должно существовать и быть единственным для каждого E, которое может быть получено из M с помощью ключа i. Приходим, таким образом, к следующему определению: секретная система есть семейство однозначно обратимых отображений T_i множества возможных сообщений во множество криптограмм, при этом отображение T_i имеет вероятность p_i. Обратно, любое множество объектов такого типа будет называться «секретной системой». Множество возможных сообщений для удобства будет называться «пространством сообщений», а множество возможных криптограмм — «пространством криптограмм».

Две секретные системы совпадают, если они образованы одним и тем же множеством отображений T_i и одинаковыми пространствами сообщений и криптограмм, причем вероятности ключей в этих системах также совпадают.

Секретную систему можно представлять себе как некоторую машину с одним или более переключающими устройствами. Последовательность букв (сообщение) поступает на вход машины, а на выхо-

де ее получается другая последовательность. Конкретное положение переключающих устройств соответствует конкретному используемому ключу. Для выбора ключа из множества возможных ключей должны быть заданы некоторые статистические методы.

Для того чтобы нашу проблему можно было рассмотреть математически, предположим, что противнику известна используемая система. Иными словами, он знает семейство отображений T_i и вероятности выбора различных ключей. Можно было бы, во-первых, возразить, что такое предположение нереалистично, так как шифровальщик противника часто не знает, какая система использовалась или чему равны рассматриваемые вероятности. На это возражение имеется два ответа.

1. Наложенное ограничение слабее, чем кажется с первого взгляда, из-за широты нашего определения секретной системы. Предположим, что шифровальщик перехватывает сообщение и не знает, использовалась ли здесь подстановка, или транспозиция, или шифр типа Виженера. Он может считать, что сообщение зашифровано с помощью системы, в которой часть ключа является указанием того, какой из трех типов имеющихся ключей был использован, а следующая часть — конкретный ключ этого типа. Указанным трем различным возможностям шифровальщик приписывает вероятности, учитывая при этом все имеющиеся у него сведения об априорных вероятностях использования шифровальщиком противника соответствующих типов шифров.

2. Наше ограничение обычно в криптографических исследованиях. Оно является пессимистичным, но безопасно, и в конечном счете реалистично, так как можно ожидать, что противник рано или поздно раскроет любую секретную систему. Поэтому даже в том случае, когда разработана совершенно новая система, так что противник не может приписать ей никаких априорных вероятностей, если только он ее уже не раскрыл, нужно иметь в виду его возможную осведомленность.

Эта ситуация аналогична ситуации, возникающей в теории игр, где предполагается, что партнер «обнаруживает» используемую стратегию игры. В обоих случаях это предположение служит для более четкого описания сведений, которыми располагает противная сторона.

Второе возможное возражение против нашего определения секретной системы состоит в том, что в нем не принимаются в расчет используемые обычно на практике вставки в сообщение посторонних нулевых знаков и использование многократных подстановок. В таких случаях для данного сообщения и ключа имеется не единственная криптограмма и шифровальщик может выбрать по своему желанию одну из нескольких различных криптограмм. Эту ситуацию можно было бы рассмотреть, но это только внесло бы дополнительные усложнения на

данном этапе рассуждений без существенного изменения каких-либо из основных выводов.

Если сообщения создаются марковским процессом, то вероятности разных сообщений определяются структурой этого марковского процесса. Однако подойдем к вопросу с более общей точки зрения и будем трактовать сообщения просто как абстрактное множество объектов, которым приписаны вероятности, причем эти объекты не обязательно состоят из последовательностей букв и не обязательно создаются марковским процессом.

Следует подчеркнуть, что далее во всех случаях секретная система означает не одно, а целое множество отображений. После того как выбран ключ, используется только одно из этих отображений и отсюда можно было бы прийти к определению секретной системы как единственного преобразования языка. Однако противник не знает, какой ключ выбран, и остальные возможные ключи столь же важны для него, как и истинный. Именно существование этих других возможных ключей и придает системе секретность. Так как мы интересуемся в первую очередь секретностью, то вынуждены предпочесть данное нами определение понятия секретной системы. Тип ситуации, когда остальные возможности так же важны, как и осуществившаяся, часто встречается в стратегических играх. Ход шахматной игры в большой степени контролируется угрозами, которые не осуществляются. Нечто подобное представляет из себя «фактическое существование» нереализованных возможностей в теории игр.

Следует отметить, что система, состоящая из единственной операции над языком, представляет собой при нашем определении вырожденный тип секретной системы. Это — система с единственным ключом, который имеет вероятность, равную единице. В такой системе нет секретности — шифровальщик противника находит сообщение, применяя к перехваченной криптограмме обратное отображение, также единственное в такой системе. В этом случае шифровальщик противника и шифровальщик получателя информации располагают одинаковой информацией. В общем же случае единственное различие их сведений состоит в том, что последнему известен конкретно использовавшийся ключ, в то время как первому известны лишь априорные вероятности различных ключей из данного множества. Процесс расшифровки для получателя информации состоит в применении к криптограмме отображения, обратного по отношению к конкретному отображению, использованному для составления криптограммы. Процесс расшифровки для противника представляет собой попытку определить сообщение (или конкретный ключ), имея в распоряжении только криптограмму и априорные вероятности различных ключей и сообщений.

Существует много трудных эпистемологических вопросов, связанных с теорией секретности, или вернее с любой теорией, связанной с реальным применением вопросов теории вероятностей (так обстоит дело, в частности, с априорными вероятностями, теоремой Байеса и т.д.). Трактуемая абстрактно теория вероятности может быть изложена на строгих логических основах с использованием современной теории меры. Однако в применениях к физическим ситуациям, особенно когда дело касается «субъективных» вероятностей и неповторимых экспериментов, возникают многочисленные вопросы, связанные с логическим обоснованием. Например, при нашем подходе к проблеме секретности допускается, что априорные вероятности различных ключей и сообщений известны шифровальщику противника, но как он может определить их эффективным способом даже при использовании всех своих сведений о данной обстановке?

Можно создать искусственные криптографические ситуации типа «урны и игральной кости», в которых априорные вероятности имеют вполне определенный смысл и идеализация, использованная здесь, является наверняка подходящей. Но в других случаях, которые можно себе представить, например, при перехвате сообщений, передаваемых между собой марсианами, высадившимися на Землю, априорные вероятности были бы настолько неопределенными, что не имели бы никакого значения.

Наиболее часто встречающиеся на практике криптографические задачи лежат где-то между этими крайними пределами. Шифровальщик противника может иметь желание разделить возможные сообщения на категории «приемлемых», «возможных, но малоправдоподобных» и «неприемлемых», но чувствуется, что более подробное подразделение не имело бы смысла.

К счастью, на практике только очень большие ошибки в априорных вероятностях ключей и сообщений могут вызвать заметные ошибки в важных параметрах. Это происходит из-за того, что число сообщений и криптограмм ведет себя как экспоненциальная функция, а измеряется логарифмической мерой.

3. Способы изображения систем

Секретная система, в том виде как она определена выше, может быть изображена различными способами. Один из них (удобный для целей иллюстрации) использует линейные схемы, изображенные на рис. 2 и рис. 4. Возможные сообщения представляются точками слева, а возможные криптограммы — точками справа. Если некоторый ключ, скажем, ключ 1, отображает сообщение M_2 в криптограмму E_2, то M_2

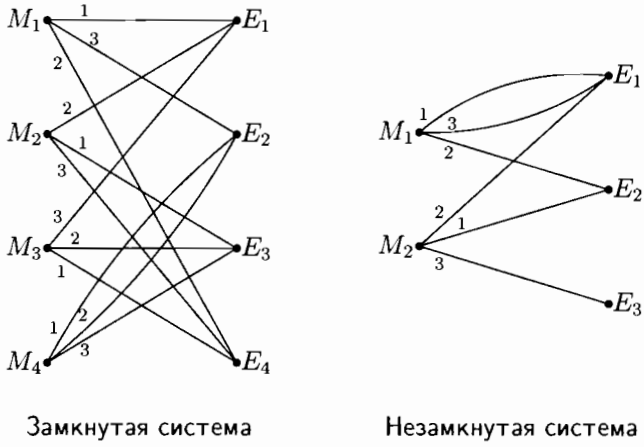


Рис. 2. *Схемы простых систем.*

и E_2 соединяются линией, обозначенной значком 1 и т.д. Для каждого ключа из каждого сообщения должна выходить ровно одна линия. Если это же верно и для каждой криптограммы, скажем, что система является *замкнутой.*

Более общий способ описания системы состоит в задании операции, с помощью которой, применяя к сообщению произвольный ключ, можно получить криптограмму. Аналогично неявным образом можно определить вероятности различных ключей или с помощью задания способа выбора ключей, или с помощью описания сведений о том, как обычно выбирает ключи противник. Вероятности сообщений определяются просто посредством изложения наших априорных сведений о языке противника, тактической обстановке (которая будет влиять на возможное содержание сообщений) и любой специальной информации, касающейся криптограммы.

4. Примеры секретных систем

В данном разделе рассматриваются несколько примеров шифров. В дальнейшем в целях иллюстрации будем часто ссылаться на эти примеры.

1. *Шифр простой подстановки.*

В таком шифре производится замена каждой буквы сообщения на некоторый определенный символ (обычно также на букву).

Таким образом, сообщение

$$M = m_1 m_2 m_3 m_4 \ldots,$$

где $m_1, m_2, \ldots$ — последовательные буквы, переходит в

$$E = e_1 e_2 e_3 e_4 \cdots = f(m_1) f(m_2) f(m_3) f(m_4) \ldots,$$

причем функция $f(m)$ имеет обратную функцию. Ключ является просто перестановкой алфавита (если буквы заменяются на буквы), например,

$$XGUACDTBFHRSLMQVYZWIEJOKNP.$$

Первая буква — X заменяет букву A, G заменяет B и т. д.

2. *Транспозиция с фиксированным периодом d.*

В этом случае сообщение делится на группы символов длины d и к каждой группе применяется одна и та же перестановка. Эта перестановка является ключом; она может быть задана некоторой перестановкой первых d целых чисел.

Таким образом, для $d = 5$ в качестве перестановки можно взять 2 3 1 5 4. Это будет означать, что

$$m_1 m_2 m_3 m_4 m_5 m_6 m_7 m_8 m_9 m_{10} \ldots$$

переходит в

$$m_2 m_3 m_1 m_5 m_4 m_7 m_8 m_6 m_{10} m_9 \ldots .$$

Последовательное применение двух или более транспозиций будет называться составной транспозицией. Если периоды этих транспозиций $d_1, \ldots, d_s$, то, очевидно, в результате получится транспозиция периода d, где d — наименьшее общее кратное $d_1, \ldots, d_s$.

3. *Шифр Виженера и его варианты.*

В шифре Виженера ключ задается набором из d букв. Такие наборы подписываются с повторением под сообщением и полученные две последовательности складываются по модулю 26 (каждая буква рассматриваемого алфавита нумеруется от $A = 0$ до $Z = 25$).

Таким образом,

$$l_i = m_i + k_i \pmod{26},$$

где k_i — буква ключа, полученная сокращением числа i по модулю d. Например, с помощью ключа GAH получаем

Сообщение	N	O	W	I	S	T	H	E
Повторяемый ключ	G	A	H	G	A	H	G	A
Криптограмма	T	O	D	O	S	A	N	E

Шифр Виженера с периодом 1 называется шифром Цезаря. Он представляет собой простую подстановку, в которой каждая буква сообщения M сдвигается вперед на фиксированное число мест по алфавиту. Это число и является ключом; оно может быть любым от 0 до 25. Так называемый шифр Бофора (Beaufort) и видоизмененный шифр Бофора

подобны шифру Виженера. В них сообщения зашифровываются с помощью равенств

$$l_i = k_i - m_i \pmod{26} \text{ и}$$
$$l_i = m_i - k_i \pmod{26}$$

соответственно. Шифр Бофора с периодом 1 называется обратным шифром Цезаря.

Повторное применение двух или более шифров Виженера будет называться составным шифром Виженера. Он имеет уравнение

$$l_i = m_i + k_i + l_i + \cdots + s_i \pmod{26},$$

где $k_i, l_i, \ldots, s_i$ вообще говоря, имеют различные периоды. Период их суммы $k_i + l_i + \cdots + s_i$, как и в составной транспозиции, будет наименьшим общим кратным отдельных периодов.

Если используется шифр Виженера с неограниченным неповторяющимся ключом, то мы имеем шифр Вернама, в котором

$$l_i = m_i + k_i \pmod{26}$$

и k_i выбираются случайно и независимо среди чисел 0, 1, ... , 25. Если ключом служит текст, имеющий смысл, то имеем шифр «бегущего ключа».

4. *Диграммная, триграммная и n-граммная подстановки.*

Вместо подстановки одной буквы можно использовать подстановку диграмм, триграмм и т. д. Для диграммной подстановки в общем виде требуется ключ, состоящий из перестановок 26^2 диграмм. Он может быть представлен с помощью таблицы, в которой ряд соответствует первой букве диграммы, а столбец — второй букве, причем клетки таблицы заполнены заменяющими символами (обычно также диграммами).

5. *Шифр Виженера с перемешанным один раз алфавитом.*

Такой шифр представляет собой простую подстановку с последующим применением шифра Виженера

$$l_i = f(m_i) + k_i,$$
$$m_i = f^{-1}(l_i - k_i).$$

«Обратным» к такому шифру является шифр Виженера с последующей простой подстановкой

$$l_i = g(m_i + k_i),$$
$$m_i = g^{-1}(l_i) - k_i.$$

6. *Матричная система*

Имеется один метод подстановки n-грамм, который заключается в применении к последовательным n-граммам некоторой матрицы, име-

ющей обратную. Предполагается, что буквы занумерованы от 0 до 25 и рассматриваются как элементы некоторого алгебраического кольца. Если к n-грамме сообщения применить матрицу a_{ij}, то получится n-грамма криптограммы

$$l_i = \sum_{j=1}^{n} a_{ij} m_j, \qquad i = 1, \ldots, n.$$

Матрица a_{ij} является ключом, и расшифровка выполняется с помощью обратной матрицы. Обратная матрица будет существовать тогда и только тогда, когда определитель $|a_{ij}|$ имеет обратный элемент в нашем кольце.

7. *Шифр Плэйфер*

Этот шифр является частным видом диграммной подстановки, которая производится с помощью перемешанного алфавита из 25 букв, записанных в виде квадрата 5×5. (Буква J часто опускается при криптографической работе, так как она редко встречается, и в тех случаях, когда она встречается, ее можно заменить буквой I). Предположим, что ключевой квадрат записывается следующим образом:

$$\begin{matrix} L & Z & Q & C & P \\ A & G & N & O & U \\ R & D & M & I & F \\ K & Y & H & V & S \\ X & B & T & E & W. \end{matrix}$$

В этом случае диграмма AC, например, заменяется на пару букв, расположенных в противоположных углах прямоугольника, определяемого буквами A и C, т. е. на LO, причем L взята первой, так как она выше A. Если буквы диграммы расположены на одной горизонтали, то используются стоящие справа от них буквы. Таким образом, RI заменяется на DF, RF заменяется на DR. Если буквы расположены на одной вертикали, то используются буквы, стоящие под ними. Таким образом, PS заменяется на UW. Если обе буквы диграммы совпадают, то можно использовать для их разделения нуль или же одну из букв опустить и т. п.

8. *Перемешивание алфавита с помощью многократной подстановки.*

В этом шифре используются последовательно d простых подстановок. Так, если $d = 4$, то

$$m_1 m_2 m_3 m_4 m_5 m_6 \ldots$$

заменяется на

$$f_1(m_1) f_2(m_2) f_3(m_3) f_4(m_4) f_1(m_5) f_2(m_6) \ldots$$

и т. д.

9. *Шифр с автоключом.*

Шифр типа Виженера, в котором или само сообщение или результирующая криптограмма используются в качестве «ключа», называется шифром с автоключом. Шифрование начинается с помощью «первичного ключа» (который является настоящим ключом в нашем смысле) и продолжается с помощью сообщения или криптограммы, смещенной на длину первичного ключа, как в указанном ниже примере, где первичным ключом является набор букв $COMET$. В качестве «ключа» используется сообщение:

Сообщение	S	E	N	D	S	U	P	P	L	I	E	S	...
Ключ	C	O	M	E	T	S	E	N	D	S	U	P	...
Криптограмма	U	S	Z	H	L	M	T	C	O	A	Y	H	...

Если в качестве «ключа» использовать криптограмму, то получится[1)]

Сообщение	S	E	N	D	S	U	P	P	L	I	E	S	...
Ключ	C	O	M	E	T	U	S	Z	H	L	O	H	...
Криптограмма	U	S	Z	H	L	O	H	O	S	T	T	S	...

10. *Дробные шифры.*

В этих шифрах каждая буква сначала зашифровывается в две (или более) буквы или в два (или более) числа, затем полученные символы каким-либо способом перемешиваются (например, с помощью транспозиции), после чего их можно снова перевести в первоначальный алфавит. Таким образом, используя в качестве ключа перемешанный 25-буквенный алфавит, можно перевести буквы в двухзначные пятеричные числа с помощью таблицы:

	0	1	2	3	4
0	L	Z	Q	C	P
1	A	G	N	O	U
2	R	D	M	I	F
3	K	Y	H	V	S
4	X	B	T	E	W

Например, букве B соответствует «число» 41. После того как полученный ряд чисел подвергнут некоторой перестановке, его можно снова разбить на пары чисел и перейти к буквам.

11. *Коды.*

В кодах слова (или иногда слоги) заменяются группами букв. Иногда затем применяется шифр того или иного вида.

[1)] Эта система является тривиальной с точки зрения секретности, так как, за исключением первых d букв, в распоряжении противника имеется весь «ключ».

5. Оценка секретных систем

Имеется несколько различных критериев, которые можно было бы использовать для оценки качества предлагаемой секретной системы. Рассмотрим наиболее важные из этих критериев.

1. *Количество секретности.*

Некоторые секретные системы являются совершенными в том смысле, что положение противника не облегчается в результате перехвата любого количества сообщений. Другие системы, хотя и дают противнику некоторую информацию при перехвате очередной криптограммы, но не допускают единственного «решения». Системы, допускающие единственное решение, очень разнообразны как по затрате времени и сил, необходимых для получения этого решения, так и по количеству материала, который необходимо перехватить для получения единственного решения.

2. *Объем ключа.*

Ключ должен быть передан из передающего пункта в приемный пункт таким способом, чтобы его нельзя было перехватить. Иногда его нужно запомнить. Поэтому желательно иметь ключ настолько малый, насколько это возможно.

3. *Сложность операции шифрования и дешифрирования.*

Операции шифрования и дешифрирования должны быть, конечно, по возможности простыми. Если эти операции производятся вручную, то их сложность приводит к потере времени, появлению ошибок и т. д. Если они производятся механически, то сложность приводит к использованию больших и дорогих устройств.

4. *Разрастание числа ошибок.*

В некоторых типах шифров ошибка в одной букве, допущенная при шифровании или передаче, приводит к большому числу ошибок в расшифрованном тексте. Такие ошибки разрастаются в результате операции дешифрирования, вызывая значительную потерю информации и часто требуя повторной передачи криптограммы. Естественно, желательно минимизировать это возрастание числа ошибок.

5. *Увеличение объема сообщения.*

В некоторых типах секретных систем объем сообщения увеличивается в результате операции шифрования. Этот нежелательный эффект можно наблюдать в системах, в которых делается попытка потопить статистику сообщения в массе добавляемых нулевых символов, или где используются многократные замены. Он имеет место также во многих системах типа «маскировки» (которые не являются обычными секретными системами в смысле нашего определения).

6. Алгебра секретных систем

Если имеются две секретные системы T и R, их часто можно комбинировать различными способами для получения новой секретной системы S. Если T и R имеют одну и ту же область (пространство сообщений), то можно образовать своего рода «взвешенную сумму»

$$S = pT + qR,$$

где $p + q = 1$. Эта операция состоит, во-первых, из предварительного выбора систем T или R с вероятностями p и q. Этот выбор является частью ключа S. После того как этот выбор сделан, системы T или R применяются в соответствии с их определениями. Полный ключ S должен указывать, какая из систем T или R выбрана и с каким ключом используется выбранная система.

Если T состоит из отображений $T_1, \ldots, T_m$ с вероятностями $p_1, \ldots$ $\ldots, p_m$, а R — из $R_1, \ldots, R_k$ с вероятностями $q_1, \ldots, q_k$, то система $S =$ $= pT + qR$ состоит из отображений $T_1, \ldots, T_m, R_1, \ldots, R_k$ с вероятностями $pp_1, \ldots$ $\ldots, pp_m, qq_1, \ldots, qq_k$ соответственно.

Обобщая далее, можно образовать сумму нескольких систем

$$S = p_1T + p_2R + \cdots + p_mU, \quad \sum p_i = 1.$$

Заметим, что любая система T может быть записана как сумма фиксированных операций

$$T = p_1T_1 + p_2T_2 + \cdots + p_mT_m,$$

где T_i — определенная операция шифрования в системе T, соответствующая выбору ключа i, причем вероятность такого выбора равна p_i.

Второй способ комбинирования двух секретных систем заключается в образовании «произведения», как показано схематически на рис. 3. Предположим, что T и R — такие две системы, что область определения (пространство языка) системы R может быть отождествлена с областью определения (пространством криптограмм) системы T. Тогда можно применить сначала систему T к нашему языку, а затем систему R к результату этой операции, что дает результирующую операцию S, которую запишем в виде произведения

$$S = RT.$$

Ключ системы S состоит как из ключа системы T, так и из ключа системы R, причем предполагается, что эти ключи выбираются соответственно их первоначальным вероятностям и независимо. Таким образом, если m ключей системы T выбирается с вероятностями

$$p_1, p_2, \ldots, p_m,$$

а n ключей системы R имеют вероятности

$$p'_1, p'_2, \ldots, p'_n,$$

то система S имеет самое большее mn ключей с вероятностями $p_i p'_j$. Во многих случаях некоторые из отображений $R_i T_j$ будут одинаковыми и могут быть сгруппированы вместе, а их вероятности при этом сложатся.

Произведение шифров используется часто; например, после подстановки применяют транспозицию или после транспозиции — код Виженера; или же применяют код к тексту и зашифровывают результат с помощью подстановки, транспозиции, дробным шифром и т. д.

Можно заметить, что такое умножение, вообще говоря, некоммутативно (т. е. не всегда $RS = SR$), хотя в частных случаях (таких, как подстановка и транспозиция) коммутативность имеет место. Так как наше умножение представляет собой некоторую операцию, оно по определению ассоциативно, т. е. $R(ST) = (RS)T = RST$. Кроме того, верны законы

$$p(p'T + q'R) + qS = pp'T + pq'R + qS$$

(взвешенный ассоциативный закон для сложения);

$$T(pR + qS) = pTR + qTS$$

$$(pR + qS)T = pRT + qST$$

(право- и левосторонние дистрибутивные законы), а также справедливо равенство

$$p_1T + p_2T + p_3R = (p_1 + p_2)T + p_3R.$$

Следует подчеркнуть, что эти операции комбинирования сложения и умножения применяются к секретным системам в целом. Произведение двух систем TR не следует смешивать с произведением отображений в системах T_iR_j, которое также часто используется в настоящей работе. Первое является секретной системой, т. е. множеством отображений с соответствующими вероятностями; второе — является фиксированным отображением. Далее, в то время как сумма двух систем $pR + qT$ является системой, сумма двух отображений не определена. Системы T и R могут коммутировать, в то время как конкретные R_j и T_i не коммутируют. Например, если R — система Бофора данного периода,

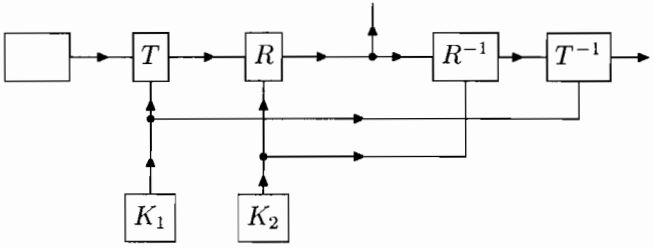


Рис. 3. *Произведение двух систем $S = RT$.*

все ключи которой равновероятны, то, вообще говоря,

$$R_iR_j \neq R_jR_i,$$

но, конечно, произведение RR не зависит от порядка сомножителей; действительно

$$RR = V$$

является системой Виженера того же самого периода со случайным ключом. С другой стороны, если отдельные отображения T_i и R_j двух систем T и R коммутируют, то и системы коммутируют.

Системы, у которых пространства M и E можно отождествить (этот случай является очень частым, если последовательности букв преобразуются в последовательности букв), могут быть названы *эндоморфными*. Эндоморфная система T может быть возведена в степень T^n.

Секретная система T, произведение которой на саму себя равно T, т. е. такая, что

$$TT = T,$$

будет называться *идемпотентной*. Например, простая подстановка, транспозиция с периодом p, система Виженера с периодом p (все с равновероятными ключами) являются идемпотентными.

Множество всех эндоморфных секретных систем, определенных в фиксированном пространстве сообщений, образует «алгебраическую систему», т. е. некоторый вид алгебры, использующей операции сложения и умножения. Действительно, рассмотренные свойства сложения и умножения можно резюмировать следующим образом.

Множество эндоморфных шифров с одним и тем же пространством сообщений и двумя операциями комбинирования — операцией взвешенного сложения и операцией умножения — образуют линейную ассоциативную алгебру с единицей, с той лишь особенностью, что коэффициенты во взвешенном сложении должны быть неотрицательными, а их сумма должна равняться единице.

Эти операции комбинирования дают способы конструирования многих новых типов секретных систем из определенных данных систем, как это было показано в приведенных примерах. Их можно также использовать для описания ситуации, с которой сталкивается шифровальщик противника, когда он пытается расшифровать криптограмму неизвестного типа. Фактически он расшифровывает секретную систему типа

$$T = p_1A + p_2B + \cdots + p_rS + p'X, \quad \sum p_i = 1,$$

где $A, B, \ldots, S$ в данном случае — известные типы шифров с их априорными вероятностями p_i, а $p'X$ соответствует возможности использования совершенно нового неизвестного шифра.

7. Чистые и смешанные шифры

Некоторые типы шифров, такие как простая подстановка, транспозиция с данным периодом, система Виженера с данным периодом, система Виженера со смешанным алфавитом и т. д. (все с равновероятными ключами), обладают некоторой однородностью по отношению к ключу. Каков бы ни был ключ, процессы шифрования, дешифрирования адресатом и дешифрирования противником являются по существу теми же самыми. Эти системы можно противопоставить системе с шифром

$$pS + qT,$$

где S — простая подстановка, а T — транспозиция с данным периодом. В таком случае процессы шифрования и дешифрирования адресатом или противником полностью меняются в зависимости от того, используется подстановка или транспозиция.

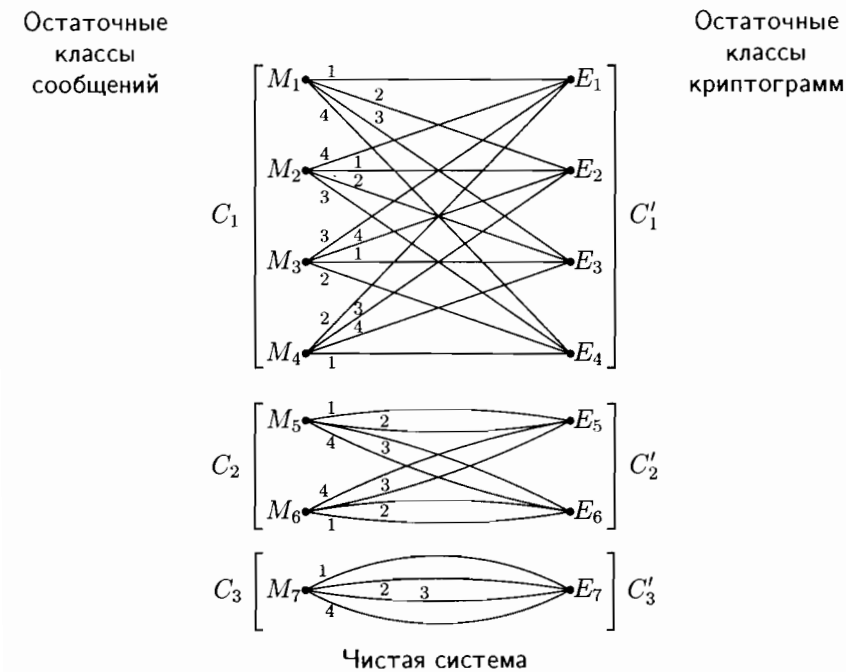


Рис. 4.

Причина однородности таких систем лежит в групповом свойстве: заметим, что в приведенных выше примерах однородных шифров произведение T_iT_j любых двух отображений из множества равно третьему отображению T_k из этого же множества. С другой стороны, T_iS_j не равно какому-нибудь отображению для шифра

$$pS + qT,$$

который содержит только подстановки и транспозиции, но не их произведения.

Было бы можно, таким образом, определить «чистый» шифр как шифр, в котором T_j образуют группу. Однако это было бы слишком сильным ограничением, так как тогда потребовалось бы, чтобы пространство E совпадало с пространством M, т. е. чтобы система была эндоморфной. Дробная транспозиция так же однородна, как и обычная транспозиция, но она не эндоморфна. Подходящим является следующее определение: шифр T является чистым, если для каждых T_i, T_j, T_k имеется такое T_s, что

$$T_iT_j^{-1}T_k = T_s,$$

и все ключи равновероятны. В противном случае шифр является смешанным. Шифры на рис. 2 являются смешанными, а на рис. 4 — чистыми, если только все ключи равновероятны.

Теорема 1. *В чистом шифре операции $T_i^{-1}T_j$, отображающие пространство сообщений в себя, образуют группу, порядок которой равен m — числу различных ключей.*

Так как

$$T_j^{-1}T_kT_k^{-1}T_j = I,$$

то каждый элемент имеет обратный. Ассоциативный закон верен, так как это операции, а групповое свойство следует из того, что

$$T_i^{-1}T_jT_k^{-1}T_l = T_s^{-1}T_kT_k^{-1}T_l = T_s^{-1}T_l,$$

где предполагалось, что $T_i^{-1}T_j = T_s^{-1}T_k$ для некоторого s.

Операция $T_i^{-1}T_j$ означает шифрование сообщения с помощью ключа j с последующим дешифрованием с помощью ключа i, что приводит нас назад к пространству сообщений. Если система T эндоморфна, т. е. T_i отображают пространство Ω_M в само себя (что имеет место для большинства шифров, в которых и пространство сообщений, и пространство криптограмм состоит из последовательностей букв), и если T_i образуют группу и равновероятны, то T — чистый шифр, так как

$$T_iT_j^{-1}T_k = T_iT_r = T_s.$$

Теорема 2. *Произведение двух чистых коммутирующих шифров является чистым шифром.*

Если T и R коммутируют, то $T_iR_j = R_lT_m$ для любых i, j при соответствующих l, m. Тогда

$$T_iR_j(T_kR_l)^{-1}T_mR_n = T_iR_jR_l^{-1}T_k^{-1}T_mR_n = R_uR_v^{-1}R_wT_rT_s^{-1}T_t = R_hT_g.$$

Условие коммутирования не является, однако, необходимым для того, чтобы произведение было чистым шифром.

Система, состоящая из одного ключа, т. е. из единственной определенной операции T_1, является чистым шифром, т. е. при единственном возможном выборе индексов имеем

$$T_1T_1^{-1}T_1 = T_1.$$

Таким образом, разложение шифра в сумму таких простых отображений представляет собой разложение в его сумму чистых шифров.

Исследование примера, приведенного на рис. 4, вскрывает некоторые свойства чистого шифра. Сообщения распадаются на определенные подмножества, которые мы будем называть *остаточными классами*, и возможные криптограммы также распадаются на соответствующие им *остаточные классы*. От каждого сообщения в любом классе к каждой криптограмме в соответствующем классе имеется не менее одной линии и нет линий между несоответствующими классами. Число сообщений в классе является делителем полного числа ключей. Число «параллельных» линий от сообщения M к криптограмме в соответствующем классе равно числу ключей, деленному на число сообщений в классе, содержащем это сообщение (или криптограмму).

В приложении[1)] показывается, что это верно для чистых шифров и в общем случае. Резюмируя сказанное, мы имеем

Теорема 3. *В чистой системе сообщения можно разделить на множество «остаточных классов» $C_1, \ldots, C_s$, а криптограммы — на соответствующее множество остаточных классов $C'_1, \ldots, C'_s$. Эти классы будут иметь следующие свойства:*

1. Остаточные классы сообщений взаимно исключают друг друга и содержат все возможные сообщения. Аналогичное утверждение верно и для остаточных классов криптограмм.

2. Если зашифровать любое сообщение из класса C_i с помощью любого ключа, то получится криптограмма из класса C'_i. Дешифрирование любой криптограммы из класса C'_i с помощью любого ключа приводит к сообщению из класса C_i.

[1)] Имеется в виду приложение к полному тексту работы К. Шеннона. — *прим. ред.*

3. Число сообщений в классе C_i, скажем, φ_i, равно числу криптограмм в классе C'_i и является делителем k — числа ключей.

4. Каждое сообщение из класса C_i может быть зашифровано в каждую криптограмму из класса C'_i при помощи точно k/φ_i различных ключей. То же самое верно и для дешифрирования.

Смысл понятия чистый шифр (и причина для выбора такого термина) лежит в том, что в чистом шифре все ключи являются по существу одинаковыми. Какой бы ключ ни использовался для заданного сообщения, апостериорные вероятности всех сообщений будут теми же самыми. Чтобы показать это, заметим, что два различных ключа, примененных к одному сообщению, дадут в результате две криптограммы из одного остаточного класса, скажем C'_i. Поэтому эти две криптограммы могут быть расшифрованы с помощью k/φ_i ключей в каждое из сообщений в классе C_i и больше ни в какие возможные сообщения. Так как все ключи равновероятны, то апостериорные вероятности различных сообщений равны

$$P_E(M) = \frac{P(M) \cdot P_M(E)}{P(E)} = \frac{P(M) \cdot P_M(E)}{\Sigma_M P(M) P_M(E)} = \frac{P(M)}{P(C_i)},$$

где M — сообщение из класса C_i, E — криптограмма из класса C'_i и сумма берется по всем M из класса C_i. Если E и M не принадлежат соответствующим остаточным классам, то $P_E(M) = 0$.

Аналогично можно показать, что набор апостериорных вероятностей различных ключей всегда одинаков, но эти вероятности ставятся в соответствие ключам лишь после того, как уже использован некоторый ключ. При изменении частного ключа это множество чисел $P_E(K)$ подвергается перестановке. Иными словами, имеем:

Теорема 4. *В чистой системе апостериорные вероятности различных сообщений $P_E(M)$ не зависят от выбора ключа. Апостериорные вероятности ключей $P_E(K)$ образуют один и тот же набор величин, но подвергаются перестановке в результате различных выборов ключа.*

Грубо говоря, можно считать, что любой выбор ключа в чистом шифре приводит к одинаковым трудностям при дешифрировании. Поскольку различные ключи все приводят к формированию криптограмм из одного и того же остаточного класса, то все криптограммы из одного остаточного класса эквивалентны с точки зрения сложности дешифрирования — они приводят к тем же самым апостериорным вероятностям сообщений и, если учитывать перестановки, к тем же самым вероятностям ключей.

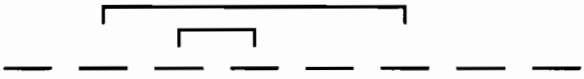
В качестве примера чистого шифра может служить простая подстановка с равновероятными ключами. Остаточный класс, соответствующий данной криптограмме E, является множеством всех криптограмм, которые могут быть получены из E с помощью операций $T_j T_k^{-1} E$. В рассматриваемом случае операция $T_j T_k^{-1}$ сама является подстановкой и поэтому любая подстановка переводит криптограмму E в другой член того же самого остаточного класса; таким образом, если криптограмма представляет собой

$$E = XCPPGCFQ,$$

то

$$E_1 = RDHHGDSN,$$
$$E_2 = ABCCDBEF \qquad \text{и т. д.}$$

принадлежат к тому же остаточному классу. В этом случае очевидно, что криптограммы по существу эквивалентны. Все существенное в простой подстановке со случайным ключом заключено в характере повторения букв, в то время как сами буквы являются несущественной маскировкой. В действительности можно бы полностью обойтись без них, указав характер повторений букв в E следующим образом:



Это обозначение описывает остаточный класс, но устраняет всю информацию относительно конкретных членов этого класса; таким образом, оно представляет как раз ту информацию, которая имеет значение для шифровальщика противника. Это связано с одним из методов подхода к раскрытию шифров типа простой подстановки методом характерных слов.

В шифре типа Цезаря имеют значение только первые разности криптограммы по модулю 26. Две криптограммы с теми же самыми разностями (Δe_i) принадлежат к одному остаточному классу. Этот шифр можно раскрыть путем простого процесса выписывания двадцати шести сообщений из этого остаточного класса и выбора того из них, которое имеет смысл.

Шифр Виженера с периодом d со случайным ключом представляет собой другой пример чистого шифра. Здесь остаточный класс сообщений состоит из всех последовательностей с теми же первыми разностями, что и у криптограммы для букв, отстоящих на расстояние d. Для

$d = 3$ остаточный класс определяется с помощью равенств

$$\begin{array}{rcl} m_1 - m_4 & = & e_1 - e_4 \\ m_2 - m_5 & = & e_2 - e_5 \\ m_3 - m_6 & = & e_3 - e_6 \\ m_4 - m_7 & = & e_4 - e_7 \\ \dots & \dots & \dots \end{array}$$

где $E = e_1 e_2 \dots$ — криптограмма, а $m_1 m_2 \dots$ является любым сообщением M в соответствующем остаточном классе.

В транспозиции с периодом d со случайным ключом остаточный класс состоит из всех способов расстановок символов криптограммы, в которых никакое e_i не выдвигается из своего блока длины d и любые два e_i с расстоянием d остаются на таком же расстоянии. Это используется для раскрытия шифра следующим образом: криптограмма записывается в виде последовательных блоков длины d один под другим, как показано ниже (для $d = 5$)

$$\begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \\ e_6 & e_7 & e_8 & e_9 & e_{10} \\ e_{11} & e_{12} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \end{array}$$

Затем столбцы переставляются до тех пор, пока не получится осмысленный текст. После того как криптограмма разбита на столбцы, оставшейся существенной информацией является только остаточный класс криптограммы.

Теорема 5. *Если шифр T — чистый, то $T_i T_j^{-1} T = T$, где T_i, T_j — любые два отображения из T. Обратно, если это выполняется для любых принадлежащих шифру T_i, T_j, то шифр T является чистым.*

Первая часть этой теоремы следует, очевидно, из определения чистого шифра. Чтобы доказать вторую часть, заметим сначала, что если $T_i T_j^{-1} T = T$, то $T_i T_j^{-1} T_s$ является отображением из T. Остается показать, что все ключи равновероятны. Имеем $T = \sum_s p_s T_s$ и

$$\sum_s p_s T_i T_j^{-1} T_s = \sum_s p_s T_s.$$

Слагаемое в стоящей слева сумме с $s = j$ дает $p_j T_i$. Единственным слагаемым с T_i в правой части является $p_i T_i$. Так как все коэффициенты неотрицательны, то отсюда следует, что

$$p_j \leqslant p_i.$$

То же самое рассуждение остается справедливым, если i и j поменять местами. Следовательно,

$$p_i = p_j$$

и T — чистый шифр. Таким образом, условие $T_i T_j^{-1} T = T$ можно было бы использовать в качестве другого определения чистого шифра.

8. Подобные системы

Две секретные системы R и S будем называть подобными, если существует отображение A, имеющее обратное A^{-1}, такое, что

$$R = AS.$$

Это означает, что шифрование с помощью R даст то же, что шифрование с помощью S с последующим применением отображения A. Если использовать запись $R \approx S$ для обозначения того, что R подобно S, то, очевидно, из $R \approx S$ следует $S \approx R$. Кроме того, из $R \approx S$ и $S \approx T$ следует, что $R \approx T$ и, наконец, $R \approx R$. Резюмируя вышеизложенное, можно сказать, что подобие систем является соотношением эквивалентности.

Криптографический смысл подобия состоит в том, что если $R \approx S$, то R и S — эквивалентны с точки зрения дешифрирования. Действительно, если шифровальщик противника перехватывает криптограмму из системы S, он может перевести ее в криптограмму из системы R простым применением к ней отображения A. Обратно, криптограмма из системы R переводится в криптограмму из системы S с помощью A^{-1}. Если R и S применяются к одному и тому же пространству сообщений или языку, то имеется взаимооднозначное соответствие между получающимися криптограммами. Соответствующие друг другу криптограммы дают одинаковое апостериорное распределение вероятностей для всех сообщений.

Если имеется некоторый способ раскрытия системы R, то любая система S, подобная R, может быть раскрыта после приведения ее к R с помощью операции A. Этот способ часто используется на практике.

В качестве тривиального примера рассмотрим простую подстановку, в которой буквы сообщения заменяются не буквами, а произвольными символами. Она подобна обычной простой подстановке с заменой на буквы. Вторым примером могут служить шифр Цезаря и обратный шифр Цезаря. Последний иногда раскрывают, переводя его сначала в шифр Цезаря. Это можно сделать, обратив алфавит в криптограмме. Шифры Виженера, Бофора и вариант Бофора все подобны, если ключ является случайным. Шифр с «автоключом» (т. е. сообщением, используемым в качестве «ключа») с используемыми вначале ключами

$K_1K_2\ldots K_d$ подобен шифру Виженера с ключом, поочередно складываемым и вычитаемым по модулю 26. Отображение A в этом случае представляет собой «дешифровку» автоключа с помощью последовательности из d таких отображений для каждого из начальных ключей.

Часть II
ТЕОРЕТИЧЕСКАЯ СЕКРЕТНОСТЬ

9. Введение

Рассмотрим вопросы, связанные с «теоретической секретностью» систем. Насколько устойчива некоторая система, если шифровальщик противника не ограничен временем и обладает всеми необходимыми средствами для анализа криптограмм? Имеет ли криптограмма единственное решение (даже если для нахождения этого решения может потребоваться такой объем работ, что его практически нельзя будет выполнить), а если нет, то сколько она имеет приемлемых решений? Какой объем текста, зашифрованного в данной системе, нужно перехватить для того, чтобы решение стало единственным? Существуют ли секретные системы, в которых вообще нельзя найти единственного решения независимо от того, каков объем перехваченного зашифрованного текста? Существуют ли секретные системы, в которых противник не получает никакой информации, сколько бы он ни перехватывал зашифрованного текста? В анализе этих вопросов найдут широкое применение понятия энтропии, избыточности, а также и другие понятия, введенные в работе «Математическая теория связи»[1].

10. Совершенная секретность

Предположим, что имеется конечное число возможных сообщений $M_1,\ldots,M_n$ с априорными вероятностями $P(M_1),\ldots,P(M_n)$ и что эти сообщения преобразуются в возможные криптограммы $E_1,\ldots,E_m$, так что

$$E = T_i M.$$

После того как шифровальщик противника перехватил некоторую криптограмму E, он может вычислить, по крайней мере в принципе, апостериорные вероятности различных сообщений $P_E(M)$. Естественно определить *совершенную секретность* с помощью следующего условия: для всех E апостериорные вероятности равны априорным вероят-

[1] Перевод этой работы см. в книге К. Шеннон «Работы по теории информации и кибернетике», М., ИЛ, 1963, с. 243–332. — *Прим. ред.*

ностям независимо от величины этих последних. В этом случае перехват сообщения не дает шифровальщику противника никакой информации[1]. Теперь он не может корректировать никакие свои действия в зависимости от информации, содержащейся в криптограмме, так как все вероятности, относящиеся к содержанию криптограммы, не изменяются. С другой стороны, если это условие равенства вероятностей не выполнено, то имеются такие случаи, в которых для определенного ключа и определенных выборов сообщений апостериорные вероятности противника отличаются от априорных. А это в свою очередь может повлиять на выбор противником своих действий и, таким образом, совершенной секретности не получится. Следовательно, приведенное определение неизбежным образом следует из нашего интуитивного представления о совершенной секретности.

Необходимое и достаточное условие для того, чтобы система была совершенно секретной, можно записать в следующем виде. По теореме Байеса

$$P_E(M) = \frac{P(M) \cdot P_M(E)}{P(E)},$$

где

$P(M)$ — априорная вероятность сообщения M;

$P_M(E)$ — условная вероятность криптограммы E при условии, что выбрано сообщение M, т. е. сумма вероятностей всех тех ключей, которые переводят сообщение M в криптограмму E;

$P(E)$ — вероятность получения криптограммы E;

$P_E(M)$ — апостериорная вероятность сообщения M при условии, что перехвачена криптограмма E.

Для совершенной секретности системы величины $P_E(M)$ и $P(M)$ должны быть равны для всех E и M. Следовательно, должно быть выполнено одно из равенств: или $P(M) = 0$ [это решение должно быть отброшено, так как требуется, чтобы равенство осуществлялось при любых значениях $P(M)$], или же

$$P_M(E) = P(E)$$

для любых M и E.

Наоборот, если $P_M(E) = P(E)$, то

$$P_E(M) = P(M),$$

[1] Пурист мог бы возразить, что противник получил некоторую информацию, а именно он знает, что послано какое-то сообщение. На это можно ответить следующим образом. Пусть среди сообщений имеется «чистый бланк», соответствующий «отсутствию сообщения». Если не создается никакого сообщения, то чистый бланк зашифровывается и посылается в качестве криптограммы. Тогда устраняется даже эта крупинка информации.

и система совершенно секретна. Таким образом, можно сформулировать следующее:

Теорема 6. *Необходимое и достаточное условие для совершенной секретности состоит в том, что*

$$P_M(E) = P(E)$$

для всех M и E, т. е. $P_M(E)$ не должно зависеть от M.

Другими словами, полная вероятность всех ключей, переводящих сообщение M_i в данную криптограмму E, равна полной вероятности всех ключей, переводящих сообщение M_j в ту же самую криптограмму E для всех M_i, M_j и E.

Далее, должно существовать по крайней мере столько же криптограмм E, сколько и сообщений M, так как для фиксированного i отображение T_i дает взаимнооднозначное соответствие между всеми M и некоторыми из E. Для совершенно секретных систем для каждого из этих E и любого M $P_M(E) = P(E) \neq 0$. Следовательно, найдется по крайней мере один ключ, отображающий данное M в любое из E. Но все ключи, отображающие фиксированное M в различные E, должны быть различными, и *поэтому число различных ключей не меньше числа сообщений M*. Как показывает следующий пример, можно получить совершенную секретность, когда число сообщений точно равно числу ключей. Пусть M_i занумерованы числами от 1 до n, так же как и E_i, и пусть используются n ключей. Тогда

$$T_i M_j = E_s,$$

где $s = i + j \pmod n$. В этом случае оказывается справедливым равенство $P_E(M) = \frac{1}{n} = P(E)$ и система является совершенно секретной. Один пример такой системы показан на рис. 5, где

$$s = j + i - 1 \pmod 5.$$

Совершенно секретные системы, в которых число криптограмм равно числу сообщений, а также числу ключей, характеризуются следующими двумя свойствами: 1) каждое M связывается с каждым E только одной линией; 2) все ключи равновероятны. Таким образом, матричное представление такой системы является «латинским квадратом».

В «Математической теории связи» показано, что количественно информацию удобно измерять с помощью энтропии. Если имеется некоторая совокупность возможностей с вероятностями $p_1, \ldots, p_n$, то энтропия дается выражением

$$H = -\sum p_i \log p_i.$$

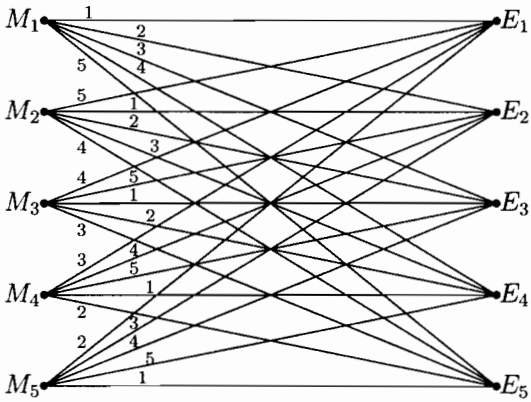


Рис. 5. *Совершенная система.*

Секретная система включает в себя два статистических выбора: выбор сообщения и выбор ключа. Можно измерять количество информации, создаваемой при выборе сообщения, через $H(M)$

$$H(M) = -\sum P(M) \log P(M),$$

где суммирование выполняется по всем возможным сообщениям. Аналогично, неопределенность, связанная с выбором ключа, дается выражением

$$H(K) = -\sum P(K) \log P(K).$$

В совершенно секретных системах описанного выше типа количество информации в сообщении равно самое большее $\log n$ (эта величина достигается для равновероятных сообщений). Эта информация может быть скрыта полностью лишь тогда, когда неопределенность ключа не меньше $\log n$. Это является первым примером общего принципа, который будет часто встречаться ниже: существует предел, которого нельзя превзойти при заданной неопределенности ключа — количество неопределенности, которое может быть введено в решение, не может быть больше чем неопределенность ключа.

Положение несколько усложняется, если число сообщений бесконечно. Предположим, например, что сообщения порождаются соответствующим марковским процессом в виде бесконечной последовательности букв. Ясно, что никакой конечный ключ не даст совершенной секретности. Предположим тогда, что источник ключа порождает ключ аналогичным образом, т. е. как бесконечную последовательность символов.

Предположим далее, что для шифрования и дешифрирования сообщения длины L_M требуется только определенная длина ключа L_K. Пусть логарифм числа букв в алфавите сообщений будет R_M, а такой же логарифм для ключа — R_K. Тогда из рассуждений для конечного случая, очевидно, следует, что для совершенной секретности требуется, чтобы выполнялось неравенство

$$R_M L_M \leqslant R_K L_K.$$

Такой вид совершенной секретности реализован в системе Вернама.

Эти выводы делаются в предположении, что априорные вероятности сообщений неизвестны или произвольны. В этом случае ключ, требуемый для того, чтобы имела место совершенная секретность, зависит от полного числа возможных сообщений.

Можно было бы ожидать, что если в пространстве сообщений имеются фиксированные известные статистические связи, так что имеется определенная скорость создания сообщений R в смысле, принятом в «Математической теории связи», то необходимый объем ключа можно было бы снизить в среднем в R/R_M раз, и это действительно верно. В самом деле, сообщение можно пропустить через преобразователь, который устраняет избыточность и уменьшает среднюю длину сообщения как раз во столько раз. Затем к результату можно применить шифр Вернама. Очевидно, что объем ключа, используемого на букву сообщения, статистически уменьшается на множитель R/R_M, и в этом случае источник ключа и источник сообщений в точности согласован — один бит ключа полностью скрывает один бит информации сообщения. С помощью методов, использованных в «Математической теории связи», легко также показать, что это лучшее, чего можно достигнуть.

Совершенно секретные системы могут применяться и на практике, их можно использовать или в том случае, когда полной секретности придается чрезвычайно большое значение, например, для кодирования документов высших военных инстанций управления или же в случаях, где число возможных сообщений мало. Так, беря крайний пример, когда имеются в виду только два сообщения — «да» или «нет», — можно, конечно, использовать совершенно секретную систему со следующей таблицей отображений:

M K	A	B
да	0	1
нет	1	0

Недостатком совершенно секретных систем для случая корреспонденции большого объема является, конечно, то, что требуется посылать эквивалентный объем ключа. В следующих разделах будет рассмотрен

вопрос о том, чего можно достигнуть при помощи меньших объемов ключа, в частности, с помощью конечного ключа.

11. Ненадежность

Предположим теперь, что для английского текста используется шифр простой подстановки и что перехвачено определенное число, скажем N, букв зашифрованного текста. Если N достаточно велико, скажем более 50, то почти всегда существует единственное решение шифра, т. е. единственная последовательность, имеющая смысл на английском языке, в которую переводится перехваченный материал с помощью простой подстановки. Для меньших N шансы на неединственность решения увеличиваются; для $N = 15$, вообще говоря, будет существовать некоторое число подходящих отрывков осмысленного английского текста, в то время как для $N = 8$ окажется подходящей значительная часть (порядка 1/8) всех возможных значащих английских последовательностей такой длины, так как из восьми букв редко повторится больше чем одна. При $N = 1$, очевидно, возможна любая буква и апостериорная вероятность любой буквы будет равна ее априорной вероятности. Для одной буквы система является совершенно секретной.

Это происходит, вообще говоря, со всеми разрешимыми шифрами. Прежде чем перехвачена криптограмма, можно представить себе априорные вероятности, связанные с различными возможными сообщениями, а также с различными ключами. После того как материал перехвачен, шифровальщик противника вычисляет их апостериорные вероятности. При увеличении числа N вероятности некоторых сообщений возрастают, но для большинства сообщений они убывают до тех пор, пока не останется только одно сообщение, имеющее вероятность, близкую к единице, в то время как полная вероятность всех других близка к нулю.

Для самых простых систем эти вычисления можно эффективно выполнить. Таблица I дает апостериорные вероятности для шифра Цезаря, примененного к английскому тексту, причем ключ выбирался случайно из 26 возможных ключей. Для того чтобы можно было использовать обычные таблицы частот букв, диграмм и триграмм, текст был начат в случайном месте (на страницу открытой наугад книги был случайно опущен карандаш). Сообщение, выбранное таким способом, начинается с «creases to» (карандаш опущен на третью букву слова increases). Если известно, что сообщение начинается не с середины, а с начала некоторого предложения, то нужно пользоваться иной таблицей, соответствующей частотам букв, диграмм и триграмм, стоящих в начале предложения.

Таблица I. *Апостериорные вероятности для криптограммы типа Цезаря*

Расшифровки	$N=1$	$N=2$	$N=3$	$N=4$	$N=5$
$CREAS$	0,028	0,0377	0,1111	0,3673	1
$DSFBT$	0,038	0,0314			
$ETGCU$	0,131	0,0881			
$FUNDV$	0,029	0,0189			
$GVIEW$	0,020				
$HWJFX$	0,053	0,0063			
$IXKGY$	0,063	0,0126			
$JYLHZ$	0,001				
$KZMIA$	0,004				
$LANJB$	0,034	0,1321	0,2500		
$MBOKC$	0,025		0,0222		
$NCPLD$	0,071	0,1195			
$ODQME$	0,080	0,0377			
$PERNF$	0,020	0,0818	0,4389	0,6327	
$QFSOG$	0,001				
$RGTPH$	0,068	0,0126			
$SHUQI$	0,061	0,0881	0,0056		
$TIVRJ$	0,105	0,2830	0,1667		
$UJWSK$	0,025				
$VKXTL$	0,009				
$WLYUM$	0,015		0,0056		
$XMZVN$	0,002				
$YNAWO$	0,020				
$ZOBXP$	0,001				
$APCYQ$	0,082	0,0503			
$BQDZR$	0,014				
H(десятичных единиц)	1,2425	0,9686	0,6034	0,285	0

Шифр Цезаря со случайным ключом является чистым, и выбор частного ключа не влияет на апостериорные вероятности. Чтобы определить эти вероятности, надо просто выписать возможные расшифровки с помощью всех ключей и вычислить их априорные вероятности. Апостериорные вероятности получатся из этих последних в результате деления их на их сумму. Эти возможные расшифровки, образующие остаточный класс этого сообщения, найдены с помощью стандартного процесса последовательного «пробегания алфавита», в таблице I они даны слева. Для одной перехваченной буквы апостериорные вероятности равны априорным вероятностям для всех букв[1)] (они приведены в таблице под рубрикой $N = 1$).

Для двух перехваченных букв эти вероятности равны априорным вероятностям диграмм, пронормированным на их сумму (они приведены в столбце $N = 2$). Триграммные частоты получены аналогично и приведены в столбце $N = 3$. Для четырех- и пятибуквенных последовательностей вероятности находились из триграммных частот с помощью умножения, так как с некоторым приближением

$$p(ijkl) = p(ijk)p_{jk}(l).$$

Заметим, что для трех букв число возможных сообщений снижается до четырех сообщений достаточно высокой вероятности, причем вероятности всех других сообщений малы по сравнению с вероятностями этих четырех сообщений. Для четырех букв имеются два возможных сообщения и для пяти — только одно, а именно правильная дешифровка.

В принципе это может быть проведено для любой системы, однако в том случае, когда объем ключа не очень мал, число возможных сообщений настолько велико, что вычисления становятся практически невыполнимыми.

Получаемое таким образом множество апостериорных вероятностей описывает, как постепенно, по мере получения зашифрованного материала, становятся более точными сведения шифровальщика противника относительно сообщения и ключа.

Это описание, однако, является слишком исчерпывающим и слишком сложным для наших целей. Хотелось бы иметь упрощенное описание такого приближения к единственности возможного решения.

Аналогичная ситуация возникает в теории связи, когда передаваемый сигнал искажается шумом. Здесь необходимо ввести подходящую меру неопределенности того, что действительно было передано, при

[1)] Вероятности в приводимой таблице были взяты из таблиц частот, данных в книге Pratt F., Secret and Urgent, Blue Ribbon Books, New York, 1939. Хотя эти таблицы и не являются полными, но для настоящих целей их достаточно.

условии, что известен только искаженный шумом вариант — принятый сигнал.

В «Математической теории связи» показано, что естественной математической мерой этой неопределенности является условная энтропия передаваемого сигнала при условии, что принятый сигнал известен. Эта условная энтропия для удобства будет называться ненадежностью.

С криптографической точки зрения секретная система почти тождественна системе связи при наличии шума. На сообщение (передаваемый сигнал) действует некоторый статистический элемент (секретная система с ее статистически выбранным ключом). В результате получается криптограмма (аналог искаженного сигнала), подлежащая дешифрированию. Основное различие заключается в следующем: во-первых, в том, что преобразование при помощи шифра имеет обычно более сложную природу, чем возникающее за счет шума в канале; и, во-вторых, ключ в секретной системе обычно выбирается из конечного множества, в то время как шум в канале чаще является непрерывным, выбранным по существу из бесконечного множества.

Учитывая эти соображения, естественно использовать ненадежность в качестве теоретической меры секретности. Следует отметить, что имеются две основные ненадежности: ненадежность ключа и ненадежность сообщения. Они будут обозначаться через $H_E(K)$ и $H_E(M)$ соответственно. Их величины определяются соотношениями

$$H_E(K) = -\sum_{E,K} P(E,K) \log P_E(K),$$

$$H_E(M) = -\sum_{E,M} P(E,M) \log P_E(M),$$

где E, M и K — криптограмма, сообщение и ключ;

$P(E,K)$ — вероятность ключа K и криптограммы E;

$P_E(K)$ — апостериорная вероятность ключа K, если перехвачена криптограмма E;

$P(E,M)$ и $P_E(M)$ — аналогичные вероятности, но не для ключа, а для сообщения.

Суммирование в $H_E(K)$ проводится по всем возможным криптограммам определенной длины (скажем, N) и по всем возможным ключам. Для $H_E(M)$ суммирование проводится по всем сообщениям и криптограммам длины N. Таким образом, $H_E(K)$ и $H_E(M)$ являются функциями от N — числа перехваченных букв. Это будет иногда указываться в обозначении так: $H_E(K,N)$, $H_E(M,N)$. Заметим, что эти ненадежности являются «полными», т. е. не делятся на N с тем,

чтобы получить скорость ненадежности, которая рассматривалась в работе «Математическая теория связи».

Те же самые рассуждения, которые были использованы в «Математической теории связи» для обоснования введения ненадежности в качестве меры неопределенности в теории связи, применимы и здесь. Так, из того, что ненадежность равна нулю, следует, что одно сообщение (или ключ) имеет единичную вероятность, а все другие — нулевую. Этот случай соответствует полной осведомленности шифровальщика. Постепенное убывание ненадежности с ростом N соответствует увеличению сведений об исходном ключе или сообщении. Кривые ненадежности сообщения и ключа, нанесенные на график как функции от N, мы будем называть характеристиками ненадежности рассматриваемой секретной системы.

Величины $H_E(K, N)$ и $H_E(M, N)$ для криптограммы шифра Цезаря, рассмотренной выше, сосчитаны и приведены в нижней строке табл. I. Числа $H_E(K, N)$ и $H_E(M, N)$ в этом случае равны и даны в десятичных единицах (т. е. при вычислениях в качестве основания логарифма бралось 10). Следует отметить, что ненадежность здесь сосчитана для частной криптограммы, так как суммирование ведется только по M (или K), но не по E. В общем случае суммирование должно было бы проводиться по всем перехваченным криптограммам длины N, в результате чего получилась бы средняя неопределенность. Вычислительные трудности не позволяют сделать это практически.