МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ»

В.Г. Спицын

Методические указания к выполнению самостоятельных работ по дисциплине “Защита информации”

Томск 2012

1. ПРОБЛЕМЫ И МЕТОДЫ ЗАЩИТЫ КОМПЬЮТЕРНОЙ ИНФОРМАЦИИ

1.1. Информационная безопасность

В настоящее время происходит быстрое развитие средств вычислительной техники и сетей передачи данных, позволяющее на основе оперативного обмена информацией создавать новые направления в науке, банковской, управленческой и предпринимательской деятельности. Проникновение компьютерных технологий во все сферы человеческой жизни привело к необходимости разработки надёжных способов решения проблемы защиты информации, циркулирующей в компьютерных системах, от несанкционированного доступа.

Под **информационной безопасностью** понимается состояние защищенности обрабатываемых, хранимых и передаваемых в информационно-телекоммуникационных системах (ИТС) данных от незаконного ознакомления, преобразования и уничтожения, а также состояние защищенности информационных ресурсов от воздействий, направленных на нарушение их работоспособности.

Основными задачами защиты пользовательской информации являются:

- обеспечение конфиденциальности информации;
- обеспечение целостности информации;
- обеспечение достоверности информации;
- обеспечение оперативности доступа к информации;
- обеспечение юридической значимости информации, представленной в виде электронного документа;
- обеспечение неотслеживаемости действий клиента.

Конфиденциальность – свойство информации быть доступной ограниченному кругу пользователей информационной системы, в которой циркулирует данная информация.

Целостность – свойство информации или программного обеспечения сохранять свою структуру и/или содержание в процессе передачи и/или хранения.

Достоверность – свойство информации, выражающееся в строгой принадлежности объекту, который является её источником, либо тому объекту, от которого эта информация принята.

Оперативность – способность информации быть доступной для конечного пользователя в соответствии с его временными потребностями.

Юридическая значимость указывает на тот факт, что документ обладает юридической силой. Данное свойство информации особенно актуально в системах электронных платежей.

Неотслеживаемость – способность совершать некоторые действия в информационной системе незаметно для других объектов. Например, для авторизации доступа к электронной платежной системе пользователь должен предоставить некоторые сведения, однозначно его идентифицирующие. По мере развития электронных платежных систем может появиться реальная опасность, что, например, все платежные операции будут контролироваться, т.е. возникнут условия для тотальной слежки за пользователями информационных систем.

Существует несколько путей решения проблемы неотслеживаемости:

- запрещение с помощью законодательных актов всякой тотальной слежки за пользователями информационных систем;
- применение криптографических методов для поддержания неотслеживаемости.

Радикальное решение проблем защиты информации и обеспечения безопасности в ИТС может быть получено только на базе использования криптографических методов и средств защиты информации.

1.2. Проблемы защиты информации в компьютерных системах

Специфическая особенность защиты информации в компьютерных системах заключается в том, что информация не является “жёстко связанной” с носителем, а может легко и быстро копироваться и передаваться по каналам связи.

Криптографические преобразования данных, основанные на современных скоростных методах, являются наиболее эффективным способом обеспечения конфиденциальности данных, их целостности и подлинности, а в сочетании с необходимыми техническими и организационными мероприятиями могут обеспечить защиту от широкого спектра потенциальных угроз.

Слово **криптография** состоит из 2-х частей и в переводе с греческого языка означает “тайнопись”:

крипто – kryhtas – тайный, скрытый;

графия – grapho – пишу (причем grapho может быть истолковано как: а) графическое воспроизведение чего-либо, например, стенография; б) описание чего-либо, например, география).

Криптография исторически появилась в связи с необходимостью передачи секретной информации. Первоначально её целью являлась разработка специальных методов преобразования информации в форму недоступную для потенциального злоумышленника. С широкомасштабным применением электронных способов передачи и обработки информации задачи криптографии расширились. Современные проблемы криптографии включают разработку систем электронной цифровой подписи и тайного электронного голосования, идентификации удаленных пользователей, методов защиты от навязывания ложных сообщений и др.

В криптографии рассматривается некоторый **злоумышленник** (оппонент, криптоаналитик противника, нарушитель, нелегальный пользователь), который осведомлен об используемых криптографических алгоритмах, протоколах и пытается скомпрометировать их. **Компрометация** криптосистемы может заключаться, например, в несанкционированном чтении информации, формировании чужой подписи, изменении результатов голосования, нарушении тайны голосования, необнаруживаемом модифицировании данных.

Разнообразные действия оппонента в общем случае называются **криптоаналитической атакой. Специфика криптографии** состоит в том, что она направлена на разработку методов, обеспечивающих стойкость к любым действиям злоумышленника. При этом следует отметить, что на момент разработки криптосистемы невозможно предусмотреть случаи атаки, которые могут быть изобретены в будущем на основе новых научных достижений.

Надёжность решения криптографической проблемы (т.е. оценка трудоёмкости атаки на криптосистему) является самостоятельным предметом исследования, называемым **криптоанализом.**

Криптография и криптоанализ составляют единую область знания – **криптологию** (**логия** – logos, в переводе с греческого – понятие, учение, наука, знание).

Необходимость все более частого применения методов криптографической защиты информации привела к возникновению одной из важных социально-этических проблем – противоречию между желанием пользователей защитить свою информацию и передачу сообщений и желанием специальных государственных служб иметь возможность доступа к информации некоторых других организаций и отдельных лиц с целью пресечения незаконной деятельности. В связи с этим сформи-

ровался широкий спектр мнений, касающийся вопроса регламентации использования алгоритмов шифрования – от полного запрета широкого применения криптографических методов до полной свободы их использования. Некоторые предложения относятся к разрешению использования только ослабленных алгоритмов или к установлению порядка обязательной регистрации ключей шифрования.

Ограничение исследовательских работ в области криптографии может привести к отрицательному эффекту. Возникновение глобальных информационных сетей типа Internet и анализ опыта их применения чётко выявляют слабость традиционных механизмов защиты информации и отставание в применении современных методов. Для обеспечения безопасности информации в Internet в настоящее время активно ведутся работы по внедрению необходимых новейших криптографических механизмов в эту сеть.

Независимо от прогресса в области разработки методов шифрования, государство всегда имеет возможность законодательно потребовать, чтобы все пользователи регистрировали свои ключи (или необходимую долю ключевой информации) в специально созданных учреждениях. В этом случае контроль информации обеспечивается независимо от уровня стойкости используемых алгоритмов.

Сдерживание открытых исследований в области криптографии упрощает некоторые проблемы спецслужб, однако в целом для государства отрицательный эффект велик, поскольку оно приводит к неизбежному отставанию в области разработки современных систем защиты информации и распространенности компьютерных преступлений.

В России и других странах СНГ в начале 1990-х годов наблюдалась тенденция опережения расширения масштабов и областей применения информационных технологий над развитием систем защиты данных. Такая ситуация является закономерной, поскольку сначала должна возникнуть практическая проблема, а затем будут найдены пути её решения.

Использование систем защиты информации зарубежного производства для борьбы с компьютерными преступлениями не могут выправить этот перекос, так как поступающие на рынок России продукты этого типа не соответствуют современным требованиям из-за существующих экспортных ограничений, принятых в США – основном производителе средств защиты информации.

Сертификаты иностранных фирм и организаций не могут быть заменой отечественным. Сам факт использования зарубежного системного и прикладного программного обеспечения создает повышенную потенциальную угрозу информационным ресурсам. Применение ино-

странных средств защиты без анализа соответствия выполняемым функциям и уровня обеспечиваемой защиты может многократно усложнить ситуацию.

1.3. Традиционные вопросы криптографии

Простейшие криптографические методы известны с древнейших времён и длительное время рассматривались как некоторое ухищрение, а не строгая научная дисциплина.

Классической задачей криптографии является обратимое преобразование некоторого понятного исходного текста (открытого текста) в кажущуюся случайной последовательность некоторых знаков, называемую **шифротекстом** или **криптограммой**. При этом шифротекст может содержать как новые, так и имеющиеся в открытом сообщении знаки. Количество знаков в криптограмме и в исходном тексте в общем случае может различаться. Обязательным требованием является то, что, используя некоторые логические замены символов в шифротексте, можно однозначно и в полном объеме восстановить исходный текст. Надежность сохранения информации в тайне определялась в прошлые времена тем, что в секрете держался сам метод преобразования.

Секретность алгоритма не может обеспечить безусловной стойкости, т.е. невозможности чтения криптограммы тем, кто обладает бесконечными вычислительными ресурсами. Поскольку секретные алгоритмы не могут быть проверены широкомасштабными криптоаналитическими исследованиями, то имеется значительно более высокая (по сравнению с открытыми алгоритмами) вероятность того, что будут найдены уязвимые места и эффективные способы доступа к зашифрованной информации.

В связи с этим в настоящее время наиболее широко используются открытые алгоритмы, прошедшие длительное тестирование и обсуждение в открытой криптографической литературе.

Стойкость современных криптосистем основывается не на секретности алгоритма, а на секретности некоторой информации сравнительно малого размера, называемой **ключом**. Ключ используется для управления процессом криптографического преобразования (шифрования) и является легко сменяемым элементом криптосистемы. Ключ может быть заменен пользователем в произвольный момент времени. В то же время сам алгоритм шифрования является долговременным элементом криптосистемы.

Следует отметить, что при прочих равных условиях секретность алгоритма шифрования существенно (при адекватной его реализации)

затрудняет проведение криптоаналитической атаки. Поэтому были предложены современные криптосистемы, в которых непосредственно алгоритм шифрования является легко сменяемым секретным элементом, но в то же время имеется возможность открытого обсуждения стойкости криптосистемы. Это реализуется в гибких криптосистемах, в которых алгоритм шифрования формируется по специальному алгоритму предвычислений (инициализации) под управлением секретного ключа пользователя. Алгоритм инициализации является открытым, а сам алгоритм шифрования – секретным, так же как и ключ шифрования.

В течение ряда веков криптография являлась предметом избранных – жрецов, правителей, крупных военачальников и дипломатов. Использование криптографических методов и способов преодоления шифров противника оказывало существенное воздействие на исход важных исторических событий. Своим превращением в научную дисциплину криптография обязана потребностям практики, порожденным электронной информационной технологией.

Пробуждение интереса к криптографии, явившееся толчком для ее развития, началось в XIX веке с появлением электросвязи. Наряду с развитием криптографических систем совершенствовались и методы, позволяющие восстанавливать исходные сообщения по шифротексту (криптоанализ), что приводило к ужесточению требований к криптографическим алгоритмам.

Голландский криптограф Керкхофф (1835–1903) впервые сформулировал изложенное ниже правило.

Стойкость шифра (т.е. криптосистемы – набора процедур, управляемых некоторой секретной информацией небольшого объема) должна быть обеспечена в том случае, когда криптоаналитику противника известен весь механизм шифрования за исключением секретного ключа – информации, управляющей процессом криптографических преобразований.

Целью этого требования было осознание необходимости испытания разрабатываемых криптосистем в условиях более жёстких по сравнению с теми, в которых мог бы действовать потенциальный нарушитель. Это правило стимулировало появление более качественных шифрующих алгоритмов. Можно сказать, что в нём содержится первый элемент стандартизации в области криптографии. В настоящее время это правило интерпретируется более широко: все долговременные элементы системы защиты считаются известными потенциальному злоумышленнику. В эту формулировку криптосистемы входят как частный случай защиты. В ней предполагается, что все элементы систем защиты подразделяются на две категории – долговременные и легкосменяемые.

К долговременным элементам относятся те элементы систем защиты, для изменения которых требуется вмешательство специалистов или разработчиков.

К легкосменяемым элементам относятся элементы системы, которые предназначены для произвольного модифицирования или модифицирования по заранее заданному правилу, исходя из случайно выбираемых начальных параметров. К легкосменяемым элементам относятся: ключ, пароль, идентификатор и т.п. Таким образом, надлежащий уровень секретности может быть обеспечен только по отношению к легкосменяемым элементам.

При тестировании новых криптосистем особый интерес представляют нападения на основе известного секретного ключа или ключа шифрования. Следует отметить различия между секретным ключом и ключом шифрования. **Секретный ключ** не всегда непосредственно используется для управления процессом шифрования, а часто служит только для выработки расширенного ключа, который используется для шифрования данных. **Ключ, используемый непосредственно для управления процедурами криптографических преобразований, будем называть ключом шифрования**. Очевидно, что существуют шифры, в которых секретный ключ непосредственно используется при шифровании данных, т.е. секретный ключ является одновременно ключом шифрования.

1.4. Современные приложения криптографии

Интенсификация информационных потоков в компьютерных сетях, являющаяся результатом все большей автоматизации передачи и обработки данных, существенно расширяет использование криптографических методов не только для обеспечения секретности данных, но и для решения более широкого круга задач. В настоящее время основными направлениями приложений методов криптографии являются:

1. Защита от несанкционированного чтения (или обеспечение конфиденциальности информации).
2. Защита от навязывания ложных сообщений (умышленных и непреднамеренных).
3. Идентификация законных пользователей.
4. Контроль целостности информации.
5. Аутентификация информации.
6. Электронная цифровая подпись.
7. Системы тайного электронного голосования.

8. Электронная жеребьёвка.
9. Защита от отказа факта приема сообщения.
10. Одновременное подписание контракта.
11. Защита документов и ценных бумаг от подделки.

Поскольку первое направление приложений уже рассматривалось, поясним кратко остальные.

Само по себе шифрование данных далеко не всегда является достаточным для **защиты от навязывания ложного сообщения**. В большинстве случаев законный получатель, проанализировав семантику сообщения, может определить, что криптограмма была модифицирована или подменена. Однако при искажении цифровых данных и в некоторых других случаях обнаружить факт искажения данных по семантике крайне сложно. Одним из способов защиты от навязывания ложного сообщения путем целенаправленного или случайного искажения шифротекста является имитозащита.

Имитозащита – защита от навязывания ложных сообщений путем формирования в зависимости от секретного ключа специальной дополнительной информации, называемой имитовставкой, которая передается вместе с криптограммой. Для вычисления имитовставки используется алгоритм, задающий зависимость имитовставки от каждого бита сообщения, причём возможным является любой из 2-х вариантов: (1) вычисление имитовставки осуществляется по открытому тексту и (2) вычисление имитовставки осуществляется по шифротексту. Чем больше длина имитовставки, тем меньше вероятность того, что искажение шифротекста не будет обнаружено законным получателем.

Идентификация законных пользователей заключается в распознавании пользователей и основывается на том, что законному пользователю известна информация, не доступная для посторонних. Например, пароль – некоторая случайная секретная информация, сформированная пользователем и не хранящаяся в явном виде в ЭВМ. Для того чтобы система защиты могла идентифицировать легальных (санкционированных) пользователей, в памяти ЭВМ хранятся образы их паролей, вычисленные по специальному криптографическому алгоритму. Этот алгоритм реализует одностороннюю функцию $y=F(x)$. Основное требование к односторонней функции состоит в том, чтобы сложность вычисления значения функции по аргументу (по входу) была низкой, а сложность определения значения аргумента, для которого найдено значение функции (значение выхода), была высокой (т.е. практически неосуществимой при использовании всех вычислительных ресурсов человечества).

Контроль целостности информации – это обнаружение любых несанкционированных изменений информации, например, данных или программ, хранимых в компьютере. Имитозащита является одним из способов контроля целостности информации, передаваемой в виде шифротекста (частным случаем). Контроль целостности информации, не являющейся секретной и хранящейся в открытом виде (программы, базы данных и т.п.), основан на выработке по некоторому криптографическому закону кода обнаружения модификации (КОМ). КОМ имеет значительно меньший объём, чем охраняемая от модифицирования информация. Основным требованием к алгоритму вычисления КОМ является задание зависимости значения КОМ от каждого двоичного представления всех символов исходного текста.

Аутентификация информации – установление санкционированным получателем (приемником) того факта, что полученное сообщение послано санкционированным отправителем (передатчиком).

Электронная цифровая подпись (ЭЦП) основывается на двухключевых криптографических алгоритмах, в которых предусматривается использование 2-х ключей – открытого и секретного. Двухключевые криптоалгоритмы позволяют обеспечить строгую доказательность факта составления того или иного сообщения конкретными абонентами (пользователями) криптосистемы. Это основано на том, что только отправитель сообщения, который держит в секрете некоторую дополнительную информацию (секретный ключ), может составить сообщение со специфической внутренней структурой. То, что сообщение имеет структуру, сформированную с помощью секретного ключа, проверяется с помощью открытого ключа (процедура проверки ЭЦП). Вероятность того, что некоторое сообщение, составленное нарушителем, может быть принято за сообщение, подписанное каким-либо абонентом системы ЭЦП, является чрезвычайно низкой, например, равной 10^{-30}. Следует отметить, что открытый ключ формируется на основе секретного ключа, или секретный и открытый ключ вырабатываются одновременно по специальным процедурам, причем определение секретного ключа по открытому является вычислительно сложной задачей.

1.5. Понятие криптографического протокола

Протокол – это совокупность действий (инструкций, команд, вычислений, алгоритмов), выполняемых в заданной последовательности двумя или более субъектами с целью достижения определённого результата. Корректность выполнения протокола зависит от действий каждого субъекта криптосистемы – оператора, сервера, программы для

ЭВМ, органа власти и т.д., осуществляемых по предписанным алгоритмам. Для того чтобы протокол приводил к желаемой цели, необходимо выполнение следующих условий [1]:

- корректность протокола; совокупность действий, предусмотренных протоколом, должна обеспечить получение требуемого результата при всех возможных ситуациях;
- полнота и однозначность определения, – протокол должен специфицировать действия каждого участника для всех возможных ситуаций;
- непротиворечивость; результаты, полученные различными участниками протокола, не должны быть противоречивыми;
- осведомленность и согласие участников протокола; каждый субъект заранее должен знать протокол и все шаги, которые он должен выполнить; все субъекты должны быть согласны играть свою роль.

Криптографические протоколы – это такие протоколы, в которых используются криптографические преобразования данных. Шифрование данных и вычисление односторонних функций представляет собой выполнение определённого, достаточно стойкого, алгоритма, что, однако, не является полной гарантией стойкости протокола. Стойкость криптографического протокола обеспечивается стойкостью используемого криптографического алгоритма именно в условиях конкретного применения.

1.6. Криптография и стеганография

Стеганографией называется техника скрытой передачи или скрытого хранения информации. Целью стеганографии является сокрытие самого факта передачи сообщений. Для этого используются невидимые чернила, невидимые простому глазу пометки у букв, пометки карандашом, плохо заметные отличия в написании букв, микрофотографии и тайники.

В настоящее время стеганографические методы используют распределение по псевдослучайному закону информации в пространстве или времени (применяется также комбинирование этих способов), зашумление и маскирование в некотором сообщении-контейнере или в служебной информации.

Принципиальное отличие криптографии от стеганографии состоит в том, что в ней не скрывается факт передачи сообщений, а скрывается только его содержание (смысл).

Стеганографические методы могут обеспечить высокий уровень защиты информации только в том случае, когда они будут дополнены

предварительным, достаточно стойким, криптографическим преобразованием сообщения.

2. ИСТОРИЧЕСКИЕ ШИФРЫ

2.1. Подстановочные и перестановочные шифры

До появления компьютеров криптография состояла из алгоритмов на символьной основе. Различные криптографические алгоритмы либо заменяли одни символы другими, либо переставляли символы. Лучшие алгоритмы делали и то и другое много раз. В настоящее время алгоритмы стали работать с битами, а не с символами, поэтому размер алфавита изменился с 26 элементов до двух. Большинство хороших криптографических алгоритмов до сих пор комбинирует подстановки и перестановки.

Подстановочным шифром (шифром замены) называется шифр, который каждый символ открытого текста в шифротексте заменяет другим символом. Получатель инвертирует подстановку шифротекста, восстанавливая открытый текст. В классической криптографии существует четыре типа подстановочных шифров:

1. **Простой подстановочный шифр**, или **моноалфавитный шифр**, – это шифр, который каждый символ открытого текста заменяет соответствующим символом шифротекста.
2. **Однозвучный подстановочный шифр** похож на простую подстановочную криптосистему за исключением того, что один символ открытого текста отображается на несколько символов шифротекста. Например, "A" может соответствовать 5, 13, 25 или 56, "B" – 7, 19, 31 или 42 и так далее.
3. **Полиграмный подстановочный шифр –** это шифр, который блоки символов шифрует по группам. Например, "ABA" может соответствовать "RTQ", "ABB" может соответствовать "SLL" и так далее.
4. **Полиалфавитный подстановочный шифр** состоит из нескольких простых подстановочных шифров. Например, могут быть использованы пять различных простых подстановочных шифров; каждый символ открытого текста заменяется с использованием одного конкретного шифра.

Примером простого подстановочного шифра является знаменитый **шифр Цезаря,** в котором каждый символ открытого текста заменяется символом, находящимся тремя символами правее по модулю 26 ("A" заменяется на "D", "B" – на "E", ..., "W" – на "Z", "X" – на "A", "Y" – на "B", "Z" – на "C"). Шифр является простым, так как алфавит шифротек-

ста представляет собой смещенный, а не случайно распределенный алфавит открытого текста.

На основе простого подстановочного шифра работает шифровальная программа ROTI3 , обычно поставляемая с системами UNIX. В этом шифре буква "A" заменяется на букву "N," "B" – на "O" и так далее. Каждая буква смещается на 13 мест. Шифрование файла *P* программой ROTI3 дважды восстанавливает первоначальный файл:

$$P = ROT13(ROT13(P)).$$

ROTI3 не используется для безопасности, она часто применяется в почте, закрывая потенциально неприятный текст.

Простое XOR , представляющее собой операцию "исключающее или" в языке C, является полиалфавитным шифром и широко используется в коммерческих программных продуктах. Это обычная операция над битами:

$$0 \oplus 0 = 0,$$
$$0 \oplus 1 = 1,$$
$$1 \oplus 0 = 1,$$
$$1 \oplus 1 = 0.$$

У полиалфавитных подстановочных шифров множественные однобуквенные ключи, каждый из которых используется для шифрования одного символа открытого текста. Первым ключом шифруется первый символ открытого текста, вторым ключом – второй символ и так далее. После использования всех ключей они повторяются циклически. Параметр, характеризующий количество ключей, называется **периодом** шифра. В классической криптографии шифры с длинным периодом было труднее раскрыть, чем шифры с коротким периодом. Использование компьютеров позволяет легко раскрыть подстановочные шифры с очень длинным периодом.

Шифр с бегущим ключом (иногда называемый книжным шифром), использующий один текст для шифрования другого текста, представляет собой другой пример подобного шифра. И хотя период этого шифра равен длине текста, он также может быть взломан [3].

Перестановочный шифр – это шифр, в котором меняется не открытый текст, а порядок символов. В **простом столбцовом перестановочном шифре** открытый текст пишется горизонтально на разграфленном листе бумаги, а шифротекст считывается по вертикали. Дешифри-

рование представляет собой запись шифротекста вертикально на листе разграфленной бумаги, и затем считывание открытого текста горизонтально. Пример использования перестановочного шифра приведен ниже для случая записи по 10 символов в строке.

Открытый текст: COMPUTER GRAPHICS MAY BE SLOW BUT AT LEAST IT'S EXPENSIVE.

Шифротекст: CAELP OPSEE MHLAN PIOSS UCWTI TSBIV EMUTE RATSG YAERB TX.

Так как символы шифротекста те же, что и в открытом тексте, частотный анализ шифротекста покажет, что каждая буква встречается приблизительно с той же частотой, что и обычно. Это дает возможность криптоаналитику применить различные методы, определяя правильный порядок символов для получения открытого текста. Применение к шифротексту второго перестановочного шифра значительно повысит безопасность. Существуют и еще более сложные перестановочные шифры, но компьютеры могут раскрыть почти все из них.

В 1920-х годах для автоматизации процесса шифрования были изобретены различные механические устройства. Большинство из них использовало понятие **ротора**, механического колеса, применяемого для выполнения подстановки.

Роторная машина, включающая клавиатуру и набор роторов, реализует вариант шифра Виженера (полиалфавитный подстановочный шифр). Каждый ротор представляет собой произвольное размещение алфавита, имеет 26 позиций и выполняет простую подстановку. Выходные штыри одного ротора соединены с входными штырями следующего ротора.

Например, в четырехроторной машине первый ротор может заменять "A" на "F", второй – "F" на "Y", третий – "Y" на "E" и четвертый – "E" на "C", "C" и будет конечным шифротекстом. Затем некоторые роторы смещаются, и в следующий раз подстановки будут другими.

2.2. Статистические свойства языка шифрования

В предыдущем разделе рассматривались шифры, применявшиеся для защиты информации в докомпьютерную эпоху. Установлено, что эти шифры легко взламываются с помощью статистических исследований языка, на котором они написаны (такой язык принято называть подлежащим).

Распределение (встречаемость) букв в английском среднестатистическом тексте представлено в табл. 2.1. Соответствующая гисто-

грамма изображена на рис. 2.1. Как видно из этих данных, наиболее часто встречающиеся буквы — это «e» и «t».

Таблица 2.1.

Среднестатистические частоты употребления английских букв

Буква	Процентное содержа-	Буква	Процентное содержа-
a	8,2	n	6,7
b	1,5	o	7,5
c	2,8	p	1,9
d	4,2	q	0,1
e	12,7	r	6,0
f	2,2	s	6,3
g	2,0	t	9,0
h	6,1	u	2,8
i	7,0	v	1,0
j	0,1	w	2,4
k	0,8	x	0,1
l	4,0	y	2,0
m	2,4	z	0,1

При взломе шифра полезно знать статистическую информацию и второго порядка, а именно, частоту встречаемости групп из двух и трех букв, называемых биграммами и триграммами. Наиболее часто встречающиеся биграммы представлены в табл. 2.2, а самые популярные триграммы выписаны в строку в порядке убывания частоты их использования:

the, ing, and, her, ere, ent, tha, nth, was, eth, for.

Рис. 2.1. Относительная встречаемость английских букв

Таблица 2.2. Частота встречаемости английских биграмм

Биграмма	Процентное содержа-	Биграмма	Процентное содер-
Th	3.15	he	2.51
An	1.72	in	1.69
Er	1.54	re	1.48
Es	1.45	on	1.45
Ea	1,31	ti	1,28
At	1.24	st	1.21
En	1,20	nd	1,18

2.3. Индекс совпадения

Криптоаналитику при вскрытии шифра необходимо понять, близок ли шифр криптограммы по своей природе к подстановочным шифрам. Другими словами, произошел ли данный шифротекст из чего-либо похожего на естественный язык.

Для проведения соответствующей статистической оценки происхождения шифротекста Фридманом в 1920 г. было предложено применять *индекс совпадения*.

Индекс совпадения

Пусть $x_1,\ldots,x_n$ – строка букв, а $f_1,\ldots,f_{25}$ – числа появлений букв в строке. Величина

$$IC(x) = \frac{\sum_{i=0}^{25} f_i(f_i - 1)}{n \cdot (n-1)}$$

называется *индексом совпадений.*

Заметим, что для случайного набора букв этот индекс равен

$$IC(x) \approx 0{,}038,$$

в то время как для осмысленного текста, написанного на английском или немецком языках он равен

$$IC(x) \approx 0{,}065.$$

Таким образом, можно сделать следующие выводы:

– Многие шифры, применявшиеся на ранней стадии развития криптологии, были взломаны, поскольку не смогли скрыть статистики подлежащего языка.

– Важнейшие приемы, лежащие в основе первых шифров – это замены и перестановки.

– Шифры работали с блоками знаков посредством снабженного ключом алгоритма или просто прибавляли очередное число из потока ключей к каждой букве открытого текста.
– Шифры, предназначенные устранить недостатки, связанные со статистикой языка, оказались менее стойкими, чем ожидалось, либо из-за конструкционных недочетов, либо в связи с неграмотной политикой использования, адаптированной в интересах операторов.

2.4. Одноразовые блокноты

И все-таки, идеальный способ шифрования существует. Он называется **одноразовым блокнотом** и был изобретен в 1917 году Мэйджором Джозефом Моборном (Major Joseph Mauborgne) и Гилбертом Вернамом (Gilbert Vernam). В классическом понимании одноразовый блокнот является большой неповторяющейся последовательностью символов ключа, распределенных случайным образом, написанных на кусочках бумаги и приклеенных к листу блокнота. Первоначально это была одноразовая лента для телетайпов. Отправитель использовал каждый символ ключа блокнота для шифрования только одного символа открытого текста.

Шифрование представляет собой сложение по модулю 26 символа открытого текста и символа ключа из одноразового блокнота. Каждый символ ключа используется только единожды и для единственного сообщения. Отправитель шифрует сообщения и уничтожает использованные страницы блокнота или использованную часть ленты. Получатель в свою очередь, используя точно такой же блокнот, дешифрирует каждый символ шифротекста. Расшифровав сообщение, получатель уничтожает соответствующие страницы блокнота или часть ленты. Новое сообщение – новые символы ключа.

Например, если сообщением является

ONETIMEPAD,

а ключевая последовательность в блокноте

TBFRGFARFM,

то шифротекст будет выглядеть следующим образом

IPKLPSFHGQ.

Дело в том, что весь алфавит имеет следующий пронумерованный вид:

1	2	3	4	5	6	7	8	9	10	11	12	13
A	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*	*K*	*L*	*M*

14	15	16	17	18	19	20	21	22	23	24	25	26
N	O	P	Q	R	S	T	U	V	W	X	Y	Z.

И поэтому

$(O+T) \bmod 26 = I$, $(N+B) \bmod 26 = P$, $(E+F) \bmod 26 = K$,

$(15+20) \bmod 26 = 9$; $(14+2) \bmod 26 = 16$; $(5+6) \bmod 26 = 11$;

и т.д.

В предположении, что злоумышленник не сможет получить доступ к одноразовому блокноту, использованному для шифрования сообщения, эта схема совершенно безопасна. Данное шифрованное сообщение на вид соответствует любому открытому сообщению того же размера. Так как все ключевые последовательности генерируются случайным образом, у противника отсутствует информация, позволяющая подвергнуть шифротекст криптоанализу.

Случайная ключевая последовательность, сложенная с неслучайным открытым текстом, дает совершенно случайный шифротекст, и никакие вычислительные мощности не смогут его распознать.

Необходимо отметить, что символы ключа должны генерироваться действительно случайным образом. Любые попытки вскрыть такую схему сталкиваются со способом, которым создается последовательность символов ключа. Использование генераторов псевдослучайных чисел неприемлемо – их свойства неслучайны.

Другой важной особенностью использования одноразовых блокнотов является то, что ключевую последовательность никогда нельзя использовать второй раз. Даже если вы используете блокнот размером в несколько гигабайт, то в случае получения криптоаналитиком ряда текстов с перекрывающимися ключами, он сможет восстановить открытый текст. Для этого достаточно сдвинуть каждую пару шифротекстов относительно друг друга и подсчитать число совпадений в каждой позиции. Если шифротексты смещены правильно, то число совпадений резко возрастет – точное значение зависит от языка открытого текста. С этой точки зрения криптоанализ не представляет большого труда.

Идея одноразового блокнота легко расширяется на двоичные данные. Вместо одноразового блокнота, состоящего из букв, используется одноразовый блокнот из битов. Применение такого блокнота для шифрования достаточно большого текста связано с рядом проблем. Так как ключевые биты должны быть случайными, и не могут использоваться снова, длина ключевой последовательности должна равняться длине сообщения.

Одноразовый блокнот удобен для нескольких небольших сообщений, но его нельзя использовать для работы с каналом связи большой пропускной способности – 1,544 Мбит/с. Можно хранить 650 Мбайт случайных данных на CD-ROM-е в 2-х экземплярах, но нужно уничтожать использованные биты. Для CD-ROM-а нет другой возможности удалить информацию, кроме как физически разрушить весь диск. Гораздо больше подходит цифровая лента.

Даже если проблемы распределения и хранения ключей решены, необходимо точно синхронизировать работу отправителя и получателя. Если получатель пропустит бит (или несколько бит пропадут при передаче), то сообщение потеряет всякий смысл. С другой стороны, если несколько бит изменятся при передаче (и ни один бит не будет удален или добавлен, что гораздо больше похоже на влияние случайного шума), то лишь эти биты будут расшифрованы неправильно. Но одноразовый блокнот не обеспечивает проверку подлинности.

Одноразовые блокноты используются и сегодня, главным образом для сверхсекретных каналов связи с низкой пропускной способностью. По слухам, "горячая линия" между Соединенными Штатами и бывшим Советским Союзом шифровалась с помощью одноразового блокнота. Многие сообщения разведчиков зашифрованы с использованием одноразовых блокнотов. Эти сообщения нераскрыты и сегодня, и навсегда останутся нераскрытыми. На этот факт не повлияет время работы суперкомпьютеров над этой проблемой.

3. ОСНОВНЫЕ ПОНЯТИЯ КРИПТОГРАФИИ

3.1. Криптографическая терминология

Отправитель и получатель. Отправитель намерен безопасно послать сообщение получателю, причем он хочет быть уверен, что перехвативший это сообщение не сможет его прочесть.

Сообщение и шифрование. Само сообщение называется открытым текстом (иногда используется термин “клер”). Изменение вида сообщения так, чтобы спрятать его суть, называется шифрованием. Шифрованное сообщение называется шифротекстом. Процесс преобразования шифротекста в открытый текст называется дешифрованием. Эта последовательность показана на рис. 3.1.

Рис. 3.1. Шифрование и дешифрирование

Обозначим открытый текст как M (от message, т.е. сообщение). Это может быть поток битов, текстовый файл, битовое изображение, оцифрованный звук, цифровое видеоизображение и т.д. Для компьютера M – это просто двоичные данные.

Обозначим шифротекст как C (от ciphertext). Это тоже двоичные данные, иногда того же размера, что и M, иногда больше. Если шифрование сопровождается сжатием, C может быть меньше, чем M. Однако, само шифрование не обеспечивает сжатие информации.

Функция шифрования E действует на M, создавая C. Соответствующая запись имеет вид:

$$E(M) = C.$$

В обратном процессе функция дешифрирования D действует на C, восстанавливая M:

$$D(C) = M.$$

Поскольку в результате проведения шифрования и последующего дешифрирования исходный текст восстанавливается, то имеет место следующее равенство:

$$D(E(M)) = M.$$

3.2. Алгоритмы и ключи

Криптографический алгоритм, также называемый **шифром**, представляет собой математическую функцию, используемую для шифрования и дешифрирования. Обычно это две связанных функции – одна для шифрования, а другая для дешифрования.

Если безопасность алгоритма основана на сохранении самого алгоритма в тайне, то это **ограниченный** алгоритм, не допускающий качественного контроля или стандартизации.

Современная криптография решает проблемы безопасности с помощью **ключа** *K*. Такой ключ может быть любым значением, выбранным из большого множества, называемого **пространством ключей**. И шифрование и дешифрирование определяет этот ключ (т.е. они зависят

от ключа, что обозначается индексом K), и теперь эти функции имеют вид:

$$E_K(M) = C,$$
$$D_K(C) = M.$$

При этом выполняется следующее равенство:

$$D_K(E_K(M)) = C.$$

Для некоторых алгоритмов при шифровании и дешифрировании используются различные ключи, то есть ключ шифрования K_1 отличается от соответствующего ключа дешифрирования K_2. Безопасность этих алгоритмов полностью основана на ключах, а не на деталях алгоритмов. Это значит, что алгоритм может быть опубликован и проанализирован. Продукты, использующие этот алгоритм, могут широко тиражироваться. Не имеет значения, что злоумышленнику известен ваш алгоритм; если ему не известен конкретный ключ, то он не сможет прочесть ваши сообщения.

Криптосистема представляет собой алгоритм, плюс все возможные открытые тексты, шифротексты и ключи. На рис. 3.2 представлена схема шифрования и дешифрирования с ключами.

Существует два основных типа алгоритмов, основанных на ключах – симметричные и с открытым ключом.

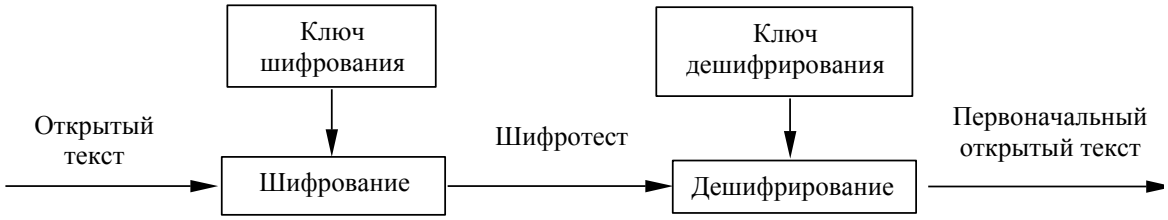

Рис. 3.2. Шифрование и дешифрирование с ключами

Симметричные алгоритмы представляют собой алгоритмы, в которых ключ шифрования может быть рассчитан по ключу дешифрирования и наоборот. В большинстве симметричных алгоритмов ключи шифрования и дешифрирования одни и те же. Эти алгоритмы, также называемые алгоритмами с секретным ключом или алгоритмами с одним ключом, требуют, чтобы отправитель и получатель согласовали используемый ключ перед началом безопасной передачи сообщений. Безопасность симметричного алгоритма определяется ключом; раскрытие клю-

ча означает, что кто угодно сможет шифровать и дешифрировать сообщения. Пока передаваемые сообщения должны быть тайными, ключ хранится в секрете.

Шифрование и дешифрирование с использованием симметричного алгоритма обозначается как:

$$E_K(M) = C,$$
$$D_K(C) = M.$$

Симметричные алгоритмы делятся на две категории. Одни алгоритмы обрабатывают открытый текст побитно (иногда побайтно), они называются **потоковыми алгоритмами** или **потоковыми шифрами**. Другие алгоритмы работают с группами битов открытого текста. Группы битов называются блоками, а алгоритмы – **блочными алгоритмами** или **блочными шифрами**. Для алгоритмов, используемых в компьютерных модемах, типичный размер блока составляет 64 бита – достаточно большое значение, чтобы помешать анализу, и достаточно небольшое и удобное – для работы.

Алгоритмы с открытым ключом (называемые асимметричными алгоритмами) разработаны таким образом, что ключ, используемый для шифрования, отличается от ключа дешифрирования. Более того, ключ дешифрирования не может быть (по крайней мере, в течение разумного интервала времени) рассчитан по ключу шифрования. Алгоритмы называются "с открытым ключом", потому что ключ шифрования может быть открытым: кто угодно может использовать ключ шифрования для шифрования сообщения, но только имеющий соответствующий ключ дешифрирования может расшифровать сообщение. В этих системах ключ шифрования часто называется **открытым** ключом, а ключ дешифрирования – **закрытым**. Закрытый ключ иногда называется секретным ключом [6]. Алгоритмы с открытым ключом были предложены Диффи и Хеллманом в 1975 году. Шифрование с открытым ключом K обозначается как:

$$E_K(M) = C.$$

Хотя открытый и закрытый ключи различны, дешифрирование с соответствующим закрытым ключом обозначается как:

$$D_K(C) = M.$$

Иногда сообщения шифруются закрытым ключом, а дешифрируются открытым, что используется для цифровой подписи.

3.3. Однонаправленные функции

Понятие **однонаправленной функции** является центральным в криптографии с открытыми ключами. Не являясь протоколами непосредственно, однонаправленные функции представляют собой краеугольный камень большинства протоколов. Однонаправленные функции относительно легко вычисляются, но инвертируются с большим трудом. То есть, зная *x* просто рассчитать *f(x)*, но по известному *f(x)* нелегко вычислить *x*. Здесь, "нелегко" означает, что для вычисления *x* по *f(x)* могут потребоваться миллионы лет, даже если над этой проблемой будут биться все компьютеры мира.

Хорошим примером однонаправленной функции служит разбитая тарелка. Легко разбить тарелку на тысячу крошечных кусочков, однако нелегко снова сложить тарелку из этих кусочков.

Математически строгого доказательства существования однонаправленных функций нет, нет и реальных свидетельств возможности их построения. Несмотря на это многие функции выглядят в точности как однонаправленные; мы можем рассчитать их, но до сих пор не знаем простого способа инвертировать их.

Предположим, что однонаправленные функции существуют. Какой смысл останавливаться на них, если непосредственно для шифрования они не используются. Сообщение, зашифрованное однонаправленной функцией, бесполезно – его невозможно дешифрировать. (Если написать на тарелке что-нибудь, затем разбить её на крошечные осколки, отдать их получателю, попросив прочитать сообщение, нетрудно догадаться о его реакции на такую однонаправленную функцию.) Для криптографии с открытыми ключами необходимо что-то другое (хотя существует и непосредственное криптографическое применение однонаправленных функций).

Однонаправленная функция с люком – это особый тип однонаправленной функции, с секретной лазейкой. Ее легко вычислить в одном направлении и трудно – в обратном. Но если вам известен секрет, вы можете легко рассчитать и обратную функцию. То есть легко вычислить *f(x)* по заданному *x*, но трудно по известному *f(x)* вычислить *x*. Однако, существует небольшая секретная информация, *y*, позволяющая, при знании *f(x)* и *y*, легко вычислить *x*.

В качестве хорошего примера однонаправленной функции с люком рассмотрим часы. Легко разобрать часы на сотни малюсеньких кусочков и трудно снова собрать из этих деталей работающие часы. Но с секретной информацией – инструкцией по сборке – намного легче решить эту задачу.

3.4. Однонаправленная хэш-функция

У **однонаправленной хэш-функции** может быть множество имен: функция сжатия, функция сокращения (contraction function), краткое изложение, характерный признак, криптографическая контрольная сумма, код целостности сообщения (message integrity check, MIC) и код обнаружения манипуляции (manipulation detection code, MDC). Как бы она ни называлась, эта функция является центральной в современной криптографии. Однонаправленные хэш-функции – это другая часть фундамента многих протоколов.

Хэш-функции, долгое время использующиеся в компьютерных науках, представляют собой функции, математические или иные, которые получают на вход строку переменной длины (называемую **прообразом**) и преобразуют ее в строку фиксированной, обычно меньшей, длины (называемую значением хэш-функции). В качестве простой хэш-функции можно рассматривать функцию, которая получает прообраз и возвращает байт, представляющий собой XOR всех входных байтов.

Смысл хэш-функции состоит в получении характерного признака, прообраза-значения, по которому анализируются различные прообразы при решении обратной задачи. Так как обычно хэш-функция представляет собой соотношение "многие к одному", невозможно со всей определенностью сказать, что две строки совпадают, но их можно использовать, получая приемлемую оценку точности.

Однонаправленная хэш-функция – это хэш-функция, которая работает только в одном направлении. Легко вычислить значение хэш-функции по прообразу, но трудно создать прообраз, значение хэш-функции которого равно заданной величине. Упоминавшиеся ранее хэш-функции, вообще говоря, не являются однонаправленными: задав конкретный байт, не представляет труда создать строку байтов, XOR которых дает заданное значение. С однонаправленной хэш-функцией такой вариант невозможен. Хэш-функция является открытой, тайны ее расчета не существует. Безопасность однонаправленной хэш-функции заключается именно в ее однонаправленности. У выхода нет видимой зависимости от входа. Изменение одного бита прообраза приводит к изменению (в среднем) половины битов значения хэш-функции. Вычислительно невозможно найти прообраз, соответствующий заданному значению хэш-функции.

С помощью однонаправленных хеш-функций можно получать характерные признаки файлов. Если вы хотите проверить, что у кого-то есть тот же файл, что и у вас, но вы не хотите, чтобы этот файл был передан вам, попросите послать вам значение хэш-функции. Если при-

сланное значение хэш-функции совпадет с рассчитанным вами, то, почти наверняка, чужой файл совпадает с вашим.

Код проверки подлинности сообщения (message authentication code, MAC), известный также как код проверки подлинности данных (data authentication code, DAC), представляет собой однонаправленную хэш-функцию с добавлением секретного ключа. Значение хэш-функции является функцией и прообраза, и ключа. Только тот, кто знает ключ, может проверить значение хэш-функции.

Код проверки подлинности сообщений MAC можно создать с помощью хэш-функции или блочного алгоритма шифрования, существуют также и специализированные MAC.

3.5. Передача информации с использованием криптографии с открытыми ключами

Симметричный алгоритм аналогичен сейфу. Ключ является комбинацией. Знающий комбинацию человек может открыть сейф, положить в него документ и снова закрыть. Кто-то другой при помощи той же комбинации может открыть сейф и забрать документ. Тем, кто не знает комбинации, придется научиться взламывать сейфы.

В 1976 году Уитфилд Диффи и Мартин Хеллман навсегда изменили эту парадигму криптографии (NSA (National Security Agency) сделало заявление, что знало о такой возможности еще в 1966 году, но доказательств не представило). Они описали **криптографию с открытыми ключами**, используя два различных ключа – один открытый и один закрытый. Определение закрытого ключа по открытому требует огромных вычислительных затрат. Кто угодно, используя открытый ключ, может зашифровать сообщение, но не расшифровать его. Расшифровать сообщение может только владелец закрытого ключа.

Это похоже на превращение криптографического сейфа в почтовый ящик. Шифрование с открытым ключом аналогично опусканию письма в почтовый ящик, любой может сделать это, опустив письмо в прорезь почтового ящика. Дешифрирование с закрытым ключом напоминает извлечение почты из почтового ящика. Обычно это гораздо сложнее – вам может понадобиться сварочный агрегат. Однако, если вы знаете секрет (у вас есть ключ от почтового ящика), вы без труда достанете вашу почту.

Математической основой процесса являются ранее обсуждавшиеся однонаправленные хэш-функции с люком. Шифрование выполняется в прямом направлении. Указания по шифрованию открыты, каждый может зашифровать сообщение. Дешифрирование выполняется в обратном

направлении. Оно настолько трудоемко, что, не зная секрета, даже на компьютерах Cray за тысячи (и миллионы) лет невозможно расшифровать сообщение. Секретом, или люком, и служит закрытый ключ, он делает дешифрирование таким же простым, как и шифрование.

Вот как, используя криптографию с открытыми ключами, Алиса может послать сообщение Бобу:

(1) Алиса и Боб согласовывают криптосистему (КС) с открытыми ключами.

(2) Боб посылает Алисе свой открытый ключ.

(3) Алиса шифрует свое сообщение открытым ключом Боба и отправляет его Бобу.

(4) Боб расшифровывает сообщение Алисы с помощью своего закрытого ключа.

Следует обратить внимание на то, что криптография с открытыми ключами устраняет проблему распределения ключей, присущую симметричным криптосистемам. Раньше Алиса и Боб должны были тайно договориться о ключе. Алиса могла выбрать любой ключ, но ей нужно было передать его Бобу. Она могла сделать это заранее, но это требует от нее определенной предусмотрительности. Она могла бы послать ключ с секретным курьером, но для этого нужно время. Криптография с открытыми ключами все упрощает. Алиса может отправить Бобу секретное сообщение без каких-либо предварительных действий. У Евы, подслушивающей абсолютно все, есть открытый ключ Боба и сообщение, зашифрованное этим ключом, но она не сможет получить ни закрытый ключ Боба, ни текст сообщения.

Обычно целая сеть пользователей согласовывает используемую криптосистему. У каждого из них есть открытый и закрытый ключ, открытые ключи помещаются в общедоступной базе данных. Теперь протокол выглядит еще проще:

(1) Алиса извлекает открытый ключ Боба из базы данных.

(2) Алиса шифрует свое сообщение с помощью открытого ключа Боба и посылает его Бобу.

(3) Боб расшифровывает сообщение Алисы с помощью своего закрытого ключа.

В первом протоколе Боб должен был послать Алисе ее открытый ключ прежде, чем она могла отправить ему сообщение. Второй протокол больше похож на обычную почту. Боб не участвует в протоколе до тех пор, пока он не начнет читать сообщение.

3.6. Смешанные криптосистемы

Первые алгоритмы с открытым ключом стали известны в то же время, когда проходило обсуждение DES как предполагаемого стандарта.

В действительности алгоритмы с открытыми ключами не заменяют симметричные алгоритмы и используются не для шифрования сообщений, а для шифрования ключей по следующим двум причинам:

1. Симметричные алгоритмы работают по крайней мере в 1000 раз быстрее, чем алгоритмы с открытыми ключами. Возможно, через 10–15 лет прогресс в развитии вычислительной техники позволит и алгоритмам с открытыми ключами достичь скоростей, сравнимых с сегодняшней скоростью симметричной криптографии. Но требования к объему передаваемой информации также возрастают, и всегда необходимо шифровать данные быстрее, чем это может сделать криптография с открытыми ключами.

2. Криптосистемы с открытыми ключами уязвимы по отношению к вскрытию с выбранным открытым текстом. Если $C = E(P)$, где P – открытый текст из n-возможных открытых текстов, то криптоаналитику нужно только зашифровать все n-возможные открытые тексты и сравнить результаты с C (помните, ключ шифрования общедоступен). Он не сможет раскрыть ключ дешифрирования, но он сможет определить P.

Вскрытие с выбранным открытым текстом может быть особенно эффективным, если число возможных шифрованных сообщений относительно мало. Например, если P – это денежная сумма в долларах, меньшая, чем \$1000000, то такое вскрытие сработает, криптоаналитик переберет весь миллион значений. (Эта проблема решается с помощью вероятностного шифрования.) Полезным может быть простое знание, что шифротекст не соответствует конкретному открытому тексту. Симметричные криптосистемы не чувствительны к вскрытиям такого типа, так как криптоаналитик не может выполнить тестовых дешифровок с неизвестным ключом.

В большинстве реализаций криптография с открытыми ключами используется для засекречивания и распространения сеансовых ключей, которые используются симметричными алгоритмами для закрытия потока сообщений. Иногда такие реализации называются смешанными (гибридными) криптосистемами.

(1) Боб посылает Алисе свой открытый ключ.

(2) Алиса создает случайный сеансовый ключ, шифрует его с помощью открытого ключа Боба и передает его Бобу

$$E_B(K).$$

(3) Боб расшифровывает сообщение Алисы, используя свой закрытый ключ, для получения сеансового ключа

$$D_B(E_B(K)) = K.$$

(4) Оба участника шифруют свои сообщения с помощью одного сеансового ключа.

Использование криптографии с открытыми ключами для распределения ключей решает очень важную проблему распределения ключей. В симметричной криптографии ключ шифрования данных не используется постоянно. Если злоумышленник получит его, то он сможет расшифровать все закрытые этим ключом сообщения. С помощью приведенного протокола создается сеансовый ключ, который уничтожается по окончании сеанса связи. Это значительно уменьшает риск компрометации сеансового ключа. Конечно, к компрометации чувствителен и закрытый ключ, но риска значительно меньше, так как в течение сеанса этот ключ используется только один раз для шифрования сеансового ключа [6].

3.7. Основные протоколы

Обмен ключами. Общепринятой криптографической техникой является шифрование каждого индивидуального обмена сообщениями отдельным ключом. Такой ключ называется сеансовым, так как он используется для единственного отдельного сеанса обмена информацией. Сеансовые ключи полезны, так как время их существования определяется длительностью сеанса связи. Передача этого общего сеансового ключа в руки обменивающихся информацией представляет собой сложную проблему.

Протокол, использующий **обмен ключами с помощью симметричной криптографии**, предполагает, что пользователи сети Алиса и Боб, получают секретный ключ от Центра распределения ключей (Key Distribution Center, KDC) Трента наших протоколов [6]. Перед началом протокола ключи должны быть у пользователей (протокол игнорирует очень насущную проблему доставки ключей, предполагается, что ключи уже у пользователей, и нарушитель Мэллори не имеет о них никакой информации).

(1) Алиса обращается к Тренту и запрашивает сеансовый ключ для связи с Бобом.

(2) Трент генерирует случайный сеансовый ключ. Он зашифровывает две копии ключа – одну для Алисы, а другую для Боба. Затем Трент посылает обе копии Алисе.

(3) Алиса расшифровывает свою копию сеансового ключа.

(4) Алиса посылает Бобу его копию сеансового ключа.

(5) Боб расшифровывает свою копию сеансового ключа.

(6) Алиса и Боб использует этот сеансовый ключ для безопасности обмена информацией.

Этот протокол основан на абсолютной надёжности Трента, для роли которого больше подходит заслуживающая доверия компьютерная программа, чем заслуживающий доверия человек. Если Меллори получит доступ к Тренту, то скомпрометированной окажется вся сеть. В его руках окажутся все секретные ключи, выделенные пользователям Трентом, он сможет прочесть все переданные сообщения, которые ему удалось перехватить, и все будущие сообщения. Ему останется только подключиться к линиям связи и подслушивать зашифрованный поток сообщений. Большой проблемой является и то, что Трент должен участвовать в каждом обмене ключами, и если с ним что-то случится, то это разрушит всю систему.

Обмен ключами, используя криптографию с открытыми ключами. Ранее нами обсуждалась смешанная КС. Для согласования сеансового ключа Алиса и Боб применяют криптографию с открытыми ключами, а затем используют этот сеансовый ключ для шифрования данных. В ряде реализаций подписанные ключи Алисы и Боба доступны в некоторой базе данных. Это облегчает протокол; теперь Алиса, даже если Боб о ней никогда не слышал, может послать Бобу сообщение.

(1) Алиса получает открытый ключ Боба из KDC.

(2) Алиса генерирует случайный сеансовый ключ, зашифровывает его открытым ключом Боба и посылает его Бобу.

(3) Боб расшифровывает сообщение Алисы с помощью своего закрытого ключа.

(4) Алиса и Боб шифруют свой обмен информацией этим сеансовым ключом.

Вскрытие "человек-в-середине". В то время как Ева не может сделать ничего лучшего, чем пытаться взломать алгоритм с открытыми ключами или выполнить вскрытие с использованием только шифротекста, у Мэллори гораздо больше возможностей [6]. Он не только может подслушать сообщения Алисы и Боба, но и изменить сообщения, удалить сообщения и создать совершенно новые. Мэллори может выдать себя за Боба, сообщающего что-то Алисе, или за Алису, сообщающую что-то Бобу. Вот как будет выполнено вскрытие.

(1) Алиса посылает Бобу свой открытый ключ. Мэллори перехватывает его и посылает Бобу свой собственный открытый ключ.

(2) Боб посылает Алисе свой открытый ключ. Мэллори перехватывает его и посылает Алисе, как и Бобу, свой собственный открытый ключ.

(3) Когда Алиса посылает сообщение Бобу, зашифрованное открытым ключом "Боба", Мэллори перехватывает его. Так как сообщение в действительности зашифровано его собственным открытым ключом, он расшифровывает его своим закрытым ключом, затем снова зашифровывает открытым ключом Боба и посылает Бобу.

(4) Когда Боб посылает сообщение Алисе, зашифрованное открытым ключом "Алисы", Мэллори перехватывает его. Так как сообщение в действительности зашифровано его собственным открытым ключом, он расшифровывает его, снова зашифровывает открытым ключом Алисы и посылает Алисе.

Это вскрытие будет работать, даже если открытые ключи Алисы и Боба хранятся в базе данных. Мэллори может перехватить запрос Алисы к базе данных и подменить открытый ключ Боба своим собственным. То же самое он может сделать и с открытым ключом Алисы. Или, еще лучше, он может взломать базу данных и подменить открытые ключи Боба и Алисы своим. Затем он дожидается момента, когда Алиса и Боб начнут обмениваться сообщениями, и начинает перехватывать и изменять эти сообщения.

Такое **вскрытие "человек-в-середине"** работает, так как у Алисы и Боба нет способа проверить, действительно ли они общаются именно друг с другом. Если вмешательство Мэллори не приводит к заметным задержкам в сети, оба корреспондента и не подумают, что кто-то, сидящий между ними, читает всю их секретную почту.

Протокол "держась за руки", изобретенный Роном Ривестом (Ron Rivest) и Эди Шамиром (Adi Shamir), предоставляет неплохую возможность избежать вскрытия "человек-в-середине" [6]. Вот как он работает.

1. Алиса посылает Бобу свой открытый ключ.
2. Боб посылает Алисе свой открытый ключ.
3. Алиса зашифровывает свое сообщение открытым ключом Боба. Половину зашифрованного сообщения она отправляет Бобу.
4. Боб зашифровывает свое сообщение открытым ключом Алисы. Половину зашифрованного сообщения он отправляет Алисе.
5. Алиса отправляет Бобу вторую половину зашифрованного сообщения.
6. Боб складывает две части сообщения Алисы и расшифровывает его с помощью своего закрытого ключа. Боб отправляет Алисе вторую половину своего зашифрованного сообщения.

7. Алиса складывает две части сообщения Боба и расшифровывает его с помощью своего закрытого ключа.

Идея в том, что половина зашифрованного сообщения бесполезна без второй половины, она не может быть дешифрирована. Боб не сможет прочитать ни одной части сообщения Алисы до этапа (6), а Алиса не сможет прочитать ни одной части сообщения Боба до этапа (7). Существует множество способов разбить сообщение на части, например:

- если используется блочный алгоритм шифрования, половина каждого блока (например, каждый второй бит) может быть передана во второй части сообщения;
- первая половина сообщения может быть однонаправленной хэш-функцией шифрованного сообщения, а вторая половина – собственно шифрованным сообщением.

Чтобы понять, как такой протокол помешает Мэллори, рассмотрим его попытку нарушить протокол. Как и раньше, он может подменить открытые ключи Алисы и Боба своим ключом на этапах (1) и (2). Но теперь, перехватив половину сообщения Алисы на этапе (3), он не сможет расшифровать ее своим закрытым ключом и снова зашифровать открытым ключом Боба. Он может создать совершенно новое сообщение и отправить половину его Бобу. Перехватив половину сообщения Боба Алисе на этапе (4), Мэллори столкнется с этой же проблемой. Он не сможет расшифровать ее своим закрытым ключом и снова зашифровать открытым ключом Алисы. Ему придется создать совершенно новое сообщение и отправить половину его Алисе. К тому времени, когда он перехватит вторые половины настоящих сообщений на этапах (5) и (6), подменять созданные им новые сообщения будет слишком поздно. Обмен данными между Алисой и Бобом изменится радикально.

Мэллори может попытаться избежать такого результата. Если он достаточно хорошо знает обоих корреспондентов, чтобы подстроиться под них при обмене данными, они могут никогда не заметить подмены. Но все-таки это сложнее, чем просто перехватывать и читать сообщения.

Обмен ключами с помощью цифровых подписей. Использование цифровой подписи в протоколе обмена сеансовым ключом также позволяет избежать вскрытия "человек-в-середине". Трент подписывает открытые ключи Алисы и Боба. Подписанные ключи включают подписанное заверение подлинности. Получив ключи, и Алиса, и Боб проверяют подпись Трента. Теперь они уверены, что присланный открытый ключ принадлежит именно указанному корреспонденту. Затем выполняется протокол обмена ключами.

Мэллори сталкивается с серьезными проблемами. Ему необходимо взламывать KDC (компрометируя Трента), но при этом он получает только закрытый ключ Трента. Этот ключ позволит ему только подписывать новые ключи, а не расшифровывать сеансовые ключи и читать произвольный поток сообщений. Для чтения сообщений Мэллори придется выдать себя за пользователя сети и обманывать пользователей, шифруя сообщения своим поддельным открытым ключом. Тем не менее, предприняв такое вскрытие и используя закрытый ключ Трента, он может создать поддельные подписанные ключи, чтобы обмануть Алису и Боба. Затем он может либо подменить этими ключами настоящие ключи в базе данных, либо перехватывать запросы пользователей к базе данных и посылать в ответ поддельные ключи. Это позволит ему осуществить вскрытие "человек-в-середине" и читать сообщения пользователей.

Такое вскрытие будет работать, но для этого Мэллори должен уметь перехватывать и изменять сообщения. В ряде сетей это намного сложнее, чем просто пассивно сидеть, просматривая сообщения в сети по мере их поступления. В широковещательных каналах, таких как радиосеть, почти невозможно подменить одно сообщение другим – хотя можно забить всю сеть. В компьютерных сетях это менее сложно и, кажется, с каждым днем становится проще и проще.

Передача ключей и сообщений. Алисе и Бобу не обязательно выполнять протокол обмена ключами перед обменом сообщениями. В этом протоколе Алиса отправляет Бобу сообщение без предварительного протокола обмена ключами.

(1) Алиса генерирует случайный сеансовый ключ, K, и зашифровывает M этим ключом

$E_K(M)$.

(2) Алиса получает открытый ключ Боба из базы данных.

(3) Алиса шифрует K открытым ключом Боба

$E_B(K)$.

(4) Алиса посылает Бобу шифрованное сообщение и сеансовый ключ

$E_K(M), E_B(K)$.

Для дополнительной защиты от вскрытия "человек-в-середине" Алиса подписывает передачу.

(5) Боб расшифровывает сеансовый ключ Алисы, *K*, используя свой закрытый ключ.

(6) Боб, используя сеансовый ключ, расшифровывает сообщение Алисы.

Подобная смешанная система и употребляется чаще всего в системах связи. Ее можно соединить с цифровыми подписями, метками времени и другими протоколами обеспечения безопасности.

Подпись документа с помощью криптографии с открытыми ключами. Существуют алгоритмы с открытыми ключами, которые можно использовать для цифровых подписей. В некоторых алгоритмах – примером является RSA (см. раздел 4.6) – для шифрования может быть использован или открытый, или закрытый ключ. Зашифруйте документ своим закрытым ключом, и вы получите надежную цифровую подпись [6].

В других случаях – примером является DSA – для цифровых подписей используется отдельный алгоритм, который невозможно использовать для шифрования. Эта идея впервые была предложена Диффи и Хеллманом. Основной протокол прост.

(1) Алиса шифрует документ своим закрытым ключом, таким образом подписывая его.

(2) Алиса посылает подписанный документ Бобу.

(3) Боб расшифровывает документ, используя открытый ключ Алисы, и таким образом проверяя подпись.

Если Боб не смог осуществить этап (3), то он знает, что подпись неправильна. Такая подпись соответствует всем требованиям:

- Эта подпись достоверна. Когда Боб расшифровывает сообщение с помощью открытого ключа Алисы, он знает, что она подписала это сообщение.
- Эта подпись неподдельна. Только Алиса знает свой закрытый ключ.
- Эту подпись нельзя использовать повторно. Подпись является функцией документа и не может быть перенесена на другой документ.
- Подписанный документ нельзя изменить. После любого изменения документа подпись не сможет больше подтверждаться открытым ключом Алисы.

- От подписи невозможно отказаться. Бобу не требуется помощь Алисы при проверке ее подписи.

Подпись документа и метки времени. На самом деле при определенных условиях Боб сможет смошенничать. Он может повторно использовать документ и подпись совместно. Это не имеет значения, если Алиса подписала контракт (одной копией подписанного контракта больше, одной меньше), но что если Алиса поставила цифровую подпись под чеком?

Предположим, что Алиса послала Бобу подписанный чек на $100. Боб отнес чек в банк, который проверил подпись и перевел деньги с одного счета на другой. Боб сохранил копию электронного чека. На следующей неделе он снова отнес его в этот или другой банк. Банк подтвердил подпись и перевел деньги с одного счета на другой. Если Алиса не проверяет свою чековую книжку, то Боб сможет проделывать это годами.

Поэтому в цифровые подписи часто включают метки времени. Дата и время подписания документа добавляются к документу и подписываются вместе со всем содержанием сообщения. Банк сохраняет эту метку времени в базе данных. Теперь, если Боб попытается получить наличные по чеку Алисы во второй раз, банк проверит метку времени по своей базе данных и откажет Бобу.

Стюарт Хабер (Stuart Haber) и В. Скотт Старнетта (W. Scott Starnetta) обеспечили безопасность протоколов связи на основе реализации цифровых меток времени со следующими свойствами:

- метка времени должна существовать сама по себе, вне зависимости от физической среды, используемой для её хранения;
- не должно существовать возможности изменить хотя бы один бит документа;
- не должно существовать возможности задать для документа метку времени, отличного от текущего времени.

4. КОМПЬЮТЕРНЫЕ АЛГОРИТМЫ ШИФРОВАНИЯ

4.1. Симметричные шифры

Работа симметричных шифров включает в себя два преобразования:

$$C = E_k(m) \quad \text{и} \quad m = D_k(C),$$

где m — открытый текст, E – шифрующая функция, D – расшифровывающая функция, k – секретный ключ, C – шифротекст. Следует отметить, что как шифрующая, так и расшифровывающая функции общеизвестны, и тайна сообщения при известном шифротексте зависит только от секретности ключа k. Хотя этот хорошо обоснованный принцип, называемый *принципом Керкхоффа,* был известен еще начиная с середины 1800-ых годов, множество компаний все еще игнорирует его. Существует много отдельных фирм, разрабатывающих собственные секретные схемы шифрования, которые теряют свою стойкость, как только кто-то из персонала допускает утечку информации о деталях алгоритмов. Опыт показывает, что наилучшими схемами шифрования являются те, которые изучались в течение длительного времени большой армией исследователей и рекомендованы к применению как наиболее безопасные

Алгоритм, символическая запись которого приведена в начале главы, называется криптосистемой с симметричным ключом, поскольку обе стороны, обменивающиеся шифрованной информацией, применяют один и тот же секретный ключ. Иногда симметричные криптосистемы используют два ключа: один для шифрования, а другой для обратного процесса. В этом случае мы будем предполагать, что шифрующий ключ легко восстанавливается по расшифровывающему и наоборот.

Позже мы встретимся с криптосистемами с открытым ключом. В них только один ключ хранится в тайне и называется секретным, в то время как второй, открытый ключ, доступен всем желающим. При этом считается невозможным для кого бы то ни было вычислить секретный ключ, опираясь на информацию об открытом.

Возвращаясь к симметричной криптографии, отметим, что число возможных ключей должно быть очень велико. Это требование возникает в связи с тем, что при проектировании криптоалгоритма мы обязаны учитывать самый плохой сценарий развития событий, ставя гипоте-

тического противника в максимально выгодное положение, т.е. мы считаем, что атакующий

– обладает полнотой информации о шифрующем (расшифровывающем) алгоритме,

– имеет в своем распоряжении некоторое количество пар (открытый текст, шифротекст) ассоциированных с истинным ключом *k*.

Если количество возможных ключей мало, то атакующий имеет возможность взломать шифр простым перебором вариантов. Он может шифровать один из данных открытых текстов, последовательно используя разные ключи, до тех пор, пока не получит соответствующий известный шифротекст. В результате искомый ключ будет найден. Именно для исключения такого сорта атаки необходимо предусмотреть достаточно большое пространство ключей. Принято считать, что вычисления, состоящие из 2^{80} шагов, в ближайшие несколько лет будут неосуществимы. Поэтому ключ, исключающий взлом простым перебором, должен насчитывать, по крайней мере, 80 битов.

Хороший разработчик шифра в совершенстве владеет двумя специальностями: он должен взламывать криптосистемы так же блестяще, как и изобретать их. В наши дни, несмотря на многочисленные попытки, все еще не найден метод аналитического доказательства стойкости симметричных криптосистем, в отличие от алгоритмов с открытым ключом, где криптостойкость шифров доказывается цепочкой логических рассуждений, начинающейся достаточно естественными предположениями. Однако симметричные алгоритмы все еще активно эксплуатируются, поскольку лучшие методы атак на эти шифры, разработанные ведущими криптоаналитиками, пока не достигают цели.

Упрощенная модель шифрования

На рис. 4.1 изображена упрощенная модель шифрования битовой строки, которая, несмотря на свою простоту, вполне подходит для практического применения. Идея модели состоит в применении к открытому тексту обратимой операции для получения шифротекста, а именно, побитовое сложение по модулю 2 открытого текста со «случайным потоком» битов. Получатель может восстановить текст с помощью обратной операции, сложив шифротекст с тем же самым случайным потоком.

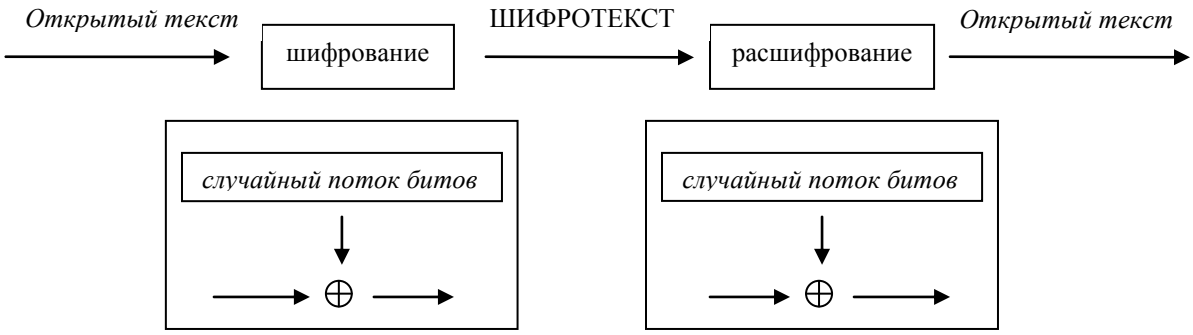

Рис. 4.1. Упрощенная модель, шифрующая строку битов

Такую модель легко реализовать на практике, поскольку для ее реализации необходима одна из простейших компьютерных операций – исключающее ИЛИ, т. е. сложение по модулю 2, которое обозначается знаком «⊕». Шифруя каждое новое сообщение своим ключом, длина которого совпадает с длиной открытого текста, мы получим абсолютно стойкую симметричную криптосистему. Напомним, что такая криптосистема называется одноразовым шифр-блокнотом. Однако, несмотря на совершенство этого алгоритма, он не применяется на практике, поскольку порождает почти неразрешимую проблему распределения ключей. В связи с этим разрабатываются симметричные криптосистемы, в которых длинное сообщение шифруется коротким ключом, причем этот ключ можно использовать несколько раз. Естественно, такие системы далеки от абсолютно стойких, но, с другой стороны, распределение ключей для них – хотя и трудная, но вполне решаемая задача.

Большинство симметричных шифров можно разделить на две больших группы. Первая – поточные шифры, где за один раз обрабатывается один элемент данных (бит или буква), а вторая – блочные шифры, в которых за один шаг обрабатывается группа элементов данных (например, 64 бита).

4.2. Поточные шифры

Рисунок 4.2 дает простую иллюстрацию поточного шифра. Обратите внимание, как она похожа на предыдущую упрощенную модель. Тем не менее, случайный поток битов теперь генерируется по короткому секретному ключу с помощью открытого алгоритма, называемого *генератором ключевого потока.* Здесь биты шифротекста получаются по

правилу: $C_i = m_i \oplus k_i$, где $m_0, m_1, \ldots$ – биты открытого текста, а $k_0, k_1, \ldots$ – биты ключевого потока.

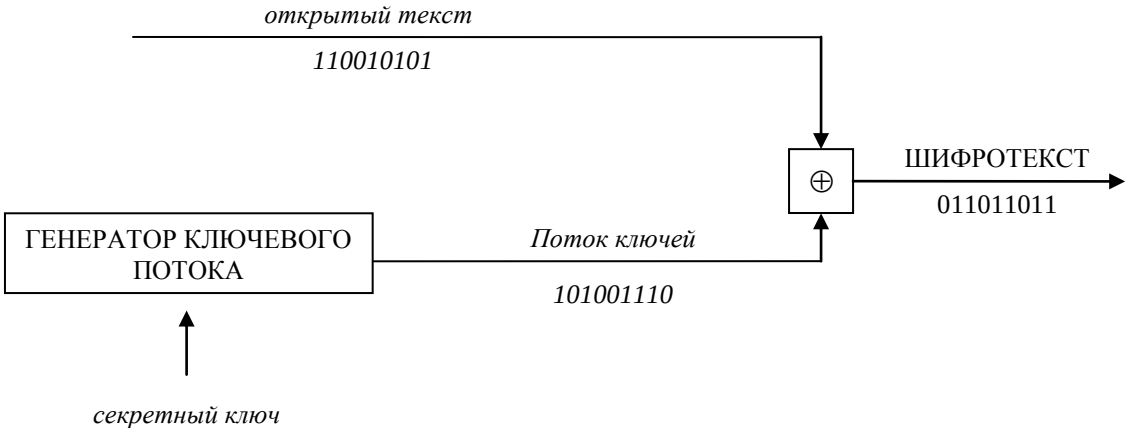

Рис. 4.2. Поточные шифры

Поскольку процесс шифрования – это сложение по модулю 2, расшифрование является, по существу, той же самой операцией:

$$m_i = C_i \oplus k_i .$$

Поточные шифры, похожие на описанные выше, просты и удобны для реализации. Они позволяют очень быстро шифровать большой объем данных. Поэтому они подходят для передачи аудио- и видеосигналов в реальном времени. Кроме того, в этом процессе не происходит накопления ошибки. Если отдельный бит шифротекста исказился в процессе передачи вследствие слабого радиосигнала или из-за вмешательства противника, то в расшифрованном открытом тексте только один бит окажется неверным. Однако повторное использование того же ключа дает тот же ключевой поток, что влечет за собой зависимость между соответствующими сообщениями. Предположим, например, что сообщения m_1 и m_2 были зашифрованы одним ключом k. Тогда противник, перехватив шифровки, легко найдет сумму по модулю 2 открытых текстов:

$$C_1 \oplus C_2 = (m_1 \oplus k) \oplus (m_2 \oplus k) = m_1 \oplus m_2 .$$

Следовательно, необходимо менять ключи либо с каждым новым сообщением, либо с очередным сеансом связи. В результате мы сталкиваемся с трудностями управления ключами и их распределения, которые преодолеваются, как мы позже увидим, с помощью криптосистем с открытым ключом. Обычно алгоритмы с открытым ключом применяются для передачи ключа, закрепленного за отдельным сообщением или

за целым сеансом связи, а фактические данные шифруются после этого с помощью поточного или блочного шифров.

Чтобы придать необходимую стойкость шифру, генератор ключевого потока производит строку битов с определенными свойствами. Как минимум, ключевой поток должен:

– Иметь большой период. Поскольку ключевой поток получается в результате детерминированного процесса из основного ключа, найдется такое число n, что $k_i = k_i + n$ для всех значений i. Число n называется периодом последовательности, и для обеспечения стойкости шифра выбирается достаточно большим.
– Иметь псевдослучайные свойства. Генератор должен производить последовательность, которая кажется случайной. Другими словами, генерируемая последовательность должна выдержать определенное число статистических тестов на случайность.
- Обладать линейной сложностью. Далее объясняется значение этого термина.

Однако перечисленных условий не хватает, поскольку восстановление значительной части этой последовательности должно быть неосуществимым в вычислительном отношении. В идеале, даже если кто-то знает первый миллиард битов ключевой последовательности, вероятность угадать следующий бит не должна превышать 50%.

4.3. Блочные шифры

На рис. 4.3 изображена схема блочного алгоритма шифрования. Блочный шифр за один прием обрабатывает блок открытого текста. Основное отличие блочного шифра от поточного состоит в том, что поточным шифрам необходимо постоянно помнить о том, какое место битовой строки они в данный момент обрабатывают, чтобы определить, какую часть ключевого потока нужно сейчас генерировать; блочные же шифры избавлены от этой необходимости. Как и в случае поточных шифров, мы пишем

$$C = E_k(m) \quad \text{и} \quad m = D_k(C),$$

где m – блок открытого текста, k – секретный ключ, E – шифрующая функция, D – расшифровывающая функция, C – блок шифротекста.

Размер блока для шифрования обычно выбирают разумно большим. В системе *DES* (стандарт шифрования данных), например, он состоит из 64 битов, а в современных блочных криптосистемах он достигает 128 битов и более.

Часто зашифрованный первый блок сообщения используют для шифрования следующего. Такой прием обычно называют *режимом шифрования.* Режимы используются, чтобы избежать некоторых атак, основанных на стирании или вставке, придавая каждому блоку шифротекста контекст, присущий всему сообщению.

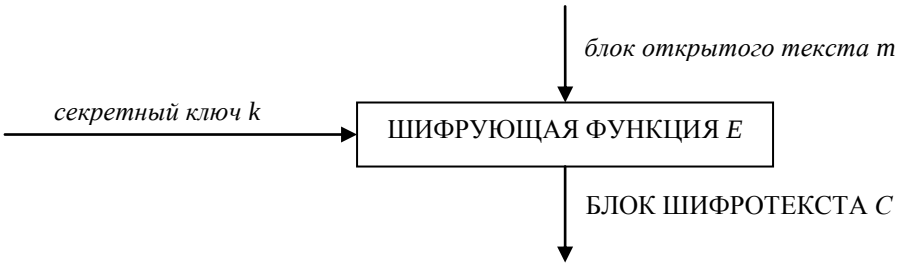

Рис. 4.3. Схема работы блочного шифра

Каждый режим шифрования предполагает свою защиту от накопления ошибок из-за сбоев передачи шифротекста. Кроме того, в зависимости от режима работы (и приложений) выбирается ключ сообщения или сеанса связи. Например, многие режимы шифрования требуют некоего начального значения, вводимого перед операциями шифрования и расшифрования.

Сегодня принято на вооружение довольно много разновидностей блочных шифров, некоторые из которых с большой долей вероятности используются Вашим web-браузером: *RC5, RC6, DES* или *3DES.* Наиболее знаменитый из них – *DES,* т.е. стандарт шифрования данных. Впервые он был опубликован в середине семидесятых годов XX века как федеральный стандарт США и вскоре оказался, де-факто, международным стандартом в банковских операциях. *DES* успешно выдержал испытание временем, но к началу 90-ых годов назрела необходимость в разработке новых стандартов. Произошло это потому, что как длина блока (64 бита), так и размер ключа (56 битов) оригинального алгоритма *DES* оказались недостаточными для обеспечения секретности сообщений. В настоящее время можно восстановить 56-битовый ключ системы *DES,* используя либо сеть компьютеров, либо специализированные аппаратные средства ЭВМ. В ответ на эту проблему национальный институт стандартов и технологий США (NIST) положил начало своего рода соревнованию по поиску нового блочного шифра, достойного названия «новый стандарт шифрования» *(Advanced Encryption Standard, AES).*

В отличие от фактически засекреченных работ над проектированием *DES,* проект *AES* осуществлялся публично. Множество исследовательских групп всего мира представили свои варианты *AES* на конкурс. В финал вышли пять алгоритмов, которые изучались глубже с целью выбора победителя. Это были криптосистемы:

– *MARS* от группы при компании *IBM;*

– *RC6,* представленная компанией *RSA Security;*

– *Twofish* от группы базирующейся в Коунтерпэйне, Беркли и других местах;

– *Serpent* от группы трех ученых, работающих в Израиле, Норвегии и Британии;

– *Rijndael* от двух бельгийских криптографов.

В конце 2000 г. NIST объявил, что победителем конкурса был выбран шифр *Rijndael.*

Криптосистема *DES* и все финалисты проекта *AES* – примеры *итерированного* блочного шифра. В таких шифрах стойкость обеспечивается повторяющимся использованием простой *раундовой функции,* преобразующей n-битовые блоки в n-битовые блоки, где n – размер блока шифра. Число раундов S может меняться или быть фиксированным. Как правило, с увеличением числа раундов уровень стойкости блочного шифра повышается.

При каждом применении раундовой функции используется подключ

$$k_i \quad \text{при } 1 \le i \le S,$$

выводящийся из основного секретного ключа k с помощью алгоритма *разворачивания ключа.* Чтобы шифротекст можно было успешно расшифровать, функция, генерирующая подключи, должна быть обратимой. При расшифровании подключи применяются в порядке, обратном тому, в котором они использовались при шифровании. Требование обратимости каждого раунда не подразумевает обратимости функций, в нем участвующих. На первый взгляд это кажется странным, но станет совершенно очевидным после подробного обсуждения криптосистемы *DES.* Функции, которые в ней используются, необратимы, но, тем не менее, каждый раунд обратим. В то же время, в схеме *Rijndael* обратимы не только раунды, но и все функции.

4.4. Шифр Фейстеля

Шифр *DES* – вариант базисного шифра Фейстеля (см. рис. 4.4), названного по имени Г. Фейстеля, работавшего в фирме *IBM* и выполнившего некоторые из самых ранних невоенных исследований в области алгоритмов шифрования. Интересная особенность шифра Фейстеля заключается в том, что функция раунда обратима вне зависимости от свойств функции F (см. рис. 4.4). Чтобы в этом убедиться, обратимся к формулам. Каждый раунд шифрования осуществляется по правилу

$$l_i = r_{i-1}, \quad r_i = l_{i-1} \oplus F(k_i, r_{i-1}).$$

Следовательно, расшифрование происходит за счет преобразований:

$$r_{i-1} = l, \quad l_{i-1} = r_i \oplus F(k_i, l_i).$$

Характерные особенности предлагаемой схемы шифрования заключаются в следующем:

– в качестве F можно выбрать любую функцию и получить шифрующую функцию, которая будет обращаться при помощи секретного ключа;

– одну и ту же микросхему можно использовать как для шифрования, так и для расшифрования. Нам необходимо лишь проследить за порядками подключей, которые в этих процессах обратны друг другу.

Для создания криптостойкого шифра необходимо решить следующие вопросы:

– как генерировать подключи,
– сколько должно быть раундов,
– как определить функцию F.

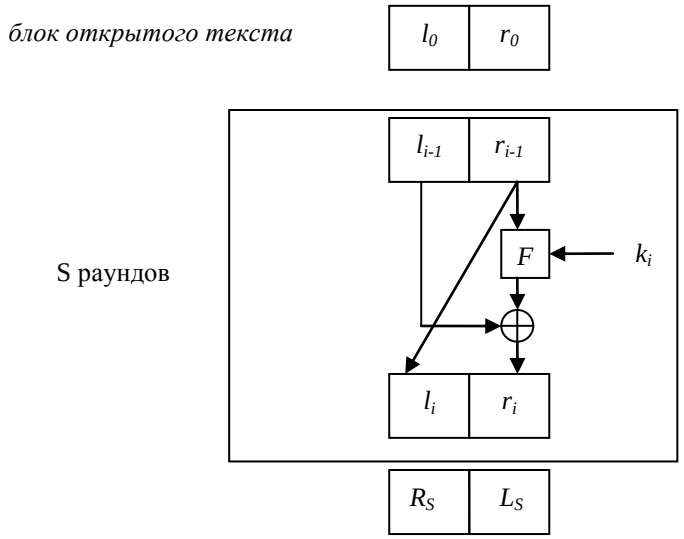

Рис. 4.4. Основные операции шифра Фейстеля

4.5. Шифр DES

Работа над *DES* была начата в начале 1970-ых годов группой сотрудников *IBM*, в которую входил и Фейстель. Отправной точкой проекта послужил более ранний шифр – «Люцифер», как называли его в *IBM*. Было известно, что управление национальной безопасности *(NSA)* внесло изменения в проект. Долгое время специалисты в области секретности считали, что изменения состояли в использовании ловушки в функции *F*. Однако теперь считается, что они были направлены на повышение безопасности шифра. В частности, эти модификации укрепили сопротивляемость шифра дифференциальному криптоанализу – технике, с которой гражданские исследователи не были знакомы до 1980 - ых годов.

В документах национального института стандартизации США *(ANSI)* криптосистема *DES* называется *алгоритмом шифрования данных (DEA),* а международная организация по стандартизации, ссылаясь на шифр *DES*, пользуется аббревиатурой *DEA-1*. Этот алгоритм являлся мировым стандартом на протяжении более чем двадцати лет и утвердился как первый доступный всем желающим официальный алгоритм. Поэтому его стоит отметить как важнейшую веху на пути криптографии от чисто военного использования к широкомасштабному применению.

Основные черты шифра *DES* определяются прежде всего тем, что он – обобщение шифра Фейстеля и, кроме того, в нем:

– число раундов *S* равно 16,

– длина блока n – 64 бита,
– размер ключа – 56 битов,
– каждый из подключей $k_1, k_2, \ldots, k_{16}$ насчитывает 48 битов.

Заметим, что для многих современных алгоритмов длина ключа в 56 битов недостаточна. Поэтому в *DES* зачастую используют три ключа вместо одного, проводя три итерации стандартного процесса. При этом, как легко подсчитать, длина ключа становится равной 168 битам. Такая версия классического шифра называется *тройным DES* или *3DES* (рис. 4.5). Есть и другой способ модификации *DES,* в которой берут два ключа, увеличивая длину основного ключа до 112 битов.

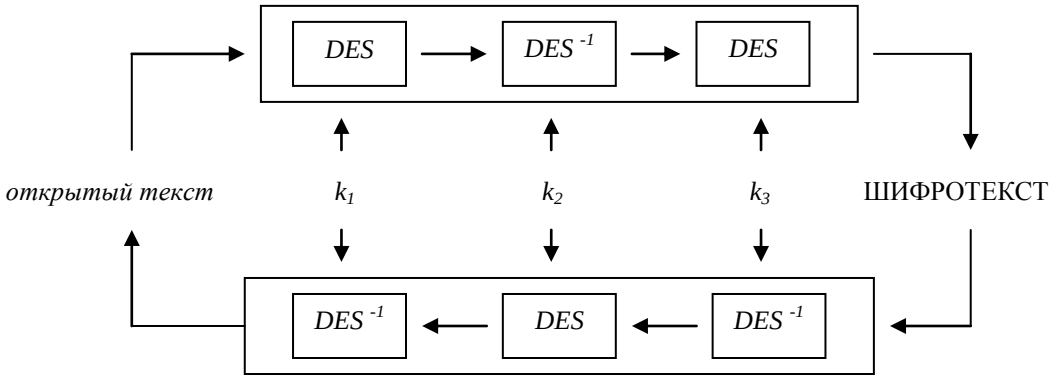

Рис. **4**.5. Тройной DES

Как уже отмечалось, в первом приближении *DES* – это шифр Фейстеля с 16 раундами (рис. 4.6), за исключением того, что как перед, так и после основных итераций алгоритма Фейстеля осуществляются некоторые перестановки. На рис. 4.6 показано что два блока меняются местами перед последней перестановкой алгоритма. Эта замена не влияет на стойкость шифра, и пользователи часто задавались вопросом: зачем ее вообще делать? Один из членов творческого коллектива, разработавшего *DES,* утверждал, что она облегчает микросхемную реализацию процедуры шифрования.

Шифр *DES* преобразует открытый текст из 64 битов следующим образом:

- производит начальную перестановку (IP);
- расщепляет блок на левую и правую половины;
- осуществляет 16 раундов с одним и тем же набором операций;
- соединяет половины блока;
- производит конечную перестановку.

Конечная перестановка обратна начальной. Это позволяет использовать одно и то же программное обеспечение и «железо» для двух сторон процесса: шифрования и расшифрования. Разворачивание ключа

дает 16 подключей по 48 битов каждый, выделяя их из 56-битного основного ключа.

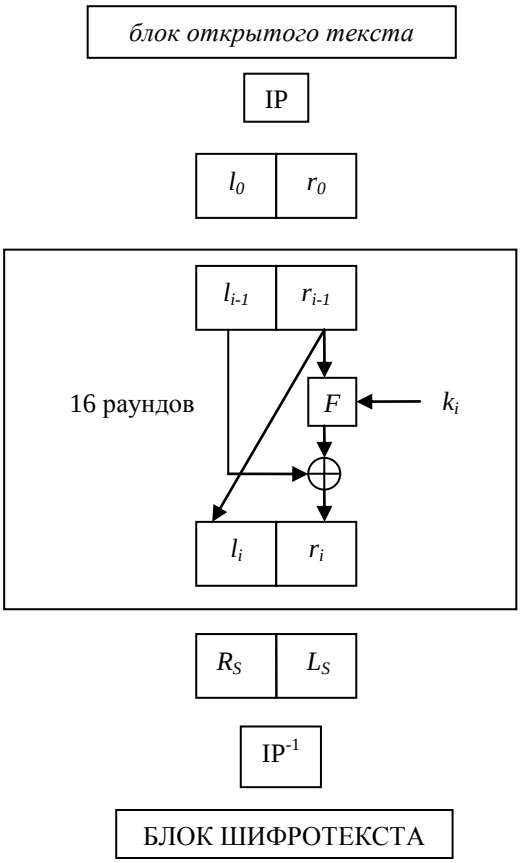

Рис. 4.6. Алгоритм *DES*

4.6. Режимы работы DES

Блочный шифр, подобный *DES* или *Rijndael,* можно по-разному использовать для шифрования строк данных. Вскоре после *DES* в США был принят еще один федеральный стандарт, *рекомендующий* четыре способа эксплуатации алгоритма *DES* для шифрования данных. С тех пор эти режимы стали общепринятыми и применяются с любыми блочными шифрами. Перечислим их.

– ***ECB***. Этот режим прост в обращении, но слабо защищен от возможных атак с удалениями и вставками. Ошибка, допущенная в одном из битов шифротекста, влияет на целый блок в расшифрованном тексте.

– ***CBC*** – наилучший способ эксплуатации блочного шифра, поскольку предназначен для предотвращения потерь в результате

атаки с использованием удалений и вставок. Здесь ошибочный бит шифротекста при расшифровании не только превращает в ошибочный блок, в котором содержится, но и портит один бит в следующем блоке открытого текста, что можно легко определить и интерпретировать как сигнал о предпринятой атаке.

– ***OFB.*** При таком методе блочный шифр превращается в поточный. Режим обладает тем свойством, что ошибка в один бит, просочившаяся в шифротекст, дает только один ошибочный бит в расшифрованном тексте.

– ***CFB***. Как и в предыдущем случае, здесь блочный шифр трансформируется в поточный. Отдельная ошибка в криптограмме при этом влияет как на блок, в котором она была допущена, так и на следующий блок, как при режиме *CBC.*

4.7. Шифр Rijndael

Победитель конкурса *AES,* объявленный в конце 2000 года, алгоритм *Rijndael,* был разработан двумя бельгийскими криптографами: Дименом (Daemen) и Рийменом (Rijmen). Эта криптосистема, относясь к блочным шифрам, имеет много общего с *DES,* хотя и не является непосредственным обобщением шифра Фейстеля. Для обеспечения криптостойкости алгоритм *Rijndael* включает в себя повторяющиеся раунды, каждый из которых состоит из замен, перестановок и прибавления ключа.

Rijndael – настраиваемый блочный алгоритм, который может работать с блоками из 128, 192 или 256 битов. Применяются ключи шифрования трех фиксированных размеров 128, 192 и 256 битов. В зависимости от размера ключа, конкретный вариант алгоритма обозначается как AES-128, AES-192, AES-256. Для каждой комбинации блока и размера ключа определено свое количество раундов. Наиболее часто используемый вариант алгоритма, при котором блоки, как и ключ, состоят из 128 битов. В этом случае в алгоритме выполняется 10 раундов. При этом осуществляется вычисление 10 подключей K_1 , ... , K_{10}, каждый из которых включает в себя четыре 32-битовых слова.

Существует множество компьютерных алгоритмов шифрования. Ниже рассматриваются наиболее часто применяемые алгоритмы.

4.8. Алгоритм криптографического преобразования ГОСТ 28147-89

Настоящий стандарт (ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИИ ГОСТ 28147-89) устанавливает единый алгоритм криптогра-

фического преобразования для систем обработки информации в сетях электронных вычислительных машин (ЭВМ), отдельных вычислительных комплексах и ЭВМ, который определяет правила шифрования данных и выработки имитовставки. Алгоритм криптографического преобразования предназначен для аппаратной или программной реализации, удовлетворяет криптографическим требованиям и по своим возможностям не накладывает ограничений на степень секретности защищаемой информации.

Стандарт обязателен для организаций, предприятий и учреждений, применяющих криптографическую защиту данных, хранимых и передаваемых в сетях ЭВМ, в отдельных вычислительных комплексах или в ЭВМ.

ГОСТ 28147-89 – отечественный стандарт на шифрование данных. Стандарт включает три алгоритма шифрования (дешифрирования) данных: режим простой замены, режим гаммирования, режим гаммирования с обратной связью и режим выработки имитовставки. С помощью имитовставки можно зафиксировать случайную или умышленную модификацию зашифрованной информации. Вырабатывать имитовставку можно или перед шифрованием (после дешифрирования) всего сообщения, или одновременно с шифрованием (дешифрированием) по блокам. При этом блок информации шифруется первыми шестнадцатью циклами в режиме простой замены, затем складывается по модулю 2 со вторым блоком, результат суммирования вновь шифруется первыми шестнадцатью циклами и т.д.

Алгоритмы шифрования ГОСТ 28147-89 обладают достоинствами других алгоритмов для симметричных систем и превосходят их своими возможностями. Так, ГОСТ 28147-89 (256-битовый ключ, 32 цикла шифрования) по сравнению с такими алгоритмами, как DES (56-битовый ключ, 16 циклов шифрования) и FEAL-1 (64-битовый ключ, 4 цикла шифрования), обладает более высокой криптостойкостью за счет более длинного ключа и большего числа циклов шифрования.

Следует отметить, что, в отличие от DES, у ГОСТ 28147-89 блок подстановки можно произвольно изменять, то есть он является дополнительным 512-битовым ключом, что позволяет увеличить криптостойкость алгоритма.

4.9. Стандарт симметричного шифрования данных IDEA

IDEA (International Data Encryption Algorithm, международный алгоритм шифрования данных) является блочным шифром, он работает с 64-битовыми блоками открытого текста. Длина ключа – 128 битов. Для

шифрования и дешифрирования используется один и тот же алгоритм. Как и другие блочные шифры, IDEA использует и запутывание, и рассеяние. Философия, лежащая в основе проекта, представляет собой "объединение операций из различных алгебраических групп" [6]. Смешиваются три алгебраических группы, и все они могут быть легко реализованы как аппаратно, так и программно:

– XOR;

– сложение по модулю 2^{16};

– умножение по модулю $2^{16}+1$.

Все эти операции (а в алгоритме используются только они, перестановки на битовом уровне не применяются) работают с 16-битовыми подблоками. Этот алгоритм даже эффективнее на 16-битовых процессорах. IDEA является частью PGP.

4.10. Однонаправленная хэш-функция MD5

MD5 (Message Digest, краткое изложение сообщения) – это однонаправленная хэш-функция. MD5 – это улучшенная версия MD4. Хотя она сложнее MD4, их схемы похожи, и результатом MD5 также является 128-битовое хэш-значение. После некоторой первоначальной обработки, MD5 обрабатывает входной текст 512-битовыми блоками, разбитыми на шестнадцать 32-битовых подблоков. Выходом алгоритма является набор из четырех 32-битовых блоков, которые объединяются в единое 128-битовое хэш-значение. Во-первых, сообщение дополняется так, чтобы его длина была на 64 бита короче числа, кратного 512. Этим дополнением является 1, за которой вплоть до конца сообщения следует столько нулей, сколько нужно. Затем к результату добавляется 64-битовое представление длины сообщения (истинной, до дополнения). Эти два действия служат для того, чтобы длина сообщения была кратна 512 битам (что требуется для оставшейся части алгоритма) и чтобы гарантировать, что разные сообщения не будут выглядеть одинаково после дополнения.

4.11. Асимметричный алгоритм шифрования данных RSA

RSA – первый полноценный алгоритм с открытым ключом, который можно использовать для шифрования и цифровых подписей.

Криптосистема RSA разработана в 1977 году и получила название в честь ее создателей: Рона Райвеста (Ron Rivest), Ади Шамира (Adi Shamir) и Леонарда Эйдельмана (Leonard Adleman). Они воспользовались тем фактом, что нахождение больших простых чисел в вычисли-

тельном отношении осуществляется легко, но разложение на множители произведения двух таких чисел практически невыполнимо.

Доказано (теорема Рабина), что раскрытие шифра RSA эквивалентно такому разложению. Поэтому для любой длины ключа можно дать нижнюю оценку числа операций для раскрытия шифра, а с учетом производительности современных компьютеров − оценить и необходимое на это время. Возможность гарантированно оценить защищенность алгоритма RSA стала одной из причин популярности этой криптосистемы на фоне десятков других схем.

Поэтому алгоритм RSA используется в банковских компьютерных сетях, особенно для работы с удаленными клиентами (обслуживание кредитных карточек). В настоящее время алгоритм RSA используется во многих стандартах, среди которых SSL, S-HHTP, S-MIME, S/WAN, STT и PCT.

Основными математическими результатами, положенными в основу этого алгоритма, являются: малая теорема Ферма и функция Эйлера.

Рассмотрим небольшой пример, иллюстрирующий применение алгоритма RSA.

Пример

Зашифруем сообщение "CAB".

Для простоты будем использовать маленькие простые числа (на практике применяются гораздо большие). Выберем $p = 3$ и $q = 11$. Определим $n = 3 \cdot 11 = 33$. Найдем $(p-1)(q-1) = 20$.

Следовательно, в качестве d, взаимно просто с 20, например, $d = 3$. Выберем число e. В качестве такого числа может быть взято любое число, для которого удовлетворяется соотношение $(e \cdot 3) (\mathrm{mod}\, 20) = 1$, например 7.

Представим шифруемое сообщение как последовательность целых чисел с помощью отображения: $A \to 1, B \to 2, C \to 3$. Тогда сообщение принимает вид 3,1,2.

Зашифруем сообщение с помощью ключа $\{7,33\}$:

$$(3^7) (\mathrm{mod}\, 33) = 2187 (\mathrm{mod}\, 33) = 9,$$

$$(1^7) (\mathrm{mod}\, 33) = 1 (\mathrm{mod}\, 33) = 1,$$

$$(2^7) (\mathrm{mod}\, 33) = 128 (\mathrm{mod}\, 33) = 29.$$

Расшифруем полученное зашифрованное сообщение (9,1,29) на основе закрытого ключа $\{3,33\}$:

$$(9^3) (\mathrm{mod}\, 33) = 729 (\mathrm{mod}\, 33) = 3,$$

$$(1^3) (\mathrm{mod}\, 33) = 1 (\mathrm{mod}\, 33) = 1,$$

$(29^3)(\text{mod } 33) = 24389\,(\text{mod } 33) = 2.$

Итак, в реальных системах алгоритм RSA реализуется следующим образом: каждый пользователь выбирает два больших простых числа (p,q), и в соответствии с описанным выше алгоритмом выбирает два простых числа e и d. Как результат умножения первых двух чисел (p,q) устанавливается n, $\{e,n\}$ образует открытый ключ, а $\{d,n\}$ – закрытый (хотя можно и наоборот).

Открытый ключ публикуется, и доступен каждому, кто желает послать владельцу ключа сообщение, которое зашифровывается указанным алгоритмом. После шифрования сообщение невозможно раскрыть с помощью открытого ключа. Владелец же закрытого ключа без труда может расшифровать принятое сообщение.

В настоящее время алгоритм RSA активно реализуется как в виде самостоятельных криптографических продуктов, так и в качестве встроенных средств в популярных приложениях. Важная проблема практической реализации – генерация больших простых чисел.

4.12. Комплекс криптографических алгоритмов PGP

Общие сведения

Комплекс криптоалгоритмов PRETTY GOOD PRIVACY (PGP, весьма хорошая секретность) – программа для ведения секретной переписки, ставшая стандартом де-факто в гражданской криптографии. PGP представляет собой комплект программ, позволяющий шифровать и подписывать электронные сообщения. Реализации PGP имеются для большинства операционных систем:

- Windows (95,NT,2000);
- UNIX (Linux, FreeBSD, и др.);
- MAC;
- MS-DOS;
- BeOS;
- VAX/VMS и других.

PGP умеет встраиваться в популярные почтовые программы для Windows:

- Outlook,
- OutlookExpress,
- Eudora,

что делает его использование достаточно легким для пользователя. Кроме того, поддержка PGP имеется в российской программе для работы с почтой TheBat.

PGP – это свободно распространяемая программа безопасной электронной почты. Разработана Филипом Циммерманном (Philip Zimmermann), а выпущена фирмой Phil's Pretty Good Software [6]. PGP является криптографической системой с высокой степенью секретности. PGP позволяет пользователям обмениваться файлами или сообщениями:

1. с использованием функций секретности;
2. с установлением подлинности;
3. с высокой степенью удобства.

Секретность предполагает прочтение сообщения только адресатом. *Установление подлинности* позволяет определить, что сообщение, полученное от какого-либо человека, было послано именно им, а не кем-нибудь другим. Нет необходимости использовать специальные секретные каналы связи, что делает PGP простым в использовании программным обеспечением. Это связано с тем, что PGP базируется на мощной новой технологии, которая называется *шифрованием с "общим ключом"*.

PGP объединяет в себе удобство использования криптографической системы с общим ключом RSA и скорость обычной криптографической системы, алгоритм "дайджеста сообщений" для реализации электронной подписи, упаковку данных перед шифрованием, хороший эргономический дизайн программы и развитую систему управления ключами. PGP выполняет функции общего ключа быстрее, чем большинство других аналогичных реализаций этого алгоритма.

Как уже было упомянуто, шифрование сообщений и электронная подпись реализована в PGP на основе технологии шифрования с открытым ключом. Для пользователя это значит, что у него (и у каждого, кто пользуется системой) имеется 2 ключа: открытый (или публикуемый) – public key, закрытый (или частный) – private key.

Публикуемый ключ пользователь раздает тем, с кем ведет шифрованную переписку. Этот ключ можно публиковать где угодно – он не содержит пароля. Тот, кто будет писать шифрованное письмо, должен иметь публикуемый ключ своего адресата. Сам ключ представляет собой текстовый файл не очень понятного содержания.

Частный ключ – это ключ, который пользователь хранит у себя и никому не показывает. Только имея свой частный ключ, пользователь расшифровывает электронные письма и убеждается в подлинности подписи.

Отличительная черта технологии шифрования с открытым ключом состоит в том, что пользователь не говорит свой пароль никому

(ведь сам пароль тоже можно украсть, перехватить и т.д.), но при этом пользуется им для шифрованной переписки.

Алгоритм шифрования довольно стоек к взломам. Время на подбор частного ключа на порядок больше времени устаревания информации. Благодаря этому, а также удобству использования и широкой реализации, PGP является фактическим стандартом на работу с конфиденциальной почтой.

Конечно, PGP – это не единственная система шифрования корреспонденции (известна также, например, система VeriSign). Однако технология PGP имеет несколько неоспоримых ***преимуществ.***

- Технология имеется для большинства операционных систем (многоплатформенность).
- Комплект программ является бесплатным и свободно распространяется для всех операционных систем.
- Технология не привязана к какому-либо центральному серверу. Открытые ключи можно передавать как угодно.
- Использовать PGP очень просто, поскольку он встраивается в почтовые программы.

Шифрование с общим ключом

В стандартных криптографических системах, таких как US Federal Data Encryption Standard (DES), один и тот же ключ используется для шифрования и расшифровки. Это значит, что ключ должен первоначально быть передан через секретные каналы так, чтобы обе стороны могли иметь его до того, как шифрованные сообщения будут посылаться по обычным каналам. Это может быть неудобно. Если имеется секретный канал для обмена ключами, тогда зачем нужна криптография?

В криптографической системе с общим ключом каждый имеет два связанных ключа: публикуемый общий ключ и секретный ключ. Каждый из них дешифрует код, сделанный с помощью другого. Знание общего ключа не позволяет вычислить соответствующий секретный ключ. Общий ключ может публиковаться и широко распространяться через коммуникационные сети. Такой протокол обеспечивает секретность без необходимости использовать специальные каналы связи, необходимые для стандартных криптографических систем.

Кто угодно может использовать общий ключ получателя, чтобы зашифровать сообщение ему, а получатель использует его собственный соответствующий секретный ключ для расшифровки сообщения. Никто, кроме получателя, не может расшифровать его, потому что никто больше не имеет доступа к секретному ключу. Даже тот, кто

шифровал сообщение, не будет иметь возможности расшифровать его.

Кроме того, обеспечивается также установление подлинности сообщения. Собственный секретный ключ отправителя может быть использован для шифрования сообщения, таким образом "подписывая" его. Так создается электронная подпись сообщения, которую получатель (или кто-либо еще) может проверять, используя общий ключ отправителя для расшифровки. Это доказывает, что отправителем был создатель сообщения, и что сообщение впоследствии не изменялось кем-либо, так как отправитель – единственный, кто обладает секретным ключом, с помощью которого была создана подпись. Подделка подписанного сообщения невозможна, и отправитель не может впоследствии изменить свою подпись.

Эти два процесса могут быть объединены для обеспечения и секретности, и установления подлинности: сначала подписывается сообщение собственным секретным ключом отправителя, а потом шифруется уже подписанное сообщение общим ключом получателя. Получатель делает наоборот: расшифровывает сообщение с помощью собственного секретного ключа, а затем проверяет подпись с помощью общего ключа отправителя сообщения. Эти шаги выполняются автоматически с помощью программного обеспечения получателя.

В связи с тем что алгоритм шифрования с общим ключом значительно медленнее, чем стандартное шифрование с одним ключом, шифрование сообщения лучше выполнять с использованием быстрого высококачественного стандартного алгоритма шифрования с одним ключом. Первоначальное незашифрованное сообщение называется *"открытым текстом"* (или просто текстом). В процессе, невидимом для пользователя, временный произвольный ключ, созданный только для этого одного "сеанса", используется для традиционного шифрования файла открытого текста. Тогда общий ключ получателя используется только для шифровки этого временного произвольного стандартного ключа. Зашифрованный ключ "сеанса" посылается, наряду с зашифрованным текстом (называемым ciphertext – зашифрованный), получателю. Получатель использует свой собственный секретный ключ, чтобы восстановить этот временный ключ сеанса, и затем применяет его для выполнения быстрого стандартного алгоритма декодирования с одним ключом, чтобы декодировать все зашифрованное сообщение.

Сертификаты ключей

Общие ключи хранятся в виде "сертификатов ключей", которые включают в себя:

- идентификатор пользователя владельца ключа (обычно это имя пользователя);
- временную метку, которая указывает на время генерации пары ключей;
- собственно ключи.

Сертификаты общих ключей содержат общие ключи, а сертификаты секретных ключей – секретные. Каждый секретный ключ также шифруется с отдельным паролем. Файл ключей содержит один или несколько таких сертификатов. В каталогах общих ключей хранятся сертификаты общих ключей, а в каталогах секретных – сертификаты секретных ключей.

На ключи также внутренне ссылаются "идентификаторы ключей", которые являются "сокращением" общего ключа (самые младшие 64 бита большого общего ключа). Когда этот идентификатор ключа отображается, то показываются лишь младшие 24 бита для краткости. Если несколько ключей могут одновременно использовать один и тот же идентификатор пользователя, то никакие два ключа не могут использовать один и тот же идентификатор ключа.

Дайджесты сообщений

PGP использует "дайджесты сообщений" для формирования подписи. **Дайджест сообщения** – это криптографически мощная 128-битная односторонняя хэш-функция от сообщения. Она несколько напоминает контрольную сумму, или CRC-код. Она однозначно представляет сообщение и может использоваться для обнаружения изменений в сообщении. В отличие от CRC-кода (контроля циклическим избыточным кодом), дайджест не позволяет создать два сообщения с одинаковым дайджестом. Дайджест сообщения шифруется секретным ключом для создания электронной подписи сообщения.

Документы подписываются посредством добавления перед ними удостоверяющей подписи, которая содержит идентификатор ключа, использованного для подписи; подписанный секретным ключом дайджест сообщения и метку даты и времени, когда подпись была сгенерирована. Идентификатор ключа используется получателем сообщения, чтобы найти общий ключ для проверки подписи. Программное обеспечение получателя автоматически ищет общий ключ отправителя и идентификатор пользователя в каталоге общих ключей получателя.

Шифрованным файлам предшествует идентификатор общего ключа, который был использован для их шифрования. Получатель использует этот идентификатор для поиска секретного ключа, необходимого для расшифровки сообщения. Программное обеспечение получателя автоматически ищет требуемый для расшифровки секретный ключ в каталоге секретных ключей получателя.

Эти два типа каталогов ключей и есть главный метод сохранения и управления общими и секретными ключами. Вместо того чтобы хранить индивидуальные ключи в отдельных файлах ключей, они собираются в каталогах ключей для облегчения автоматического поиска ключей либо по идентификатору ключа, либо по идентификатору пользователя. Каждый пользователь хранит свою собственную пару каталогов ключей. Индивидуальный общий ключ временно хранится в достаточно большом отдельном файле.

Алгоритмы секретных ключей

PGP предлагает на выбор различные алгоритмы секретных ключей. Под *алгоритмом секретного ключа* понимается симметричный блочный шифр, использующий один ключ для шифрования и дешифрирования. Могут быть использованы следующие три симметричных блочных шифра:

- CAST,
- Triple-DES,
- IDEA.

На рис. 4.15 приведен пример диалоговой панели для установления свойств PGP.

Все три шифра работают с 64-битовыми блоками открытого текста и шифротекста. Длина ключа для CAST и IDEA – 128 битов, в то время как Triple-DES использует 168-битовый ключ. Подобно Data Encryption Standard (DES), любой из этих шифров может быть использован в режимах CFB и CBC. Криптографический **режим** обычно объединяет базовый шифр, какую-то обратную связь и ряд простых операций. FIPS PUB 81 определяет четыре режима работы: ECB, CBC, OFB и CFB. Банковские стандарты ANSI определяют для шифрования ECB и CBC, а для проверки подлинности – CBC и n-битовый CFB.

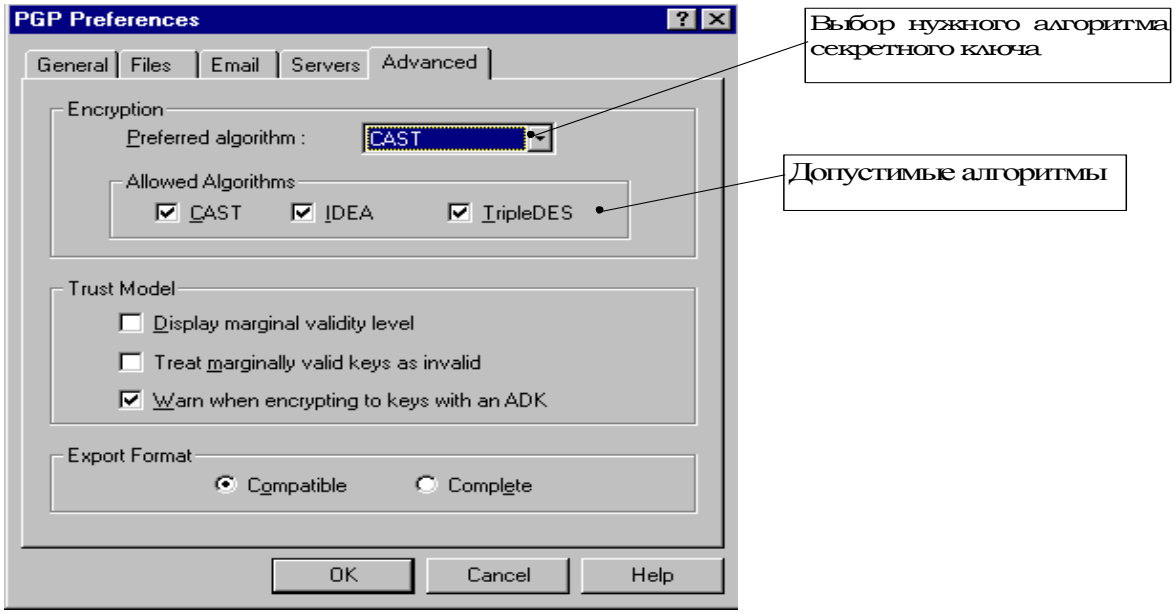

Рис. 4.15. Пример диалоговой панели для установления свойств PGP

Распределенный подход к управлению ключами

Самой интересной особенностью PGP является *распределенный подход к управлению ключами*. Центров сертификации ключей нет, вместо этого в PGP поддерживается "сеть доверия". Каждый пользователь сам создает и распространяет свой открытый ключ. Пользователи подписывают ключи друг друга, создавая взаимосвязанное сообщество пользователей PGP. Таким образом, распределенное управление ключами реализуется с помощью поручителей. Поручители – это пользователи системы, которые подписывают открытые ключи своих друзей.

Варианты PGP

Одной из свободно распространяемых версией PGP является 6.0.2. 7-я версия PGP называется "PGP Desktop" [3, 6]. В этой версии, кроме основного предназначения PGP, добавлены следующие модули (программы):

- PGP Disk – система шифрования данных на жестких дисках и других носителях.
- PGP IGO – система шифрования данных, передаваемых с помощью ICQ.
- PGP Net – система шифрования трафика и пакетный фильтр.

Существуют также 8 и 9 версии PGP.

Все файловые популярные системы (FAT, FAT32, NTFS, ext2, ufs), применяющиеся в Windows, Linux, FreeBSD, позволяют прочитать данные при физическом доступе к носителю. Т.е. если у Вас в руках оказался винчестер – то, если не приняты специальные меры, данные с него можно прочитать. Поэтому, серверы располагаются обычно в недоступных местах – ведь современные файловые системы обеспечивают защиту при сетевом доступе, но не при физическом.

Менеджер зачастую разъезжает с ноутбуком. На ноутбуке могут содержаться секретные данные. Представим, что случилась неприятность – ноутбук украли. Практически гарантированно злоумышленники получат информацию с жесткого диска. И стоимость этой информации может быть на несколько порядков выше стоимости самого ноутбука.

Именно эти проблемы очень удобно решает PGP Disk. На жестком диске создается файл, который при штатной работе отображается в логический диск. Вся информация в файле шифрованная, т.е, получив этот файл и не зная пароля, его практически невозможно расшифровать. Превращение файла в диск может сделать либо владелец ключа, либо человек, который ввел пароль. Система PGP Disk разработана под Windows-операционные системы. И в реализации под UNIX ее нет. Под UNIX-системы существуют реализации шифрованных файловых систем (secure filesystems).

PGP Net призвано строить частные виртуальные сети с шифрованным каналом, а также является пакетным фильтром (Firewall). В отличие от других продуктов из PGP серии, имеется множество реализаций как VPN, так и FireWalls. Системы VPN и FireWall встроены в Windows.

Пример использования PGP

Главное свойство технологии PGP заключается в том, что она позволяет шифровать и подписывать электронные письма. Удивительно, но все письма, которые пересылаются через Internet, ходят в открытом виде. Что это значит?

Письмо, посланное из точки А в точку Б, проходит через несколько Internet-серверов, на которых письмо можно без особого труда отловить и прочитать. Есть еще один аспект электронной почты: подделать имя пославшего сообщения не представляет никакого труда. Поле From является лишь информативным полем и легко изменяется (для этого даже не нужно быть администратором сервера).

Напрашивается следующий вывод: если пользователь согласен, чтобы его почта могла прочитываться третьим лицом, то использование PGP не нужно. В противном же случае PGP – очень удобное

средство. Делается это средствами PGP очень просто – нажатием двух кнопочек в Outlook и при отправке письма введением пароля.

Краткое изложение основных характеристик PGP

PGP – это протокол, который определяет правила взаимодействия существующих алгоритмов. PGP интегрирует известные криптосистемы для более надежной защиты. Реализацию PGP можно считать удавшейся, так как он до сих пор не был взломан.

PGP обладает следующими характеристиками:

- основывается на алгоритме RSA (шифрование с общим ключом);
- использует для различных целей такие алгоритмы, как IDEA, DES, Triple-DES, CAST, MD5;
- применяется распределенное управление ключами (с помощью поручителей);
- ключи хранятся в виде сертификатов ключей (в двух каталогах);
- шифрует и дешифрирует документы (файлы);
- можно обеспечить не только секретность сообщений, но и их подлинность (подпись сообщений);
- некоторые версии не могут быть использованы вне США;

5. КОМПЬЮТЕРНАЯ БЕЗОПАСНОСТЬ И ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ КРИПТОГРАФИИ

5.1. Общие сведения

Особую актуальность использование криптографических средств защиты информации приобрело в связи с предоставлением ряда банковских услуг через Internet (Internet-banking), а также построением корпоративных виртуальных сетей передачи данных. Активное развитие криптографических методов и инструментов способствовало появлению новой области человеческой деятельности – электронной коммерции.

Следует отметить, что при построении отечественных информационно-телекоммуникационных систем (ИТС) на основе импортных программно-аппаратных средств возникают проблемы с обеспечением безопасности таких систем. Объясняется это тем, что:

- импортные компоненты не могут считаться безопасными (доверенными);

- использование открытых сетей передачи данных позволяет проводить удалённые атаки.

Проблемы, связанные с обеспечение информационной безопасности в современных ИТС (имеющих как распределённый, так и локальный характер), обычно включают:

1. Защиту информационных ресурсов локальной рабочей станции (ЛРС) от несанкционированного доступа.
2. Защиту в локальных сетях передачи данных.
3. Защиту межсетевого взаимодействия:
   - создание защищённых виртуальных сетей (VPN);
   - обеспечение защиты технологии клиент/сервер;
   - обеспечение защиты информационных ресурсов корпоративной сети, имеющей выход в общедоступные сети передачи данных, от атак извне;
   - средств защиты пользовательского взаимодействия типа «абонент А – абонент В».
4. Защиту электронной почты и документооборота.
5. Защиту электронных платёжных систем.

Встраивание программных средств защиты информации может осуществляться одним из следующих способов:

- с использованием программных интерфейсов;
- с использованием криптосервера.

Основная проблема встраивания заключается в корректном использовании вызываемых функций.

Программным интерфейсом (далее API) называется детальное описание функций и используемых ими параметров. Формат обращения к функциям, описанный в программном интерфейсе, поддерживает прикладное ПО. Для того чтобы интегрировать защитный механизм в данное ПО, необходимо согласовать форматы функций (вызываемых и реализованных).

Наиболее известными API являются Crypto API, GSSAPI и SSPI. В ОС Windows NT 4.0 используется программный интерфейс Crypto API, дающий возможность шифрования (RC2, RC4, DES, RSA), выработки и проверки электронной надписи, однако в этой системе нет действующих программных средств с механизмами криптографической защиты [2]. Необходимо отметить, что алгоритмы шифрования, применяемые в Crypto API, используют укороченные ключи.

Подход к решению проблемы защиты на основе криптографического сервера заключается в локализации средства защиты информации, работающего с важнейшей информацией (секретные ключи, пароли и

т.д.) на основе выделенной рабочей станции. В прикладное ПО, работающее на другой (находящейся на связи) рабочей станции, встраиваются только вызовы функций, реализованных в ПО криптосервера. Использование криптосервера позволяет выполняться средству защиты информации в доверенной программной среде.

Остановимся подробнее на существующей классификации уровней защиты информации.

Физический и канальный уровни. На этих уровнях в качестве механизмов безопасности обычно осуществляется шифрование соединения или трафика (зашифровываться может как весь трафик, так и выборочная часть), т.е. обеспечивается конфиденциальность передаваемой информации. В качестве вероятных угроз здесь можно выделить следующие:

- несанкционированное подключение;
- ошибочная коммутация;
- прослушивание;
- перехват;
- фальсификация информации;
- имитоатаки;
- физическое уничтожение канала связи.

Для защиты информации на данном уровне обычно применяют шифрующие модемы, специализированные канальные адаптеры. К достоинствам реализации средств защиты на уровнях этого типа можно отнести:

- простоту применения;
- аппаратную реализацию;
- полную защиту трафика;
- прозрачность выполнения средствами защиты информации своих функций.

К недостаткам применения средств защиты на канальном или физическом уровне следует отнести:

- негибкость решения (фиксированный тип канала, сложность адаптации к сетевой топологии, фиксированная производительность);
- низкая совместимость;
- высокая стоимость.

Сетевой уровень. При защите информации на сетевом уровне необходимо решить следующие задачи:

1. Защита информации непосредственно в пакетах, передаваемых по сети.

2. Защита трафика сети, то есть шифрование всей информации в канале.
3. Контроль доступа к ресурсам сети, т.е. защита от нелегальных пользователей и от доступа легальных пользователей к информации, им не предназначенной.

Реализация средств защиты на сетевом уровне представляет больше возможностей для обеспечения безопасности передаваемых данных. Например, на данном уровне может быть реализована аутентификация рабочей станции, являющейся источником сообщений. Из угроз, специфичных для сетевого уровня, следует выделить:

- анализ служебной информации сетевого уровня, т.е. адресной информации и топологии сети;
- атаки на систему маршрутизации;
- фальсификации адресов; атаки на систему управления;
- прослушивание, перехват и фальсификацию информации;
- имитоатаки.

Для устранения перечисленных угроз применяются следующие решения:

- пакетная фильтрация;
- административная защита на маршрутизаторах (в том числе шифрующих);
- протоколы защиты информации сетевого уровня (защита и аутентификация трафика);
- динамическое распределение сетевых адресов;
- защиту топологии.

К достоинствам средств защиты на сетевом уровне относятся:

- полнота контроля трафика;
- универсальность;
- прозрачность;
- совместимость;
- адаптивность к сетевой технологии.

К недостаткам средств защиты на сетевом уровне можно отнести только неполноту контролируемых событий (неподконтрольность транспортных и прикладных протоколов). Сетевой уровень – это та ступень организации, на которой сеть становится полносвязной системой. На более низких уровнях защита может быть реализована только как набор двухточечных защищённых звеньев.

Только на сетевом уровне появляется возможность установления защищённого соединения между двумя компьютерами, расположенными в произвольных точках сети; на этой ступени появляется и понятие

топологии, можно различить внешние и внутренние каналы. При сетевом взаимодействии реализуются такие возможности защиты информации, как фильтрация трафика между внутренней (корпоративной) сетью и внешней коммуникационной средой, защита от несанкционированного доступа из внешней сети во внутреннюю, маскировка топологий внутренних сетей.

Транспортный уровень. Самыми опасными угрозами на этом уровне следует считать:

- несанкционированные соединения, разведку приложений;
- атаки на систему управления;
- прослушивание, перехват и фальсификацию информации;
- имитоатаки.

Традиционными решениями подобных проблем являются:

- защита в составе межсетевых экранов (контроль доступа к приложениям, управление доступом к серверам);
- proxy-системы;
- протоколы защиты транспортного уровня (защита и аутентификация данных).

Среди достоинств средств защиты на транспортном уровне можно выделить следующие:

- развитая функциональность;
- высокая гибкость системы.

Недостатками средств защиты являются:

- неполнота защиты;
- неподконтрольность событий в рамках прикладных и сетевых протоколов.

Прикладной уровень. При создании и пересылке сложного электронного документа, состоящего из нескольких полей, необходимо обеспечить защиту какого-то отдельно взятого поля данных. Решение этой задачи может заключаться в применении средств криптографической защиты на прикладном уровне. Информация, зашифрованная на прикладном уровне, затем разбивается на пакеты и сетевые кадры, надёжность доставки которых обеспечивается транспортной средой, – следовательно, криптографическая защита объекта прикладного уровня должна быть действительной и для всех нижестоящих уровней.

Необходимо отметить, что при защите информации на прикладном уровне процедуры передачи, разборки на пакеты, маршрутизации и обратной сборки не могут нанести ущерба конфиденциальности информации. Кроме того, на прикладном уровне существует возможность обеспечения аутентификации пользователя, породившего информацию.

Из основных угроз, возникающих на прикладном уровне, следует отметить:

- несанкционированный доступ к данным;
- разведку имен и паролей пользователей;
- атаки на систему разграничения прав доступа пользователей;
- маскировку под легитимного пользователя;
- атаки на систему управления, атаки через стандартные прикладные протоколы;
- фальсификацию информации;
- имитоатаки.

Традиционно применяемыми решениями по защите информации и информационных ресурсов прикладного уровня являются:

- встроенная защита приложений;
- межсетевые экраны с фильтрацией сетевых прикладных протоколов;
- proxy-системы и т.д..

Криптографическая защита информации на прикладном уровне является наиболее предпочтительным вариантом защиты информации с точки зрения её гибкости, однако эти меры могут оказаться наиболее сложными в части программно-технической реализации.

5.2. Обзор стандартов в области защиты информации

Международные стандарты. Особое место в современных ИТС занимают вопросы стандартизации методов и средств защиты информации и информационных ресурсов. Международные стандарты представлены серией ISO и ISO/IES и выпущены в свет Международной организацией по стандартизации (ISO) и Международной электротехнической комиссией (IES).

ISO 7498-2 (X.800). Стандарт посвящен описанию архитектуры безопасности по отношению к модели взаимодействия открытых систем (OSI), включая описание расположения механизмов и служб безопасности на уровнях OSI.

ISO 8372. Этот стандарт описывает режимы блочного шифрования:
а) режим электронной книги (ECB);
б) режим сцепления блоков (CBC);
в) режим с обратной связью по шифротексту (CFB);
г) режим с обратной связью по выходу (OFB). Впервые был опубликован в 1987 г.

ISO 8730. Данный стандарт совместно со стандартом ISO 8731 описывает процедуру использования алгоритмов генерации MAC в банковских системах и является международным аналогом стандарта ANSI X9.9. В нём вводятся независимые от алгоритмов методы и требования использования MAC, включая форматы представления данных, а также способ, при помощи которого данный стандарт может быть применён к выбранному алгоритму.

ISO 8732. Стандарт посвящен управлению ключами в банковских системах и является аналогом стандарта ANSI X9.17.

ISO 9564. Стандарт посвящен методам управления и обеспечения безопасности персонального идентификационного номера (PIN). В части 9564-1 раскрываются принципы и средства защиты PIN в течение его жизненного цикла, позволяющие сохранить его втайне от злоумышленников, а часть 9564-2 посвящена использованию алгоритмов шифрования для защиты PIN.

ISO/IES 9594-8 (X.509). Данный стандарт посвящен двум типам аутентификации – простой и строгой. Представленная строгая аутентификация состоит из двух и трёх передаваемых сообщений и основана на использовании ЭЦП и параметров, зависящих от времени. В стандарте также описывается процедура аутентичного распределения открытых ключей на основе сертификатов в формате X.509.

ISO/IES 9797. Этот стандарт описывает алгоритм генерации кодов аутентификации сообщений (MAC), основанный на использовании алгоритма блочного шифрования.

ISO/IES 9798. Данный стандарт состоит из пяти частей. В первой части (9798-1) описывается механизм аутентификации пользователей, основанный на использовании симметричного алгоритма шифрования (9798-2), алгоритм генерации ЭЦП с асимметричными ключами (9798-3), криптографической функции проверки целостности или MAC (9798-4), некоторых дополнительных механизмов (9798-5).

ISO/IES 10181 (X.810-X.816). Стандарт состоит из описания серии архитектур безопасности, а именно: архитектура аутентификации, архитектура контроля доступа, архитектура обеспечения неотрицаемости и разбора конфликтных ситуаций, архитектура обеспечения целостности, архитектура обеспечения конфиденциальности и архитектура аудита систем безопасности.

ISO 11166. Стандарт состоит из нескольких частей и описывает асимметричные механизмы распределения симметричных ключей. Часть 11166-1 посвящена основным принципам и процедурам распределения ключей и форматам представления ключей, а также сертифика-

ции ключевой информации. Часть 11166-2 описывает применение RSA для шифрования и генерации ЭЦП.

ISO 11568. Стандарт описывает средства и процедуры управления ключами при финансовых операциях для симметричных и асимметричных алгоритмов.

ISO/IES 13888. Данный стандарт посвящен проблеме неотрицаемости ряда фактов, связанных с процессом передачи сообщений. В нём также приводятся механизмы, разрешающие случаи, связанные с отказом от факта отправки сообщения, отказа от факта приёма сообщения, а также механизмы неотрицаемости, связанные с использованием доверенной третьей стороны.

ISO/IES 14888. Стандарт состоит из нескольких частей и посвящен построению схем ЭЦП. Часть 14888-1 знакомит с общепринятыми определениями и моделями построения схем ЭЦП. В части 14888-2 рассматриваются схемы ЭЦП, в которых ключ проверки подписи является результатом общедоступной функции от информации, идентифицирующей подписывающего. Часть 14888-3 посвящена распределению открытых ключей при помощи сертификатов открытых ключей.

Стандарты серии ANSI. В этом пункте перечислены стандарты, разработанные Американским национальным институтом стандартов (ANSI) и посвящённые криптографическим алгоритмам и их применению в банковских системах.

ANSI X9.92. Стандарт специфицирует DES-алгоритм, который в рамках этого стандарта представляется как алгоритм шифрования данных (Data Encryption Algorithm-DEA).

ANSI X3.106. Стандарт описывает виды применения алгоритма DES.

ANSI X9.9. Стандарт описывает применение MAC на основе использования DES в широком круге банковских систем.

ANSI X9.17. Данный стандарт основан на стандарте ISO 8732 и посвящён процедурам управления ключами, а также средствам распределения и защиты ключей применительно к банковским системам. В нём нашли отражение специфические аспекты использования ключей, такие как: счётчики ключей, преобразование ключей и подтверждение подлинности ключей.

ANSI X9.19. Стандарт посвящён вопросам использования алгоритма DES для генерации MAC в конкретных типах банковских систем.

ANSI X9.23. Стандарт описывает форматы представления данных при использовании алгоритма DES в банковских системах и методы обработки получаемых из каналов связи данных.

ANSI X9.24. Данный стандарт описывает методы управления ключами, аутентификацию (основанную на DES) и шифрование PIN, ключей и других данных, а также руководящие принципы защиты ключей на всех этапах жизненного цикла.

ANSI X9.28. Этот стандарт является продолжением стандарта ANSI X9.17 и описывает процедуры распределения ключей между пользователями, которые не имеют между собой и третьей стороной предварительного распределения ключевой информации.

ANSI X9.31. Стандарт описывает алгоритм генерации и проверки ЭЦП на основе использования алгоритма RSA [2].

Существует также большое количество стандартов в области безопасности Internet, например RFC, разработанный IETF (Internet Engineering Steering Group), и стандарты PKCS (Public-key Cryptography Standards) [2].

Ниже излагается расшифровка некоторых терминов, используемых в описании стандартов.

IP – адрес. **API** – программный интерфейс – детальное описание функций и используемых ими параметров. Формат обращения к функциям, описанный в программном интерфейсе, поддерживает прикладное программное обеспечение. **MAC** – алгоритм генерации кодов аутентификации сообщений (основанный на использовании алгоритмов блочного шифрования). **PIN** – персональный идентификационный номер, **трафик** – объём передаваемой информации. **Межсетевой экран** реализуется в виде контроля доступа к приложениям, управления доступом к серверам.

5.3. Подсистема информационной безопасности

Понятие подсистемы информационной безопасности (ПИБ) включает в себя весь комплекс средств и мер по защите информации в ИТС.

Типовая ПИБ строится с учетом следующих принципов:

1. Применяемые в ПИБ средства и технологии защиты должны обладать свойствами модульности, масштабируемости, открытости и возможности адаптации системы к различным условиям функционирования.
2. ПИБ ИТС должна предполагать полную независимость функционирования каждого из комплексов защиты. Нарушение функционирования любого компонента не должно приводить к изменению других характеристик защиты, тем более к отказу всей системы.

3. Применяемые средства и технологии должны обеспечивать открытость архитектуры, наращиваемость, а также предоставлять интерфейс для подключения внешних систем защиты информации.

Основные задачи подсистемы защиты локальных рабочих мест:

1. Разграничение доступа к ресурсам пользователей автоматизированных рабочих мест (АРМ) ИТС.
2. Криптографическая защита хранящейся на АРМ информации (конфиденциальность и целостность).
3. Аутентификация пользователей и авторизация их действий.
4. Обеспечение невозможности влияния программно-технического обеспечения АРМ на регламент функционирования прикладных процессов ИТС.

Основные задачи подсистемы защиты локальных вычислительных сетей (ЛВС). Для данной подсистемы характерна не только защита от несанкционированного доступа (НДС) к информации и информационным ресурсам в рамках ЛВС, но также и криптографическая защита информации, передаваемой в ЛВС на прикладном, системном уровне и на уровне структурированной кабельной системы.

Основные задачи подсистемы защиты межсетевого взаимодействия. Для этой подсистемы основным является защита ИТС от негативных воздействий со стороны внешних сетей передачи данных. Подсистема должна обеспечивать не только защиту информации (конфиденциальность и целостность), передаваемой во внешнюю среду, но также и защиту от внешнего разрушающего воздействия информации, находящейся в ИТС.

Основные задачи подсистемы аудита и мониторинга:

1. Осуществление централизованного, удалённого, автоматизированного контроля работы подконтрольных серверов, АРМ и других подсистем ПИБ.
2. Централизованное ведение протокола всех событий, происходящих на подконтрольных серверах, АРМах и подсистемах.
3. Выполнение автоматизированного анализа механизма принятия решений в нештатных ситуациях.

Основные задачи подсистемы технологической защиты:

1. Выработка принципов построения технологического процесса обработки информации в ИТС.
2. Определение контрольных точек технологического процесса.
3. Определение мест использования средств и методов защиты информации для регистрации, подтверждения и проверки

прохождения информации через контрольные точки технологического процесса.

4. Исключение сосредоточения полномочий у отдельных должностных лиц.

5.4. Защита локальной рабочей станции

Под локальной рабочей станцией (ЛРС) подразумевается компьютер, не подсоединённый к каналам передачи данных. Любая защита в современных информационных системах начинается, прежде всего, с конкретного рабочего места. По статистике, большинство нарушений приходится именно на внутренних нарушителей.

Задачи обеспечения информационной безопасности осложняются тем, что функционирование средств защиты информации происходит в **недоверенной программной среде**. Имеется в виду среда, в которой невозможно однозначно доказать отсутствие влияния программного окружения на функционирование средств защиты информации. Недоверенность программной среды вытекает из того факта, что на сегодняшний день ПО, установленное на ЛРС, чаще всего зарубежного производства, функциональный состав которого в большинстве случаев является неопределённым. При этом под ПО подразумевается как прикладное программное обеспечение, так и обеспечение безопасности ОС.

Следует отметить, что наибольшую опасность представляют недокументированные возможности и закладки, находящиеся в ядре ОС. Так, например, в ОС Windows NT и в интерфейсе WIN 32 имеется ряд недокументированных возможностей.

Угрозы и задачи информационной безопасности для локальных рабочих станций. Потенциальных нарушителей можно разделить на следующие категории:

- зарегистрированные пользователи ЛРС;
- имеющие доступ к штатным средствам управления ЛРС (клавиатура, устройства считывания информации и т.д.);
- пользователи, не имеющие физического доступа к ЛРС.

При этом нарушители обычно ставят перед собой следующие цели:

- компрометация секретной информации, в том числе и информации, относящейся к работе средств защиты информации ЛРС (ключевая информация, пароли пользователей и т.д.);
- изменение или уничтожение любой секретной информации;
- нарушение работоспособности всей системы в целом или её отдельных компонентов.

Под нарушениями имеются в виду не только умышленные действия, но и неумышленные нарушения и ошибки обслуживающего персонала.

Обобщённый перечень угроз можно классифицировать по следующим признакам.

1) По характеру доступа:

- с доступом к ПО;
- с доступом только к аппаратным ресурсам;
- без доступа к ЛРС.

Доступ к ПО подразумевает, что нарушитель имеет возможность взаимодействовать с установленным ПО, которое доступно для модификации и анализа. Доступ к аппаратному обеспечению подразумевает воздействие нарушителя на аппаратную часть с целью введение её в режим, нарушающий надёжное функционирование средств защиты информации. Отсутствие доступа к ЛРС подразумевает, что нарушитель имеет возможность воздействовать на систему только через каналы передачи информации, возникающие в ходе работы ЛРС. В качестве таких каналов можно отметить:

- электромагнитный канал;
- цепи заземления и питания;
- виброакустический канал.

Например, электромагнитный канал возникает вследствие появления электромагнитного излучения от ЛРС. Данное излучение модулируется в соответствии с процессами обработки информации, среди которых могут быть и процессы, обрабатывающие секретную информацию.

2) По характеру проявления:

– активные

– пассивные.

Активные проявления характеризуются тем, что нарушитель является инициатором негативных воздействий на систему (воздействия могут быть как умышленными, так и неумышленными); пассивные же угрозы заключаются в анализе информации, возникающей в ходе функционирования ЛРС.

3) По используемым в ходе реализации угрозы средствам:

– с использованием штатных средств, входящих в состав ЛРС;

– с использованием дополнительных угроз.

5.5. Методы и средства обеспечения информационной безопасности локальных рабочих станций

Создание защищённой системы является комплексной задачей. Конкретные средства защиты информации реализуют только типовые функции по её защите. Реальная защитная система строится, исходя из возможных угроз и выбранной политики безопасности.

На основе вышеизложенного перечня угроз можно предложить следующие меры по защите данных:

1. Обеспечение конфиденциальности и достоверности пользовательской информации. Реализуется на основе применения средств шифрования и ЭЦП, выполненных как в виде абонентского шифрования (отдельная программа-вызов, запуск которой предоставляет пользователю возможность применять функции шифрования в ЭЦП), так и в виде средств прозрачного шифрования. Например, вызовы функций шифрования встраиваются в используемое пользовательское ПО, в ходе работы которого незаметно для пользователя происходит вызов этих функций из прикладного ПО.
2. Разграничение доступа пользователей к информации, хранящейся и обрабатываемой на ЛРС при применении многопользовательского режима. Обычно используются штатные средства современных ОС.
3. Аутентификация пользователей и авторизация доступа к защищаемым объектам с использованием криптографических методов.
4. Контроль целостности секретной пользовательской или системной информации, основанный на применении программ генерации контрольных сумм и их проверки (бесключевые хэш-функции или имитовставки).
5. Контроль процесса загрузки ОС, при котором осуществляется загрузка строго определённой версии ОС и блокировка несанкционированной загрузки с внешних носителей или из сети.
6. Контроль целостности аппаратных ресурсов, который необходимо осуществлять непосредственно перед тем, как пользователь начнёт работать с ЛРС.
7. Исключение негативного влияния на процесс функционирования пользовательского приложения со стороны программного окружения, установленного на ЛРС.
8. Контроль запуска задач пользователем, т.е. пользователю для запуска должны быть доступны только его задачи.

9. Защита от побочного электромагнитного излучения и утечки информации по цепям питания и заземления.
10. Физическое удаление остаточной информации, возникающей в ходе функционирования ПО (уничтожение временных файлов и т.д.).
11. Аудит и протоколирование работы средств защиты информации, системного и прикладного ПО.

Аудит. Аудит связан с действиями, затрагивающими безопасность системы. К их числу относятся:

- вход в систему (успешный или нет);
- выход из системы;
- обращение к удалённой системе;
- операции с файлами (открыть, закрыть, переименовать, удалить);
- смена привилегий или иных атрибутов безопасности (режима доступа и т.п.).

Полный перечень событий, подлежащих регистрации, зависит от выбранной политики безопасности и от общей спецификации системы.

При протоколировании события необходимо записывать:

1) дату и время события;
2) уникальный идентификатор;
3) отмечать пользователя, являющегося инициатором действия;
4) тип события;
5) результат действия (успех или неудача);
6) источник запроса (например, имя терминала);
7) имена затронутых объектов (например, файлов);
8) описания изменений, внесённых в базы данных защиты (например, новая метка безопасности объекта);
9) метки безопасности субъектов и объектов события.

При такой организации система аудита может фиксировать все события, связанные с функционированием прикладного и системного ПО. Анализ реализации подсистем аудита в современных ОС (операционных системах) показывает, что ядро системы (являющееся доверенным) взаимодействует с прикладным ПО (работа которого подлежит аудиту) через интерфейс обращений к данному сервису ОС и является наиболее удобным местом для осуществления мониторинга и аудита. Ядро ОС при обращении к нему прикладного ПО фиксирует события, подлежащие аудиту, а подсистема аудита заносит их в журнал событий, хранящийся на жёстком диске.

Генерировать события может и прикладное ПО, но, в отличие от ядра ОС, данные действия не заносятся напрямую в журнал событий, а передаются в ядро системы, которое отправляет их в буфер (отведённый под запись событий) и впоследствии фиксирует в журнал событий. При осуществлении записи или чтения в данный буфер, производится его блокировка с целью запрещения одновременного чтения из него и записи в него. К каждому событию, помещённому в буфер, дописывается метка времени, последовательный номер и идентификатор процесса, породившего данное событие.

Общая схема архитектуры построения системы аудита представлена на рис. 5.2.

Необходимо подчеркнуть важность не только сбора информации, но и её регулярного и целенаправленного анализа. В этом плане удобно использовать средства аудита СУБД, поскольку к регистрационной информации могут естественным образом применяться SQL-запросы.

5.6. Защита в локальных сетях

Поскольку все глобальные сети передачи данных представляют собой объединение локальных сетей (ЛВС), имеющих различную топологию построения и использующих различные сетевые протоколы (TPC/IP, IPX/SPX, Netbios, DecNet и др.), безопасность ЛВС имеет первостепенное значение.

Общие вопросы безопасности в ЛВС. Среди основных угроз для ЛВС следует отметить следующие:

1) анализ сетевого трафика с целью получения доступа к конфиденциальной информации, например к передаваемым в открытом виде по сети пользовательским паролям;
2) нарушение целостности передаваемой информации;
3) получение несанкционированного доступа к информационным ресурсам;
4) попытка совершения ряда действий от имени зарегистрированного пользователя в системе.

Обеспечение защиты в соответствии с заданной политикой безопасности в ЛВС является комплексной задачей и осуществляется при помощи:

1) корректного администрирования сетевых настроек ОС;
2) дополнительных защитных механизмов – шифрования, ЭЦП, аутентификации сторон и т.д.;
3) организационных методов защиты и контроля за их неукоснительным соблюдением, например с использованием системы

аудита.

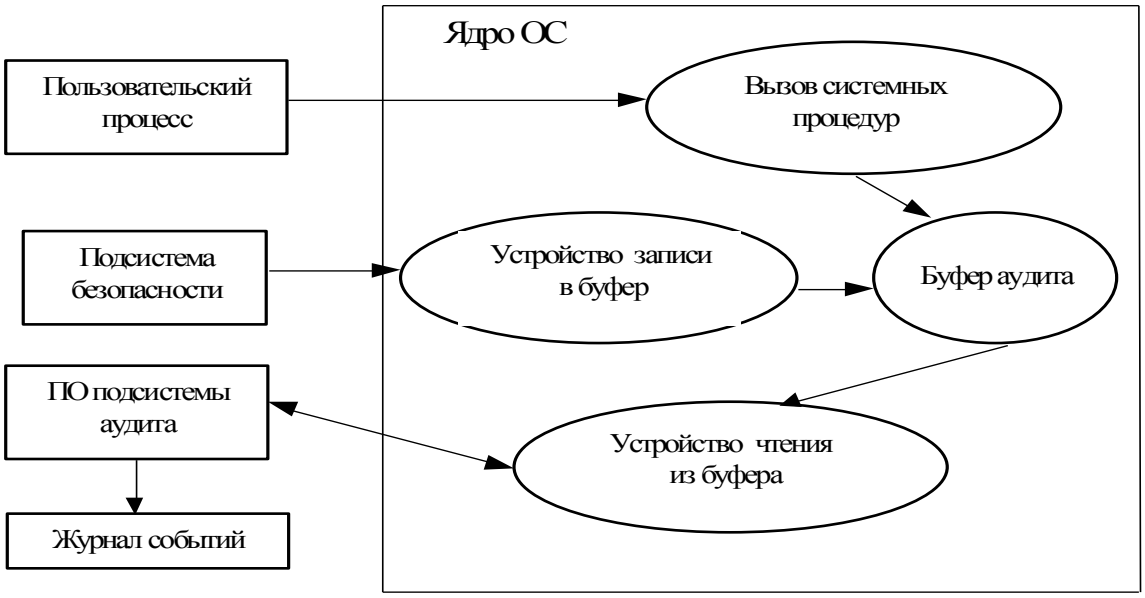

Рис. 5.2. Архитектура системы аудита

Безопасность сетей Novell NetWare. В ОС Novell NetWare возможны атаки типа «человек-в-середине» при условии, что нарушитель способен просматривать пакеты, пересылаемые между клиентом и сервером. Для этого он может внедрить ложный ARP-сервер. Поэтому является необходимым как применение дополнительных средств защиты, так и выполнение следующих рекомендаций:

1) физически защитить сервер;
2) защитить критичные файлы системы, например вычислить контрольные суммы на файлы, находящиеся в SYS: LOGIN, SYS: PUBLIC и SYS: SYSTEM, и постоянно производить проверку полученных контрольных сумм;
3) доступ пользователей к ресурсам должен быть организован в соответствии с выбранной политикой безопасности;
4) включить accounting, посредством которого можно регистрировать все попытки произвести login и logout к серверу;
5) стараться не применять процедуру RCONSOLE (удалённой консоли), которая является не только удобным средством, позволяющим осуществлять удалённое управление сервером (включая загрузку и выгрузку), но и вполне доступной мишенью для нарушителя;
6) добавить команду EXIT в System Login Script.

Безопасность в сетях Windows. Из достаточно большого числа

уязвимостей ОС Windows, связанных как с непроработанностью вопросов сетевой безопасности, ошибками при проектировании её сетевой части, так и с некорректным администрированием, наиболее проблематичными являются следующие:

1) отправка атакуемой системе непрерывного потока произвольных UDP-пакетов (UDP flood) по порту 53 (DNS) приводит к краху DNS-сервера (Service Pack 3);
2) атака, направленная на DNS-сервер;
3) атака типа «человек-в-середине».

Кроме того, причиной успешных атак на ОС Windows может стать некорректная реализация выбранной политики безопасности (разрешения отдельным пользователям или группам пользователей при работе с рабочей станцией из ЛВС задаются некорректно). Чтобы избежать этих неприятностей следует:

1) удалить учётную запись пользователя Guest;
2) ограничить доступ пользователей к рабочей станции из ЛВС путём задания соответствующих прав, например запретить доступ к сети группе Everyone;
3) по возможности не использовать удалённое редактирование системного реестра.

Выполнение перечисленных рекомендаций позволит обезопасить работу в ЛВС под управлением ОС Windows.

Интеграция протоколов безопасности в прикладное ПО реализована за счёт создания нового интерфейса для приложений Win32 – SSPI (Security Support Provider Interface). Его применение позволит унифицировать обращение к функциональным возможностям данных протоколов и изолировать прикладное ПО от выполнения функций безопасности. Вынесение этих функций за рамки прикладного ПО позволит минимизировать влияние ошибок или закладок в программах на уровень обеспечиваемой безопасности.

В заключение следует отметить, что развитая сетевая поддержка в ОС Windows существенно затрудняет обеспечение безопасности ЛВС. Следовательно, при эксплуатации этой операционной системы целесообразно всемерно упрощать способы и протоколы взаимодействия компьютеров и прикладных систем между собой.

Основная литература

1. Смарт Н. Криптография. - Учебник для вузов: / пер. С. А. Кулешов, ред. пер. С. К. Ландо. - М. : Техносфера, 2005. - 528 с. – 11 экз.

2. Зубов А. Ю. Криптографические методы защиты информации. Совершенные шифры : Учебное пособие для вузов / А. Ю. Зубов. - М.: "Гелиос АРВ", 2005. – 190 с. – 30 экз.

Дополнительная литература

3. Молдовян А.А., Молдовян Н.А., Советов Б.Я. Криптография. - Серия «Учебники для вузов. Специальная литература». - СПб.: Издательство «Лань», 2000, 224 с. – 6 экз.
4. Петров А. А. Компьютерная безопасность. Криптографические методы защиты. – М.: ДМК, 2000, 448 с. - 3 экз.
5. Мельников В.П., Клейменов С.А., Петраков А.М.; / ред.: С. А. Клейменов Информационная безопасность и защита информации. Учебное пособие для вузов. - М. : Academia, 2006. - 330, 158 с. – 30 экз.
6. Шнайер Б. Прикладная криптография: Протоколы, алгоритмы, исходные тексты на языке Си : Пер. с англ. / Брюс Шнайер; Ред. пер. П. В. Семьянов. - М. : ТРИУМФ, 2003. 796 с. – 1 экз.
7. Завгородний В.И. Комплексная защита информации в компьютерных системах: Учебное пособие для вузов - М. : Логос, 2001. - 264 с. – 10 экз.
8. Романец Ю.В. Защита информации в компьютерных системах и сетях / Ю. В. Романец, П. А. Тимофеев, В. Ф. Шаньгин ; ред. : В. Ф. Шаньгин. - 2-е изд., перераб и доп. - М. : Радио и связь, 2001. - 376 с. – 42 экз.
9. http://www.ssl.stu.neva.ru/- Санкт-Петербургский центр защиты информации

ОГЛАВЛЕНИЕ