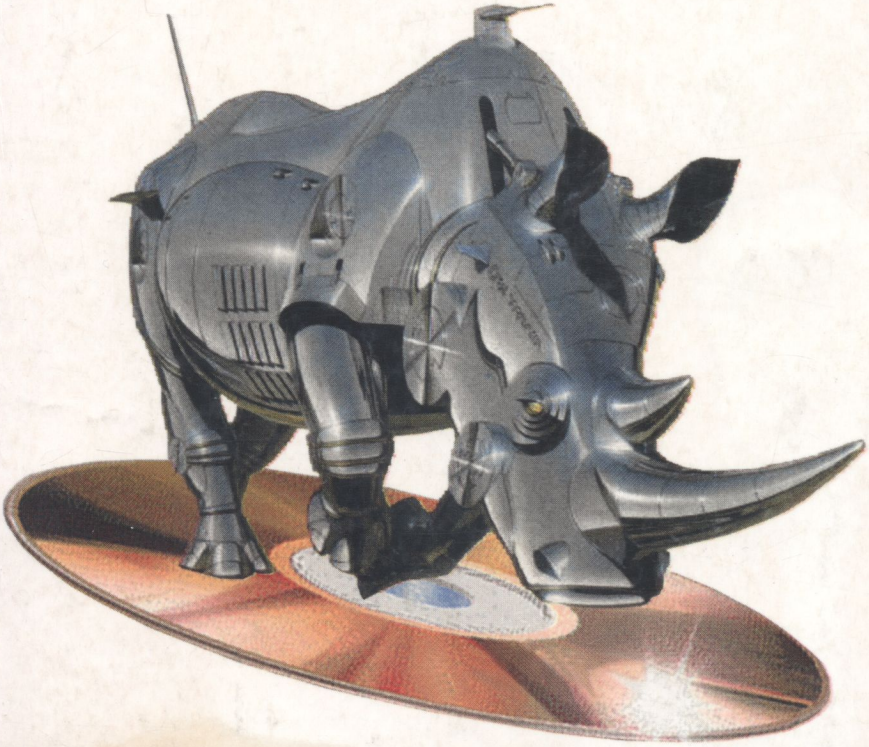
Б. Анин

bhv
www.bhv.ru
www.bhv.kiev.ua

ЗАЩИТА компьютерной информации

- Криптографические методы и протоколы
- Борьба с программами-шпионами и кибершпионажем
- Средства защиты операционных систем и компьютерных сетей



МАСТЕР

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ

Борис Анин

ЗАЩИТА КОМПЬЮТЕРНОЙ ИНФОРМАЦИИ

BHV
Санкт-Петербург

Дюссельдорф • Киев • Москва • Санкт-Петербург

УДК 681.3.06

В книге рассматриваются вопросы обеспечения безопасности конфиденциальной компьютерной информации. Кроме обобщенной характеристики хакерских атак, содержится описание технологий преодоления защиты современных компьютерных систем, включая внедрение программных закладок, нахождение брешей в защитных механизмах компьютерной сети, взлом парольной защиты распространенных операционных систем, а также приводятся эффективные способы противодействия этим действиям. Значительная часть книги посвящена новейшим криптографическим методам защиты компьютерных данных. Кроме того, имеется англо-русский криптологический словарь, где даются толкования многих специальных терминов.

Для широкого круга пользователей

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зав. редакцией	*Наталья Таркова*
Редактор	*Татьяна Кручинина*
Компьютерная верстка	*Натальи Смирновой*
Корректор	*Светлана Журавина*
Дизайн обложки	*Натальи Смирновой*
Зав. производством	*Николай Тверских*

Анин Б. Ю.
Защита компьютерной информации. — СПб.: БХВ-Петербург, 2000. — 384 с.: ил.

ISBN 5-8206-0104-1

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 12.10.00.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 30,96.
Доп. тираж 3000 экз. Заказ № 603
"БХВ-Петербург", 198005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар, № 77.99.1.953.П.950.3.99 от 01.03.1999 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в Академической типографии "Наука" РАН.
199034, Санкт-Петербург, 9-я линия, 12.

Содержание

Предисловие

Люди, уходя из дома, обычно закрывают входную дверь на замок. Они также запирают свои автомобили, оставляя их припаркованными на улице или на стоянке. И, как правило, не сообщают номер своей кредитной карты первому встречному коробейнику, который пристает к прохожим на улице, настырно предлагая купить у него товары сомнительного качества. Однако подавляющее большинство людей до конца не осознает, насколько сильно они рискуют, если не заботятся о защите информации, находящейся в их компьютерах.

Достоверно известно, что лишь отдельные пользователи предпринимают хоть какие-то меры, призванные сберечь их данные. Остальные всерьез задумываются об этом только тогда, когда теряют информацию, хранимую в компьютере. Более того, их компьютерные системы зачастую совершенно не защищены от краж и вандализма.

Каждый раз, используя свой компьютер, его владелец добавляет туда определенную порцию информации. Именно эта совокупная информация и является наиболее ценным компонентом всей компьютерной системы. А это значит, что если не предпринять специальных мер для ее защиты, издержки, которые понесет пользователь, попытавшись восстановить утраченные данные, значительно превысят стоимость аппаратных средств, используемых для хранения этих данных. Еще более чреватой опасными последствиями является ситуация, при которой налоговая и банковская информация пользователя или его деловая переписка попадает в чужие руки. Трудно себе вообразить, что кто-то, находясь в здравом уме и твердой памяти, по доброй воле предоставляет свою личную информацию людям, с которыми не имеет или не желает иметь никаких дел.

Но даже если вам нечего скрывать от посторонних (в это трудно поверить, но предположим, что это действительно так), непременно отыщется кто-нибудь, кто не прочь превратить ваш компьютер в груду бесполезного хлама, как только представится такая возможность. Расплодившиеся киберпанки, крэкеры, фрикеры и брейкеры с большим энтузиазмом занимаются электронным воровством, что можно сравнить с осквернением могил и разрисовыванием стен зданий неприличными надписями.

Не следует пренебрегать и защитой данных от стихийных бедствий. Ведь совершенно не важно, испортится ваш жесткий диск от компьютерного вируса, от рук злобного хакера, или ту же самую медвежью услугу вам окажет ураган, наводнение либо шаровая молния.

Не менее важно отметить, что вне зависимости от того, насколько ценна ваша информация, российским законодательством она безусловно признается объектом вашей собственности. И вы, как владелец своей информации, имеете право определять правила ее обработки и защиты. Базовым в этом отношении является Закон Российской Федерации "Об информации, информатизации и защите информации", принятый 25 января 1995 г. В соответствии с ним любой российский гражданин может предпринимать необходимые меры для предотвращения утечки, хищения, утраты, искажения и подделки информации. Вопрос состоит в том, какие действия являются на самом деле необходимыми для адекватной защиты вашей информации.

Рискну предположить, что вы вряд ли покидаете свой дом даже на короткое время без того, чтобы не запереть дверь, хотя это довольно хлопотное занятие. Во-первых, необходимо обладать минимумом технических знаний, чтобы подобрать и установить надежный замок. Во-вторых, требуется постоянный контроль за состоянием замка, чтобы содержать его в исправности. В-третьих, чтобы замок предотвращал проникновение в дом посторонних людей, вы должны соблюдать определенные правила (хранить ключи в надежном месте, а также не оставлять дверь незапертой). Подобные же правила применимы и в случае защиты информации в компьютерных системах. Именно поэтому так важно отыскать разумный компромисс между ценностью ваших данных и неудобствами, связанными с использованием необходимых мер безопасности.

Как и дверной замок, любая система компьютерной защиты информации не является полностью безопасной. Всегда найдется кто-нибудь, способный взломать защитные механизмы компьютера. К счастью, такие люди встречаются очень редко, иначе единственный способ защитить наши данные состоял бы в их уничтожении.

Таким образом, задача обеспечения информационной безопасности противоречива по самой своей сути. С одной стороны, средств обеспечения безопасности никогда не бывает слишком много в том смысле, что защиту всегда можно тем или иным способом преодолеть (просто каждый раз, когда повышается уровень защиты, приходится придумывать более изощренный способ ее обхода). С другой стороны, чем сильнее кого-то или что-то защищают, тем больше возникает неудобств и ограничений, и в результате вместо чувства спокойствия информационная защита вызывает лишь раздражение и стремление от нее отмахнуться, как от надоедливой мухи. Например, за счет жесткого контроля за доступом к компьютерной системе с помощью паролей несомненно снижается вероятность их подбора взломщиком, однако одновременно это заставляет рядовых пользователей прилагать значительно больше усилий для придумывания и запоминания паролей. А установка строгих ограничений на доступ к информации создает дополнительные трудности при совместной работе с этой информацией. Поэтому идеальной и универсальной системы защиты информации не существует:

здесь все слишком индивидуально, и вариант защиты, наиболее близкий к оптимальному, все время приходится подбирать заново.

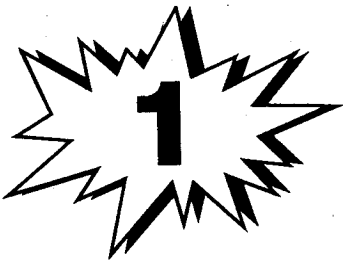
Внимательно изучив предлагаемую вашему вниманию книгу, вы сможете подобрать оптимальные методы для защиты компьютерной информации. При этом очень существенную роль будет играть степень изолированности вашего компьютера от внешнего мира.

Если компьютер находится у вас дома или в офисе и физически не связан с другими компьютерами (с помощью модема или аппаратной сети), то, по всей вероятности, самая большая опасность состоит в том, что кто-то занесет в него вирус или троянскую программу. Вторая, крайне неблагоприятная перспектива, заключается в том, что ваш компьютер может быть похищен. Остальных неприятностей можно избежать, применяя парольную защиту компьютера и шифруя файлы, содержащие конфиденциальную информацию. В этом случае снабдить свой компьютер довольно надежной защитой сможет любой человек, даже не очень сведущий в вопросах информационной безопасности.

Если компьютерная система подключена к сети, то потребуется принять дополнительные меры безопасности, квалифицированно установить и правильно использовать которые под силу только подготовленному специалисту, имеющему опыт работы в области защиты информации. Защитные механизмы, встроенные в сетевые операционные системы, всевозможные брандмауэры и анализаторы безопасности сетей требуют очень тонкой настройки. Поэтому даже незначительная ошибка при их установке и настройке чревата серьезными последствиями, поскольку злоумышленнику достаточно обнаружить всего лишь одно слабое место в системе защиты, чтобы осуществить ее взлом. Однако и в случае, если компьютер работает в составе сети, изучение предлагаемой книги будет полезно всем — от технаря до менеджера, не имеющего глубоких познаний в этих вопросах. Ведь правильное использование средств защиты информации возможно лишь в том случае, когда все люди, так или иначе причастные к работе со средствами обработки и хранения данных, осознают (пусть даже в самых общих чертах) принципы обеспечения надежного функционирования этих средств и скрупулезно следуют данным принципам на практике.

Компьютерная безопасность

Глава 1



Угрозы компьютерной безопасности

Компьютерная преступность в России

В странах, где высок уровень компьютеризации, проблема борьбы с компьютерной преступностью уже довольно давно стала одной из первостепенных. И это не удивительно. Например, в США ущерб от компьютерных преступлений составляет ежегодно около 5 млрд долларов, во Франции эти потери доходят до 1 млрд франков в год, а в Германии при помощи компьютеров преступники каждый год ухитряются похищать около 4 млрд марок. И число подобных преступлений увеличивается ежегодно на 30—40%.

Поскольку Россия никогда не входила (и в ближайшем будущем вряд ли войдет) в число государств с высоким уровнем компьютеризации (на большей части ее территории отсутствуют разветвленные компьютерные сети и далеко не везде методы компьютерной обработки информации пришли на смену традиционным), то довольно долго российское законодательство демонстрировало чрезмерно терпимое отношение к компьютерным преступлениям. Положительные сдвиги произошли только после ряда уголовных дел, самым громким из которых стало дело одного из программистов Волжского автомобильного завода, умышленно внесшего деструктивные изменения в программу, которая управляла технологическим процессом, что нанесло заводу значительный материальный ущерб. Отечественное законодательство претерпело существенные изменения, в результате которых был выработан ряд законов, устанавливающих нормы использования компьютеров в России.

Главной вехой в цепочке этих изменений стало введение в действие 1 января 1997 г. нового Уголовного кодекса. В нем содержится глава "Преступления в сфере компьютерной информации", где перечислены следующие преступления:

- неправомерный доступ к компьютерной информации (статья 272);

- создание, использование и распространение вредоносных компьютерных программ (статья 273);
- нарушение правил эксплуатации компьютеров, компьютерных систем и сетей (статья 274).

Отметим, что уголовная ответственность за перечисленное наступает только в том случае, когда уничтожена, блокирована, модифицирована или скопирована информация, хранящаяся в электронном виде. Таким образом, простое несанкционированное проникновение в чужую информационную систему без каких-либо неблагоприятных последствий наказанию не подлежит. Сравните: вторжение в квартиру, дом или офис против воли их владельца однозначно квалифицируется как уголовно наказуемое действие вне зависимости от последствий.

Следует сказать, что наличие законодательства, регламентирующего ответственность за компьютерные преступления, само по себе не является показателем степени серьезности отношения общества к таким преступлениям. К примеру, в Англии полное отсутствие специальных законов, карающих именно за компьютерные преступления, на протяжении многих лет отнюдь не мешает английской полиции эффективно расследовать дела такого рода. И действительно, все эти злоупотребления можно успешно квалифицировать по действующему законодательству, исходя из конечного результата преступной деятельности (хищение, вымогательство, мошенничество или хулиганство). Ответственность за них предусмотрена уголовным и гражданским кодексами. Ведь убийство и есть убийство, вне зависимости от того, что именно послужило орудием для него.

По данным Главного информационного центра МВД России в 1997 г. доля компьютерных преступлений составила 0,02% от общего числа преступлений в области кредитно-финансовой сферы. В абсолютных цифрах общее количество компьютерных преступлений в этом году превысило сотню, а суммарный размер ущерба —20 млрд рублей.

Однако к этой статистике следует относиться осторожно. Дело в том, что долгое время в правоохранительных органах не было полной ясности относительно параметров и критериев, по которым следовало фиксировать совершенные компьютерные преступления, а также попытки их совершить. Можно предположить, что данные, учтенные официальной статистикой, составляют лишь вершину айсберга, подводная часть которого представляет существенную угрозу обществу. И для этого имеются серьезные основания.

Российским правоохранительным органам становятся известны не более 5—10% совершенных компьютерных преступлений. Их раскрываемость тоже не превышает 1—5%. Это связано с тем, что хищение информации долгое время может оставаться незамеченным, поскольку зачастую данные просто копируются. Жертвы компьютерной преступности (большинство среди них —

частные предприятия) проявляют нежелание контактировать с правоохранительными органами, опасаясь распространения среди вкладчиков и акционеров сведений о собственной халатности и ненадежной работе своей фирмы, что может инициировать отток финансов и последующее банкротство.

Тенденции

По свидетельству экспертов самым привлекательным сектором российской экономики для преступников является кредитно-финансовая система. Анализ преступных деяний, совершенных в этой сфере с использованием компьютерных технологий, а также неоднократные опросы представителей банковских учреждений позволяют выделить следующие наиболее типичные способы совершения преступлений:

- Наиболее распространенными являются компьютерные преступления, совершаемые путем несанкционированного доступа к банковским базам данных посредством телекоммуникационных сетей. В 1998 г. российскими правоохранительными органами были выявлены 15 подобных преступлений, в ходе расследования которых установлены факты незаконного перевода 6,3 млрд рублей.
- За последнее время не отмечено ни одно компьютерное преступление, которое было совершено одним человеком. Более того, известны случаи, когда преступными группировками нанимались бригады из десятков хакеров, которым предоставлялось отдельное охраняемое помещение, оборудованное по последнему слову техники, для того чтобы они осуществляли хищение крупных денежных средств путем нелегального проникновения в компьютерные сети крупных коммерческих банков.
- Большинство компьютерных преступлений в банковской сфере совершается при непосредственном участии самих служащих коммерческих банков. Результаты исследований, проведенных с привлечением банковского персонала, показывают, что доля таких преступлений приближается к 70% от общего количества преступлений в банковской сфере. Например, в 1998 г. работники правоохранительных органов предотвратили хищение на сумму в 2 млрд рублей из филиала одного крупного коммерческого банка. Преступники оформили проводку фиктивного платежа с помощью удаленного доступа к банковскому компьютеру через модем, введя пароль и идентификационные данные, которые им передали сообщники, работающие в этом филиале банка. Затем похищенные деньги были переведены в соседний банк, где преступники попытались снять их со счета с помощью поддельного платежного поручения.
- Много компьютерных преступлений совершается в России с использованием возможностей, которые предоставляет своим пользователям сеть Internet.

Internet как среда и как орудие совершения компьютерных преступлений

Уникальность сети Internet заключается в том, что она не находится во владении какого-то физического лица, частной компании, государственного ведомства или отдельной страны. Поэтому практически во всех ее сегментах отсутствует централизованное регулирование, цензура и другие методы контроля информации. Благодаря этому открываются практически неограниченные возможности доступа к любой информации, которые используются преступниками. Сеть Internet можно рассматривать не только как инструмент совершения компьютерных преступлений, но и как среда для ведения разнообразной преступной деятельности.

При использовании сети Internet в качестве среды для преступной деятельности привлекательной для правонарушителей является сама возможность обмена информацией криминального характера. Применять в своей деятельности коммуникационные системы, обеспечивающие такую же оперативную и надежную связь по всему миру, раньше были в состоянии только спецслужбы сверхдержав — Америки и России, которые обладали необходимыми космическими технологиями.

Другая особенность сети Internet, которая привлекает преступников, — возможность осуществлять в глобальных масштабах информационно-психологическое воздействие на людей. Преступное сообщество весьма заинтересовано в распространении своих доктрин и учений, в формировании общественного мнения, благоприятного для укрепления позиций представителей преступного мира, и в дискредитации правоохранительных органов.

Однако наибольший интерес сеть Internet представляет именно как орудие для совершения преступлений (обычно в сфере экономики и финансов). В самом простом варианте эти преступления связаны с нарушением авторских прав. К такого рода преступлениям, в первую очередь, относится незаконное копирование и продажа программ, находящихся на серверах компаний, которые являются владельцами этих программ.

Во вторую группу преступлений можно включить нелегальное получение товаров и услуг, в частности, бесплатное пользование услугами, предоставляемыми за плату различными телефонными компаниями, за счет отличных знаний принципов функционирования и устройства современных автоматических телефонных станций. Другие способы незаконного пользования услугами основываются на модификации сведений о предоставлении этих услуг в базах данных компаний, которые их оказывают. Информация о предоставлении какой-то услуги в кредит может либо просто уничтожаться, либо изменяться для того, чтобы потребителем уже оплаченной кем-то услуги стал член преступного сообщества.

Наряду с нелегальным получением товаров и услуг интерес для преступных группировок представляют такие сферы мошеннической деятельности, как

азартные игры (казино, лотереи и тотализаторы), организация финансовых пирамид, фиктивных брачных контор и фирм по оказанию мифических услуг. Во всех случаях оперативность взаимодействия с жертвами мошенничества и анонимность самого мошенника весьма привлекательны при совершении компьютерных преступлений в сети Internet.

Одна из предпосылок повышенного интереса, который преступники проявляют к сети Internet, заключается в том, что с развитием компьютерных сетей информация становится все более ценным товаром. Особенно это касается информации, имеющей отношение к банковской сфере — данные о вкладах и вкладчиках, финансовом положении банка и клиентов; кредитной и инвестиционной политике банка, а также о направлениях его развития. Поскольку в современных условиях субъекты кредитно-финансовой деятельности не могут существовать без взаимного информационного обмена, а также без общения со своими территориально удаленными филиалами и подразделениями, то часто для этих целей они используют Internet. А это значит, что у преступников появляется реальный шанс получить доступ к сугубо секретной информации о потенциальных объектах своей преступной деятельности. Уничтожение такой информации преступниками является разновидностью недобросовестной конкуренции со стороны предприятий, которые находятся под "крышей" этих преступников. Даже одна угроза ее уничтожения может сама по себе послужить эффективным средством воздействия на руководство банка с целью вымогательства или шантажа.

Через Internet преступники стремятся также получить возможность нужным для себя образом модифицировать конфиденциальную служебную информацию, которая используется руководством банка для принятия каких-либо важных решений. Дело в том, что ввиду высокой трудоемкости оценки степени доверия к потенциальному получателю банковского кредита в большинстве банков промышленно-развитых стран эта операция автоматизирована. В секрете держатся не только исходные данные для принятия подобных решений, но и сами алгоритмы их выработки. Нетрудно догадаться, какими могут быть последствия, если такие алгоритмы знают посторонние лица, которые могут оказаться в состоянии модифицировать их так, чтобы они вырабатывали благоприятные для этих лиц решения.

Дополнительная сфера компьютерных преступлений, совершаемых через Internet, появилась с возникновением электронных банковских расчетов, т. е. с введением в обращение так называемой электронной наличности. Есть разные способы ее хищения, но все они основываются на модификации информации, отображающей электронную наличность. Информация о наличности, имеющейся на счетах клиентов, переписывается на счета, которыми безраздельно распоряжаются преступники. Изменения также могут быть внесены в сам алгоритм, определяющий правила функционирования системы обработки информации об электронных банковских расчетах. На-

пример, меняется курс валют, чтобы для клиентов банка валюта пересчитывалась по заниженному курсу, а разница зачислялась на счета преступников.

Синдром Робина Гуда

В 1998 г. в Экспертно-криминалистическом центре МВД была проведена классификация компьютерных преступников. Обобщенный портрет отечественного хакера, созданный на основе этого анализа, выглядит примерно так:

- мужчина в возрасте от 15 до 45 лет, имеющий многолетний опыт работы на компьютере либо почти не обладающий таким опытом;
- в прошлом к уголовной ответственности не привлекался;
- яркая мыслящая личность;
- способен принимать ответственные решения;
- хороший, добросовестный работник, по характеру нетерпимый к насмешкам и к потере своего социального статуса среди окружающих его людей;
- любит уединенную работу;
- приходит на службу первым и уходит последним; часто задерживается на работе после окончания рабочего дня и очень редко использует отпуск и отгулы.

По сведениям того же Экспертно-криминалистического центра МВД, принципиальная схема организации взлома защитных механизмов банковской информационной системы заключается в следующем. Профессиональные компьютерные взломщики обычно работают только после тщательной предварительной подготовки. Они снимают квартиру на подставное лицо в доме, в котором не проживают сотрудники спецслужб или городской телефонной сети. Подкупают сотрудников банка, знакомых с деталями электронных платежей и паролями, и работников телефонной станции, чтобы обезопаситься на случай поступления запроса от службы безопасности банка. Нанимают охрану из бывших сотрудников МВД. Чаще всего взлом банковской компьютерной сети осуществляется рано утром, когда дежурный службы безопасности теряет бдительность, а вызов помощи затруднен.

Компьютерные преступления парадоксальны тем, что в настоящее время трудно найти другой вид преступления, после совершения которого его жертва не выказывает заинтересованности в поимке преступника, а сам преступник, будучи пойман, рекламирует свою деятельность на поприще компьютерного взлома, мало что утаивая от представителей правоохранительных органов. Этот парадокс вполне объясним:

- жертва компьютерного преступления совершенно убеждена, что затраты на его раскрытие (включая потери, понесенные в результате утраты банком своей репутации) существенно превосходят уже причиненный ущерб;

- Преступник, даже получив максимальный срок тюремного наказания (не очень большой, а если повезет, то условный или сокращенный), приобретет широкую известность в деловых и криминальных кругах, что в дальнейшем позволит ему с выгодой использовать полученные знания.

Таким образом, для общественного мнения в России характерен "синдром Робина Гуда" — преступники-хакеры представляются некими благородными борцами против толстосумов-банкиров. А посему хакерство в России, по видимому, просто обречено на дальнейшее развитие.

Суммируя информацию о компьютерных взломах, которая время от времени появляется в отечественной прессе, можно привести следующее обобщенное (так сказать, "среднеарифметическое") описание одного такого взлома. И если в этом описании кто-то увидит определенные преувеличения или несуразности, то их следует отнести на счет воспаленного воображения некоторых российских журналистов, явно страдающих "синдромом Робина Гуда" в особо тяжелой форме.

История одного компьютерного взлома

В назначенный день около восьми часов вечера все участники планировавшегося взлома компьютерной защиты одного из банков собрались в довольно просторной московской квартире, которая за 3 месяца до начала операции была снята на подставное имя в неприметном доме на тихой улице. Дом был выбран отнюдь не случайно: они выяснили, что в этом доме не проживают сотрудники правоохранительных органов, спецслужб, налоговой полиции и городских телефонных станций, дипломаты, депутаты, террористы, киберпанки и резиденты иностранных разведок.

Одновременно со взломщиками на "боевое" дежурство заступили подкупленный служащий местной телефонной станции, который на запрос об определении номера телефона квартиры, откуда осуществлялся взлом, должен был сообщить "липовый" номер телефона, и высокопоставленный агент в российских ракетно-космических войсках, который в случае неудачного взлома пообещал организовать нанесение ракетного удара для уничтожения банка, телефонной станции и штаб-квартиры взломщиков с целью замести следы преступления.

У подъезда дома стоял ничем не примечательный шестисотый "Мерседес", в котором сидели два представителя заказчика — крупной московской бандитской группировки. Один банк "кинул" дружественную группировке коммерческую фирму на очень большую сумму. Этот банк было решено наказать, запустив в его компьютерную сеть зловредный вирус, который должен был минимум на сутки вывести из строя все банковские коммуникации, включая электрическую, газовую, водопроводную и канализационную сеть, а также мусоропровод.

Главарь банды взломщиков провел инструктаж о бережном обращении с казенным имуществом, раздал компьютеры, бронежилеты, рации и сообщил позывные.

Временная штаб-квартира взломщиков была оснащена по последнему слову техники. На телефонную линию установлено устройство для противодействия подслушиванию. Окна и двери залиты эпоксидной смолой, чтобы не подглядывали соседи. Вся аппаратура подключена к мощным аккумуляторам, чтобы случайные перебои в электропитании не смогли помешать взлому. А сами компьютеры густо посыпаны средством против тараканов и мышей, чтобы исключить непредвиденные сбои в работе оборудования.

Подготовительный этап операции, основная задача которого состояла в подборе входных паролей, начался в 9 часов вечера и продолжался всю ночь, пока в 6 часов утра в штаб-квартире не раздался телефонный звонок. Звонил подкупленный работник банка, который сообщил пароли для входа в банковскую компьютерную сеть.

Главарь приказал по рации всем участникам операции приготовиться к проведению ее заключительной фазы. "Первый готов, второй готов, третий готов" — понеслось в ответ. "Поехали!" — послышалась команда главаря. По этой команде дюжие охранники окружили дом: им было приказано никого не впускать и не выпускать без письменного разрешения с подписью главаря, заверенной нотариусом.

Семь смертоносных программ-вирусоносителей устремились по телефонным линиям в атаку на главный сервер банка. Банковские программы защиты были сначала парализованы, а потом полностью смяты превосходящими силами противника. Вырвавшиеся на оперативный простор вирусы учинили в компьютерной сети банка настоящий дебош.

Получив сигнал о проникновении вирусов, главный сервер компьютерной сети банка отключил все коммуникации. В результате город остался без тепла, воды и электричества.

Пока поднятые по тревоге сотрудники отдела безопасности банка торговались, сколько им заплатят за срочность, а потом вылавливали расплодившиеся вирусы, прошли почти сутки. За это время в московской квартире, из которой осуществлялся взлом, были уничтожены все следы пребывания компьютерных взломщиков.

Прямой ущерб, понесенный банком из-за непрохождения платежей, составил сотни тысяч долларов. А заказчикам взлом компьютерной защиты банка обошелся всего в 20 тыс. фальшивых долларов.

В заключение можно заметить, что зарубежным банкирам еще повезло, что операция прошла успешно, и для заметания следов совершенного преступления компьютерным взломщикам не пришлось воспользоваться услугами своего агента в ракетно-космических войсках.

Компьютер глазами хакера

В деловом мире наконец-то признали важность решения проблемы защиты компьютерных данных. Громкие процессы, связанные с проникновением злоумышленников в корпоративные компьютерные системы, особенно дело Левина, названное Интерполом самым серьезным транснациональным сетевым компьютерным преступлением, в результате которого американский Сити-банк потерял 400 тыс. долларов, привлекли пристальное внимание не только специалистов в области компьютерной обработки данных, но и директоров компаний. Последние пусть с опозданием, но все-таки поняли, что с пуском в эксплуатацию каждой новой компьютерной системы, имеющей выход в глобальную компьютерную сеть Internet, они рискуют распахнуть перед злоумышленниками всех мастей (профессиональными взломщиками и грабителями, обиженными подчиненными или ничем не брезгующими конкурентами) окно, через которое те могут беспрепятственно проникать в святая святых компании и наносить существенный материальный ущерб. В результате как неосведомленность руководителей, так и бюджетные ограничения теперь не являются основными препятствиями на пути внедрения мер защиты информации в компьютерных системах, а главную роль играет выбор конкретных инструментов и решений.

Выяснилось, что создание хорошо защищенной компьютерной системы невозможно без тщательного анализа потенциальных угроз для ее безопасности. Специалисты составили перечень действий, которые нужно произвести в каждом конкретном случае, чтобы представлять сценарии возможных нападений на компьютерную систему. Этот перечень включал:

- определение ценности информации, хранимой в компьютерной системе;
- оценку временных и финансовых затрат, которые может позволить себе злоумышленник для преодоления механизмов защиты компьютерной системы;
- вероятную модель поведения злоумышленника при атаке на компьютерную систему;
- оценку временных и финансовых затрат, необходимых для организации адекватной защиты компьютерной системы.

Таким образом, при проведении анализа потенциальных угроз безопасности компьютерной системы эксперт ставил себя на место злоумышленника, пытающегося проникнуть в эту систему. А для этого ему необходимо было понять, что представляет собой злоумышленник, от которого требуется защищаться. И в первую очередь нужно было как можно точнее ответить на следующие вопросы:

- насколько высок уровень профессиональной подготовки злоумышленника;
- какой информацией об атакуемой компьютерной системе он владеет;

- как злоумышленник осуществляет доступ к этой системе;
- каким способом атаки он воспользуется с наибольшей вероятностью.

Однако экспертам, которым приходилось ставить себя на место злоумышленника, чтобы ответить на перечисленные выше вопросы, очень не понравилось именоваться злоумышленниками. Во-первых, это слово — длинное и неуклюжее, а во-вторых, оно не совсем адекватно отражает суть решаемой задачи, которая состоит не в нахождении брешей в защите компьютерной системы, а в их ликвидации. Поэтому они взяли на вооружение другой термин, который более точно соответствует возложенной на них миссии, и стали разрабатывать сценарии поведения так называемых *хакеров*.

Кто такие хакеры

Хотя последнее время термин *хакер* можно довольно часто встретить на страницах компьютерной прессы, у пользующихся им людей до сих пор не сложилось единого мнения о том, кого именно следует именовать хакером. Часто хакером называют любого высококлассного специалиста в области вычислительной техники и всего, что с ней связано. Однако среди журналистов, затрагивающих тему хакерства, имеются серьезные разногласия относительно того, как хакеры применяют свои уникальные познания на практике.

Одни предлагают называть хакером лишь того, кто пытается взломать защиту компьютерных систем, чтобы затем выдать обоснованные рекомендации по улучшению их защитных механизмов, другие — именовать хакером высококвалифицированного специалиста, который взламывает компьютеры исключительно в преступных целях. Чтобы не ввязываться в полемику, которая в основном касается нравственно-этической стороны деятельности хакеров, для начала назовем хакером любого человека, стремящегося обойти защиту компьютерной системы, вне зависимости от того, преследуются ли по закону его действия. При этом основной целью хакера является получение дополнительных привилегий и прав доступа к компьютерной системе. По отношению к атакуемой им компьютерной системе хакер может быть:

- посторонним лицом, не имеющим никаких легальных привилегий и прав доступа;
- пользователем компьютерной системы, обладающим ограниченными привилегиями и правами доступа.

Статистика компьютерных преступлений свидетельствует о том, что уровень профессиональной подготовки хакера варьируется в очень широких пределах. Хакером может стать даже школьник, случайно обнаруживший программу взлома на одном из специализированных хакерских серверов в сети Internet. В то же время отмечено и появление настоящих хакерских банд, главарями которых являются компьютерные специалисты высочайшей квалификации.

В дальнейшем под хакером будет пониматься только высококвалифицированный специалист, поскольку именно его действия представляют наибольшую угрозу безопасности компьютерных систем. Для такого хакера характерны следующие черты и особенности поведения:

- он всегда в курсе последних новинок в области компьютерной техники, устройств связи и программных средств;
- перед тем как атаковать компьютерную систему, он всеми доступными способами пытается собрать максимум информации об этой системе, включая данные об используемом в ней программном обеспечении и ее администраторах;
- добывая нужную ему информацию, он не брезгует агентурными и оперативно-техническими методами (например, устанавливая подслушивающие устройства в местах, часто посещаемых обслуживающим персоналом компьютерных систем, которые он намеревается взломать);
- перед попыткой взлома компьютерной системы он опробует методы, которые планирует применить для атаки на эту систему, на заранее подготовленной модели, имеющей те же средства обеспечения безопасности, что и атакуемая система;
- сама атака компьютерной системы осуществляется по возможности быстро, чтобы ее администраторы не смогли зафиксировать факт совершения атаки и не успели предпринять меры для отражения атаки и для выявления личности и местонахождения атакующего;
- хакер не пользуется слишком изощренными методами взлома защиты компьютерной системы, поскольку чем сложнее алгоритм атаки, тем вероятнее возникновение ошибок и сбоев при его реализации;
- чтобы минимизировать время, необходимое для взлома, и количество возможных ошибок, хакер обычно атакует компьютерную систему при помощи заранее написанных программ, а не вручную, набирая необходимые команды на клавиатуре компьютера;
- хакер никогда не действует под собственным именем и тщательно скрывает свой сетевой адрес, а на всякий случай у него имеется тщательно продуманный план замести следы или оставить ложный след (например, одновременно ведя неумелую и заведомо обреченную на провал атаку, благодаря чему журнал аудита атакуемой компьютерной системы оказывается забит до отказа сообщениями о событиях, затрудняющих для системного администратора выяснение характера действительной атаки и принятие мер, чтобы не допустить ее в будущем);
- хакеры широко применяют программные закладки, которые самоуничтожаются либо при их обнаружении, либо по истечении фиксированного периода времени.

Методы взлома компьютерных систем

В общем случае программное обеспечение любой универсальной компьютерной системы состоит из трех основных компонентов: операционной системы, сетевого программного обеспечения (СПО) и системы управления базами данных (СУБД). Поэтому все попытки взлома защиты компьютерных систем можно разделить на три группы:

- атаки на уровне операционной системы;
- атаки на уровне сетевого программного обеспечения;
- атаки на уровне систем управления базами данных.

Атаки на уровне систем управления базами данных

Защита СУБД является одной из самых простых задач. Это связано с тем, что СУБД имеют строго определенную внутреннюю структуру, и операции над элементами СУБД заданы довольно четко. Есть четыре основных действия — поиск, вставка, удаление и замена элемента. Другие операции являются вспомогательными и применяются достаточно редко. Наличие строгой структуры и четко определенных операций упрощает решение задачи защиты СУБД. В большинстве случаев хакеры предпочитают взламывать защиту компьютерной системы на уровне операционной системы и получать доступ к файлам СУБД с помощью средств операционной системы. Однако в случае, если используется СУБД, не имеющая достаточно надежных защитных механизмов, или плохо протестированная версия СУБД, содержащая ошибки, или если при определении политики безопасности администратором СУБД были допущены ошибки, то становится вполне вероятным преодоление хакером защиты, реализуемой на уровне СУБД.

Кроме того, имеются два специфических сценария атаки на СУБД, для защиты от которых требуется применять специальные методы. В первом случае результаты арифметических операций над числовыми полями СУБД округляются в меньшую сторону, а разница суммируется в некоторой другой записи СУБД (как правило, эта запись содержит личный счет хакера в банке, а округляемые числовые поля относятся к счетам других клиентов банка). Во втором случае хакер получает доступ к полям записей СУБД, для которых доступной является только статистическая информация. Идея хакерской атаки на СУБД — так хитро сформулировать запрос, чтобы множество записей, для которого собирается статистика, состояло только из одной записи.

Атаки на уровне операционной системы

Защищать операционную систему, в отличие от СУБД, гораздо сложнее. Дело в том, что внутренняя структура современных операционных систем чрезвычайно сложна, и поэтому соблюдение адекватной политики безопас-

ности является значительно более трудной задачей. Среди людей несведущих бытует мнение, что самые эффективные атаки на операционные системы могут быть организованы только с помощью сложнейших средств, основанных на самых последних достижениях науки и техники, а хакер должен быть программистом высочайшей квалификации. Это не совсем так.

Никто не спорит с тем, что пользователю следует быть в курсе всех новинок в области компьютерной техники. Да и высокая квалификация — совсем не лишнее. Однако искусство хакера состоит отнюдь не в том, чтобы взламывать любую самую "крутую" компьютерную защиту. Нужно просто суметь найти слабое место в конкретной системе защиты. При этом простейшие методы взлома оказываются ничуть не хуже самых изощренных, поскольку чем проще алгоритм атаки, тем больше вероятность ее завершения без ошибок и сбоев, особенно если возможности предварительного тестирования этого алгоритма в условиях, приближенных к "боевым", весьма ограничены.

Успех реализации того или иного алгоритма хакерской атаки на практике в значительной степени зависит от архитектуры и конфигурации конкретной операционной системы, являющейся объектом этой атаки. Однако имеются атаки, которым может быть подвергнута практически любая операционная система:

- кража пароля;
  - подглядывание за пользователем, когда тот вводит пароль, дающий право на работу с операционной системой (даже если во время ввода пароль не высвечивается на экране дисплея, хакер может легко узнать пароль, просто следя за перемещением пальцев пользователя по клавиатуре);
  - получение пароля из файла, в котором этот пароль был сохранен пользователем, не желающим затруднять себя вводом пароля при подключении к сети (как правило, такой пароль хранится в файле в незашифрованном виде);
  - поиск пароля, который пользователи, чтобы не забыть, записывают на календарях, в записных книжках или на оборотной стороне компьютерных клавиатур (особенно часто подобная ситуация встречается, если администраторы заставляют пользователей применять трудно запоминаемые пароли);
  - кража внешнего носителя парольной информации (дискеты или электронного ключа, на которых хранится пароль пользователя, предназначенный для входа в операционную систему);
  - полный перебор всех возможных вариантов пароля;
  - подбор пароля по частоте встречаемости символов и биграмм, с помощью словарей наиболее часто применяемых паролей, с привлечением знаний о конкретном пользователе — его имени, фамилии, номера те-

лефона, даты рождения и т. д., с использованием сведений о существовании эквивалентных паролей, при этом из каждого класса опробуется всего один пароль, что может значительно сократить время перебора;

- сканирование жестких дисков компьютера (хакер последовательно пытается обратиться к каждому файлу, хранимому на жестких дисках компьютерной системы; если объем дискового пространства достаточно велик, можно быть вполне уверенным, что при описании доступа к файлам и каталогам администратор допустил хотя бы одну ошибку, в результате чего все такие каталоги и файлы будут прочитаны хакером; для сокрытия следов хакер может организовать эту атаку под чужим именем: например, под именем пользователя, пароль которого известен хакеру);
- сборка "мусора" (если средства операционной системы позволяют восстанавливать ранее удаленные объекты, хакер может воспользоваться этой возможностью, чтобы получить доступ к объектам, удаленным другими пользователями: например, просмотрев содержимое их "мусорных" корзин);
- превышение полномочий (используя ошибки в программном обеспечении или в администрировании операционной системы, хакер получает полномочия, превышающие полномочия, предоставленные ему согласно действующей политике безопасности);
  - запуск программы от имени пользователя, имеющего необходимые полномочия, или в качестве системной программы (драйвера, сервиса, демона и т. д.);
  - подмена динамически загружаемой библиотеки, используемой системными программами, или изменение переменных среды, описывающих путь к таким библиотекам;
  - модификация кода или данных подсистемы защиты самой операционной системы;
- отказ в обслуживании (целью этой атаки является частичный или полный вывод из строя операционной системы);
  - захват ресурсов (хакерская программа производит захват всех имеющихся в операционной системе ресурсов, а затем входит в бесконечный цикл);
  - бомбардировка запросами (хакерская программа постоянно направляет операционной системе запросы, реакция на которые требует привлечения значительных ресурсов компьютера);
  - использование ошибок в программном обеспечении или администрировании.

Если в программном обеспечении компьютерной системы нет ошибок и ее администратор строго соблюдает политику безопасности, рекомендованную

разработчиками операционной системы, то атаки всех перечисленных типов малоэффективны. Дополнительные меры, которые должны быть предприняты для повышения уровня безопасности, в значительной степени зависят от конкретной операционной системы, под управлением которой работает данная компьютерная система. Тем не менее, приходится признать, что вне зависимости от предпринятых мер полностью устранить угрозу взлома компьютерной системы на уровне операционной системы невозможно. Поэтому политика обеспечения безопасности должна проводиться так, чтобы, даже преодолев защиту, создаваемую средствами операционной системы, хакер не смог нанести серьезного ущерба.

Атаки на уровне сетевого программного обеспечения

СПО является наиболее уязвимым, потому что канал связи, по которому передаются сообщения, чаще всего не защищен, и всякий, кто может иметь доступ к этому каналу, соответственно, может перехватывать сообщения и отправлять свои собственные. Поэтому на уровне СПО возможны следующие хакерские атаки:

- прослушивание сегмента локальной сети (в пределах одного и того же сегмента локальной сети любой подключенный к нему компьютер в состоянии принимать сообщения, адресованные другим компьютерам сегмента, а следовательно, если компьютер хакера подсоединен к некоторому сегменту локальной сети, то ему становится доступен весь информационный обмен между компьютерами этого сегмента);
- перехват сообщений на маршрутизаторе (если хакер имеет привилегированный доступ к сетевому маршрутизатору, то он получает возможность перехватывать все сообщения, проходящие через этот маршрутизатор, и хотя тотальный перехват невозможен из-за слишком большого объема, чрезвычайно привлекательным для хакера является выборочный перехват сообщений, содержащих пароли пользователей и их электронную почту);
- создание ложного маршрутизатора (путем отправки в сеть сообщений специального вида хакер добивается, чтобы его компьютер стал маршрутизатором сети, после чего получает доступ ко всем проходящим через него сообщениям);
- навязывание сообщений (отправляя в сеть сообщения с ложным обратным сетевым адресом, хакер переключает на свой компьютер уже установленные сетевые соединения и в результате получает права пользователей, чьи соединения обманным путем были переключены на компьютер хакера);
- отказ в обслуживании (хакер отправляет в сеть сообщения специального вида, после чего одна или несколько компьютерных систем, подключенных к сети, полностью или частично выходят из строя).

Поскольку хакерские атаки на уровне СПО спровоцированы открытостью сетевых соединений, разумно предположить, что для отражения этих атак необходимо максимально защитить каналы связи и тем самым затруднить обмен информацией по сети для тех, кто не является легальным пользователем. Ниже перечислены некоторые способы такой защиты:

- максимальное ограничение размеров компьютерной сети (чем больше сеть, тем труднее ее защитить);
- изоляция сети от внешнего мира (по возможности следует ограничивать физический доступ к компьютерной сети извне, чтобы уменьшить вероятность несанкционированного подключения хакера);
- шифрование сетевых сообщений (тем самым можно устранить угрозу перехвата сообщений, правда, за счет снижения производительности СПО и роста накладных расходов);
- электронная цифровая подпись сетевых сообщений (если все сообщения, передаваемые по компьютерной сети, снабжаются электронной цифровой подписью, и при этом неподписанные сообщения игнорируются, то можно забыть про угрозу навязывания сообщений и про большинство угроз, связанных с отказом в обслуживании);
- использование брандмауэров[1] (брандмауэр является вспомогательным средством защиты, применяемым только в том случае, если компьютерную сеть нельзя изолировать от других сетей, поскольку брандмауэр довольно часто не способен отличить потенциально опасное сетевое сообщение от совершенно безвредного, и в результате типичной является ситуация, когда брандмауэр не только не защищает сеть от хакерских атак, но и даже препятствует ее нормальному функционированию).

Защита системы от взлома

Перечисленные выше методы хакерской атаки на компьютерную систему являются наиболее типичными и описаны в общей форме. Самые распространенные из этих методов будут рассмотрены ниже более подробно, поскольку их применение в конкретных случаях имеет свои особенности, которые требуют применения дополнительных защитных мер. А пока для обобщенной модели взлома компьютерных систем можно сформулировать универсальные правила, которых следует придерживаться, чтобы свести риск к минимуму.

- Не отставайте от хакеров: будьте всегда в курсе последних разработок из области компьютерной безопасности. Оформите подписку на несколько

[1] Брандмауэры фильтруют сообщения, передаваемые через маршрутизатор сети, с целью отсеивания потенциально опасных сообщений, которые возможно были отправлены хакерами в ходе атаки на эту сеть.

специализированных журналов, в которых подробно освещаются вопросы защиты компьютерных систем от взлома. Регулярно просматривайте материалы, помещаемые на хакерских серверах Internet (например, astalavista.box.sk).

- Руководствуйтесь принципом разумной достаточности: не стремитесь построить абсолютно надежную защиту. Ведь чем мощнее защита, тем больше ресурсов компьютерной системы она потребляет и тем труднее использовать ее.
- Храните в секрете информацию о принципах действия защитных механизмов компьютерной системы. Чем меньше хакеру известно об этих принципах, тем труднее будет для него организовать успешную атаку.
- Постарайтесь максимально ограничить размеры защищаемой компьютерной сети и без крайней необходимости не допускайте ее подключения к Internet.
- Перед тем как вложить денежные средства в покупку нового программного обеспечения, поищите информацию о нем, имеющуюся на хакерских серверах Internet.
- Размещайте серверы в охраняемых помещениях. Не подключайте к ним клавиатуру и дисплеи, чтобы доступ к этим серверам осуществлялся только через сеть.
- Абсолютно все сообщения, передаваемые по незащищенным каналам связи, должны шифроваться и снабжаться цифровой подписью.
- Если защищаемая компьютерная сеть имеет соединение с незащищенной сетью, то все сообщения, отправляемые в эту сеть или принимаемые из нее, должны проходить через брандмауэр, а также шифроваться и снабжаться цифровой подписью.
- Не пренебрегайте возможностями, которые предоставляет аудит. Интервал между сеансами просмотра журнала аудита не должен превышать одних суток.
- Если окажется, что количество событий, помещенных в журнал аудита, необычайно велико, изучите внимательно все новые записи, поскольку не исключено, что компьютерная система подверглась атаке хакера, который пытается замести следы своего нападения, зафиксированные в журнале аудита.
- Регулярно производите проверку целостности программного обеспечения компьютерной системы. Проверяйте ее на наличие программных закладок.
- Регистрируйте все изменения политики безопасности в обычном бумажном журнале. Регулярно сверяйте политику безопасности с зарегистрированной в этом журнале. Это поможет обнаружить присутствие программной закладки, если она была внедрена хакером в компьютерную систему.

- Пользуйтесь защищенными операционными системами.
- Создайте несколько ловушек для хакеров (например, заведите на диске файл с заманчивым именем, прочитать который невозможно с помощью обычных средств, и если будет зафиксировано успешное обращение к этому файлу, значит в защищаемую компьютерную систему была внедрена программная закладка).
- Регулярно тестируйте компьютерную систему с помощью специальных программ, предназначенных для определения степени ее защищенности от хакерских атак.

Глава 2

Программы-шпионы

Программные закладки

Современная концепция создания компьютерных систем предполагает использование программных средств различного назначения в едином комплексе. К примеру, типовая система автоматизированного документооборота состоит из операционной среды, программных средств управления базами данных, телекоммуникационных программ, текстовых редакторов, антивирусных мониторов, средств для криптографической защиты данных, а также средств аутентификации и идентификации пользователей. Главным условием правильного функционирования такой компьютерной системы является обеспечение защиты от вмешательства в процесс обработки информации тех программ, присутствие которых в компьютерной системе не обязательно. Среди подобных программ, в первую очередь, следует упомянуть компьютерные вирусы. Однако имеются вредоносные программы еще одного класса. От них, как и от вирусов, следует с особой тщательностью очищать свои компьютерные системы. Это так называемые *программные закладки*, которые могут выполнять хотя бы одно из перечисленных ниже действий:

- вносить произвольные искажения в коды программ, находящихся в оперативной памяти компьютера (программная закладка первого типа);
- переносить фрагменты информации из одних областей оперативной или внешней памяти компьютера в другие (программная закладка второго типа);
- искажать выводимую на внешние компьютерные устройства или в канал связи информацию, полученную в результате работы других программ (программная закладка третьего типа).

Программные закладки можно классифицировать и по методу их внедрения в компьютерную систему:

- программно-аппаратные закладки, ассоциированные с аппаратными средствами компьютера (их средой обитания, как правило, является BIOS — набор программ, записанных в виде машинного кода в постоянном запоминающем устройстве — ПЗУ);

- загрузочные закладки, ассоциированные с программами начальной загрузки, которые располагаются в загрузочных секторах (из этих секторов в процессе выполнения начальной загрузки компьютер считывает программу, берущую на себя управление для последующей загрузки самой операционной системы);
- драйверные закладки, ассоциированные с драйверами (файлами, в которых содержится информация, необходимая операционной системе для управления подключенными к компьютеру периферийными устройствами);
- прикладные закладки, ассоциированные с прикладным программным обеспечением общего назначения (текстовые редакторы, утилиты, антивирусные мониторы и программные оболочки);
- исполняемые закладки, ассоциированные с исполняемыми программными модулями, содержащими код этой закладки (чаще всего эти модули представляют собой пакетные файлы, т. е. файлы, которые состоят из команд операционной системы, выполняемых одна за одной, как если бы их набирали на клавиатуре компьютера);
- закладки-имитаторы, интерфейс которых совпадает с интерфейсом некоторых служебных программ, требующих ввод конфиденциальной информации (паролей, криптографических ключей, номеров кредитных карточек);
- замаскированные закладки, которые маскируются под программные средства оптимизации работы компьютера (файловые архиваторы, дисковые дефрагментаторы) или под программы игрового и развлекательного назначения.

Чтобы программная закладка могла произвести какие-либо действия по отношению к другим программам или по отношению к данным, процессор должен приступить к исполнению команд, входящих в состав кода программной закладки. Это возможно только при одновременном соблюдении следующих условий:

- программная закладка должна попасть в оперативную память компьютера (если закладка относится к первому типу, то она должна быть загружена до начала работы другой программы, которая является целью воздействия закладки, или во время работы этой программы);
- работа закладки, находящейся в оперативной памяти, начинается при выполнении ряда условий, которые называются активизирующими.

Интересно, что иногда сам пользователь провоцируется на запуск исполняемого файла, содержащего код программной закладки. Известен такой случай. Среди пользователей свободно распространялся набор из архивированных файлов. Для извлечения файлов из него требовалось вызвать специальную утилиту, которая, как правило, есть почти у каждого пользователя и запускается после указания ее имени в командной строке. Однако мало кто из пользователей замечал, что в полученном наборе файлов уже имелась программа с таким же именем и что запускалась именно она. Кроме разар-

хивирования файлов, эта программная закладка дополнительно производила ряд действий негативного характера.

С учетом замечания о том, что программная закладка должна быть обязательно загружена в оперативную память компьютера, можно выделить *резидентные* закладки (они находятся в оперативной памяти постоянно, начиная с некоторого момента и до окончания сеанса работы компьютера, т. е. до его перезагрузки или до выключения питания) и *нерезидентные* (такие закладки попадают в оперативную память компьютера аналогично резидентным, однако, в отличие от последних, выгружаются по истечении некоторого времени или при выполнении особых условий).

Существуют три основные группы деструктивных действий, которые могут осуществляться программными закладками:

- копирование информации пользователя компьютерной системы (паролей, криптографических ключей, кодов доступа, конфиденциальных электронных документов), находящейся в оперативной или внешней памяти этой системы либо в памяти другой компьютерной системы, подключенной к ней через локальную или глобальную компьютерную сеть;
- изменение алгоритмов функционирования системных, прикладных и служебных программ (например, внесение изменений в программу разграничения доступа может привести к тому, что она разрешит вход в систему всем без исключения пользователям вне зависимости от правильности введенного пароля);
- навязывание определенных режимов работы (например, блокирование записи на диск при удалении информации, при этом информация, которую требуется удалить, не уничтожается и может быть впоследствии скопирована хакером).

У всех программных закладок (независимо от метода их внедрения в компьютерную систему, срока их пребывания в оперативной памяти и назначения) имеется одна важная общая черта: они обязательно выполняют операцию записи в оперативную или внешнюю память системы. При отсутствии данной операции никакого негативного влияния программная закладка оказать не может. Ясно, что для целенаправленного воздействия она должна выполнять и операцию чтения, иначе в ней может быть реализована только функция разрушения (например, удаление или замена информации в определенных секторах жесткого диска).

Модели воздействия программных закладок на компьютеры

Перехват

В модели *перехват* программная закладка внедряется в ПЗУ, системное или прикладное программное обеспечение и сохраняет всю или выбранную ин-

формацию, вводимую с внешних устройств компьютерной системы или выводимую на эти устройства, в скрытой области памяти локальной или удаленной компьютерной системы. Объектом сохранения, например, могут служить символы, введенные с клавиатуры (все повторяемые два раза последовательности символов), или электронные документы, распечатываемые на принтере.

Данная модель может быть двухступенчатой. На первом этапе сохраняются только, например, имена или начала файлов. На втором накопленные данные анализируются злоумышленником с целью принятия решения о конкретных объектах дальнейшей атаки.

Модель типа "перехват" может быть эффективно использована при атаке на защищенную операционную систему Windows NT. После старта Windows NT на экране компьютерной системы появляется приглашение нажать клавиши <Ctrl>+<Alt>+<Del>. После их нажатия загружается динамическая библиотека MSGINA.DLL, осуществляющая прием вводимого пароля и выполнение процедуры его проверки (аутентификации). Описание всех функций этой библиотеки можно найти в файле Winwlx.h. Также существует простой механизм замены исходной библиотеки MSGINA.DLL на пользовательскую (для этого необходимо просто добавить специальную строку в реестр операционной системы Windows NT и указать местоположение пользовательской библиотеки). В результате злоумышленник может модифицировать процедуру контроля за доступом к компьютерной системе, работающей под управлением Windows NT.

Искажение

В модели *искажение* программная закладка изменяет информацию, которая записывается в память компьютерной системы в результате работы программ, либо подавляет/инициирует возникновение ошибочных ситуаций в компьютерной системе.

Можно выделить *статическое* и *динамическое* искажение. Статическое искажение происходит всего один раз. При этом модифицируются параметры программной среды компьютерной системы, чтобы впоследствии в ней выполнялись нужные злоумышленнику действия. К статическому искажению относится, например, внесение изменений в файл AUTOEXEC.BAT операционной системы Windows 95/98, которые приводят к запуску заданной программы, прежде чем будут запущены все другие, перечисленные в этом файле.

Специалистам российского Федерального агентства правительственной связи и информации (ФАПСИ)[1] удалось выявить при анализе одной из отече-

[1] ФАПСИ создано 24 декабря 1991 г. указом Президента России. Одним из основных направлений деятельности этого ведомства является организация и ведение внешней разведывательной деятельности в сфере шифрованной, засекреченной и иных видов специальной связи с использованием радиоэлектронных средств и методов.

ственных систем цифровой подписи интересное статистическое искажение. Злоумышленник (сотрудник отдела информатизации финансовой организации, в которой была внедрена данная система) исправил в исполняемом EXE-модуле программы проверки правильности цифровой подписи символьную строку "ПОДПИСЬ НЕКОРРЕКТНА" на символьную строку "ПОДПИСЬ КОРРЕКТНА". В результате вообще перестали фиксироваться документы с неверными цифровыми подписями, и, следовательно, в электронные документы стало можно вносить произвольные изменения уже после их подписания электронной цифровой подписью.

Динамическое искажение заключается в изменении каких-либо параметров системных или прикладных процессов при помощи заранее активизированных закладок. Динамическое искажение можно условно разделить так: искажение *на входе* (когда на обработку попадает уже искаженный документ) и искажение *на выходе* (когда искажается информация, отображаемая для восприятия человеком, или предназначенная для работы других программ).

Практика применения цифровой подписи в системах автоматизированного документооборота показала, что именно программная реализация цифровой подписи особенно подвержена влиянию программных закладок типа "динамическое искажение", которые позволяют осуществлять проводки фальшивых финансовых документов и вмешиваться в процесс разрешения споров по фактам неправомерного применения цифровой подписи. Например, в одной из программных реализаций широко известной криптосистемы PGP электронный документ, под которым требовалось поставить цифровую подпись, считывался блоками по 512 байт, причем процесс считывания считался завершенным, если в прочитанном блоке данные занимали меньше 512 байт. Работа одной программной закладки, выявленной специалистами ФАПСИ, основывалась на навязывании длины файла. Эта закладка позволяла считывать только первые 512 байт документа, и в результате цифровая подпись определялась на основе только этих 512 байт. Такая же схема действовала и при проверке поставленной под документом цифровой подписи. Следовательно, оставшаяся часть этого документа могла быть произвольным образом искажена, и цифровая подпись под ним продолжала оставаться "корректной".

Существуют 4 основных способа воздействия программных закладок на цифровую подпись:

- искажение входной информации (изменяется поступающий на подпись электронный документ);
- искажение результата проверки истинности цифровой подписи (вне зависимости от результатов работы программы цифровая подпись объявляется подлинной);
- навязывание длины электронного документа (программе цифровой подписи предъявляется документ меньшей длины, чем на самом деле, и в

результате цифровая подпись ставится только под частью исходного документа);

- искажение программы цифровой подписи (вносятся изменения в исполняемый код программы с целью модификации реализованного алгоритма).

В рамках модели "искажение" также реализуются программные закладки, действие которых основывается на инициировании или подавлении сигнала о возникновении ошибочных ситуаций в компьютерной системе, т. е. тех, которые приводят к отличному от нормального завершению исполняемой программы (предписанного соответствующей документацией).

Для инициирования статической ошибки на устройствах хранения информации создается область, при обращении к которой (чтение, запись, форматирование и т. п.) возникает ошибка, что может затруднить или блокировать некоторые нежелательные для злоумышленника действия системных или прикладных программ (например, не позволять осуществлять корректно уничтожить конфиденциальную информацию на жестком диске).

При инициировании динамической ошибки для некоторой операции генерируется ложная ошибка из числа тех ошибок, которые могут возникать при выполнении данной операции. Например, для блокирования приема или передачи информации в компьютерной системе может постоянно инициироваться ошибочная ситуация "МОДЕМ ЗАНЯТ". Или при прочтении первого блока информации длиной 512 байт может устанавливаться соответствующий флажок для того, чтобы не допустить прочтения второго и последующих блоков и в итоге подделать цифровую подпись под документом.

Чтобы маскировать ошибочные ситуации, злоумышленники обычно используют подавление статической или динамической ошибки. Целью такого подавления часто является стремление блокировать нормальное функционирование компьютерной системы или желание заставить ее неправильно работать. Чрезвычайно важно, чтобы компьютерная система адекватно реагировала на возникновение всех без исключения ошибочных ситуаций, поскольку отсутствие должной реакции на любую ошибку эквивалентно ее подавлению и может быть использовано злоумышленником. Известен случай успешной атаки пары аргентинских самолетов-торпедоносцев на английский эсминец "Шеффилд", закончившийся нанесением серьезных повреждений этому кораблю. Из-за ошибок в программном обеспечении установленная на нем система противовоздушной обороны не смогла выбрать цель, которую полагалось сбивать первой, поскольку атакующие самолеты летели слишком близко друг от друга.

Разновидностью искажения является также модель типа *троянский конь*. В этом случае программная закладка встраивается в постоянно используемое программное обеспечение и по некоторому активизирующему событию вызывает возникновение сбойной ситуации в компьютерной системе. Тем самым достигаются сразу две цели: парализуется ее нормальное функциони-

рование, а злоумышленник, получив доступ к компьютерной системе для устранения неполадок, сможет, например, извлечь из нее информацию, перехваченную другими программными закладками. В качестве активизирующего события обычно используется наступление определенного момента времени, сигнал из канала модемной связи или состояние некоторых счетчиков (например, счетчика количества запусков программы).

Уборка мусора

Как известно, при хранении компьютерных данных на внешних носителях прямого доступа выделяется несколько уровней иерархии: сектора, кластеры и файлы. Сектора являются единицами хранения информации на аппаратном уровне. Кластеры состоят из одного или нескольких подряд идущих секторов. Файл — это множество кластеров, связанных по определенному закону.

Работа с конфиденциальными электронными документами обычно сводится к последовательности следующих манипуляций с файлами:

- создание;
- хранение;
- коррекция;
- уничтожение.

Для защиты конфиденциальной информации обычно используется шифрование. Основная угроза исходит отнюдь не от использования нестойких алгоритмов шифрования и "плохих" криптографических ключей (как это может показаться на первый взгляд), а от обыкновенных текстовых редакторов и баз данных, применяемых для создания и коррекции конфиденциальных документов!

Дело в том, что подобные программные средства, как правило, в процессе функционирования создают в оперативной или внешней памяти компьютерной системы временные копии документов, с которыми они работают. Естественно, все эти временные файлы выпадают из поля зрения любых программ шифрования и могут быть использованы злоумышленником для того, чтобы составить представление о содержании хранимых в зашифрованном виде конфиденциальных документов.

Важно помнить и о том, что при записи отредактированной информации меньшего объема в тот же файл, где хранилась исходная информация до начала сеанса ее редактирования, образуются так называемые "хвостовые" кластеры, в которых эта исходная информация полностью сохраняется. И тогда "хвостовые" кластеры не только не подвергаются воздействию программ шифрования, но и остаются незатронутыми даже средствами гарантированного стирания информации. Конечно, рано или поздно информация из "хвостовых" кластеров затирается данными из других файлов, однако по

оценкам специалистов ФАПСИ из "хвостовых" кластеров через сутки можно извлечь до 85%, а через десять суток — до 25—40% исходной информации.

Пользователям необходимо иметь в виду и то, что команда удаления файла (DEL) операционной системы DOS[2] не изменяет содержания файла, и оно может быть в любой момент восстановлено, если поверх него еще не был записан другой файл. Распространенные средства гарантированного стирания файлов предварительно записывают на его место константы или случайные числа и только после этого удаляют файл стандартными средствами DOS. Однако даже такие мощные средства оказываются бессильными против программных закладок, которые нацелены на то, чтобы увеличить количество остающихся в виде "мусора" фрагментов конфиденциальной информации. Например, программная закладка может инициировать статическую ошибку, пометив один или несколько кластеров из цепочки, входящей в файл, меткой "СБОЙНЫЙ". В результате при удалении файла средствами операционной системы или средствами гарантированного уничтожения та его часть, которая размещена в сбойных кластерах, останется нетронутой и впоследствии может быть восстановлена с помощью стандартных утилит.

Наблюдение и компрометация

Помимо перечисленных, существуют и другие модели воздействия программных закладок на компьютеры. В частности, при использовании модели типа *наблюдение* программная закладка встраивается в сетевое или телекоммуникационное программное обеспечение. Пользуясь тем, что подобное программное обеспечение всегда находится в состоянии активности, внедренная в него программная закладка может следить за всеми процессами обработки информации в компьютерной системе, а также осуществлять установку и удаление других программных закладок. Модель типа *компрометация* позволяет получать доступ к информации, перехваченной другими программными закладками. Например, инициируется постоянное обращение к такой информации, приводящее к росту соотношения сигнал/шум. А это, в свою очередь, значительно облегчает перехват побочных излучений данной компьютерной системы и позволяет эффективно выделять сигналы, сгенерированные закладкой типа "компрометация", из общего фона излучения, исходящего от оборудования.

Защита от программных закладок

Задача защиты от программных закладок может рассматриваться в трех принципиально различных вариантах:

---

[2] За англоязычной аббревиатурой DOS скрывается Disk Operating System — Дисковая операционная система, разработанная корпорацией Microsoft в 1981 г. Эта операционная система управляется командной строкой, то есть, в ответ на приглашение пользователь должен набрать команду на клавиатуре и затем нажать специальную клавишу ввода, чтобы набранную им команду получила и выполнила операционная система.

- не допустить внедрения программной закладки в компьютерную систему;
- выявить внедренную программную закладку;
- удалить внедренную программную закладку.

При рассмотрении этих вариантов решение задачи защиты от программных закладок сходно с решением проблемы защиты компьютерных систем от вирусов. Как и в случае борьбы с вирусами, задача решается с помощью средств контроля за целостностью запускаемых системных и прикладных программ, а также за целостностью информации, хранимой в компьютерной системе и за критическими для функционирования системы событиями. Однако данные средства действенны только тогда, когда сами они не подвержены влиянию программных закладок, которые могут:

- навязывать конечные результаты контрольных проверок;
- влиять на процесс считывания информации и запуск программ, за которыми осуществляется контроль;
- изменять алгоритмы функционирования средств контроля.

При этом чрезвычайно важно, чтобы включение средств контроля выполнялось до начала воздействия программной закладки либо когда контроль осуществлялся только с использованием программ управления, находящихся в ПЗУ компьютерной системы.

Защита от внедрения программных закладок

Универсальным средством защиты от внедрения программных закладок является создание *изолированного* компьютера. Компьютер называется изолированным, если выполнены следующие условия:

- в нем установлена система BIOS, не содержащая программных закладок;
- операционная система проверена на наличие в ней закладок;
- достоверно установлена неизменность BIOS и операционной системы для данного сеанса;
- на компьютере не запускалось и не запускается никаких иных программ, кроме уже прошедших проверку на присутствие в них закладок;
- исключен запуск проверенных программ в каких-либо иных условиях, кроме перечисленных выше, т. е. вне изолированного компьютера.

Для определения степени изолированности компьютера может использоваться модель ступенчатого контроля. Сначала проверяется, нет ли изменений в BIOS. Затем, если все в порядке, считывается загрузочный сектор диска и драйверы операционной системы, которые, в свою очередь, также анализируются на предмет внесения в них несанкционированных изменений. И наконец, с помощью операционной системы запускается драйвер контроля вызовов программ, который следит за тем, чтобы в компьютере запускались только проверенные программы.

Интересный метод борьбы с внедрением программных закладок может быть использован в информационной банковской системе, в которой циркулируют исключительно файлы-документы. Чтобы не допустить проникновения программной закладки через каналы связи, в этой системе не допускается прием никакого исполняемого кода. Для распознавания событий типа "ПОЛУЧЕН ИСПОЛНЯЕМЫЙ КОД" и "ПОЛУЧЕН ФАЙЛ-ДОКУМЕНТ" применяется контроль за наличием в файле запрещенных символов: файл признается содержащим исполняемый код, если в нем присутствуют символы, которые никогда не встречаются в файлах-документах.

Выявление внедренной программной закладки

Выявление внедренного кода программной закладки заключается в обнаружении признаков его присутствия в компьютерной системе. Эти признаки можно разделить на следующие два класса:

- качественные и визуальные;
- обнаруживаемые средствами тестирования и диагностики.

К качественным и визуальным признакам относятся ощущения и наблюдения пользователя компьютерной системы, который отмечает определенные отклонения в ее работе (изменяется состав и длины файлов, старые файлы куда-то пропадают, а вместо них появляются новые, программы начинают работать медленнее, или заканчивают свою работу слишком быстро, или вообще перестают запускаться). Несмотря на то что суждение о наличии признаков этого класса кажется слишком субъективным, тем не менее, они часто свидетельствуют о наличии неполадок в компьютерной системе и, в частности, о необходимости проведения дополнительных проверок присутствия программных закладок. Например, пользователи пакета шифрования и цифровой подписи "Криптоцентр" с некоторых пор стали замечать, что цифровая подпись под электронными документами ставится слишком быстро. Исследование, проведенное специалистами ФАПСИ, показало присутствие программной закладки, работа которой основывалась на навязывании длины файла. В другом случае тревогу забили пользователи пакета шифрования и цифровой подписи "Криптон", которые с удивлением отметили, что скорость шифрования по криптографическому алгоритму ГОСТ 28147-89 вдруг возросла более, чем в 30 раз. А в третьем случае программная закладка обнаружила свое присутствие в программе клавиатурного ввода тем, что пораженная ею программа перестала нормально работать.

Признаки, выявляемые с помощью средств тестирования и диагностики, характерны как для программных закладок, так и для компьютерных вирусов. Например, загрузочные закладки успешно обнаруживаются антивирусными программами, которые сигнализируют о наличии подозрительного кода в загрузочном секторе диска. С инициированием статической ошибки на дисках хорошо справляется Disk Doctor, входящий в распространенный

комплект утилит Norton Utilities. А средства проверки целостности данных на диске типа Adinf позволяют успешно выявлять изменения, вносимые в файлы программными закладками. Кроме того, эффективен поиск фрагментов кода программных закладок по характерным для них последовательностям нулей и единиц (сигнатурам), а также разрешение выполнения только программ с известными сигнатурами.

Удаление внедренной программной закладки

Конкретный способ удаления внедренной программной закладки зависит от метода ее внедрения в компьютерную систему. Если это программно-аппаратная закладка, то следует перепрограммировать ПЗУ компьютера. Если это загрузочная, драйверная, прикладная, замаскированная закладка или закладка-имитатор, то можно заменить их на соответствующую загрузочную запись, драйвер, утилиту, прикладную или служебную программу, полученную от источника, заслуживающего доверия. Наконец, если это исполняемый программный модуль, то можно попытаться добыть его исходный текст, убрать из него имеющиеся закладки или подозрительные фрагменты, а затем заново откомпилировать.

Троянские программы

Троянской программой (троянцем, или троянским конем) называется:

- программа, которая, являясь частью другой программы с известными пользователю функциями, способна втайне от него выполнять некоторые дополнительные действия с целью причинения ему определенного ущерба;
- программа с известными ее пользователю функциями, в которую были внесены изменения, чтобы, помимо этих функций, она могла втайне от него выполнять некоторые другие (разрушительные) действия.

Таким образом, троянская программа — это особая разновидность программной закладки. Она дополнительно наделена функциями, о существовании которых пользователь даже не подозревает. Когда троянская программа выполняет эти функции, компьютерной системе наносится определенный ущерб. Однако то, что при одних обстоятельствах причиняет непоправимый вред, при других — может оказаться вполне полезным. К примеру, программу, которая форматирует жесткий диск, нельзя назвать троянской, если она как раз и предназначена для его форматирования (как это делает команда format операционной системы DOS). Но если пользователь, выполняя некоторую программу, совершенно не ждет, что она отформатирует его винчестер, — это и есть самый настоящий троянец.

Короче говоря, троянской можно считать любую программу, которая втайне от пользователя выполняет какие-то нежелательные для него действия. Эти действия могут быть любыми — от определения регистрационных номеров

программного обеспечения, установленного на компьютере, до составления списка каталогов на его жестком диске. А сама троянская программа может маскироваться под текстовый редактор, под сетевую утилиту или любую программу, которую пользователь пожелает установить на свой компьютер.

Откуда берутся троянские программы

Троянская программа — это плод труда программиста. Никаким другим способом создать ее невозможно. Программист, пишущий троянскую программу, прекрасно осознает, чего он хочет добиться, и в своих намерениях он всегда весьма далек от альтруизма.

Большинство троянских программ предназначено для сбора конфиденциальной информации. Их задача, чаще всего, состоит в выполнении действий, позволяющих получить доступ к данным, которые не подлежат широкой огласке. К таким данным относятся пользовательские пароли, регистрационные номера программ, сведения о банковских счетах и т. д. Остальные троянцы создаются для причинения прямого ущерба компьютерной системе, приводя ее в неработоспособное состояние.

К последним можно отнести, например, троянскую программу PC CYBORG, которая завлекала ничего не подозревающих пользователей обещаниями предоставить им новейшую информацию о борьбе с вирусом, вызывающим синдром приобретенного иммунодефицита (СПИД). Проникнув в компьютерную систему, PC CYBORG отсчитывала 90 перезагрузок этой системы, а затем прятала все каталоги на ее жестком диске и шифровала находящиеся там файлы.

Другая троянская программа называлась AOLGOLD. Она рассылалась по электронной почте в виде заархивированного файла. В сопроводительном письме, прилагавшемся к этому файлу, говорилось о том, что AOLGOLD предназначена для повышения качества услуг, которые предоставляет своим пользователям крупнейший американский Internet-провайдер America Online (AOL). Архив состоял из двух файлов, один из которых именовался INSTALL.BAT. Пользователь, запустивший INSTALL.BAT, рисковал стереть все файлы из каталогов C:\, C:\DOS, C:\WINDOWS и C:\WINDOWS\SYSTEM на своем жестком диске.

Подобного рода троянские программы, как правило, создаются подростками, которые хотя и одержимы страстью к разрушению, но не имеют глубоких познаний в программировании и поэтому не могут причинить существенный ущерб компьютерным системам, подвергшимся нападению созданных ими троянцев. Например, программа AOLGOLD стирала себя с жесткого диска, будучи запущена из любого другого дискового раздела за исключением C.

Другое дело — троянские программы, авторами которых являются профессиональные программисты, занимающиеся разработкой программного обес-

печения в солидных фирмах. Троянцы, входящие в распространенные компьютерные приложения, утилиты и операционные системы, представляют значительно большую угрозу компьютерам, на которых они установлены, поскольку их действия носят не деструктивный характер, а имеют целью сбор конфиденциальной информации о системе. Обнаружить такие троянские программы удается, как правило, чисто случайно. А поскольку программное обеспечение, частью которого они являются, в большинстве случаев используется не только какой-то одной компанией, закупившей это программное обеспечение, но также на крупных Internet-серверах и, кроме того, распространяется через Internet, последствия могут оказаться самыми плачевными.

Случается и так, что троянцы встраиваются в некоторые утилиты программистами, не имеющими никакого отношения к разработке этих утилит. Например, в дистрибутив сканера SATAN, предназначенный для установки на компьютеры с операционной системой Linux, распространявшийся через Internet, попала троянская программа, которая "обосновалась" в утилите fping. При первом же запуске модифицированной утилиты fping в файл /etc/passwd добавлялась запись для пользователя с именем suser, который в результате мог войти в Linux и тайно получить там полномочия администратора. Однако у автора этой троянской программы были явные пробелы в компьютерном образовании. В частности, он не знал некоторых нюансов хранения паролей в операционных системах семейства UNIX. В результате файл /etc/passwd был соответствующим образом изменен всего лишь на двух компьютерах, на которых был установлен этот испорченный дистрибутив сетевого анализатора SATAN для Linux.

Где обитают и как часто встречаются троянские программы

В настоящее время троянские программы можно отыскать практически где угодно. Они написаны для всех без исключения операционных систем и для любых платформ. Не считая случаев, когда троянские программы пишутся самими разработчиками программного обеспечения, троянцы распространяются тем же способом, что и компьютерные вирусы. Поэтому самыми подозрительными на предмет присутствия в них троянцев, в первую очередь, являются бесплатные и условно-бесплатные программы, скачанные из Internet, а также программное обеспечение, распространяемое на пиратских компакт-дисках.

Например, в январе 1999 г. было обнаружено, что популярная утилита TCP Wrapper, предназначенная для администрирования UNIX-систем и бесплатно распространяемая через Internet, на многих ftp-сайтах была заменена внешне похожей на нее программой, которая на самом деле являлась троянцем. После инсталляции он отправлял электронное сообщение по опреде-

ленным внешним адресам, оповещая своего хозяина об успешном внедрении. Потом он ждал, пока будет установлено удаленное соединение с портом 421 зараженного им компьютера, и предоставлял привилегированные права доступа через этот порт.

Другая троянская программа распространялась среди пользователей AOL в виде вложения в письмо, рассылаемое по электронной почте. Открывшие это вложение заражали свой компьютер троянцем, который пытался найти пароль для подключения к AOL и в случае успеха шифровал его, а потом отсылал электронной почтой куда-то в Китай.

В настоящее время существует целый ряд троянских программ, которые можно совершенно свободно скачать, подключившись к глобальной компьютерной сети Internet. Наибольшую известность среди них получили троянцы Back Orifice, Net Bus и SubSeven (рис. 2.1—2.3). На Web-узле группы разработчиков Back Orifice, которая именует себя Cult of Dead Cow (Культ мертвой коровы), можно даже найти с десяток постеров, которые предназначены для рекламы ее последней разработки — троянца Back Orifice 2000 (один из таких постеров приведен на рис. 2.4).

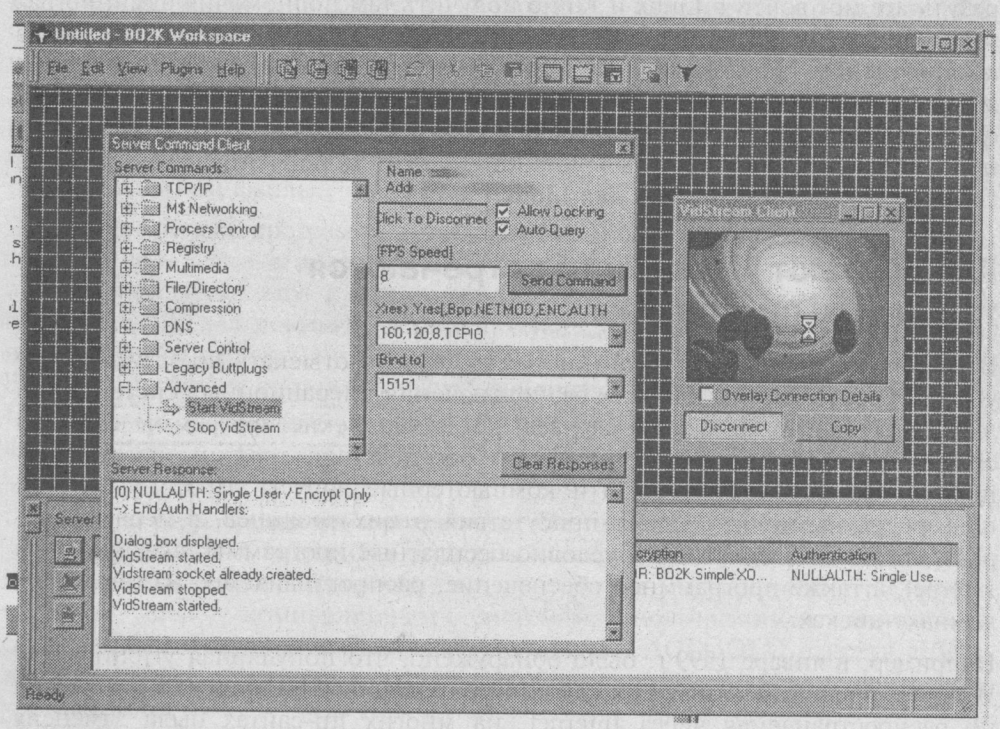


Рис. 2.1. Основное рабочее окно троянской программы Back Orifice 2000

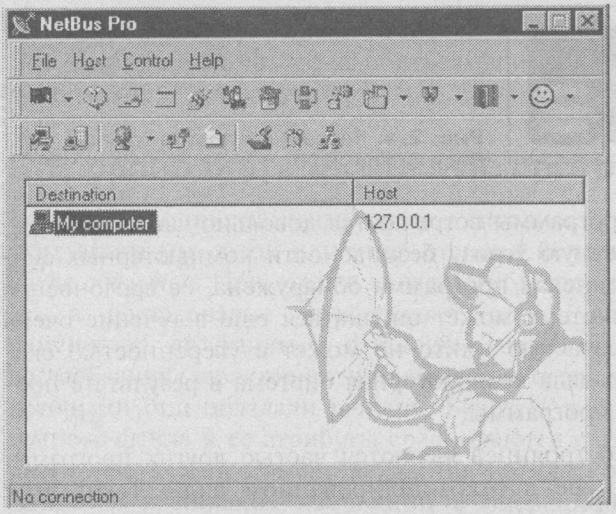


Рис. 2.2. Основное рабочее окно троянской программы NetBus

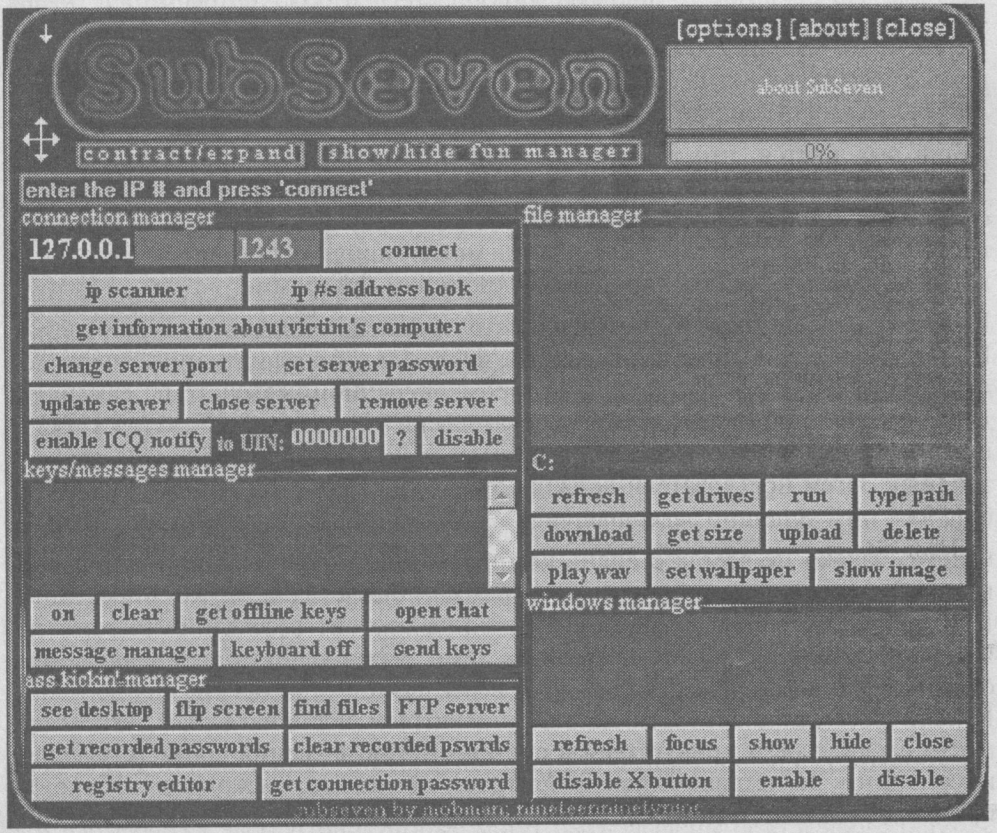


Рис. 2.3. Основное рабочее окно троянской программы SubSeven

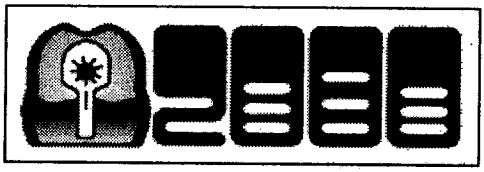


Рис. 2.4. Рекламный постер троянца Back Orifice 2000

Таким образом, троянские программы встречаются довольно часто и, следовательно, представляют серьезную угрозу безопасности компьютерных систем. Даже после того как троянская программа обнаружена, ее вредоносное влияние на компьютерную систему может ощущаться еще в течение очень длительного времени. Ведь зачастую никто не может с уверенностью сказать, насколько сильно пострадала компьютерная система в результате проникновения в нее троянской программы.

Дело в том, что большинство троянцев являются частью других программ, которые хранятся в компьютере в откомпилированном виде. Текст этих программ представляет собой последовательность команд на машинном языке, состоящую из нулей и единиц. Рядовой пользователь, как правило, не имеет ни малейшего понятия о внутренней структуре таких программ. Он просто запускает их на исполнение путем задания имени соответствующей программы в командной строке или двойным щелчком на имени ее файла.

Когда выясняется, что в какую-то откомпилированную программу проник троянец, в сети Internet немедленно начинают распространяться бюллетени с информацией об обнаруженном троянце. Чаще всего в этих бюллетенях кратко сообщается о том, какой вред может причинить данный троянец и где можно найти замену пораженной троянцем программе.

Иногда ущерб, который может нанести троянец, оценить довольно легко. Например, если он предназначен для пересылки по электронной почте содержимого файла /etc/passwd, в котором операционные системы семейства UNIX хранят информацию о пользовательских паролях, достаточно установить "чистую" версию программы взамен той, в которой обосновался этот троянец. Затем пользователи должны будут обновить свои пароли, и на этом борьба с ним успешно завершается.

Однако далеко не всегда степень компрометации компьютерной системы, в которой поселилась троянская программа, бывает так легко определить. Предположим, что цель внедрения троянца состоит в создании дыры в защитных механизмах компьютерной системы, через которую злоумышленник сможет, например, проникать в нее, имея администраторские полномочия. И если взломщик окажется достаточно хитрым и смекалистым, чтобы замести следы своего присутствия в системе путем внесения соответствующих изменений в регистрационные файлы, то определить, насколько глубоко он проник сквозь системные защитные механизмы, будет почти невозможно, если учесть еще тот факт, что саму троянскую программу обнаружат лишь

несколько месяцев спустя после ее внедрения в компьютерную систему. В этом случае может понадобиться целиком переустановить операционную систему и все приложения.

Как распознать троянскую программу

Большинство программных средств, предназначенных для защиты от троянских программ, в той или иной степени использует так называемое *согласование объектов*. При этом в качестве объектов фигурируют файлы и каталоги, а согласование представляет собой способ ответить на вопрос, изменились ли файлы и каталоги с момента последней проверки. В ходе согласования характеристики объектов сравниваются с характеристиками, которыми они обладали раньше. Берется, к примеру, архивная копия системного файла и ее атрибуты сравниваются с атрибутами этого файла, который в настоящий момент находится на жестком диске. Если атрибуты различаются и никаких изменений в операционную систему не вносилось, значит в компьютер, скорее всего, проник троянец.

Одним из атрибутов любого файла является отметка о времени его последней модификации: всякий раз, когда файл открывается, изменяется и сохраняется на диске, автоматически вносятся соответствующие поправки. Однако отметка времени не может служить надежным индикатором наличия в системе троянца. Дело в том, что ею очень легко манипулировать. Можно подкрутить назад системные часы, внести изменения в файл, затем снова вернуть часы в исходное состояние, и отметка о времени модификации файла останется неизменной.

Может быть, иначе обстоит дело с размером файла? Отнюдь. Нередко текстовый файл, который изначально занимал, скажем, 8 Кбайт дискового пространства, после редактирования и сохранения имеет тот же самый размер. Несколько иначе ведут себя двоичные файлы. Вставить в чужую программу фрагмент собственного кода так, чтобы она не утратила работоспособности и в откомпилированном виде сохранила свой размер, достаточно непросто. Поэтому размер файла является более надежным показателем, чем отметка о времени внесения в него последних изменений.

Злоумышленник, решивший запустить в компьютер троянца, обычно пытается сделать его частью системного файла. Такие файлы входят в дистрибутив операционной системы и их присутствие на любом компьютере, где эта операционная система установлена, не вызывает никаких подозрений. Однако любой системный файл имеет вполне определенную длину. Если данный атрибут будет каким-либо образом изменен, это встревожит пользователя.

Зная это, злоумышленник постарается достать исходный текст соответствующей программы и внимательно проанализирует его на предмет присутствия в нем избыточных элементов, которые могут быть удалены безо всякого ощутимого ущерба. Тогда вместо найденных избыточных элементов он вставит в

программу своего троянца и перекомпилирует ее заново. Если размер полученного двоичного файла окажется меньше или больше размера исходного, процедура повторяется. И так до тех пор, пока не будет получен файл, размер которого в наибольшей степени близок к оригиналу (если исходный файл достаточно большой, этот процесс может растянуться на несколько дней).

Итак, в борьбе с троянцами положиться на отметку о времени последней модификации файла и его размер нельзя, поскольку злоумышленник может их довольно легко подделать. Более надежной в этом отношении является так называемая *контрольная сумма* файла. Для ее подсчета элементы файла суммируются, и получившееся в результате число объявляется его контрольной суммой. Например, в операционной системе SunOS существует специальная утилита sum, которая выводит на устройство стандартного вывода STDOUT контрольную сумму файлов, перечисленных в строке аргументов этой утилиты.

Однако и контрольную сумму в общем случае оказывается не так уж трудно подделать. Поэтому для проверки целостности файловой системы компьютера используется особая разновидность алгоритма вычисления контрольной суммы, называемая *односторонним хэшированием*.

Функция хэширования называется односторонней, если задача отыскания двух аргументов, для которых ее значения совпадают, является труднорешаемой. Отсюда следует, что функция одностороннего хэширования может быть применена для того, чтобы отслеживать изменения, вносимые злоумышленником в файловую систему компьютера, поскольку попытка злоумышленника изменить какой-либо файл так, чтобы значение, полученное путем одностороннего хэширования этого файла, осталось неизменным, обречена на неудачу.

Исторически сложилось так, что большинство утилит, позволяющих бороться с проникновением в компьютерную систему троянских программ путем однонаправленного хэширования файлов, было создано для операционных систем семейства UNIX. Одной из наиболее удобных в эксплуатации и эффективных является утилита TripWire, которую можно найти в Internet по адресу **http://www.tripwiresecurity.com/**. Она позволяет производить однонаправленное хэширование файлов при помощи нескольких алгоритмов, в том числе — MD4[3], MD5[4] и SHA[5]. Вычисленные хэш-значения файлов хранятся в специ-

[3] В алгоритме хэширования MD4 исходная битовая последовательность дополняется так, чтобы ее длина в битах плюс 64 нацело делилась на 512. Затем к ней приписывается 64-битовое значение ее первоначальной длины. Полученная таким образом новая последовательность обрабатывается блоками по 512 бит с помощью специальной итерационной процедуры. В результате на выходе MD4 получается так называемая "выжимка" исходной последовательности, имеющая длину 128-бит. Алгоритм MD4 оптимизирован для 32-разрядных аппаратных платформ и работает довольно быстро.

альной базе данных, которая, в принципе, является самым уязвимым звеном утилиты TripWire. Поэтому пользователям TripWire предлагается в обязательном порядке принимать дополнительные меры защиты, чтобы исключить доступ к этой базе данных со стороны злоумышленника (например, помещать ее на съемном носителе, предназначенном только для чтения).

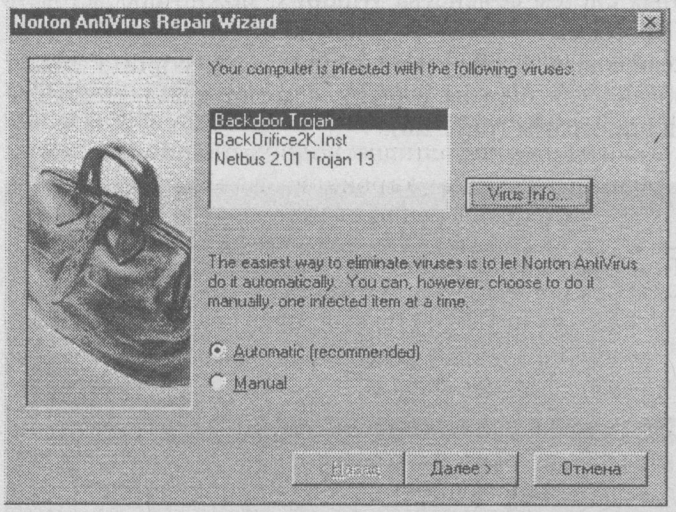


Рис. 2.5. Окно антивирусной программы Norton Antivirus 2000

Средства борьбы с троянцами в операционных системах семейства Windows (95/98/NT) традиционно являются частью их антивирусного программного обеспечения. Поэтому, чтобы отлавливать Back Orifice, NetBus, SubSeven и другие подобные им троянские программы, необходимо обзавестись самым современным антивирусом (например, программой Norton Antivirus 2000 компании Symantec (рис. 2.5), которая позволяет обнаруживать присутствие

---

[4] Алгоритм хэширования MD5 очень похож на MD4 и по способу дополнения исходной битовой последовательности, и по методу ее обработки, и по размеру получаемой "выжимки" (те же 128 бит). Однако каждый 512-битовый блок подвергается не трем, как в MD4, а четырем циклам преобразований, и поэтому алгоритм MD5 работает несколько медленнее, чем MD4.

[5] Алгоритм SHA (Secure Hash Algorithm — Стойкий алгоритм хэширования) был принят в качестве федерального стандарта США в 1993 году. Последовательность преобразований, которым в рамках этого алгоритма подвергается исходная битовая последовательность, в целом аналогичная действиям, осуществляемым алгоритмом MD4, однако получаемая на выходе "выжимка" несколько длиннее — 160 бит. В исходном варианте SHA вскоре после его опубликования был обнаружен существенный изъян, и год спустя в этот алгоритм были внесены поправки. Более поздняя модификация SHA получила название SHA-1.

в компьютерной системе наиболее распространенных троянцев и избавляться от них). Следует регулярно проверять свой компьютер на присутствие в нем вирусов.

Тем, кто хочет иметь в своем распоряжении утилиту, предназначенную именно для обнаружения троянцев в компьютерах, которые работают под управлением операционных систем семейства Windows, можно посоветовать обратить свои взоры на программу The Cleaner компании MooSoft Development (**http://www.homestead.com/moosoft/ cleaner.html**). Эта утилита может быть с успехом использована для борьбы с более чем четырьмя десятками разновидностей троянских программ (рис. 2.6).

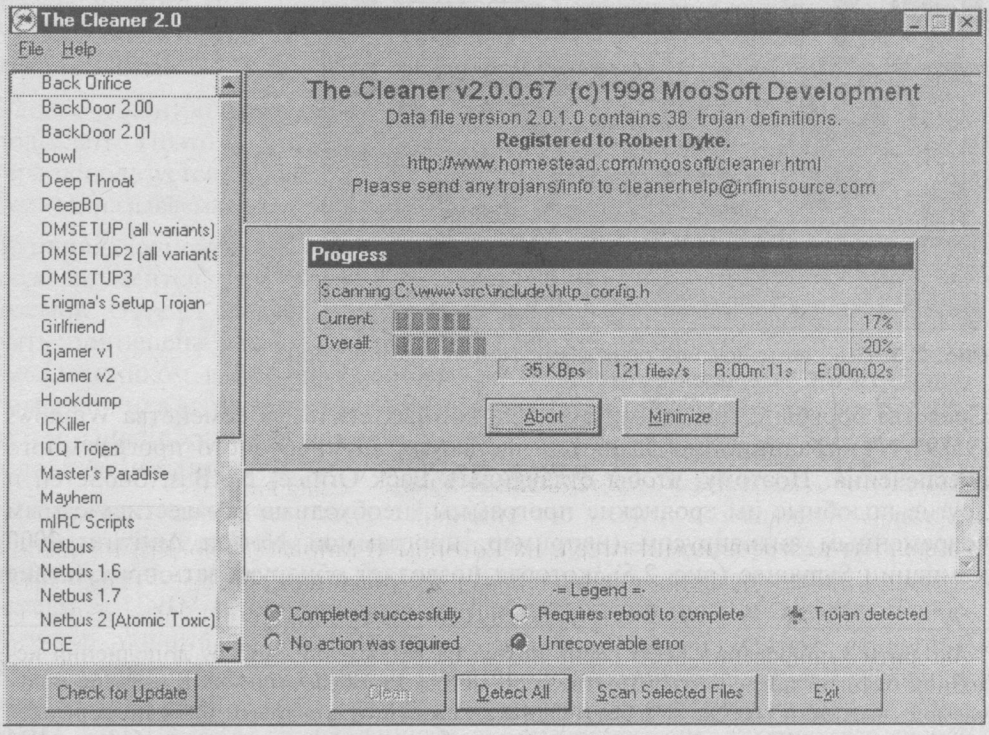


Рис. 2.6. Основное рабочее окно программы The Cleaner

Обзор средств борьбы с троянскими программами был бы далеко не полным, если обойти вниманием недавно появившиеся на рынке программные пакеты, предназначенные для комплексной защиты от угроз, с которыми сталкиваются пользователи настольных компьютеров при работе в Internet. Одним из таких пакетов является eSafe Protect компании Aladdin Knowledge Systems (демонстрационную версию eSafe Protect можно найти в Internet по адресу **www.esafe.com**).

Функционально eSafe Protect делится на три компонента — антивирус, персональный брандмауэр и модуль защиты компьютерных ресурсов. Антивирус избавляет компьютер от вредоносных программ благодаря применению антивирусного модуля VisuSafe, сертифицированного американским Национальным агентством компьютерной безопасности. Персональный брандмауэр контролирует весь входящий и исходящий трафик по протоколу TCP/IP, наделяя используемые IP-адреса определенными правами (например, ограничивая доступ в Internet в определенные часы или запрещая посещение некоторых Web-узлов). С помощью окна, представленного на рис. 2.7, осуществляется доступ к конфигурационным настройкам входящих в eSafe Protect компонентов, производится запуск антивируса и дискретно устанавливается степень защищенности компьютерной системы.

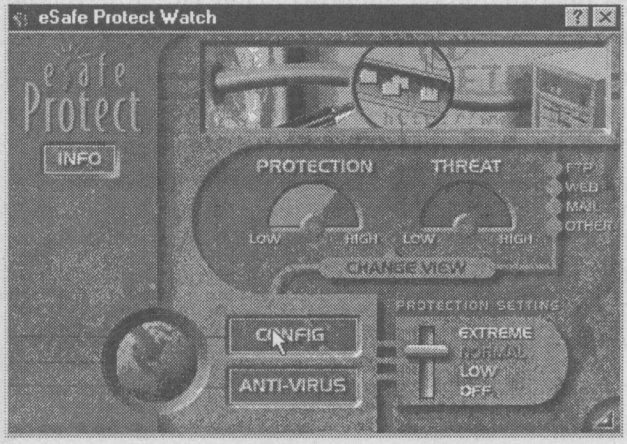


Рис. 2.7. Рабочее окно программного пакета eSafe Protect

Для защиты ресурсов компьютера, на котором установлен программный пакет eSafe Protect, создается специальная изолированная область — так называемая *песочница*. Все автоматически загружаемые из Internet Java-аплеты и компоненты ActiveX сначала помещаются в "песочницу", где они находятся под неусыпным наблюдением eSafe Protect. Если попавшая в "песочницу" программа попытается выполнить какое-либо недозволенное действие, то оно будет немедленно блокировано. В течение заданного интервала времени (от 1 до 30 дней) каждое приложение, загруженное в компьютер из Internet, проходит "карантинную" проверку в "песочнице". Полученная в ходе такой проверки информация заносится в особый журнал. По истечении "карантина" приложение будет выполняться вне "песочницы", однако ему будут дозволены только те действия, перечень которых определяется на основе имеющихся журнальных записей.

Таким образом, по сравнению с другими подобными программными пакетами eSafe Protect обеспечивает наиболее развитые и эффективные средства

комплексной защиты от троянских программ. Входящий в состав eSafe Protect антивирус помогает быстро выявлять троянцев и расправляться с ними, используя технологии, которые хорошо зарекомендовали себя при борьбе с вирусами. Персональный брандмауэр блокирует любые попытки связаться извне с проникшими в компьютерную систему троянскими программами. И наконец с помощью "песочницы" своевременно предотвращается внедрение троянцев в компьютеры под видом Java-аплетов и компонентов ActiveX.

Клавиатурные шпионы

Одна из наиболее распространенных разновидностей программных закладок — *клавиатурные шпионы*. Такие программные закладки нацелены на перехват паролей пользователей операционной системы, а также на определение их легальных полномочий и прав доступа к компьютерным ресурсам.

Клавиатурные шпионы — явление отнюдь не новое в мире компьютеров. В свое время они разрабатывались и для OS/370, и для UNIX, и для DOS. Их поведение в общем случае является довольно традиционным: типовой клавиатурный шпион обманным путем завладевает пользовательскими паролями, а затем переписывает эти пароли туда, откуда их может без особого труда извлечь злоумышленник. Различия между клавиатурными шпионами касаются только способа, который применяется ими для перехвата пользовательских паролей. Соответственно все клавиатурные шпионы делятся на три типа — *имитаторы, фильтры и заместители*.

Имитаторы

Клавиатурные шпионы этого типа работают по следующему алгоритму. Злоумышленник внедряет в операционную систему программный модуль, который имитирует приглашение пользователю зарегистрироваться для того, чтобы войти в систему. Затем внедренный модуль (в принятой терминологии — имитатор) переходит в режим ожидания ввода пользовательского идентификатора и пароля. После того как пользователь идентифицирует себя и введет свой пароль, имитатор сохраняет эти данные там, где они доступны злоумышленнику. Далее имитатор инициирует выход из системы (что в большинстве случаев можно сделать программным путем), и в результате перед глазами у ничего не подозревающего пользователя появляется еще одно, но на этот раз уже настоящее приглашение для входа в систему.

Обманутый пользователь, видя, что ему предлагается еще раз ввести пароль, приходит к выводу о том, что он допустил какую-то ошибку во время предыдущего ввода пароля, и послушно повторяет всю процедуру входа в систему заново. Некоторые имитаторы для убедительности выдают на экран монитора

правдоподобное сообщение о якобы совершенной пользователем ошибке. Например, такое: "НЕВЕРНЫЙ ПАРОЛЬ. ПОПРОБУЙТЕ ЕЩЕ РАЗ".

Написание имитатора не требует от его создателя каких-либо особых навыков. Злоумышленнику, умеющему программировать на одном из универсальных языков программирования (к примеру, на языке BASIC), понадобятся на это считанные часы. Единственная трудность, с которой он может столкнуться, состоит в том, чтобы отыскать в документации соответствующую программную функцию, реализующую выход пользователя из системы.

Перехват пароля зачастую облегчают сами разработчики операционных систем, которые не затрудняют себя созданием усложненных по форме приглашений пользователю зарегистрироваться для входа в систему. Подобное пренебрежительное отношение характерно для большинства версий операционной системы UNIX, в которых регистрационное приглашение состоит из двух текстовых строк, выдаваемых поочередно на экран терминала:

login:

password:

Чтобы подделать такое приглашение, не нужно быть семи пядей во лбу. Однако само по себе усложнение внешнего вида приглашения не создает для хакера, задумавшего внедрить в операционную систему имитатор, каких-либо непреодолимых препятствий. Для этого требуется прибегнуть к более сложным и изощренным мерам защиты. В качестве примера операционной системы, в которой такие меры в достаточно полном объеме реализованы на практике, можно привести Windows NT.

Системный процесс WinLogon, отвечающий в операционной системе Windows NT за аутентификацию пользователей, имеет свой собственный рабочий стол — совокупность окон, одновременно видимых на экране дисплея. Этот рабочий стол называется *столом аутентификации*. Никакой другой процесс, в том числе и имитатор, не имеет доступа к рабочему столу аутентификации и не может расположить на нем свое окно.

После запуска Windows NT на экране компьютера возникает так называемое начальное окно рабочего стола аутентификации, содержащее указание нажать на клавиатуре клавиши <Ctrl>+<Alt>+<Del>. Сообщение о нажатии этих клавиш передается только системному процессу WinLogon, а для остальных процессов, в частности, для всех прикладных программ, их нажатие происходит совершенно незаметно. Далее производится переключение на другое, так называемое *регистрационное* окно рабочего стола аутентификации. В нем-то как раз и размещается приглашение пользователю ввести свое идентификационное имя и пароль, которые будут восприняты и проверены процессом WinLogon.

Для перехвата пользовательского пароля внедренный в Windows NT имитатор обязательно должен уметь обрабатывать нажатие пользователем клавиш

<Ctrl>+<Alt>+<Del>. В противном случае произойдет переключение на регистрационное окно рабочего стола аутентификации, имитатор станет неактивным и не сможет ничего перехватить, поскольку все символы пароля, введенные пользователем, минуют имитатор и станут достоянием исключительно системного процесса WinLogon. Как уже говорилось, процедура регистрации в Windows NT устроена таким образом, что нажатие клавиш <Ctrl>+<Alt>+<Del> проходит бесследно для всех процессов, кроме WinLogon, и поэтому пользовательский пароль поступит именно ему.

Конечно, имитатор может попытаться воспроизвести не начальное окно рабочего стола аутентификации (в котором высвечивается указание пользователю одновременно нажать клавиши <Ctrl>+<Alt>+<Del>), а регистрационное (где содержится приглашение ввести идентификационное имя и пароль пользователя). Однако при отсутствии имитаторов в системе регистрационное окно автоматически заменяется на начальное по прошествии короткого промежутка времени (в зависимости от версии Window NT он может продолжаться от 30 с до 1 мин), если в течение этого промежутка пользователь не предпринимает никаких попыток зарегистрироваться в системе. Таким образом, сам факт слишком долгого присутствия на экране регистрационного окна должен насторожить пользователя Windows NT и заставить его тщательно проверить свою компьютерную систему на предмет наличия в ней программных закладок.

Подводя итог сказанному, можно отметить, что степень защищенности Windows NT от имитаторов достаточно высока. Рассмотрение защитных механизмов, реализованных в этой операционной системе, позволяет сформулировать два необходимых условия, соблюдение которых является обязательным для обеспечения надежной защиты от имитаторов:

- системный процесс, который при входе пользователя в систему получает от него соответствующие регистрационное имя и пароль, должен иметь свой собственный рабочий стол, недоступный другим процессам;
- переключение на регистрационное окно рабочего стола аутентификации должно происходить абсолютно незаметно для прикладных программ, которые к тому же никак не могут повлиять на это переключение (например, запретить его).

К сожалению, эти два условия ни в одной из операционных систем, за исключением Windows NT, не соблюдаются. Поэтому для повышения их защищенности от имитаторов можно порекомендовать воспользоваться административными мерами. Например, обязать каждого пользователя немедленно сообщать системному администратору, когда вход в систему оказывается невозможен с первого раза, несмотря на корректно заданное идентификационное имя и правильно набранный пароль.

Фильтры

Фильтры "охотятся" за всеми данными, которые пользователь операционной системы вводит с клавиатуры компьютера. Самые элементарные фильтры просто сбрасывают перехваченный клавиатурный ввод на жесткий диск или в какое-то другое место, к которому имеет доступ злоумышленник. Более изощренные программные закладки этого типа подвергают перехваченные данные анализу и отфильтровывают информацию, имеющую отношение к пользовательским паролям.

Фильтры являются резидентными программами, перехватывающими одно или несколько прерываний, которые связаны с обработкой сигналов от клавиатуры. Эти прерывания возвращают информацию о нажатой клавише и введенном символе, которая анализируется фильтрами на предмет выявления данных, имеющих отношение к паролю пользователя.

Известны несколько фильтров, созданных специально для различных версий операционной системы DOS. В 1997 г. отмечено появление фильтров для операционных систем Windows 3.11 и Windows 95.

Надо сказать, что изготовить подобного рода программную закладку не составляет большого труда. В операционных системах Windows 3.11 и Windows 95/98 предусмотрен специальный программный механизм, с помощью которого в них решается ряд задач, связанных с получением доступа к вводу с клавиатуры, в том числе и проблема поддержки национальных раскладок клавиатур. К примеру, любой клавиатурный русификатор для Windows представляет собой самый что ни на есть настоящий фильтр, поскольку призван перехватывать все данные, вводимые пользователем с клавиатуры компьютера. Нетрудно "доработать" его таким образом, чтобы вместе со своей основной функцией (поддержка национальной раскладки клавиатуры) он заодно выполнял бы и действия по перехвату паролей. Тем более, что во многих учебных пособиях и руководствах пользователя операционных систем Windows имеются исходные тексты программных русификаторов клавиатуры. "Перепрофилировав" этот русификатор так, чтобы он взял на себя выполнение функций клавиатурного шпиона, его можно встроить перед настоящим русификатором или после него, и в результате вся информация, вводимая пользователем с клавиатуры, пойдет и через клавиатурного шпиона. Таким образом задача создания фильтра становится такой простой, что не требует наличия каких-либо специальных знаний у злоумышленника. Ему остается только незаметно внедрить изготовленную им программную закладку в операционную систему и умело замаскировать ее присутствие.

В общем случае можно утверждать, что если в операционной системе разрешается переключать клавиатурную раскладку во время ввода пароля, то для этой операционной системы возможно создание фильтра. Поэтому, чтобы обезопасить ее от фильтров, необходимо обеспечить выполнение следующих трех условий:

- во время ввода пароля переключение раскладок клавиатуры не разрешается;
- конфигурировать цепочку программных модулей, участвующих в работе с паролем пользователя, может только системный администратор;
- доступ к файлам этих модулей имеет исключительно системный администратор.

Соблюсти первое из этих условий в локализованных для России версиях операционных систем принципиально невозможно. Дело в том, что средства создания учетных пользовательских записей на русском языке являются неотъемлемой частью таких систем. Только в англоязычных версиях систем Windows NT и UNIX предусмотрены возможности, позволяющие поддерживать уровень безопасности, при котором соблюдаются все 3 перечисленные условия.

Заместители

Заместители полностью или частично подменяют собой программные модули операционной системы, отвечающие за аутентификацию пользователей. Подобного рода клавиатурные шпионы могут быть созданы для работы в среде практически любой многопользовательской операционной системы. Трудоемкость написания заместителя определяется сложностью алгоритмов, реализуемых подсистемой аутентификации, и интерфейсов между ее отдельными модулями. Также при оценке трудоемкости следует принимать во внимание степень документированности этой подсистемы. В целом можно сказать, что задача создания заместителя значительно сложнее задачи написания имитатора или фильтра. Поэтому фактов использования подобного рода программных закладок злоумышленниками пока отмечено не было. Однако в связи с тем, что в настоящее время все большее распространение получает операционная система Windows NT, имеющая мощные средства защиты от имитаторов и фильтров, в самом скором будущем от хакеров следует ожидать более активного использования заместителей в целях получения несанкционированного доступа к компьютерным системам.

Поскольку заместители берут на себя выполнение функций подсистемы аутентификации, перед тем как приступить к перехвату пользовательских паролей они должны выполнить следующие действия:

- подобно компьютерному вирусу внедриться в один или несколько системных файлов;
- использовать интерфейсные связи между программными модулями подсистемы аутентификации для встраивания себя в цепочку обработки введенного пользователем пароля.

Для того чтобы защитить систему от внедрения заместителя, ее администраторы должны строго соблюдать адекватную политику безопасности. И что осо-

бенно важно, подсистема аутентификации должна быть одним из самых защищенных элементов операционной системы. Однако, как показывает практика, администраторы, подобно всем людям, склонны к совершению ошибок. А следовательно, соблюдение адекватной политики безопасности в течение неограниченного периода времени является невыполнимой задачей. Кроме того, как только заместитель попал в компьютерную систему, любые меры защиты от внедрения программных закладок перестают быть адекватными, и поэтому необходимо предусмотреть возможность использования эффективных средств обнаружения и удаления внедренных клавиатурных шпионов. Это значит, что администратор должен вести самый тщательный контроль целостности исполняемых системных файлов и интерфейсных функций, используемых подсистемой аутентификации для решения своих задач.

Но и эти меры могут оказаться недостаточно эффективными. Ведь машинный код заместителя выполняется в контексте операционной системы, и поэтому заместитель может предпринимать особые меры, чтобы максимально затруднить собственное обнаружение. Например, он может перехватывать системные вызовы, используемые администратором для выявления программных закладок, с целью подмены возвращаемой ими информации. Или фильтровать сообщения, регистрируемые подсистемой аудита, чтобы отсеивать те, которые свидетельствуют о его присутствии в компьютере.

Как защитить систему от клавиатурных шпионов

Клавиатурные шпионы представляют реальную угрозу безопасности современных компьютерных систем. Чтобы отвести эту угрозу, требуется реализовать целый комплекс административных мер и программно-аппаратных средств защиты. Надежная защита от клавиатурных шпионов может быть построена только тогда, когда операционная система обладает определенными возможностями, затрудняющими работу клавиатурных шпионов. Они были подробно описаны выше, и не имеет смысла снова на них останавливаться. Однако необходимо еще раз отметить, что единственной операционной системой, в которой построение такой защиты возможно, является Windows NT. Да и то с оговорками, поскольку все равно ее придется снабдить дополнительными программными средствами, повышающими степень ее защищенности. В частности, в Windows NT необходимо ввести контроль целостности системных файлов и интерфейсных связей подсистемы аутентификации.

Кроме того, для надежной защиты от клавиатурных шпионов администратор операционной системы должен соблюдать политику безопасности, при которой только администратор может:

- конфигурировать цепочки программных модулей, участвующих в процессе аутентификации пользователей;
- осуществлять доступ к файлам этих программных модулей;
- конфигурировать саму подсистему аутентификации.

И наконец, при организации защиты от клавиатурных шпионов всегда следует иметь в виду, что ни неукоснительное соблюдение адекватной политики безопасности, ни использование операционной системы, имеющей в своем составе средства, существенно затрудняющие внедрение клавиатурных шпионов и облегчающие их своевременное обнаружение, ни дополнительная реализация контроля за целостностью системных файловой и интерфейсных связей сами по себе не могут служить залогом надежной защиты информации в компьютере. Все эти меры должны осуществляться в комплексе. Ведь жертвой клавиатурного шпиона может стать любой пользователь операционной системы, поскольку ее администраторы тоже люди, время от времени и они допускают ошибки в своей работе, а для внедрения клавиатурного шпиона достаточно всего одной оплошности администратора.

Глава 3

Парольная защита операционных систем

Парольные взломщики

Проблему безопасности компьютерных сетей надуманной не назовешь. Практика показывает: чем масштабнее сеть и чем более ценная информация доверяется подключенным к ней компьютерам, тем больше находится желающих нарушить ее нормальное функционирование ради материальной выгоды или просто из праздного любопытства. В самой крупной компьютерной сети в мире (Internet) атаки на компьютерные системы возникают подобно волнам цунами, сметая все защитные барьеры и оставляя после себя впавшие в паралич компьютеры и опустошенные винчестеры. Эти атаки не знают государственных границ. Идет постоянная виртуальная война, в ходе которой организованности системных администраторов противостоит изобретательность компьютерных взломщиков.

Основным защитным рубежом против злонамеренных атак в компьютерной сети является система парольной защиты, которая имеется во всех современных операционных системах. В соответствии с установившейся практикой перед началом сеанса работы с операционной системой пользователь обязан зарегистрироваться, сообщив ей свое имя и пароль. Имя требуется операционной системе для идентификации пользователя, а пароль служит подтверждением правильности произведенной идентификации. Информация, введенная пользователем в диалоговом режиме, сравнивается с той, что имеется в распоряжении операционной системы. Если проверка дает положительный результат, то пользователю становятся доступны все ресурсы операционной системы, связанные с его именем.

Что такое парольный взломщик

Трудно представить, что сегодня какому-нибудь злоумышленнику может придти в голову шальная мысль о том, чтобы попытаться подобрать имя и

пароль для входа в операционную систему, по очереди перебирая в уме все возможные варианты и вводя их с клавиатуры. Скорость такого подбора пароля будет чрезвычайно низкой, тем более что в операционных системах с хорошо продуманной парольной защитой количество подряд идущих повторных вводов конкретного пользовательского имени и соответствующего ему пароля всегда можно ограничить двумя-тремя и сделать так, что если это число будет превышено, то вход в систему с использованием данного имени блокируется в течение фиксированного периода времени или до прихода системного администратора.

Гораздо более эффективным является другой метод взлома парольной защиты операционной системы, при котором атаке подвергается системный файл, содержащий информацию о ее легальных пользователях и их паролях. Однако любая современная операционная система надежно защищает при помощи шифрования пользовательские пароли, которые хранятся в этом файле. Доступ к таким файлам, как правило, по умолчанию запрещен даже для системных администраторов, не говоря уже о рядовых пользователях. Тем не менее, в ряде случаев злоумышленнику удается путем различных ухищрений получить в свое распоряжение файл с именами пользователей и их зашифрованными паролями. И тогда ему на помощь приходят так называемые *парольные взломщики* — специализированные программы, которые служат для взлома паролей операционных систем.

Как работает парольный взломщик

Криптографические алгоритмы, применяемые для шифрования паролей пользователей в современных операционных системах, в подавляющем большинстве случаев являются слишком стойкими, чтобы можно было надеяться отыскать методы их дешифрования, которые окажутся более эффективными, чем тривиальный перебор возможных вариантов. Поэтому парольные взломщики иногда просто шифруют все пароли с использованием того же самого криптографического алгоритма, который применяется для их засекречивания в атакуемой операционной системе, и сравнивают результаты шифрования с тем, что записано в системном файле, где находятся шифрованные пароли ее пользователей. При этом в качестве вариантов паролей парольные взломщики используют символьные последовательности, автоматически генерируемые из некоторого набора символов. Данный способ позволяет взломать все пароли, если известно их представление в зашифрованном виде и они содержат только символы из данного набора. Максимальное время, которое потребуется для взлома пароля, можно вычислить по следующей формуле:

$$T = \frac{1}{S}\sum_{i=1}^{L} N^i ,$$

где N — число символов в наборе, L — предельная длина пароля, S — количество проверок в секунду (зависит от операционной системы и быстродействия компьютера, на котором производится взлом ее парольной защиты).

Из приведенной формулы видно, что за счет очень большого числа перебираемых комбинаций, которое растет экспоненциально с увеличением числа символов в исходном наборе, такие атаки парольной защиты операционной системы могут занимать слишком много времени. Однако хорошо известно, что большинство пользователей операционных систем не затрудняют себя выбором стойких паролей (т. е. таких, которые трудно взломать). Поэтому для более эффективного подбора паролей парольные взломщики обычно используют так называемые словари, представляющие собой заранее сформированный список слов, наиболее часто применяемых на практике в качестве паролей.[1]

Для каждого слова из словаря парольный взломщик использует одно или несколько правил. В соответствии с этими правилами слово изменяется и порождает дополнительное множество опробуемых паролей. Производится попеременное изменение буквенного регистра, в котором набрано слово, порядок следования букв в слове меняется на обратный, в начало и в конец каждого слова приписывается цифра 1, некоторые буквы заменяются на близкие по начертанию цифры (в результате, например, из слова password получается pa55w0rd). Это повышает вероятность подбора пароля, поскольку в современных операционных системах, как правило, различаются пароли, набранные заглавными и строчными буквами, а пользователям этих систем настоятельно рекомендуется выбирать пароли, в которых буквы чередуются с цифрами.

Одни парольные взломщики поочередно проверяют каждое слово из словаря, применяя к нему определенный набор правил для генерации дополнительного множества опробуемых паролей. Другие предварительно обрабатывают весь словарь при помощи этих же правил, получая новый словарь большего размера, и затем из него черпают проверяемые пароли. Учитывая, что обычные словари человеческих языков состоят всего из нескольких сотен тысяч слов, а скорость шифрования паролей достаточно высока, парольные взломщики, осуществляющие поиск с использованием словаря, работают достаточно быстро.

Пользовательский интерфейс подавляющего большинства парольных взломщиков трудно назвать дружественным. После их запуска на экране дисплея, как правило, появляется лаконичный запрос File?, означающий, что необходимо ввести имя файла, где хранится словарь. Да и документацию

---

[1] В Internet существует несколько депозитариев словарей для парольных взломщиков. Один из наиболее популярных таких депозитариев можно найти по адресу **http://sdg.ncsa.uiuc.edu/~mag/Misc/Wordlists.html**, а самый авторитетный и полный — по адресу **ftp://coast.cs.purdue.edu/pub/dict/wordlists**.

к парольным взломщикам обильной не назовешь. Правда, для этого есть свои объективные причины.

Во-первых, парольные взломщики предназначены исключительно для подбора паролей. Такая узкая специализация не способствует разнообразию их интерфейса и обилию сопроводительной документации.

Во-вторых, авторами большей части парольных взломщиков являются люди компьютерного подполья, которые создают их "на лету" для удовлетворения собственных сиюминутных потребностей, и поэтому редко снабжают их подробной документацией и встроенными справочными системами. Приятное исключение из этого правила составляют только парольные взломщики, созданные специалистами в области компьютерной безопасности для выявления слабостей в парольной защите операционных систем. В этом случае дистрибутив парольного взломщика, помимо самой программы, обязательно включает разнообразные дополнительные сведения, касающиеся технических сторон ее эксплуатации, а также небольшой словарь.

Взлом парольной защиты операционной системы UNIX

В операционной системе UNIX информацию о пароле любого пользователя можно отыскать в файле passwd, находящемся в каталоге etc. Эта информация хранится в зашифрованном виде и располагается через двоеточие сразу после имени соответствующего пользователя. Например, запись, сделанная в файле passwd относительно пользователя с именем bill, будет выглядеть примерно так:

bill:5fg63fhD3d5g:9406:12:Bill Spencer:/home/fsg/will:/bin/bash

Здесь 5fg63fhD3d5g — это и есть информация о пароле пользователя bill.

При первоначальном задании или изменении пользовательского пароля операционная система UNIX генерирует два случайных байта (в приведенном выше примере 5 и f), к которым добавляются байты пароля. Полученная в результате байтовая строка шифруется при помощи специальной криптографической процедуры Crypt[2] (в качестве ключа используется пароль пользователя) и в зашифрованном виде (g63fhD3d5g) вместе с двумя случайными байтами (5f) записывается в файл /etc/passwd после имени пользователя и двоеточия.

Если злоумышленник имеет доступ к парольному файлу операционной системы UNIX, то он может скопировать этот файл на свой компьютер и затем воспользоваться одной из программ для взлома парольной защиты UNIX.

---

[2] Ее исходный текст на языке программирования "Си" можно найти в Internet по адресу **ftp://ftp.ifi.uio.no/pub/gnu/glibc-crypt-2.0.6.tar.gz**.

Самой эффективной и популярной такой программой является Crack. И хотя она предназначена для запуска на компьютерах, работающих только под управлением операционных систем семейства UNIX, инициируемый ею процесс поиска паролей может быть без особых усилий распределен между различными платформами, подключенными к единой компьютерной сети. Среди них могут оказаться и IBM-совместимые персональные компьютеры с операционной системой Linux, и рабочие станции RS/6000 с AIX, и Macintosh с A/UX.

CrackJack — еще одна известная программа для взлома паролей операционной системы UNIX. К сожалению, работает она только под управлением операционной системы DOS, но зато весьма непритязательна в том, что касается компьютерных ресурсов. К другим недостаткам этого парольного взломщика можно отнести запрет на одновременное использование сразу нескольких словарей и принципиальную невозможность запуска CrackJack под Windows 95/98.

В отличие от CrackJack, парольный взломщик PaceCrack95 работает под Windows 95/98 в качестве полноценного DOS-приложения. К тому же он достаточно быстр, компактен и эффективен (рис. 3.1).

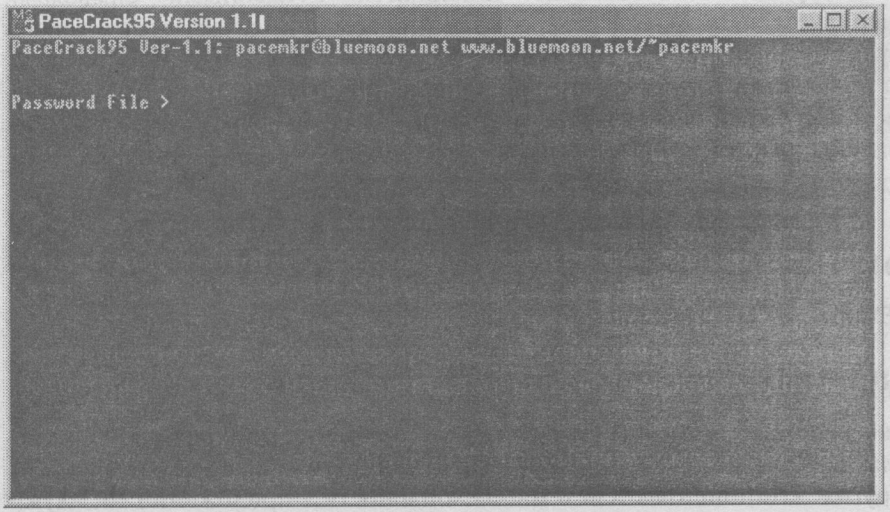


Рис. 3.1. Основное рабочее окно парольного взломщика PaceCrack95

Парольные взломщики Q-Crack и John the Ripper примечательны тем, что существуют версии этих взломщиков, предназначенные для работы не только на DOS/Windows-платформах, но и на компьютерах с операционной системой Linux.

А парольный взломщик Hades лучше остальных документирован и содержит ряд очень полезных утилит, которые позволяют осуществлять слияние не-

скольких словарей в один большой словарь, удалять из словаря повторяющиеся слова и добавлять в уже имеющийся словарь пароли, взломанные в процессе работы Hades.

Существует множество других программ взлома паролей операционной системы UNIX. Они устойчивы к сбоям электропитания компьютеров, на которых работают (XiT), позволяют планировать время своей работы (Star Cracker), при выполнении монополизируют процессор для достижения максимальной производительности (Killer Cracker), не только взламывают пароли операционной системы UNIX, но и помогают преодолеть парольную защиту других программ, которые требуют, чтобы пользователь зарегистрировался путем ввода своего имени и соответствующего пароля (Claymore).[3]

Что же касается защиты от взлома паролей операционной системы UNIX, то ее пользователям следует настоятельно порекомендовать применять только стойкие пароли, а в качестве критерия стойкости пароля использовать его отсутствие в словарях, предназначенных для парольных взломщиков. Да и сами файлы с информацией о пользовательских паролях следует прятать подальше от посторонних любопытных глаз. Достигается это обычно с помощью так называемого *затенения* (shadowing), когда в файле passwd шифрованные пароли пользователей заменяются служебными символами (например, звездочками), а вся парольная информация скрывается в какомнибудь укромном месте. И хотя существуют программы, специально созданные для отыскания спрятанных парольных файлов операционной системы UNIX, к счастью для системных администраторов эти программы далеко не универсальны и успешно срабатывают не для всех операционных систем семейства UNIX.

Взлом парольной защиты операционной системы Windows NT

База данных учетных записей пользователей

Одним из основных компонентов системы безопасности Windows NT является *диспетчер учетных записей* пользователей. Он обеспечивает взаимодействие других компонентов системы безопасности, приложений и служб Windows NT с базой данных учетных записей пользователей (Security Account Management Database, SAM). Эта база обязательно имеется на каждом компьютере с операционной системой Windows NT. В ней хранится вся ин-

[3] Все перечисленные программы для взлома паролей операционной системы UNIX можно найти в Internet по адресу **http://www.rat.pp.se/hotel/panik/password_cracking.html** (кстати, там же находится хороший депозитарий словарей) или по адресу **http://www.outpost9.com/files/crackers.html**.

формация, используемая для аутентификации пользователей Windows NT при интерактивном входе в систему и при удаленном доступе к ней по компьютерной сети.

База данных SAM представляет собой один из кустов (hive) системного реестра (registry) Windows NT. Этот куст принадлежит ветви (subtree) HKEY_LOCAL_MACHINE и называется SAM. Он располагается в каталоге \winnt_root\System32\Config (winnt_root — условное обозначение каталога с системными файлами Windows NT) в отдельном файле, который тоже называется SAM.

Информация в базе данных SAM хранится в основном в двоичном виде. Доступ к ней обычно осуществляется через диспетчер учетных записей. Изменять записи, находящиеся в базе данных SAM, при помощи программ, позволяющих напрямую редактировать реестр Windows NT (REGEDT или REGEDT32), не рекомендуется. По умолчанию этого и нельзя делать, т. к. доступ к базе данных SAM запрещен для всех без исключения категорий пользователей операционной системы Windows NT.

Хранение паролей пользователей

Именно в учетных записях базы данных SAM находится информация о пользовательских именах и паролях, которая необходима для идентификации и аутентификации пользователей при их интерактивном входе в систему. Как и в любой другой современной многопользовательской операционной системе, эта информация хранится в зашифрованном виде. В базе данных SAM каждый пароль пользователя обычно бывает представлен в виде двух 16-байтовых последовательностей, полученных разными методами.

При использовании первого метода строка символов пользовательского пароля хэшируется с помощью функции MD4. В итоге из символьного пароля, введенного пользователем[4], получается 16-байтовая последовательность — хэшированный пароль Windows NT. Данная последовательность затем шифруется по DES-алгоритму[5], и результат шифрования сохраняется в базе данных SAM. При этом в качестве ключа используется так называемый *относительный идентификатор пользователя* (Relative Identifier, RID), который

[4] Длина пароля в Windows NT ограничена 14 символами. Это ограничение накладывается диспетчером учетных записей, который не позволяет вводить пароли длиной более 14 символов.

[5] DES-алгоритм является одним из самых распространенных алгоритмов шифрования данных. В США он имеет статус федерального стандарта. Это блочный алгоритм шифрования с симметричным ключом длиной 64 бита, из которых только 56 непосредственно используются при шифровании, а остальные 8 предназначены для контроля четности байтов ключа.

представляет собой автоматически увеличивающийся порядковый номер учетной записи данного пользователя в базе данных SAM.

Для совместимости с другим программным обеспечением корпорации Microsoft (Windows for Workgroups, Windows 95/98 и Lan Manager) в базе данных SAM хранится также информация о пароле пользователя в стандарте Lan Manager. Для ее формирования используется второй метод. Все буквенные символы исходной строки пользовательского пароля приводятся к верхнему регистру, и, если пароль содержит меньше 14 символов, то он дополняется нулями. Из каждой 7-байтовой половины преобразованного таким образом пароля пользователя отдельно формируется ключ для шифрования фиксированной 8-байтовой последовательности по DES-алгоритму. Полученные в результате две 8-байтовые половины хэшированного пароля Lan Manager еще раз шифруются по DES-алгоритму (при этом в качестве ключа используется RID пользователя) и помещаются в базу данных SAM.

Использование пароля

Информация о паролях, занесенная в базу данных SAM, служит для аутентификации пользователей Windows NT. При интерактивном или сетевом входе в систему введенный пользователем пароль сначала хэшируется и шифруется, а затем сравнивается с 16-байтовой последовательностью, записанной в базе данных SAM. Если они совпадают, пользователю разрешается вход в систему.

Обычно в базе данных SAM хранятся в зашифрованном виде оба хэшированных пароля. Однако в некоторых случаях операционная система вычисляет только один из них. Например, если пользователь домена Windows NT изменит свой пароль, работая на компьютере с Windows for Workgroups, то в его учетной записи останется только пароль Lan Manager. А если пользовательский пароль содержит более 14 символов или если эти символы не входят в так называемый *набор поставщика оборудования* (original equipment manufacturer, OEM), то в базу данных SAM будет занесен только пароль Windows NT.

Возможные атаки на базу данных SAM

Обычно основным предметом вожделения, испытываемого взломщиком парольной защиты операционной системы, являются административные полномочия. Их можно получить, узнав в хэшированном или символьном виде пароль администратора системы, который хранится в базе данных SAM. Поэтому именно на базу данных SAM бывает направлен главный удар взломщика парольной защиты Windows NT.

По умолчанию в операционной системе Windows NT доступ к файлу \winnt_root\System32\Config\SAM заблокирован для всех без исключения ее

пользователей. Тем не менее, с помощью программы NTBACKUP любой обладатель права на резервное копирование файлов и каталогов Windows NT может перенести этот файл с жесткого диска на магнитную ленту. Резервную копию реестра также можно создать утилитой REGBAK из Windows NT Resource Kit. Кроме того, несомненный интерес для любого взломщика представляют резервная копия файла SAM (SAM.SAV) в каталоге \winnt_root\System32\Config и сжатая архивная копия SAM (файл SAM._) в каталоге \winnt_root\Repair.

При наличии физической копии файла SAM извлечь хранимую в нем информацию не представляет никакого труда. Загрузив файл SAM в реестр любого другого компьютера с Windows NT (например, с помощью команды Load Hive программы REGEDT32), можно в деталях изучить учетные записи пользователей, чтобы определить значения RID пользователей и шифрованные варианты их хэшированных паролей. Зная RID пользователя и имея зашифрованную версию его хэшированного пароля, компьютерный взломщик может попытаться расшифровать этот пароль, чтобы использовать его, например, для получения сетевого доступа к другому компьютеру. Однако для интерактивного входа в систему одного лишь знания хэшированного пароля недостаточно. Необходимо получить его символьное представление.

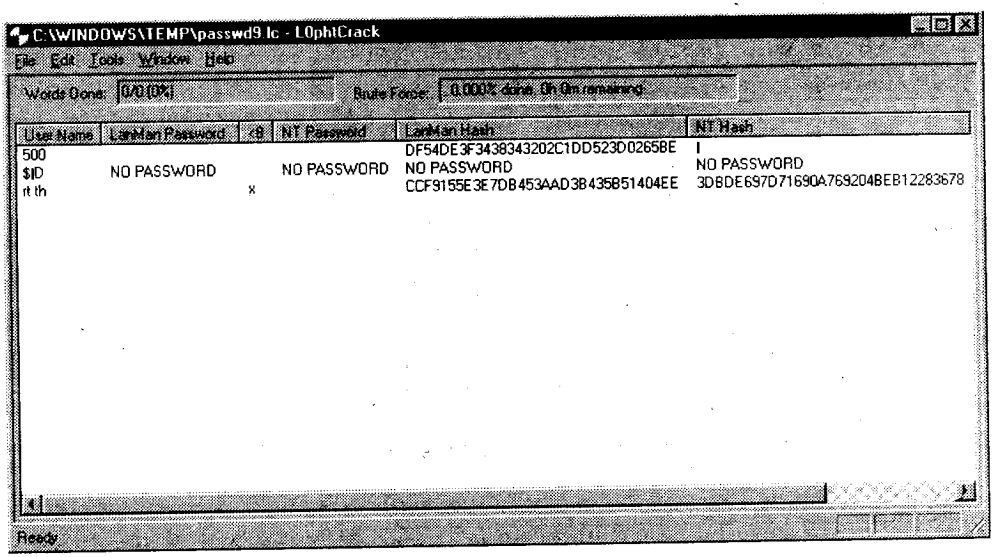


Рис. 3.2. Основное рабочее окно парольного взломщика LOphtCrack

Для восстановления пользовательских паролей операционной системы Windows NT в символьном виде существуют специальные парольные взломщики, которые выполняют как прямой подбор паролей, так и поиск по словарю, а также используют комбинированный метод взлома парольной защиты,

когда в качестве словаря задействуется файл с заранее вычисленными хэшированными паролями, соответствующими символьным последовательностям, которые часто применяются в качестве паролей пользователей операционных систем. Одной из самых известных программ взлома паролей операционной системы Windows NT является LOphtCrack (**http://www.l0pht.com/l0phtcrack/**). На рис. 3.2 приведено основное рабочее окно этой программы, предназначенной для выполнения на компьютерах с операционной системой Windows 95/98/NT.

Другим распространенным парольным взломщиком Windows NT является Advanced NT Security Explorer (сокращенно — ANTExp). Его можно найти в Internet по адресу **http://www.elcomsoft.com/antexp.html**. ANTExp имеет удобный пользовательский интерфейс. Пользователь может задать набор символов, из которых будут формироваться последовательности, используемые в качестве вариантов паролей, а также верхнюю и нижнюю границу длины перебираемых паролей. Кроме того, можно выбрать тип атаки на парольную защиту Windows NT и применить либо атаку методом "грубой силы", либо словарную атаку (рис. 3.3).

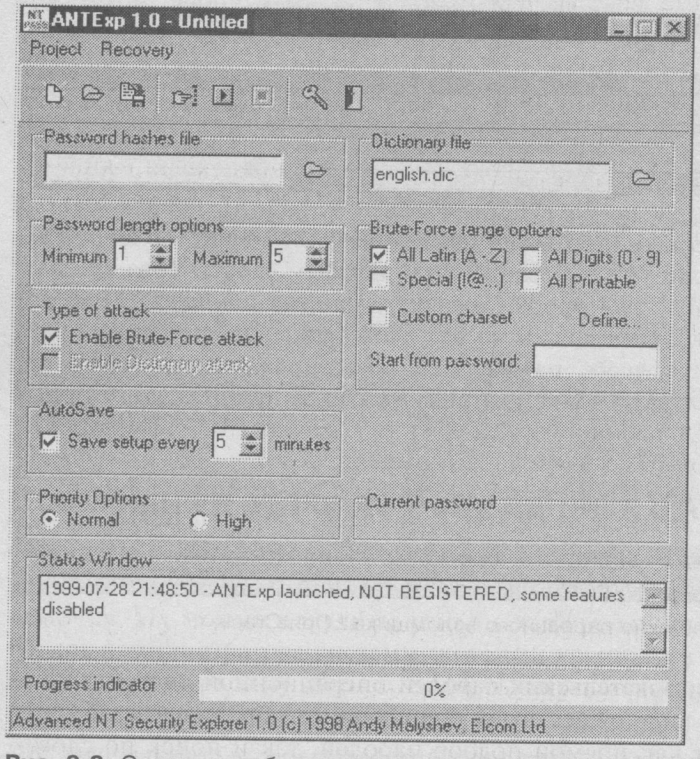


Рис. 3.3. Основное рабочее окно парольного взломщика ANTExp

Защита системы от парольных взломщиков

Итак, вывод однозначен: одна из главных задач системного администратора Windows NT состоит в защите от несанкционированного доступа той информации, которая хранится в базе данных SAM. С этой целью ему, прежде всего, необходимо ограничить физический доступ к компьютерам сети и прежде всего — к контроллерам доменов. Дополнительно, при наличии соответствующих программно-аппаратных средств, следует установить пароли BIOS на включение компьютеров и на изменение настроек BIOS. Затем, используя настройки BIOS, рекомендуется отключить загрузку компьютеров с гибких и компакт-дисков. А для обеспечения контроля доступа к файлам и папкам операционной системы Windows NT системный раздел жесткого диска должен иметь формат NTFS.

Каталог \winnt_root\repair нужно средствами операционной системы закрыть для доступа всех пользователей, включая администраторов, и разрешать к ней доступ только во время работы утилиты RDISK, создающей в этом каталоге архивные копии системного реестра Windows NT. Системные администраторы также должны внимательно следить за тем, где и как хранятся дискеты аварийного восстановления (Emergency Repair Disks) и архивные копии на магнитных лентах, если на последних присутствует дубликат системного реестра Windows NT.

Если компьютер с операционной системой Windows NT входит в домен, то по умолчанию имена и хэшированные пароли последних 10-ти пользователей, регистрировавшихся на этом компьютере, сохраняются (кэшируются) в его локальном системном реестре (в разделе SECURITY\Policy\Secrets куста HKEY_LOCAL_MACHINE). Чтобы отменить кэширование паролей на компьютерах домена, нужно с помощью утилиты REGEDT32 в раздел Microsoft\WindowsNT\CurrentVersion\Winlogon куста HKEY_LOCAL_MACHINE добавить параметр CashedLogonsCount, установив его значение равным нулю, а тип — REG_SZ.

Для защиты базы данных SAM можно применить утилиту SYSKEY, входящую в состав пакета обновления Windows NT Service Pack 3 (рис. 3.4). Эта утилита позволяет включить режим дополнительного шифрования информации о паролях, которая хранится в базе данных SAM. Уникальный 128-битовый ключ для дополнительного шифрования паролей (так называемый *ключ шифрования паролей* — Password Encryption Key, PEK) автоматически сохраняется в системном реестре для дальнейшего использования.

Перед помещением в системный реестр ключ PEK шифруется при помощи другого 128-битового ключа, который называется системным ключом (System Key), и может храниться либо в системном реестре, либо в файле с именем STARTUP.KEY в корневом каталоге на отдельной дискете. Можно не сохранять системный ключ на магнитном носителе, и тогда каждый раз при запуске операционной системы он будет вычисляться с помощью алго-

ритма MD5 на основе пароля, набираемого на клавиатуре в диалоговом окне утилиты SYSKEY. Последние два способа хранения системного ключа обеспечивают максимальную защиту паролей в базе данных SAM, однако приводят к невозможности автоматической перезагрузки операционной системы, поскольку для завершения процесса перезагрузки потребуется либо вставить дискету с системным ключом и подтвердить ее наличие в дисководе путем нажатия кнопки **OK** в появившемся диалоговом окне, либо вручную ввести системный ключ с клавиатуры.

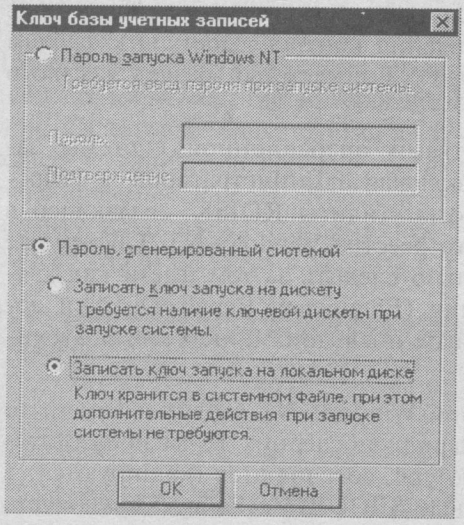


Рис. 3.4. Диалоговое окно утилиты SYSKEY

Для повышения стойкости паролей пользователей операционной системы Windows NT ко взлому рекомендуется с помощью утилиты Диспетчер пользователей (User Manager) задать минимальную длину пользовательских паролей равной не менее 8 символов и включить режим устаревания паролей, чтобы пользователи периодически их обновляли (рис. 3.5). При этом чем выше вероятность атак на парольную защиту Windows NT, тем короче должен быть срок такого устаревания. А чтобы пользователи не вводили свои старые пароли повторно, необходимо включить режим хранения некоторого числа ранее использовавшихся паролей.

Утилита PASSPROP из состава Windows NT Resource Kit, запущенная с ключом /COMPLEX, заставляет пользователей вводить более устойчивые пароли, которые или сочетают буквы в разном регистре, или буквы с цифрами, или буквы со специальными символами. Более строгие правила фильтрации нестойких паролей можно задать после установки любого из пакетов обновления Windows NT, начиная с Service Pack 2. Тогда специальная библиотека PASSFILT.DLL, находящаяся в каталоге \winnt_root\System32, будет следить за тем, чтобы каждый пользовательский

пароль состоял не менее чем из 5 символов, не содержал имени пользователя, включал символы, по крайней мере, трех наборов из четырех возможных, составленных из прописных букв, строчных букв, цифр и специальных символов (знаков препинания и т. д.) соответственно. Чтобы задать такой режим проверки паролей пользователей, необходимо в раздел HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa системного реестра с помощью программы REGEDT32 добавить параметр Notification Packages типа REG_MULTI_SZ и вписать в него строку PASSFILT. Если этот параметр уже имеется, то новую строку следует дописать после уже существующей.

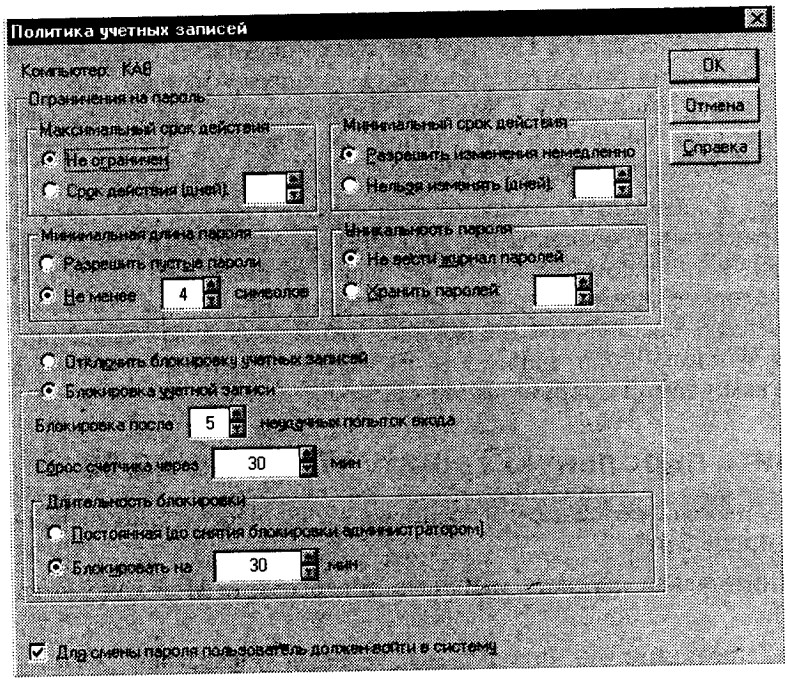


Рис. 3.5. Диалоговое окно, предназначенное для задания правил работы с паролями из Диспетчера пользователей

В заключение необходимо отметить, что хотя в умелых руках злоумышленника программы взлома паролей операционных систем представляют огромную опасность для их парольной защиты, сами парольные взломщики все же являются не менее ценным инструментом для системных администраторов, которые заинтересованы в выявлении слабых мест в парольной защите своих операционных систем. Основная проблема состоит не в том, что на свете существуют парольные взломщики, а в том, что ими недостаточно часто пользуются системные администраторы. Надеюсь, что после выхода в свет этой книги положение дел изменится к лучшему.

Как сделать парольную защиту Windows 95/98 более надежной

В настоящее время Windows корпорации Microsoft является наиболее распространенной операционной системой. Подавляющее большинство компьютеров, которые повсеместно используются для решения самых разнообразных задач, функционируют именно под ее управлением.

Однако, признавая отменные потребительские качества операционных систем корпорации Microsoft, нельзя обойти вниманием имеющиеся в них значительные изъяны. В первую очередь, это касается парольной защиты Windows 95 и Windows 98. Фактически парольная защита в Windows 98 осталась на прежнем уровне по сравнению с Windows 95. Поэтому все сказанное в ходе дальнейшего изложения в равной степени относится к ним обеим. А следовательно, рассматривая парольную защиту Windows 95 и Windows 98, можно вести речь не о двух отдельных операционных системах, а о единой, обобщенной операционной системе, которую будем условно называть Windows 95/98.

Далеко не все пользователи знают, что для парольной защиты Windows 95/98 характерны существенные недостатки, связанные с тем, что при ее разработке корпорация Microsoft не уделила должного внимания проблемам безопасности компьютеров, для которых предназначалась эта операционная система. Рассмотрим подробнее, что представляет собой парольная защита в Windows 95/98, какие у нее имеются изъяны и как от них избавиться средствами самой операционной системы.

Как установить парольную защиту Windows 95/98

Чтобы установить парольную защиту в Windows 95/98, необходимо выполнить следующую процедуру[6]:

1. Дважды щелкните левой кнопкой мыши на пиктограмме **Мой компьютер** (My Computer).
2. Теперь дважды щелкните на пиктограмме **Панель управления** (Control Panel). Если вы не можете отыскать пиктограмму **Мой компьютер** (My Computer), щелкните на кнопке **Пуск** (Start), выберите раздел **Настройка** (Settings), затем **Панель управления** (Control Panel) и перейдите к пункту 3.
3. Найдите пиктограмму, которая называется **Пароли** (Passwords), и дважды щелкните на ней. После двойного щелчка на пиктограмме **Пароли** (Passwords) Windows может отобразить диалоговое окно, в котором вам будет задан вопрос о том, хотите ли вы одновременно с этим изменить пароль экранной заставки Windows 95/98. Это окно появится только в

[6] Предполагается, что при инсталляции Windows 95/98 пользователь принял решение не применять пароли, но впоследствии обстоятельства коренным образом изменились, и он понял, что без парольной защиты ему никак не обойтись.

том случае, если вы уже используете опцию экранной заставки. Суть вопроса в том, чтобы предоставить вам возможность использовать одинаковый пароль и для открытия сеанса работы с Windows 95/98, и для экранной заставки.

4. Щелкните на опции **Сменить пароль Windows** (Change Windows Passwords).
5. Введите новый пароль в поле **Новый пароль** (New Password) диалогового окна **Изменение пароля Windows** (Change Windows Passwords) (рис. 3.6). Не обращайте внимания на поле **Старый пароль** (Old Password), оно должно остаться незаполненным. Однако если вы хотите изменить существующий пароль, то сначала введите его, а затем новый пароль. Операционная система Windows 95/98 внесет соответствующие изменения.
6. В поле **Подтверждение пароля** (Confirm New Password) введите пароль, указанный вами в пункте 5. Затем щелкните на кнопке **OK**.[7] Windows 95/98 выведет сообщение о том, что смена пароля прошла успешно.

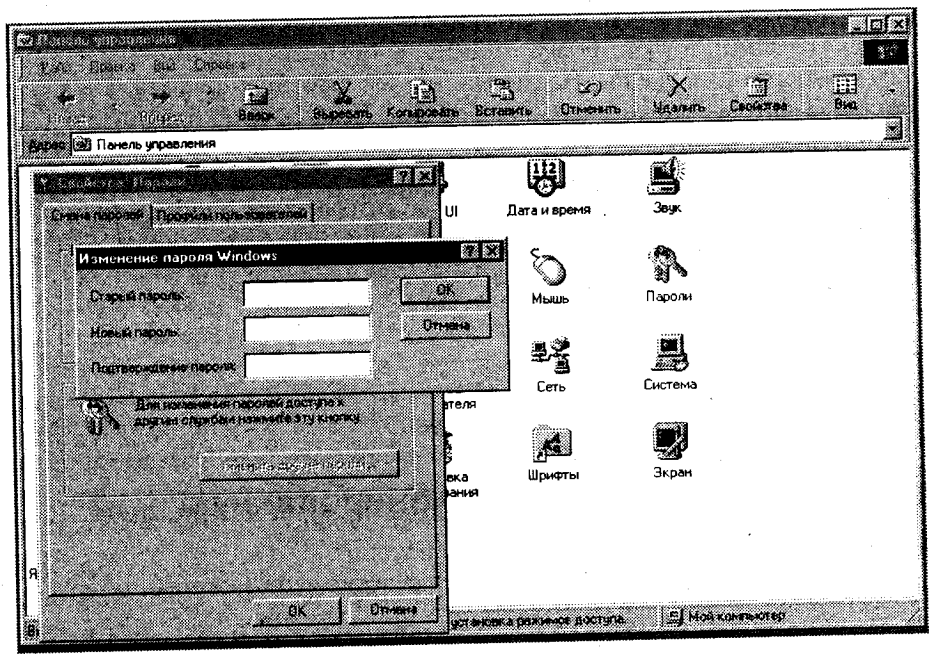


Рис. 3.6. Диалоговое окно **Изменение пароля Windows**

[7] Все надежные программы парольной защиты требуют, чтобы при смене пароля пользователь вводил его дважды. Тем самым пользователь подтверждает, что в пароль не вкралась случайная "опечатка", которая не позволит открыть сеанс работы с операционной системой при повторном включении компьютера. На заре компьютерной эры эта проблема, по-видимому, возникала так часто, что, в конце концов, кому-то пришла в голову идея требовать от пользователей вводить пароль дважды. Данный метод действует и поныне.

7. Щелкните на кнопке **OK** для закрытия диалогового окна **Свойства: Пароли** (Passwords Properties).

Почему парольная защита Windows 95/98 ненадежна

После того как была установлена парольная защита, каждый раз, когда включается компьютер, Windows 95/98 станет вежливо осведомляться о том, каковы ваши регистрационное имя и пароль. Продолжение нормальной работы с Windows 95/98 будет возможно только в том случае, если регистрационное имя и соответствующий ему пароль были введены без ошибок. Означает ли это, что система парольной защиты Windows 95/98 гарантирует: злоумышленник не сможет знакомиться с содержимым всех ваших файлов и запускать любые программы на вашем компьютере?

Отнюдь. Почему? Да по той простой причине, что в основе функционирования Windows 95/98 лежат принципы, которые были применены на практике при создании другой, более примитивной операционной системы корпорации Microsoft — DOS. Несмотря на то, что Windows 95/98 старается "упрятать" DOS как можно дальше, она по-прежнему использует эту безнадежно устаревшую операционную систему для обеспечения работоспособности старых программ, которые остались у пользователей, сменивших DOS на Windows 95/98. А ведь хорошо известно, что в DOS напрочь отсутствуют защитные механизмы, которые предотвращают несанкционированный доступ к файлам и программам компьютера, работающего под ее управлением. Ведь в сущности DOS и задумывалась-то именно для выполнения совершенно противоположной задачи — предоставить всем возможность обращаться к любым файлам. Как следствие, существует несколько способов обойти парольную защиту Windows 95/98, загружая на компьютере операционную систему DOS вместо Windows 95/98.

Как предотвратить несанкционированную загрузку системы

В ходе инсталляции операционной системы Windows 95/98 корпорация Microsoft настойчиво рекомендует ее пользователям сразу же создать так называемый *загрузочный диск*. В этой рекомендации есть свой резон: если возникнут какие-либо проблемы при переходе на Windows 95/98, вы сможете воспользоваться загрузочным диском, чтобы внести необходимые поправки.

Если вы не создали загрузочную дискету во время инсталляции операционной системы, вы можете сделать это в любой момент. Выберите опцию **Установка и удаление программ** (Add/Remove Programs) на **Панели управления** (Control Panel) Windows 95/98, а затем щелкните на переключателе **Загрузочный диск** (Startup Disk). После этого нажмите кнопку **Создать диск** (Create

Disk) и вставьте гибкий диск в дисковод. Чтобы создать загрузочный диск Windows 95/98, вам потребуется всего одна дискета. Диалоговые окна, с которыми пользователь имеет дело при создании загрузочного диска, показаны на рис. 3.7.

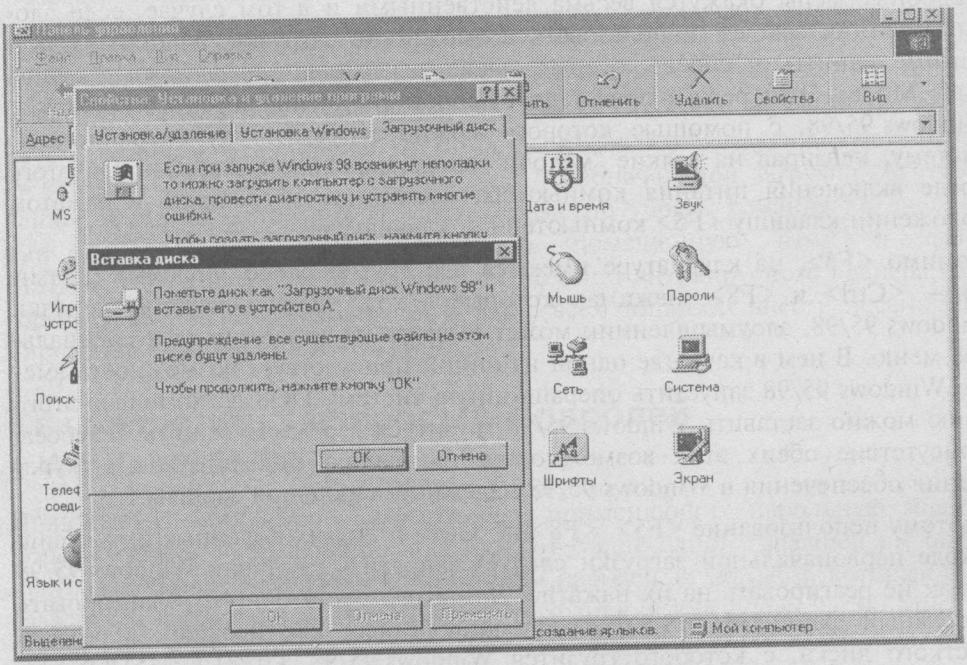


Рис. 3.7. Создание загрузочной дискеты Windows 95/98

Таким образом, для создания загрузочного диска не требуется каких-либо особенных познаний. Зато с его помощью можно без всяких хлопот обойти систему парольной защиты Windows 95/98. Эта система активируется только при загрузке Windows 95/98. При использовании же загрузочного диска происходит запуск не Windows 95/98, а ее подсистемы, функционально эквивалентной операционной системе DOS и не имеющей никаких средств обеспечения безопасной работы с компьютером (наподобие парольной защиты). Затем путем нехитрых манипуляций злоумышленник может не только установить собственный пароль для последующего открытия сеанса работы с Windows 95/98, но и сделать так, чтобы все остальные ее пользователи ничего не заметили и продолжали регистрироваться в Windows 95/98 посредством своих законных паролей.

Чтобы предотвратить несанкционированный доступ к компьютеру через загрузочный диск, необходимо применять дополнительные физические меры защиты (например, закрывать на замок дверь, ведущую в помещение, где находится компьютер), блокировать клавиатуру или кнопку включения пи-

тания компьютера при помощи ключа, использовать установки BIOS компьютера для предотвращения неконтролируемой загрузки операционной системы с гибкого диска, а также для задания пароля включения питания и изменения установок BIOS.

Подобные меры окажутся весьма действенными и в том случае, если злоумышленник захочет использовать возможность загрузки Windows 95/98 в режиме защиты от сбоев, который также называется *защищенным режимом* (Safe Mode). Этот режим представляет собой специальный способ работы с Windows 95/98, с помощью которого можно запустить эту операционную систему, невзирая на всякие "мелочи" вроде парольной защиты. Для этого после включения питания компьютера достаточно удерживать в нажатом положении клавишу <F5> компьютерной клавиатуры.

Помимо <F5>, на клавиатуре имеются две другие особо "опасные" клавиши — <Ctrl> и <F8>, нажимая которые в ходе первоначальной загрузки Windows 95/98, злоумышленник может вызвать на экран дисплея специальное меню. В нем в качестве одной из опций присутствует возможность вместо Windows 95/98 запустить операционную систему DOS. С помощью этого меню можно заставить Windows 95/98 грузиться в режиме защиты от сбоев. Присутствие обеих этих возможностей одинаково нежелательно с точки зрения обеспечения в Windows 95/98 надежной парольной защиты.

Поэтому использование <F5>, <F8> и <Ctrl> и других подобных им клавиш в ходе первоначальной загрузки следует запретить, заставив Windows 95/98 никак не реагировать на их нажатие. Для этого достаточно отредактировать системный файл MSDOS.SYS, располагающийся, как правило, в разделе жесткого диска, с которого грузится Windows 95/98. Открыв MSDOS.SYS при помощи обычного редактора, вы увидите текст примерно следующего содержания:

```
[Paths]
WinDir=C:\WINDOWS
WinBootDir=C:\WINDOWS
HostWinBootDrv=C

[Options]
BootGUI=1
;
;The following lines are required for compatibility with other programs.
;Do not remove them (MSDOS.SYS needs to be >1024 bytes).
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxa
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxb
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxc
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxd
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxe
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxf
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxg
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxh
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxi
```

```
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxj
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxk
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxl
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxm
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxo
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxp
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxq
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxr
;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxs
```

Добавьте в раздел `[Options]` файла MSDOS.SYS одну из двух команд: `BootKeys=0` или `BootDelay=0`. Обе эти команды позволяют дезактивировать работу клавиш <F5>, <F8> и <Ctrl> в ходе первоначальной загрузки. А чтобы слишком умный взломщик не смог незаметно убрать команду `BootKeys=0` или `BootDelay=0`, предусмотрительно помещенную вами в файл MSDOS.SYS, следует защитить его при помощи антивирусного сканера, который своевременно предупредит вас обо всех попытках внести несанкционированные изменения в этот файл.

Как запретить кэширование паролей в Windows 95/98

Пользователи миллионов компьютеров, применяющих парольную защиту Windows 95/98, часто даже не подозревают о еще одной грозящей им опасности. Проблема связана с кэшированием паролей — методом, который был разработан корпорацией Microsoft для их хранения в Windows 95/98. Многие выбирают кэширование паролей, даже не догадываясь об этом, поскольку кэширование в Windows 95/98 разрешено по умолчанию. В этом случае пользовательский пароль помещается операционной системой в отдельный файл на магнитном диске. Например, если ваше имя — Вадим, и оно также является идентификатором для входа в систему, то ваш пароль будет сохранен в файле C:\WINDOWS\ВАДИМ.PWL.

При кэшировании пароль записывается в PWL-файл в зашифрованном виде. В корпорации Microsoft утверждают, что применяемый метод шифрования является лучшим среди разрешенных правительством США для экспорта за пределы страны. Суть возражений оппонентов Microsoft состоит в том, что реализация этого метода в Windows 95/98 далеко не безупречна, и в результате его стойкость совершенно не отвечает современным требованиям. Программы, предназначенные для дешифрования парольных файлов, можно легко отыскать в глобальной сети Internet (например, по адресу **http://www.c2.org/hacksoft/**), что свидетельствует об уязвимости метода шифрования паролей в Windows 95/98.

Существует 2 пути решения этой проблемы. Во-первых, можно получить от Microsoft "заплату", которая позволяет использовать в Windows 95/98 более

совершенный метод шифрования паролей. Такая "заплата" доступна пользователям глобальных компьютерных сетей, в которых присутствует корпорация Microsoft.

Во-вторых, кэширование паролей можно отключить. Правда, чтобы сделать это, придется немного повозиться. Для начала потребуется установить программу, которая называется Редактор системных правил (System Policy Editor). Эта программа входит в комплект Windows 95/98 Resource Kit (Набор ресурсов Windows 95/98), который включен в поставку Windows 95/98 на компакт-диске. Чтобы ее установить, надо выполнить следующее:

1. Используя пиктограмму **Установка и удаление программ** (Add/Remove Programs) в Панели управления (Control Panel), выберите вкладку **Установка Windows** (Windows Settings).
2. Нажмите на кнопку **Установить с диска** (Have Disk) и в появившемся диалоговом окне укажите каталог E:\ADMIN\APPTOOLS\POLEDIT для Windows 95 или E:\TOOLS\RESKIT\NETADMIN\POLEDIT для Windows 98 (если ваш CD-ROM установлен как диск E).
3. В диалоговом окне, появившемся после нажатия кнопки **Установить с диска** (Have Disk), выберите программу Редактор системных правил (System Policy Editor), а затем нажмите кнопку **Установить** (Install).
4. Нажмите кнопку **OK**, чтобы закрыть диалоговое окно **Установка и удаление программ** (Add/Remove Programs).

Процесс установки редактора системных правил проиллюстрирован на рис. 3.8.

После установки для запуска редактора системных правил следует сначала выбрать пункт **Программы** (Programs) меню **Пуск** (Start), затем **Стандартные** (Accessories), далее **Служебные** (System Tools) и наконец — **Редактор системных правил** (System Policy Editor). Немного погодя перед вами появится диалоговое окно этой программы.

Выполнив команду **Создать** (New File) меню **Файл** (File), вы увидите пиктограммы **Стандартный компьютер** (Default Computer) и **Стандартный пользователь** (Default User). Щелкнув дважды сначала на пиктограмме **Стандартный компьютер** (Default Computer), а затем — на кнопках **Сеть** (Network) и **Пароли** (Passwords), можно пометить галочкой строку **Отключить кэширование паролей** (Disable Password Cashing). Последовательность всплывающих в ходе выполнения этой процедуры окон показана на рис. 3.9.

Затем внесенные изменения следует сохранить на диске в виде файла системных правил с расширением POL, назвав его по своему усмотрению. При следующем запуске Windows 95/98 файл системных правил загрузится автоматически и будет определять работу компьютера.

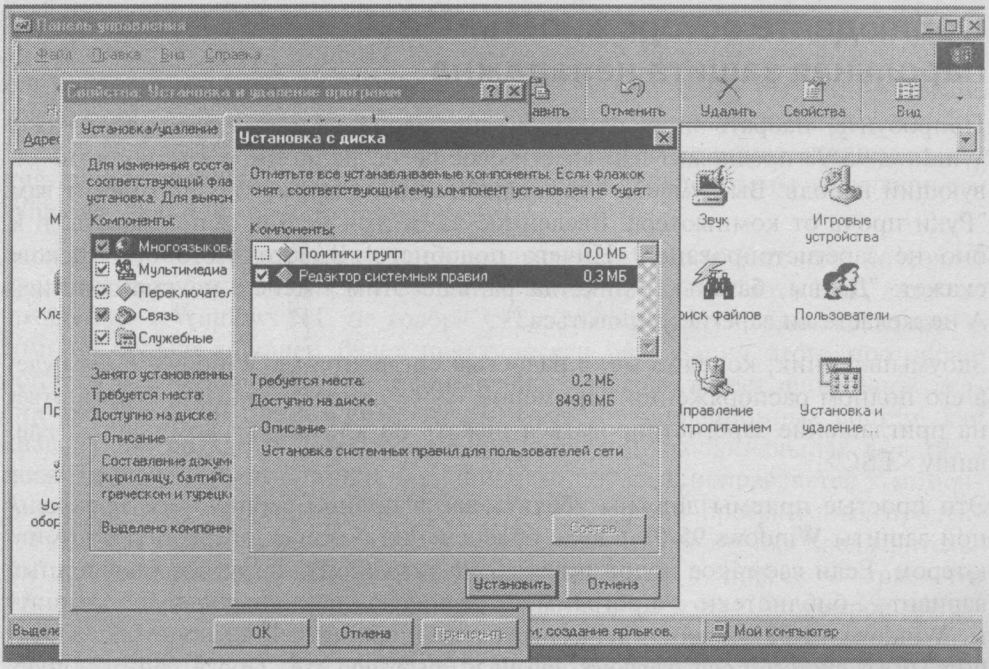


Рис. 3.8. Установка Редактора системных правил

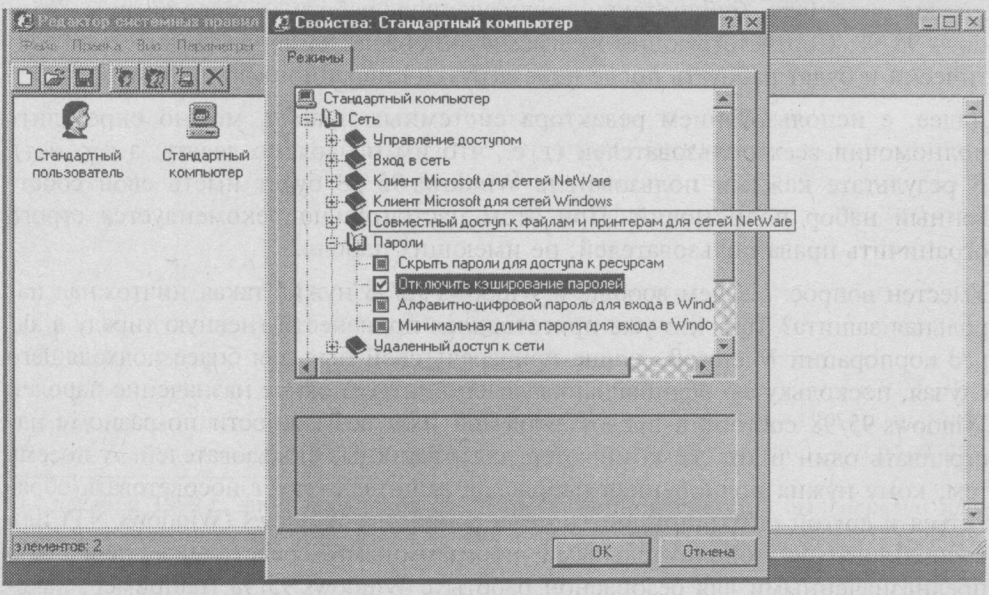


Рис. 3.9. Отключение кэширования паролей с помощью Редактора системных правил

Соблюдайте осторожность: парольная защита ненадежна

Попробуйте набрать произвольную последовательность символов, когда Windows 95/98 предложит вам ввести свое регистрационное имя и соответствующий пароль. Вы думаете, эта операционная система гневно ответит вам: "Руки прочь от компьютера! Введенное вами имя нельзя использовать, т. к. оно не зарегистрировано"? Ничего подобного! Вместо этого она ласково скажет: "Да вы, батенька, никогда раньше этим именем не пользовались. А не желаете ли зарегистрироваться?"

Злоумышленник, конечно же, с радостью согласится, и Windows 95/98 будет в его полном распоряжении. Примерно то же самое получится, если в ответ на приглашение зарегистрироваться нажать на клавиатуре компьютера клавишу <ESC>.

Эти простые приемы должны убедить вас в полной бесполезности парольной защиты Windows 95/98 в деле обеспечения безопасной работы с компьютером. Если вас такое положение дел не устраивает, загрузите улучшенный вариант библиотеки программ, реализующий парольную защиту в Windows 95/98, с Internet-сервера корпорации Microsoft по адресу **http://www.microsoft.com/windows/download/mspwlupd.exe**. После запуска программы mspwlupd.exe надо будет ответить **Да** (Yes) на вопрос о том, желаете ли вы заменить библиотеку, реализующую работу с паролями в Windows 95/98. Соответствующее программное обеспечение установится автоматически и будет работать после перезагрузки операционной системы.

Далее, с использованием редактора системных правил, можно определить полномочия всех пользователей (т. е., что им позволено делать, а что нет). В результате каждый пользователь Windows 95/98 будет иметь свой собственный набор полномочий. При этом настоятельно рекомендуется строго ограничить права пользователей, не имеющих пароля.

Уместен вопрос: а зачем вообще в Windows 95/98 нужна такая ничтожная парольная защита? Тем, кто уже приготовился произнести гневную тираду в адрес корпорации Microsoft, лучше приберечь свой пыл для более подходящего случая, поскольку ею официально заявлено, что основное назначение паролей Windows 95/98 состоит в предоставляемой ими возможности по-разному настраивать один и тот же компьютер для различных пользователей. А посему тем, кому нужна полноценная парольная защита, следует посоветовать обратиться к другой операционной системе семейства Windows (Windows NT) или воспользоваться дополнительными программными средствами, специально предназначенными для безопасной работы с Windows 95/98 (например, пакетом программ Norton Your Eyes Only компании Symatec).

Что же касается новейшей операционной системы Windows 2000[8], то хотя в дополнение ко всему лучшему, что имелось в предыдущих версиях операционных систем семейства Windows, в ней должно было быть реализовано множество совершенно новых возможностей и технологий, способных повысить эффективность работы практически любого пользователя, делать какие-либо определенные выводы относительно подсистемы безопасности Windows 2000 пока еще преждевременно.

Конечно, чисто теоретически использование протокола Kerberos[9] позволяет существенно улучшить механизм аутентификации в Windows 2000 даже по сравнению с Windows NT, не говоря уже о Windows 95/98. Однако вопрос о том, насколько хорошей будет практическая реализация этого протокола, остается открытым вплоть до официального выхода в свет финальной версии Windows 2000. Дело в том, что бета-версии операционных систем, как правило, не обладают функциональной полнотой их финальных версий, а также часто содержат ошибки, большинство которых исправляется к моменту выпуска финальных версий. Поэтому бета-версии программных продуктов могут служить лишь подсобным материалом для сугубо предварительного тестирования. "Сенсационная" информация в прессе о том, что пользователи могут регистрироваться в третьей по счету бета-версии Windows 2000 без ввода пароля, отнюдь не означает, что то же самое можно будет проделывать в ее финальной версии, которая, как планируется, должна поступить в продажу 17 февраля 2000 г. И целенаправленный поиск дыр в подсистеме безопасности операционной системы Windows 2000 лучше всего отложить до момента выхода ее финальной версии.

---

[8] 15 декабря 1999 г. финальный код этой операционной системы был передан для промышленного тиражирования с тем, чтобы успеть начать ее поставки 17 февраля 2000 г.

[9] Протокол Kerberos был разработан в Массачусетском технологическом институте для аутентификации пользователей. Корпорация Microsoft встроила Kerberos в Windows 2000 в качестве протокола аутентификации, используемого по умолчанию.

Глава 4

Безопасность компьютерной сети

Сканеры

Когда-то давным-давно жесткие диски персональных компьютеров были объемом всего-навсего 10 Мбайт, а их оперативная память не превышала 640 Кбайт. Модемы работали на скоростях от 300 до 1200 бит/с, и мало кто из пользователей "персоналок" слышал о глобальной компьютерной сети Internet. Конечно, эта сеть существовала уже тогда, но использовалась исключительно в военных целях, а работа с ней осуществлялась только при помощи командной строки. Но не это служило основным препятствием для массового доступа к сети Internet. Вычислительные машины, которые могли быть задействованы в качестве серверов, были очень дорогими — их стоимость исчислялась цифрами с пятью-шестью нулями (в долларах). Да и сами персональные компьютеры стоили тогда весьма недешево, и были по карману только обеспеченным людям.

Итак, представим себе пригородный район, в котором проживают люди с достатком. Солидные дома с просторными гаражами и аккуратно подстриженными газонами. Близится полночь. На улицах пустынно, в окнах темно. И только одно окно ярко светится в темноте. Там, за персональным компьютером сидит юноша лет 15—17 и обзванивает при помощи модема телефонные номера, которые ему сообщил приятель. В большинстве случаев на другом конце провода оказывается другой модем, и на экране персонального компьютера появляется приглашение зарегистрироваться, т. е. ввести имя и пароль. Каждый раз, получив такое регистрационное приглашение, юноша начинает лихорадочно перебирать различные пары имен и соответствующих им паролей. Наконец, одна пара подходит, и юный взломщик получает возможность управлять удаленным компьютером, сидя дома.

Сейчас уже трудно поверить, что первым компьютерным взломщикам приходилось так напрягаться. Ведь известно, что они очень не любят выполнять рутинную работу и при всяком удобном случае стараются заставить свои

компьютеры заниматься такой работой. Поэтому неудивительно, что компьютерные взломщики вскоре создали специальное программное средство, чтобы не набирать вручную дюжину команд. Это программное средство назвали *боевым номеронабирателем.*

Боевой номеронабиратель представляет собой программу, обзванивающую заданные пользователем телефонные номера в поисках компьютеров, которые в ответ на поступивший звонок выдают регистрационное приглашение. Программа аккуратно сохраняет в файле на жестком диске все такие телефонные номера вместе с данными о скорости соединения с ними. Одним из самых известных и совершенных боевых номеронабирателей является TONELOC, предназначенный для работы в среде операционной системы DOS (он может быть запущен под управлением Windows 95/98 в режиме командной строки).[1]

Дальнейшее совершенствование боевых номеронабирателей привело к созданию *сканеров.* Первые сканеры были весьма примитивными и отличались от боевых номеронабирателей только тем, что специализировались исключительно на выявлении компьютеров, подключенных к сети Internet или к другим сетям, использующим протокол TCP/IP. Они были написаны на языке сценариев программной оболочки операционной системы UNIX. Такие сканеры пытались подсоединиться к удаленной хост-машине через различные порты TCP/IP, отправляя всю информацию, которая выводилась на устройство стандартного вывода этой хост-машины, на экран монитора того компьютера, где был запущен сканер.

Ныне сканеры стали весьма грозным оружием как нападения, так и защиты в Internet. Что же представляет собой современный сканер?

Сканер в вопросах и ответах

Что такое сканер?

Сканер — это программа, предназначенная для автоматизации процесса поиска слабостей в защите компьютеров, подключенных к сети в соответствии с протоколом TCP/IP. Наиболее совершенные сканеры обращаются к портам TCP/IP удаленного компьютера и в деталях протоколируют отклик, который они получают от этого компьютера. Запустив сканер на своем компьютере, пользователь, скажем, из подмосковной Малаховки, даже не выходя из дома, может найти бреши в защитных механизмах сервера, расположенного, например, в Лос-Анджелесе.

---

[1] Если у кого-то появится непреодолимое желание попробовать боевой номеронабиратель в действии, то он может обратиться в Internet по адресу **ftp://ftp.fc.net/pub/defcon/TONELOC/tl110.zip**.

Каковы системные требования для работы со сканерами?

Большинство сканеров предназначено для работы в среде операционной системы UNIX, хотя к настоящему времени такие программы имеются практически для любой операционной системы. Возможность запустить сканер на конкретном компьютере зависит от операционной системы, под управлением которой работает этот компьютер, и от параметров подключения к Internet. Есть сканеры, которые функционируют только в среде UNIX, а с остальными операционными системами оказываются несовместимыми. Другие отказываются нормально работать на устаревших компьютерах с Windows 3.11 и с медленным (до 14 400 бит/с) доступом к Internet, осуществляемым по коммутируемым линиям. Такие компьютеры будут надоедать сообщениями о переполнении стека, нарушении прав доступа или станут просто зависать.

Критичным является и объем оперативной памяти компьютера. Сканеры, которые управляются при помощи командной строки, как правило, более умеренны в своих требованиях, предъявляемых к объему оперативной памяти. А самыми "прожорливыми" являются сканеры, снабженные оконным графическим интерфейсом пользователя.

Трудно ли создать сканер?

Написать сканер не очень трудно. Для этого достаточно хорошо знать протоколы TCP/IP, уметь программировать на C или Perl и на языке сценариев, а также разбираться в программном обеспечении сокетов. Но и в этом случае не следует ожидать, что созданный вами сканер принесет большую прибыль, поскольку в настоящее время предложение на рынке сканеров значительно превышает спрос на них. Поэтому наибольшая отдача от усилий, вложенных в написание сканера, будет скорее моральной (от осознания хорошо выполненной работы), чем материальной.

Что не по силам даже самому совершенному сканеру?

Не следует переоценивать положительные результаты, которых можно достичь благодаря использованию сканера. Действительно, сканер может помочь выявить дыры в защите хост-машины, однако в большинстве случаев информацию о наличии этих дыр сканер выдает в довольно завуалированном виде, и ее надо еще уметь правильно интерпретировать. Сканеры редко снабжаются достаточно полными руководствами пользователя. Кроме того, сканеры не в состоянии сгенерировать пошаговый сценарий взлома исследуемой компьютерной системы. Поэтому для эффективного использования сканеров на практике прежде всего необходимо научиться правильно интерпретировать собранные с их помощью данные, а это возможно только при наличии глубоких знаний в области сетевой безопасности и богатого опыта.

Насколько легальны сканеры?

Обычно сканеры создаются и используются специалистами в области сетевой безопасности. Как правило, они распространяются через сеть Internet, чтобы с их помощью системные администраторы могли проверять компьютерные сети на предмет наличия в них изъянов. Поэтому обладание сканерами, равно как и их использование на практике, вполне законно. Однако рядовые пользователи, не являющиеся системными администраторами, должны быть готовы к тому, что, если они будут применять сканеры для обследования чужих сетей, то могут встретить яростное сопротивление со стороны администраторов этих сетей. Более того, некоторые сканеры в процессе поиска брешей в защите компьютерных сетей предпринимают такие действия, которые по закону могут квалифицироваться как несанкционированный доступ к компьютерной информации, или как создание, использование и распространение вредоносных программ, или как нарушение правил эксплуатации компьютеров, компьютерных систем и сетей. И если следствием этих деяний стало уничтожение, блокирование, модификация или копирование информации, хранящейся в электронном виде, то виновные в них лица в соответствии с российским законодательством подлежат уголовному преследованию. А значит, прежде чем начать пользоваться первым попавшимся под руку бесплатным сканером для UNIX-платформ, стоит убедиться, а не копирует ли случайно этот сканер заодно и какие-нибудь файлы с диска тестируемой им хост-машины (например, файл password из каталога /ETC).

В чем различие между сканерами и сетевыми утилитами?

Часто к сканерам ошибочно относят утилиты типа host, rusers, finger, Traceroute, Showmount. Связано это с тем, что, как и сканеры, данные утилиты позволяют собирать полезную статистическую информацию о сетевых службах на удаленном компьютере. Эту информацию можно затем проанализировать на предмет выявления ошибок в их конфигурации.

Действительно, сетевые утилиты выполняют ряд функций, которые характерны для сканеров. Однако в отличие от последних использование этих утилит вызывает меньше подозрений у системных администраторов. Выполнение большинства сетевых утилит на удаленной хост-машине практически не оказывает никакого влияния на ее функционирование. Сканеры же зачастую ведут себя как слон в посудной лавке и оставляют следы, которые трудно не заметить. Кроме того, хороший сканер — явление довольно редкое, а сетевые утилиты всегда под рукой. К недостаткам сетевых утилит можно отнести то, что приходится выполнять вручную слишком большую работу, чтобы добиться того же результата, который при помощи сканера получается автоматически.

Упомянутые выше сетевые утилиты можно встретить в любой операционной системе семейства UNIX. Однако предоставляемые ими возможности для сбора данных об удаленной хост-машине интересуют не только пользователей UNIX. Поэтому неудивительно, что многие из этих утилит были перенесены в другие операционные системы.

Для Windows 95/98 имеются программные пакеты NetScan Tools (**http://www.eskimo.com/~nwps/index.html**), Network Toolbox (**http://www.jriver.com/netbox.html**) и TCP/IP Surveyor (**http://www.rocketdownload.com/details/inte/wssrv32nsrc.htm**), которые реализуют выполнение сетевых утилит host, rusers, finger, Traceroute, ping, WHOIS (рис. 4.1—4.3). Последний из упомянутых пакетов не только осуществляет сбор информации о сети и подключенных к ней компьютерах, но и представляет собранную таким образом информацию в виде графа, вершинами которого служат найденные в сети маршрутизаторы, серверы и рабочие станции.

Пользователям персональных компьютеров типа Macintosh можно посоветовать обратить внимание на программные пакеты MacTCP Watcher (**http://waldo.wi.mit.edu/WWW/tools/util/Mac/MacTCP_Watcher**), QueryIt! (**http://tucows.online.ru/mac/adnload/dlqueryitmac.html**) и WhatRoute (**http://homepages.ihug.co.nz/~bryanc**).

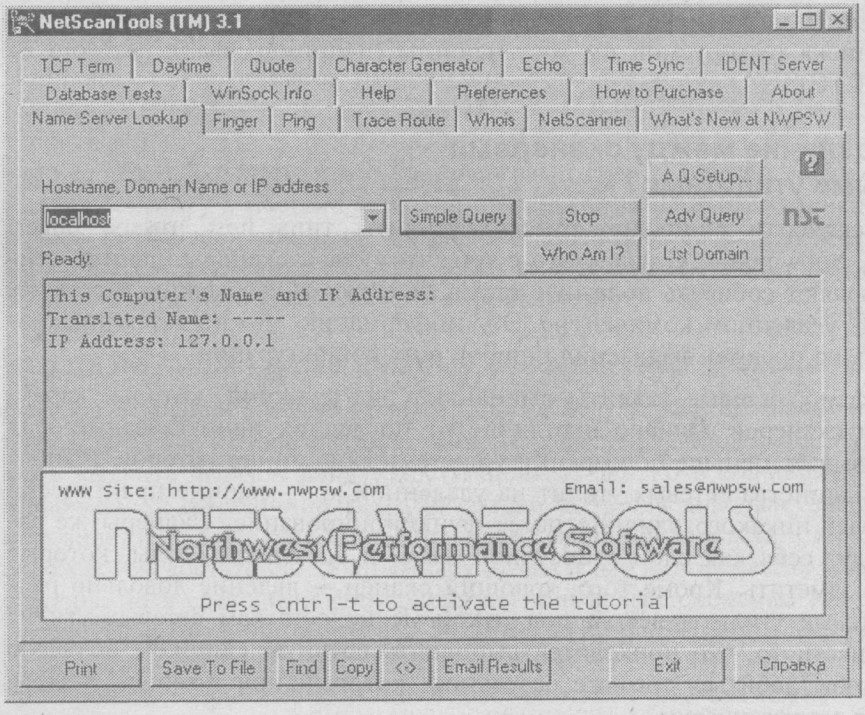


Рис. 4.1. Основное рабочее окно программного пакета NetScan Tools

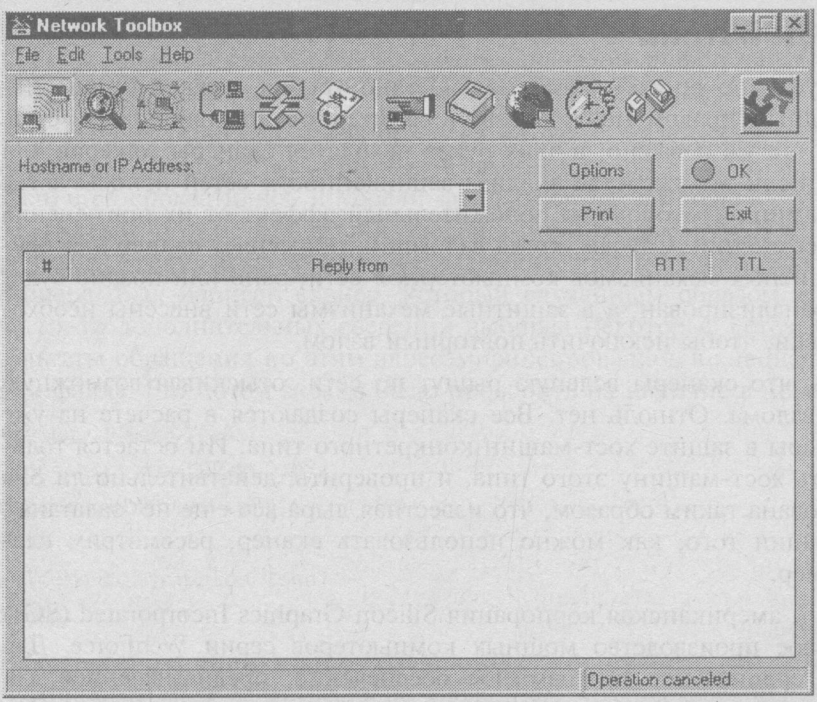


Рис. 4.2. Основное рабочее окно программного пакета Network Toolbox

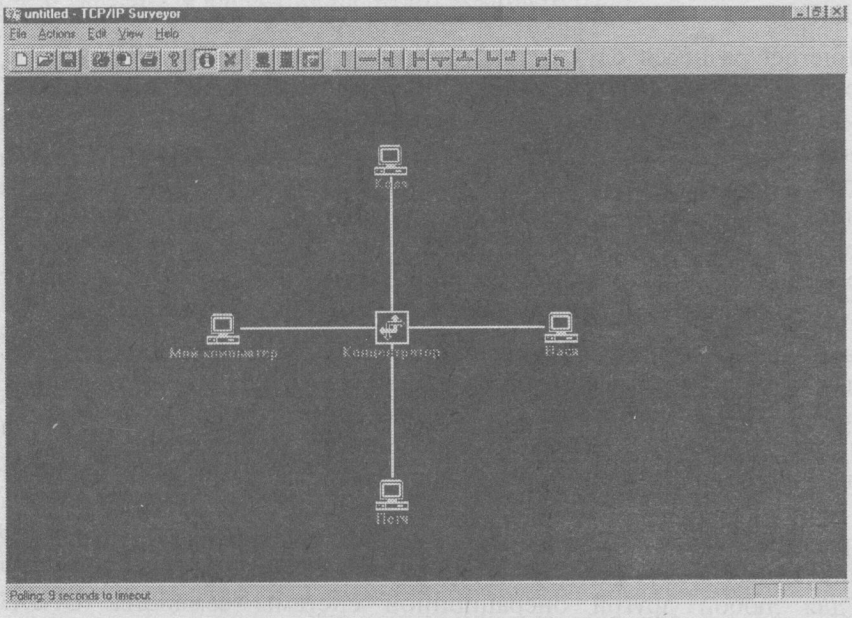


Рис. 4.3. Основное рабочее окно программного пакета TCP/IP Surveyor

Сканер в действии

Сканеры безусловно способствуют повышению уровня сетевой безопасности. Их с полным правом можно назвать санитарами компьютерных сетей. При этом не так уж и важно, в чьих руках находятся сканеры: хакеров или взломщиков. Если эти средства анализа защищенности сетей используются системным администратором, то положительный эффект от их применения не вызывает сомнений. Случай, когда взломщик применяет сканер для преодоления защитных механизмов компьютерной сети, рано или поздно будет выявлен, проанализирован, а в защитные механизмы сети внесены необходимые поправки, чтобы исключить повторный взлом.

Глупо думать, что сканеры вслепую рыщут по сети, отыскивая возможную мишень для взлома. Отнюдь нет. Все сканеры создаются в расчете на уже найденные дыры в защите хост-машин конкретного типа. Им остается только обнаружить хост-машину этого типа, и проверить, действительно ли она сконфигурирована таким образом, что известная дыра все еще не "залатана". Для иллюстрации того, как можно использовать сканер, рассмотрим следующий пример.

В конце 1995 г. американская корпорация Silicon Graphics Incorporated (SGI) начала массовое производство мощных компьютеров серии WebForce. Для них имелось специальное программное обеспечение, предназначенное для создания Web-страниц, насыщенных мультимедийными приложениями. Эти компьютеры работали под управлением операционной системы IRIX (разновидность UNIX).

Вскоре в Internet разнесся слух о том, что некоторые версии IRIX имеют дефект, который позволяет удаленному пользователю регистрироваться в ней под именем lp, при этом пароля вводить не нужно. В результате компьютерный взломщик, беспрепятственно вошедший в IRIX под именем lp, обладал достаточными привилегиями, чтобы скопировать файл password, содержащий зашифрованные имена и пароли пользователей, с компьютера серии WebForce на свой компьютер. Дальше взломщик мог в спокойной обстановке попытаться дешифровать скопированный им файл, чтобы получить полный список имен и паролей пользователей компьютера SGI.

Итак, дыра в защите компьютеров серии WebForce была обнаружена. Оставалось только отыскать эти компьютеры в Internet. Поскольку их системные администраторы лучше разбирались в компьютерной графике и почти ничего не смыслили в вопросах сетевой безопасности, то некоторые каталоги на дисках вверенных их заботам компьютеров были доступны для всеобщего обозрения через Internet. В одном из этих каталогов содержался стандартный файл password из дистрибутива операционной системы IRIX. Большинство перечисленных в нем регистрационных имен пользователей были характерны для любой другой операционной системы семейства UNIX. Однако имелись среди них и имена, которые являлись уникальными имен-

но для IRIX. Таким образом, пользуясь любой поисковой машиной сети Internet, можно было задать в качестве искомой информации эти имена и получить готовый список компьютеров серии WebForce.

Тем не менее, данный метод оказался не очень надежным, поскольку через несколько месяцев системные администраторы компьютеров серии WebForce спохватились и убрали файл password из общедоступных каталогов. Кроме того, не все версии операционной системы IRIX имели рассмотренный дефект. По этой причине взломщики решили прибегнуть к помощи сканера. Был написан сканер, который случайным образом или на основе каких-то дополнительных сведений выбирал некоторые сетевые адреса. Результаты обращения по этим адресам фиксировались в специальном текстовом файле. Его потом можно было проверить на наличие в нем строк типа:

```
Trying 199.200.0.1
Connecting to 199.200.0.1
Escape character is "]"
IRIX 4.1
Welcome to Graphics Town!
Login:
```

Некоторые взломщики автоматизировали этот процесс, заставив сканер регистрироваться в качестве lp на удаленном компьютере. Если регистрация проходила успешно, то информация об этом факте заносилась в файл вместе с указанием адреса найденного компьютера.

SATAN, Jackal и другие сканеры

Среди сканеров наибольшую известность получил SATAN (Security Administrator's Tool for Analyzing Networks). Этот сканер распространяется через Internet (его можно найти, например, по адресу **http://www.fish.com**), начиная с апреля 1995 г.

Ни одно средство анализа сетевой безопасности не вызывало столько споров, сколько пришлось на долю сканера SATAN. Газеты и журналы отреагировали на его появление пространными статьями, в которых изображали SATAN чуть ли не сатанинским изобретением. В телевизионных выпусках новостей звучали грозные предупреждения об опасности, которую для всех пользователей сетей представлял этот сканер.

И действительно, SATAN был довольно необычным для своего времени программным пакетом. Предназначенный для использования на рабочих станциях, работающих под управлением операционной системы UNIX, он обладает дружественным пользовательским интерфейсом. В нем имеются готовые формуляры, в которые пользователю остается только внести минимальные сведения об анализируемом сетевом объекте, а также таблицы, ко-

торые автоматически заполняются по мере накопления информации. При нахождении какого-либо изъяна в сетевой защите пользователю предлагается воспользоваться контекстно-зависимой справкой, чтобы лучше понять суть обнаруженного изъяна.

Под стать своему необычному детищу были и сами авторы SATAN — американцы Д. Фармер (D. Farmer) и У. Венема (W. Venema) — талантливые программисты, хакеры и авторитетные специалисты в области сетевой безопасности.

SATAN предназначен исключительно для работы в среде UNIX. Для его установки и запуска необходимы права администратора. Основная задача SATAN, как это следует из его названия, состоит в том, чтобы отыскивать изъяны в защитных механизмах компьютерной сети, в которой используется протокол TCP/IP. Следует особо подчеркнуть, что речь идет об уже известных изъянах. Это означает, что SATAN не делает ничего такого, что не по силам совершить опытному компьютерному взломщику, который вручную пытается преодолеть сетевую защиту.

Еще одной отличительной особенностью SATAN является его повышенная требовательность к системным ресурсам, особенно к производительности центрального процессора и объему оперативной памяти. Поэтому тем пользователям, которые хотят воспользоваться сканером SATAN, но имеют компьютеры с ограниченными ресурсами, можно посоветовать их увеличить. Если это по каким-либо причинам не реализуемо, следует попытаться избавиться от выполнения лишних процессов и работать с SATAN с помощью командной строки.

Сканер Strobe (Super Optimized TCP Port Surveyor) исследует порты TCP/IP выбранного компьютера и протоколирует все полученные сведения. Основное достоинство Strobe — быстрота сканирования. Если компьютер, на котором запущен Strobe, подключен к Internet через модем, работающий со скоростью 28800 бит/с, то на полное сканирование сервера ему понадобится всего около 30 с. Однако полученная информация будет довольно скудной — лишь общие диагностические сведения о сетевых службах сервера. Кроме того, в хост-машине, подвергшейся сканированию с помощью Strobe, этот факт не останется не замеченным (скорее всего, он будет зафиксирован в файле /var/adm/messages). Сканер Strobe можно скачать из Internet по адресу **http://www.thedarkside.demon.co.uk/download/security/strobe.tgz**.

Jackal (доступен в Internet по адресу **http://www.giga.or.at/pub/hacker/unix**) — это сканер-"призрак". Он анализирует сетевые домены, проникая сквозь брандмауэры и не оставляя при этом никаких следов своего проникновения.

Сканер NSS (Network Security Scanner) замечателен прежде всего тем, что написан на языке Perl и, следовательно, может быть легко перенесен на любую платформу, для которой имеется интерпретатор этого языка. Другим достоинством NSS является скорость его работы. Желающие познакомиться

с NSS поближе также могут обратиться в Internet по адресу **http://www.giga.or.at/pub/hacker/unix**.

Более узкая специализация характерна для IdentTCPscan (предназначен для определения регистрационного имени пользователя, работающего с данным портом TCP/IP), FSPScan (осуществляет поиск серверов, которые поддерживают сетевой протокол FSP), XSCAN (анализирует X-серверы с целью нахождения изъянов в их конфигурации) и CONNECT (ищет серверы, работающие с протоколом TFTP). Все эти сканеры можно найти там же, где и Jackal с NSS.

Исторически сложилось, что подавляющее большинство сканеров было создано для операционной системы UNIX. Поэтому довольно трудно отыскать предназначенные для других платформ сканеры, которые по своей мощи, удобству использования и богатству возможностей могли бы сравниться с SATAN. Кроме того, многие отличные от UNIX платформы, поддерживающие протокол TCP/IP, реализуют его в неполном объеме.

Пользователям Windows 95/98, которые непременно хотят поскорее попробовать свои силы в поиске изъянов сетевой защиты, можно посоветовать применить сканер, который входит в упоминавшийся выше программный пакет Network Toolbox.

Следует отметить, что в средствах защиты сетей от взлома постоянно обнаруживаются новые изъяны. Соответственно претерпевают изменения и сканеры. Внесение в них поправок — дело не особенно трудоемкое: достаточно дописать исходный код сканера с учетом недавно найденных брешей в сетевых защитных механизмах, и новая, более совершенная, версия сканера готова! Вот почему сканеры продолжают оставаться одним из самых главных средств защиты сетей. Системным администраторам необходимо иметь в своем распоряжении как можно больше таких средств — хороших и разных.

Анализаторы протоколов

В настоящее время технология построения компьютерных сетей Ethernet стала самым распространенным решением. Сети Ethernet завоевали огромную популярность благодаря хорошей пропускной способности, простоте установки и приемлемой стоимости сетевого оборудования. Участки сетей, для которых скорости передачи данных 10 Мбит/с недостаточно, можно довольно легко модернизировать, чтобы повысить эту скорость до 100 Мбит/с (Fast Ethernet) или даже до 1 Гбит/с (Gigabit Ethernet).

Однако технология Ethernet не лишена существенных недостатков. Основной из них — передаваемая информация не защищена. Компьютеры, подключенные к сети Ethernet, оказываются в состоянии перехватывать информацию, адресованную своим соседям. Основной причиной тому является

принятый в сетях Ethernet так называемый *широковещательный механизм обмена сообщениями.*

Локальное широковещание

В сети типа Ethernet подключенные к ней компьютеры, как правило, совместно используют один и тот же кабель, который служит средой для пересылки сообщений между ними. Если в комнате одновременно громко говорят несколько людей, разобрать что-либо из сказанного ими будет очень трудно. Когда по сети начинают "общаться" сразу несколько компьютеров, выделить из их "цифрового гвалта" полезную информацию и понять, кому именно она предназначена, практически невозможно. В отличие от человека, компьютер не может поднять руку и попросить тишины, поэтому для решения данной проблемы требуются иные, более сложные действия.

Компьютер сети Ethernet, желающий передать какое-либо сообщение по общему каналу, должен удостовериться, что этот канал в данный момент свободен. В начале передачи компьютер прослушивает несущую частоту сигнала, определяя, не произошло ли искажения сигнала в результате возникновения коллизий с другими компьютерами, которые ведут передачу одновременно с ним. При наличии коллизии компьютер прерывает передачу и "замолкает". По истечении некоторого случайного периода времени он пытается повторить передачу.

Если компьютер, подключенный к сети Ethernet, ничего не передает сам, он, тем не менее, продолжает "слушать" все сообщения, передаваемые по сети другими компьютерами. Заметив в заголовке поступившей порции данных свой сетевой адрес, компьютер копирует эти данные в свою локальную память.

Существуют два основных способа объединения компьютеров в сеть Ethernet. В первом случае компьютеры соединяются при помощи *коаксиального* кабеля. Этот кабель черной змейкой вьется от компьютера к компьютеру, соединяясь с сетевыми адаптерами Т-образным разъемом. Такая топология на языке профессионалов называется *сетью Ethernet 10Base2.* Однако ее еще можно назвать сетью, в которой "все слышат всех". Любой компьютер, подключенный к сети, способен перехватывать данные, посылаемые по этой сети другим компьютером.

Во втором случае каждый компьютер соединен кабелем типа *витая пара* с отдельным портом центрального коммутирующего устройства — концентратора или с коммутатора. В таких сетях, которые называются *сетями Ethernet 10BaseT,* компьютеры поделены на группы, именуемые *доменами коллизий.* Домены коллизий определяются портами концентратора или коммутатора, замкнутыми на общую шину. В результате коллизии возникают не между всеми компьютерами сети, а по отдельности — между теми из них, которые

входят в один и тот же домен коллизий, что повышает пропускную способность всей сети.

В последнее время в крупных сетях стали появляться коммутаторы нового типа, которые не используют широковещание и не замыкают группы портов между собой. Вместо этого все передаваемые по сети данные буферизуются в памяти и отправляются по мере возможности. Однако подобных сетей пока довольно мало — не более 10% от общего числа сетей типа Ethernet.

Таким образом, принятый в подавляющем большинстве Ethernet-сетей алгоритм передачи данных требует от каждого компьютера, подключенного к сети, непрерывного "прослушивания" всего без исключения сетевого трафика. Предложенные алгоритмы доступа, при использовании которых компьютеры отключались бы от сети на время передачи "чужих" сообщений, так и остались нереализованными из-за их чрезмерной сложности и малой эффективности.

Анализатор протоколов как он есть

Как уже было сказано, сетевой адаптер каждого компьютера в сети Ethernet как правило "слышит" все, о чем "толкуют" между собой его соседи по сегменту этой сети. Но обрабатывает и помещает в свою локальную память он только те порции (так называемые *кадры*) данных, которые содержат его уникальный сетевой адрес.

В дополнение к этому подавляющее большинство современных Ethernet-адаптеров допускают функционирование в особом режиме, называемом *беспорядочным* (promiscuous). При использовании данного режима адаптер копирует в локальную память компьютера все без исключения передаваемые по сети кадры данных.

Специализированные программы, переводящие сетевой адаптер в беспорядочный режим и собирающие весь трафик сети для последующего анализа, называются *анализаторами протоколов*. Администраторы сетей широко применяют анализаторы протоколов для осуществления контроля за работой этих сетей и определения их перегруженных участков, отрицательно влияющих на скорость передачи данных. К сожалению, анализаторы протоколов используются и злоумышленниками, которые с их помощью могут перехватывать чужие пароли и другую конфиденциальную информацию.

Надо отметить, что анализаторы протоколов представляют серьезную опасность. Само присутствие в компьютерной сети анализатора протоколов указывает на то, что в ее защитных механизмах имеется брешь. Установить анализатор протоколов мог посторонний человек, который проник в сеть извне (например, если сеть имеет выход в Internet). Но это могло быть и делом рук доморощенного злоумышленника, имеющего легальный доступ к сети. В любом случае к сложившейся ситуации нужно отнестись со всей серьезностью.

Специалисты в области компьютерной безопасности относят атаки на компьютеры при помощи анализаторов протоколов к так называемым *атакам второго уровня*. Это означает, что компьютерный взломщик уже сумел проникнуть сквозь защитные барьеры сети и теперь стремится развить свой успех. При помощи анализатора протоколов он может попытаться перехватить регистрационные имена и пароли пользователей, их секретные финансовые данные (например, номера кредитных карточек) и конфиденциальные сообщения (к примеру, электронную почту). Имея в своем распоряжении достаточные ресурсы, компьютерный взломщик в принципе может перехватывать всю информацию, передаваемую по сети.

Анализаторы протоколов существуют для любой платформы. Но даже если окажется, что для какой-то платформы анализатор протоколов пока еще не написан, с угрозой, которую представляет атака на компьютерную систему при помощи анализатора протоколов, по-прежнему приходится считаться. Дело в том, что анализаторы протоколов исследуют не конкретный компьютер, а протоколы. Поэтому анализатор протоколов может обосноваться в любом узле сети и оттуда перехватывать сетевой трафик, который в результате широковещательных передач попадает в каждый компьютер, подключенный к сети.

Одна из первых атак, проведенных при помощи анализаторов протоколов, была зафиксирована в 1994 г. в США. Тогда неизвестный злоумышленник разместил анализатор протоколов на различных хостах и магистральных узлах сетей Internet и Milnet, в результате чего ему удалось перехватить более 100 тыс. регистрационных имен и паролей пользователей. Среди пострадавших от атаки оказались Калифорнийский государственный университет и Ракетная лаборатория министерства обороны США.

Наиболее частыми целями атак компьютерных взломщиков, которые те осуществляют, используя анализаторы протоколов, являются университеты. Хотя бы из-за огромного количества различных регистрационных имен и паролей, которые могут быть украдены в ходе такой атаки. Да и сами студенты отнюдь не брезгуют возможностями анализаторов протоколов. Нередким является случай, когда несколько студентов, заняв компьютер, подключенный к локальной сети университетской библиотеки, быстро устанавливают с нескольких дискет анализатор протоколов. Затем они просят ничего не подозревающую жертву, сидящую за соседним компьютером: "Вы не могли бы заглянуть в свой почтовый ящик, а то у нас почему-то электронная почта не работает?" Несколько минут спустя вся эта группа компьютерных взломщиков-любителей, перехватив регистрационное имя и пароль доступа соседа к почтовому серверу, с удовольствием знакомится с содержимым его почтового ящика и посылает письма от его имени.

Использование анализатора протоколов на практике не является такой уж легкой задачей, как это может показаться. Чтобы добиться от него хоть какой-

то пользы, компьютерный взломщик должен хорошо знать сетевые технологии. Просто установить и запустить анализатор протоколов нельзя, поскольку даже в небольшой локальной сети из пяти компьютеров трафик составляет тысячи и тысячи пакетов в час. И следовательно, за короткое время выходные данные анализатора протоколов заполнят жесткий диск полностью.

Поэтому компьютерный взломщик обычно настраивает анализатор протоколов так, чтобы он перехватывал только первые 200—300 байт каждого пакета, передаваемого по сети. Обычно именно в заголовке пакета размещается информация о регистрационном имени и пароле пользователя, которые, как правило, больше всего интересуют взломщика. Тем не менее, если в распоряжении взломщика имеется достаточно пространства на жестком диске, то увеличение объема перехватываемого трафика пойдет ему только на пользу. В результате он может дополнительно узнать много интересного.

На серверах в сети Internet есть множество анализаторов протоколов, которые отличаются лишь набором доступных функций. Например, поиск по запросам protocol analyzer и sniffer на сервере **www.softseek.com** сразу дает ссылки на добрый десяток программных пакетов.

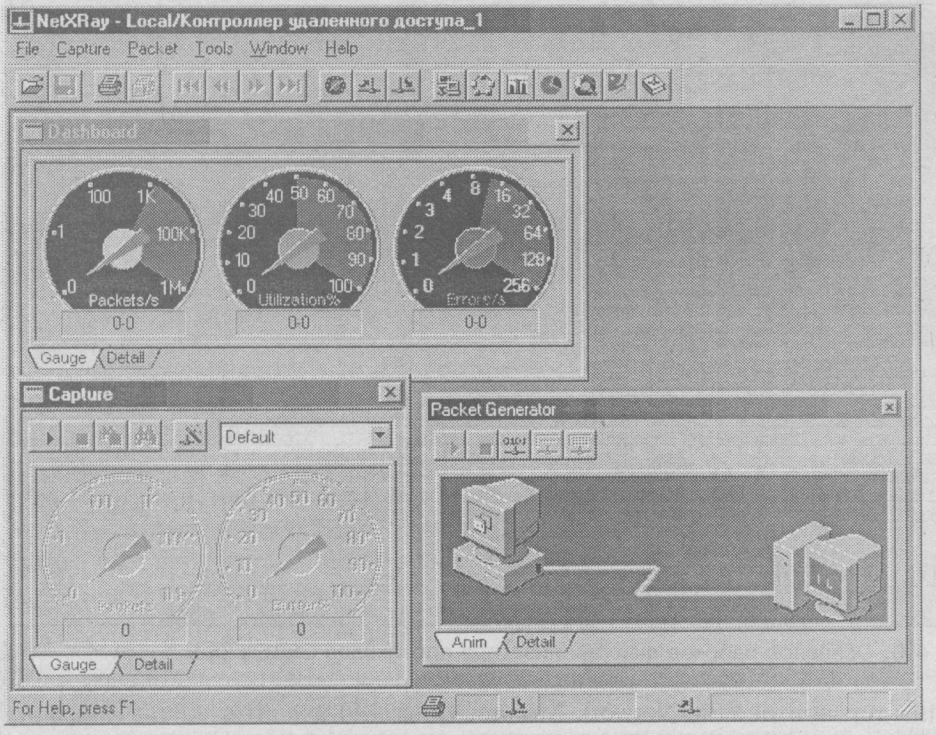


Рис. 4.4. Основное рабочее окно анализатора протоколов NetXRay

Для компьютеров, работающих под управлением операционных систем Windows, одними из лучших являются анализаторы протоколов Lan Explorer компании Intellimax и NetXRay компании Network Associates. NetXRay (в переводе с английского — Сетевой рентген) обладает обширным набором функций, которые позволяют делать моментальный снимок "внутренностей" сети Ethernet, определять, какие ее узлы и сегменты несут наибольшую нагрузку, составлять отчеты и строить диаграммы на основе полученных данных (рис. 4.4). Бесплатная версия NetXRay доступна в Internet по адресу **http://www.nai.com/asp_set/products/tnv/snifferbasic_intro.asp**. Анализатор протоколов Lan Explorer (в переводе с английского — Анализатор ЛВС) не уступает по своей функциональности NetXRay, имеет очень хороший пользовательский интерфейс, удобен и прост в использовании (рис. 4.5). Пробная 15-дневная версия Lan Explorer доступна по адресу **http://www.intellimax.com/ftpsites.htm**.

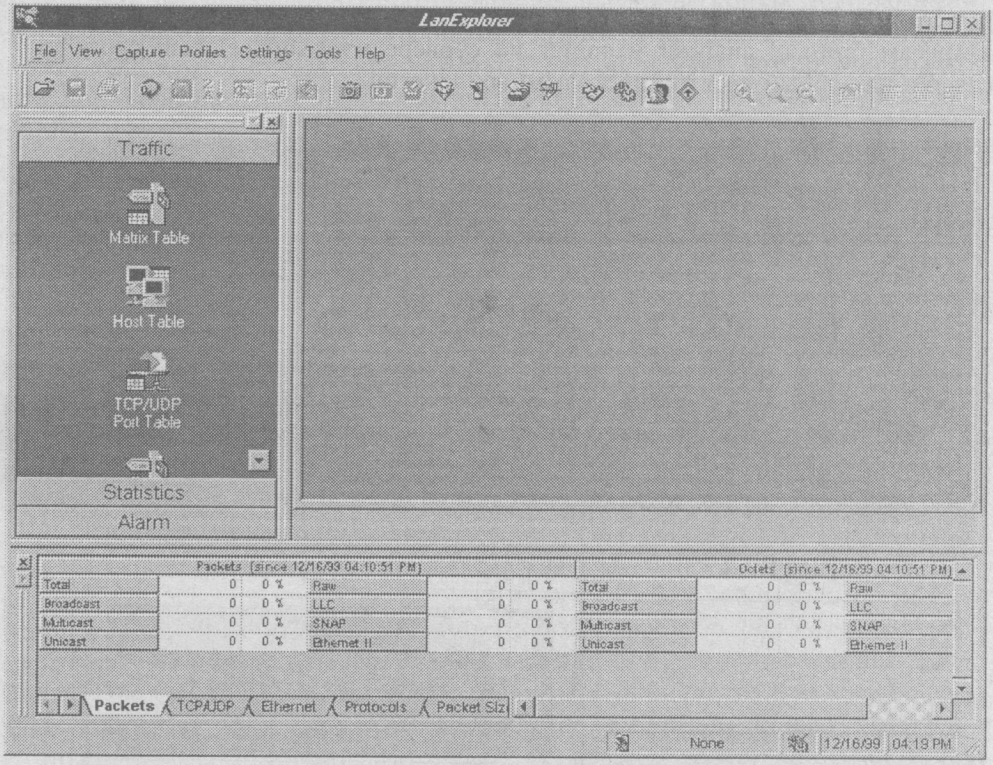


Рис. 4.5. Основное рабочее окно анализатора протоколов Lan Explorer

Анализатор протоколов Network Monitor (рис. 4.6) входит в состав операционной системы Windows NT 4.0 Server корпорации Microsoft. Для его установки следует в **Панели управления** (Control Panel) дважды щелкнуть на

пиктограмме **Сеть** (Network), затем перейти на вкладку **Службы** (Services), нажать кнопку **Добавить** (Add) и в появившемся диалоговом окне выбрать Network Monitor Tools and Agent. После установки Network Monitor можно запустить из папки Network Analysis Tools раздела **Администрирование** (Administrative Tools) в меню **Программы** (Programs).

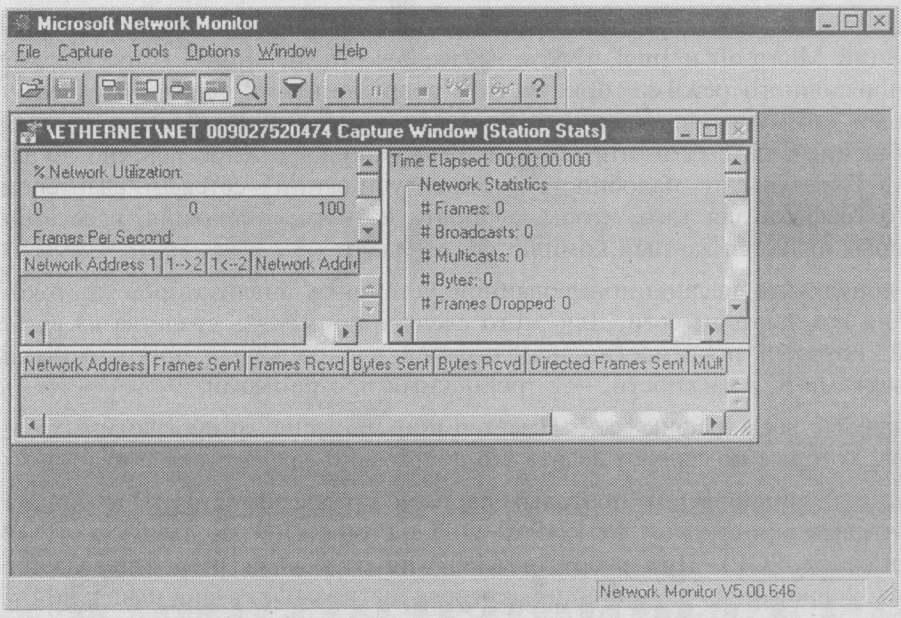


Рис. 4.6. Основное рабочее окно анализатора протоколов Network Monitor

Защита от анализаторов протоколов

Надо сразу оговориться, что в советах относительно того, как защищаться от анализатора протоколов, нуждаются только те, кто желает дать отпор компьютерным взломщикам, использующим анализаторы протоколов для организации атак на компьютерные системы, подключенные к сети. В руках сетевого администратора анализатор протоколов является весьма полезным инструментом, помогающим находить и устранять неисправности, избавляться от узких мест, снижающих пропускную способность сети, и обнаруживать проникновение в нее компьютерных взломщиков.

Посоветовать можно следующее:

- Обзаведитесь сетевым адаптером, который принципиально не может функционировать в беспорядочном режиме. Такие адаптеры в природе существуют. Одни адаптеры не поддерживают беспорядочный режим на аппаратном уровне (их меньшинство), а остальные просто снабжаются

драйвером, не допускающим работу в беспорядочном режиме, хотя этот режим и реализован в них аппаратно. Чтобы отыскать адаптер, не поддерживающий беспорядочный режим, достаточно связаться со службой технической поддержки любой компании, торгующей анализаторами протоколов, и выяснить, с какими адаптерами их программные пакеты не работают.

- Учитывая, что спецификация PC99, подготовленная по инициативе корпораций Microsoft и Intel, требует безусловного наличия в сетевой карте беспорядочного режима, приобретите современный сетевой интеллектуальный коммутатор, который буферизует каждое отправляемое по сети сообщение в памяти и отправляет его по мере возможности точно по адресу. В результате надобность в "прослушивании" сетевым адаптером всего трафика для того, чтобы выбирать из него сообщения, адресатом которых является данный компьютер, отпадает.
- Не допускайте несанкционированной установки анализаторов протоколов на компьютеры сети. Для этого следует применять средства из арсенала, который повсеместно используется для борьбы с программными закладками и, в частности, — с троянскими программами.
- Шифруйте весь трафик сети. Имеется широкий спектр программных пакетов, которые позволяют делать это достаточно эффективно и надежно.

Возможность шифрования почтовых паролей предоставляется надстройкой над почтовым протоколом POP (Post Office Protocol) — протоколом APOP (Authentication POP). При работе с APOP по сети каждый раз передается новая зашифрованная комбинация, которая не позволяет злоумышленнику извлечь какую-либо практическую пользу из информации, перехваченной с помощью анализатора протоколов. Проблема только в том, что не все почтовые серверы и клиенты поддерживают APOP.

Другой продукт под названием Secure Shell, или сокращенно — SSL, был изначально разработан финской компанией SSH Communications Security (**http://www.ssh.fi**) и в настоящее время имеет множество реализаций, доступных бесплатно через Internet. SSL представляет собой защищенный протокол для осуществления безопасной передачи сообщений по компьютерной сети с помощью шифрования.

Особую известность среди компьютерных пользователей приобрела серия программных пакетов, предназначенных для защиты передаваемых по сети данных путем шифрования и объединенных присутствием в их названии аббревиатуры PGP, которая означает Pretty Good Privacy. Бесплатно распространяемые версии программ шифрования из этой серии можно отыскать в Internet по адресу **http://www.pgpi.org**.

Криптографические методы защиты информации

Глава 5

Основы криптографии

Зачем нужна криптография

Криптография бывает разная. Есть криптография, которая не позволяет вашему малолетнему сынишке насладиться созерцанием изображений обнаженной натуры, втайне переписанных вами ночью из сети Internet на домашний компьютер. Но кроме этого существует и криптография, которая не дает вашему правительству узнать про вас то, что вы не хотите, чтобы оно знало.

Если я напишу письмо, положу его в сейф, а сейф спрячу подальше от посторонних глаз и на спор попрошу вас прочесть мое письмо, — это не безопасность, а банальная игра в прятки. Но если я предоставлю вам свободный доступ не только к моему запертому сейфу, но и к сотне других подобных ему сейфов, к их ключевым комбинациям, позволю вам и лучшим взломщикам сейфов в мире во всех подробностях изучить конструкцию моего сейфа вместе с замком, но все равно никто не сможет открыть мой сейф и прочесть находящееся в нем письмо — вот это настоящая безопасность!

В течение многих лет такая криптография была прерогативой исключительно военных. Агентство национальной безопасности (АНБ)[1] в США, а также соответствующие ведомства в Англии, России, и в других развитых странах тратили огромные деньги на обеспечение безопасности собственных линий связи и на добывание полезной информации из чужих. А простые граждане, не имея достаточных финансовых средств и не обладая необходимыми знаниями, были бессильны защитить свою частную жизнь от вмешательства со стороны государства.

---

[1] АНБ создано 4 ноября 1952 г. согласно секретной директиве президента США. Основные функции АНБ состоят в перехвате и чтении переписки всех без исключения зарубежных государств, независимо от того, дружественно или враждебно они настроены по отношению к США.

За последние 20 лет положение в корне изменилось. Этому способствовали чрезвычайно интенсивные научные исследования в области криптографии, которые привели к тому, что современная компьютерная криптография "обитает" не только в стенах строго охраняемых военных учреждений, но и на страницах общедоступных журналов, монографий и учебников. Теперь при желании кто угодно может зашифровать свои данные таким образом, чтобы защитить их от самого могущественного противника — правительственных ведомств, специализирующихся на краже информации.

Нуждается ли рядовой обыватель в подобной защите? Конечно, нуждается. Под ее прикрытием он может вести учет своих финансовых затрат, не опасаясь, что с ними ознакомятся налоговые службы. Он может разрабатывать новое изделие для последующей выгодной продажи на рынке, надежно пряча детали своей разработки от конкурентов. Он может завести любовницу и не бояться, что волей случая жена окажется посвященной в подробности его тайной любовной переписки. Ведь любой не застрахован от ситуации, когда он занимается какой-то деятельностью, которую он сам не считает незаконной, но которая впоследствии может быть объявлена кем-то вне закона. Каковы бы ни были причины, человек имеет право на частную жизнь, в том числе — на сохранение в тайне любых сведений, которые он сочтет нужным не предавать огласке. И никому, в том числе и государству, не должно быть до них никакого дела.

На практике все обстоит совершенно иначе. Своим Указом № 334 от 3 апреля 1995 г. президент России строго-настрого запретил "деятельность юридических и физических лиц, связанную с разработкой, производством, реализацией и эксплуатацией шифровальных средств, [...] предоставлением услуг в области шифрования информации", без государственных лицензий. А российские спецслужбы обязал "осуществлять выявление юридических и физических лиц, нарушающих требования настоящего Указа". Что с такими лицами будут делать — вырывать им ноздри или просто ставить клеймо на лоб, чтобы другим неповадно было, — в президентском Указе не сказано. Но и без этого у государственных чиновников в России появились все необходимые полномочия, чтобы при необходимости можно было воспрепятствовать внедрению надежных систем криптографической защиты информации. Надо сказать, что аналогично действует и американское правительство, которое путем введения государственного стандарта шифрования с передачей ключей на хранение пытается узаконить для себя право шпионить за своими гражданами.

Таким образом, становится совершенно очевидно, что надежность шифровальных средств — традиционно лишнее качество для "Большого брата", независимо от его национальной принадлежности. Совершенно в духе знаменитого романа "1984", принадлежащего перу английского писателя Джорджа Оруэлла, звучат аргументы сторонников введения ничем не ограниченного права государства на перехват информации частного характера. Они заявля-

ют, что с человеком, которому есть что скрывать от собственного государства, происходит что-то неладное, поэтому к нему необходимо как следует присмотреться.

В любом демократическом государстве правоохранительным органам традиционно предоставлялась возможность на законных основаниях вести слежку, если разрешение на нее было получено в судебном порядке. Ныне этого недостаточно: правительства принуждают своих граждан самих принять все необходимые меры, чтобы можно было без помех вести за ними наблюдение. Тем самым государственная власть узурпирует права, которые в демократическом государстве должны принадлежать народу. И если в такой стране начнет править диктатор, его власть будет неограниченной. Свергнуть его будет невозможно, ибо любая попытка граждан объединиться вместе, чтобы противостоять диктатуре, станет ему известна. Настолько всемогущей является современная технология.

В связи с вышесказанным необходимо, чтобы все спецслужбы, которые обладают правом доступа к современным техническим средствам слежки за гражданами своей страны, действовали в рамках законности и под контролем компетентных органов, подотчетных народу, ибо пересечь обратно границу, один раз перейденную в направлении от демократии к диктатуре, будет уже невозможно. Ведь даже в США — стране с самой развитой, по общему признанию, демократией в мире, никто не имеет достаточных гарантий того, что не разделит судьбу многих честных соотечественников, которые в разное время подвергались преследованию только потому, что их убеждения были неугодны властям.

Самая затяжная и беспощадная война нашего столетия ведется поразительно тихо. В ней обходятся без ракетно-бомбовых ударов и без расстрела заложников. Однако и здесь порочат, калечат и уничтожают людей точно так же, как и на обычной войне. А отвоевываются не чужие территории или право распоряжаться судьбами побежденных, а информация. И противостоят друг другу не "хорошие" и "плохие" парни, даже не озверевшие наемники, а государство и его граждане.

В информационной войне государство пользуется двумя основными видами оружия. Во-первых, в любую сертифицированную государством систему защиты информации встраивается так называемый *потайной ход*, который иногда еще именуют *лазейкой*. Еще в 1991 г. в сенат США был внесен законопроект, который требовал, чтобы любое американское криптографическое оборудование содержало потайной ход, известный АНБ. Поэтому совершенно не удивительно, что один из крупнейших разработчиков криптографического оборудования в США — компания AT&T — нисколько не скрывает того факта, что у американского правительства есть "ключики" от "потайных ходов", ведущих во все ее системы вне зависимости от того, насколько сложные пароли придумают неискушенные пользователи.

Второй способ обмана граждан является более изощренным и состоит в применении *системы с депонированием ключей*. В этом случае прочесть сообщение можно только при наличии всех частей ключа, использованного для засекречивания этого сообщения. Тайна переписки гарантируется до тех пор, пока государству не понадобится с ней ознакомиться. Тогда стоит заручиться согласием, скажем, тех доверенных лиц или организаций, которым были отданы на хранение части ключа, — и тайное станет явным. В США данный подход был подвергнут справедливой критике со стороны большинства специалистов, а корпоративные пользователи с негодованием отвергли саму идею довериться правительству в таком деликатном вопросе, как коммерческая тайна. И это в законопослушной Америке, где ставшие явными попытки спецслужб вторгнуться в частную жизнь своих граждан встречают немедленный и достойный отпор!

Из сказанного следует сделать один важный вывод. Недостаточно уметь защищать себя с помощью закона, непременно требуется еще и криптографическая защита. Шифрование играет слишком большую роль в современном мире, чтобы отдать его на откуп государству.

Криптографические средства можно объявить вне закона. Но ваша личная информация всегда будет легальной, и, следовательно, должна быть надежно защищена от посторонних.

Терминология

Шифрование и расшифрование

Предположим, что отправитель хочет послать сообщение получателю. Более того, отправитель желает засекретить это сообщение, чтобы никто, кроме получателя, не смог его прочитать.

Сообщение состоит из *открытого текста*. Процесс преобразования открытого текста с целью сделать непонятным его смысл для посторонних называется *шифрованием*. В результате шифрования сообщения получается *шифртекст*. Процесс обратного преобразования шифртекста в открытый текст называется *расшифрованием*.

Наука, которая учит, как следует поступать, чтобы сохранить содержание сообщений в тайне, называется *криптографией*. Людей, занимающихся криптографией, зовут *криптографами*. *Криптоаналитики* являются специалистами в области *криптоанализа* — науки о вскрытии шифров, которая отвечает на вопрос о том, как прочесть открытый текст, скрывающийся под шифрованным. Раздел науки, объединяющий криптографию и криптоанализ, именуется *криптологией*.

Обозначим открытый текст буквой P (от английского слова plaintext). Это может быть текстовый файл, битовое изображение, оцифрованный звук —

что угодно. Единственное ограничение связано с тем, что, поскольку предметом изложения является компьютерная криптография, под P понимаются исключительно двоичные данные.

Шифртекст обозначается буквой C (от английского слова ciphertext) и также представляет собой двоичные данные. Объем полученного шифртекста иногда совпадает с объемом соответствующего открытого текста, а иногда превышает его. После зашифрования преобразованный открытый текст может быть передан по каналам компьютерной сети или сохранен в памяти компьютера.

На вход функции шифрования E подается P, чтобы на выходе получить C. В обозначениях, принятых в математике, это записывается как:

$$E(P) = C$$

При обратном преобразовании шифртекста в открытый текст на вход функции расшифрования D поступает C, а на выходе получается P:

$$D(C) = M$$

Поскольку смысл любого криптографического преобразования открытого текста состоит в том, чтобы потом этот открытый текст можно было восстановить в первозданном виде, верно следующее соотношение:

$$D(E(P)) = P$$

Аутентификация, целостность и неоспоримость

Помогая сохранить содержание сообщения в тайне, криптография может быть использована, чтобы дополнительно обеспечить решение следующих задач:

- Аутентификация. Получателю сообщения требуется убедиться, что оно исходит от конкретного отправителя. Злоумышленник не может прислать фальшивое сообщение от чьего-либо имени.
- *Целостность.* Получатель сообщения в состоянии проверить, были ли внесены какие-нибудь изменения в полученное сообщение в ходе его передачи. Злоумышленнику не позволено заменить настоящее сообщение на фальшивое.
- *Неоспоримость.* Отправитель сообщения должен быть лишен возможности впоследствии отрицать, что именно он является автором этого сообщения.

Перечисленные задачи часто приходится решать на практике для организации взаимодействия людей при помощи компьютеров и компьютерных сетей. Подобные же задачи возникают и в случае личностного человеческого общения: часто требуется проверить, а действительно ли ваш собеседник тот, за кого он себя выдает, и подлинны ли предъявленные им документы, будь то паспорт, водительское удостоверение или страховой полис. Вот по-

чему в обыденной жизни не обойтись без аутентификации, проверки целостности и доказательства неоспоримости, а значит и без криптографии.

Шифры и ключи

Криптографический алгоритм, также называемый *шифром* или *алгоритмом шифрования*, представляет собой математическую функцию, используемую для шифрования и расшифрования. Если быть более точным, таких функций две: одна применяется для шифрования, а другая — для расшифрования.

Когда надежность криптографического алгоритма обеспечивается за счет сохранения в тайне сути самого алгоритма, такой алгоритм шифрования называется *ограниченным*. Ограниченные алгоритмы представляют значительный интерес с точки зрения истории криптографии, однако совершенно непригодны при современных требованиях, предъявляемых к шифрованию. Ведь в этом случае каждая группа пользователей, желающих обмениваться секретными сообщениями, должна обзавестись своим оригинальным алгоритмом шифрования. Применение готового оборудования и стандартных программ исключено, поскольку тогда любой сможет приобрести это оборудование и эти программы и ознакомиться с заложенным в них алгоритмом шифрования. Придется разрабатывать собственный криптографический алгоритм, причем делать это надо будет каждый раз, когда кто-то из пользователей группы захочет ее покинуть или когда детали алгоритма случайно станут известны посторонним.

В современной криптографии эти проблемы решаются с помощью использования ключа, который обозначается буквой K (от английского слова key). Ключ должен выбираться среди значений, принадлежащих множеству, которое называется ключевым пространством. И функция шифрования E, и функция расшифрования D зависят от ключа. Сей факт выражается присутствием K в качестве подстрочного индекса у функций E и D:

$$E_K(P) = C$$

$$D_K(C) = P$$

По-прежнему справедливо следующее тождество:

$$D_K(E_K(P)) = P$$

Некоторые алгоритмы шифрования используют различные ключи для шифрования и расшифрования. Это означает, что ключ шифрования K_1 отличается от ключа расшифрования K_2. В этом случае справедливы следующие соотношения:

$$E_{K_1}(P) = C$$

$$D_{K_2}(C) = P$$

$$D_{K_2}(E_{K_1}(P)) = P$$

Надежность алгоритма шифрования с использованием ключей достигается за счет их надлежащего выбора и последующего хранения в строжайшем секрете. Это означает, что такой алгоритм не требуется держать в тайне. Можно организовать массовое производство криптографических средств, в основу функционирования которых положен данный алгоритм. Знание криптографического алгоритма не позволит злоумышленнику прочесть зашифрованные сообщения, поскольку он не знает секретный ключ, использованный для их зашифрования.

Под *криптосистемой* понимается алгоритм шифрования, а также множество всевозможных ключей, открытых и шифрованных текстов.

Симметричные алгоритмы шифрования

Существуют две разновидности алгоритмов шифрования с использованием ключей — *симметричные* и *с открытым ключом*. Симметричным называют криптографический алгоритм, в котором ключ, используемый для шифрования сообщений, может быть получен из ключа расшифрования и наоборот. В большинстве симметричных алгоритмов применяют всего один ключ. Такие алгоритмы именуются *одноключевыми*, или алгоритмами с секретным ключом, и требуют, чтобы отправитель сообщений и их получатель заранее условились о том, каким ключом они будут пользоваться. Надежность одноключевого алгоритма определяется выбором ключа, поскольку его знание дает возможность злоумышленнику без помех расшифровывать все перехваченные сообщения. Поэтому выбранный ключ следует хранить в тайне от посторонних.

Шифрование и расшифрование в симметричных криптографических алгоритмах задаются уже знакомыми формулами:

$$E_K(P) = C$$

$$D_K(C) = P$$

Симметричные алгоритмы шифрования бывают двух видов. Одни из них обрабатывают открытый текст побитно. Они называются *потоковыми алгоритмами*, или *потоковыми шифрами*. Согласно другим, открытый текст разбивается на блоки, состоящие из нескольких бит. Такие алгоритмы называются *блочными*, или *блочными шифрами*. В современных компьютерных алгоритмах блочного шифрования обычно длина блока составляет 64 бита.

Алгоритмы шифрования с открытым ключом

Алгоритмы шифрования с открытым ключом, также называемые *асимметричными* алгоритмами шифрования, устроены так, что ключ, используемый для шифрования сообщений, отличается от ключа, применяемого для их

расшифрования. Более того, ключ расшифрования не может быть за обозримое время вычислен, исходя из ключа шифрования. Свое название алгоритмы с открытым ключом получили благодаря тому, что ключ шифрования не требуется держать в тайне. Любой может им воспользоваться, чтобы зашифровать свое сообщение, но только обладатель соответствующего секретного ключа расшифрования будет в состоянии прочесть это шифрованное сообщение. Ключ шифрования обычно называют *открытым ключом*, а ключ расшифрования — *тайным ключом*. Иногда тайный ключ называют также секретным, однако чтобы избежать путаницы с симметричными алгоритмами, это название не будет использоваться при дальнейшем изложении.

Несмотря на тот факт, что сообщения шифруются с помощью открытого ключа, а расшифровываются с помощью тайного ключа, процесс шифрования и расшифрования все равно записывается так:

$E_K(P) = C$

$D_K(C) = P$

Иногда сообщения шифруются с использованием тайного ключа, а расшифровываются посредством открытого ключа. Несмотря на возможную путаницу, этот факт математически по-прежнему выражается в виде:

$E_K(P) = C$

$D_K(C) = P$

Криптоаналитические атаки

Криптография ставит своей целью сохранение переписки в тайне от посторонних людей, которые захотят с ней ознакомиться. Таких людей криптографы называют злоумышленниками, противниками, перехватчиками или просто врагами. При этом предполагается, что они могут перехватывать любые сообщения, которыми обмениваются отправитель и получатель.

Криптоанализ заключается в получении доступа к открытому тексту шифрованного сообщения. В ходе успешного криптоаналитического исследования криптосистемы может быть найден не только открытый текст, но и сам ключ. Криптоаналитик занимается поисками слабостей в криптосистеме, которые могут позволить ему прочесть шифрованное сообщение, или отыскать ключ, или и то, и другое вместе. Если противник узнал ключ не с помощью криптоанализа, а каким-то другим способом (выкрал или купил), то говорят, что ключ был *скомпрометирован*.

Попытка криптоанализа называется атакой. Успешная криптоаналитическая атака зовется *взломом*, или *вскрытием*.

В современной криптологии принято считать, что надежность шифра определяется только секретностью используемого ключа. Правило, впервые сформулированное голландцем А. Керкхоффом (1835—1903), гласит о том,

что весь механизм шифрования, за исключением значения ключа, предположительно известен противнику. Это предположение является довольно естественным. Например, хотя ФАПСИ вряд ли знакомит АНБ со своими криптографическими алгоритмами, какое-то представление о них там все равно имеется. Следовательно, правило Керкхоффа является относительно хорошим допущением при рассмотрении надежности алгоритмов шифрования. Если шифр невозможно взломать, зная абсолютно все детали алгоритма шифрования, значит это тем более нельзя сделать, не обладая подобными знаниями во всей их полноте.

Известны 4 основных типа криптоаналитических атак. При рассмотрении каждой из них подразумевается, что криптоаналитик в курсе всех деталей подвергаемого криптоанализу алгоритма шифрования.

1. *Атака со знанием только шифртекста.* В распоряжении криптоаналитика имеются несколько сообщений, которые были зашифрованы с использованием одного и того же алгоритма шифрования. Задача криптоаналитика состоит в нахождении открытого текста наибольшего числа перехваченных сообщений. Он может также попытаться найти ключи, которые применялись для шифрования этих сообщений, чтобы потом прочесть другие сообщения, зашифрованные с использованием тех же ключей.

   Дано:

   $$C_1 = E_{K_1}(P_1), \quad C_2 = E_{K_2}(P_2), \ldots, \quad C_i = E_{K_i}(P_i).$$

   Найти:

   $P_1, P_2, \ldots, P_i$ или $K_1, K_2, \ldots, K_i$.

2. *Атака со знанием открытого текста.* Криптоаналитик имеет доступ не только к шифрованным текстам нескольких сообщений, но и знает их открытые тексты. От него требуется найти ключи, которые использовались для шифрования этих сообщений.

   Дано:

   $$P_1, C_1 = E_{K_1}(P_1), \quad P_2, C_2 = E_{K_2}(P_2), \ldots, \quad P_i, C_i = E_{K_i}(P_i).$$

   Найти:

   $K_1, K_2, \ldots, K_i$.

3. *Атака с выбранным открытым текстом.* Криптоаналитик не только знает шифрованные и открытые тексты нескольких сообщений, но и может определять содержание этих сообщений. Данная разновидность криптоаналитической атаки является более мощной по сравнению с атакой со знанием открытого текста, поскольку здесь криптоаналитик может по своему усмотрению выбирать открытый текст, подлежащий зашифрованию, и, тем самым, получать больше информации об используемых ключах. Его задача по-прежнему состоит в нахождении ключей.

Дано:

$P_1, C_1 = E_{K_1}(P_1)$, $P_2, C_2 = E_{K_2}(P_2)$,..., P_i,

где $C_i = E_{K_i}(P_i)$, где $P_1, P_2, \ldots, P_i$ выбраны криптоаналитиком.

Найти:

$K_1, K_2, \ldots, K_i$.

4. *Адаптивная атака с выбранным открытым текстом.* Эта атака является разновидностью атаки с выбранным открытым текстом. Криптоаналитик не только выбирает открытые тексты посылаемых шифрованных сообщений, но и может менять свой выбор в зависимости от результатов их шифрования.

Имеются по крайней мере еще 3 разновидности криптоаналитических атак.

1. *Атака с выбранным шифртекстом.* Криптоаналитику предоставлена возможность выбора шифртекстов, подлежащих расшифрованию получателем. Он также имеет доступ к соответствующим открытым текстам. Требуется найти ключи.

   Дано:

   $C_1, P_1 = D_{K_1}(C_1)$, $C_2, P_2 = D_{K_2}(C_2)$,..., $C_i, P_i = D_{K_i}(C_i)$.

   Найти :

   $K_1, K_2, \ldots, K_i$. Этой криптоаналитической атаке, как правило, подвергаются алгоритмы шифрования с открытым ключом. Хотя иногда она эффективна и против симметричных криптосистем. Атаку с выбранным открытым текстом и с выбранным шифртектом называют атакой с выбранным текстом.

2. *Атака с выбранным ключом.* В ходе этой атаки криптоаналитик обладает некоторыми знаниями относительно правил, по которым отправитель и получатель сообщений выбирают ключи шифрования.

3. *Атака с применением грубой силы.* Криптоаналитик занимается подкупом, шантажом или пытками, чтобы получить сведения, необходимые ему для взлома криптосистемы. Подкуп иногда выделяют в отдельную категорию и называют атакой с покупкой ключа. Эти атаки являются очень эффективными и зачастую предоставляют наиболее легкий путь для получения доступа к открытым текстам шифрованных сообщений.

Атаки со знанием открытого текста и с выбранным открытым текстом не так уж редко встречаются на практике, как можно подумать. Известны случаи, когда криптоаналитику удавалось подкупить шифровальщика, чтобы он зашифровал сообщение, открытый текст которого известен криптоаналитику. Иногда даже не требуется никого подкупать, поскольку открытые тексты многих сообщений начинаются и заканчиваются стандартными фразами.

С этой точки зрения зашифрованная программа на языке C особенно уязвима, поскольку содержит множество зарезервированных слов типа `#define`, `#include`, `if`, `then` и `do`.

Не следует забывать и о правиле Керкхоффа. Попытка добиться высокой надежности криптографического алгоритма за счет сохранения в тайне принципов его работы является малопродуктивной. Криптоаналитик может выполнить дизассемблирование любой сверхсложной программы шифрования и методом обратного проектирования воспроизвести алгоритм, положенный в основу ее функционирования. Такое случается довольно часто. Лучшие алгоритмы шифрования являются общественным достоянием уже в течение многих лет, и над их взломом продолжают безуспешно трудиться самые способные криптоаналитики в мире.

Надежность алгоритма шифрования

Различные криптографические алгоритмы обладают разной надежностью, чаще называемой *стойкостью алгоритма шифрования* или *стойкостью шифра*. Стойкость зависит от того, насколько легко криптоаналитик может взломать шифр. Если при этом стоимость затрат превышает ценность полученной в результате информации, то владельцу этого шифра, возможно, и беспокоиться не о чем. Если время, потраченное на взлом шифра, больше, чем период, в течение которого ваши данные должны храниться в секрете, то они вероятно вне опасности. Если противник не накопил достаточного количества ваших сообщений, зашифрованных с помощью одного ключа, чтобы суметь определить этот ключ, время его менять, может быть, еще и не пришло.

Слова "может быть", "вероятно" и "возможно" употреблены здесь не зря. Ведь всегда существует шанс, что в криптоанализе произойдут революционные изменения. Свести к минимуму вредные последствия очередного такого прорыва поможет соблюдение простого правила: ценность секретных данных должна быть всегда ниже, чем стоимость преодоления защитных средств, используемых для сохранения этих данных в тайне.

Под вскрытием (взломом) шифра обычно понимается решение одной из перечисленных ниже задач:

- *Полное вскрытие.* Криптоаналитик нашел ключ K такой, что $D_K(C)=P$.
- Глобальная дедукция. Не зная K, криптоаналитик отыскал альтернативный D_K алгоритм A такой, что $A(C) = P$.
- *Локальная дедукция.* Криптоаналитику удалось определить открытый текст, соответствующий конкретному перехваченному шифртексту.
- *Частичная дедукция.* Криптоаналитик получил неполную информацию о ключе или открытом тексте. Это могут быть несколько битов ключа, или дополнительные данные о структуре открытого текста, или что-то еще в том же духе.

Криптографический алгоритм называется безусловно стойким, если вне зависимости от того каким объемом перехваченного шифртекста располагает криптоаналитик, у него нет достаточной информации, чтобы восстановить исходный открытый текст. Существует всего один безусловно стойкий шифр (о нем речь пойдет ниже). Все остальные шифры можно вскрыть с помощью атаки со знанием только шифртекста: достаточно перебрать все возможные ключи и проверить, имеет ли смысл открытый текст, полученный с их помощью.

Сложность криптоаналитической атаки

Сложность криптоаналитической атаки на алгоритм шифрования может быть охарактеризована с помощью трех величин:

- *Сложность по данным.* Количество входных данных, необходимое для успешной криптоаналитической атаки на алгоритм шифрования.
- *Вычислительная сложность.* Время, требуемое для успешной криптоаналитической атаки на алгоритм шифрования.
- *Сложность по памяти.* Объем памяти, которая нужна для успешной криптоаналитической атаки на алгоритм шифрования.

Часто под сложностью криптоаналитической атаки понимается максимальная среди этих величин. А для некоторых атак приходится искать компромисс между сложностью по данным, вычислительной сложностью и сложностью по памяти. Например, для реализации более быстрой атаки может потребоваться дополнительная память.

Сложность криптоаналитической атаки, как правило, выражается в виде экспоненциальной функции. К примеру, если атака имеет сложность 2^{128}, то это значит, что для взлома шифра требуется выполнить 2^{128} операций.

При оценке сложности атаки часто приходится оперировать очень большими числами. Чтобы было понятно, насколько они велики, в табл. 5.1 для них приведены некоторые физические аналогии.

Таблица 5.1. *Физические аналогии для очень больших чисел*

Физическая аналогия	Число
Время, оставшееся до наступления следующего ледникового периода	$16 \cdot 10^3$ (2^{14}) лет
Время, оставшееся до превращения Солнца в новую звезду	10^9 (2^{30}) лет
Возраст Земли	10^9 (2^{30}) лет
Возраст Вселенной	10^{10} (2^{32}) лет
Количество атомов, из которых состоит Земля	10^{51} (2^{170})

Таблица 5.1 (окончание)

Физическая аналогия	Число
Количество атомов, из которых состоит Солнце	10^{57} (2^{190})
Количество атомов, из которых состоит наша Галактика	10^{67} (2^{223})
Количество атомов, из которых состоит Вселенная	10^{77} (2^{265})
Объем Вселенной	10^{84} (2^{280}) см^3

В то время как сложность атаки на данный алгоритм шифрования является постоянной величиной (по крайней мере, до тех пор, пока криптоаналитик не придумает более эффективный метод взлома), вычислительная мощь современных компьютеров растет буквально не по дням, а по часам. Такой феноменальный рост наблюдается в течение последних пятидесяти лет, и есть все основания полагать, что в ближайшее время данная тенденция сохранится. Большинство криптоаналитических атак идеально подходят для реализации на параллельных компьютерах: полномасштабная атака на шифр разбивается на миллионы крошечных атак, которые ведутся независимо друг от друга и, следовательно, не требуют организации взаимодействия между процессорами. Поэтому в корне неверно заявлять о достаточной вычислительной стойкости алгоритма шифрования только потому, что его невозможно взломать при современном уровне развития технологии. Хорошая криптосистема всегда проектируется с достаточным запасом на будущее. При этом необходимо принимать во внимание прогнозируемый рост компьютерной производительности, чтобы алгоритм шифрования оставался вычислительно стойким в течение многих лет.

Шифры замены и перестановки

Шифры появились на свет задолго до изобретения компьютера. Получившие широкое распространение криптографические алгоритмы выполняли либо замену одних букв на другие, либо переставляли буквы друг с другом. Самые стойкие шифры делали одновременно и то, и другое, причем многократно.

Шифры замены

Шифром замены называется алгоритм шифрования, который производит замену каждой буквы открытого текста на какой-то символ шифрованного текста. Получатель сообщения расшифровывает его путем обратной замены.

В классической криптографии различают 4 разновидности шифров замены:

- *Простая замена*, или *одноалфавитный шифр*. Каждая буква открытого текста заменяется на один и тот же символ шифртекста.

- *Омофонная замена.* Аналогична простой замене с единственным отличием: каждой букве открытого текста ставятся в соответствие несколько символов шифртекста. Например, буква "А" заменяется на цифру 5, 13, 25 или 57, а буква "Б" — на 7, 19, 31 или 43 и так далее.
- *Блочная замена.* Шифрование открытого текста производится блоками. Например, блоку "АБА" может соответствовать "РТК", а блоку "АББ" — "СЛЛ".
- Многоалфавитная замена. Состоит из нескольких шифров простой замены. Например, могут использоваться пять шифров простой замены, а какой из них конкретно применяется для шифрования данной буквы открытого текста, — зависит от ее положения в тексте.

Примером шифра простой замены может служить программа ROT13, которую обычно можно найти в операционной системе UNIX. С ее помощью буква "A" открытого текста на английском языке заменяется на букву "N", "B" — на "O" и так далее. Таким образом, ROT13 циклически сдвигает каждую букву английского алфавита на 13 позиций вправо. Чтобы получить исходный открытый текст надо применить функцию шифрования ROT13 дважды:

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

Все упомянутые шифры замены легко взламываются с использованием современных компьютеров, поскольку замена недостаточно хорошо маскирует стандартные частоты встречаемости букв в открытом тексте.

Разновидностью шифра замены можно считать код, который вместо букв осуществляет замену слов, фраз и даже целых предложений. Например, кодовый текст "ЛЕДЕНЕЦ" может соответствовать фразе открытого текста "ПОВЕРНУТЬ ВПРАВО НА 90°". Однако коды применимы только при определенных условиях: если, например, в коде отсутствует соответствующее значение для слова "МУРАВЬЕД", то вы не можете использовать это слово в открытом тексте своего сообщения, предназначенном для кодирования.

Шифры перестановки

В *шифре перестановки* буквы открытого текста не замещаются на другие, а меняется сам порядок их следования. Например, в шифре простой колонной перестановки исходный открытый текст записывается построчно (число букв в строке фиксировано), а шифртекст получается считыванием букв по колонкам. Расшифрование производится аналогично: шифртекст записывается поколонно, а открытый текст можно затем прочесть по горизонтали.

Для повышения стойкости полученный шифртекст можно подать на вход второго шифра перестановки. Существуют еще более сложные шифры перестановки, однако почти все они легко взламываются с помощью компьютера.

Хотя во многих современных криптографических алгоритмах и используется перестановка, ее применение ограничено узкими рамками, поскольку в этом

случае требуется память большого объема, а также накладываются ограничения на длину шифруемых сообщений. Замена получила значительно большее распространение.

Роторные машины

В 20-е годы были изобретены разнообразные механические устройства, призванные автоматизировать процесс шифрования и расшифрования. Большинство из них состояло из клавиатуры для ввода открытого текста и набора *роторов* — специальных вращающихся колес, каждое из которых реализовывало простую замену. Например, ротор мог заменять "А" на "Ф", "Б" на "У", "С" на "Л" и т. д. При этом выходные контакты одного ротора подсоединялись к входным контактам следующего за ним.

Тогда, например, если на клавиатуре 4-роторной машины нажималась клавиша "А", то первый ротор мог превратить ее в "Ф", которая, пройдя через второй ротор, могла стать буквой "Т", которую третий ротор мог заменить на букву "К", которая могла быть преобразована четвертым ротором в букву "Е" шифртекста. После этого роторы поворачивались, и в следующий раз замена была иной. Чтобы сбить с толку криптоаналитиков, роторы вращались с разной скоростью.

Наиболее известной роторной машиной стала немецкая "Энигма", которую Германия использовала для засекречивания своей переписки во время Второй мировой войны.

Операция сложения по модулю 2

Операция сложения по модулю 2, которая в языке программирования C обозначается знаком ^, а в математике — знаком ⊕, представляет собой стандартную операцию над битами:

```
0 ⊕ 0 = 0
0 ⊕ 1 = 1
1 ⊕ 0 = 1
1 ⊕ 1 = 0
```

С помощью сложения по модулю 2 можно выполнять многоалфавитную замену, прибавляя к битам ключа соответствующие биты открытого текста. Этот алгоритм шифрования является симметричным. Поскольку двойное прибавление одной и той же величины по модулю 2 восстанавливает исходное значение, шифрование и расшифрование выполняются одной и той же программой.

К сожалению, данный алгоритм обладает очень слабой стойкостью. Тем не менее АНБ одобрило его использование в цифровых сотовых телефонах американских производителей для засекречивания речевых переговоров. Он также часто встречается в различных коммерческих программных продуктах.

Следует помнить, что с помощью данного алгоритма вы можете надежно спрятать от человека, не сведущего в криптоанализе, содержание своей переписки, однако опытному криптоаналитику понадобится всего несколько минут, чтобы его взломать. Делается это очень просто, если предположить, например, что открытый текст сообщения написан на английском языке.

1. Сначала следует определить длину ключа. Шифртекст последовательно складывается по модулю 2 со своей копией, сдвинутой на различное число байт, и в полученном векторе подсчитывается число совпадающих компонентов. Когда величина сдвига кратна длине ключа, это число превысит 6% от общей длины исследуемого шифртекста. Если не кратна, то совпадений будет меньше 0,4%. Проанализировав полученные данные, можно сделать обоснованный вывод о длине ключа.
2. Затем надо сложить шифртекст по модулю 2 со своей копией, сдвинутой на величину длины ключа. Эта операция аннулирует ключ и оставит в наличии открытый текст сообщения, сложенный по модулю 2 со своей копией, сдвинутой на величину длины ключа.

Одноразовые блокноты

Хотите верьте, хотите нет, но на самом деле все-таки существует алгоритм шифрования, который невозможно вскрыть. Зовется он *одноразовым блокнотом*. В классическом виде одноразовый блокнот представляет собой очень длинную последовательность случайных букв, записанную на листах бумаги, которые скреплены между собой в блокнот. Отправитель использует каждую букву из блокнота, чтобы зашифровать ровно одну букву открытого текста сообщения. Шифрование состоит в сложении буквы открытого текста и буквы из одноразового блокнота по модулю N, где N — количество букв в алфавите. После зашифрования отправитель уничтожает использованный одноразовый блокнот. Чтобы отправить новое сообщение, ему придется изготовить или найти новый одноразовый блокнот.

Получатель, владеющий копией одноразового блокнота, которым воспользовался отправитель сообщения, получает открытый текст путем сложения букв шифртекста и букв, извлеченных из имеющейся у него копии одноразового блокнота. Эту копию он затем уничтожает.

Если предположить, что у криптоаналитика нет доступа к одноразовому блокноту, данный алгоритм шифрования абсолютно надежен. Перехваченному шифрованному сообщению с одинаковой вероятностью соответствует произвольный открытый текст той же длины, что и сообщение.

Однако у алгоритма шифрования с помощью одноразового блокнота есть весьма существенный недостаток. Последовательность букв, которая содержится в одноразовом блокноте, должна быть по-настоящему случайной, а не просто псевдослучайной, поскольку любая криптоаналитическая атака на

него будет, в первую очередь, направлена против метода генерации содержимого этого блокнота.

Другая важная особенность применения одноразового блокнота состоит в том, чтобы никогда не пользоваться им дважды, поскольку криптоаналитик может отыскать участки сообщений, для шифрования которых был применен один и тот же одноразовый блокнот. Делается это следующим образом. Нужно последовательно сдвигать одно сообщение относительно другого и подсчитывать количество совпадений. Как только это количество резко увеличится, значит, случайная последовательность, использованная для зашифрования двух различных отрезков сообщений, была одной и той же. Дальнейший криптоанализ осуществляется достаточно просто.

Одноразовые блокноты могут состоять не из байтов, а из битов. Тогда шифрование будет заключаться в выполнении сложения по модулю 2. Для получения открытого текста достаточно опять сложить по модулю 2 шифртекст и содержимое одноразового блокнота. Надежность будет по-прежнему такой же, как при посимвольном модульном сложении.

Еще один недостаток блокнотного способа шифрования заключается в том, что случайная последовательность должна быть той же длины, что и само сообщение. Чтобы послать короткую шифровку резиденту, одноразовый блокнот еще сгодится, но как быть с каналом связи, пропускная способность которого измеряется десятками мегабит в секунду?

Конечно, при необходимости можно для хранения случайных последовательностей воспользоваться компакт-дисками с многократной перезаписью или цифровой магнитной лентой. Однако это будет достаточно дорогостоящее решение проблемы. Кроме того, необходимо будет обеспечить синхронную работу приемной и передающей аппаратуры, а также отсутствие искажений при передаче. Ведь даже если несколько бит сообщения пропущены при его передаче, получатель так и не сможет прочесть открытый текст.

Несмотря на все перечисленные выше недостатки, в настоящее время одноразовые блокноты активно используются для шифрования сверхсекретных сообщений. Не имея в своем распоряжении соответствующего блокнота, эти сообщения невозможно прочитать вне зависимости от того, насколько быстро работают суперкомпьютеры, которые используются в ходе криптоаналитической атаки. Даже инопланетные пришельцы со своими сверхсветовыми звездолетами и чудо-компьютерами не смогут их прочесть, если, конечно, не вернутся в прошлое на машине времени и не добудут одноразовые блокноты, использованные для шифрования этих сообщений.

Компьютерные алгоритмы шифрования

Существует великое множество алгоритмов шифрования, придуманных специально в расчете на реализацию в виде компьютерных программ. Среди наиболее известных можно упомянуть:

- Data Encryption Standard (DES). Симметричный алгоритм шифрования, являющийся в США государственным стандартом.
- RSA. Алгоритм шифрования с открытым ключом, названный по первым буквам фамилий его создателей (Rivest, Shamir, Adleman).
- ГОСТ 28147-89. Симметричный алгоритм шифрования, одобренный сначала в СССР, а затем и в России для использования в качестве государственного стандарта.

Глава 6

Криптографические ключи

Длина секретного ключа

Надежность симметричной криптосистемы зависит от стойкости используемого криптографического алгоритма и от длины секретного ключа. Допустим, что сам алгоритм идеален — вскрыть его можно только путем опробования всех возможных ключей. Этот вид криптоаналитической атаки называется методом *тотального перебора*. Чтобы применить данный метод, криптоаналитику понадобится немного шифртекста и соответствующий открытый текст. Например, в случае блочного шифра ему достаточно получить в свое распоряжение по одному блоку шифрованного и соответствующего открытого текста. Сделать это не так уж и трудно.

Криптоаналитик может заранее узнать содержание сообщения, а затем перехватить его при передаче в зашифрованном виде. По некоторым признакам он также может догадаться, что посланное сообщение представляет собой не что иное, как текстовый файл, подготовленный с помощью распространенного редактора, компьютерное изображение в стандартном формате, каталог файловой подсистемы или базу данных. Для криптоаналитика важно то, что в каждом из этих случаев в открытом тексте перехваченного шифрсообщения известны несколько байтов, которых ему хватит, чтобы предпринять атаку со знанием открытого текста.

Подсчитать сложность атаки методом тотального перебора достаточно просто. Если ключ имеет длину 64 бита, то суперкомпьютер, который может опробовать 1 млн ключей за 1 с, потратит более 5 тыс. лет на проверку всех возможных ключей. При увеличении длины ключа до 128 бит, этому же суперкомпьютеру понадобится 10^{25} лет, чтобы перебрать все ключи. Вселенная существует всего-навсего 10^{10} лет, поэтому можно сказать, что 10^{25} — это достаточно большой запас надежности для тех, кто пользуется 128-битными ключами.

Однако прежде чем броситься спешно изобретать криптосистему с длиной ключа в 4 Кбайт, следует вспомнить о сделанном выше предположении, а именно: используемый алгоритм шифрования идеален в том смысле, что вскрыть его можно только методом тотального перебора. Убедиться в этом на практике бывает не так просто, как может показаться на первый взгляд. Криптография требует утонченности и терпения. Новые сверхсложные криптосистемы при более внимательном рассмотрении зачастую оказываются очень нестойкими. А внесение даже крошечных изменений в стойкий криптографический алгоритм может существенно понизить его стойкость. Поэтому надо пользоваться только проверенными шифрами, которые известны уже в течение многих лет, и не бояться проявлять болезненную подозрительность по отношению к новейшим алгоритмам шифрования, вне зависимости от заявлений их авторов об абсолютной надежности этих алгоритмов.

Важно также не забывать о правиле Керкхоффа: стойкость алгоритма шифрования должна определяться ключом, а не деталями самого алгоритма. Чтобы быть уверенным в стойкости используемого шифра, недостаточно проанализировать его при условии, что противник досконально знаком с алгоритмом шифрования. Нужно еще и рассмотреть атаку на этот алгоритм, при которой враг может получить любое количество шифрованного и соответствующего открытого текста. Более того, для повышения надежности следует предположить, что криптоаналитик имеет возможность организовать атаку с выбранным открытым текстом произвольной длины.

К счастью, в реальной жизни большинство людей, интересующихся содержанием ваших шифрованных файлов, не обладают квалификацией высококлассных специалистов и вычислительными ресурсами, которые имеются в распоряжении правительств мировых супердержав. Последние же вряд ли будут тратить время и деньги, чтобы прочесть ваше пылкое сугубо личное послание. Однако, если вы планируете свергнуть "антинародное правительство", вам необходимо всерьез задуматься о стойкости применяемого алгоритма шифрования.

Сложность и стоимость атаки методом тотального перебора

Атака методом тотального перебора, как правило, представляет собой разновидность атаки со знанием открытого текста. Если предположить, что атака методом тотального перебора является наиболее эффективной среди возможных атак на используемый вами симметричный алгоритм шифрования, то ключ должен быть достаточно длинным, чтобы успешно отразить эту атаку. Насколько длинным?

Среди параметров, которые необходимо принимать во внимание при рассмотрении атаки методом тотального перебора, прежде всего, надо упомянуть об

общем количестве проверяемых ключей и о времени, затрачиваемом противником на проверку одного ключа. Количество ключей для конкретного алгоритма обычно фиксировано. Например, DES-алгоритм использует 56-битный ключ. Это означает, что его ключевое пространство содержит 2^{56} ключей.

Скорость проверки ключей играет менее важную роль, чем их количество. Для простоты изложения можно считать, что вне зависимости от алгоритма шифрования, время, которое требуется на проверку одного ключа, одинаково. На практике данное предположение неверно, и для разных криптографических алгоритмов это время может различаться в десятки раз. Поскольку нашей целью является отыскание такой длины ключа, при которой стойкость алгоритма шифрования против атаки методом тотального перебора в миллионы раз превышает предел, делающий эту атаку неосуществимой на практике, то сделанное нами предположение вполне оправдано.

При решении вопроса о достаточной длине ключа в качестве алгоритма шифрования чаще всего рассматривается DES-алгоритм. В 1977 г. американские криптологи У. Диффи (W.Diffie) и М. Хеллман (M.Hellman) заявили, что при существующем уровне развития компьютерной технологии можно построить специализированный суперкомпьютер для вскрытия ключей DES-алгоритма методом тотального перебора. Имея в своем составе 1 млн микросхем, каждая из которых способна проверять 1 млн ключей в секунду, этот суперкомпьютер перебрал бы все 2^{56} ключей за 20 час.

Атака методом тотального перебора идеально подходит для реализации на параллельном суперкомпьютере, состоящем из многих процессоров. Отдельным процессорам, ведущим поиск ключа, нет необходимости устанавливать связь с другими процессорами суперкомпьютера во время выполнения своей части поиска. Следовательно, все процессоры специализированного суперкомпьютера, предназначенного для параллельного поиска ключей, необязательно находятся даже в одном городе, не говоря уже об одном помещении.

В 1993 г. американский криптолог М. Винер (M.Wiener) спроектировал суперкомпьютер для атаки на DES-алгоритм методом тотального перебора. Рассуждения Винера верны не только для DES-алгоритма, но и практически для любого другого алгоритма шифрования. Суперкомпьютер, разработанный Винером, состоит из специализированных микросхем, плат и стоек. По мнению Винера, для того чтобы гарантировать вскрытие 56-битного ключа за 7 час, на изготовление такого суперкомпьютера потребуется не более 1 млн долларов. По закону Мура, вычислительная мощь компьютеров удваивается каждые полтора года. Поэтому к 2001 г. стоимость суперкомпьютера, придуманного Винером, уменьшится в 10 раз и составит всего-навсего 100 тыс. долларов. Это означает, что уже сейчас крупные компании и "крутые" криминальные структуры могут вскрывать 56-битные ключи. Для военных криптоаналитиков в большинстве индустриально развитых стран доступны 64-битные ключи.

В 1996 г. Диффи, Винер и другие авторитетные американские криптологи опубликовали результаты своей исследовательской работы по определению длины ключа, необходимой для адекватной защиты информации от атаки методом тотального перебора (табл. 6.1).

Таблица 6.1. *Стоимость и вычислительная сложность атаки методом тотального перебора*

Кто атакует	Бюджет	Сложность атаки		Стойкий ключ
		40 бит	56 бит	
Хакер	1000 долл.	1 неделя	Никогда	45 бит
Малый бизнес	10 тыс. долл.	12 мин.	556 дней	64 бита
Крупная компания	10 млн долл.	0.005 с	6 мин	70 бит
Федеральное агентство	300 млн долл.	0.0002 с	12 с	75 бит

К приведенным в табл. 6.1 цифрам следует относиться с осторожностью. Теоретический расчет затрат на проведение атак методом тотального перебора на криптографические ключи разной длины всегда существенно отличается от того, с чем криптоаналитики сталкиваются на практике при покупке или разработке суперкомпьютеров для ведения такого рода атак. Объясняется это тем, что одни сделанные допущения оказываются весьма далеки от реальности, в то время как другие факторы просто не принимаются во внимание. В данном случае Диффи, Винер и другие посчитали, что при создании специализированного суперкомпьютера для атаки методом тотального перебора будут использоваться заказные микросхемы ценой не более 10 долл. По оценкам АНБ, такие микросхемы стоят, как правило, в 100 раз дороже. У АНБ вызвало сомнение и допущение о том, что вне зависимости от алгоритма шифрования, лишь длина ключа определяет сложность криптоаналитической атаки. Кроме того, при составлении таблицы не были учтены затраты на научно-исследовательские и опытно-конструкторские работы, которые для первого экземпляра суперкомпьютера обычно составляют не менее 10 млн долл. Не были также приняты во внимание расходы на приобретение компьютерной памяти.

Из сказанного можно сделать весьма важный вывод. Если кто-то очень захочет узнать использованный вами ключ, ему нужно всего лишь потратить достаточное количество денег. Поэтому определяющей является стоимость зашифрованной вами информации. Если цена ей в базарный день — около 2 долл., вряд ли кто-то решится потратить 1 млн, чтобы ее заполучить. Но если прибыль от прочтения вашей шифровки составляет 100 млн долл., — берегитесь! Единственным утешением может послужить тот факт, что с течением времени любая информация очень быстро устаревает и теряет свою ценность.

Программная атака

Без специализированного компьютерного оборудования, ведущего параллельный поиск ключей, атака методом тотального перебора имеет значительно меньше шансов на успех. Однако если вы не припасли лишний миллион долларов, который можно потратить на изготовление такого оборудования, есть другой, более дешевый, способ попытаться вскрыть интересующий вас ключ. В мире имеется огромное количество компьютеров (по оценкам экспертов, в 1996 г. их число достигло 200 млн), которые, чтобы не простаивать, могли бы опробовать ключи. Эксперимент, проведенный в начале 1997 г., показал, что таким способом за две недели можно вскрыть 48-битный ключ. И хотя этот ключ был найден методом тотального перебора после проверки чуть более половины всех возможных ключей, полученный результат впечатляет, поскольку в ходе атаки одновременно использовались не более 5 тысяч компьютеров из существующих 200 миллионов, а в общей сложности в атаке оказались задействованными лишь 7 тысяч компьютеров.

Основное препятствие на пути к использованию миллионов вычислительных устройств, разбросанных по всему миру, заключается в невозможности сделать так, чтобы их владельцы приняли участие в атаке. Можно, конечно, вежливо попросить каждого из них об услуге, но во-первых, на это уйдет уйма времени, а во-вторых, ответом в большинстве случаев будет, скорее всего, твердое "нет". Можно попытаться тайком проникнуть на чужие компьютеры через сеть, но на это понадобится еще больше времени, да вдобавок вас могут арестовать.

Более разумным представляется создание компьютерного вируса, который вместо того, чтобы стирать файлы с жесткого диска и выдавать на дисплей глупые сообщения, незаметно для владельца компьютера будет перебирать возможные ключи. Проведенные исследования показывают, что в распоряжении вируса будет от 70 до 90% процессорного времени зараженного им компьютера. После вскрытия ключа вирус может породить новый вирус, содержащий информацию о найденном ключе, и отправить его странствовать по компьютерной сети до тех пор, пока он не доберется до своего хозяина.

При более тонком подходе вирус, обнаруживший ключ, выдаст на экран компьютера информацию вида:

В ВАШЕМ КОМПЬЮТЕРЕ ОБНАРУЖЕНА СЕРЬЕЗНАЯ ОШИБКА!!!
ПОЖАЛУЙСТА, ПОЗВОНИТЕ ПО ТЕЛЕФОНУ (095)123-45-67
И ЗАЧИТАЙТЕ ОПЕРАТОРУ СЛЕДУЮЩЕЕ 48-БИТОВОЕ ЧИСЛО:
XXXXXXXX XXXXXXX XXXXXXX XXXXXXX XXXXXXX XXXXXXX
ПЕРВОМУ, КТО СООБЩИТ ОБ ЭТОЙ ОШИБКЕ, ГАРАНТИРОВАНО
ВОЗНАГРАЖДЕНИЕ В РАЗМЕРЕ 100 (СТА) ДОЛЛАРОВ.

Если вирусу удастся заразить 10 млн компьютеров, каждый из которых станет проверять хотя бы 1 тыс. ключей в секунду, то 56-битный ключ будет найден менее чем через 3 месяца. Дополнительно придется раскошелиться на подкуп производителей антивирусных программ, однако к компьютерной криптографии, о которой сейчас идет речь, эта проблема никакого отношения не имеет.

"Китайская лотерея"

Допустим, что для атаки методом тотального перебора во всякий без исключения китайский радиоприемник и телевизор встраивается специальная микросхема, проверяющая 1 млн ключей в секунду. Каждая из них автоматически перебирает свое подмножество ключей после получения из эфира фрагментов шифрованного и соответствующего открытого текста. Как только правительство Китая пожелает вскрыть какой-нибудь ключ, оно принимает постановление, которое обязывает всех владельцев телевизоров и радиоприемников включить свои аппараты в определенное время, чтобы они могли принять пару фрагментов текста и приступить к перебору ключей.

За найденный ключ полагается значительный приз. Благодаря этому радиоприемники и телевизоры со встроенными микросхемами хорошо раскупаются, а вскрытые ключи своевременно доводятся до сведения китайского правительства. Если учесть, что у каждого из десяти китайцев есть радиоприемник или телевизор, то получится, что на вскрытие 64-битного ключа китайскому правительству потребуется самое большее 43 часа. В табл. 6.2 приведена сложность вскрытия 64-битного ключа с помощью "китайской лотереи" при ее проведении в Китае, а также в США, Ираке и Израиле.

Таблица 6.2. *Сложность вскрытия 64-битного ключа с помощью "китайской лотереи"*

Страна	Численность населения	Количество телевизоров и радиоприемников	Сложность вскрытия 64-битового ключа с помощью "китайской лотереи"
Китай	1 190 431 000	257 000 000	43 часа
США	260 714 000	739 000 000	6,9 часов
Ирак	19 890 000	4 730 000	44 дня
Израиль	5 051 000	3 640 000	58 дней

Биотехнология

Предположим такую ситуацию: с использованием генной инженерии удалось вывести особую породу травоядных динозавров. Их нарекли "криптозаврами", поскольку они состоят из клеток, которые могут проверять крип-

тографические ключи. Шифрованный и соответствующий открытый текст передаются в клетки криптозавра с помощью магнитооптических средств. Найденный ключ выводится специальными клетками, способными перемещаться в пределах тела криптозавра. Поскольку динозавр в среднем состоит из 2^{14} клеток, то при условии, что каждая из них успевает проверить 1 миллион ключей в секунду, на поиск 64-битного ключа уйдет не более 1 с.

Для этой же цели можно использовать клетки морских водорослей. Например, на площади в 500 км^2 и на глубине в 1 м их можно разместить в количестве 10^{23}. Если каждая из них будет обрабатывать 1 млн ключей в секунду, то 128-битный ключ будет вскрыт менее чем за сотню лет. Шифрованный и открытый тексты передаются водорослям с помощью спутника, а сигналом о нахождении искомого ключа служит изменение окраски водорослей вокруг клетки, отыскавшей этот ключ. За советами по разведению морских водорослей криптоаналитики могут обратиться на биологический факультет ближайшего университета.

Термодинамические ограничения

Из второго закона термодинамики следует, что для записи одного бита информации путем соответствующего изменения состояния среды требуется не менее kT эрг энергии, где k — постоянная Больцмана и T — температура системы. Учитывая, что температура Вселенной составляет 3,2° по Кельвину, для изменения одного бита информации компьютеру понадобится не менее $4{,}4 \cdot 10^{-16}$ эрг энергии. Уменьшение температуры среды ниже уровня температуры Вселенной потребует дополнительных затрат энергии и поэтому не имеет особого смысла.

За год вся энергия, излучаемая Солнцем, составляет $1{,}21 \cdot 10^{-41}$ эрг. Ее хватит, чтобы произвести $2{,}7 \cdot 10^{56}$ записей одного бита информации, что эквивалентно прогону 187-битного счетчика через все возможные состояния. Если наше довольно "прохладное" Солнце заменить сверхновой звездой, то счетчик можно сделать 219-битным.

Сказанное означает, что атака методом тотального перебора против криптосистемы с 256-битным ключом энергетически неосуществима до тех пор, пока компьютеры являются объектами материального мира и их функционирование подчиняется его законам, в частности — второму закону термодинамики.

Однонаправленные функции

Понятие *однонаправленной функции* является основным в криптографии с открытым ключом. К однонаправленным относят такие функции, которые достаточно легко вычислить, но значительно труднее обратить. То есть, при

наличии x нетрудно определить f(x), однако, при условии знания только f(x) на нахождение соответствующего значения x уйдут миллионы лет вычислений на всех компьютерах, которые только есть в мире.

Аналогом однонаправленной функции в быту является разбитая вдребезги стеклянная бутылка. Расколотить ее на мелкие осколки очень легко, однако попробуйте снова собрать целую бутылку из осколков!

Строгое математическое доказательство существования однонаправленных функций, а также правила их построения пока не придуманы. Тем не менее, существует множество функций, которые все считают однонаправленными: их значения довольно эффективно вычисляются, однако обратить эти функции каким-либо простым методом не удается. Хорошим примером может служить вычисление функции x^2 в конечных полях.

Какой прок от однонаправленных функций в криптографии? Ведь если с ее помощью зашифровать сообщение, прочесть его не сможет никто. Вернемся к аналогии с бутылкой. Напишите на ней открытый текст, разбейте ее вдребезги и дайте осколки своему приятелю, чтобы он прочитал написанный вами текст. И не забудьте упомянуть про однонаправленные функции, чтобы произвести на него должное впечатление вашими глубокими познаниями в криптографии! К сожалению, дальнейший путь этих осколков лежит только в мусорное ведро, ибо в таком виде ваше послание не примет ни одно почтовое отделение.

Поэтому в криптографии большим спросом пользуются однонаправленные функции с лазейкой, которые представляют собой особую разновидность однонаправленных функций. Однонаправленную функцию с лазейкой попрежнему трудно обратить, но только не зная секрета вычисления обратной к ней функции. То есть, при данном x легко найти f(x) и наоборот — трудно отыскать x, зная одно лишь значение f(x). Однако существует такая секретная информация (y), что если известны y и f(x), то вычислить x будет значительно проще.

Хорошим аналогом однонаправленной функции с лазейкой служат обыкновенные часы. Их очень легко разобрать на большое количество мельчайших деталей, из которых потом будет весьма трудно снова собрать работающий часовой механизм. Однако при наличии инструкции по сборке часов сделать это не так и сложно.

Особый интерес для криптографов представляют однонаправленные хэш-функции. Алгоритмы хэширования, реализуемые с помощью хэш-функций, позволяют преобразовывать строки переменной длины, называемые *образами*, в строки фиксированной длины, которые принято именовать *хэш-значениями*. Обычно хэш-значение гораздо меньше любого из образов. Примером простейшей хэш-функции является преобразование байтовой строки в хэш-значение, равное одному байту, который получается сложением всех байтов этой строки по модулю 2. Однако такая хэш-функция не является

однонаправленной: нетрудно подобрать строку символов, суммирование которых по модулю 2 даст заранее заданное значение.

Однонаправленная хэш-функция позволяет легко сгенерировать хэш-значение. Однако, зная только его, будет очень трудно подобрать соответствующий ему образ. Качественная однонаправленная хэш-функция чаще всего является непротиворечивой: весьма сложно получить два различных образа, для которых хэш-значение будет одним и тем же.

Процесс хэширования в криптографии — не тайна. Однонаправленная хэш-функция обеспечивает необходимый уровень защиты благодаря своей однонаправленности. По выходу такой хэш-функции невозможно сказать, что было подано на ее вход, а изменение даже одного бита образа приводит к смене в среднем половины бит соответствующего хэш-значения.

Длина открытого ключа

Многие современные алгоритмы шифрования с открытым ключом основаны на однонаправленности функции разложения на множители числа, являющегося произведением двух больших простых чисел. Эти алгоритмы также могут быть подвергнуты атаке, подобной методу тотального перебора, применяемому против шифров с секретным ключом, с одним лишь отличием: опробовать каждый ключ не потребуется, достаточно суметь разложить на множители большое число.

Конечно, разложение большого числа на множители — задача трудная. Однако сразу возникает резонный вопрос, насколько трудная. К несчастью для криптографов, сложность ее решения уменьшается. И что еще хуже, эта сложность падает значительно более быстрыми темпами, чем ожидалось ранее. Например, в середине 70-х годов считалось, что для разложения на множители числа из 125 цифр потребуются десятки квадрильонов лет. А всего два десятилетия спустя с помощью компьютеров, подключенных к сети Internet, удалось разложить на множители число из 129 цифр. Этот прорыв стал возможен благодаря тому, что за прошедшие 20 лет были не только предложены новые, более быстрые, методы разложения на множители больших чисел, но и возросла производительность используемых компьютеров.

Поэтому квалифицированный криптограф должен проявлять очень большую осторожность и осмотрительность, когда речь заходит о длине открытого ключа. Необходимо учитывать, насколько ценна засекречиваемая с его помощью информация и как долго она должна оставаться в тайне для посторонних.

А почему, спрашивается, не взять 10000-битный ключ? Ведь тогда отпадут все вопросы, связанные со стойкостью асимметричного алгоритма шифрования с открытым ключом, основанном на разложении большого числа на множители. Но дело в том, что обеспечение достаточной стойкости шифра

не является единственной заботой криптографа. Имеются дополнительные соображения, влияющие на выбор длины ключа, и среди них — вопросы, связанные с практической реализуемостью алгоритма шифрования при выбранной длине ключа.

Чтобы оценить длину открытого ключа, будем измерять доступную криптоаналитику вычислительную мощь в так называемых *мопс-годах*, т. е. количеством операций, которые компьютер, способный работать со скоростью 1 миллион операций в секунду, выполняет за год. Допустим, что хакер имеет доступ к компьютерным ресурсам общей вычислительной мощью 10000 мопс-лет, крупная корпорация — 10^7 мопс-лет, правительство — 10^9 мопс-лет. Это вполне реальные цифры, если учесть, что при реализации упомянутого выше проекта разложения числа из 129 цифр его участники задействовали всего 0,03% вычислительной мощи Internet, и чтобы добиться этого, им не потребовалось принимать какие-либо экстраординарные меры или выходить за рамки закона.

Предположим еще, что вычислительная мощь возрастает в 10 раз каждые 5 лет, а метод, который используется для разложения больших чисел на множители, позволяет это делать с трудоемкостью, указанной в табл. 6.3.

Таблица 6.3. *Трудоемкость разложения больших чисел на множители*

Количество бит в двоичном представлении числа	Количество мопс-лет для разложения на множители
768	$3 \cdot 10^5$
1024	$3 \cdot 10^7$
1280	$3 \cdot 10^9$
1536	$3 \cdot 10^{11}$
2048	$3 \cdot 10^{14}$

Сделанные предположения позволяют оценить длину стойкого открытого ключа в зависимости от срока, в течение которого необходимо хранить зашифрованные с его помощью данные в секрете (табл. 6.4). При этом необходимо помнить, что криптографические алгоритмы с открытым ключом часто применяются для защиты очень ценной информации на весьма долгий период времени. Например, в системах электронных платежей или при нотариальном заверении электронной подписи. Идея потратить несколько месяцев на разложение большого числа на множители может показаться кому-то очень привлекательной, если в результате он получит возможность рассчитываться за свои покупки по вашей кредитной карточке. Кроме того, я думаю, что вам совсем не улыбается перспектива быть вызванным через

20 лет на заседание суда, на котором рассматривается дело о наследстве, и отстаивать невозможность подделать электронную подпись вашего дедушки, использованную им для составления завещания в вашу пользу.

Таблица 6.4. *Рекомендуемая длина открытого ключа (в битах)*

Год	Хакер	Крупная корпорация	Правительство
2000	1024	1280	1536
2005	1280	1536	2048
2010	1280	1536	2048
2015	1536	2048	2048

С приведенными в табл. 6.4 данными согласны далеко не все авторитетные криптографы. Некоторые из них наотрез отказываются делать какие-либо долгосрочные прогнозы, считая это бесполезным делом. Другие, например, специалисты из АНБ, чересчур оптимистичны, рекомендуя для систем цифровой подписи длину открытого ключа всего 512—1024 бита, что в свете данных из табл. 6.4 является совершенно недостаточным для обеспечения надлежащей долговременной защиты.

Какой длины должен быть ключ

Криптоаналитическая атака против алгоритма шифрования обычно своим острием бывает направлена в самое уязвимое место этого алгоритма. Например, для организации шифрованной связи часто используются криптографические алгоритмы как с секретным, так и с открытым ключом. Такая криптосистема называется *гибридной*. Стойкость каждого из алгоритмов, входящих в состав гибридной криптосистемы, должна быть достаточной, чтобы успешно противостоять вскрытию. Например, глупо применять симметричный алгоритм с ключом длиной 128 бит совместно с асимметричным алгоритмом, в котором длина ключа составляет всего 386 бит. И наоборот, не имеет смысла задействовать симметричный алгоритм с ключом длиной 56 бит вместе с асимметричным алгоритмом с ключом длиной 1024 бита.

В табл. 6.5 перечисляются пары длин ключей для симметричного и асимметричного криптографического алгоритма, при которых стойкость обоих алгоритмов против криптоаналитической атаки методом тотального перебора приблизительно одинакова. Из данных, приведенных в табл. 6.5, например, следует, что если используется симметричный алгоритм со 112-битным ключом, то вместе с ним должен применяться асимметричный алгоритм с 1792-битным ключом. Однако на практике ключ для асимметричного алгоритма шифрования обычно выбирают несколько более стойким, чем для

симметричного, поскольку с помощью первого защищаются значительно большие объемы информации и на более продолжительный срок.

Таблица 6.5. *Длины ключей для симметричного и асимметричного алгоритмов шифрования с одинаковой стойкостью против криптоаналитической атаки методом тотального перебора*

Длина ключа для симметричного алгоритма	Длина ключа для асимметричного алгоритма
56	384
64	512
80	768
112	1792
128	2304

Работа с ключами

Предположим, некто Иванов и Петров пользуются надежной системой связи. Они делятся друг с другом своими соображениями на разные темы, играют в покер по переписке, заключают взаимовыгодные контракты, заверяя их своими цифровыми подписями, а затем производят расчеты между собой посредством электронных платежей. Алгоритм шифрования, используемый ими для этих целей, обладает сверхвысокой стойкостью. Но к сожалению, они покупают криптографические ключи в фирме "Ключкис и К°", девиз которой гласит: "Полностью доверьтесь нам. Надежность — девичья фамилия бывшей тещи нашего генерального директора".

Давиду Ключкису вместе с его компанией не надо напрягаться и вскрывать алгоритм шифрования, применяемый Ивановым и Петровым. С помощью дубликатов проданных им ключей он может читать абсолютно всю их конфиденциальную переписку, подделывать их цифровые подписи и снимать деньги с их электронных счетов.

Работа с ключами является самым уязвимым местом в любой криптосистеме. Создать стойкий алгоритм шифрования тоже нелегко, однако в этом деле можно положиться на результаты многочисленных научных исследований, проводимых учеными-криптографами. Сохранить ключ в секрете от посторонних значительно сложнее.

А значит, нет необходимости потеть над вскрытием криптографического алгоритма. Проще добыть ключ, с которым не очень бережно обращается его владелец. Да и в человеке гораздо легче отыскать изъяны, чем в шифре.

Потратить 10 млн долл. на разработку и изготовление специализированного криптоаналитического суперкомпьютера? Ну уж нет! Лучше за 1000 долл.

подкупить шифровальщика, который будет регулярно передавать ключи к шифру. А если бюджет позволяет, то за миллион долларов можно накупить этих ключей на много-много лет вперед. Американский военнослужащий Джон Уокер, к примеру, годами снабжал советскую разведку ключами к шифрам ВМС США, что позволило КГБ прочесть огромное количество американских шифровок. Ну а за несколько миллионов долларов доставать где-то ключи, чтобы потом с их помощью читать шифрованные сообщения, нет нужды и вовсе. За такие огромные деньги лучше сразу приобрести открытые тексты этих сообщений. Например, всего 2 млн долларов понадобилось советской разведке, чтобы купить высокопоставленного сотрудника американской контрразведки Олдрича Эймса вместе с женой.

Ну а если из бюджета не удается выкроить нужную сумму на подкуп, то можно воспользоваться другими методами. Человек слаб: его можно напоить, усыпить или просто оглушить тяжелым предметом, чтобы выкрасть криптографические ключи. Его можно также соблазнить. Морские пехотинцы, охранявшие американское посольство в Москве, по просьбе своих русских любовниц пропускали в шифровальный отсек посольства сотрудников КГБ — на "экскурсию".

Это значит, что криптографические ключи нуждаются в такой же защите, как и сами данные, которые шифруются с их помощью. К сожалению, во многих коммерческих средствах шифрования часто считается достаточным провозгласить об использовании DES-алгоритма или другого достаточно стойкого шифра, а о генерации и хранении ключей к ним не говорить ни слова. Например, программа DiskLock (версия 2.1) для персонального компьютера Macintosh позволяет шифровать файлы по DES-алгоритму. Однако ключ, использованный этой программой для зашифрования, она сохраняет на носителе информации вместе с файлом. Зная, где лежит ключ, его можно оттуда извлечь и затем без особых хлопот прочесть с помощью этого ключа содержимое зашифрованного файла. В данном случае совершенно неважно, что сам алгоритм шифрования достаточно надежен. Имеет значение только то, что его реализация абсолютно никуда не годится.

Генерация случайных и псевдослучайных последовательностей

Зачем вообще нужно рассматривать вопрос о генераторах случайных и псевдослучайных последовательностей в криптографии? Ведь в руководстве по каждому компилятору описана функция, позволяющая генерировать случайные числа. Наверное, лучше не городить огород, а воспользоваться тем, что уже имеется?

Нет, не лучше. К сожалению, датчики "случайных" чисел, встроенные в компиляторы, не подходят для криптографических приложений, поскольку генерируемые ими числа недостаточно случайны. Надеяться породить нечто со-

вершенно случайное на компьютере, который по своей природе является строго детерминированным устройством, по меньшей мере, безосновательно.

Псевдослучайные последовательности

Самое лучшее, на что способен компьютер, — это сгенерировать псевдослучайную последовательность, которая хотя и выглядит случайной, но, на самом деле, таковой не является. Период псевдослучайной последовательности должен быть достаточно большим, чтобы ее подпоследовательность требуемой длины была апериодичной, т. е. имела период, совпадающий с ее длиной. Например, если нужна строка из миллиона случайных бит, то для ее порождения не стоит использовать генератор последовательностей, которые повторяются через каждые 65536 бит.

Псевдослучайная битовая последовательность должна, по возможности, не отличаться от по-настоящему случайной. Необходимо, чтобы в ней число единиц примерно совпадало с числом нулей, а половина всех "полосок" (подряд идущих идентичных компонентов последовательности) имела длину 1, одна четвертая — длину 2, одна восьмая — длину 4 и т. д. Кроме только что перечисленных, существует еще ряд общепринятых тестов, которые позволяют проверить, действительно ли данная последовательность является псевдослучайной.

Созданию хороших генераторов псевдослучайных последовательностей уделяется достаточно большое внимание в математике. В настоящее время удается порождать последовательности с периодом порядка 2000—3000 бит. Проблема в том, что все генераторы псевдослучайных последовательностей при определенных условиях дают предсказуемые результаты и корреляционные зависимости. А это как раз то, чего ждут от псевдослучайных последовательностей криптоаналитики, чтобы предпринять эффективную атаку на криптосистемы, где эти последовательности используются.

Криптографически надежные псевдослучайные последовательности

В криптографии к псевдослучайным последовательностям предъявляется гораздо большие требования, чем простое наличие у них определенных признаков статистической случайности. Чтобы псевдослучайная последовательность была криптографически надежной, необходимо, чтобы она была непредсказуемой. Это значит, что для криптографически надежной псевдослучайной битовой последовательности невозможно заранее сказать, каким будет ее следующий бит, даже зная алгоритм генерации этой последовательности и все ее предыдущие биты. Как и любой криптографический алгоритм, генератор криптографически надежной псевдослучайной последовательности может быть атакован и вскрыт криптоаналитиком. Криптография

учит, как сделать такие генераторы стойкими к криптоаналитическим атакам различных типов.

По-настоящему случайные последовательности

Последовательность называется *по-настоящему случайной*, если ее нельзя воспроизвести. Это означает, что если запустить генератор по-настоящему случайных последовательностей дважды при одном и том же входе, то на его выходе получатся разные случайные последовательности. Основная трудность состоит в том, чтобы суметь отличить случайную последовательность от неслучайной. Если несколько раз зашифровать строку символов с помощью криптографического алгоритма, соответствующего ГОСТ 28147-89, то получится последовательность, очень напоминающая по-настоящему случайную. Чтобы доказать ее неслучайность, другого способа, кроме аренды у АНБ соответствующих вычислительных мощностей и программы вскрытия, не существует. Однако вряд ли ваше предложение об аренде будет воспринято там всерьез.

Генерация ключей

Стойкость шифра должна определяться только секретностью ключа. Если для генерации ключей используется нестойкий алгоритм, криптосистема будет нестойкой. Вскрытию подвергнется не сам шифр, а алгоритм генерации ключей.

Сокращенные ключевые пространства

Длина ключа в DES-алгоритме составляет 56 бит. В принципе, в качестве ключа может быть использован любой 56-битный вектор. На практике это правило часто не соблюдается. Например, широко распространенная программа шифрования файлов Norton Discreet, входящая в пакет Norton Utilities (версии 8.0 или более младшей версии), который предназначен для работы в операционной системе DOS, предлагает пользователю программную реализацию DES-алгоритма. Однако при вводе ключа разрешается подавать на вход программы только те символы, старший бит представления которых в коде ASCII равен нулю. Более того, пятый бит в каждом байте введенного ключа является отрицанием шестого бита, и в нем игнорируется младший бит. Это означает, что мощность ключевого пространства сокращается до каких-то жалких 2^{40} ключей. Таким образом из-за плохой процедуры генерации ключей программа Norton Discreet реализует алгоритм шифрования, ослабленный в десятки тысяч раз по сравнению с настоящим DES-алгоритмом.

В табл. 6.6 приведено количество возможных ключей в зависимости от различных ограничений на символы, которые могут входить в ключевую последовательность. Табл. 6.7 содержит сложность атаки методом тотального перебора при условии, что перебор ведется со скоростью 1 млн ключей в секунду.

Таблица 6.6. *Количество возможных ключей в зависимости от ограничений на символы ключевой последовательности*

Символы ключа	4 байта	5 байт	6 байт	7 байт	8 байт
Строчные буквы (26)	$4{,}7 \cdot 10^5$	$1{,}3 \cdot 10^7$	$3{,}2 \cdot 10^8$	$8{,}1 \cdot 10^9$	$2{,}2 \cdot 10^{11}$
Строчные буквы и цифры (36)	$1{,}8 \cdot 10^6$	$6{,}1 \cdot 10^7$	$2{,}3 \cdot 10^9$	$7{,}9 \cdot 10^{10}$	$2{,}9 \cdot 10^{12}$
Буквы и цифры (62)	$1{,}6 \cdot 10^7$	$9{,}3 \cdot 10^8$	$5{,}8 \cdot 10^{10}$	$3{,}6 \cdot 10^{11}$	$2{,}3 \cdot 10^{14}$
Печатаемые символы (95)	$8{,}2 \cdot 10^7$	$7{,}8 \cdot 10^9$	$7{,}5 \cdot 10^{11}$	$7{,}1 \cdot 10^{13}$	$6{,}7 \cdot 10^{15}$
Все ASCII-символы	$4{,}4 \cdot 10^9$	$1{,}2 \cdot 10^{10}$	$2{,}9 \cdot 10^{14}$	$7{,}3 \cdot 10^{16}$	$1{,}9 \cdot 10^{19}$

Таблица 6.7. *Сложность атаки методом тотального перебора при условии, что перебор ведется со скоростью 1 миллион ключей в секунду*

Символы ключа	4 байта	5 байт	6 байт	7 байт	8 байт
Строчные буквы (26)	0,6 сек.	13 сек.	6 мин.	2,3 ч.	2,5 дн.
Строчные буквы и цифры (36)	1,8 сек.	2 мин.	37 мин.	23 ч.	34 дн.
Буквы и цифры (62)	16 сек.	16 мин.	17 ч.	42 дн.	7,0 лет
Печатаемые символы (95)	1,5 мин.	2,2 ч.	8,6 дн.	2,3 лет	211 лет
Все ASCII-символы	1,3 ч.	14 дн.	9,0 лет	2400 лет	590000 лет

Из табл. 6.6 следует, что возможность опробовать 1 млн ключей в секунду позволяет в разумные сроки вскрывать 8-байтовые ключи из строчных букв и цифр, 7-байтовые буквенно-цифровые ключи, 6-байтовые ключи, составленные из печатаемых ASCII-символов, и 5-байтовые ключи, в которые могут входить любые ASCII-символы. А если учесть, что вычислительная мощь компьютеров увеличивается вдвое каждые полтора года, то для успешного отражения атаки методом тотального перебора в течение ближайшего десятилетия необходимо заблаговременно позаботиться о том, чтобы используемый ключ был достаточно длинным.

Плохие ключи

Когда отправитель сам выбирает ключ, с помощью которого он шифрует свои сообщения, его выбор обычно оставляет желать лучшего. Например, Петр Сергеевич Иванов скорее предпочтет использовать в качестве ключа Ivanov, чем &7)g*. И вовсе не потому, что он принципиально не желает соблюдать элементарные правила безопасности. Просто свою фамилию Иванов помнит гораздо лучше, чем абракадабру из шести произвольно взятых символов. Од-

нако тогда сохранить свою переписку в тайне ему не поможет и самый стойкий алгоритм шифрования в мире, особенно если используемые Ивановым ключи всегда совпадают с именами его ближайших родственников и записывает он эти ключи на клочках бумаги, которые наклеивает на компьютер. В ходе хорошо организованной атаки методом тотального перебора квалифицированный криптоаналитик не будет опробовать все ключи последовательно, один за другим. Он сначала проверит те из них, которые хоть что-то значат для Иванова. Такая разновидность атаки методом тотального перебора называется *словарной атакой*, поскольку в ходе нее противник использует словарь наиболее вероятных ключей. В этот словарь обычно входят:

- Имя, фамилия, отчество, инициалы, год рождения и другая личная информация, имеющая отношение к данному человеку. Например, при словарной атаке против Петра Сергеевича Иванова в первую очередь следует проверить PSI, PSIPSI, PIVANOV, Pivanov, psivanov, peteri, petel, IvanovP, peterivanov, Peter-Ivanov и т. д.
- Словарная база данных, составленная из имен людей, героев мультфильмов и мифических животных, ругательств, чисел (как цифрами, так и прописью), названий художественных фильмов, научно-фантастических романов, астероидов, планет и цветов радуги, общепринятых сокращений и т. д. В общей сложности для одного конкретного человека такая база данных насчитывает более 60 тыс. словарных единиц.
- Слова, которые получены путем внесения различных изменений в словарную базу данных, составленную на предыдущем этапе. Сюда относятся обратный порядок написания слова, замена в нем латинских букв o, l, z, s на цифры 0, 1, 2 и 5 соответственно, использование слова во множественном числе и т. д. Это даст дополнительно еще около миллиона словарных единиц для опробования в качестве возможного ключа к шифру.
- Слова, полученные с помощью замены строчных букв на заглавные. Такой замене в принципе может подвергаться любое число букв. Например, вместе со словом Ivanov будут проверяться слова iVanov, ivAnov, ivaNov, ivanOv, ivanoV, IVanov, IvAnov, IvaNov, IvanOv, IvanoV и т. д. Однако, вычислительная мощь современных компьютеров позволяет проверять только одно-, двух- и трехбуквенные замены строчных букв на заглавные.
- Слова на различных иностранных языках. Хотя компьютерные пользователи в основном работают с англоязычными операционными системами (DOS, UNIX, Windows и другими), существуют локализованные версии распространенных операционных систем, в которых допускается использование другого языка. Это означает, в качестве ключа на вход программы шифрования может быть подана любая фраза на родном языке ее пользователя. Следует также учитывать, что ключ может быть транслитерирован с любого языка (например, с русского или китайского) на английский и затем в таком виде введен в программу шифрования.

- Пары слов. Поскольку количество вероятных пар слов, из которых может состоять криптографический ключ, слишком велико, на практике криптоаналитики обычно ограничиваются словами из трех и четырех букв.

Случайные ключи

Хороший ключ представляет собой случайный битовый вектор. К примеру, если его длина составляет 56 бит, то это значит, что в процессе его генерации с одинаковой вероятностью может получиться любой из 2^{56} возможных ключей. Источником случайных ключей обычно служит либо природный случайный генератор (хорошей аналогией такого генератора является маленький ребенок, который только что научился ходить, — временные интервалы между его падениями абсолютно случайны). Кроме того, источником случайного ключа может быть криптографически надежный генератор псевдослучайных битовых последовательностей. Лучше, чтобы процесс генерации ключей был автоматизирован. Если под рукой нет компьютера для запуска программы, реализующей псевдослучайный генератор, или ваш ребенок давно уже вышел из младенческого возраста, можно бросать монетку или игральную кость.

Использование хорошего генератора случайных чисел является очень важным моментом при генерации криптографических ключей, однако не следует слишком много спорить о том, какой из этих генераторов является более случайным. Важнее применять стойкие алгоритмы шифрования и надежные процедуры работы с ключами. Если у вас появились сомнения относительно случайности при выборе ключа, можно использовать один из методов генерации ключей, описанных ниже в этой главе.

Во всех алгоритмах шифрования имеются так называемые *нестойкие* ключи. Это означает, что некоторые из ключей к шифру являются менее надежными, чем остальные. Поэтому при генерации ключей нужно автоматически проверять их на стойкость и генерировать новые вместо тех, которые эту проверку не прошли. К примеру, в DES-алгоритме имеются всего 24 нестойких ключа из общего количества 2^{56}, и следовательно вероятность наткнуться на нестойкий ключ пренебрежимо мала. Кроме того, откуда криптоаналитику знать, что для зашифрования конкретного сообщения или файла был применен именно нестойкий ключ? А сознательный отказ от использования нестойких ключей дает противнику дополнительную информацию о вашей криптосистеме, что нежелательно. С другой стороны, проверка ключей на нестойкость достаточно проста, чтобы ею пренебрегать.

Генерация открытых ключей гораздо более затруднена, чем генерация секретных ключей, поскольку открытые ключи должны обладать определенными математическими свойствами (например, должны быть результатом произведения двух простых чисел).

Пользоваться случайными ключами не всегда удобно. Иногда ключ требуется сохранить в памяти, а запомнить 36f9 67a3 f9cb d931 человеку не так-то

просто. В этом случае для генерации можно использовать некое правило, которое будет очевидно для вас, но недоступно для постороннего. Два варианта такого правила:

- Составьте ключ из нескольких слов, разделенных знаками препинания. Например, очень просто и надолго запоминаются ключи типа Yankee!Go home.
- Используйте в качестве ключа сочетание букв, которые представляют собой акроним более длинного слова. К примеру, броское название немецкого вина Liebenfraumilch позволяет путем отбрасывания гласных букв и добавления восклицательного знака сгенерировать ключ Lbnfrmlch!.

Пароль

Более привлекателен подход, при котором вместо отдельного слова используется достаточно длинное легко запоминающееся предложение на русском, английском или другом языке, которое преобразуется в ключ. Такое выражение в криптографии называется *паролем*. Для преобразования пароля в псевдослучайный битовый ключ можно воспользоваться любой однонаправленной хэш-функцией.

Пароль следует выбирать достаточно длинным, чтобы полученный в результате его преобразования ключ был случайным. Из теории информации известно, что в предложении на английском языке каждая буква содержит примерно 1,3 бита информации. Тогда, чтобы получить 64-битный ключ, пароль должен состоять примерно из 49 букв, что соответствует английской фразе из 10 слов.

Необходимо, чтобы при желании пароль было легко вспомнить, и в то же время требуется, чтобы он был достаточно уникален. Цитата из Козьмы Пруткова, которая у всех на слуху, вряд ли подойдет, поскольку его сочинения имеются в форме, доступной для воспроизведения на компьютере, и следовательно, могут быть использованы в словарной атаке. Лучше воспользоваться творчеством малоизвестного поэта или драматурга, процитировав его с ошибками. Большего эффекта можно добиться, если в цитате, использованной для генерации ключа, будут присутствовать иностранные слова. Идеально подходят для этой цели незатейливые ругательства — их вам не придется записывать, чтобы запомнить. Достаточно шарахнуть себя по пальцу молотком, и пароль автоматически придет вам в голову. Надо только сдержаться и не произнести его вслух, чтобы не подслушали посторонние.

Несмотря на все сказанное, залогом наилучшей защиты служит не шаманство при выборе пароля, а случайность полученного ключа. Хороший ключ — это случайный ключ, а значит, заранее будьте готовы к тому, что запомнить его наизусть будет очень трудно.

Стандарт ANSI X9.17

Американский национальный институт стандартов (ANSI) разработал метод генерации 64-битных ключей при помощи DES-алгоритма. Его основное назначение состоит в получении большого количества ключей для многократных сеансов связи. Вместо DES-алгоритма можно использовать любой другой стойкий алгоритм шифрования.

Пусть функция $E_K(P)$ осуществляет шифрование P по DES-алгоритму на заранее заготовленном ключе K, который используется только для генерации секретных ключей. Пусть далее V_0 является начальным 64-битным значением, которое держится в тайне от противника, а T_i представляет собой отметку времени, когда был сгенерирован i-й ключ. Тогда очередной случайный ключ R_i вычисляется с помощью преобразования:

$$R_i = E_K(E_K(T_i) \oplus V_i)$$

Чтобы получить очередное значение V_i, надо вычислить

$$V_i = E_K(E_K(T_i) \oplus R_i)$$

Нелинейные ключевые пространства

Одна из проблем, которую приходится решать военным криптографам, состоит в том, чтобы в случае захвата противником разработанного ими стойкого криптографического оборудования максимально затруднить его использование для защиты вражеских коммуникаций. Первым шагом на пути к решению этой проблемы является аппаратная реализация алгоритма шифрования в виде модуля, который противник не сможет вскрыть с целью ознакомления с особенностями алгоритма.

Затем нужно позаботиться о том, чтобы используемые ключи имели специальный вид. Если введенный ключ имеет отклонения от этого вида, то для шифрования сообщений будет применяться значительно менее стойкий криптографический алгоритм. Желательно, чтобы шансы случайно придать ключу специальный вид, необходимый для шифрования по стойкому алгоритму, были пренебрежимо малы.

Ключевое пространство такого алгоритма шифрования называется *нелинейным*, поскольку ключи не являются в равной степени стойкими. Если же все ключи шифра обладают одинаковой стойкостью, то его ключевое пространство называют *линейным* или *однородным*. Один из способов добиться нелинейности ключевого пространства состоит в разделении используемого ключа на две части: собственно ключа шифрования и некоторой фиксированной строки открытого текста, зашифрованной с помощью этого ключа. После расшифрования строки криптомодуль сравнивает полученный открытый текст с эталонным и при совпадении работает по стойкому алгоритму шифрования, а при несовпадении использует значительно менее нестойкий.

Например, блочный алгоритм шифрования с длиной блока 64 бита и длиной ключа 128 бит может использовать "составной" ключ из 192 бит. Тогда вероятность случайно использовать стойкий ключ будет достаточно мала — всего 2^{-64}.

Однако применить данный подход можно только при условии, что противник не в состоянии восстановить реализованный алгоритм шифрования методом обратного проектирования. Необходимо также добиться, чтобы разница в стойкости ключей не слишком бы бросалась противнику в глаза, и он ни о чем бы не догадался.

Передача ключей

Предположим, что Иванов и Петров обмениваются сообщениями, которые они защищают с использованием симметричного криптографического алгоритма. Для того чтобы зашифровать и потом расшифровать эти сообщения, они должны пользоваться одними и теми же ключами. Петров генерирует ключи при помощи хорошего генератора псевдослучайных битовых последовательностей. Вопрос в том, как Петров передает сгенерированные ключи Иванову. Например, они могут встретиться в каком-нибудь безлюдном месте (в кабинке вокзального туалета или на одном из спутников Сатурна), однако нет необходимости доказывать, что это не всегда удобно как для Петрова, так и для Иванова.

Посылать ключи Иванову, пользуясь тем же каналом связи, по которому они обмениваются шифрованными сообщениями, Петров тоже вряд ли захочет. Ведь если канал связи настолько ненадежен, что требует шифрования посылаемых сообщений, то отправка по нему ключа приведет к тому, что кто-нибудь перехватит этот ключ и сможет прочесть с его помощью все шифрованные сообщения Иванова и Петрова.

Упоминавшийся выше американский стандарт ANSI X9.17 определяет две разновидности ключей. Ключи шифрования данных используются для зашифрования сообщений, отправляемых по каналам связи. Ключи шифрования ключей применяются для зашифрования других ключей, которые предназначены для передачи получателям этих шифрованных сообщений. Ключи шифрования ключей передаются по каналам связи, исключающим подслушивание. Например, с помощью курьеров. Однако, поскольку делать это приходится значительно реже, чем посылать ключи шифрования данных, такой способ передачи ключей абонентам сети связи вполне пригоден для использования на практике.

Другой способ распределения ключей состоит в разбиении ключа на части и в передаче этих частей ключа по различным каналам связи. Одну часть можно отправить телеграфом, другую — доверить телефону, третью — послать с почтовым голубем и т. д. Даже если противнику удастся завладеть

всеми частями ключа, кроме одной, без подсказки он все равно не сумеет сообразить, что именно попало ему в руки.

Итак, Петров передает сгенерированные ключи Иванову либо при личной встрече с глазу на глаз, либо делит ключи на части, которые передает по различным каналам. Петров также может отсылать ключи Иванову, пользуясь тем же каналом, по которому он отправляет шифрованные сообщения, однако перед отсылкой ключей Петров должен зашифровать их с помощью ключа шифрования ключей. Поскольку количество передаваемых ключей не очень велико, ключ шифрования ключей часто менять не потребуется. Однако компрометация этого ключа даст противнику возможность прочесть всю шифрованную переписку, которая велась с его помощью, а посему оберегать ключ шифрования ключей следует с особенной тщательностью.

С ростом количества абонентов сети связи применять подход, рекомендуемый стандартом ANSI X9.17, становится все трудней и трудней. Если n человек захотят обмениваться шифрованными сообщениями, общее число переданных ими ключей составит n(n-1)/2. При n=1000 потребуется почти 500 тыс. пересылок ключей между абонентами, что может вылиться в серьезную проблему. В этом случае лучше производить рассылку ключей из единого центра (назовем его центром распределения ключей, сокращенно — ЦРК).

Проверка подлинности ключей

Как Иванову убедиться в том, что полученные им ключи были действительно переданы Петровым, а не противником, который притворился Петровым? Если Петров отдает ключи Иванову с глазу на глаз, то сделать это довольно легко. Однако если Петров присылает ключи с курьером, то Иванову придется положиться на честность курьера. А если полученные Ивановым ключи зашифрованы с помощью ключа шифрования ключей, ему остается лишь надеяться, что этот ключ имеется только у Петрова. Наконец, если рассылку ключей производит ЦРК, Иванов должен убедиться в надежности процедур рассылки, которые приняты в ЦРК.

Если противник полностью контролирует все каналы связи, которыми пользуется Иванов, он может заставить поверить Иванова во что угодно. Противник может прислать ему фальшивый ключ шифрования ключей якобы от Петрова и затем передавать Иванову поддельные ключи шифрования данных, зашифрованные на этом ключе. Противник будет в состоянии обзавестись собственным ЦРК и слать Иванову ключи от имени подлинного ЦРК. И ничего не подозревающий Иванов станет шифровать свои сообщения с помощью ключей, известных противнику.

Подобные рассуждения иногда приводятся, чтобы доказать ненужность криптографических методов, применяемых при передаче ключей. Ведь получается, что у Иванова и Петрова нет иного пути удостовериться в подлинности используемых ими ключей на все 100%, кроме личной встречи.

Эти рассуждения наивны. На практике установить тотальный контроль над чьими бы то ни было линиями связи невозможно. Действительно, Иванов не в силах убедиться наверняка, что переданный ему ключ не был подменен всемогущим противником. Однако, чтобы его контроль был действительно тотальным, такому противнику придется прибегнуть к средствам, которые в реальном мире ему вряд ли будут доступны.

Зная голос Петрова, Иванов может проверить подлинность полученного ключа по телефону. Если это открытый ключ, то его можно просто продиктовать. Если это секретный ключ, то его можно подвергнуть преобразованию с помощью однонаправленной хэш-функции и передать по телефону полученное значение для сравнения с аналогичным значением, подсчитанным на другом конце провода.

Нередко бывает, что Иванов желает убедиться: переданный ему ключ не только является подлинным, но и не был искажен при передаче. Ведь в результате Иванов может оказаться не в состоянии прочесть многие мегабайты шифртекста. Если соответствующий открытый текст был ASCII-файлом, то можно проверить ключ, попытавшись расшифровать с его помощью шифртекст, и получив абракадабру вместо осмысленного текста, сделать вывод о том, что ключ был передан с искажениями. Если открытый текст является достаточно случайным, можно прибегнуть к другим методам проверки ключа.

Один из них состоит в добавлении к передаваемому ключу так называемого *верификационного блока* — специального заголовка длиной несколько байт, который известен и отправителю, и получателю. Петров посылает его Иванову в шифрованном виде вместе с ключом. С помощью этого ключа Иванов расшифровывает верификационный блок и убеждается, что полученный открытый текст совпадает с известным ему эталоном. В качестве верификационного блока можно использовать контрольную сумму, вычисленную для данного ключа. В этом случае Иванову нет необходимости знать верификационный блок заранее, до прихода ключа. Имея ключ, Иванов вычислит для него контрольную сумму, а затем сравнит ее со значением, которое он получит, расшифровав верификационный блок.

К сожалению, данный метод не свободен от существенных недостатков. Во-первых, при его использовании противник может организовать криптоаналитическую атаку со знанием открытого текста. Во-вторых, он облегчает вскрытие шифров с относительно коротким ключом (подобных DES-алгоритму). Криптоаналитику достаточно вычислить контрольные суммы для каждого возможного ключа, а затем использовать эти контрольные суммы для определения ключей, примененных при шифровании перехватываемых в дальнейшем сообщений. Чтобы избавиться от этих недостатков, к вычисленной контрольной сумме ключа каждый раз необходимо добавлять случайные или хотя бы различные биты.

Контроль за использованием ключей

Прогресс в области вычислительной техники идет семимильными шагами. Ныне даже персональные компьютеры повсеместно работают под управлением многозадачных операционных систем. В результате пользователь часто оказывается не в состоянии определить, когда операционная система прерывает выполнение его программы шифрования, записывает ее саму, а также все ее данные на диск и переключается на работу с другим приложением. После того как операционная система наконец возобновляет процесс шифрования, все выглядит вполне пристойно: пользователь даже не успевает осознать, что шифровальная программа вместе с используемым ею ключом побывала на диске. В итоге ключ так и останется на диске в незашифрованном виде, пока поверх него не будут записаны другие данные. Когда это случится — через полсекунды, через месяц или вообще никогда, не может сказать никто. Однако враг не дремлет, и вполне может произойти так, что ключ все еще будет храниться на диске в открытую, когда вражеский агент проверит этот диск в поисках полезной информации.

В некоторых случаях для организации обмена шифрованными сообщениями применяются *сеансовые ключи*. Они называются так потому, что используются лишь в одном сеансе связи, а затем уничтожаются. В результате вероятность их компрометации уменьшается. Еще больше понизить эту вероятность можно с помощью следующего метода.

К сгенерированному ключу (назовем его основным) добавляется битовый управляющий вектор (УВ), который содержит информацию об ограничениях, накладываемых на использование этого ключа. УВ подвергается хэшированию и затем складывается с основным ключом по модулю 2. Полученный результат служит в качестве ключа для зашифрования сеансового ключа. Зашифрованный сеансовый ключ хранится вместе с УВ. Чтобы получить сеансовый ключ в исходном виде, надо применить хэширование к УВ, сложить его с основным ключом по модулю 2 и использовать результат для расшифрования сеансового ключа. Достоинством этого метода является возможность задействовать УВ произвольной длины и открыто хранить его вместе с зашифрованным основным ключом.

Обновление ключей

Иногда при частой смене ключей оказывается очень неудобно каждый раз передавать их абонентам сети для использования при шифровании и расшифровании сообщений. В качестве выхода из этой неудобной ситуации можно предложить генерацию новых ключей из старых, называемую в криптографии *обновлением ключей*.

Если Иванов и Петров владеют общим криптографическим ключом, то, подав его на вход одной и той же однонаправленной функции, они получат

одинаковый результат, из которого смогут выбрать необходимое число бит, чтобы составить из них новый ключ. Необходимо только помнить о том, что новый ключ будет обладать такой же стойкостью, что и старый. Если противник знает старый ключ, он может вычислить для этого ключа соответствующее значение однонаправленной функции и получить в свое распоряжение новый ключ.

Хранение ключей

Проще всего хранить ключи для криптосистемы, у которой имеется один-единственный пользователь. Пользователь просто запоминает этот ключ и при необходимости вводит его с клавиатуры компьютера по памяти. Однако поскольку по-настоящему случайный ключ запомнить нелегко, для его хранения можно использовать магнитную карточку, или пластиковый ключ с размещенным на нем ПЗУ (так называемый *ПЗУ-ключ*), или интеллектуальную карту. Для ввода такого ключа достаточно вставить его физический носитель в специальный считыватель, подключенный к компьютеру. При этом действительное значение вводимого ключа пользователю неизвестно, и, следовательно, он не сможет его разгласить или скомпрометировать. Способ использования ключа определяется управляющим вектором, записанным на физический носитель вместе с этим ключом.

ПЗУ-ключ придумали очень умные люди. Пользователь обычно гораздо лучше осознает, как правильно обращаться с обычным ключом от замка. Придание криптографическому ключу такого же вида, какой имеет ставший нам привычным замковый ключ, позволяет чисто интуитивно избегать многих ошибок, связанных с хранением криптографических ключей.

С целью дальнейшего уменьшения вероятности компрометации ключа можно разделить его на две части. Первую следует реализовать в виде ПЗУ-ключа, а вторую поместить в память компьютера. Тогда потеря "пэ-зэушной" части ключа или его половинки, хранимой в памяти компьютера, не приведет к разглашению криптографического ключа в целом. А части ключа при необходимости можно заменить отдельно друг от друга.

Труднозапоминаемые ключи можно хранить на компьютерном диске в зашифрованном виде. Например, открытый ключ, состоящий из большого количества цифр, лучше всего зашифровать с помощью DES-алгоритма и запомнить на диске. Более короткий ключ к DES-алгоритму легче припомнить, когда понадобится расшифровать открытый ключ.

Если ключи генерируются с использованием хорошего датчика псевдослучайных битовых последовательностей, может оказаться более удобным не хранить сгенерированные ключи, а каждый раз заново генерировать их, задавая соответствующее начальное значение датчика, которое легко запоминается.

Запасные ключи

Рассмотрим такой возможный случай. Давид Ключкис, президент фирмы "Ключкис и К°", постоянно шифрует всю информацию, связанную с делами своей фирмы. Однако, к несчастью, он переходил улицу в неположенном месте, и его сбил 10-тонный грузовик. При этом больше всего пострадала та часть его организма, которая отвечала за хранение ключей к используемому им алгоритму шифрования. И если у Абрама Ключмана, первого заместителя Ключкиса, нет копии этих ключей, их фирма окажется в незавидном положении. Целью шифрования является защита информации от посторонних. Поэтому при условии, что Ключкис использовал стойкий криптографический алгоритм, зашифрованные им данные, касающиеся дел фирмы, будут потеряны навсегда.

Избежать этой ситуации Ключману поможет схема с депонированием ключей: все служащие компании, включая ее руководителей, должны регулярно сдавать копии своих криптографических ключей начальнику службы безопасности Бориса Ключевского, который будет класть их на хранение в сейф. Узнав, что Ключкис в бессознательном состоянии попал в больницу, Ключман просто попросит Ключевского принести требуемые криптографические ключи. Будет еще лучше, если Ключман узнает комбинацию, открывающую сейф, заранее, чтобы не оказаться в безвыходном положении, если впечатлительного Ключевского хватит удар, когда он узнает о несчастье, приключившемся с Ключкисом.

Недостаток этой схемы хранения запасных ключей состоит в том, что руководство фирмы вынуждено доверить все свои секреты Ключевскому. Взамен лучше разделить ключи на части и раздать их различным сотрудникам службы безопасности. В результате ни один из них не сможет получить единоличный доступ к зашифрованной информации. А еще лучше записать каждую часть ключа на отдельную интеллектуальную карту. Тогда их владельцы не только не будут знать значение части ключа, которая передана им на хранение, но и можно будет проконтролировать, сколько раз владелец конкретной интеллектуальной карты участвовал в получении целого ключа из частей для осуществления доступа к зашифрованной этим ключом информации.

Скомпрометированные ключи

Любой стойкий шифр обеспечивает надежную защиту только до тех пор, пока ключ к нему хранится в тайне. Если Иванов потеряет этот ключ, если у него этот ключ украдут, если этот ключ зачитает диктор в вечернем выпуске телевизионных новостей или этот ключ будет скомпрометирован каким-либо другим образом, обеспечиваемая им защита будет сведена на нет.

Скомпрометированный ключ к симметричному алгоритму шифрования необходимо побыстрее сменить. После этого остается только надеяться, что

противник успел узнать из прочитанной шифрпереписки не слишком много интересного для себя. С открытыми ключами, которые используются не только для шифрования данных, но и для аутентификации и цифровой подписи документов, дело обстоит сложнее. Поэтому так важно, чтобы о компрометации открытого ключа все заинтересованные стороны узнали как можно скорее. Если все сообщения Иванова аккуратно снабжены датой, это поможет значительно уменьшить вероятность потенциального ущерба, поскольку тогда их получатели окажутся в состоянии отобрать и проверить те из них, которые могли быть сфальсифицированы.

Если Иванов не знает, когда именно был скомпрометирован его ключ, остается только посоветовать ему аккуратнее обращаться со своими ключами и использовать для разных целей различные ключи. Ведь входные двери в его офис и в его квартиру вряд ли открываются одним и тем же ключом.

Продолжительность использования ключа

Любой ключ должен использоваться в течение ограниченного периода времени. Тому есть несколько причин:

- Чем дольше ключ находится в действии, тем больше вероятность того, что он будет скомпрометирован.
- Длительное пользование одним и тем же ключом увеличивает потенциальный ущерб, который может быть нанесен в случае его компрометации.
- Ключ, очень долго применявшийся для шифрования информации, становится лакомым кусочком для противника, у которого появляется стимул потратить на его вскрытие значительные ресурсы, поскольку полученная выгода позволит оправдать понесенные расходы.
- Криптоаналитическую атаку на шифр вести тем легче, чем больше перехваченного шифртекста для него накоплено.

Продолжительность использования ключа зависит от криптосистемы. В различных криптосистемах эта продолжительность должна быть разной. Для шифрования речевых сообщений, передаваемых по телефону, имеет смысл менять ключ после каждого разговора. В выделенных каналах связи продолжительность использования ключа определяется ценностью шифруемой информации и скоростью ее передачи. При скорости в 9600 бит в секунду смену ключа следует производить реже, чем при скорости в несколько гигабит в секунду. Если условия позволяют, такие ключи необходимо менять, по крайней мере, ежедневно.

Не требуют частой смены ключи шифрования ключей. Они используются от случая к случаю, и поэтому объем перехваченного противником шифртекста для них невелик. Кроме того, про свойства соответствующего ему открытого текста противнику заранее ничего не известно, поскольку хороший ключ

представляет собой достаточно случайный набор бит. Однако компрометация ключа шифрования ключей влечет за собой гораздо более серьезные потери, чем это происходит при потере сеансового ключа или ключа шифрования данных. Необходим разумный компромисс между вероятностью вскрытия ключа шифрования ключей из-за его слишком длительного использования и возможностью компрометации этого ключа при его передаче абонентам сети связи. В большинстве случаев разумной представляется ежемесячная, а иногда даже ежегодная смена ключа шифрования ключей.

Ключи, применяемые для шифрования файлов, которые хранятся на компьютерных дисках, слишком часто менять не надо. Регулярное повторное шифрование файлов на новых ключах только даст больше полезной информации криптоаналитику, который будет пытаться их вскрыть. Лучше применить подход, при котором каждый файл шифруется при помощи своего ключа. А сами ключи, в свою очередь, зашифровываются на ключе шифрования ключей, который затем прячется в надежное место (например, в стальной сейф).

Что касается открытых ключей, то продолжительность их использования сильно варьируется в зависимости от области применения. Если открытый ключ применяется для целей аутентификации или для цифровой подписи, он продолжает оставаться актуальным годами, иногда даже десятилетиями. Но даже в этом случае не следует пренебрегать сменой ключа каждые 2—3 года, чтобы в распоряжении криптоаналитика накапливалось меньше шифртекста, необходимого для организации атаки. А старый ключ все равно надо продолжать хранить в секрете — он может понадобиться, чтобы, например, подтвердить подлинность подписи, поставленной в течение периода, пока этот ключ был действующим.

Уничтожение ключей

Использованные криптографические ключи ни в коем случае не должны попасть в руки противника. Поэтому, как только в ключах отпала надобность, их следует уничтожить. Если ключи хранятся на бумажном носителе, его надо сжечь или пропустить через специальный аппарат для уничтожения бумаг, который должен быть достаточно качественным. Ведь будет очень обидно, если ваш алгоритм шифрования, способный выдержать атаку методом грубой силы в течение нескольких миллионов лет, вскроют только потому, что за несколько десятков тысяч долларов кто-то наймет сотню безработных, и за год они соберут воедино недостаточно тщательно "пережеванный" лист бумаги с записанными на нем ключами.

Если ключ хранился в СППЗУ, необходимо несколько раз записать информацию поверх него. Если для хранения ключа использовалось ПЗУ, его надо разнести молотком на мелкие кусочки и пустить их по ветру. Если ключ лежал на компьютерном диске, на место ключа придется многократно запи-

сать ничего не значащие данные или уничтожить диск подобно ПЗУ. При работе на компьютере в многозадачном режиме следует обратить особое внимание на способность операционной системы создавать временные файлы на диске для хранения рабочей копии программы шифрования и ее данных. А сверхосторожный пользователь обязательно напишет программу, которая будет отыскивать копии ключа на свободных секторах диска и затем затирать их.

Глава 7

Криптографические протоколы

Что такое криптографический протокол

Протокол — это последовательность шагов, которые предпринимают две или большее количество сторон для совместного решения задачи. Все шаги следуют в порядке строгой очередности, и ни один из них не может быть сделан прежде, чем закончится предыдущий. Кроме того, любой протокол подразумевает участие, по крайней мере, двух сторон. В одиночку можно, например, смешать и выпить коктейль, но к протоколу это не имеет никакого отношения. Поэтому придется угостить кого-нибудь сделанным коктейлем, чтобы его приготовление и дегустация стали настоящим протоколом. И наконец протокол обязательно предназначен для достижения какой-то цели.

Протоколы имеют и другие отличительные черты:

- каждый участник протокола должен быть заранее оповещен о шагах, которые ему предстоит предпринять;
- все участники протокола должны следовать его правилам добровольно, без принуждения;
- необходимо, чтобы протокол допускал только однозначное толкование, а его шаги были совершенно четко определены и не допускали возможности их неправильного понимания;
- протокол должен содержать описание реакции его участников на любые ситуации, возникающие в ходе реализации этого протокола — иными словами, недопустимым является положение, когда для возникшей ситуации протоколом не определено соответствующее действие.

Криптографическим протоколом называется такой, в основе которого лежит криптографический алгоритм. Однако целью криптографического протокола зачастую является не только сохранение информации в тайне от посторонних. Участники криптографического протокола могут быть близкими друзь-

ями, у которых нет друг от друга секретов, а могут являться настолько непримиримыми врагами, что каждый из них отказывается сообщить другому, какое сегодня число. Тем не менее, им может понадобиться поставить свои подписи под совместным договором или удостоверить свою личность. В этом случае криптография нужна, чтобы предотвратить или обнаружить подслушивание посторонними лицами, не являющимися участниками протокола, а также не допустить мошенничества. Поэтому часто требуется, чтобы криптографический протокол обеспечивал следующее: его участники не могут сделать или узнать больше того, что определено протоколом.

Зачем нужны криптографические протоколы

В повседневной жизни нам приходится сталкиваться с протоколами буквально на каждом шагу — играя в любые игры, делая покупки в магазинах или голосуя на выборах. Многими протоколами нас научили пользоваться родители, школьные учителя и друзья. Остальные мы сумели узнать самостоятельно.

В настоящее время люди все чаще контактируют друг с другом при помощи компьютеров. Компьютеры же, в отличие от большинства людей, в школу не ходили, у них не было родителей, да и учиться самостоятельно они не в состоянии. Поэтому компьютеры приходится снабжать формализованными протоколами, чтобы они смогли делать то, что люди выполняют особо не задумываясь. Например, если в магазине не окажется кассового аппарата, вы все равно сможете купить в нем необходимую вещь. Однако такое кардинальное изменение протокола поставило бы бедный компьютер в полный тупик.

Большинство протоколов, которые люди используют при общении друг с другом с глазу на глаз, хорошо себя зарекомендовали только потому, что их участники имеют возможность вступить в непосредственный контакт. Взаимодействие с другими людьми через компьютерную сеть, наоборот, подразумевает анонимность. Будете ли вы играть с незнакомцем в преферанс, не видя, как он тасует колоду и раздает карты? Доверите ли вы свои деньги совершенно постороннему человеку, чтобы он купил вам что-нибудь в магазине? Пошлете ли вы свой бюллетень голосования по почте, зная, что с ним сможет ознакомиться любой из почтовых работников и потом рассказать всем о ваших нетрадиционных политических пристрастиях? Думаю, что нет.

Глупо считать, что компьютерные пользователи ведут себя более честно, чем абсолютно случайные люди. То же самое касается и сетевых администраторов, и проектировщиков компьютерных сетей. Большинство из них и в самом деле честные люди, однако есть и такие, кто может причинить вам большие неприятности. Поэтому так нужны криптографические протоколы, использование которых позволяет защититься от непорядочных людей.

Распределение ролей

Чтобы описание протоколов было более наглядным, имена их участников однозначно определяют роли, им уготованные (табл. 7.1). Пусть Антон и Борис принимают участие во всех двухсторонних протоколах. Как правило, начинает выполнение шагов, предусмотренных протоколом, Антон, а ответные действия предпринимает Борис. Если протокол является трех- или четырехсторонним, исполнение соответствующих ролей берут на себя Владимир и Георгий. Об остальных персонажах подробнее будет рассказано позже.

Таблица 7.1. *Роли, которые играют участники протоколов*

Участник протокола	Роль в протоколе
Антон	Первый участник протоколов
Борис	Второй участник протоколов
Владимир	Участник трех- и четырехсторонних протоколов
Георгий	Участник четырехсторонних протоколов
Дмитрий	Доверенное лицо, наделенное правами арбитра
Зиновий	Злоумышленник
Кирилл	Контролер
Олег	Охранник
Петр	Подслушивает за участниками протоколов
Сергей	Свидетель

Протокол с арбитражем

Арбитр — участник протокола, которому остальные участники полностью доверяют, предпринимая соответствующие действия для завершения очередного шага протокола. Это значит, что у арбитра нет личной заинтересованности в достижении тех или иных целей, преследуемых участниками протокола, и он не может выступить на стороне любого из них. Участники протокола также принимают на веру все, что скажет арбитр, и беспрекословно следуют всем его рекомендациям.

В протоколах, которым мы следуем в повседневной жизни, роль арбитра чаще играет адвокат. Однако попытки перенести протоколы с адвокатом в качестве арбитра из повседневной жизни в компьютерные сети наталкиваются на существенные препятствия.

- Легко довериться адвокату, про которого известно, что у него незапятнанная репутация, и с которым можно установить личный контакт. Од-

нако если два участника протокола не доверяют друг другу, арбитр, не облаченный в телесную оболочку и существующий где-то в недрах компьютерной сети, вряд ли будет пользоваться у них доверием.

- Расценки за услуги, оказываемые адвокатом, известны. Кто и каким образом будет оплачивать издержки за аналогичные услуги арбитра в компьютерной сети?
- Введение арбитра в любой протокол увеличивает время, затрачиваемое на реализацию этого протокола.
- Поскольку арбитр контролирует каждый шаг протокола, его участие в очень сложных протоколах может стать узким местом при их реализации. Увеличение числа арбитров позволяет избавиться от данного узкого места, однако одновременно возрастут и накладные расходы на реализацию протокола.
- В силу того, что все участники протокола должны пользоваться услугами одного и того же арбитра, действия злоумышленника, который решит нанести им ущерб, будут направлены, в первую очередь, против этого арбитра. Следовательно, арбитр представляет собой слабое звено любого протокола с арбитражем.

Несмотря на отмеченные препятствия, протоколы с арбитражем находят широкое применение на практике. В ходе дальнейшего изложения доверенное лицо, наделенное правами арбитра, будет именоваться Дмитрием.

Протокол с судейством

Чтобы снизить накладные расходы на арбитраж, протокол, в котором участвует арбитр, часто делится на две части. Первая полностью совпадает с обычным протоколом без арбитража, а ко второй прибегают только в случае возникновения разногласий между участниками. Для разрешения конфликтов между ними используется особый арбитр — *судья*.

Подобно арбитру, судья является незаинтересованным участником протокола, которому остальные участники доверяют. Однако в отличие от арбитра, судья участвует отнюдь не в каждом шаге протокола. Услугами судьи пользуются, только если требуется разрешить сомнения относительно правильности действий участников протокола. Если таких сомнений ни у кого не возникает, судейство не понадобится.

В компьютерных протоколах с судейством предусматривается наличие данных, проверив которые доверенное третье лицо может решить, не смошенничал ли кто-либо из участников этого протокола. Хороший протокол с судейством также позволяет выяснить, кто именно ведет себя нечестно. Это служит прекрасным превентивным средством против мошенничества со стороны участников такого протокола.

Самоутверждающийся протокол

Самоутверждающийся протокол не требует присутствия арбитра для завершения каждого шага протокола. Он также не предусматривает наличие судьи для разрешения конфликтных ситуаций. Самоутверждающийся протокол устроен так, что, если один из его участников мошенничает, другие смогут моментально распознать нечестность, проявленную этим участником, и прекратить выполнение дальнейших шагов протокола.

Конечно же, хочется, чтобы существовал универсальный самоутверждающийся протокол на все случаи жизни. Однако на практике в каждом конкретном случае приходится конструировать свой специальный самоутверждающийся протокол.

Разновидности атак на протоколы

Атаки на протоколы бывают направлены против криптографических алгоритмов, которые в них задействованы, против криптографических методов, применяемых для их реализации, а также против самих протоколов. Для начала предположим, что используемые криптографические алгоритмы и методы являются достаточно стойкими, и рассмотрим атаки собственно на протоколы.

Если некто, не являющийся участником протокола, попытается подслушать информацию, которой обмениваются его участники, — это *пассивная атака* на протокол. Она так названа потому, что атакующий (будем именовать его Петром) может только накапливать данные и наблюдать за ходом событий, но не в состоянии влиять на него. Пассивная атака подобна криптоаналитической атаке со знанием только шифртекста. Поскольку участники протокола не обладают надежными средствами, позволяющими им определить, что они стали объектом пассивной атаки, для защиты от нее используются протоколы, дающие возможность предотвращать возможные неблагоприятные последствия пассивной атаки, а не распознавать ее.

Атакующий может попытаться внести изменения в протокол ради собственной выгоды. Он может выдать себя за участника протокола, внести изменения в сообщения, которыми обмениваются участники протокола, подменить информацию, которая хранится в компьютере и используется участниками протокола для принятия решений. Это *активная атака* на протокол, поскольку атакующий (назовем его Зиновием) может вмешиваться в процесс выполнения шагов протокола его участниками.

Итак, Петр пытается собрать максимум информации об участниках протокола и об их действиях. У Зиновия же совсем другие интересы — ухудшение производительности компьютерной сети, получение несанкционированного доступа к ее ресурсам, внесение искажений в базы данных. При этом и Петр, и Зиновий не обязательно являются совершенно посторонними лица-

ми. Они могут быть легальными пользователями, системными и сетевыми администраторами, разработчиками программного обеспечения и даже участниками протокола, которые ведут себя непорядочно или даже вовсе не соблюдают этот протокол. В последнем случае атакующий называется *мошенником*. Пассивный мошенник следует всем правилам, которые определены протоколом, но при этом еще и пытается узнать о других участниках больше, чем предусмотрено этим протоколом. Активный мошенник вносит произвольные изменения в протокол, чтобы нечестным путем добиться для себя наибольшей выгоды.

Защита протокола от действий нескольких активных мошенников представляет собой весьма нетривиальную проблему. Тем не менее при некоторых условиях эту проблему удается решить, предоставив участникам протокола возможность вовремя распознать признаки активного мошенничества. А защиту от пассивного мошенничества должен предоставлять любой протокол вне зависимости от условий, в которые поставлены его участники.

Протокол обмена сообщениями с использованием симметричного шифрования

Предположим, что Антон и Борис хотят обмениваться секретными сообщениями по каналу связи, не защищенному от подслушивания. Естественно, им придется воспользоваться шифрованием. Однако чтобы Антон успешно зашифровал свое сообщение Борису, а тот, получив это сообщение, смог его расшифровать, они должны действовать в соответствии со следующим протоколом:

1. Антон и Борис уславливаются о том, какой криптосистемой они будут пользоваться.
2. Антон и Борис генерируют ключи для шифрования и расшифрования своих сообщений и затем обмениваются ими.
3. Антон шифрует сообщение с использованием криптографического алгоритма и ключа, о которых он заранее договорился с Борисом.
4. Антон отправляет зашифрованное сообщение Борису.
5. Борис расшифровывает полученное сообщение, применяя тот же криптографический алгоритм и ключ, которыми пользовался Антон.

Если Петр имеет возможность перехватывать сообщения, которые передают друг другу Антон и Борис, он может попытаться прочесть эти сообщения. Если Антон и Борис используют стойкий алгоритм шифрования, — они в безопасности. Однако Петр может оказаться в состоянии подслушивать за Антоном и Борисом, когда они выполняют шаг 1 и шаг 2 протокола. Поэтому стойкая криптосистема не должна опираться на сохранение в тайне алгоритма шифрования. Необходимо, чтобы ее стойкость определялась од-

ной только длиной секретного ключа. Тогда Антон и Борис могут условиться, каким шифром они будут пользоваться, ничуть не заботясь о том, что их подслушает Петр. Тем не менее, шаг 2 протокола все равно должен выполняться ими в обстановке строжайшей секретности. Требуется, чтобы свои сгенерированные ключи Антон и Борис хранили в секрете до, во время и после процесса выполнения всех шагов протокола. Иначе Петр сможет прочесть всю их шифрованную переписку.

Что касается Зиновия, то в зависимости от преследуемых целей он может действовать по-разному. Если Зиновий прервет связь между Антоном и Борисом, они будут не в состоянии обмениваться сообщениями. Зиновий может заменить шифрованное сообщение, посланное Антоном, на свое собственное. Попытавшись расшифровать поддельное сообщение, Борис вместо осмысленного открытого текста получит абракадабру, и решит, что Антон не слишком серьезно отнесся к шифрованию своего сообщения или что при передаче оно было просто искажено. Хуже, если Зиновий узнает ключ, которым пользуются Антон и Борис. Тогда он сможет зашифровать любое сообщение и отправить его Борису от имени Антона, а у Бориса не будет возможности распознать подделку.

Если Антон вознамерится навредить Борису, тот будет не в силах ему помешать. Например, Антон может заставить Бориса отказаться от обмена сообщениями. Для этого Антону достаточно передать копию ключа Петру с условием, что Петр опубликует открытые тексты сообщений Бориса в газете для всеобщего обозрения. Это значит, что, используя приведенный выше протокол, Антон и Борис должны полностью доверять друг другу.

Подводя итог сказанному, следует еще раз подчеркнуть, что в протоколах обмена сообщениями с использованием симметричного шифрования ключи должны храниться и распределяться между участниками протокола в строжайшем секрете. Ключи являются не менее ценными, чем сама информация, которая шифруется с их использованием, поскольку знание ключей противником открывает ему неограниченный доступ к этой информации.

Протокол обмена сообщениями с использованием шифрования с открытым ключом

Симметричную криптосистему можно сравнить с сейфом. Криптографический ключ соответствует комбинации, отпирающей сейф. Каждый, кто знает комбинацию, может открыть сейф. Тому, кто комбинацией не владеет, но все равно желает знакомиться с содержимым сейфа, лучше записаться на курсы медвежатников, если таковые, конечно, существуют.

В 1976 г. американские криптологи У. Диффи и М. Хеллман изобрели криптографию с открытым ключом. Они предложили использовать два вида ключей — *открытые* и *тайные*. Вычислить тайный ключ, зная только от-

крытый, очень сложно. Человек, владеющий открытым ключом, может с его помощью зашифровать сообщение. Расшифровать это сообщение в состоянии только тот, кто имеет соответствующий тайный ключ. В отличие от симметричной криптосистемы, для алгоритма шифрования с открытым ключом больше подходит сравнение с почтовым ящиком. Опустить почту в этот ящик очень просто, равно как и зашифровать сообщение с применением открытого ключа. А извлечение из почтового ящика находящейся в нем корреспонденции сродни расшифрованию сообщения. Желающего сделать это без ключа вряд ли ожидает легкая работа. Обладатель же ключа откроет почтовый ящик, т. е. расшифрует сообщение, без особого труда.

При зашифровании и расшифровании сообщений в криптосистеме с открытым ключом используется однонаправленная функция с лазейкой. Причем зашифрование соответствует вычислению значения этой функции, а расшифрование — ее обращению. Поскольку и открытый ключ, и сама функция не являются секретными, каждый может зашифровать свое сообщение с их помощью. Однако обратить функцию, чтобы получить открытый текст зашифрованного сообщение, в разумные сроки не сможет никто. Для этого необходимо знать лазейку, т. е. тайный ключ. Тогда расшифрование будет таким же легким, как и зашифрование.

Протокол обмена сообщениями с использованием шифрования с открытым ключом выглядит следующим образом:

1. Антон и Борис уславливаются о том, какой криптосистемой они будут пользоваться.
2. Борис посылает Антону свой открытый ключ.
3. Антон шифрует открытый текст своего сообщения при помощи открытого ключа, присланного ему Борисом, и шлет полученный в результате шифртекст Борису.
4. Борис расшифровывает сообщение Антона, используя свой тайный ключ.

Применение криптографии с открытым ключом позволяет решить проблему передачи ключей, которая присуща симметричным криптосистемам. Без какой-либо предварительной подготовки Антон может отправить зашифрованное сообщение Борису. И хотя в распоряжении Петра окажутся и сам алгоритм шифрования, и зашифрованное Антоном сообщение, и даже использованный им для этого ключ, Петр все равно не сможет расшифровать данное сообщение.

Чтобы упростить протокол обмена шифрованными сообщениями, открытые ключи всех абонентов единой сети связи часто помещаются в справочную базу данных, находящуюся в общем пользовании этих абонентов. Тогда протокол обмена сообщениями с использованием шифрования с открытым ключом имеет следующий вид:

1. Антон находит открытый ключ Бориса в базе данных.

2. Антон шифрует открытый текст своего сообщения при помощи открытого ключа Бориса.
3. Борис расшифровывает сообщение Антона, используя свой тайный ключ.

Второй протокол в большей степени напоминает отправку писем по почте, поскольку получатель сообщения не является участником протокола до тех самых пор, пока не захочет ознакомиться с открытым текстом этого сообщения.

Гибридные криптосистемы

На практике криптосистемы с открытым ключом используются для шифрования не сообщений, а ключей. На это есть две основные причины:

- Алгоритмы шифрования с открытым ключом в среднем работают в тысячи раз медленнее, чем алгоритмы с симметричным ключом. И хотя темпы роста компьютерной производительности очень высоки, требования к скорости шифрования растут не менее стремительно. Поэтому криптосистемы с открытым ключом вряд ли когда-нибудь смогут удовлетворить современные потребности в скорости шифрования.
- Алгоритмы шифрования с открытым ключом уязвимы по отношению к криптоаналитическим атакам со знанием открытого текста. Пусть C=E(P), где C обозначает шифртекст, P — открытый текст, E — функцию шифрования. Тогда, если P принимает значения из некоторого конечного множества, состоящего из n открытых текстов, криптоаналитику достаточно зашифровать все эти тексты, используя известный ему открытый ключ, и сравнить результаты с C. Ключ таким способом ему вскрыть не удастся, однако открытый текст будет успешно определен.

Чем меньше количество (n) возможных открытых текстов, тем эффективнее будет атака на криптосистему с открытым ключом. Например, если криптоаналитику известно, что шифрованию подверглась сумма сделки, не превышающая 1 млн долл., он может перебрать все числа от 1 до 1 000 000.

В большинстве случаев криптографические алгоритмы с открытым ключом применяются для шифрования сеансовых ключей при их передаче абонентам сети связи. Соответствующий протокол выглядит обычно так:

1. Антон генерирует сеансовый ключ K и шифрует его с использованием открытого ключа B, принадлежащего Борису:

   $E_B(K)$

2. Борис расшифровывает сообщение Антона при помощи своего тайного ключа и получает в результате сеансовый ключ K, сгенерированный Антоном:

   $K=D_B(E_B(K))$

3. Антон и Борис обмениваются сообщениями, зашифрованными одним и тем же сеансовым ключом K.

Данный протокол позволяет не хранить ключи в течение неопределенного периода времени до тех пор, пока они не понадобятся в очередном сеансе связи. Сеансовый ключ можно сгенерировать и отправить по требуемому адресу непосредственно перед посылкой сообщения, которое предполагается зашифровать на этом ключе, а потом его сразу уничтожить. Тем самым уменьшается вероятность компрометации сеансового ключа. И хотя тайный ключ также может быть скомпрометирован, риск здесь значительно меньше, поскольку тайный ключ используется очень редко — только для однократного зашифрования сеансового ключа.

"Шарады" Меркля

Оригинальный протокол обмена ключами был предложен американским криптологом Р. Мерклем (R. Merkle) в 1974 г. Данный протокол основывается на так называемых "шарадах", которые требуется решить для получения ключа. Это гораздо легче сделать отправителю и получателю шифрованных сообщений, чем криптоаналитику, перехватывающему их. В результате Антон может послать шифрованное сообщение Борису, предварительно не передавая ему секретного ключа:

1. Борис генерирует 2^{20} (около 1 млн) сообщений вида: "Это шарада под номером x. Это секретный ключ под номером y", где x — случайное число, y — случайный секретный ключ. Пара x, y является уникальной для каждого сообщения. Затем Борис шифрует все эти сообщения при помощи какого-либо симметричного алгоритма на различных 20-битных ключах и отсылает Антону.
2. Антон случайным образом выбирает одно из шифрованных сообщений, присланных Борисом, и методом тотального перебора получает открытый текст этого сообщения.
3. Антон шифрует свое собственное секретное сообщение при помощи другого симметричного алгоритма на ключе y, вскрытом им методом тотального перебора, и посылает это сообщение вместе с соответствующим x.
4. Борис, зная соответствие между x и y, расшифровывает сообщение, пришедшее ему от Антона.

Конечно же, Петр может вскрыть ключ, который был использован Антоном для зашифрования секретного сообщения, однако для этого Петру придется выполнить значительно больший объем вычислений, чем Антону и Борису. Петр должен будет прочесть все 2^{20} шифрованных сообщений, отправленных Борисом, чтобы найти y, соответствующее значению x, выбранному Антоном. На решение этих "шарад" Петр потратит примерно квадрат времени, которое потребуется Антону и Борису, чтобы их составить. И хотя такое от-

личие в трудоемкости по криптографическим меркам довольно невелико, в некоторых случаях этого может оказаться достаточно. Например, если Антон и Борис смогут опробовать по 10 тыс. ключей в секунду, им понадобится немногим более 1 минуты, чтобы ключ шифрования у оказался в руках Антона. В то же время у Петра, обладающего примерно такой же вычислительной мощью для опробования ключей, как Антон и Борис, на вскрытие правильного ключа уйдет больше года.

Цифровая подпись

Собственноручная подпись под документом с давних пор используется людьми в качестве доказательства того факта, что человек, подписавший данный документ, ознакомился и согласен с его содержанием. Почему же подпись заслужила такое доверие со стороны человечества? Перечислим основные причины.

- Подлинность подписи можно проверить. Ее присутствие в документе позволяет убедиться, действительно ли он был подписан человеком, который обладает правом ставить эту подпись.
- Подпись нельзя подделать. Наличие подлинной подписи является доказательством того, что именно тот человек, которому она принадлежит, поставил эту подпись под документом.
- Подпись, которая уже стоит под одним документом, не может быть использована еще раз для подписания второго документа. Иными словами, подпись является неотъемлемой частью документа и не может быть перенесена в другой документ.
- Подписанный документ не подлежит никаким изменениям.
- От подписи невозможно отречься. Тот, кто поставил подпись под документом, не может впоследствии заявить, что он не подписывал этот документ.

На самом деле, ни одно из перечисленных свойств подписи не выполняется на все 100%. Подписи подделываются и копируются, от них отрекаются, а в уже подписанные документы впоследствии вносятся произвольные изменения. Однако люди вынуждены мириться с недостатками, присущими подписи, поскольку мошеннические трюки с подписями проделывать не так-то просто и шансы быть пойманными у таких мошенников достаточно велики.

Попытка использовать подпись в компьютерных файлах сопряжена с еще большими трудностями. Во-первых, любой файл может быть скопирован вместе с имеющейся в нем подписью. Во-вторых, после подписания в файл можно внести любые изменения, которые в принципе не поддаются обнаружению.

Подписание документов при помощи симметричных криптосистем и арбитра

Предположим, что Антон должен послать подписанное им цифровое сообщение Борису. Для этого потребуются симметричная криптосистема и арбитр Дмитрий, который может обмениваться шифрованной информацией и с Антоном, и с Борисом. Чтобы общаться с Антоном, Дмитрий использует ключ K_A, а сообщения, предназначенные Борису, Дмитрий шифрует с помощью ключа $K_Б$. Считается, что передача ключей всем заинтересованным сторонам произошла еще до того, как у Антона появилась потребность отправить Борису подписанный цифровой документ. Тогда Антон, Борис и Дмитрий могут действовать в соответствии со следующим протоколом:

1. Антон шифрует свое сообщение Борису на ключе K_A и посылает это сообщение Дмитрию.
2. Дмитрий расшифровывает полученное сообщение, присоединяет к нему собственное заявление о принадлежности этого сообщения Антону, шифрует сообщение Антона и свое заявление на ключе $K_Б$, а затем отправляет их Борису.
3. Борис расшифровывает сообщение, пришедшее от Дмитрия. В результате Борис может прочесть как открытый текст сообщения Антона, так и подтверждение того факта, что этот текст принадлежит Антону.

Дмитрий уверен в том, что автором полученного сообщения является Антон, поскольку лишь Антон владеет секретным ключом K_A и, следовательно, только он мог зашифровать это сообщение с помощью K_A.

Данный протокол позволяет удостоверять принадлежность цифрового документа конкретному лицу подобно тому, как это делается при помощи анализа его собственноручной подписи, но при одном важном условии: Дмитрий пользуется абсолютным доверием со стороны всех без исключения участников протокола, поэтому что бы он ни сказал, принимается остальными на веру без тени сомнения.

Если Борис захочет продемонстрировать документ, подписанный Антоном, кому-нибудь еще, например, Владимиру, им всем снова придется прибегнуть к услугам вездесущего и кристально честного Дмитрия:

1. Борис шифрует сообщение Антона и заявление Дмитрия о том, что это сообщение действительно пришло от Антона, с помощью ключа $K_Б$ и отсылает их Дмитрию.
2. Дмитрий расшифровывает полученное от Бориса сообщение, заглядывает в свой архив и убеждается, что Борис не внес никаких изменений в исходный открытый текст сообщения Антона.
3. Дмитрий зашифровывает сообщение Бориса с помощью ключа K_B, используемого для связи с Владимиром, и шлет это сообщение Владимиру.

4. Владимир расшифровывает сообщение Дмитрия и получает как открытый текст сообщения Антона, так и подтверждение того факта, что этот текст на самом деле принадлежит Антону.

Заметим, что от протокола к протоколу у Дмитрия появляется все больше обязанностей. Он шифрует и расшифровывает сообщения, сравнивает их между собой, дописывает к ним заявления об их авторстве, ведет архив сообщений и разрешает возникающие конфликты. При этом Дмитрий не может допустить ни малейшей ошибки, поскольку иначе он безвозвратно утратит доверие остальных участников протокола. Да и проникновение злоумышленника Зиновия в архив Дмитрия чревато последствиями, которые надолго отобьют у всех охоту ставить свои подписи под цифровыми документами. Поэтому практическое использование подписей, которые заверяются при помощи симметричной криптосистемы и арбитра, является весьма ограниченным.

Подписание документов при помощи криптосистем с открытым ключом

Протокол подписания документа при помощи криптосистемы с открытым ключом достаточно незатейлив:

1. Антон шифрует документ с использованием тайного ключа, тем самым проставляя под этим документом свою подпись, и отправляет его Борису.
2. Борис расшифровывает документ с использованием открытого ключа Антона, тем самым проверяя подлинность подписи.

От Дмитрия здесь совершенно не требуется вести обширную шифрованную переписку с участниками протокола. Он даже не нужен, чтобы проверять подлинность подписи. Если Борис не сможет расшифровать документ с использованием открытого ключа, принадлежащего Антону, значит, подпись под этим документом недействительна. Тем не менее, Дмитрий понадобится, чтобы удостовериться, что открытый ключ Антона действительно принадлежит именно Антону.

Отметка о времени подписания документа

К сожалению, при некоторых обстоятельствах Борис может обмануть Антона, если воспроизведет точную копию документа вместе с подписью, поставленной под ним Антоном. Еще одна копия документа, который представляет собой обычный деловой договор между двумя сторонами, погоды не сделает. Однако если у Бориса в руках окажется точная копия чека на кругленькую сумму, выписанного Антоном, последний может понести значительный материальный ущерб.

Поэтому, как и в обыденной жизни, на все документы обычно ставятся дата и время их подписания, которые, в свою очередь, также снабжаются цифровой подписью наравне с самим документом. В результате банк сможет разо-

блачить попытку дважды обналичить один и тот же чек, а у смошенничавшего Бориса в тюрьме будет много свободного времени, которое он сможет посвятить более тщательному изучению криптографических протоколов.

Использование однонаправленных хэш-функций для подписания документов

На практике криптосистемы с открытым ключом не всегда эффективны при подписании документов значительного объема. В целях экономии времени можно подписывать не сам документ, а хэш-значение, вычисленное для этого документа при помощи однонаправленной хэш-функции. Участники протокола должны только заранее условиться о том, какой алгоритм шифрования с открытым ключом и какую однонаправленную хэш-функцию они будут использовать для подписания документов:

1. Антон подвергает документ хэшированию при помощи однонаправленной хэш-функции.
2. Антон шифрует вычисленное хэш-значение с использованием собственного тайного ключа, тем самым ставя под документом свою подпись, и отправляет это хэш-значение Борису в зашифрованном виде вместе с документом.
3. Борис вычисляет хэш-значение документа, расшифровывает хэш-значение, присланное ему Антоном, с использованием открытого ключа Антона. Если два полученных хэш-значения совпадают, то подпись Антона под документом верна.

Для подписания документа непременно должна быть использована однонаправленная хэш-функция, поскольку в противном случае злоумышленник Зиновий окажется в состоянии сгенерировать множество документов, хэш-значения которых совпадут. А тогда подписание Антоном одного конкретного документа станет равносильно подписанию им всех документов, имеющих такие же хэш-значения.

Кроме того, при использовании однонаправленных хэш-функций для подписания документов подпись можно хранить отдельно. А следовательно в архиве, в котором данный протокол применяется для проверки подлинности подписи, достаточно хранить только зашифрованные хэш-значения файлов. При возникновении разногласий относительно авторства документа достаточно найти в архиве соответствующее шифрованное хэш-значение, расшифровать его и проверить, совпадает ли оно с хэш-значением, вычисленным для спорного документа.

Дополнительная терминология

Криптографы придумали множество алгоритмов цифровой подписи. Все они основаны на криптосистемах с открытым ключом. При этом секретная информация используется для того, чтобы поставить подпись под докумен-

том, а общедоступная — чтобы проверить подлинность этой подписи. Поэтому часто процесс подписания документа называют *шифрованием на тайном ключе*, а процесс проверки подлинности подписи — *расшифрованием на открытом ключе*. Однако это справедливо только для одного алгоритма шифрования с открытым ключом — RSA, а подавляющее большинство остальных алгоритмов цифровой подписи совершенно непригодны для шифрования сообщений.

По этой причине в ходе дальнейшего изложения о процессах подписания документа и проверки подлинности подписи будет говориться без упоминания конкретных алгоритмов, которые используются для этих целей. Подписание документа P при помощи тайного ключа K будет обозначаться $S_K(P)$, а проверка подлинности подписи с использованием соответствующего открытого ключа — $V_K(P)$.

Битовая строка, которая присоединяется к документу при его подписании (например, хэш-значение файла, зашифрованное на тайном ключе) будет именоваться цифровой подписью. И наконец, протокол, при помощи которого получатель сообщения убеждается в подлинности и целостности этого сообщения, будет называться *аутентификационным протоколом*.

Несколько подписей под одним документом

Каким образом Антон и Борис могут поставить свои цифровые подписи под одним и тем же документом? Если не задействовать однонаправленные хэш-функции, существуют 2 способа сделать это.

- Первый заключается в создании двух идентичных копий документа, одну из которых подписывает Антон, а другую — Борис. Однако тогда придется хранить документ, длина которого в 2 раза превышает размер исходного документа, предназначенного для совместного подписания Антоном и Борисом.
- Второй способ состоит в том, чтобы сначала документ подписал Антон, а затем подпись Антона заверил своей подписью Борис. Но в этом случае будет невозможно убедиться в подлинности подписи Антона, не проверив подпись Бориса.

Если использовать однонаправленную хэш-функцию, от перечисленных недостатков можно легко избавиться:

1. Антон подписывает хэш-значение документа.
2. Борис подписывает хэш-значение того же самого документа.
3. Борис отсылает свою подпись Антону.
4. Антон шлет документ вместе со своей подписью и подписью Бориса Владимиру.
5. Владимир проверяет подлинность подписей Антона и Бориса.

Неоспоримость

Вполне возможна ситуация, при которой Антон, подписав некоторый документ, впоследствии попытается ее оспорить — заявит о том, что подпись под документом поставил не он. Причиной может послужить, например, потеря Антоном своего тайного ключа в людном месте или намеренная его публикация Антоном в разделе частных объявлений популярной газеты. Эта ситуация называется *отказ от обязательств*.

Уменьшить ущерб отказа от обязательств помогает введение в подпись даты и времени, когда она была поставлена под документом. Конечно, ничего нельзя поделать в случае, если Антон заранее предпринял определенные действия для создания условий, при которых его подпись под документом должна будет признана недействительной. Но можно, по крайней мере, сделать так, чтобы эти действия не позволили Антону объявить поддельными все остальные его подписи, которые он ранее поставил под другими документами:

1. Антон подписывает документ.
2. Антон генерирует заголовок, в который помещает свои личные данные, присоединяет этот заголовок к подписанному им документу, еще раз подписывает итоговый документ и посылает его Дмитрию.
3. Дмитрий проверяет вторую подпись Антона, добавляет к его сообщению отметку о дате и времени получения этого сообщения, подписывает его, а затем отправляет Антону и Борису.
4. Борис проверяет подпись Дмитрия, персональные данные Антона и его подпись.
5. Антон знакомится с содержанием сообщения, присланного Дмитрием. Если Антон обнаруживает в этом сообщении что-то подозрительное, он должен немедленно заявить об этом во всеуслышание.

Цифровая подпись и шифрование

Аналогии из повседневной жизни помогают лучше понять, как должны быть устроены криптографические протоколы, чтобы обеспечивать максимальную защиту от мошенников и злоумышленников. Возьмем, к примеру, обыкновенное письмо, приготовленное Антоном для отправки Борису по почте. Антон всегда подписывает свои письма Борису, прежде чем вкладывает их в конверты. А почему бы Антону не подписывать сами конверты? Да потому, что получив конверт с подписью Антона и обнаружив в нем неподписанное письмо, Борис не сможет убедиться в том, что подлинное письмо Антона не было подменено на пути следования от отправителя к адресату.

Аналогичная ситуация сложилась и в криптографии: если Антон хочет послать шифрованное сообщение, завизированное своей подписью, то ему

лучше сначала подписать открытый текст этого сообщения и только затем зашифровать его вместе с поставленной под ним подписью.

1. Антон подписывает сообщение P при помощи своего тайного ключа K_A:

   $$S_{K_A}(P)$$

2. Антон шифрует подписанное сообщение, используя открытый ключ Бориса $K_Б$, и посылает его Борису

   $$E_{K_Б}(S_{K_A}(P))$$

3. Борис расшифровывает сообщение Антона с помощью своего тайного ключа $K_Б$:

   $$D_{K_Б}(E_{K_Б}(S_{K_A}(P))) = S_{K_A}(P)$$

4. С помощью открытого ключа Антона K_A, Борис проверяет подпись Антона и получает открытый текст его сообщения:

   $$V_{K_A}(S_{K_A}(P)) = P$$

В результате подпись становится неотъемлемой частью сообщения (злоумышленник не может вместо шифрованного сообщения подставить свое собственное, сохранив под ним подлинную подпись отправителя). Кроме того, отправитель видит открытый текст сообщения, когда ставит под ним свою подпись.

Антон может использовать разные ключи для зашифрования документа и для того, чтобы поставить под ним подпись. Это позволит ему при необходимости передать ключ шифрования правоохранительным органам, не компрометируя собственную подпись, а также по своему выбору отдать один из двух ключей на хранение доверенным третьим лицам, сохранив второй в тайне от посторонних.

Однако совместное использование шифрования и цифровой подписи таит в себе особую опасность, если для зашифрования и генерации цифровой подписи используется один и тот же криптографический алгоритм. Рассмотрим для примера протокол, согласно которому после приема сообщения Борис немедленно должен послать отправителю этого сообщения подтверждение о его получении.

1. Антон подписывает сообщение при помощи своего тайного ключа, шифрует это сообщение, используя открытый ключ Бориса, и посылает его Борису:

   $$E_{K_Б}(S_{K_A}(P))$$

2. Борис расшифровывает сообщение Антона на своем тайном ключе, удостоверяется в подлинности подписи Антона с помощью его открытого ключа и восстанавливает открытый текст сообщения:

   $$S_{K_A}(D_{K_B}(E_{K_Б}(S_{K_A}(P)))) = P$$

3. Борис подписывает сообщение Антона с использованием своего тайного ключа, шифрует это сообщение вместе со своей подписью на открытом ключе Антона и отсылает обратно:

$$E_{K_A}(S_{K_Б}(P))$$

4. Антон расшифровывает сообщение Бориса при помощи своего тайного ключа и проверяет подлинность подписи Бориса, используя его открытый ключ. Если в результате он получит сообщение, идентичное тому, которое он ранее отправил Борису, значит, сеанс связи прошел корректно.

К сожалению, если для шифрования и проверки цифровой подписи применяется одинаковый криптографический алгоритм (т. е., $V_K=E_K$ и $S_K=D_K$), Зиновий, являясь законным пользователем сети связи, при определенных условиях сможет читать шифрованную переписку Антона. Для этого, перехватив сообщение Антона $E_{K_Б}(S_{K_A}(P))$, Зиновий шлет его от своего имени Борису. Борис после расшифрования и проверки цифровой подписи Зиновия получит:

$$E_{K_З}(D_{K_Б}(E_{K_Б}(D_{K_A}(P)))) = E_{K_З}(D_{K_A}(P))$$

В соответствии с протоколом Борис должен отправить Зиновию подтверждение правильности приема пришедшего ему сообщения:

$$E_{K_З}(E_{K_Б}(E_{K_З}(D_{K_A}(P))))$$

В результате Зиновию останется только преобразовать сообщение, присланное ему Борисом, при помощи своего тайного ключа и известных открытых ключей Антона и Бориса — и вот он, заветный открытый текст P сообщения, отправленного Антоном!

Конечно, все это стало возможным не только потому, что для шифрования и проверки цифровой подписи использовался один и тот же криптографический алгоритм. Борису следовало бы быть более внимательным и не отсылать подтверждение правильности приема сообщения, присланного Зиновием. Попытка расшифровать это сообщение привела к тому, что Борис получил открытый текст, не имеющий никакого смысла.

Отсюда вывод. Ни в коем случае не пытайтесь подписывать или шифровать произвольные сообщения, а затем делиться полученными результатами с другими людьми. Это для вас может плохо кончиться.

Основные криптографические протоколы

Обмен ключами

Распространенным приемом в криптографии является шифрование каждого передаваемого сообщения с помощью отдельного ключа. Такой ключ называется *сеансовым*, поскольку используется только на протяжении одного се-

анса связи. Каким образом сеансовый ключ попадает в распоряжение отправителя и получателя шифрованного сообщения?

Обмен ключами для симметричных криптосистем

Предположим, что Антон и Борис, являющиеся пользователями компьютерной сети, получили секретные ключи от Дмитрия (доверенного лица, наделенного правами арбитра). Секретные ключи попали к Антону и Борису еще до начала сеанса связи, а злоумышленник Зиновий ровным счетом ничего не знает о том, какие это ключи. Тогда для обмена шифрованными сообщениями по компьютерной сети Антон и Борис могут воспользоваться следующим криптографическим протоколом:

1. Антон связывается с Дмитрием и запрашивает у него сеансовый ключ для связи с Борисом.
2. Дмитрий генерирует случайный сеансовый ключ и создает две шифрованных копии этого ключа — один раз Дмитрий шифрует сеансовый ключ с помощью секретного ключа Антона, второй — с помощью секретного ключа Бориса. Затем Дмитрий отсылает обе копии Антону.
3. Антон расшифровывает свою копию сеансового ключа.
4. Антон отправляет Борису его копию сеансового ключа.
5. Борис расшифровывает свою копию сеансового ключа.
6. Антон отправляет Борису сообщение, зашифрованное с использованием сеансового ключа, копия которого имеется у них обоих.

И Антон, и Борис полностью полагаются на честность Дмитрия. Если Зиновию удастся его подкупить или обмануть, ни о какой секретности обмена сообщениями между Антоном и Борисом не может быть и речи. В этом случае Зиновий получит доступ ко всем ключам, используемым абонентами компьютерной сети, и сможет прочесть шифрованные сообщения, которыми они обмениваются по сети. Для этого Зиновию достаточно аккуратно копировать всю передаваемую через нее информацию.

Другой существенный недостаток этого протокола состоит в том, что арбитр является потенциальным узким местом в обмене сообщениями между Антоном и Борисом. Если Дмитрий по какой-либо причине не сможет вовремя снабдить их ключами, шифрованная связь между ними будет прервана.

Обмен ключами для криптосистем с открытым ключом

Обычно Антон и Борис используют криптографический алгоритм с открытым ключом, чтобы договориться о сеансовом ключе, при помощи которого они будут шифровать свои данные. В этом случае протокол обмена сеансовыми ключами значительно упрощается, и Антон может послать шифрован-

ное сообщение Борису, даже если Борис и не слышал о существовании Антона:

1. Антон находит открытый ключ Бориса в справочнике, в базе данных или получает его от Дмитрия.
2. Антон генерирует случайный сеансовый ключ, шифрует его при помощи открытого ключа Бориса и отправляет ему в зашифрованном виде.
3. Борис расшифровывает сообщение Антона с использованием своего тайного ключа.
4. Антон и Борис обмениваются сообщениями, зашифрованными на одном и том же сеансовом ключе.

Атака методом сведения к середине

Петр, возможности которого ограничены лишь перехватом передаваемых сообщений, может попытаться вскрыть применяемый криптографический алгоритм с открытым ключом или организовать атаку со знанием только шифртекста. По сравнению с Петром у Зиновия выбор гораздо богаче: он может не только "подслушивать" за Антоном и Борисом, но также видоизменять и даже уничтожать сообщения, которыми они обмениваются между собой. Более того, Зиновий может перевоплотиться в Антона, когда общается с Борисом. Он также может попытаться выдать себя за Бориса, вступая в контакт с Антоном. При этом Зиновий может действовать, например, следующим образом:

1. Антон посылает Борису свой открытый ключ. Зиновий перехватывает этот ключ, а вместо него отсылает Борису свой собственный открытый ключ.
2. Борис посылает Антону свой открытый ключ. Зиновий перехватывает этот ключ, а вместо него отсылает Антону свой собственный открытый ключ.
3. Антон шлет Борису сообщение, зашифрованное с помощью открытого ключа, который Антон ошибочно считает принадлежащим Борису. Зиновий перехватывает это сообщение, расшифровывает его при помощи своего тайного ключа, произвольным образом изменяет его, снова зашифровывает на открытом ключе Бориса и затем отсылает адресату.
4. Борис шлет Антону сообщение, зашифрованное с помощью открытого ключа, который Борис ошибочно считает принадлежащим Антону. Зиновий перехватывает это сообщение, расшифровывает его при помощи своего тайного ключа, произвольным образом изменяет его, снова зашифровывает на открытом ключе Антона и затем отсылает адресату.

Даже если в процессе обмена сообщениями Антон и Борис не передают свои открытые ключи по каналам связи, а просто извлекают их из базы данных, Зиновий может оказаться в состоянии перехватывать запросы в эту базу данных и подменять открытые ключи Антона и Бориса своими собст-

венными. А еще Зиновий может проникнуть в базу данных, где хранятся открытые ключи Антона и Бориса, заменить их на свои собственные, а затем преспокойно дожидаться, пока Антон и Борис не захотят обменяться сообщениями.

Такая атака методом сведения к середине стала возможна потому, что Антон и Борис не в состоянии достоверно убедиться в том, что общаются друг с другом напрямую. И если, модифицируя сообщения Антона и Бориса, Зиновий научится делать это с такой быстротой, что задержка в передаче сообщений будет незаметной, ни Антон, ни Борис так никогда и не догадаются, что кто-то вклинился между ними и читает всю их якобы секретную переписку.

Блокировочный протокол

Блокировочный протокол, изобретенный американскими криптологами Р. Райвестом и А. Шамиром, позволяет существенно осложнить жизнь Зиновию и ему подобным личностям:

1. Антон посылает Борису свой открытый ключ.
2. Борис посылает Антону свой открытый ключ.
3. Антон зашифровывает свое сообщение и отсылает половину этого сообщения Борису.
4. Борис зашифровывает свое сообщение и отсылает половину этого сообщения Антону.
5. Антон отсылает вторую половину своего сообщения Борису.
6. Борис расшифровывает сообщение Антона с помощью своего тайного ключа. Борис отсылает вторую половину своего сообщения Антону.
7. Антон расшифровывает сообщение Бориса с помощью своего тайного ключа.

Необходимо, чтобы одну часть сообщения, посланного Антоном или Борисом, нельзя было прочитать, не имея на руках другой его части. Тогда Борис не сможет прочитать сообщение Антона до тех пор, пока не дойдет до шага 6 блокировочного протокола, а Борис не сможет ознакомиться с сообщением Антона до шага 7. Существует несколько способов добиться этого:

- если для шифрования используется блочный криптографический алгоритм, то первую часть отсылаемого сообщения можно, например, составить из первых половин блоков шифртекста, а вторую — из вторых;
- расшифрование сообщения можно поставить в зависимость от случайного битового вектора, который отсылается вместе со второй частью сообщения;
- первая часть сообщения представляет собой хэш-значение, полученное для шифртекста этого сообщения, а вторая часть содержит собственно сам шифртекст.

На шаге 1 блокировочного протокола Зиновий по-прежнему может подставить свой собственный открытый ключ вместо открытого ключа Антона. Однако теперь на шаге 3 Зиновий будет не в состоянии дешифровать перехваченное сообщение Антона и зашифровать его заново при помощи открытого ключа Бориса. Зиновию непременно придется придумать свое собственное сообщение (случайный битовый вектор или хэш-значение) и послать его половину Борису. То же самое касается и сообщения Бориса Антону, которое Зиновий перехватит на шаге 4 блокировочного протокола. А после того как Зиновий получит в свое распоряжение вторые половины подлинных сообщений Антона и Бориса, будет уже слишком поздно изменить придуманные им ранее фальшивые сообщения, которые он уже отослал от их имени. Следовательно, Зиновию придется дать волю своей фантазии и продолжить вести всю переписку от имени Антона и Бориса.

Конечно, может случиться так, что у Зиновия очень живое воображение и что он весьма близко знаком с Антоном и Борисом. В результате они никогда не заподозрят, что на самом деле общаются не друг с другом, а с Зиновием. Однако в любом случае Зиновию будет значительно труднее, чем тогда, когда Антон и Борис используют обычный, а не блокировочный протокол.

Протокол обмена ключами с цифровой подписью

При обмене сеансовыми ключами атаку методом сведения к середине можно попытаться отразить также с помощью цифровой подписи. В этом случае Антон и Борис получают сеансовые ключи, подписанные Дмитрием, который является доверенным лицом, наделенным правами арбитра. Каждый сеансовый ключ снабжается свидетельством о его принадлежности определенному лицу. Получая от Дмитрия ключ, Антон и Борис могут проверить его подпись и убедиться, что этот ключ принадлежит именно тому человеку, которому они собираются послать сообщение.

В результате Зиновию придется весьма несладко. Он не сможет выдать себя за Антона или Бориса, поскольку не знает их тайных ключей. Зиновий будет лишен возможности подменить открытые ключи Антона и Бориса на свои собственные, поскольку в его распоряжении есть только сеансовые ключи с подписью Дмитрия, удостоверяющей их принадлежность Зиновию. Зиновию остается только перехватывать шифрованные сообщения, которыми обмениваются Антон и Борис, и пытаться их читать с помощью криптоаналитических методов.

Однако следует отметить, что в протоколе обмена сеансовыми ключами с цифровой подписью появился новый участник — Дмитрий. Если Зиновий сумеет перевоплотиться в Дмитрия, он сможет подписывать и навязывать участникам протокола известные ему фальшивые сеансовые ключи. Таким образом, несмотря ни на что, у Зиновия все же остается возможность атаковать этот протокол, воспользовавшись методом сведения к середине.

Одновременная передача ключа и сообщения

Антон и Борис совершенно необязательно передают друг другу сеансовые ключи перед тем, как приступить к обмену сообщениями. Вместо этого они могут воспользоваться следующим протоколом:

1. Антон генерирует случайный сеансовый ключ K и с помощью него зашифровывает сообщение M, получая в результате $E_K(M)$.
2. Антон извлекает открытый ключ Бориса B из базы данных.
3. Антон шифрует сеансовый ключ K при помощи B, получая в результате $E_B(K)$.
4. Антон посылает Борису $E_K(M)$ и $E_B(K)$.
5. Борис расшифровывает сеансовый ключ K с помощью своего тайного ключа.
6. Борис расшифровывает сообщение Антона на сеансовом ключе K.

Данный протокол наглядно демонстрирует использование современных гибридных криптосистем на практике. Для достижения большей стойкости в него может быть добавлена цифровая подпись, а также применены другие приемы, которые помогут сделать его более стойким против различных атак.

Множественная рассылка ключей и сообщений

Почему бы Антону не послать шифрованное сообщение сразу нескольким адресатам? Он может сделать это, например так:

1. Антон генерирует случайный сеансовый ключ K и с его помощью зашифровывает сообщение M, получая в результате $E_K(M)$.
2. Антон извлекает открытые ключи Бориса, Владимира и Георгия из базы данных.
3. Антон шифрует K при помощи открытых ключей Бориса, Владимира и Георгия, получая в результате $E_Б(K)$, $E_B(K)$и $E_Г(K)$.
4. Антон посылает шифрованное сообщение $E_K(M)$, а также многократно зашифрованный сеансовый ключ ($E_Б(K)$, $E_B(K)$и $E_Г(K)$) всем, кто пожелает их получить.
5. Только Борис, Владимир и Георгий смогут расшифровать сеансовый ключ K при помощи своих тайных ключей.
6. Только Борис, Владимир и Георгий смогут прочесть шифрованное сообщение Антона с помощью сеансового ключа K.

Аутентификация

Когда Антон проходит процедуру регистрации для получения доступа к глобальной компьютерной сети Internet, каким образом информационно-

коммерческая служба, услугами которой он пользуется, узнает, что это именно Антон, а не Зиновий, пытающийся бесплатно (за счет Антона) попутешествовать по Всемирной паутине? Чаще всего эта проблема решается при помощи пароля: Антон должен ввести некую секретную информацию, известную только ему и его поставщику услуг Internet. Причем вводить эту информацию надо каждый раз при прохождении процедуры регистрации.

Аутентификация при помощи однонаправленных функций

На сервере, который проверяет правильность введенного Антоном пароля, совершенно не обязательно хранить пароли пользователей. Достаточно научить сервер отличать правильные пароли от неправильных. Это легко сделать при помощи однонаправленных функций. Тогда на сервере будут присутствовать не сами пароли, а их хэш-значения:

1. Антон посылает на сервер свой пароль.
2. С помощью однонаправленной функции сервер вычисляет хэш-значение для присланного Антоном пароля.
3. Сервер сравнивает вычисленное хэш-значение с эталоном, который хранится в его памяти, и делает вывод о правильности пароля Антона.

Поскольку при такой схеме аутентификации пользователей уже не требуется держать пароли на сервере, то опасаться, что кто-либо взломает защиту сервера и украдет файл с паролями, нет никаких причин. Список хэш-значений, соответствующих паролям зарегистрированных пользователей, совершенно бесполезен для Зиновия, т. к. найти пароль по его хэш-значению путем обращения однонаправленной функции ему не удастся.

Отражение словарной атаки при помощи "изюминок"

Файл с хэш-значениями, полученными из паролей, может подвергнуться словарной атаке. Составив словарь примерно из 1 млн самых распространенных паролей, Зиновий применит к ним хэш-функцию. В результате он получит файл объемом до 10 Мбайт, который уместится на нескольких дискетах. Далее Зиновий украдет с сервера файл с паролями, зашифрованными с помощью хэш-функции, и сравнит его со своим файлом, содержащим хэш-значения для самых часто используемых паролей. Совпадающие хэш-значения позволят Зиновию определить некоторые из паролей.

Отразить словарную атаку помогает введение в схему аутентификации зарегистрированных пользователей так называемых *изюминок*. "Изюминка" представляет собой случайную битовую строку, которая присоединяется к паролю прежде, чем к нему будет применена хэш-функция. Затем на серве-

ре запоминается как вычисленное хэш-значение, так и соответствующая ему "изюминка".

Если количество возможных "изюминок" достаточно велико, от словарной атаки мало толку. Зиновий должен будет вычислять хэш-значение не только для каждого пароля, но и для каждой "изюминки". Смысл введения "изюминок" в схему аутентификации пользователей состоит в том, чтобы заставить злоумышленника совершать пробное шифрование всех паролей, входящих в составленный им словарь, каждый раз, когда он пытается вскрыть какой-либо отдельный пароль, вместо того чтобы заранее вычислить хэш-значения этих паролей, а затем просто сравнивать полученные значения с эталонными, украденными с сервера.

"Изюма" понадобится много. Во многих версиях операционной системы UNIX используются "изюминки" длиной 12 бит. Этого явно недостаточно: существуют программы, которые при помощи словарной атаки позволяют за неделю вскрыть от 30 до 40% всех паролей, применяемых пользователями UNIX.

Таким образом, схема аутентификации с "изюминками" не может служить панацеей. Она обеспечивает защиту только от словарной атаки всего файла с паролями, зашифрованными с помощью однонаправленной функции. Однако эта схема бессильна, когда мощной и слаженной атаке подвергается отдельный пароль конкретного пользователя.

Схема аутентификации с "изюминками" эффективна и тогда, когда при регистрации на разных серверах используется один и тот же пароль. Но если пароль плохой, лучше он все равно не станет, каким бы количеством "изюма" для его защиты вы ни запаслись.

Периодическая сменяемость паролей

Даже при наличии "изюминок" схема аутентификации пользователей путем проверки их паролей имеет один очень серьезный недостаток. Ведь не исключено, что линия связи, соединяющая персональный компьютер Антона с сервером информационно-коммерческой службы, проходит по территориям 33-х стран, законодательство которых по-разному трактует права своих и иностранных граждан на сохранение в тайне их личной переписки. И поэтому узнать пароль Антона, в принципе, может любой, кто сумеет подключиться к этой линии связи или обратиться в память сервера и узнать пароль, прежде чем для него будет вычислено соответствующее хэш-значение. Чтобы уменьшить ущерб от компрометации пароля, его следует периодически менять. Делается это следующим образом.

Антон генерирует случайное число R и пересылает его серверу, который вычисляет, исходя из этого числа и однонаправленной функции f, значения $x_1=f(R)$, $x_2=f(f(R))$, $x_3=f(f(f(R)))$ и т. д. 101 раз. Первые 100 вычисленных

значений $x_1, x_2, \ldots, x_{100}$ передаются Антону в качестве списка паролей, который он должен хранить в тайне. А x_{101} запоминается на сервере.

Теперь, когда Антон захочет зарегистрироваться для работы на сервере, ему достаточно ввести свое имя и число x_{100}. Сервер вычислит $f(x_{100})$ и сравнит его с x_{101}. В случае совпадения Антону будет разрешен доступ к серверу, который затем заменит x_{101} на x_{100}. А Антон вычеркнет x_{101} из своего списка паролей.

В следующий раз, когда Антон снова захочет получить доступ к серверу, он найдет в списке паролей следующее по порядку не зачеркнутое значение x_j. Сервер вычислит $f(x_j)$ и сравнит с запомненным x_{j+1}.

Злоумышленник Зиновий, взломавший сервер, все равно не сможет узнать очередной пароль Антона x_j по хранимому на сервере значению x_{j+1}, поскольку функция f является однонаправленной. По той же самой причине, даже если Зиновий перехватит x_j, это позволит ему зарегистрироваться под именем Антона всего лишь один раз. А значения $x_1, x_2, \ldots, x_{j-1}$ так и останутся для Зиновия тайной за семью печатями при условии, что Антон достаточно ответственно относится к хранению списка паролей, полученного им с сервера.

Аутентификация при помощи криптосистем с открытым ключом

Ни "изюминки", ни периодически сменяемый пароль не гарантируют защиту от злонамеренных действий Петра, перехватывающего все сообщения Антона, которые тот пересылает по линии связи, соединяющей его с сервером. Решить эту проблему помогает применение криптографии с открытым ключом.

Предполагается, что на сервере имеется файл, в котором хранятся открытые ключи всех пользователей. В первом приближении криптографический протокол, который позволяет производить аутентификацию пользователей, выглядит так:

1. Сервер генерирует случайную битовую строку и посылает ее Антону.
2. Антон шифрует эту строку при помощи своего тайного ключа и отсылает обратно на сервер.
3. На сервере производится расшифрование сообщения, пришедшего от Антона, с помощью его открытого ключа.
4. Если полученный в результате открытый текст идентичен случайной битовой строке, ранее посланной Антону, последний получает доступ к серверу.

Поскольку только Антон знает свой тайный ключ, никто не сможет выдать себя за Антона. Этот тайный ключ, скорее всего, представляет собой длинную битовую строку, которую очень трудно воспроизводить по памяти, и

поэтому в распоряжении Антона должен быть интеллектуальный терминал, надежно защищенный от взлома и способный автоматически осуществлять набор требуемого тайного ключа при шифровании сообщений. Однако несмотря на это дополнительное условие, важнее всего то, что тайный ключ Антона не требуется передавать на сервер ни под каким видом. Следовательно, для надежной аутентификации пользователей совсем не обязательно наглухо защищать от подслушивания канал связи, соединяющий терминал Антона с сервером. Да и информация о пользователях, хранимая на сервере, может быть безо всякой опаски сделана доступной для всех, кто пожелает с ней ознакомиться.

Конечно, со стороны Антона глупо шифровать на своем тайном ключе произвольную битовую последовательность, причем вне зависимости от ее происхождения. Поэтому на практике для аутентификации пользователей обычно используется более сложный криптографический протокол:

1. Антон производит некоторые специальные вычисления на основе предварительно сгенерированной им случайной битовой последовательности и своего тайного ключа. Затем он отсылает вычисленные им значения на сервер.
2. Сервер посылает Антону случайную битовую последовательность, отличную от той, которая была задействована Антоном на шаге 1.
3. Антон производит некоторые специальные вычисления на основе обеих случайных битовых последовательностей (сгенерированной им самим, а также полученной с сервера) и своего тайного ключа. Затем он отсылает вычисленные значения на сервер.
4. На сервере производятся некоторые специальные вычисления на основе присланных Антоном значений и его открытого ключа, чтобы убедиться в том, что именно он является обладателем соответствующего тайного ключа.
5. Если проверка дает положительный результат, Антону разрешается доступ к серверу.

Если Антон, в свою очередь, не доверяет серверу в такой же степени, в какой сервер не доверяет ему, то он может потребовать, чтобы сервер аналогичным образом "удостоверил свою личность", воспользовавшись тем же самым протоколом.

Формальный анализ криптографических протоколов

Криптографические протоколы, предназначенные для аутентификации и обмена ключами, играют очень важную роль в обеспечении надежного функционирования современных компьютерных систем. Усилиями крипто-

логов было создано много таких протоколов, однако по прошествии времени выяснилось, что не все из них позволяют должным образом защитить компьютерную систему от несанкционированного доступа и предотвратить компрометацию ключей. Причем для некоторых протоколов этот временной промежуток оказался довольно значительным и растянулся на несколько лет. Как следствие, возникла настоятельная потребность в разработке формальных методов, которые позволили бы находить изъяны в криптографических протоколах в более короткие сроки. И хотя большинство из этих методов являются универсальными и могут быть использованы для анализа произвольных криптографических протоколов, особое внимание изначально стало уделяться именно протоколам, позволяющим производить аутентификацию пользователей и обмен ключами.

К настоящему времени имеется 4 основных подхода к анализу криптографических протоколов. Первый из них применяет верификационные методы, которые не были специально разработаны для анализа криптографических протоколов. Некоторые из этих методов представляют протоколы в виде конечных автоматов, другие распространяют на них теорию исчисления предикатов первого порядка.

Второй подход заключается в использовании экспертных систем, позволяющих определить, возникают ли в ходе выполнения шагов протокола такие нежелательные события, как компрометация ключа или разглашение пароля. Этот подход дает возможность эффективнее находить в криптографических протоколах конкретные изъяны, однако никоим образом не гарантирует полного отсутствия таковых.

Авторами третьего подхода являются американские криптологи М. Бэрроуз, М. Абади, Р. Нидхэм (M. Burrows, M. Abadi, R. Needham). Они разработали формальную логическую модель, названную по первым буквам их фамилий *БАН-логикой* (*BAN-logic*). Составить с ее помощью общее представление о надежности криптографического протокола нельзя. Однако основное преимущество БАН-логики перед другими подходами заключается в том, что она состоит из набора простых логических правил, которые легко применяются на практике и весьма полезны при обнаружении отдельных изъянов в анализируемых протоколах.

При четвертом подходе выполнение шагов протокола моделируется с помощью алгебраической системы, которая строится исходя из знаний, имеющихся у участников протокола относительно этого протокола. Затем построенная алгебраическая модель подвергается формальному анализу на предмет достижимости некоторых из ее состояний.

В целом формальный анализ криптографических протоколов пока является довольно новой и до конца не сформировавшейся областью криптологии, и поэтому делать далеко идущие выводы о превосходстве какого-то одного из перечисленных четырех подходов еще рановато. Поживем — увидим.

Многоключевая криптография с открытым ключом

В криптографии с открытым ключом используются 2 ключа, один из которых применяется для зашифрования сообщений, а другой — для расшифрования. Вместо этого можно задействовать любое количество ключей. Рассмотрим для примера трехключевую криптосистему с открытым ключом.

Имеется три ключа K_A, $K_Б$ и K_C, которые распределены между участниками процесса обмена сообщениями следующим образом:

Антон	K_A
Борис	$K_Б$
Владимир	K_B
Георгий	K_A и $K_Б$
Денис	$K_Б$ и K_B
Евгений	K_B и K_A

Если Антон зашифрует свое сообщение с помощью ключа K_A, расшифровать его смогут либо Денис, либо Борис совместно с Владимиром, которые имеют в своем распоряжении $K_Б$ и K_B. Сообщение, зашифрованное Борисом, сможет прочесть Евгений. Владимир может зашифровать сообщение, которое в состоянии прочесть Георгий. Сообщение, зашифрованное Георгием при помощи ключа K_A, смогут прочесть Денис. Если Георгий воспользуется ключом $K_Б$, то его сообщение прочитает Евгений, а если применит $K_Б$ и $K_Б$, то с этим сообщением сможет ознакомиться Владимир. Аналогичным образом может зашифровать свое сообщение Денис, и тогда его прочтут либо Антон, либо Георгий, либо Евгений. Всевозможные сочетания ключей шифрования и расшифрования перечислены в табл. 7.2.

Таблица 7.2. *Сочетания ключей шифрования и расшифрования в трехключевой криптосистеме с открытым ключом*

Зашифровано с помощью ключей:	Расшифровывается с помощью ключей:
K_A	$K_Б$ и K_B
$K_Б$	K_A и K_B
K_C	K_A и $K_Б$
K_A и $K_Б$	K_B
K_A и K_B	$K_Б$
$K_Б$ и K_B	K_A

Множественная рассылка шифрованных сообщений

Допустим, что требуется рассылать групповые шифрованные сообщения 100 различным абонентам компьютерной сети связи, причем заранее неизвестно

какие конкретно сообщения надо будет послать отдельным группам этих абонентов. В такой ситуации можно выдать каждому абоненту свой ключ и шифровать все сообщения отдельно. В результате придется зашифровать и послать слишком много сообщений. Можно снабдить каждого абонента ключами, число которых равняется количеству различных групп, составленных из 100 абонентов. Но тогда потребуется слишком много ключей (порядка 2^{100}).

Выход из положения позволяет найти *многоключевая криптография с открытым ключом*. Предположим, что в компьютерной сети связи имеются всего 3 абонента — Антон, Борис и Владимир. Каждый из них получит по 2 ключа: Антон — $K_А$ и $K_Б$, Борис — $K_Б$ и $K_В$, Владимир — $K_В$ и $K_А$. Тогда если потребуется послать сообщение для Антона, его надо зашифровать с помощью $K_С$. Сообщение, предназначенное Борису, следует зашифровать при помощи $K_А$. А если сообщение должен прочесть только Владимир, необходимо использовать $K_Б$. Сообщение для Антона и Бориса шифруется на ключах $K_А$ и $K_В$, для Антона и Владимира — на $K_Б$ и $K_В$, для Бориса и Владимира — на $K_А$ и $K_Б$.

В случае трех абонентов эта схема множественной рассылки сообщений группам абонентов выглядит не слишком впечатляюще, но при 100 абонентах ее преимущества становятся очевидны. Вместо порядка 2^{100} ключей абонентам сети надо раздать всего 100 ключей, и посылать требуется всего одно шифрованное сообщение, а не n, где n — количество абонентов в группе ($n \geq 1$).

Недостаток рассмотренной схемы — необходимо точно указывать имена абонентов, которым предназначено рассылаемое сообщение, чтобы не заставлять всех остальных абонентов сети связи лихорадочно перебирать имеющиеся у них ключи в поисках того, который позволит им правильно прочесть принятое шифрованное сообщение. Не нужно доказывать, что при определенных условиях раскрытие этих имен перед злоумышленниками может быть весьма нежелательным.

Распределение ответственности

Предположим, что вы собираетесь в отпуск. Ваш шеф попросил, чтобы вы сообщили пароль со своего компьютера кому-либо из коллег. Однако вы не можете полностью переложить ответственность за сохранность своих ценных данных ни на одного из товарищей по работе. В этой ситуации вам следует подумать о распределении ответственности.

Существует возможность разделить секретное сообщение на части, каждая из которых не имеет никакого смысла, но если их определенным образом соединить вместе, снова получится исходное сообщение. Таким образом вы сможете поделить свой пароль на части и, уходя в отпуск, оставить каждому из коллег по одной его части, чтобы, только собравшись все вместе, они оказались в состоянии запустить ваш компьютер. Сделать это в одиночку из них не сможет никто.

Простейший криптографический протокол позволяет Дмитрию поровну распределить между Антоном и Борисом ответственность за сохранение сообщения в тайне:

1. Дмитрий генерирует случайную битовую строку R, которая имеет ту же длину, что и исходное сообщение M.
2. Дмитрий складывает M с R по модулю 2 и получает S.
3. Дмитрий вручает R Антону, а S — Борису.

Чтобы восстановить сообщение M в исходном виде, Антон и Борис должны совместно выполнить последний шаг протокола:

1. Антон и Борис складывают R и S по модулю 2 и получают M.

В умелых руках данный протокол является весьма надежным. Знание S или R не позволяет реконструировать M. Дмитрий шифрует сообщение при помощи одноразового блокнота и отдает полученный в результате шифртекст одному человеку, а сам блокнот — другому.

Этот протокол можно легко применять для любого числа участников. Если участников 4, он будет выглядеть следующим образом:

1. Дмитрий генерирует три случайных битовых строки R, S и T, которые имеют ту же длину, что и исходное сообщение M.
2. Дмитрий складывает M, R, S и T по модулю 2 и получает U.
3. Дмитрий вручает R Антону, S — Борису, T — Владимиру, U — Георгию.

Чтобы восстановить сообщение M в исходном виде, Антон, Борис, Владимир и Георгий должны совместно выполнить последний шаг протокола:

1. Антон, Борис, Владимир и Георгий складывают R, S, T и U по модулю 2 и получают M.

Одним из участников протокола является Дмитрий, который наделен неограниченными правами. Например, Дмитрий может зашифровать какую-нибудь абракадабру вместо M, а потом утверждать, что Антон и другие участники протокола являются хранителями настоящей тайны, а не какой-то чепухи. Чтобы разоблачить Дмитрия, им необходимо собраться вместе и восстановить исходное сообщение. А еще Дмитрий может сначала раздать части своего сообщения Антону, Борису, Владимиру и Георгию, а затем сложить M и U по модулю 2 и заявить, что только Антон, Борис и Владимир нужны для восстановления сообщения в исходном виде, а от Георгия можно избавиться. Поскольку сообщение M всецело принадлежит только Дмитрию, он может распоряжаться им, как того пожелает.

Основной недостаток криптографического протокола, распределяющего ответственность за сохранение сообщения в тайне, состоит в том, что если хотя бы один из его участников потеряет доверенную ему часть, будет также безвозвратно утрачено и само сообщение. Если, конечно, Дмитрий не пом-

нит его наизусть. В результате в распоряжении участников протокола останется только длина исходного сообщения. Но вряд ли они захотят довольствоваться только этим.

Существуют ситуации, в которых необходимо уметь восстанавливать секретное сообщение даже в отсутствие некоторых участников протокола. Например, вы пишете программу для управления запуском межконтинентальных ракет. Естественно, вы хотите сделать так, чтобы если офицер, имеющий доступ к "ядерной кнопке", сойдет с ума и захочет уничтожить какой-либо континент, у него это ни в коем случае не получится. У "ядерной кнопки" должны собраться вместе по меньшей мере пятеро сумасшедших офицеров, чтобы с помощью вашей программы стереть в порошок Америку.

Ситуацию можно еще усложнить. Допустим, что для нажатия "ядерной кнопки" в нашей стране нужны 1 генерал и 3 полковника. А если генерал, по причине своего старческого склероза, забыл приехать к "ядерной кнопке" в назначенное время, достаточно и пятерых полковников. При этом, если собрались всего 4 полковника, Америке ничего не грозит до тех пор, пока 5 полковников не окажутся в полном сборе. Или пока генерал не приедет.

Криптологи придумали так называемый пороговый протокол, который позволяет распределять ответственность даже еще более сложным образом. В общем случае берется любое секретное сообщение (пароль, код запуска баллистических ракет, рецепт приготовления "Кока-колы") и делится на n частей (называемых *долями*) так, что для реконструкции исходного сообщения обязательно нужны m из них. Такой протокол более точно именуется *(m, n)-пороговым* протоколом.

Например, при использовании (3,4)-порогового протокола Дмитрий может поделить между Антоном, Борисом, Владимиром и Георгием секретный пароль для включения своего персонального компьютера таким образом, что трое из них, собравшись вместе, будут в состоянии восстановить этот пароль и включить компьютер Дмитрия. Если Владимир неожиданно попадет в больницу в бессознательном состоянии, то же самое смогут сделать Антон, Борис и Георгий. Однако, если Борис в это время будет в командировке, без него Антон и Георгий так и не смогут запустить компьютер Дмитрия.

Распределение ответственности и мошенничество

Существует множество способов мошенничества со стороны как участников порогового протокола, распределяющего между ними ответственность за хранение тайны, так и посторонних лиц. Для примера рассмотрим три сценария такого мошенничества:

1. Антон, Борис, Владимир и Георгий получают от шефа приказ включить персональный компьютер Дмитрия. Все четверо собираются вместе и выполняют операцию сложения имеющихся у них долей по модулю 2. Однако Георгий, которого подкупила конкурирующая фирма, вместо

того чтобы честно прибавить свою долю, выданную ему Дмитрием, прибавляет случайную битовую строку. В результате компьютер Дмитрия отказывается включаться, и никто не может объяснить, в чем тут дело.

2. К Антону, Борису, Владимиру и Георгию, получившим от шефа распоряжение включить персональный компьютер Дмитрия, присоединяется Зиновий, который утверждает, что он явился по приказу шефа для воссоздания пароля, поскольку якобы является законным обладателем соответствующей пятой доли. После того как ничего не подозревающие Антон, Борис, Владимир и Георгий огласили свои доли, Зиновий с торжествующем воплем убегает из комнаты: теперь он может в одиночку реконструировать пароль Дмитрия, т. к. знает все необходимые для этого доли.
3. Вместо того чтобы убежать, Зиновий дожидается, пока все участники протокола по очереди огласят свои доли, а затем называет случайную битовую строку требуемой длины. В этом случае Зиновий не только сумеет узнать все доли, но и никто из участников протокола так и не осознает, что Зиновий является посторонним лицом и не имеет никакого права присутствовать при восстановлении секретного пароля Дмитрия.

Для предотвращения мошенничества можно соответствующим образом усовершенствовать пороговый протокол, однако подробное описание этих усовершенствований выходит за рамки данной книги. Поэтому просто перечислим некоторые из них.

- Можно обойтись без Дмитрия, который делит исходное секретное сообщение на части. В этом случае никто из участников протокола не знает содержание сообщения до тех пор, пока все они не соберутся вместе и не реконструируют его.
- От участников протокола не требуется оглашать свою долю публично. Если секретом является тайный ключ в криптосистеме с открытым ключом, которая используется для цифровой подписи, каждый участник выполняет частичное (долевое) подписание соответствующего документа. После того как это сделает последний из них, под документом появится полная цифровая подпись. При этом ни один из участников так и не узнает долю другого.
- Участники протокола могут убедиться, что на самом деле получили от Дмитрия настоящую долю, позволяющую им принимать участие в реконструкции секретного сообщения наравне с другими участниками протокола. При этом от участников совершенно не требуется собираться вместе и восстанавливать исходное сообщение.
- Можно устроить так, что если более m участников протокола высказались за реконструкцию секретного сообщения и при этом менее n участников были против, сообщение будет восстановлено в первозданном виде, а иначе — не будет.

- Можно добиться, чтобы в случае, если один из участников протокола окажется не в состоянии выполнить необходимые действия для восстановления исходного секретного сообщения, почти мгновенно и без каких-либо существенных накладных расходов будет введен новый протокол, участниками которого станут оставшиеся участники первоначального.

Вспомогательные криптографические протоколы

Отметка о времени создания файла

Существует немало ситуаций, когда людям приходится доказывать, что некий документ уже существовал в определенный момент времени. Например, в споре об авторском праве на изобретение побеждает тот, кто предъявит самую раннюю копию описания сути этого изобретения. Если соответствующая документация оформляется на бумаге, заверенную нотариусом копию документа можно передать на хранение в нотариальную контору. При возникновении конфликта нотариус или другой полномочный представитель юридической конторы засвидетельствует существование документа в определенный момент времени.

В компьютерном мире все значительно сложнее. Файл, в котором хранится документ, может быть скопирован и изменен бессчетное число раз. А поменять отметку о времени последней модификации этого файла по силам даже начинающему пользователю. И никто не сможет просто взглянуть на файл и сказать: "Да, я абсолютно уверен, что данный файл был создан именно 15 марта 1996 г." Необходимо пользоваться специальным криптографическим протоколом, который обладает следующими свойствами:

- файл снабжается отметкой о времени его создания, и эта отметка является неотъемлемой частью файла вне зависимости от физического носителя, используемого для его хранения;
- в файл нельзя внести какие-либо изменения так, чтобы эти изменения остались незамеченными;
- имеющуюся отметку о времени создания файла невозможно поменять на другую.

Отметка о времени создания файла и арбитраж

Файл можно снабдить отметкой о времени его создания при помощи протокола, в котором принимает участие доверенное лицо (Дмитрий):

1. Антон передает копию своего документа Дмитрию.
2. Дмитрий фиксирует время получения копии документа, которую с этого момента хранит у себя, чтобы предъявить по первому требованию Антона.

Теперь при возникновении разногласий относительно времени создания документа Антон обращается к Дмитрию, который демонстрирует копию документа и дает свидетельские показания о том, когда этот документ был им получен.

Данный протокол позволяет добиться желаемой цели, однако имеет ряд существенных недостатков, связанных с тем, что Антон должен отдать копию своего документа на хранение Дмитрию.

- Любой, кто способен перехватывать информацию, передаваемую Антоном по каналам связи, сможет прочесть этот документ. Даже если Антон шифрует все свои сообщения, его документ скорее всего попадет в компьютерную базу данных Дмитрия. Антону остается только надеяться, что Дмитрий достаточно ответственно относится к защите переданных ему на хранение документов от посягательств посторонних лиц.
- При большом числе клиентов требования к объемам хранимых у Дмитрия данных возрастут неимоверно. То же касается и пропускной способности каналов связи, соединяющих Дмитрия с его клиентами.
- Ошибка при передаче документа на хранение Дмитрию или сбой в его компьютере сведут на нет способность Антона доказать, что этот документ уже существовал в определенный момент времени.
- У Антона могут возникнуть проблемы, когда он попытается найти доверенное лицо, столь же честное, как и неподкупный Дмитрий из описанного выше протокола с арбитражем. И тогда у отчаявшегося Антона вполне может возникнуть непреодолимое желание сговориться с нечистым на руку арбитром и поставить на документ фальшивую отметку о времени его создания.

Чтобы избавиться если не от всех, то хотя бы от большинства перечисленных недостатков, можно прибегнуть к помощи однонаправленных функций:

1. Антон вычисляет хэш-значение для своего документа.
2. Антон передает вычисленное хэш-значение Дмитрию.
3. Дмитрий добавляет к этому хэш-значению отметку о времени его получения и ставит свою цифровую подпись под итоговым документом.
4. Дмитрий отсылает подписанное им хэш-значение с проставленной отметкой времени обратно Антону.

Теперь Антону нечего беспокоиться о том, достаточно ли надежно хранит Дмитрий доверенные ему документы. Дмитрию вообще не надо ничего держать в своей базе данных, и, следовательно, сами собой отпадают проблемы, связанные с хранением огромных объемов информации. Антон может сразу проверить полученный от Дмитрия итоговый документ на предмет наличия в нем искажений, внесенных при передаче по каналам связи. Однако по-

прежнему нерешенной остается проблема сговора между Антоном и Дмитрием с целью снабдить документ нужной отметкой о времени его создания.

Связующий протокол

Вряд ли Антон является единственным клиентом Дмитрия. Скорее всего, Дмитрий уже ставил отметки о времени создания на документы других людей, прежде чем с аналогичной просьбой к нему обратился Антон. А это значит, что отметке о времени создания документа Антона предшествует какая-то отметка, внесенная в документ другого человека. Более того, разумно предположить, что в скором времени к Дмитрию поступят и другие документы, которые потребуется снабдить отметкой о времени создания.

Пусть A — это идентификатор Антона, присвоенный ему в компьютерной сети, H_n — хэш-значение документа, для которого Антон хочет получить отметку о времени создания, а T_{n-1} — отметка о времени создания предыдущего документа. Тогда, чтобы затруднить сговор между Антоном и Дмитрием, можно воспользоваться так называемым связующим протоколом:

1. Антон посылает Дмитрию H_n и A.
2. Дмитрий отсылает обратно Антону $T_n = S_K(n,A,H_n,T_n,I_{n-1},H_{n-1},T_{n-1},L_n)$, где L_n представляет собой следующую связующую информацию, подвергнутую хэшированию при помощи функции H вида $L_n = H(I_{n-1},H_{n-1},T_{n-1},L_{n-1})$. Здесь наличие S_k свидетельствует о том, что сообщение подписано цифровой подписью Дмитрия. Присутствие A необходимо, чтобы идентифицировать Антона в качестве получателя этого сообщения. Параметр n задает порядковый номер отметки о времени создания документа, хэш-значение которого Дмитрий получил от Антона. T_n — это сама отметка о времени. Остальные параметры перечисляют идентификатор, хэш-значение, отметку времени и связующую информацию, которые были вычислены для предыдущего документа, присланного Дмитрию для постановки отметки о времени создания.
3. После того как Дмитрий получит следующий документ, на который потребуется поставить отметку о времени создания, он перешлет Антону идентификатор I_{n+1} своего очередного клиента, приславшего этот документ.

Теперь, если кто-то поставит под сомнение время создания документа Антоном, он может связаться с I_{n-1} и I_{n+1}, т. е. с авторами предыдущего и следующего документов, присланных Дмитрию. Если и у них возникнут сомнения относительно времени создания документов, то они, в свою очередь, могут обратиться к I_{n-2} и I_{n+2}. Каждый из них может доказать, что на его документ была поставлена отметка времени, предшествующая отметке следующего клиента и идущая за отметкой предыдущего.

Связующий протокол значительно осложняет сговор между Антоном и Дмитрием. Дмитрий не сможет перенести отметку о времени создания до-

кумента в будущее, поскольку тогда от него потребуется умение это будущее предсказывать — он должен точно знать, какой именно документ будет прислан непосредственно перед этим моментом времени. Точно так же Дмитрий будет не в состоянии сделать отметку о времени создания документа более ранней, чем это есть на самом деле: она должна быть встроена во временную отметку следующего документа, а на этот документ соответствующая отметка уже поставлена. У Антона и Дмитрия остается единственная возможность смошенничать: они могут породить цепочку фиктивных документов до и после документа Антона, причем эта цепочка должна быть настолько длинной, чтобы у проверяющего не хватило терпения изучить ее до самого конца.

Распределенный протокол

Люди не вечны. Поэтому вполне может случиться так, что когда Антон обратится за содействием к I_{n-1}, тот уже успеет отправиться в мир иной, где отметки о времени создания документов вовсе не так важны для его обитателей. В этом случае можно посоветовать Антону воспользоваться распределенным протоколом и встроить в отметку о времени создания своего документа временные отметки, по крайней мере, 10-и других человек. Таким образом, у Антона будет больше шансов найти людей, которые при необходимости смогут помочь удостоверить время создания его документа. Распределенный протокол также позволяет избавиться от услуг Дмитрия:

1. С помощью генератора криптографически надежной псевдослучайной последовательности Антон получает k чисел R_1, R_2, . . . , R_k, в качестве начального значения используя хэш-значение H_n, вычисленное для своего документа,.
2. Антон интерпретирует сгенерированные числа R_1, R_2, . . . , R_k, как идентификаторы абонентов компьютерной сети, и отсылает H_n каждому из них.
3. Все выбранные Антоном абоненты сети добавляют к хэш-значению H_n отметку о времени его получения, подписывают итоговое сообщение своей цифровой подписью и отсылают обратно Антону.
4. Антон собирает и хранит все k подписей вместе с отметкой о времени создания своего документа.

Криптографически надежный генератор псевдослучайных чисел необходим для того, чтобы Антон не смог повлиять на выбор идентификаторов и, тем самым, подобрать людей, с которыми он в состоянии вступить в сговор. Даже если Антон попытается соответствующим образом изменить свой документ, чтобы получить по нему такое хэш-значение, которое позволило бы сгенерировать необходимый Антону набор идентификаторов, его шансы преуспеть в этом занятии будут ничтожны.

Антон может смошенничать, только убедив всех без исключения k участников протокола вступить с ним в сговор. Поскольку эти участники выбираются наугад, вряд ли у Антона что-нибудь получится. Однако чем более коррумпированным является общество, тем больше должно быть значение k.

Дополнительно придется позаботиться о том, как поступать с людьми, которые вовремя не отреагируют на просьбу Антона подписать высланное им хэш-значение. Вероятнее всего, для официального признания правильности отметки о времени создания документа от Антона потребуется собрать не менее K подписей, где значение K фиксируется в законодательном порядке.

Подсознательный канал

Предположим, что Антон и Борис были арестованы и препровождены в тюрьму. Их тюремный охранник Олег не возражает против того, чтобы они обменивались сообщениями, но при одном условии: эти сообщения не должны быть зашифрованы. Вдруг Антон и Борис захотят разработать план совместного побега! Поэтому Олег желает читать все, что Антон и Борис сообщают друг другу.

А еще Олег надеется обмануть Антона, договорившись с Борисом, и в удобный момент подменить настоящее сообщение от одного из них другому на фальшивое. Антону и Борису ничего не остается, как воспользоваться услугами Олега. Ведь иначе они не смогут общаться друг с другом и координировать свои планы. Поэтому Антону и Борису придется организовать между собой так называемый *подсознательный канал*. Пользуясь этим каналом, они смогут передавать секретную информацию, а Олег, который будет читать все их сообщения, так ничего и не заподозрит.

Простым подсознательным каналом может служить число слов в предложениях невинного на первый взгляд текста. Нечетное количество слов соответствует 0, а четное — 1. Таким образом, если мой сообщник подсчитает количество слов в предложениях предыдущего абзаца, то получит секретное сообщение "11010". К сожалению, в этом алгоритме отсутствует ключ, а сам алгоритм является ограниченным — здесь стойкость всецело зависит от сохранения в тайне самого алгоритма.

Американский криптолог Г. Симмонс (G. Simmons) первым сумел встроить подсознательный канал в обыкновенную цифровую подпись. В этом случае алгоритм цифровой подписи со встроенным подсознательным каналом будет неотличим для Олега от стандартного алгоритма цифровой подписи. Олег не только не сможет читать сообщения, передаваемые Антоном и Борисом по подсознательному каналу. В его мозгу не мелькнет даже намека на мысль о возможном существовании такового.

Протокол, придуманный Симмонсом для реализации подсознательного канала с помощью цифровой подписи, в общих чертах выглядит так:

1. Антон генерирует произвольное сообщение, выглядящее вполне невинно.
2. При помощи секретного ключа, который имеется и у Бориса, Антон подписывает свое сообщение так, что в подписи прячется подсознательный канал.
3. Антон отдает подписанное сообщение Олегу.
4. Олег знакомится с сообщением Антона, проверяет цифровую подпись. Не обнаружив ничего подозрительного, Олег передает это сообщение Борису.
5. Борис проверяет цифровую подпись Антона под полученным из рук Олега сообщением и убеждается, что оно пришло от Антона.
6. Используя секретный ключ, которым он владеет совместно с Антоном, Борис извлекает секретную информацию, содержащуюся в подсознательном канале.

Что касается надежд Олега подсунуть одному из "друзей по переписке", томящихся в тюрьме, фальшивое сообщение, якобы исходящее от другого, то им не суждено сбыться. Дело в том, что Олег не сможет сгенерировать подлинную цифровую подпись Антона и, следовательно, не в состоянии послать фальшивую информацию по подсознательному каналу. А поскольку у Олега нет секретного ключа, совместно используемого Антоном и Борисом, то он не сможет прочесть ни одного сообщения, которыми они обмениваются. Сообщения, под которыми стоит обыкновенная цифровая подпись, для Олега ничем не отличаются от сообщений с цифровой подписью со встроенным в нее подсознательным каналом.

К сожалению, в некоторых алгоритмах подсознательный канал устроен таким образом, что секретная информация, необходимая Борису, чтобы читать сообщения Антона, совпадает с секретной информацией, имеющейся у Антона и позволяющей ему посылать невинно выглядящие сообщения Борису. В этом случае ничто не может помешать Борису перевоплотиться в Антона и посылать фальшивые сообщения, якобы исходящие от его имени. Однако имеется ряд других алгоритмов, свободных от этого недостатка. В них секретный ключ, находящийся в распоряжении Антона, отличается от ключа, которым пользуется Борис, и, следовательно, Антону не приходится опасаться злоупотреблений со стороны Бориса.

Практическое применение подсознательного канала

На практике подсознательный канал наиболее подходит для организации связи в шпионских сетях. Если в стране, куда заслан шпион, почти все ее граждане регулярно посылают и получают электронные сообщения, снабженные цифровой подписью, то, совершенно не рискуя попасть под подозрение, он может подписывать свои невинно выглядящие сообщения с по-

мощью цифровой подписи, в которую встроен подсознательный канал для передачи шпионских посланий.

Другие области практического применения подсознательного канала менее очевидны. Используя подсознательный канал, Антон может подписать документ под угрозой физической расправы, вставив в него секретное сообщение вида "Меня заставили поставить эту подпись", чтобы иметь возможность впоследствии ее оспорить. Правоохранительные органы могут наносить специальную маркировку на электронную наличность, чтобы отслеживать ее движение. Офисная программа, с помощью которой ставится электронная подпись под документами компании, может иметь двойное назначение и дополнительно использоваться для передачи коммерческих секретов конкурентам.

Подобных возможностей — масса, и поэтому, чтобы воспрепятствовать злонамеренному использованию цифровой подписи для организации подсознательного канала связи, были разработаны специальные алгоритмы, которые не могут быть модифицированы с целью встраивания в них подсознательного канала.

Неоспоримая цифровая подпись

Довольно легко получить точную копию обычной цифровой подписи. Иногда это приходится очень кстати, особенно если цифровая подпись стоит под объявлением, предназначенным для всеобщего ознакомления. Однако свободно циркулирующее личное или деловое письмо, в подлинности которого в состоянии убедиться всякий, может поставить его автора в довольно неловкое положение, не говоря уже об опасности прямого шантажа. Поэтому лучше применять такую цифровую подпись, которую можно проверить, но нельзя продемонстрировать посторонним лицам без согласия ее владельца.

Предположим, что Антон возглавляет компанию "ПрограммКомп", которая торгует универсальным текстовым редактором "УниЛекс'97" собственного изготовления. Чтобы исключить заражение компьютерным вирусом машин, принадлежащих покупателям этого программного продукта, "ПрограммКомп" снабжает каждую копию "УниЛекса'97" цифровой подписью. Однако необходимо, чтобы только легальные покупатели "УниЛекса'97" могли проверить подлинность цифровой подписи. Если в купленной на законных основаниях копии "УниЛекса'97" вдруг обнаружится вирус, "ПрограммКомп" не сможет оспорить подлинность своей цифровой подписи.

Для этого "ПрограммКомп" должна воспользоваться *неоспоримой цифровой подписью*. Подобно обыкновенной цифровой подписи, неоспоримая подпись определяется содержанием подписанного документа и зависит от тайного ключа ее автора. Основное отличие неоспоримой цифровой подписи от обыкновенной состоит в том, что неоспоримую подпись нельзя проверить

без согласия поставившего ее человека. С этой точки зрения ее лучше было бы назвать "подписью без права передачи". Тем не менее, свое название неоспоримая цифровая подпись получила благодаря другому своему свойству: если Антон будет вынужден либо признать подлинность предъявленной ему неоспоримой цифровой подписи, либо отвергнуть ее, то он не сможет отказаться от нее в случае, если эта подпись является настоящей.

Математический аппарат, используемый для конструирования неоспоримой цифровой подписи, довольно сложен и громоздок, однако основная идея соответствующего протокола достаточно прозрачна:

1. Антон знакомит Бориса со своей цифровой подписью.
2. Борис генерирует случайное число и передает его в распоряжение Антона.
3. На основе полученного от Бориса случайного числа и собственного тайного ключа Антон выполняет специальные вычисления, результат которых отсылает обратно Борису. Антон может сделать это только в том случае, если цифровая подпись является подлинной.
4. Борис подтверждает получение вычисленного Антоном значения.

С помощью дополнительных математических ухищрений можно добиться, чтобы Антон мог доказать, что он не подписывал документ, авторство подписи под которым необоснованно приписывается ему, и не смог "отвертеться", если подпись действительно принадлежит ему.

Один из недостатков протокола, позволяющего ставить неоспоримую цифровую подпись под документом, состоит в том, что Борис не в состоянии убедить третьего участника этого протокола (Владимира) в подлинности цифровой подписи Антона. А все потому, что Владимир не имеет возможности удостовериться, насколько случайно число, сгенерированное Борисом на шаге 2. С таким же успехом Борис мог выполнить все шаги протокола в обратном порядке без участия Антона, а затем продемонстрировать Владимиру полученный результат. Владимиру необходимо принять участие во всех шагах протокола самому (вместо Бориса), чтобы проверить 100-процентную подлинность цифровой подписи Антона.

В результате Борис может на законных основаниях приобрести копию "УниЛекса'97", сделать с нее пиратскую копию и перепродать Владимиру. Как только Владимир захочет проверить подлинность цифровой подписи под пиратской копией "УниЛекса'97", Борис обратится к Антону с просьбой произвести проверку подписи под купленной легально копией этого текстового редактора. Сгенерированное Владимиром случайное число Борис переадресует Антону, а поступивший от Антона ответ он перешлет Владимиру.

Но даже с учетом присущих ей изъянов неоспоримая цифровая подпись может быть с успехом использована там, где Антону не требуется, чтобы кто-либо имел возможность проверять эту подпись бесконтрольно. Например, если Антон продал конфиденциальную информацию, заверенную с

помощью неоспоримой цифровой подписи, то с его стороны вполне естественно потребовать, чтобы только покупатель этой информации мог проверить подлинность подписи Антона.

Цифровая подпись с назначенным конфирмантом

Если дела у компании "ПрограммКомп" резко пойдут в гору и "УниЛекс'97" станут покупать нарасхват, то Антону вскоре придется заниматься одной только проверкой подлинности цифровых подписей на проданных копиях вместо того, чтобы готовить к выпуску новую, более универсальную, версию своего текстового редактора. Поэтому Антон вряд ли откажется, если ему Борис предложит стать его *конфирмантом*, т. е. взять на себя верификацию подписей Антона. Для этого Антон просто будет ставить под копиями "УниЛекса'97" не обыкновенную цифровую подпись, а цифровую подпись с назначенным конфирмантом.

В результате Антон сможет переложить ответственность за проверку подлинности своей цифровой подписи целиком на Бориса, причем даже без его предварительного согласия, не дожидаясь, пока тот сам предложит свои услуги. Для этого Антону достаточно будет просто воспользоваться открытым ключом Бориса. В случае отсутствия Антона по причине командировки в другой город, приступа желчно-каменной болезни или даже безвременной кончины, Борис все равно будет в состоянии в любой момент времени проверить подлинность подписи Антона.

Цифровая подпись с назначенным конфирмантом позволяет найти разумный компромисс между обычной и неоспоримой подписями. С одной стороны, вполне понятно стремление Антона ограничить круг людей, которые могут проверить подлинность его цифровой подписи. С другой стороны, у Антона открываются слишком широкие возможности для злоупотреблений. Например, он может отказаться сотрудничать с кем бы то ни было, просто заявив, что потерял ключи, необходимые для проверки своей цифровой подписи. В этом случае заранее назначенный Антоном конфирмант (Борис) заменит Антона в его отсутствие и произведет все требуемые действия.

Цифровая подпись с назначенным конфирмантом может оказаться полезной не только для компании, торгующей программным обеспечением. Если Борис поместит свой открытый ключ в общедоступный справочник, то его услугами в качестве назначенного конфирманта сможет воспользоваться любой, кто в них нуждается. Взимая небольшую плату за проверку каждой цифровой подписи, Борис сможет обеспечить себе безбедную старость.

Цифровая подпись по доверенности

Кроме проверки своей цифровой подписи, Антон может поручить другому лицу ставить эту подпись вместо себя. В жизни случается всякое, и Антон

не застрахован от несчастного случая. Или, например, попытка продать текстовый редактор "УниЛекс'97" якутским оленеводам может потребовать от Антона посещения районов, в которых компьютерные сети — все еще большая редкость. Каким же образом Антон может передать Борису право подписывать все документы вместо себя, не предоставляя ему доступ к своему тайному ключу?

Для этого Антону следует воспользоваться *цифровой подписью по доверенности*. Тогда в случае необходимости Антон сможет передать Борису доверенность на право ставить свою цифровую подпись. Для этого поставленная по доверенности цифровая подпись должна обеспечивать:

- *Распознаваемость*. Подпись по доверенности всегда можно отличить от оригинальной подписи ее настоящего владельца.
- *Невозможность фальсификации*. Только истинный владелец подписи и его доверенное лицо в состоянии поставить эту подпись под документом.
- *Проверяемость*. Изучение цифровой подписи, поставленной по доверенности, позволяет убедиться, что ее владелец действительно предоставил эту доверенность лицу, поставившему данную подпись.
- *Опознаваемость*. Цифровая подпись по доверенности дает возможность ее владельцу однозначно идентифицировать человека, который поставил эту подпись.
- *Неоспоримость*. Цифровая подпись по доверенности, поставленная под документом, не может быть впоследствии оспорена лицом, на законных основаниях поставившим ее под этим документом.

В некоторых случаях требуется, чтобы возможность однозначно идентифицировать лицо, поставившее цифровую подпись по доверенности, была предоставлена не только законному владельцу подписи, но и любому, кто пожелает это сделать.

Групповые подписи

Групповыми подписями называются цифровые подписи, которые обладают следующими свойствами:

- только члены группы могут подписывать документы;
- владелец документа может убедиться, действительно ли поставленная под этим документом подпись принадлежит одному из членов группы;
- по подписи невозможно определить, кто именно из членов группы ее поставил;
- в случае возникновения разногласий имеется возможность однозначно идентифицировать члена группы, которому принадлежит та или иная подпись.

Поставить групповую подпись под документом можно, например, при помощи следующего протокола с арбитражем:

1. Дмитрий генерирует большое количество пар ключей, состоящих из открытого и соответствующего ему тайного ключа. Каждый член группы получает от Дмитрия свой набор тайных ключей. При этом никакой тайный ключ из одного набора не повторяется в другом.
2. Дмитрий в случайном порядке помещает все открытые ключи в общедоступную базу данных, сохраняя в секрете от посторонних информацию о том, кому из членов группы принадлежит каждый из этих ключей.
3. Чтобы подписать документ, член группы выбирает ключ из списка, предоставленного ему Дмитрием.
4. Чтобы проверить подлинность групповой подписи, достаточно воспользоваться соответствующим открытым ключом, обратившись за ним в базу данных.
5. В случае возникновения спора Дмитрий знает, кому из членов группы принадлежит ключ, с помощью которого была поставлена подпись, вызвавшая разногласия.

Недостатком данного протокола является наличие арбитра, который знает все секретные ключи и может подделать любую подпись. Кроме того, сгенерированное Дмитрием множество пар ключей должно быть очень большим, чтобы воспрепятствовать попыткам выяснить, какой из ключей кому принадлежит.

Цифровая подпись с дополнительной защитой

Предположим, что в распоряжении злоумышленника Зиновия имеется множество высокопроизводительных компьютеров. И все они работают днем и ночью, чтобы вскрыть тайный ключ Антона, которым тот пользуется, подписывая своей цифровой подписью важные финансовые документы. Цель оправдывает средства: ведь в случае успеха Зиновий сможет свободно распоряжаться весьма значительными денежными средствами Антона, подделывая его подпись.

Чтобы противостоять злодейским планам Зиновия, Антону следует воспользоваться цифровой *подписью с дополнительной защитой*. Тогда, если Зиновий вскроет тайный ключ Антона с помощью тотального перебора и подделает его подпись, Антон сможет разоблачить подделку. Идея, которая лежит в основе цифровой подписи с дополнительной защитой, достаточно проста. Каждому открытому ключу ставится в соответствие не один тайный ключ, а некоторое множество тайных ключей. Любой из них позволяет поставить под документом цифровую подпись, но она будет отлична от подписей, изготовленных при помощи других ключей. Антон знает только один тайный ключ, остальные ему неизвестны.

Теперь, если Зиновий накопит достаточно документов, подписанных Антоном, и попробует использовать всю мощь своих компьютеров для нахождения тайного ключа Антона, то самое большее, чего сможет добиться Зиновий, — это отыскание отдельного ключа. А если количество возможных тайных ключей, соответствующих данному открытому ключу, чрезвычайно велико, то Зиновий вряд ли найдет именно тот ключ, который использует Антон для подписи своих документов.

Если Зиновий попытается подделать подпись Антона, воспользовавшись тайным ключом, вскрытым тотальным перебором, то эта подделка будет заметно отличаться от настоящей подписи Антона. И когда дело дойдет до суда, Антон сможет продемонстрировать, что с помощью его собственного ключа получается совсем другая подпись, чем та, которую обманным путем воспроизвел Зиновий. Однако в случае, если Антон окажется не в состоянии доказать суду существование двух различных подписей, это будет означать, что подпись под спорным документом является подлинной.

Таким образом, цифровая подпись с дополнительной защитой позволяет отразить атаку, предпринятую Зиновием, имеющим в своем распоряжении самые современные суперкомпьютеры. Но с бандитом, который проникнет в квартиру Антона и украдет его тайный ключ, или с самим Антоном, подписавшим документ и потом сделавшим вид, что потерял ключ, ничего не сможет поделать даже самая изощренная схема цифровой подписи с дополнительной защитой. В первом случае остается только посоветовать установить более совершенную сигнализацию или обзавестись сторожевой собакой. А во втором — тщательнее выбирать себе деловых партнеров.

Предсказание бита

Непревзойденный маг и волшебник Антон решает публично продемонстрировать свои необычайные интеллектуальные способности. Он точно знает, какую карту выберет из колоды Борис еще до того, как это сделает сам Борис! Поэтому Антон записывает название этой карты на листке бумаги, кладет листок в конверт и запечатывает его. Затем Антон передает запечатанный конверт на хранение кому-то из публики. "Возьми из колоды любую карту", — предлагает Антон Борису. Борис выбирает наугад карту и показывает зрителям: "Туз пик". Антон вскрывает конверт и извлекает из него листок бумаги, на котором, как все могут убедиться, черным по белому написано: "Туз пик". Шквал аплодисментов.

Но как узнать наверняка, что Антон не подменил конверт, прежде чем его вскрыть? Специальный криптографический протокол позволяет Антону сохранить предсказанное им значение, состоящее из одного или нескольких битов, в тайне до тех пор, пока он не пожелает ознакомить с ним остальных. С другой стороны, следуя данному протоколу, Борис сможет удостове-

риться, что Антон не смошенничал и не изменил свой выбор уже после того, как этот выбор был им сделан.

Информация о сделанном Антоном выборе фиксируется в виде обязательства, которое передается на хранение Борису. Обязательство представляет собой значение, которое, после того как Антон познакомит Бориса со своим выбором, Борис может предъявить Антону и потребовать, чтобы Антон преобразовал это значение и получил величину, известную Борису. Если Антон сумеет это сделать, значит мошенничества с его стороны проявлено не было.

Предсказание бита с помощью симметричной криптосистемы

Антон может сделать предсказание, выполнив шаги протокола, который использует симметричную криптосистему:

1. Борис генерирует случайную битовую строку S и посылает ее Антону.
2. Антон присоединяет бит b, значение которого он собирается предсказать, к S, шифрует полученную в результате битовую строку (S,b) при помощи симметричной криптосистемы E_K и некоторого секретного ключа K, а затем отсылает результат обратно Борису.

На этом завершается процесс предсказания. Поскольку Борис не может дешифровать полученное от Антона сообщение $E_K(S,b)$, то предсказание, сделанное Антоном, остается для Бориса тайной за семью печатями.

Теперь, если Антон пожелает ознакомить Бориса со своим предсказанием, они могут продолжить совместное выполнение следующих шагов протокола:

1. Антон посылает Борису ключ K.
2. С помощью K Борис расшифровывает $E_K(S,b)$ и узнает значение b. Чтобы убедиться в честности Антона, Борис проверяет, содержит ли открытый текст расшифрованного сообщения сгенерированную им случайную битовую строку S.

Если бы посланное Борису сообщение изначально не содержало случайной битовой строки, Антон мог бы впоследствии попытаться подобрать другой ключ, использование которого позволило бы ему получить вместо предсказанного значения b противоположное. Поскольку b принимает всего 2 возможных значения, сделать это будет очень просто. Однако при наличии в сообщении случайной битовой строки, сгенерированной Борисом, Антону придется подобрать ключ, который при расшифровании сообщения даст не только инвертированный бит $\overline{b}$, но и S. Если Антон пользуется хорошей симметричной криптосистемой, то его шансы найти такой ключ будут пренебрежимо малы. Следовательно, Антон не сможет изменить предсказанное им значение после того, как сделает свой выбор, выполнив первые 2 шага протокола.

Предсказание бита с помощью однонаправленной функции

Антон может предсказать битовое значение, воспользовавшись свойствами, которыми обладают однонаправленные функции:

- Антон генерирует две случайные битовые строки S_1 и S_2;
- Антон присоединяет к сгенерированным S_1 и S_2 бит b, значение которого собирается предсказать;
- Антон вычисляет значение однонаправленной функции H, используя в качестве аргумента битовую строку (S_1,S_2,b), и вместе с S_1 отсылает Борису полученный результат.

Теперь у Бориса имеется вся необходимая информация, которая не позволит Антону впоследствии изменить предсказанное значение b. Однако проверить правильность предсказания без участия Антона Борис не сможет, поскольку в этом случае ему придется вычислять $H^{-1}(S_1,S_2,b)$.

Когда потребуется, чтобы Антон ознакомил со своим предсказанием Бориса, им обоим необходимо будет продолжить выполнение следующих шагов протокола:

1. Антон посылает Борису исходную битовую строку (S_1,S_2,b).
2. Борис вычисляет $H(S_1,S_2,b)$, а затем сравнивает вычисленное значение и S_1 со значением и случайной битовой строкой, присланными ему Антоном на шаге 3. В случае совпадения b действительно представляет собой значение, предсказанное Антоном на шаге 2.

По сравнению с предсказанием бита при помощи симметричной криптосистемы, при предсказании бита с помощью однонаправленной функции не требуется, чтобы Борис посылал какие-либо сообщения Антону. А Антону необходимо отослать всего одно сообщение, чтобы сделать предсказание, и еще одно, чтобы Борис смог с этим предсказанием ознакомиться.

Теперь, при предсказании бита с помощью однонаправленной функции, Борису уже не обязательно генерировать какие-либо случайные битовые строки, поскольку Антон использует для предсказания однонаправленную функцию и не сможет смошенничать, составив фальшивое сообщение (S_1, S_2', b') такое, что $H(S_1, S_2', b') = H(S_1, S_2, b)$. Антон посылает Борису случайную битовую строку S_1, чтобы у Антона не было возможности путем подбора S_1 и S_2 добиться, чтобы изменилось b и чтобы при этом было сохранено значение $H(S_1, S_2, b)$, которое Антон ранее отослал Борису. Храня S_2 в секрете от Бориса, Антон не дает Борису вычислить значения $H(S_1, S_2, b)$ и $H(S_1, S_2, b')$, а затем определить b, сравнив эти значения с тем, что прислал ему Антон.

Предсказание с помощью генератора псевдослучайных битовых последовательностей

Для предсказания битового значения Антон может использовать генератор псевдослучайных битовых последовательностей:

1. Борис генерирует случайную битовую строку $S_Б$ и посылает ее Антону.
2. Антон вычисляет случайное начальное значение для генератора псевдослучайных битовых последовательностей. Затем для каждого бита присланной Борисом битовой строки $S_Б$ Антон отправляет Борису либо (а) выходное значение генератора псевдослучайных битовых последовательностей, если этот бит равен 1, либо (б) результат сложения выходного значения генератора псевдослучайных битовых последовательностей со значением, предсказанным Антоном, если соответствующий бит $S_Б$ равен 0.

Когда Антон захочет раскрыть Борису свое предсказание, они должны будут перейти к выполнению последних двух шагов протокола:

1. Антон высылает Борису случайное начальное значение, вычисленное им для генератора псевдослучайных битовых последовательностей на шаге 2.
2. Борис повторяет действия Антона, предпринятые им на шаге 2, чтобы убедиться в том, что Антон не смошенничал.

Если сгенерированная Борисом случайная битовая строка является достаточно длинной, а генератор псевдослучайных битовых последовательностей, которым пользуется Антон, в требуемой степени непредсказуем, то у Антона практически нет шансов обмануть Бориса.

Бросание монеты

Для разрешения неожиданно возникшего спорного вопроса Антон и Борис решают бросить монету. Однако и у того, и у другого при себе не оказалось ни одной. Тогда они решают "бросить" монету в уме: сначала Антон загадает, что выпадет — "орел" или "решка", а потом Борис подумает и объявит, какой стороной упала "брошенная" им монета. Спрашивается: могут ли Антон и Борис сделать это так, чтобы полностью быть уверенными в том, что никто из них не смошенничал?

Могут, если воспользуются криптографическим протоколом, который заставит их действовать таким образом, что:

- Антону придется бросить монету прежде, чем Борис попытается предсказать, какой стороной она упадет;
- Антон не сможет изменить результат бросания монеты после того, как услышит, на какую сторону монеты сделал свою ставку Борис;
- Борис не узнает, что выпало — "орел" или "решка", до тех пор, пока не примет окончательное решение и не сообщит о нем Антону.

Бросание монеты с помощью предсказания бита

В этом случае криптографический протокол, которого должны придерживаться Антон и Борис при бросании монеты, выглядит следующим образом:

1. Антон делает предсказание битового значения в соответствии с одной из схем, описанных в разделе "Предсказание бита".
2. Борис пытается догадаться, какое значение предсказал Антон, и информирует о своей догадке Антона.
3. Антон сообщает Борису предсказанное значение. Борис выигрывает, если его догадка была правильной.

Бросание монеты с помощью однонаправленной функции

Если Антон и Борис сумеют заранее договориться об использовании конкретной однонаправленной функции f(x), криптографический протокол бросания монеты будет выглядеть так:

1. Антон выбирает случайное число x и вычисляет значение y=f(x).
2. Антон посылает y Борису.
3. Борис пытается догадаться, является ли x четным или нечетным числом, и сообщает о своей догадке Антону.
4. Антон информирует Бориса о том, какое число x он выбрал.
5. Борис проверяет, действительно ли f(x)=y, а также узнает, была ли верна его догадка.

Здесь все зависит от свойств однонаправленной функции f. Если Антон вдруг сможет найти два числа x и x′ такие, что x является четным, а x′ — нечетным, и при этом y=f(x)=f(x′), то Борис всегда будет в проигрыше. Необходимо также, чтобы наименее значимый бит f(x) не зависел от x. Например, если f(x) будет четным в 90 процентах всех случаев, когда x является четным, Антон будет брать верх над Борисом почти всегда.

Бросание монеты с помощью криптосистемы с открытым ключом

В этом случае от алгоритмов шифрования (E) и расшифрования (D) требуется, чтобы они были коммутативны, т. е.:

$$D_{K_1}(E_{K_2}(E_{K_1}(M))) = E_{K_2}(M),$$

где K_1 и K_2 — криптографические ключи, P — открытый текст сообщения. Для симметричных криптоалгоритмов в общем случае это условие не выполняется, однако существуют алгоритмы шифрования с открытым ключом,

для которых оно верно. Последние могут быть использованы в протоколе бросания монеты:

1. Антон и Борис генерируют каждый для себя по паре ключей, состоящей из открытого и тайного ключа.
2. Антон генерирует две случайные битовые строки P_1 и P_2, одна из которых означает, что при бросании монеты выпал "орел", а другая — что получилась "решка".
3. При помощи своего открытого ключа Антон шифрует сначала P_1, а потом P_2, и отсылает оба полученных в результате шифрсообщения ($E_A(P_1)$ и $E_A(P_2)$) Борису.
4. Борис выбирает одно из присланных ему Антоном шифрсообщений (для этой цели Борис может воспользоваться, например, известной считалкой "Эне, бене, раба, квинтер, минтер, жаба" или сходить за советом к астрологу). Борис шифрует выбранное шифрсообщение с помощью своего открытого ключа и отсылает результат ($E_Б(E_A(P))$, где P — это либо P_1, либо P_2) Антону.
5. Антон расшифровывает пришедшее от Бориса сообщение на своем тайном ключе и посылает то, что у него получилось ($D_A(E_Б(E_A(P)))=E_Б(P)$) обратно Борису.
6. Борис расшифровывает это сообщение Антона ($D_Б(E_Б(P))=P$) и узнает, какой стороной упала монета. Затем Борис шлет P Антону.
7. Антон проверяет, действительно ли P — это одна из тех двух случайных битовых строк, которые он сгенерировал на шаге 2.
8. Чтобы окончательно убедиться в честности друг друга, Антон и Борис обмениваются парами ключей, которые они сгенерировали на шаге 1.

Читателю предоставляется возможность самому доказать, что каждый из участников этого протокола немедленно обнаружит, если другой участник попытается смошенничать.

Интересно отметить, что участники протокола узнают результат подбрасывания монеты не одновременно, а по очереди. Поэтому в некоторый момент времени один из участников знает, как "легла" подброшенная монета, а другой — еще нет. А следовательно, в случае неблагоприятного исхода, тот, кто уже знает результат, может повести себя неспортивно, отказавшись от дальнейшего выполнения шагов протокола.

На практике протоколы бросания монеты часто используются для генерации сеансовых ключей. В этом случае Антон и Борис могут сгенерировать случайную битовую последовательность таким образом, что посторонние окажутся не в состоянии повлиять на ее выбор. Тем не менее при генерации сеансовых ключей с помощью протокола бросания монеты Антону и Борису все равно придется шифровать все свои сообщения, чтобы защититься от возможного подслушивания.

Игра в покер

Подобно тому, как Антон и Борис бросали монету, не имея при себе самой монеты, они могут сыграть в покер без колоды карт. Соответствующий криптографический протокол аналогичен протоколу, который помог Антону и Борису организовать бросание монеты — в обоих необходимо обеспечить, чтобы используемые алгоритмы шифрования и расшифрования были коммутативны.

Отличие между этими протоколами состоит в том, что теперь Антону требуется сгенерировать, зашифровать и отослать Борису не 2 битовых последовательности, а 52 — по числу карт в воображаемой колоде. Среди них Борис случайным образом выбирает 5 битовых последовательностей, шифрует их при помощи своего открытого ключа и посылает Антону. Антон расшифровывает полученные последовательности и шлет обратно Борису, который тоже расшифровывает их. Затем Борис выбирает еще 5 битовых последовательностей и посылает Антону, который опять их расшифровывает. В результате и у Антона, и у Бориса на руках окажется по 5 карт, которыми они и будут играть друг против друга. Если потребуется, дополнительные карты могут быть розданы обоим игрокам по той же схеме.

По окончании игры Антон и Борис открывают свои карты и обмениваются ключами, чтобы иметь возможность проверить, насколько честно они следовали правилам карточной игры. К сожалению, криптографические протоколы, предназначенные для игры в покер, устроены таким образом, что в ходе нее игроки могут слегка мошенничать. Характер этого мошенничества зависит от используемого криптоалгоритма. Например, при шифровании при помощи квадратичных вычетов играющие имеют возможность пометить несколько карт. Не бог весть что, но ведь покер — это такая игра, в которой даже небольшая порция дополнительной информации о сопернике может в конечном счете решить исход игры.

Специальные криптографические протоколы

Доказательство с нулевым разглашением конфиденциальной информации

Антон: "Я знаю пароль для входа в компьютерную сеть Центробанка, рецепт приготовления "Байкала", а также почему Ельцин всегда выглядел так, будто только что проглотил живого лягушонка!"

Борис: "Нет, не знаешь!"

Антон: "Нет, знаю!"

Борис: "Чем докажешь?"

Антон: "Хорошо, я тебе все расскажу".

Антон долго шепчет что-то на ухо Борису.

Борис: "Действительно интересно! Надо сообщить об этом газетчикам!"

Антон: "Е-мое, как же я так лопухнулся..."

К сожалению, в обычных условиях Антон может доказать Борису, что знает какую-либо тайну, единственным способом — рассказав, в чем состоит ее суть. Но тогда Борис автоматически узнает эту тайну и сможет поведать о ней первому встречному. Есть ли у Антона возможность помешать Борису это сделать?

Конечно есть. В первую очередь, Антону не следует доверять свою тайну Борису. Но тогда как Антон сможет убедить Бориса, что действительно входит в число посвященных в эту тайну?

Антону надо воспользоваться *протоколом* доказательства с *нулевым разглашением* конфиденциальной информации. С помощью этого протокола Антон окажется в состоянии доказать Борису, что он обладает некой секретной информацией, однако сообщать данную информацию Борису будет совсем необязательно.

Доказательство носит интерактивный характер. Борис задает Антону серию вопросов. Если Антон знает секрет, то ответит правильно на все заданные ему вопросы. Если не знает, вероятность правильного ответа на каждый из вопросов будет невелика. После примерно 10-ти вопросов Борис сможет твердо узнать, обманывает ли его Антон. При этом шансы Бориса извлечь для себя какую-либо полезную информацию о сути самого секрета практически равны нулю.

Протокол доказательства с нулевым разглашением конфиденциальной информации

Использование доказательства с нулевым разглашением конфиденциальной информации можно пояснить на конкретном примере. Предположим, что имеется пещера. Вход в пещеру находится в точке A, а в точке B пещера разветвляется на две половины — C и D. У пещеры есть секрет: только тот, кто знает волшебные слова, может открыть дверь, расположенную между C и D.

Антону волшебные слова известны, Борису — нет. Антон хочет доказать Борису, что знает волшебные слова, но так, чтобы Борис по-прежнему оставался в неведении относительно этих слов. Тогда Антон может воспользоваться следующим протоколом:

1. Борис стоит в точке A.
2. По своему выбору Антон подходит к двери либо со стороны точки C, либо со стороны точки D.
3. Борис перемещается в точку B.

4. Борис приказывает Антону появиться или через левый проход к двери, или через правый.
5. Антон подчиняется приказу Бориса, в случае необходимости используя волшебные слова, чтобы пройти через дверь.
6. Шаги 1—5 повторяются n раз, где n — параметр протокола.

Допустим, что у Бориса есть видеокамера, с помощью которой он фиксирует все исчезновения Антона в недрах пещеры и все его последующие появления. Если Борис покажет записи всех n экспериментов, произведенных им совместно с Антоном, могут ли эти записи послужить доказательством знания Антоном волшебных слов для другого человека (например, для Владимира)?

Вряд ли. Владимир никогда не сможет удостовериться в том, что Антон каждый раз предварительно не сообщал Борису о своих намерениях, чтобы потом Борис приказывал ему выходить именно с той стороны двери, с какой Антон зашел. Или что из сделанной видеозаписи не вырезаны все неудачные эксперименты, в ходе которых Антон не смог выполнить распоряжения Бориса.

Это означает, что Борис не в состоянии убедить Владимира, лично не присутствовавшего при проведении экспериментов в пещере, в том, что Антон действительно подтвердил свое знание секрета. А значит использованный Антоном протокол доказательства характеризуется именно нулевым разглашением конфиденциальной информации. Если Антон не знает волшебные слова, открывающие дверь в пещере, то, наблюдая за Антоном, не сможет ничего узнать и Борис. Если Антону известны волшебные слова, то Борису не поможет даже подробная видеозапись проведенных экспериментов. Во-первых, поскольку при ее просмотре Борис увидит только то, что уже видел живьем. А во-вторых, потому что практически невозможно отличить сфальсифицированную Борисом видеозапись от подлинной.

Протокол доказательства с нулевым разглашением срабатывает в силу того, что не зная волшебных слов, Антон может выходить только с той стороны, с которой зашел. Следовательно лишь в 50% всех случаев Антон сумеет обмануть Бориса, догадавшись, с какой именно стороны тот попросит его выйти. Если количество экспериментов равно n, то Антон успешно пройдет все испытания только в одном случае из 2^n. На практике можно ограничиться n=16. Если Антон правильно исполнит приказ Бориса во всех 16-ти случаях, значит он и правда знает секрет волшебных слов.

Пример с пещерой является наглядным, но имеет существенный изъян. Борису значительно проще проследить, как в точке В Антон поворачивает в одну сторону, а потом появляется с противоположной стороны. Протокол доказательства с нулевым разглашением здесь попросту не нужен.

Поэтому предположим, что Антону известны не какие-то там волшебные слова, типа "Сезам, откройся". Нет, Антон владеет более интересной ин-

формацией — он первым сумел найти решение этой труднорешаемой задачи. Чтобы доказать данный факт Борису, Антону совсем не обязательно всем и каждому демонстрировать свое решение. Ему достаточно применить следующий протокол доказательства с нулевым разглашением конфиденциальной информации:

1. Антон использует имеющуюся у него информацию и сгенерированное случайное число, чтобы свести труднорешаемую задачу к другой труднорешаемой задаче, изоморфной исходной задаче. Затем Антон решает эту новую задачу.
2. Антон задействует протокол предсказания бита для найденного на шаге 1 решения, чтобы впоследствии, если у Бориса возникнет необходимость ознакомиться с этим решением, Борис мог бы достоверно убедиться, что предъявленное Антоном решение действительно было получено им на шаге 1.
3. Антон показывает новую труднорешаемую задачу Борису.
4. Борис просит Антона или доказать, что две труднорешаемые задачи (старая и новая) изоморфны, или предоставить решение, которое Антон должен был найти на шаге 1, и доказать, что это действительно решение задачи, к которой Антон свел исходную задачу на том же шаге.
5. Антон выполняет просьбу Бориса.
6. Антон и Борис повторяют шаги 1—6 n раз, где n — параметр протокола.

Труднорешаемые задачи, способ сведения одной задачи к другой, а также случайные числа должны по возможности выбираться так, чтобы у Бориса не появилось никакой информации относительно решения исходной задачи даже после многократного выполнения шагов протокола.

Не все труднорешаемые задачи могут быть использованы при доказательстве с нулевым разглашением конфиденциальной информации, однако большинство из них вполне пригодны для таких целей. Примерами могут служить отыскание в связном графе цикла Гамильтона (замкнутого пути, проходящего через все вершины графа только один раз) и определение изоморфизма графов (два графа изоморфны, если они отличаются только названиями своих вершин).

Параллельные доказательства с нулевым разглашением конфиденциальной информации

Обычный протокол доказательства с нулевым разглашением конфиденциальной информации требует, чтобы Антон и Борис последовательно повторили его шаги n раз. Можно попробовать выполнять действия, предусмотренные этим протоколом, одновременно:

1. Антон использует имеющуюся у него информацию и n сгенерированных случайных чисел, чтобы свести труднорешаемую задачу к n другим труд-

норешаемым задачам, изоморфным исходной задаче. Затем Антон решает эти n новых задач.

2. Антон задействует протокол предсказания бита для найденных на шаге 1 n решений, чтобы впоследствии, если у Бориса возникнет необходимость ознакомиться с этими решениями, Борис мог бы достоверно убедиться, что предъявленные Антоном решения действительно были получены им на шаге 1.
3. Антон показывает n новых труднорешаемых задач Борису.
4. Для каждой из n новых труднорешаемых задач Борис просит Антона или доказать, что она изоморфна исходной труднорешаемой задаче или предоставить решение этой задачи, которое Антон должен был найти на шаге 1, и доказать, что оно действительно является ее решением.
5. Антон выполняет все просьбы Бориса.

На первый взгляд параллельный протокол обладает тем же свойством нулевого разглашения конфиденциальной информации, что и обычный. Однако строгого доказательства этого факта еще не найдено. А пока с полной определенностью можно сказать лишь одно: некоторые интерактивные протоколы доказательства с нулевым разглашением в некоторых ситуациях можно выполнять параллельно, и от этого они не утрачивают свойство нулевого разглашения конфиденциальной информации.

Неинтерактивные протоколы доказательства с нулевым разглашением конфиденциальной информации

Постороннего человека, не участвующего в выполнении шагов интерактивного протокола доказательства с нулевым разглашением конфиденциальной информации, невозможно убедить в том, в чем в ходе реализации протокола убеждается Борис, а именно — что Антон действительно владеет конфиденциальной информацией. Чтобы преодолеть этот недостаток, потребуется применить *неинтерактивный* протокол, в котором вместо Бориса используется однонаправленная функция:

1. Антон использует имеющуюся у него информацию и n сгенерированных случайных чисел, чтобы свести труднорешаемую задачу к n другим труднорешаемым задачам, изоморфным исходной задаче. Затем Антон решает эти n новых задач.
2. Антон задействует протокол предсказания бита для найденных на шаге 1 n решений.
3. Антон подает n обязательств, полученных им на шаге 2, на вход однонаправленной функции.

4. Для каждой i-й труднорешаемой задачи, к которой Антон свел исходную задачу на шаге 1, он берет i-й бит значения, вычисленного с помощью однонаправленной функции, и (а) если этот бит равен 1, то Антон доказывает, что исходная и i-я задачи изоморфны, или (б) если этот бит равен 0, то Антон помещает в общедоступную базу данных решение i-й задачи, вычисленное на шаге 1.
5. Антон передает в общедоступную базу данных все обязательства, которые были получены им на шаге 2.
6. Борис, Владимир или любое другое заинтересованное лицо могут проверить правильность выполнения Антоном шагов 1—5.

Удивительно, но факт: Антон предоставляет в общее пользование данные, которые позволяют любому убедиться в том, что он владеет некоторым секретом, и которые одновременно с этим не содержат никакой информации о сути самого секрета.

Роль Бориса в этом протоколе исполняет однонаправленная функция. Если Антон не знает решения труднорешаемой задачи, он все равно может выполнить действия, предусмотренные или пунктом (а), или пунктом (б) шага 4 протокола, но отнюдь не обоими пунктами сразу. Поэтому, чтобы смошенничать, Антону придется научиться предсказывать значения однонаправленной функции. Однако если функция действительно является однонаправленной, Антон не сможет ни догадаться, какими будут ее значения, ни повлиять на нее с тем, чтобы на ее выходе получилась нужная Антону битовая последовательность.

В отличие от интерактивного протокола, здесь требуется большее количество итераций. Поскольку генерация случайных чисел возложена на Антона, подбором этих чисел он может попытаться добиться, чтобы на выходе однонаправленной функции получилась битовая последовательность нужного ему вида. Ведь даже если Антон не знает решения исходной труднорешаемой задачи, он всегда в состоянии выполнить требования или пункта (а), или пункта (б) шага 4 протокола. Тогда Антон может попытается догадаться, на какой из этих пунктов падет выбор, и выполнить шаги 1—3 протокола. А если его догадка неверна, он повторит все сначала. Именно поэтому в неинтерактивных протоколах необходим больший запас прочности, чем в интерактивных. Рекомендуется выбирать $n = 64$ или даже $n = 128$.

Доказано, что в общем случае любое математическое доказательство может быть соответствующим образом преобразовано в доказательство с нулевым разглашением конфиденциальной информации. А это означает, что теперь математику вовсе не обязательно публиковать результаты своих научных исследований. Он может доказать своим коллегам, что нашел решение какой-то математической проблемы, не раскрывая перед ними сути найденного решения.

Удостоверение личности с нулевым разглашением конфиденциальной информации

В повседневной жизни людям регулярно приходится удостоверять свою личность. Обычно они делают это путем предъявления паспортов, водительских прав, студенческих билетов и других подобных документов. Такой документ обычно имеет некоторую индивидуальную отличительную особенность, которая позволяет однозначно связать его с определенным лицом. Чаще всего это фотография, иногда — подпись, реже — отпечатки пальцев или рентгеновский снимок зубов. Можно ли делать то же самое с помощью криптографии?

Конечно. В этом случае для удостоверения личности Антона используется его тайный криптографический ключ. Применяя доказательство с нулевым разглашением конфиденциальной информации, Антон может продемонстрировать любому, что знает свой тайный ключ, и тем самым однозначно идентифицировать себя. Идея цифровой идентификации весьма заманчива и таит в себе массу разнообразных возможностей, однако у нее есть ряд существенных недостатков.

Во-первых, злоумышленник Зиновий под фальшивым предлогом может попросить Антона предъявить свое цифровое удостоверение личности. Одновременно с помощью современных средств связи Зиновий инициализирует процесс идентификации Антона совсем в другом месте и будет переадресовывать все запросы из этого места Антону, а данные им ответы — пересылать обратно. Например, Зиновий может связаться с ювелирным магазином и выдав себя за Антона, оплатить из его кармана весьма дорогую покупку.

Во-вторых, Зиновий может запросто обзавестись несколькими тайными ключами, а следовательно и заиметь соответствующее число цифровых удостоверений личности. Одно из них он использует единственный раз для финансовой аферы и больше им пользоваться не будет. Свидетелем преступления станет человек, которому Зиновий предъявит свое "одноразовое" удостоверение личности, однако доказать, что это был именно Зиновий, не удастся. Ведь предусмотрительный Зиновий никогда не удостоверял таким образом свою личность прежде. Не станет он делать этого и впредь. А свидетель сможет только показать, какое удостоверение личности было предъявлено преступником. Однозначно связать это удостоверение с личностью Зиновия будет нельзя.

В-третьих, Антон может попросить Зиновия одолжить на время его цифровое удостоверение личности. Мол, Антону надо съездить в Соединенные Штаты, а поскольку он — бывший сотрудник советской разведки, работавший против США, американское правительство наотрез отказывает ему во въездной визе. Зиновий с радостью соглашается: после отъезда Антона он

сможет пойти практически на любое преступление, поскольку обзавелся "железным" алиби. С другой стороны, ничто не мешает совершить преступление Антону. Кто поверит лепету Зиновия о том, что он одолжил свое цифровое удостоверение личности какому-то другому человеку?

Избавиться от перечисленных недостатков помогают дополнительные меры предосторожности. В первом случае мошенничество стало возможным, поскольку Зиновий, проверяя цифровое удостоверение личности Антона, мог одновременно общаться с внешним миром по телефону или по радио. Если Зиновия поместить в экранированную комнату без всяких средств связи, никакого мошенничества не было бы.

Чтобы исключить вторую форму мошенничества, необходимо ввести ограничение на количество ключей, которые человеку разрешается использовать, чтобы удостоверить свою личность (как правило, такой ключ должен существовать в единственном числе). И наконец, чтобы не допустить третий вид мошенничества, требуется либо заставить всех граждан удостоверять свою личность как можно чаще (например, у каждого фонарного столба, как это делается в тоталитарных государствах), либо дополнить средства цифровой идентификации другими идентификационными методами (например, проверкой отпечатков пальцев).

Неосознанная передача информации

Предположим, что Борис безуспешно пытается разложить на простые множители 700-битовое число. При этом ему известно, что данное число является произведением семи 100-битовых множителей. На помощь Борису приходит Антон, который случайно знает один из множителей. Антон предлагает Борису продать этот множитель за 1000 руб. — по 10 руб. за бит. Однако у Бориса имеются в наличии лишь 500 руб. Тогда Антон выражает желание отдать Борису 50 бит за половину цены. Борис сомневается, поскольку даже купив эти 50 бит, он все равно не сможет убедиться, что они действительно являются частью искомого множителя, пока не узнает все его биты целиком.

Чтобы выйти из тупика, Антон и Борис должны воспользоваться протоколом неосознанной передачи информации. В соответствии с ним Антон передает Борису несколько шифрованных сообщений. Борис выбирает одно из них и отсылает все сообщения обратно. Антон расшифровывает выбранное Борисом сообщение и снова отсылает Борису. При этом Антон остается в неведении относительно того, какое именно сообщение выбрал для себя Борис.

Протокол неосознанной передачи информации не решает всех проблем, которые стоят перед Антоном и Борисом, желающими заключить сделку о купле-продаже одного из множителей 700-битового числа. Чтобы сделка стала честной, Антон должен будет доказать Борису, что проданные 50 бит действительно являются частью одного из простых множителей, на которые рас-

кладывается это число. Поэтому Антону, скорее всего, придется дополнительно воспользоваться еще и протоколом доказательства с нулевым разглашением информации.

Следующий протокол позволяет Антону послать два сообщения, одно из которых будет принято Борисом, но какое именно, Антон так и не узнает.

1. Антон генерирует две пары ключей, состоящих из открытого и тайного ключа, и отсылает оба открытых ключа Борису.
2. Борис генерирует ключ для симметричного алгоритма (например, для DES-алгоритма), шифрует этот ключ при помощи одного из открытых ключей, присланных Антоном, и отсылает обратно Антону.
3. Антон расшифровывает ключ Бориса с помощью каждого из двух своих тайных ключей, сгенерированных им на шаге 1, и получает две битовых последовательности. Одна из них является подлинным ключом для DES-алгоритма, а другая содержит произвольный набор бит.
4. Антон шифрует два сообщения по DES-алгоритму, используя в качестве ключей обе битовые последовательности, которые были получены им на шаге 3, и отсылает результаты шифрования Борису.
5. Борис расшифровывает оба присланных Антоном сообщения на ключе, сгенерированном на шаге 2, и обретает два открытых текста сообщения, один из которых представляет собой настоящую тарабарщину, а второй — содержательное послание.

Теперь у Бориса имеется одно из двух сообщений Антона, однако последний не может со всей определенностью сказать, какое именно. К сожалению, если в протоколе не предусмотреть дополнительный проверочный шаг, у Антона будет возможность смошенничать (например, зашифровать на шаге 4 два идентичных сообщения). Поэтому необходим еще один, заключительный, шаг протокола:

6. После того как отпала надобность хранить в секрете второе сообщение (к примеру, у Бориса нашлись еще 500 руб., чтобы выкупить у Антона оставшуюся половину множителя), Антон предоставляет Борису свои тайные ключи, чтобы тот мог убедиться в честности Антона.

Протокол защищен от атаки со стороны Антона, поскольку на шаге 3 Антон не в состоянии отличить произвольную битовую последовательность от подлинного ключа DES-алгоритма, сгенерированного Антоном. Протокол также обеспечивает защиту от атаки со стороны Бориса, т. к. у него нет тайных ключей Антона, чтобы определить битовую последовательность, использованную Антоном в качестве ключа DES-алгоритма для шифрования второго сообщения.

Конечно, протокол неосознанной передачи информации отнюдь не гарантирует, что Антон не пошлет Борису какие-нибудь бессмысленные послания, типа "Борис — лох" или "Мяу-мяу", вместо битов одного из семи про-

стых множителей, на которые раскладывается исходное 700-битовое число. Или что Борис вообще захочет с ними ознакомиться и примет участие в выполнении шагов этого протокола.

На практике протокол *неосознанной передачи информации* используется довольно редко. Обычно он служит в качестве одного из строительных блоков для построения других протоколов.

Анонимные совместные вычисления

Иногда бывает так, что группе людей требуется совместно вычислить некоторую функцию от многих переменных. Каждый участник вычислительного процесса является источником значений одной или нескольких переменных этой функции. Результат вычислений становится известен всем членам группы, однако ни один из них не в состоянии выяснить что-либо о значениях, поданных на вход функции другим членом группы.

Вычисление средней зарплаты

Допустим, что начальник отдела приказал своим подчиненным подсчитать среднюю зарплату в отделе. Начальник осведомлен о зарплате любого сотрудника, но слишком занят более важными делами, чтобы отвлекаться на подобные пустяки. Каждый сотрудник прекрасно знает собственную зарплату, но категорически не желает сообщать о ней сослуживцам. Чтобы сотрудники отдела (Антон, Борис, Владимир и Георгий) смогли просуммировать свои оклады, сохранив их в тайне от других, им следует воспользоваться следующим протоколом:

1. Антон генерирует случайное число, прибавляет его к своей зарплате, шифрует полученную сумму при помощи открытого ключа Бориса и затем передает то, что у него получилось, Борису.
2. На своем тайном ключе Борис расшифровывает результат, вычисленный Антоном, прибавляет к нему свою зарплату, шифрует полученную сумму при помощи открытого ключа Владимира и затем передает то, что у него получилось, Владимиру.
3. На своем тайном ключе Владимир расшифровывает результат, вычисленный Борисом, прибавляет к нему свою зарплату, шифрует полученную сумму при помощи открытого ключа Георгия и затем передает то, что у него получилось, Георгию.
4. На своем тайном ключе Георгий расшифровывает результат, вычисленный Владимиром, прибавляет к нему свою зарплату, шифрует полученную сумму при помощи открытого ключа Антона и затем передает то, что у него получилось, Антону.
5. На своем тайном ключе Антон расшифровывает результат, вычисленный Георгием, вычитает из него случайное число, сгенерированное на шаге 1,

делит на количество сотрудников отдела и получает искомую среднюю зарплату в отделе.

Точность вычисления средней зарплаты зависит от честности каждого сотрудника. Если хотя бы один из участников протокола соврет относительно своего жалованья, итоговое значение будет неверным. Особенно большими потенциальными возможностями для злоупотреблений обладает Антон. На шаге 5 он может вычесть любое число, какое только придет ему в голову, и никто не заметит подделки. Поэтому необходимо обязать Антона воспользоваться какой-либо из схем предсказания бита. Однако, если от Антона потребуется раскрыть перед всеми случайное число, сгенерированное им на шаге 1, зарплату Антона узнает Борис. Это значит, что начальнику отдела все же придется отвлечься и самому выполнить вычисления, предусмотренные шагом 2 протокола. Ведь он и так знает зарплату Антона.

Как найти себе подобного

Антон любит играть с резиновыми куклами, изготовители которых потрудились на славу, тщательно скопировав в натуральную величину определенные особенности анатомического строения женщины. А Борису нравится во всех красочных подробностях наблюдать за жизнью соседей из многоквартирного дома напротив при помощи современных оптических приспособлений. Оба тщательно скрывают свои пристрастия от родственников, друзей и коллег по работе, но очень хотели бы найти людей, которые разделяют их интересы.

Фирма "Совместные анонимные вычисления" готова оказать необходимую помощь Антону, Борису и им подобным в подборе таких же чудаков, как они сами. Сотрудники фирмы составили всеобъемлющий список всех человеческих чудачеств, каждое из которых снабжено уникальным идентификатором из семи цифр. Обратившись в фирму, Антон и Борис принимают участие в выполнении шагов некоторого протокола, после чего узнают, испытывают ли они склонность к одним и тем же чудачествам. При положительном ответе они смогут связаться друг с другом. Если ответ будет отрицательным, об их необычных пристрастиях не узнает никто, включая сотрудников "Совместных анонимных вычислений".

Протокол выглядит так:

1. Используя однонаправленную функцию, Антон преобразует 7-значный идентификатор своего чудачества в другое 7-значное число.
2. Трактуя полученное на шаге 1 число как телефонный номер, Борис набирает этот номер и оставляет его абоненту свои координаты. Если на вызов никто не отвечает или такого телефонного номера не существует, Антон применяет к нему однонаправленную функцию и получает новое семизначное число. Так продолжается до тех пор, пока кто-нибудь не ответит на телефонный звонок Антона.

3. Антон сообщает в фирму, сколько раз Борис должен применять однонаправленную функцию, чтобы получить искомый телефонный номер.
4. С помощью однонаправленной функции Борис преобразует 7-значный идентификатор своего чудачества столько раз, сколько это делал Антон, и получает 7-значное число, которое трактует как телефонный номер. Борис звонит по полученному им номеру и спрашивает, нет ли для него какой-либо информации.

Следует отметить, что Борис может предпринять атаку с выбранным открытым текстом. Узнав идентификаторы распространенных человеческих чудачеств, Борис будет по очереди перебирать их, применять к ним однонаправленную функцию и звонить по получающимся у него телефонным номерам. Поэтому необходимо сделать так, чтобы количество возможных чудачеств было достаточно велико, и подобного рода атака стала в результате неосуществимой.

Депонирование ключей

С незапамятных времен одним из наиболее распространенных методов слежки является подслушивание, включающее перехват сообщений, которыми обмениваются люди, являющиеся объектами наблюдения. Сегодня, благодаря широкому распространению стойких криптосистем с открытым ключом, у преступников и террористов появилась возможность обмениваться посланиями по общедоступным каналам связи, не боясь подслушивания со стороны кого бы то ни было. В связи с этим у правоохранительных органов возникла настоятельная необходимость при определенных условиях осуществлять оперативный доступ к открытым текстам шифрованных сообщений, циркулирующих в коммерческих коммуникационных сетях.

В 1993 г. американское правительство впервые публично объявило о своих планах внедрения Стандарта шифрования данных с депонированием ключа. В соответствии с этим стандартом для шифрования данных предполагается использовать защищенную микросхему под названием Clipper, которая снабжается уникальным идентификационным номером и депонируемым ключом. Депонируемый ключ состоит из двух частей, которые раздельно хранятся в двух различных уполномоченных правительственных ведомствах. Для шифрования открытого текста сообщения микросхема генерирует сеансовый ключ. Этот ключ шифруется при помощи депонируемого ключа и в зашифрованном виде присоединяется к шифрованному тексту сообщения вместе с идентификационным номером микросхемы. В случае возникновения необходимости ознакомиться с содержанием сообщения, зашифрованного при помощи микросхемы Clipper, правоохранительным органам достаточно в установленном порядке обратиться в уполномоченные правительственные ведомства за хранящимся там депонируемым ключом,

расшифровать с его помощью сеансовый ключ, а затем прочесть искомый открытый текст сообщения.

В самом общем случае Стандарт шифрования данных с депонированием ключа реализуется с помощью следующего криптографического протокола:

1. Антон генерирует пару ключей, состоящую из открытого и тайного ключа, и делит их на n частей.
2. Антон посылает каждую часть тайного ключа и соответствующую ей часть открытого ключа отдельному доверенному лицу.
3. Каждое доверенное лицо проверяет полученные от Антона части открытого и тайного ключа и помещает их на хранение в надежное место.
4. Если правоохранительные органы добиваются разрешения ознакомиться с перепиской Антона, они обращаются к его доверенным лицам и реконструируют соответствующий тайный ключ.

Существуют различные варианты протокола шифрования данных с депонированием ключа. Например, в него можно встроить пороговую схему, чтобы для восстановления тайного ключа нужно было собрать не все n, а лишь не менее m ($m<n$) частей этого ключа, распределенных Антоном среди своих доверенных лиц. Кроме того, протокол шифрования данных с депонированием ключа можно дополнить действиями, позаимствованными из протокола с неосознанной передачей информации, чтобы доверенные лица не знали, чей конкретно ключ они реконструируют в данный момент по просьбе правоохранительных органов.

Депонирование ключей и политика

После заявления американского правительства о планах внедрения Стандарта шифрования данных с депонированием ключа вокруг крошечной кремниевой пластинки по имени Clipper разразился политический конфликт такой силы, что в печати ее вскоре окрестили "Боснией телекоммуникаций". В непримиримой схватке лицом к лицу сошлись защитники национальной безопасности США и поборники гражданских свобод вместе с поставщиками информационных технологий.

Правительственные агентства США, особенно АНБ, убеждали законодателей в том, что если использование стойких криптосистем станет повсеместным, это даст возможность преступникам и враждебно настроенным странам мешать расследованию их противоправной деятельности. Сторонники депонирования ключей резонно напоминали о том, что успехи криптоаналитических спецслужб англо-американских союзников в значительной мере способствовали достижению победы во второй мировой войне. Важная роль, которую криптоанализ играл в обеспечении безопасности страны, была впоследствии подтверждена и американским послевоенным законодательством, установившим контроль над экспортом шифраторов по тем же правилам,

которые действовали в отношении поставок военного снаряжения. Примеру США последовали все государства, производящие коммерческие программы шифрования, причем некоторые из них (Израиль, Россия и Франция) также стали осуществлять контроль над импортом шифровальных средств и их использованием внутри страны.

Кроме того, в ходе "холодной" войны АНБ неуклонно повышало свой шпионский потенциал. Соответственно росло его влияние на политическую жизнь страны. К началу 90-х годов АНБ превратилось в монстра, внушавшего американским политикам и законодателям благоговейный страх. Для них мнение руководства АНБ было намного весомее возражений его оппонентов. А директор АНБ и его помощники неустанно твердили о том, что недоверие к Стандарту шифрования данных с депонированием ключей вызвано, главным образом, его незнанием. И в этом была доля правды, поскольку значительная часть проекта с самого начала хранилась в строгой тайне, включая сам алгоритм под условным названием Skipjack (Попрыгунчик), используемый для шифрования сообщений.

Со своей стороны, борцы за гражданские права в США посчитали решение, предложенное американской администрацией, еще большим злом, чем проблемы, с которыми приходилось сталкиваться в борьбе с преступностью и терроризмом. По их мнению, это предложение создавало основу для превращения национальной информационной инфраструктуры США в систему тотальной слежки, которая могла быть использована практически бесконтрольно.

Сказано круто. Однако несмотря на их экстремизм, борцов за гражданские права поддержали американские бизнесмены, для которых повсеместное внедрение депонирования ключей представлялось слишком сильнодействующим лекарством против преступности и терроризма. Ведь в качестве побочного эффекта оно было способно серьезно задержать развитие американской информационной инфраструктуры. Стандарт шифрования, базирующийся на секретной технологии и обеспечивающий американцам возможность доступа к каналам связи, которые этот стандарт предназначен защищать, вряд ли может рассчитывать на принятие за пределами США. Для сохранения ведущей позиции США на мировом рынке требуются разработка и поддержка общедоступных стандартов защиты информации, которые в равной мере обеспечивают интересы всех без исключения сторон.

Много неясностей было связано с системой передачи ключей на хранение. Управляющие банками, компьютерными фирмами и представители других отраслей почти в один голос заявили, что покуда ключи находятся у правительства, игра ведется не по правилам. Некоторые из них даже предложили отдавать ключи в руки неправительственных организаций, как это делается, например, в Австралии и Канаде.

Кроме того, депонирование ключей обладает одним неустранимым дефектом. Хорошая практика защиты сообщений с помощью шифрования состо-

ит в том, чтобы хранить ключи к шифрам только в течение короткого периода времени, пока эти ключи действительно нужны. Затем устаревшие ключи уничтожаются, после чего вероятность их воссоздания практически равна нулю. Предложение передавать ключи на хранение лишает шифрование этого преимущества, поскольку требует, чтобы ключи хранились бесконечно долго и могли быть использованы для прочтения более ранних шифрованных сообщений.

В настоящее время несколько влиятельных американских компаний занимаются разработкой и внедрением альтернативных правительственным систем надежного копирования ключей и хранения полученных копий в интересах корпораций и индивидуальных пользователей. Ведь многие из них сталкиваются с проблемой доступа к зашифрованным файлам, когда их работники скоропостижно умирают, увольняются, уезжают в отпуск, заболевают или просто бесследно исчезают. Различие между передачей копий ключей правительству и хранением их в частном агентстве может для кого-то показаться несущественным, но для корпораций эта разница может быть решающей.

Коммерческие программные средства со встроенным депонированием ключей шифруют файлы, используя стандартные алгоритмы шифрования, и завершают эту процедуру добавлением зашифрованной копии использованного ключа в полученный файл. Шифрование ключа производится с помощью другого ключа. Как правило, это открытый ключ службы депонирования. После этого обеспечить доступ к информации в файле могут либо владелец исходного ключа, который был использован для шифрования файла, либо частное агентство, где хранится соответствующий секретный ключ, который был применен для шифрования исходного ключа.

О своей готовности выступить в качестве держателей депонированных секретных ключей заявили многие американские компании, производящие программное обеспечение. Но у них неожиданно появились серьезные конкуренты: расширить пределы своей компетенции до хранения секретных криптографических ключей собрались банки.

Таким образом, в конкурентную борьбу за право хранить у себя резервные копии ключей к чужим шифрам вступили правительство, частные фирмы, банки. Кто из них станет победителем в этом соревновании, — пока неясно. Вероятно, будет найден какой-то компромиссный вариант, который удовлетворит все соревнующиеся стороны.

Глава 8

Надежность криптосистем

Безопасность криптосистем можно сравнить с надежностью цепи: чем крепче ее самое слабое звено, тем труднее порвать эту цепь. В хорошей криптосистеме должно быть досконально проверено все — алгоритм, протокол, ключи и т. п. Если криптографический алгоритм достаточно стоек, а генератор случайных чисел, используемый для порождения ключей, никуда не годится, любой опытный криптоаналитик в первую очередь обратит внимание именно на него. Если удастся улучшить генератор, но не будут зачищены ячейки памяти компьютера после того, как в них побывал сгенерированный ключ, грош цена такой безопасности.

Рассмотрим следующую ситуацию. В криптосистеме применяются стойкий криптографический алгоритм и действительно случайные ключи, которые аккуратно удаляются из памяти компьютера после их использования. Однако перед шифрованием файл, в котором наряду с вашим адресом и фамилией указаны все ваши доходы за текущий год, был по ошибке отправлен электронной почтой в налоговую службу. В этом случае можно спросить, зачем тогда вам понадобились и стойкий алгоритм, и случайные ключи, и зачистка компьютерной памяти в придачу?!

Криптографу не позавидуешь: в проектируемой им криптосистеме он должен предусмотреть защиту от атак всех типов, какие только сможет придумать воспаленное воображение криптоаналитика. Криптоаналитику же наоборот достаточно отыскать единственное слабое звено в цепи криптографической защиты и организовать атаку только против этого звена.

Кроме этого, всегда следует учитывать, что на практике угроза информационной безопасности любого объекта исходит не только от криптоаналитика. В конце концов, каким бы длинным ни был криптографический ключ, используемый вами для шифрования файлов, все равно, если правоохранительным органам понадобится узнать, что хранится в вашем компьютере, они просто установят камеру и скрупулезно запишут всю информацию, появляющуюся на его экране. Недаром, по признанию официальных лиц из

АНБ, большинство сбоев в обеспечении информационной безопасности происходит не из-за найденных слабостей в криптографических алгоритмах и протоколах, а из-за вопиющих оплошностей при их реализации. Какой бы стойкостью ни обладал криптографический алгоритм, ее не требуется преодолевать в лоб, т. к. при успешной атаке ее удается попросту обойти. Однако и пренебрегать хорошими криптографическими алгоритмами тоже не следует, чтобы криптография не стала самым слабым звеном в цепи, которое не выдержит напора атакующего.

Как выбрать хороший криптографический алгоритм

При выборе хорошего криптографического алгоритма можно:

- воспользоваться известным алгоритмом, сравнительно давно опубликованным в специальном издании, посвященном проблемам криптографии (если никто пока не сообщил о том, что сумел вскрыть этот алгоритм, значит на него следует обратить внимание);
- довериться известной фирме, специализирующейся на продаже средств шифрования (вряд ли эта фирма будет рисковать своим добрым именем, торгуя нестойкими криптографическими алгоритмами);
- обратиться к независимому эксперту (непредвзятость во мнении позволит ему объективно оценить достоинства и недостатки различных криптографических алгоритмов);
- обратиться за поддержкой в соответствующее правительственное ведомство (вряд ли правительство будет вводить своих граждан в заблуждение, давая им ложные советы относительно стойкости того или иного криптографического алгоритма);
- попытаться создать собственный криптографический алгоритм.

Все перечисленные варианты имеют существенные изъяны. Не следует полагаться только на одну фирму, на одного эксперта или на одно ведомство. Многие люди, называющие себя независимыми экспертами, мало понимают в криптографии. Большинство фирм, производящих средства шифрования, — тоже ничуть не лучше. В АНБ и ФАПСИ работают лучшие криптографы в мире, однако по понятным соображениям они не спешат поделиться своими секретами с первым встречным. Впрочем, и со вторым тоже. И даже если вы гений в области криптографии, глупо использовать криптографический алгоритм собственного изобретения без того, чтобы его всесторонне не проанализировали и не протестировали опытные криптологи.

Поэтому наиболее предпочтительной представляется первая из перечисленных возможностей. Данный подход к оценке стойкости криптографических алгоритмов можно было бы признать идеальным, если бы не один его не-

достаток. К сожалению, ничего не известно о результатах криптоаналитических исследований этих алгоритмов, которые несомненно активно велись в прошлом и продолжают также активно проводиться во всем мире многочисленными сотрудниками различных правительственных ведомств, в компетенцию которых входят криптологические изыскания. Эти ведомства, скорее всего, гораздо лучше финансируются, чем академические институты, ведущие аналогичные исследования. Да и начали они заниматься криптологией значительно раньше, чем ученые, не имеющие воинских званий, и специалисты из частных фирм. Поэтому можно предположить, что военные нашли гораздо более простые способы вскрытия известных шифров, нежели те, которые изобретены за пределами строго охраняемых зданий сверхсекретных правительственных ведомств.

Ну и пусть. Даже если вас арестуют и в качестве улики конфискуют у вас жесткий диск с файлами, зашифрованными по DES-алгоритму, вряд ли криптоаналитики, состоящие на государственной службе, придут на судебное заседание, чтобы клятвенно подтвердить, что данные для вашего обвинительного заключения получены путем дешифрования конфискованных файлов. Тот факт, что можно вскрыть какой-то конкретный криптографический алгоритм, часто является значительно большим секретом, чем информация, полученная путем вскрытия этого алгоритма.

Лучше исходить из предположения, что АНБ, ФАПСИ и иже с ними могут прочесть любое сообщение, которое они пожелают прочесть. Однако эти ведомства не в состоянии читать все сообщения, с содержанием которых хотят ознакомиться. Главной причиной является ограниченность в средствах, ассигнуемых правительством на криптоанализ. Другое разумное предположение состоит в том, что компетентным органам гораздо легче получить доступ к зашифрованной информации с помощью грубой физической силы, чем путем изящных, но очень трудоемких математических выкладок, приводящих к вскрытию шифра.

Однако в любом случае гораздо надежнее пользоваться известным криптографическим алгоритмом, который придуман уже довольно давно и сумел выстоять против многочисленных попыток вскрыть его, предпринятых авторитетными криптологами.

Криптографические алгоритмы, предназначенные для экспорта из США

В настоящее время у пользователей персональных компьютеров имеется возможность применять алгоритмы шифрования, встроенные в различные программные продукты. Достаточно приобрести, например, текстовый редактор Word, редактор электронных таблиц Excel или операционные системы Windows NT и NetWare. Кроме встроенных алгоритмов шифрования, все эти

программные продукты имеют еще одно общее свойство: они изготовлены в Соединенных Штатах. Прежде чем начать торговать ими за рубежом, американские производители в обязательном порядке должны получить разрешение у своего правительства на экспорт данных продуктов за пределы США.

Многие придерживаются сейчас такого мнения: ни один криптографический алгоритм, разрешенный к экспорту из США, не является достаточно стойким, чтобы его не могли вскрыть криптоаналитики из АНБ. Считается, что американские компании, желающие продавать за рубежом свою продукцию, которая позволяет шифровать данные, по настоянию АНБ переделывают используемые криптографические алгоритмы так, что:

- время от времени отдельные биты ключа подмешиваются в шифртекст;
- ключ имеет длину всего 30 бит вместо официально заявляемых 100 бит, поскольку большинство ключей оказываются эквивалентны;
- в начало каждого шифруемого сообщения вставляется фиксированный заголовок, чтобы облегчить криптоаналитическую атаку со знанием открытого текста;
- любое шифрованное сообщение содержит некоторый фрагмент открытого текста вместе с соответствующим ему шифртекстом.

Исходные тексты американских шифровальных программ передаются на хранение в АНБ, однако за пределами этого сверхсекретного агентства доступ к ним закрыт наглухо. Вполне естественно, что ни АНБ, ни американские компании, получившие от АНБ разрешение на экспорт своих шифровальных средств, не заинтересованы в рекламе слабостей криптографических алгоритмов, положенных в основу функционирования этих средств. Поэтому желательно проявлять осторожность, если вы собираетесь защищать свои данные при помощи американских программ шифрования, экспорт которых за пределы страны разрешен правительством США.

Симметричный или асимметричный криптографический алгоритм?

Какой алгоритм лучше — симметричный или асимметричный? Вопрос не вполне корректен, поскольку предусматривает использование одинаковых критериев при сравнении криптосистем с секретным и открытым ключами. А таких критериев просто не существует.

Тем не менее, дебаты относительно достоинств и недостатков двух основных видов криптосистем ведутся давно, начиная с момента изобретения первого алгоритма с открытым ключом. Отмечено, что симметричные криптографические алгоритмы имеют меньшую длину ключа и работают быстрее, чем асимметричные.

Однако, по мнению американского криптолога У. Диффи — одного из изобретателей криптосистем с открытым ключом — их следует рассматривать не как совершенно новую разновидность универсальных криптосистем. Криптография с открытым ключом и криптография с секретным ключом предназначены для решения абсолютно разных проблем, связанных с засекречиванием информации. Симметричные криптографические алгоритмы служат для шифрования данных, они работают на несколько порядков быстрее, чем асимметричные алгоритмы. Однако криптография с открытым ключом успешно используется в таких областях, для которых криптография с секретным ключом подходит плохо, — например, при работе с ключами и с подавляющим большинством криптографических протоколов.

Шифрование в каналах связи компьютерной сети

Одной из отличительных характеристик любой компьютерной сети является ее деление на так называемые *уровни*, каждый из которых отвечает за соблюдение определенных условий и выполнение функций, необходимых для общения между компьютерами, связанными в сеть. Это деление на уровни имеет фундаментальное значение для создания стандартных компьютерных сетей. Поэтому в 1984 г. несколько международных организаций и комитетов объединили свои усилия и выработали примерную модель компьютерной сети, известную под названием *OSI* (Open Systems Interconnection — Модель открытых сетевых соединений).

Согласно модели OSI коммуникационные функции разнесены по уровням. Функции каждого уровня независимы от функций ниже- и вышележащих уровней. Каждый уровень может непосредственно общаться только с двумя соседними. Модель OSI определяет 7 уровней: верхние 3 служат для связи с конечным пользователем, а нижние 4 ориентированы на выполнение коммуникационных функций в реальном масштабе времени.

Теоретически шифрование данных для передачи по каналам связи компьютерной сети может осуществляться на любом уровне модели OSI. На практике это обычно делается либо на самых нижних, либо на самых верхних уровнях. Если данные шифруются на нижних уровнях, шифрование называется *канальным*, а если на верхних, то такое шифрование называется *сквозным*. Оба этих подхода к шифрованию данных имеют свои преимущества и недостатки.

Канальное шифрование

При канальном шифровании шифруются абсолютно все данные, проходящие по каждому каналу связи, включая открытый текст сообщения, а также

информацию о его маршрутизации и об используемом коммуникационном протоколе. Однако в этом случае любой интеллектуальный сетевой узел (например, коммутатор) будет вынужден расшифровывать входящий поток данных, чтобы соответствующим образом его обработать, снова зашифровать и передать на другой узел сети.

Тем не менее канальное шифрование представляет собой очень эффективное средство защиты информации в компьютерных сетях. Поскольку шифрованию подлежат все данные, передаваемые от одного узла сети к другому, у криптоаналитика нет никакой дополнительной информации о том, кто служит источником этих данных, кому они предназначены, какова их структура и т. д. А если еще позаботиться и о том, чтобы, пока канал простаивает, передавать по нему случайную битовую последовательность, сторонний наблюдатель не сможет даже сказать, где начинается и где заканчивается текст передаваемого сообщения.

Не слишком сложной является и работа с ключами. Одинаковыми ключами следует снабдить только два соседних узла сети связи, которые затем могут менять используемые ключи независимо от других пар узлов.

Самый большой недостаток канального шифрования заключается в том, что данные приходится шифровать при передаче по каждому физическому каналу компьютерной сети. Отправка информации в незашифрованном виде по какому-то из каналов ставит под угрозу обеспечение безопасности всей сети. В результате стоимость реализации канального шифрования в больших сетях может оказаться чрезмерно высокой.

Кроме того, при использовании канального шифрования дополнительно потребуется защищать каждый узел компьютерной сети, по которому передаются данные. Если абоненты сети полностью доверяют друг другу и каждый ее узел размещен там, где он защищен от злоумышленников, на этот недостаток канального шифрования можно не обращать внимания. Однако на практике такое положение встречается чрезвычайно редко. Ведь в каждой фирме есть конфиденциальные данные, знакомиться с которыми могут только сотрудники одного определенного отдела, а за его пределами доступ к этим данным необходимо ограничивать до минимума.

Сквозное шифрование

При сквозном шифровании криптографический алгоритм реализуется на одном из верхних уровней модели OSI. Шифрованию подлежит только содержательная часть сообщения, которое требуется передать по сети. После зашифрования к ней добавляется служебная информация, необходимая для маршрутизации сообщения, и результат переправляется на более низкие уровни с целью отправки адресату.

Теперь сообщение не требуется постоянно расшифровывать и зашифровывать при прохождении через каждый промежуточный узел сети связи. Сообщение остается зашифрованным на всем пути от отправителя к получателю.

Основная проблема, с которой сталкиваются пользователи сетей, где применяется сквозное шифрование, связана с тем, что служебная информация, используемая для маршрутизации сообщений, передается по сети в незашифрованном виде. Опытный криптоаналитик может извлечь для себя массу полезной информации, зная кто с кем, как долго и в какие часы общается через компьютерную сеть. Для этого ему даже не потребуется быть в курсе предмета общения.

По сравнению с канальным, сквозное шифрование характеризуется более сложной работой с ключами, поскольку каждая пара пользователей компьютерной сети должна быть снабжена одинаковыми ключами, прежде чем они смогут связаться друг с другом. А поскольку криптографический алгоритм реализуется на верхних уровнях модели OSI, приходится также сталкиваться со многими существенными различиями в коммуникационных протоколах и интерфейсах в зависимости от типов сетей и объединяемых в сеть компьютеров. Все это затрудняет практическое применение сквозного шифрования.

Комбинированное шифрование

Комбинация канального и сквозного шифрования данных в компьютерной сети обходится значительно дороже, чем каждое из них по отдельности. Однако именно такой подход позволяет наилучшим образом защитить данные, передаваемые по сети. Шифрование в каждом канале связи не позволяет противнику анализировать служебную информацию, используемую для маршрутизации. А сквозное шифрование уменьшает вероятность доступа к незашифрованным данным в узлах сети.

При комбинированном шифровании работа с ключами ведется так: сетевые администраторы отвечают за ключи, используемые при канальном шифровании, а о ключах, применяемых при сквозном шифровании, заботятся сами пользователи.

Шифрование файлов

На первый взгляд, шифрование файлов можно полностью уподобить шифрованию сообщений, отправителем и получателем которых является одно и то же лицо, а средой передачи служит одно из компьютерных устройств хранения данных (магнитный или оптический диск, магнитная лента, оперативная память). Однако все не так просто, как кажется на первый взгляд.

Если при передаче по коммуникационным каналам сообщение затеряется по пути от отправителя к получателю, его можно попытаться передать сно-

ва. При шифровании данных, предназначенных для хранения в виде компьютерных файлов, дела обстоят иначе. Если вы не в состоянии расшифровать свой файл, вам вряд ли удастся сделать это и со второй, и с третьей, и даже с сотой попытки. Ваши данные будут потеряны раз и навсегда. Это означает, что при шифровании файлов необходимо предусмотреть специальные механизмы предотвращения возникновения ошибок в шифртексте.

Криптография помогает превратить большие секреты в маленькие. Вместо того чтобы безуспешно пытаться запомнить содержимое огромного файла, человеку достаточно его зашифровать и сохранить в памяти использованный для этой цели ключ. Если ключ применяется для шифрования сообщения, то его требуется иметь под рукой лишь до тех пор, пока сообщение не дойдет до своего адресата и не будет им успешно расшифровано. В отличие от сообщений, шифрованные файлы могут храниться годами, и в течение всего этого времени необходимо помнить и держать в секрете соответствующий ключ.

Есть и другие особенности шифрования файлов, о которых необходимо помнить вне зависимости от применяемого криптографического алгоритма:

- нередко после шифрования файла его незашифрованная копия остается на другом магнитном диске, на другом компьютере или в виде распечатки, сделанной на принтере;
- размер блока в блочном алгоритме шифрования может значительно превышать размер отдельной порции данных в структурированном файле, в результате чего зашифрованный файл окажется намного длиннее исходного;
- скорость шифрования файлов при помощи выбранного для этой цели криптографического алгоритма должна соответствовать скоростям, на которых работают устройства ввода/вывода современных компьютеров;
- работа с ключами является довольно непростым делом, поскольку разные пользователи должны иметь доступ не только к различным файлам, но и к отдельным частям одного и того же файла.

Если файл представляет собой единое целое (например, содержит отрезок текста), восстановление этого файла в исходном виде не потребует больших усилий: перед использованием достаточно будет просто расшифровать весь файл. Однако если файл структурирован (например, разделен на записи и поля, как это делается в базах данных), то расшифровывание всего файла целиком каждый раз, когда необходимо осуществить доступ к отдельной порции данных, сделает работу с таким файлом чрезвычайно неэффективной. Шифрование порций данных в структурированном файле делает его уязвимым по отношению к атаке, при которой злоумышленник отыскивает в этом файле нужную порцию данных и заменяет ее на другую по своему усмотрению.

У пользователя, который хочет зашифровать каждый файл, размещенный на жестком диске компьютера, имеются две возможности. Если он использует

один и тот же ключ для шифрования всех файлов, то впоследствии окажется не в состоянии разграничить доступ к ним со стороны других пользователей. Кроме того, в результате у криптоаналитика будет много шифртекста, полученного на одном ключе, что существенно облегчит вскрытие этого ключа.

Лучше шифровать каждый файл на отдельном ключе, а затем зашифровать все ключи при помощи *мастер-ключа*. Тем самым пользователи будут избавлены от суеты, связанной с организацией надежного хранения множества ключей. Разграничение доступа групп пользователей к различным файлам будет осуществляться путем деления множества всех ключей на подмножества и шифрования этих подмножеств на различных мастер-ключах. Стойкость такой криптосистемы будет значительно выше, чем в случае использования единого ключа для шифрования всех файлов на жестком диске, поскольку ключи, применяемые для шифрования файлов, можно генерировать случайным образом и, следовательно более стойкими против словарной атаки.

Аппаратное и программное шифрование

Аппаратное шифрование

Большинство средств криптографической защиты данных реализовано в виде специализированных физических устройств. Эти устройства встраиваются в линию связи и осуществляют шифрование всей передаваемой по ней информации. Преобладание *аппаратного* шифрования над программным обусловлено несколькими причинами.

- *Более высокая скорость.* Криптографические алгоритмы состоят из огромного числа сложных операций, выполняемых над битами открытого текста. Современные универсальные компьютеры плохо приспособлены для эффективного выполнения этих операций. Специализированное оборудование умеет делать их гораздо быстрее.
- *Аппаратуру легче физически защитить от проникновения извне.* Программа, выполняемая на персональном компьютере, практически беззащитна. Вооружившись отладчиком, злоумышленник может внести в нее скрытые изменения, чтобы понизить стойкость используемого криптографического алгоритма, и никто ничего не заметит. Что же касается аппаратуры, то она обычно помещается в особые контейнеры, которые делают невозможным изменение схемы ее функционирования. Чип покрывается специальным химическим составом, и в результате любая попытка преодолеть защитный слой этого чипа приводит к самоуничтожению его внутренней логической структуры. И хотя иногда электромагнитное излучение может служить хорошим источником информации о том, что происходит внутри микросхемы, от этого излучения легко избавиться, заэкранировав микросхему. Аналогичным образом можно заэкранировать и компьютер, однако сделать это гораздо сложнее, чем миниатюрную микросхему.

- *Аппаратура шифрования более проста в установке.* Очень часто шифрование требуется там, где дополнительное компьютерное оборудование является совершенно излишним. Телефоны, факсимильные аппараты и модемы значительно дешевле оборудовать устройствами аппаратного шифрования, чем встраивать в них микрокомпьютеры с соответствующим программным обеспечением.

Даже в компьютерах установка специализированного шифровального оборудования создает меньше проблем, чем модернизация системного программного обеспечения с целью добавления в него функций шифрования данных. В идеале шифрование должно осуществляться незаметно для пользователя. Чтобы добиться этого при помощи программных средств, средства шифрования должны быть упрятаны глубоко в недра операционной системы. С готовой и отлаженной операционной системой проделать это безболезненно не так-то просто. Но даже любой непрофессионал сможет подсоединить шифровальный блок к персональному компьютеру, с одной стороны, и к внешнему модему, с другой.

Современный рынок аппаратных средств шифрования информации предлагает потенциальным покупателям 3 разновидности таких средств — самодостаточные шифровальные модули (они самостоятельно выполняют всю работу с ключами), блоки шифрования в каналах связи и шифровальные платы расширения для установки в персональные компьютеры. Большинство устройств первого и второго типов являются узко специализированными, и поэтому прежде, чем принимать окончательное решение об их приобретении, необходимо досконально изучить ограничения, которые при установке накладывают эти устройства на соответствующее "железо", операционные системы и прикладное программное обеспечение. А иначе можно выбросить деньги на ветер, ни на йоту не приблизившись к желанной цели. Правда, иногда выбор облегчается тем, что некоторые компании торгуют коммуникационным оборудованием, которое уже имеет предустановленную аппаратуру шифрования данных.

Платы расширения для персональных компьютеров являются более универсальным средством аппаратного шифрования и обычно могут быть легко сконфигурированы таким образом, чтобы шифровать всю информацию, которая записывается на жесткий диск компьютера, а также все данные, пересылаемые на дискеты и в последовательные порты. Как правило, защита от электромагнитного излучения в шифровальных платах расширения отсутствует, поскольку нет смысла защищать эти платы, если аналогичные меры не предпринимаются в отношении всего компьютера.

Программное шифрование

Любой криптографический алгоритм может быть реализован в виде соответствующей программы. Преимущества такой реализации очевидны: программные

средства шифрования легко копируются, они просты в использовании, их нетрудно модифицировать в соответствии с конкретными потребностями.

Во всех распространенных операционных системах имеются встроенные средства шифрования файлов. Обычно они предназначены для шифрования отдельных файлов, и работа с ключами целиком возлагается на пользователя. Поэтому применение этих средств требует особого внимания. Во-первых, ни в коем случае нельзя хранить ключи на диске вместе с зашифрованными с их помощью файлами, а во-вторых, незашифрованные копии файлов необходимо удалить сразу после шифрования.

Конечно, злоумышленник может добраться до компьютера и незаметно внести нежелательные изменения в программу шифрования. Однако основная проблема состоит отнюдь не в этом. Если злоумышленник в состоянии проникнуть в помещение, где установлен компьютер, он вряд ли будет возиться с программой, а просто установит скрытую камеру в стене, подслушивающее устройство — в телефон или датчик для ретрансляции электромагнитного излучения — в компьютер. В конце концов, если злоумышленник может беспрепятственно все это сделать, сражение с ним проиграно, даже еще не начавшись.

Сжатие и шифрование

Алгоритмы сжатия данных очень хорошо подходят для совместного использования с криптографическими алгоритмами. Тому есть две причины:

- При вскрытии шифра криптоаналитик, как правило, полагается на избыточность, свойственную любому открытому тексту. Сжатие помогает избавиться от этой избыточности.
- Шифрование данных является весьма трудоемкой операцией. При сжатии уменьшается длина открытого текста, за счет чего сокращается время, которое будет потрачено на его шифрование.

Надо только не забыть сжать файл до того, как он будет зашифрован. После шифрования файла при помощи качественного криптографического алгоритма полученный шифртекст сжать не удастся, поскольку его характеристики будут близки к характеристикам совершенно случайного набора букв. Кстати, сжатие может служить своеобразным тестом для проверки качества криптографического алгоритма: если шифртекст поддается сжатию, значит этот алгоритм лучше заменить на более совершенный.

Как спрятать один шифртекст в другом

Антон и Борис несколько месяцев обменивались шифрованными сообщениями. Контрразведка перехватила все эти сообщения, но так и не смогла про-

честь ни единого слова. Контрразведчикам надоело коллекционировать переписку Антона и Бориса, не зная ее содержания, и они решили арестовать подозрительную парочку. Первый же допрос начался словами: "Где ключи к шифру?" "К какому такому шифру?!" — в один голос воскликнули Антон и Борис, но тут же осеклись и побледнели, заметив на столе у следователя зловещего вида клещи, покрытые пятнами то ли ржавчины, то ли крови.

Антон и Борис смогли бы выкрутиться из создавшегося положения, если бы шифровали каждое свое сообщение так, чтобы оно допускало два различных расшифрования в зависимости от используемого ключа. Свое настоящее секретное сообщение Борису Антон мог бы зашифровать на одном ключе, а вполне невинный открытый текст — на другом. Теперь, если от Антона потребуют ключ к шифру, он отдаст подставной ключ, который позволит прочесть невинное сообщение, а другой ключ сохранит в тайне.

Самый простой способ сделать это потребует использования одноразового блокнота. Пусть P — секретный открытый текст, D — невинный открытый текст, C — шифрованный текст, K — настоящий ключ, а K' — подставной ключ. Антон шифрует P:

$$P \oplus K = C$$

Поскольку у Бориса имеется копия ключа K, он может без проблем расшифровать сообщение Антона:

$$C \oplus K = P$$

Если контрразведчики попытаются заставить Антона и Бориса выдать используемый ими ключ, то вместо K они могут сообщить в контрразведку:

$$K' = C \oplus D$$

В результате контрразведчики смогут прочитать невинный открытый текст:

$$C \oplus K' = D$$

Так как Антон и Борис пользуются одноразовым блокнотом, то K является полностью случайным и доказать, что K' является подставным ключом, практически невозможно (не прибегая к пыткам).

Антон мог бы зашифровать P не с помощью одноразового блокнота, а пользуясь любым из своих самых любимых криптографических алгоритмов и ключом K. Сложив C с фрагментом какого-либо общеизвестного произведения (например, с отрывком из второй главы "Идиота") по модулю 2, Антон получит K'. Теперь если к Антону пристанут злые "дяденьки" из контрразведки, он предъявит им C вместе с K' и скажет, что K' — это одноразовый блокнот для C и что он просто захотел попрактиковаться в криптографии, зашифровав для этой цели отрывок из первой попавшейся книги. И пока контрразведчики не получат в свое распоряжение ключ K, доказать, что Антон занимался чем-то противозаконным, они не смогут.

Почему криптосистемы ненадежны

В настоящее время криптография успешно используется почти во всех информационных системах — от Internet до баз данных. Без нее обеспечить требуемую степень конфиденциальности в современном, до предела компьютеризированном мире уже не представляется возможным. Кроме того, с помощью криптографии предотвращаются попытки мошенничества в системах электронной коммерции и обеспечивается законность финансовых сделок. Со временем значение криптографии, по всей вероятности, возрастет. Для этого предположения имеются веские основания.

Однако с огорчением приходится признать, что подавляющее большинство криптографических систем не обеспечивает того высокого уровня защиты, о котором с восторгом обычно говорится в их рекламе. Многие из них до сих пор не были взломаны по той простой причине, что пока не нашли широкого распространения. Как только эти системы начнут повсеместно применяться на практике, они, словно магнит, станут привлекать пристальное внимание злоумышленников, которых сегодня развелось великое множество. При этом удача и везение будут явно на стороне последних. Ведь для достижения своих целей им достаточно найти в защитных механизмах всего лишь одну брешь, а обороняющимся придется укреплять все без исключения уязвимые места.

Реализация

Понятно, что никто не в состоянии предоставить стопроцентную гарантию безопасности. Тем не менее, криптографическую защиту без особых усилий можно спроектировать так, чтобы она противостояла атакам злоумышленников вплоть до того момента, когда им станет проще добыть желаемую информацию другим путем (например, с помощью подкупа персонала или внедрения программ-шпионов). Ведь криптография действительно хороша именно тем, что для нее уже давно придуманы эффективные алгоритмы и протоколы, которые необходимы, чтобы надежно защитить компьютеры и компьютерные сети от электронного взлома и проявлений вандализма.

Вот почему в реальной жизни криптографические системы редко взламываются чисто математическими методами. Ведь криптографический алгоритм или протокол от его практической реализации в виде работающей программы, как правило, отделяет зияющая пропасть. Даже доказанный по всем правилам формальной логики факт, что криптографическая защита совершенна с математической точки зрения, совсем не означает, что она останется таковой после того, как над ее внедрением поработают программисты.

Известно, что под давлением бюджетных ограничений, дефицита времени и личных неурядиц программисты неизбежно допускают весьма серьезные ошибки при реализации алгоритмов — используют плохие датчики случайных

чисел для генерации криптографических ключей, не учитывают специфику аппаратной среды, в которой предстоит эксплуатировать созданные ими программные средства, а также регулярно забывают удалять ключевую и другую секретную информацию из оперативной памяти компьютера или с магнитного носителя после того, как надобность в ее хранении там отпала. Единственный способ научиться избегать этих и им подобных ошибок состоит в том, чтобы вновь и вновь стараться создать совершенные системы криптографической защиты данных, а потом не менее упорно пытаться их взломать.

Конечно, после того как брешь в системе криптографической защиты найдена, ее довольно легко можно залатать. Но сам поиск подобного рода дефектов является невероятно сложной задачей. Никакое предварительное тестирование не поможет обнаружить в криптографической системе все дефекты, поскольку ни один тест в отдельности не может дать полной гарантии их отсутствия. Ведь если программа шифрования правильно зашифровывает и расшифровывает файлы, это еще совсем не значит, что она надежно защищает их содержимое.

Учет реальных потребностей пользователей

Немало проблем, связанных с использованием криптографических средств, создают сами пользователи. Безопасность заботит их меньше всего. В первую очередь им требуются простота, удобство и совместимость с уже существующими (как правило, недостаточно защищенными) программными продуктами. Они выбирают легко запоминающиеся криптографические ключи, записывают их где попало, запросто делятся ими с друзьями и знакомыми. Поэтому грамотно спроектированная криптографическая система обязательно должна принимать во внимание специфические особенности поведения людей.

Еще труднее оказывается убедить людей в необходимости строго и неукоснительно применять криптографическую защиту данных. Пользователи с готовностью приносят в жертву собственную безопасность, если средства ее обеспечения мешают им поскорее сделать свою работу. Поэтому только в случае, если при проектировании криптографической системы были учтены реальные потребности пользователей, она действительно в состоянии защитить их компьютеры и компьютерные сети.

Законодательные ограничения

В Своде законов США имеется пункт 2778, который называется "Контроль за экспортом и импортом вооружений". Именно этот пункт является юридической основой для ряда инструкций, именуемых "Правилами контроля за перемещением оружия в мире" (International Traffic in Arms Regulations, сокращенно — ITAR). Раздел 120.1 ITAR впрямую причисляет к военному

снаряжению, за перемещением которого Соединенные Штаты осуществляют самый строгий контроль, программное обеспечение, предназначенное для целей эффективного шифрования данных[1]. А это означает, что американским компаниям, желающим экспортировать программы эффективного шифрования, необходимо зарегистрироваться в Государственном департаменте США в качестве торговца военным имуществом и получить там лицензию на экспорт.

Известно, что при выдаче таких лицензий Госдепартамент целиком и полностью полагается на мнение АНБ. В результате лицензия на экспорт криптографических средств никому не выдается до тех пор, пока АНБ не одобрит такое решение. В свою очередь, АНБ отнюдь не заинтересовано в свободном распространении надежных программ шифрования за пределами страны. Поэтому все программные средства, произведенные в США и легально экспортируемые за рубеж, обеспечивают ослабленную криптографическую защиту.

Чтобы повысить свою конкурентоспособность на мировом рынке, производители средств криптографической защиты в США вынуждены искать лазейки в законодательстве. Например, известная американская фирма RSA Data Security попыталась обойти закон путем финансирования усилий китайских ученых, которых правительство Китая официально уполномочило разработать новые программные средства шифрования данных. Предполагалось, что эти средства, созданные на основе алгоритмов, переданных американской фирмой китайцам, смогут обеспечить более надежную криптографическую защиту информации, чем те, которые Китай в состоянии импортировать из США в соответствии с действующим американским законодательством. Это, несомненно, радостное событие для Китая, однако следует отметить, что ради удовлетворения потребностей рядового пользователя за рубежом, не обладающего возможностями и ресурсами, сравнимыми с теми, которые имеются в распоряжении китайского правительства, американские производители программ эффективного шифрования вряд ли будут искать какие-либо пути, ведущие в обход американского законодательства.

Следуя примеру США, ряд государств, в том числе и Россия, ввели ограничения на экспорт, импорт и использование шифровальных средств. Тем не менее, многих российских граждан ничуть не пугают законодательные ограничения на эксплуатацию шифровальных средств. Они твердо придерживаются мнения о том, что принадлежащая им информация безусловно является объектом их собственности, и что они, как собственники своей информации, имеют право самостоятельно определять правила ее хранения

[1] Эффективное шифрование данных определяется как шифр, взлом которого требует такого объема вычислений, что при современном уровне развития вычислительной техники вскрыть его практически невозможно.

и защиты. Остается только со знанием дела решить, какие именно шифровальные средства применять для адекватной защиты этой информации, а какие не использовать ни в коем случае, ввиду их слабой надежности.

Слишком малая длина ключа

Слишком малая длина ключа — одна из самых очевидных причин ненадежности криптографических систем. Причем недостаточную длину ключа могут иметь даже те криптосистемы, в которых применяются самые надежные алгоритмы шифрования, поскольку:

- в них изначально может присутствовать возможность работы с ключом переменной длины для того, чтобы при использовании этих систем на практике можно было выбрать нужную длину ключа, исходя из желаемой надежности и эффективности;
- они разрабатывались тогда, когда данная длина используемого ключа считалась более чем достаточной для обеспечения необходимого уровня криптографической защиты;
- на них распространяются экспортные ограничения, которые устанавливают допустимую длину ключа на уровне, не отвечающем современным требованиям.

Первым надежным криптографическим алгоритмом, который вплотную столкнулся с проблемой выбора адекватной длины ключа, стал RSA. Дело в том, что его вскрытие требует разложения на множители (*факторизации*) очень больших чисел. В марте 1994 г. за вполне приемлемое время было факторизовано 428-битовое число, а на сегодняшний день достаточно реальным представляется факторизация 512-битовых чисел. Достигнутый прогресс в решении задачи факторизации очень больших чисел связан не только с ростом вычислительных мощностей современного компьютерного парка, но и с разработкой новых эффективных алгоритмов. На том, что эта задача является очень трудоемкой, еще совсем недавно была основана надежность криптографического алгоритма, используемого в распространенной программе PGP. Поэтому можно утверждать, что сегодня разложение на множители является одной из самых динамично развивающихся областей криптографии.

В начале 1998 г. из-за слишком малой длины ключа (56 бит) фактически "приказал долго жить" DES-алгоритм, долгое время являвшийся официальным стандартом шифрования данных в США. Сейчас американским Национальным институтом стандартов объявлен конкурс на новый стандарт шифрования данных Advanced Encryption Standard (AES). Согласно условиям этого конкурса, кандидаты на роль AES должны представлять собой симметричные алгоритмы шифрования с ключом длиной более 128 бит.

Потайные ходы

Причины появления *потайных ходов* в криптографических системах довольно очевидны: их разработчики хотят иметь контроль над шифруемой в этих системах информацией и оставляют для себя возможность расшифровывать ее, не зная ключа пользователя. Средство, с помощью которых данная возможность реализуется на практике, и принято именовать потайным ходом. Иногда потайные ходы применяются для целей отладки, а после ее завершения разработчики в спешке просто забывают убрать их из конечного продукта.

Классический пример потайного хода, который хакерами единодушно признается самым талантливым "хаком" по взлому системы парольной защиты всех времен и народов, привел Кен Томпсон (один из авторов компилятора для языка программирования C) в своей лекции по случаю вручения ему престижной премии Тьюринга. Дело в том, что в операционной системе UNIX пользовательские пароли хранятся в зашифрованном виде в специальной базе данных. В компилятор языка C Томпсоном был предусмотрительно вставлен код, распознававший, когда на вход компилятора поступала программа, содержавшая приглашение пользователю зарегистрироваться (login). Тогда компилятор добавлял в эту программу код, который распознавал пароль, выбранный самим Томпсоном. Таким образом, Томпсон получал возможность успешно проходить процедуру регистрации и идентификации, не зная легальных паролей, хранимых в зашифрованной базе данных.

Стандартный способ закрыть такой потайной ход состоит в том, чтобы удалить из исходного текста компилятора "вредный" код, а затем его перекомпилировать. Но при перекомпиляции опять не обойтись без компилятора. И Томпсон дописал свой компилятор так, чтобы тот распознавал, когда на его вход поступала исправленная версия его самого. В этом случае компилятор добавлял в нее код, который, в свою очередь, при компиляции программ с приглашением login дописывал в них код, дающий Томпсону привилегированный доступ, а также код, который позволял компилятору распознавать свою обновленную версию при перекомпиляции. Таким образом, не имеет значения, насколько надежным был криптографический алгоритм, который использовался для шифрования паролей пользователей операционной системы UNIX. Потайной ход, придуманный Томпсоном, оставался открыт для него при любых условиях.

Шифрование вокруг нас

Итак, для того чтобы создать надежную криптографическую систему, необходимо обладать достаточными познаниями в области современной криптографии, аккуратно и безошибочно воплотить эти познания в виде работающей программы с дружественным интерфейсом, убрав из нее все потайные ходы по окончании отладки. Далее требуется передать эту систему

в ФАПСИ, чтобы получить там лицензию, дающую право на ее легальное распространение и использование на территории России. Однако несмотря на богатый научный потенциал российских криптографов и высокую квалификацию отечественных программистов, единственным лицензированным ФАПСИ шифром в настоящее время является ГОСТ 28147-89, разработанный еще в недрах КГБ. Все остальные криптосистемы, предлагаемые зарубежными и российскими фирмами в виде законченных продуктов или библиотек, включая как устоявшие зарубежные стандарты, так и самостоятельные оригинальные разработки, являются незаконными.

Поскольку наше государство оказывается не в состоянии обеспечить всех своих граждан, остро нуждающихся в надежной информационной защите, сертифицированными криптографическими средствами, виртуальная среда обитания российского компьютерного пользователя буквально нашпигована шифровальными программами (сотни таких программ можно найти в Internet, например, по адресу **ftp.elf.stuba.sk/pub/security**). Их распространению способствует нечеткость президентского Указа № 334, в котором не оговорено, что же конкретно понимается под термином "шифрование" данных. Если предположить, что шифрование — это такая нестандартная кодировка данных, которая серьезно затрудняет возможность их перекодировки в стандартное представление без соответствующего аппаратного или программного обеспечения, то в категорию шифрсистем тут же попадут архиваторы (pkzip, arj и rar), известные текстовые редакторы (Word и Lexicon), а также средства редактирования графических изображений (Paint и CorelDraw), поскольку все они используют свою собственную нестандартную кодировку, не позволяющую без соответствующих программ просматривать закодированные с их помощью данные.

Попытка придумать универсальный критерий подразделения кодировок на стандартные и нестандартные заранее обречена на провал, т. к. разработчиков программного обеспечения нельзя заставить пользоваться только кодировкой, одобренной указом президента в качестве стандартной. Поэтому лучше к системам шифрования относить, например, программные средства, к которым прилагается документация с явным указанием того факта, что они предназначены именно для шифрования данных.

Учитывая неразбериху, царящую в российском законодательстве, неудивительно, что российские пользователи для защиты своей конфиденциальной информации активно применяют архиваторы с парольной защитой, Norton Diskreet, Word, Excel, многочисленные условно-бесплатные программы (PGP, CodeDrag, SecurPC, Secur-all 32, BestCrypt NP, Kremlin и др.), криптографические системы отечественных фирм ("Лан Крипто", "Анкорт" и др.), собственные кустарные разработки, а также программы неизвестного происхождения. Большинство из них крайне слабы, и программы их взлома за вполне умеренную плату можно получить, например, в Internet по адресу **www.accessdata.com**. Исключение в списке заведомо ненадежных криптогра-

фических систем, потенциально доступных пользователю в России, составляют лишь несколько оригинальных разработок российских фирм. Однако ввиду воздвигнутой нашим государством информационной блокады вокруг криптографии и всего, что с ней связано, можно только строить предположения, какие именно.

Подводя итог сказанному, можно сделать вывод о том, что ситуация на рынке криптографических систем не внушает оптимизма. Законодательные ограничения, ошибки в реализации, недружественный интерфейс, недостаточная длина ключа и наличие потайных ходов приводят к тому, что отыскать надежную криптосистему практически невозможно.

Поскольку криптография призвана обслуживать потребности человечества в довольно деликатной сфере (с помощью криптографических методов сохраняется в тайне конфиденциальная информация, не подлежащая, по мнению ее владельцев, бесконтрольному распространению), некоторые исследователи усматривают в сложившейся ситуации действие определенных тайных сил, которые пытаются направлять и контролировать прогресс человечества в области криптографии. Одна из главных забот этих тайных сил — взять каждого "под колпак", т. е. иметь наиболее полное досье на любого человека. Поэтому тайные общества так заинтересованы в единоличном владении элитарными криптографическими знаниями и созданными на основе этих знаний надежными средствами криптографической защиты данных, бесконтрольное распространение которых может поставить под угрозу их способность ведения тотальной слежки. Безрезультатно заканчиваются многообещающие криптографические исследования, при загадочных обстоятельствах обрываются жизни талантливых криптографов, возникают всевозможные препоны на пути свободного обмена информацией о последних криптографических изысканиях.

Другие исследователи закулисных пружин истории идут еще дальше и утверждают, что именно наиболее полные и достоверные знания из области криптологии (науки, объединяющей криптографию и криптоанализ), позволили нынешним тайным властителям, распоряжающимся судьбой человечества, достичь вершин своего могущества. Гипотеза этих исследователей состоит в том, что криптология является одним из эффективных инструментов познания окружающего мира: информация о главных направлениях его развития в зашифрованном виде доступна каждому и ее можно извлечь путем дешифрования. Кто знает, как это делается, обладает почти неограниченной властью над миром, поскольку может с большой достоверностью предсказывать будущее.

Объединены ли тайные верховные правители в единую организацию? Вряд ли. Скорее всего, между их различными сообществами существует серьезная конкуренция. Да и могущество их простирается до определенных пределов.

Поэтому время от времени вполне вероятно появление надежных криптосистем, хотя бы на ограниченный период времени.

Проверить эти гипотезы на практике представляется невозможным, и некоторым они могут показаться слишком смелыми, но иметь о них представление совершенно необходимо. Хотя бы для того, чтобы в случае приобретения вами заведомо ненадежной криптосистемы для нужд вашей фирмы или организации оправдать свою оплошность перед руководством вмешательством неких тайных всемогущих сил.

Приложение

Англо-русский криптологический словарь с толкованиями

Вряд ли нужно кому-то доказывать, что криптология традиционно является одной из важнейших областей науки. И сегодня информацию о ее достижениях можно почерпнуть не только из публикаций в солидных научных журналах, но даже из статей в бульварной прессе. Перед читателями вырисовывается волнующая панорама событий: они узнают об ожесточенных столкновениях на почве криптологии между правительственными спецслужбами и независимыми учеными, читают о "потайных ходах" в распространенных шифрсистемах, слышат сенсационные заявления об изобретении новых, абсолютно стойких шифров, вскрыть которые не под силу ни одному, даже самому одаренному криптоаналитику, и уверения в том, что невскрываемых средств криптографической защиты в природе не существует. Чтобы разглядеть истину в этих спорах, надо в первую очередь иметь очень четкое представление об используемой терминологии.

С огорчением приходится признать, что серьезных работ по криптологии в нашей печати выходит пока еще довольно мало. Сказывается наследие недавнего прошлого, когда криптологические методы держались в строгом секрете из опасения раскрыть состояние дел в советской криптологии перед западным противником в холодной войне.

В то же время имеется богатая научная, учебная и популярная литература по криптологии на английском языке. Регулярно проходят конференции, симпозиумы и семинары, труды которых публикуются в виде отдельных сборников. Массовым тиражом издаются специализированные криптологические журналы.

В этих условиях и профессиональным российским исследователям-криптологам, и простым любителям криптографических головоломок будет весьма полезен данный словарь, призванный помочь им познакомиться с достойными внимания образцами многообразной печатной продукции, относящейся к области криптологии, на английском языке. Его основу составляют англоязычные термины, появлению которых мы обязаны исключительно этой науке (шифр, маркировка и др.). Однако, как и любая наука, криптология не может существовать изолированно от других областей знаний. Поэтому в словаре можно найти множество понятий, позаимствован-

ных, например, из теории связи и из математики, а также из общеупотребимой научной лексики. Обойти вниманием эти понятия было бы, по меньшей мере, неразумно.

Не требуется каких-либо специальных навыков для того, чтобы пользоваться данным словарем, поскольку он построен традиционным образом. Все термины, будь то слова, словосочетания или сокращения, вынесенные в заголовки словарных статей, выделены полужирным шрифтом и упорядочены по алфавиту. При определении порядка следования заголовков вспомогательные небуквенные символы (амперсанды, апострофы, дефисы, кавычки, пробелы, цифры) и артикли во внимание не принимались. Если требуется отыскать в словаре какое-либо словосочетание, не попавшее в заголовок, следует обратиться к словарной статье для первого слова в этом словосочетании.

Известно, что один и тот же термин можно переводить на русский язык поразному в зависимости от контекста. В связи с этим близкие по значению русскоязычные варианты английских терминов разделены в словаре запятой, родственные — точкой с запятой, а имеющие разный смысл снабжаются арабскими цифрами в круглых скобках. В квадратные скобки заключены равноправные варианты значений словарных единиц, а дополнительный пояснительный текст к терминам, во всем словаре выделенный курсивом, — в круглые (но не везде, а в зависимости от его расположения в словарной статье). Знак равенства используется при отсылках к другим словарным статьям, косая черта применяется для разделения вариантов значений словарных единиц, а тильда заменяет заголовок внутри словарной статьи.

Лексикографические источники

Англо-русский словарь персоналий. – М., 1993.

Англо-русский словарь по вычислительной технике. – М., 1989.

Анин Б. Радиошпионаж. – М., 1996.

Безопасность компьютерных систем и сетей: Англо-русский словарь. – М., 1995.

Большой англо-русский словарь. – М., 1987.

Великобритания: Лингвострановедческий словарь. – М., 1980.

Гарднер М. От мозаик Пенроуза к надежным шифрам. – М., 1993.

Горохов П. Информационная безопасность: Англо-русский словарь. – М., 1995.

Дьюдни А. К. О шифровальных и дешифровальных машинах // В мире науки. – 1988. – №№ 11–12.

Жельников В. Криптография от папируса до компьютера. – М., 1996.

Защита информации (малый тематический выпуск) // ТИИЭР. – 1988. – № 5.

Исаев К. Немного о киберпанке // Защита информации. "Конфидент". – 1996. – № 5.

Кан Д. Взломщики кодов (главы из книги) // Защита информации. "Конфидент". – 1997. – №№ 1–6.

Колесников О. Использование запретов двоичных функций при решении систем уравнений // Обозрение прикладной и промышленной математики. – 1995. – том 2. – вып. 3.

Математический энциклопедический словарь. – М., 1988.

Никитин Ф. Глобальная сеть нуждается в глобальной защите // Мир связи. – 1996. – октябрь-ноябрь.

Новый большой англо-русский словарь. – М., 1993.

Пробелков П. Сколько стоит "сломать" Netscape? // Защита информации. "Конфидент". – 1996. – № 5.

Реймонд Э. Новый словарь хакера. – М., 1996.

Словарь терминов // Защита информации. "Конфидент". – 1995. – №№ 4–6.

Тайли Э. Безопасность компьютера. – Минск, 1998.

Шнейер Б. Слабые места криптографических систем // Открытые системы. – 1999. – № 1.

Энкни Р. Введение в криптографические стандарты // Защита информации. "Конфидент". – 1996. – № 4.

Advances in cryptology: Proceedings of Eurocrypt'94. – Berlin, 1995.

Andreassen K. Computer cryptology. – Englewood Cliffs, 1988.

Anonymous A. Maximum Security. – N.-Y., 1998.

Bamford J. Loud and clear // Washington Post. – 1999. – November 14.

Basic cryptanalysis. – US Army field manual № 34-40-2, 1990.

Beker H., Piper F. Cipher systems. – L., 1982.

Brazier J. Possible NSA decryption capabilities. – Internet, cryptome.org/nsa-study.doc.

Campbell D. Development of surveillance technology and risk of abuse of economic information. – Luxembourg, 1999.

Crypto-Gram. – 1998. – May-December.

Crypto-Gram. – 1999. – January-February .

Cryptologia. – 1993–1998. – №№ 1-4.

Deavours C.A., Kruth L. Machine cryptography and modern cryptanalysis. – Dedham, 1985.

Fast software encryption. – N.-Y., 1994.

Feistel H. Cryptography and computer privacy // Scientific American. – 1973. – Vol. 228. – № 5.

Gardner M. Penrose tiles to trapdoor ciphers. – N.-Y., 1989.

Harris R. Enigma. – L., 1997.

Kahn D. The codebreakers. – N.-Y., 1967.

Kahn D. Soviet COMINT in the cold war // Cryptologia. – 1998. – № 1.

Kelsey J., Scheier B., Hall C. Side-channel cryptanalysis of product ciphers. – Internet, www.counterpane.com.

Koblitz N. A course in number theory and cryptography. – N.-Y., 1994.

Kocher P. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems // Lecture Notes in Computer Science. – 1996. – Vol. 1109.

Loeb V. Weaving a web of secrets // Washington Post. — 1999. — December 1.

Madsen W. Crypto AG: NSA's Trojan whore? // Covert Action Quarterly. — № 63 (Winter 1998).

National information systems security (INFOSEC) glossary. – NSTISSI No. 4009, 1997.

Pope T. Password files // Dr. Dobb's journal. – 1996. – № 1.

Proceedings of the IEEE. – 1988. – № 5.

Rhee M. Cryptography and secure communications. – Singapore, 1994.

Robshaw M. Security estimates for 512-bit RSA. – Internet, www.rsa.com / pubs / techreports / security_estimates.pdf.

Robshaw M. Stream ciphers. – Internet, www.rsa.com / pubs / techreports / tr-701.pdf.

Schneier B. Applied cryptography. – N.-Y., 1996.

Schneier B. Cryptographic design vulnerabilities // Computer. – 1998. – №9.

Schneier B. Differential and linear cryptanalysis // Dr. Dobb's journal. – 1996. – № 1.

Schneier B. A self-study course in block-cipher cryptanalysis. – Internet, www.counterpane.com.

Sussman V. Policing cyberspace // U.S.News & World Report. – 1995. – January 23.

Thompson N. No more top-secret pizza boxes // Baltimore Sun. — 1999. — April 6.

Uncle Sam and his 40000 snoopers // Nation Review (AU), 1973.

U.S. Electronic Espionage: A Memoir // Ramparts. – 1996. – № 2.

Wayner P. Don't lose your cryptokey // Byte. – 1996. – № 5.

Webster's third new international dictionary. – Springfield, 1981.

Сокращения

Английские

abbr — abbreviation — сокращение

adj — adjective — имя прилагательное

adv — adverb — наречие

n — noun — имя существительное

num — numeral — числительное

pl — plural — множественное число

prep — preposition — предлог

prop — proper name — имя собственное

suff — suffix — суффикс

superl — superlative degree — превосходная степень

v — verb — глагол

Русские

амер. — американский — American

букв. — буквально — literally

воен. — военное дело — military

в т.ч. — в том числе — including

г. — год — year

грам. — грамматический — grammatical

жарг. — жаргон — jargon

знач. — значение — meaning

и т. д. — и так далее — et cetera, and so on

лат. — латинский язык — Latin

напр. — например — for example

нем. — немецкий — German

прил. — имя прилагательное — adjective

проф. — профессиональное слово или выражение — professional word or phrase

разг. — разговорное слово или выражение — colloquial word or phrase

см. — смотри — see

сущ. — имя существительное — noun

тж. — также — also

Условные обозначения

~ (*тильда*) — заменяет заглавное слово или заглавное словосочетание внутри словарной статьи

[ ] (*квадратные скобки*) — в них помещаются равноправные варианты значений словарных единиц

( ) (*круглые скобки*) — содержат дополнительный пояснительный текст к терминам

= (*знак равенства*) — используется при отсылках к другим словарным статьям

/ (*косая черта*) — применяется для разделения вариантов значений словарных единиц

Криптологический словарь

A

ABC *abbr* АЧК (*см. American Black Chamber*).

abuse-free *adj* не допускающий злоупотреблений [злонамеренного использования]; ~ **cryptosystem** криптосистема, не допускающая злонамеренного использования.

Abwehr *prop нем.* "абвер" (*немецкая военная разведка во время второй мировой войны*); ~ **Enigma** "Энигма" "абвера".

ACA *abbr* АКА (*см. American Cryptogram Association*).

accept *v* принимать; ~ **as an authenticator** принимать в качестве аутентификатора.

access *n* доступ; ~ **control cryptosystem** криптосистема управления доступом; ~ **to keys** доступ к ключам; ~ **to two cryptograms in different keys for the same plaintext is very helpful to a cryptanalyst** доступ к двум криптограммам, полученным из одного и того же открытого текста с использованием разных ключей, очень помогает криптоаналитику.

access *v* осуществлять доступ; ~ **Internet traffic** осуществлять доступ к трафику "Интернет"; ~ **private and commercial communications** осуществлять доступ к частным и коммерческим линиям связи.

ACCSA *abbr* АККБОВС (*см. Allied Communications and Computer Security Agency*).

A-code *abbr* А-код (*см. authentication code*).

A^2-code *abbr* А^2-код (*см. authentication code with arbitration*).

activation *n* активация (*приведение в действие*); ~ **key** ключ активации (*криптографического устройства*).

active *adj* (1) активный; ~ **eavesdropping** активный перехват (*у противника имеется возможность не только наблюдать за обменом сообщениями, но и влиять на него, напр., задерживать или удалять сигналы, передаваемые по каналам связи*); ~ **wiretap** = ~ eavesdropping; (2) действующий; ~ **key** действующий ключ.

actual *adj* (1) фактический; ~ **key** фактический ключ; (2) существующий, действующий; ~ **cryptosystem** существующая криптосистема.

actually *adv* фактически; ~**-used key** фактически используемый ключ.

Adams *prop* Адамс К. (*один из разработчиков блочного алгоритма шифрования KACT*).

adaptive *adj* адаптивный; ~**-chosen-plaintext attack** адаптивная атака на основе выбранного открытого текста (*противник может по своему усмотрению выбирать открытый текст, подлежащий зашифрованию, и в дальнейшем модифицировать его, зная результат предыдущего сеанса шифрования*); ~**-chosen-plaintext attack secure** защищенный от адаптивной атаки на основе выбранного открытого текста.

add *v* складывать; ~ **bit by bit** складывать бит за битом; ~ **bitwise** складывать побитово.

additional *adj* дополнительный; ~ **decryption key** дополнительный ключ расшифрования (*предназначен для расшифрования криптограммы в случае, если ключ, при помощи которого производилось шифрование, по какой-то причине недоступен — утерян, умышленно уничтожен и т. д.*); ~ **key** дополнительный ключ.

additive *adj* аддитивный; ~ **combiner** аддитивный смеситель (*функция или устройство для объединения нескольких числовых последовательностей в одну путем покомпонентного сложения соответствующих элементов этих последовательностей*); ~ **stream cipher** аддитивный шифр [шифр гаммирования].

add-on *adj* дополнительный; ~ **cryptographic software package** дополнительный пакет программ для выполнения криптографических функций.

addition *n* сложение; ~ **modulo 2** сложение по модулю 2.

adjacent *adj* соседний; ~ **network nodes** соседние узлы сети (*связи*).

ADK *abbr* ДКР (*см. additional decryption key*).

Adelman *prop* Адельман Л. (*американский криптолог, один из авторов алгоритма шифрования РША*).

advance *adj* предварительный, заблаговременный; ~ **distribution of keys** предварительное распределение ключей.

advanced *adj* (1) новейший; ~ **Encryption Standard** новейший стандарт шифрования (*США*); (2) основанный на последних достижениях (*науки, техники и т. д.*); ~ **cryptanalysis** криптоанализ на основе последних достижений в этой области.

advances *n pl* успехи, прогресс, достижения; ~ **in the cryptanalysis of a cipher** успехи в криптоанализе шифра; ~ **in cryptology** достижения криптологии.

adversary *n* нарушитель; противник.

AES *abbr* НСШ (*см. Advanced Encryption Standard*).

affine *adj* аффинный; ~ **cipher** аффинный шифр; ~ **mapping** аффинное отображение.

agency *n* агентство; управление.

Agnes *prop* Агнесса (*название электромеханического устройства, которое во время второй мировой войны использовалось сотрудниками английской дешифровальной службы для автоматизации процесса вскрытия немецкой шифровальной машины "Энигма"*).

agree *v* договориться; ~ **on a key in advance** заранее договориться о ключе.

agreed-upon *adj* согласованный; ~ **key** согласованный ключ (*в одноключевой криптосистеме*).

aids *n pl* средства.

AK *abbr* АК (*см. automatic remote rekeying*).

AKDC *abbr* ЦАРК (*см. automatic key distribution center*).

AKDS *abbr* САРК (*см. automated key distribution system*).

A-key *abbr* В-ключ (*см. auxiliary key*).

Alberti *prop* Альберти Л. (*итальянский криптолог, автор самой старинной из сохранившихся на Западе рукописей, посвященных криптоанализу*).

ALG *abbr* АЛГ (*см. algorithm*).

algebraic *adj* алгебраический; ~ **attack** алгебраическая атака (*ведется с привлечением алгебраической теории*); ~ **cipher** алгебраический шифр; ~ **normal form** алгебраическая нормальная форма.

algorithm *n* алгоритм; ~**-supported encryption** = algorithmic encryption.

algorithmic *adj* алгоритмический; ~ **encryption** шифрование по алгоритму.

Alice *prop* Алиса (*гипотетическая личность, выполняющая необходимые действия при реализации криптографических протоколов*).

Allied Communications and Computer Security Agency *n* Агентство коммуникационной и компьютерной безопасности объединенных вооруженных сил (*НАТО*).

all-key *adj* характеризующийся перебором всех возможных вариантов ключа; ~ **attack** атака с перебором всех возможных вариантов ключа.

allophone *n* аллофон (*набор звуков, имеющих одинаковые свойства или одинаковую информативность*).

almost *adv* почти; ~ **perfect non-linear function** почти совершенная нелинейная функция.

alphabet *n* алфавит; ~ **wheel** = cipher clock.

alphabetic *adj* алфавитный; ~ **character** буква; ~ **progression** алфавитная прогрессия.

alphanumeric *adj* = alphanumerical.

alphanumerical *adj* буквенно-цифровой; ~ **character** буква или цифра; ~ **key** буквенно-цифровой ключ.

alterable *adj* изменяемый; ~ **key** изменяемый ключ.

alternative *adj* альтернативный; ~ **hypothesis** альтернативная гипотеза.

amateur *adj* любительский; ~ **cryptanalyst** криптоаналитик-любитель; ~ **cryptographer** криптограф-любитель.

ambiguous *adj* неоднозначный; ~ **cryptogram** неоднозначная криптограмма.

American *adj* Американский; ~ **Black Chamber** (1) "Американский черный кабинет" (*дешифровальное бюро, существовавшее с 1917 по 1929 г. при государственном департаменте США*); (2) "Американский черный кабинет" (*название книги мемуаров бессменного руководителя "Американского черного кабинета" г. Ярдли*); ~ **Cryptogram Association** Американская криптологическая ассоциация; ~ **National Standards Institute** Американский национальный институт стандартов (*занимается стандартизацией во многих технических областях, включая выработку криптографических стандартов*); ~ **Standard Code for Information Interchange** Американский стандартный код для обмена информацией.

A-mode *abbr* А-режим (*см. asynchronous mode*).

amount *n* количество; ~ **of information in a message** количество информации в сообщении.

anagram *n* анаграмма.

anagramming *n* анаграммирование.

analog *adj* аналоговый; ~ **communication** аналоговая связь.

analysis *n* анализ; криптоанализ; ~ **of pseudorandom number generators** анализ генераторов псевдослучайных чисел.

analyst *n* аналитик; криптоаналитик.

analyzability *n* подверженность криптоанализу.

ancestor *n* предшественник; ~ **key** предшествующий [исходный] ключ.

Andreyev *prop* Андреев Н. (*генерал КГБ, в 60-е–70-е годы возглавлявший одно из главных управлений КГБ, в состав которого входили подразделения радиоразведки*).

ANF *abbr* АНФ (*см. algebraic normal form*).

ANSI *abbr* АНИС (*см. American National Standards Institute*).

ant *n* "муравей" (*сверхминиатюрная аппаратная закладка, предназначенная для внедрения в аппаратуру шифрования*).

antenna *n* антенна.

aperiodic *adj* апериодичный; ~ **polyalphabetic substitution cipher** апериодичный шифр многоалфавитной замены.

aperiodic *n* = ~ polyalphabetic substitution cipher.

APNF *abbr* ПСНФ (*см. almost perfect non-linear function*).

a posteriori *adj лат.* апостериорный; ~ **probability** апостериорная вероятность.

append *v* добавлять; ~ **MAC to information being authenticated** добавлять КАС к информации, подлежащей аутентификации.

application *n* приложение, практическое применение; ~**-directed cryptography** = applied cryptography.

applied *adj* прикладной; ~ **cryptanalysis** прикладной криптоанализ; ~ **cryptography** прикладная криптография.

approach *n* метод; подход; ~ **to the solution of a cipher** метод вскрытия шифра.

appropriate *adj* соответствующий; ~ **key** соответствующий ключ (*напр., ключ расшифрования, соответствующий ключу зашифрования*).

approximation *n* приближение, аппроксимация.

a priory *adj лат.* априорный; ~ **probability** априорная вероятность.

arbiter *n* арбитр.

arbitrary *adj* произвольный; ~**-chosen key** произвольно выбранный ключ; ~ **key** произвольный ключ.

arbitrate *n* разрешать (*споры, конфликты*).

arbitration *n* арбитраж.

arbitrator *n* = arbiter.

arithmetic *n* арифметика; ~ **of primes, prime products, Galois fields** арифметика простых чисел, произведений простых чисел, полей Галуа.

-ary *suff* —арный; **N-~ alphabet** N-арный алфавит (*алфавит из N знаков*).

ascending *adj* возрастающий; ~ **key** возрастающий ключ.

ASCII *abbr* АСКОИ (*см. American Standard Code for Information Interchange*).

Asiacrypt *prop* Азиякрипто (*название конференции по криптологии, которая проводится в азиатско-тихоокеанском регионе*).

assess *v* оценивать; ~ **security of cryptosystems** оценивать стойкость криптосистем.

assignment *n* назначение, распределение.

assumption *n* = crib.

as yet *adv* все еще, до сих пор; ~ **unbroken knapsack cryptosystem** все еще невскрытая ранцевая криптосистема.

asymmetric *adj* асимметричный; ~ **algorithm** асимметричный алгоритм (*шифрования*); ~ **encryption** асимметричное шифрование; ~ **key** асимметричный ключ (*только для шифрования или только для расшифрования*); ~ **keys** ключи асимметричной криптосистемы (*разные для зашифрования и расшифрования*).

asymmetrically *adv* асимметрично; ~ **encrypted** зашифрованный с помощью асимметричного алгоритма.

asynchronous *adj* асинхронный; ~ **attack** асинхронная атака; ~ **attack secure** защищенный от асинхронной атаки; ~ **mode** асинхронный режим; ~ **stream cipher** асинхронный потоковый шифр.

attachment *n* приставка.

attack *n* атака, криптоанализ, попытка вскрытия (*криптосистемы*); ~ **by iteration** криптоанализ методом итераций; ~ **on an encryption algorithm** атака на алгоритм шифрования; ~**-proof** криптостойкий, стойкий к криптоанализу; ~**-resistant** = ~-proof.

attack *v* атаковать, подвергать криптоанализу, пытаться вскрыть; ~ **the DES** пытаться вскрыть DES.

attacker *n* нарушитель; противник; криптоаналитик.

attribute *n* свойство; ~ **of a cryptosystem** свойство криптосистемы.

attribute *v* приписывать авторство; ~ **a message to a transmitter** приписывать авторство сообщения передатчику.

audio *adj* звуковой; ~ **encryption** шифрование звуковых сигналов.

Auscrypt *prop* Авскрипто (*название конференции по криптологии, проводимой в Австралии*).

authentic *adj* истинный, настоящий; ~ **messages** истинные сообщения.

authenticate *n* аутентифицировать; ~ **a message** аутентифицировать сообщение.

authenticated *adj* аутентифицированный; ~ **communication** связь с (*взаимной*) аутентификацией партнеров; ~ **key agreement protocol** протокол согласования ключа с аутентификацией.

authenticating *adj* аутентификационный.

authentication *n* (1) аутентификация, опознавание; ~ **channel** канал аутентификации; ~ **code** аутентификационный код; ~ **code with arbitration** аутентификационный код с арбитражем; ~ **function** аутентификационная функция; ~ **message** аутентификационное сообщение; ~ **pattern** комбинация для аутентификации; ~ **system** система аутентификации (*криптосистема или криптографический процессор, используемые для аутентификации*); (2) подтверждение права на доступ; (3) проверка подлинности [достоверности]; ~ **exchange** проверка подлинности с обменом информацией; ~ **problem** задача проверки на достоверность.

authenticator *n* аутентификатор (*избыточная информация, добавляемая в сообщение для его аутентификации*).

authenticity *n* подлинность; ~ **system** система с проверкой подлинности.

authority *n* (1) полномочие; (2) авторитет, крупный специалист; ~ **on encryption** крупный специалист в области шифрования.

authorized *adj* (1) санкционированный; ~ **originator for messages** санкционированный источник сообщений; (2) назначенный; ~ **key** назначенный ключ.

autoclear *n* автоматическое переключение [автоматический переключатель] связи с засекреченной на открытую.

autocorrelation *n* автокорреляция; ~ **function** функция автокорреляции; ~ **test** автокорреляционный тест (*используется для проверки случайности двоичной последовательности*).

autoencipherment *n* шифрование на автоключе.

autokey *n* автоключ; ~ **cipher system** автоключевая шифрсистема (*каждая буква открытого / шифрованного текста сообщения, полученная в результате расшифрования / зашифрования очередной буквы шифрованного / открытого текста, используется в качестве ключа для расшифрования / зашифрования следующей*).

autokeying *n* (1) шифрование на автоключе; (2) формирование автоключа.

auto-manual *adj* полуавтоматический; ~ **system** полуавтоматическая система (*программируемое переносное криптографическое оборудование*);.

automated *adj* автоматизированный; ~ **cryptanalysis** автоматизированный (*компьютерный*) криптоанализ; ~ **key distribution** автоматизированное распределение ключей; ~ **key distribution system** система автоматизированного распределения ключей.

automatic *adj* автоматический; ~ **key distribution center** центр автоматического распределения ключей; ~ **key management** автоматическое управление клю-

чами; ~ **remote rekeying** автоматический удаленный ввод ключа (*процедура, применяемая для смены ключа в расположенном на удалении криптооборудовании и не требующая участия местного обслуживающего персонала*).

automorphism *n* автоморфизм.

auto-session *adj* автоматически генерируемый для сеанса; ~ **key** автоматически генерируемый сеансовый ключ.

auxiliary *adj* вторичный; ~ **key** вторичный ключ.

avalanche *n* "снежная лавина" (*свойство блочных шифров реагировать на небольшие изменения на входе лавинообразными изменениями на выходе уже после нескольких циклов работы*).

B

back-break *v* вскрывать (*криптограмму, ключ*) "задним числом" (*то есть по прошествии некоторого периода времени, за который информация, полученная в результате вскрытия, успевает устареть и потерять свою актуальность*).

backdoor *n* = trapdoor.

backup *n* создание резервных копий.

backward *adj* обратный; ~ **asymmetric encryption** шифрование с обратной асимметрией (*ключ отправителя не может быть определен по ключу получателя*).

Baconian *adj* бэконовский; ~ **cipher** бэконовский шифр (*назван так по имени английского философа Ф. Бэкона, который изобрел этот шифр*).

Bad-Guy *n жарг. тж.* **bad-guy** (1) злоумышленник; (2) криптоаналитик (*противника*).

balanced *adj* сбалансированный; ~ **function** сбалансированная функция (*двоичная функция от n переменных, таблица истинности которой содержит* 2^{n-1} *нулей*).

band *n* полоса, диапазон, спектр (*частот сигнала*); ~**scrambler** спектрально-полосный скремблер; ~**scrambler-inverter** спектрально-полосный скремблер с инверсией отдельных полос; ~**shift** сдвиг частотных полос; ~**shift inverter** скремблер со сдвигом частотных полос и инверсией частот ~**splitter** = ~scrambler; ~**splitting** разделение спектра сигнала на отдельные полосы.

base *adj* основной; ~ **key** основной ключ.

base *n* основание; **to the** ~ **b** по основанию b.

basic *adj* элементарный; ~ **cryptanalysis** элементарный криптоанализ [основы криптоанализа].

basis *n* базис (*векторного пространства*).

batch *adj* групповой; ~ **key** групповой ключ.

BC *abbr* СБ (*см. block chaining*).

B-Crypt *abbr* (*British Telecom Cryptoalgorithm*) Б-Крипто (*алгоритм шифрования, разработанный английской фирмой "Бритиш Телеком"*).

B-Dienst *abbr* (*от нем. Beobachtungs-Dienst*) "Служба наблюдения" (*военно-морская спецслужба радиоперехвата гитлеровской Германии*).

Beale *prop* Биль Т. (*американский авантюрист, который, накопив вместе с сообщниками огромные сокровища, зарыл их где-то на территории штата Виржиния и с помощью трех шифров засекретил: (а) местонахождение этих сокровищ; (б) их подробное описание; (в) имена законных наследников*); ~ **cipher** шифр Биля.

Beaufort *prop* Бофор Ф. (*адмирал английского флота*); ~ **cipher** шифр Бофора; ~ **square** квадрат Бофора.

best *superl* лучший, наилучший; ~ **of known attacks on a cipher** наилучший из методов криптоанализа данного шифра; ~ **possible cryptanalytic algorithm** наилучший криптоаналитический алгоритм.

Beurling *prop* Берлинг А. (*шведский криптоаналитик, на счету которого несколько успешных вскрытий советских и немецких шифров в 30-е—40-е годы*).

Bible *prop* Библия; ~ **code** "библейский" код (*согласно гипотезе израильского математика Ильи Рипса, Библия содержит кодированный текст, дешифровав который можно узнать многое о событиях, произошедших спустя тысячи лет после ее написания*).

bicomposite *adj* двухсоставной; ~ **modulus N** двухсоставной модуль N.

bidirectional *adj* двухсторонний; ~ **asymmetric encryption** шифрование с двухсторонней асимметрией (*ключ получателя не может быть определен по ключу отправителя и наоборот*); ~ **key** двухсторонний ключ (*может быть использован как для зашифрования, так и для расшифрования сообщения*).

bifid *adj* раздвоенный; ~ **cipher** раздвоенный шифр (*разновидность шифра колонной перестановки*).

Big Brother *prop* "Большой Брат" (*аллегория тотальной слежки со стороны государства за своими гражданами из романа английского писателя Дж. Оруэлла "1984"*); ~ **is watching you** "Большой Брат" следит за тобой.

big-O *n* "О-большое" (*функция $f(x)=O(g(x))$, если существует константа C такая, что $f(x)\leq Cg(x)$*).

bigram *n* биграмма (*два символа текста*); ~ **cipher** биграммный шифр; ~ **count** маркировка биграмм; ~ **counting** = ~ count; ~ **frequency** частота встречаемости [появления] биграмм (*в тексте*).

bijection *n* биекция (*взаимно однозначное отображение*).

bijective *adj* биективный; ~ **public-key cryptosystem** биективная криптосистема с открытым ключом.

bilateral *adj* двухсторонний; ~ **key** ключ, известный двум пользователям [ключ для двухсторонней связи]; ~ **key exchange** двухсторонний обмен ключами.

bi-letter *adj* = two-letter.

bilinear *adj* билинейный; ~ **cipher** билинейный шифр.

biliteral *adj* состоящий из пар символов, двухбуквенный; ~ **cipher alphabet** шифралфавит, составленный из пар символов; ~ **cipher system** двухбуквенная шифрсистема (*при зашифровании каждый символ открытого текста заменяется на два символа шифрованного*); ~ **cipher system with variants** вариантная двухбуквенная шифрсистема (*при зашифровании каждый символ открытого текста может заменяться на различные пары символов шифрованного*).

biliteral *n* двухбуквенная шифрсистема.

binary *adj* двоичный; ~ **additive stream cipher** шифр гаммирования с двоичной шифрующей [ключевой] последовательностью; ~ **cryptosystem** криптосистема с двоичным ключом; ~ **function** двоичная функция; ~ **key** двоичный ключ; ~ **power** степень двойки (*число вида 2^n, где n — целое*); ~ **sequence** двоичная последовательность; ~ **sequence generator** генератор двоичных последовательностей; ~ **substitution** = ~ substitution cipher; ~ **substitution cipher** шифр двоичной замены.

binomial *adj* биномиальный; ~ **distribution** биномиальное распределение.

birthday *n* день рождения; ~ **attack** криптоанализ на основе парадокса дней рождений; ~ **paradox** парадокс дней рождения.

bit *n* бит (*единица информации*); ~**-by-bit encryption** побитовое шифрование; ~**-cipher feedback** обратная связь по битам шифртекста; **m-**~ **ciphertext feedback cipher** шифр с обратной связью по m битам (*предшествующего*) шифртекста; ~**-differences** различия в битах; **128-**~ **key** 128-битовый ключ; ~**-length** длина (*сообщения / ключа*) в битах; ~**map** двоичное [побитовое] отображение; ~ **mapping** = ~ map; ~ **of a key** бит ключа; ~ **operation** битовая операция; ~**-oriented cipher feedback** = ~-cipher feedback; **m-**~ **output feedback** = m-~ ciphertext feedback cipher; ~ **pattern** битовая комбинация [набор битов] (*напр., для представления знака кода*); ~ **per second** бит в секунду; ~**-security** стойкость по битам (*невозможность определения отдельных битов открытого текста по имеющемуся шифрованному тексту*); ~**-sequence** битовая [двоичная] последовательность; ~**serial** побитовый; ~ **stream cipher** поточный шифр с двоичной шифрующей [ключевой] последовательностью; ~ **stream generator** = binary sequence generator; ~**string** строка битов; ~ **substitution cipher** шифр побитовой замены.

bitwise *adj* побитовый, поразрядный; ~ **substitution cipher** = bit substitution cipher; ~ **XOR** побитовое исключающее ИЛИ.

bitwise *adv* побитово, поразрядно.

black *adj* (1) черный; ~ **box cryptanalysis** криптоанализ методом "черного ящика"; ~ **chamber** "черный кабинет"; ~ **Code** "Черный" код (*американский дипломатический код, выкраденный итальянской спецслужбой из посольства США в Италии незадолго до второй мировой войны*); (2) шифрованный, зашифрованный; ~ **box** блок обработки шифрованной информации; ~ **designation** обозначение систем, линий и аппаратуры связи для передачи и обработки шифрованной информации; ~ **side** сторона шифратора, обрабатывающая секретные данные.

blank *n* пробел (*в тексте*).

Bletchley Park *prop* Блетчли-Парк (*поместье викторианской эпохи примерно в 100 км от Лондона, где в годы второй мировой войны размещалась английская дешифровальная служба, занимавшаяся чтением немецкой шифрпереписки*).

blind *adj* слепой; ~ **signature scheme** схема слепой цифровой подписи [схема цифровой подписи вслепую].

block *n* блок; ~**-by-**~ поблочно [блоками]; ~ **chaining** сцепление блоков; ~ **cipher** блочный шифр; ~**-cipher cryptanalysis** криптоанализ блочных шифров; ~**-ciphering algorithm** алгоритм блочного шифрования; ~ **cipher with chaining** блочный шифр со сцеплением блоков; ~ **encryption** блочное шифрование; ~ **frequency analysis** криптоанализ на основе частоты встречаемости [появления] блоков (*в тексте*); ~ **note** = one-time pad; ~ **of plaintext** блок открытого текста; ~ **padding** холостое заполнение блока (*ничего не значащей или фальшивой информацией*); ~ **size** размер [длина] блока.

blocking *n* разделение (*текста*) на блоки, формирование блоков (*сообщения*).

BND *abbr* БНД (*разведывательное агентство Германии, в функции которого, в частности, входит ведение радиоразведки*).

Bob *prop* Боб (*гипотетическая личность, выполняющая необходимые действия при реализации криптографических протоколов, в которых имеется по крайней мере два участника*).

bogus *adj* поддельный; ~ **key** поддельный ключ.

Bombe *prop тж.* **bombe** "Бомба" ["бомба"] (*электромеханическое устройство, созданное в годы второй мировой войны англичанами для автоматизации процесса вскрытия немецкой шифровальной машины "Энигма"*); ~ **operator** оператор "Бомбы".

book *n* книга; ~ **cipher** книжный шифр (*шифр, составленный с использованием книги шифрования*); ~ **code** = ~ cipher.

Boole *prop* Буль Дж. (*английский математик*).

Boolean *adj* булев, двоичный; ~ **equation** булево уравнение; ~ **function** булева [двоичная] функция.

boot *n* начальная загрузка (*компьютера*); ~ **protection encryption** шифрование для защиты [от] начальной загрузки.

box *n* блок, модуль, стойка.

BP *abbr* БП (*см. Bletchley Park*).

break *n* вскрытие (*ключа или шифра*); ~ **into Enigma** вскрытие "Энигмы"; ~ **length** длина шифртекста, достаточная для вскрытия ключа; ~ **time of a base key** время, требуемое для вскрытия основного ключа.

break *v* (1) вскрывать; ~ **a cipher** вскрывать шифр; ~ **a code** вскрывать код; ~ **a cryptographic protocol** вскрывать криптографический протокол; ~ **an encryption** = ~ a cipher; (2) дешифровывать; ~ **a cryptogram** дешифровывать криптограмму; (3) разбивать, разделять; ~ **a text into digraphs** разбивать текст на биграммы.

breakability *n* вскрываемость, дешифруемость.

breakable *adj* вскрываемый, дешифруемый, поддающийся вскрытию или дешифрованию; ~ **cipher** вскрываемый шифр.

breaker *n* криптоаналитик (*букв. взломщик*).

break down *v* = break.

break in *v* взламывать, вторгаться; ~ **a system** взламывать [вторгаться в] систему.

break-in *n* взлом, вторжение; ~ **of a system** взлом системы [вторжение в систему].

breaking *n* вскрытие; ~ **algorithm** алгоритм вскрытия (*шифра*); ~ **program** программа вскрытия (*ключа или шифра*).

broadcast *n* циркулярная передача, циркулярная связь; ~ **authentication** аутентификация при циркулярной передаче; ~ **channel** канал циркулярной передачи; ~ **encryption scheme** схема шифрования для циркулярной связи.

broadcasting *n* = broadcast.

brute *adj* грубый; ~**-force attack** атака с применением грубой силы [силовая атака]; ~**-force search** = ~-force attack.

build *v* (1) встраивать; ~ **a trapdoor into a knapsack one-way function** встраивать лазейку [потайной ход] в одностороннюю ранцевую функцию; (2) создавать, конструировать; ~ **a secure cryptographic system** создавать надежную криптографическую систему.

built-in *adj* встроенный, вмонтированный; ~ **encryption module** встроенный модуль шифрования; ~ **key** вмонтированный ключ.

bulk *n* большое количество; ~ **encryption** групповое шифрование (*одновременное шифрование информации, передаваемой по всем каналам многоканальной телекоммуникационной системы*); ~ **encryption device** групповой шифратор.

burst *n* пакет; ~ **communication** сверхбыстрая пакетная связь; ~ **transmission** сверхбыстрая пакетная передача.

busting *n* вскрытие (*шифра или ключа*).

button *n* кнопка, клавиша.

by-pass *n* обход; ~ **mode** режим обхода [режим работы с обходом] (*открытый текст подается минуя криптографическое устройство и в силу этого не подвергается шифрующим преобразованиям*).

byte *n* байт (*набор из 8 бит*); ~ **of a key** байт ключа; ~**-oriented block-ciphering algorithm** алгоритм побайтового блочного шифрования (*при зашифровании и расшифровании используются только операции над байтами*); ~ **per second** байт в секунду; ~ **substitution table** таблица байтовых замен (*одномерный массив из 256 чисел от 0 до 255, каждое из которых встречается в нем лишь единожды*); ~ **transposition cipher** шифр побайтовой перестановки.

C

C *abbr* (1) ДСП (*см. confidential*); (2) ШТ (*см. ciphertext*); (3) К (*см. commercial*).

CA *abbr* ОС (*см. certification authority*).

cable *n* (1) кабель; ~ **communication** кабельная связь; ~ **interception** перехват в кабельной линии связи; ~ **tapping** = ~ interception; (2) = cablegram.

cable *v* телеграфировать.

cablegram каблограмма, телеграмма.

Caesar *prop* Цезарь Ю. (*древнеримский император*); ~ **cipher** шифр Цезаря.

calculate *v* вычислять; ~ **an encryption key given a decryption key** вычислять ключ зашифрования при известном ключе расшифрования.

call *n* сигнал; ~ **sign** позывной сигнал; ~ **sign cipher** шифратор позывных сигналов (*используется для шифрования позывных сигналов и другой сигнальной информации*); ~**word** слово-позывной.

call up *v* вызывать; ~ **procedure** процедура вызова.

candidate *n* кандидат; ~ **for a key** = ~ key; ~ **key** возможный ключ (*при криптоанализе*).

canister *n* канистра (*контейнер, используемый для хранения и транспортировки перфорированной или телетайпной ленты с ключевой информацией*).

CANYON *prop* "Каньон" (*серия разведывательных спутников АНБ США, первый из которых был выведен на орбиту в 1968 г.*).

CAPI *abbr* КПИП (*см. Cryptographic Application Programming Interface*).

Capstone *n* "Кэпстоун" [*букв.* "Облицовочный камень"] (*название долгосрочной программы правительства США по разработке открытых криптографических стандартов*).

capture *v* (1) захватывать; ~ **a codebook** захватывать кодовую книгу; (2) перехватывать; ~ **all communications** перехватывать все сообщения.

cargo *n* груз; ~ **vector** вектор груза.

carrier *n* (1) носитель (*информации*); (2) несущая (*частота*).

CAST *abbr* КАСТ (*блочный алгоритм шифрования, разработанный сотрудниками транснациональной корпорации "Нортел Нетворкс" Карлайслем Адамсом и Стэффордом Таваресом*).

category *n* категория (*пометка на документе, свидетельствующая о степени его секретности*).

catoprical *adj* катоптрический; ~ **cryptography** катоптрическая криптография (*с использованием зеркал*).

cascade *adj* каскадный; ~ **cipher** каскадный шифр.

cascade *v* каскадировать, включать каскадно; ~ **linear feedback shift registers** каскадировать регистры сдвига с линейной обратной связью.

cascaded *adj* каскадный; ~ **connection of shift registers** каскадное соединение регистров сдвига; ~ **superencipherment** каскадное наложение шифров.

CBC *abbr* СБШ (*см. cipher block chaining*).

CBMS *abbr* КСПС (*см. computer-based message system*).

CCB *abbr* ЦШБ (*см. Central Cipher Bureau*).

CCCC *abbr* ЦРКЦ (*см. Centralized COMINT Communications Center*).

CCCEP *abbr* ПАКСЗС (*см. Commercial Communications Security Endorsement Program*).

CCH *abbr* ЦИК (*см. Center for Cryptologic History*).

CCI *abbr* ККИ (*см. controlled cryptographic item*).

CCSC *abbr* ЦБКВС (*см. Commercial Computer Security Center*).

CDK *abbr* НСК (*см. cryptodeveloper kit*).

cellular *adj* (1) клеточный; ~ **automaton** клеточный автомат; (2) сотовый; ~ **mobile phones** сотовые мобильные телефоны.

center *n* центр; ~ **for Cryptologic History** Центр истории криптологии; ~ **key** ключ центра распределения ключей.

central *adj* центральный; ~ **Cipher Bureau** Центральное шифровальное бюро (*немецкое правительственное ведомство, в 60-е—80-е годы занимавшееся перехватом и дешифрованием сообщений*); ~ **Intelligence Agency** Центральное разведывательное управление (*правительственное ведомство США, одной из функций которого являются организация и проведение радиоразведывательных операций*); ~ **keying authority** центр распределения [распространения] ключей (*выдает ключи и образцы подписей абонентам сети связи, ведет учет их работы и выступает в качестве свидетеля или арбитра при возникновении споров*); ~ **Office for Cryptology** Центральная криптологическая служба (*подразделение германской Федеральной разведывательной службы, занимающееся перехватом*

и дешифрованием сообщений); ~ **Security Service** Центральная служба безопасности (*подразделение министерства обороны США, находящееся в подчинении директора АНБ и осуществляющее радиоперехват в интересах этого агентства*); ~ **Service for Security and Information Systems** Центральная служба безопасности и информационных систем (*французское правительственное ведомство, занимающееся перехватом и дешифрованием сообщений*).

centralized *adj* централизованный; ~ **COMINT Communications Center** Централизованный радиоразведывательный коммуникационный центр (*ретрансляционная станция АНБ США, предназначенная для передачи радиоразведывательных данных*); ~ **key distribution** централизованное распределение ключей; ~ **key generation** централизованная генерация ключей; ~ **key infrastructure** централизованная инфраструктура ключей; ~ **key management** централизованное управление ключами.

centrally *adv* централизованно; ~**-produced key** ключ, генерируемый в центре (*распределения ключей*).

cepstrum *adj* кепстральный; ~ **analysis** кепстральный анализ.

CER *abbr* ПКО (*см. cryptographic equipment room*).

certificate *n* сертификат (*сообщение, содержащее ключ, подлинность которого устанавливается при помощи криптографической аутентификации*); ~ **revocation list** список аннулированных сертификатов.

certification *n* аттестация (*проверка на соответствие определенным требованиям*); сертификация (*доказательство соответствия нормам и стандартам*); освидетельствование, удостоверение; ~ **authority** орган сертификации; ~ **authority workstation** рабочая станция органа сертификации (*компьютер, на котором установлена защищенная операционная система и специальное программное обеспечение для оформления сертификатов*); ~ **of origination** удостоверение источника; ~ **of receipt** удостоверение получателя.

certificational *adj* сертификационный; ~ **weakness** сертификационная нестойкость (*потенциальная возможность вскрыть шифр с помощью любого криптоаналитического метода, являющегося более эффективным по сравнению с полным перебором*).

certified *adj* сертифицированный, заверенный; достоверный; ~ **cryptosystem** сертифицированная криптосистема; ~ **key** заверенный ключ; ~ **prime** достоверно простое число; ~ **public directory** заверенный открытый справочник (*ключей или криптоалгоритмов*).

certify *v* сертифицировать, заверять, удостоверять.

CFB *abbr* ОСШ (*см. cipher feedback*).

CFM *abbr* РШО (*см. cipher feedback mode*).

chain *n* = chaining.

chained *adj* сцепленный; ~ **key** сцепленный ключ.

chaining *n* сцепление; ~ **cipher** шифр со сцеплением блоков.

CHALET *prop* "Шале" (*серия разведывательных спутников АНБ США, первый из которых был выведен на орбиту в 1978 г.*).

change *n* смена, замена; ~ **of keys** смена ключей.

change *v* сменять, заменять; ~ **a code** заменять [сменять] код; ~ **a key** заменять [сменять] ключ.

changeover *n* переход; ~ **to a spare key** переход на запасной ключ.

channel *n* канал, линия (*связи*); ~ **capacity** пропускная способность канала.

character *n* символ, знак; **n-~ code** n-буквенный код; ~ **dependency** [взаимная] зависимость символов (*текста*); ~ **frequency** частота встречаемости [появления] букв (*в тексте*); ~ **frequency count** подсчет частоты встречаемости букв (*в тексте*); ~**-oriented communication** посимвольная передача данных; ~ **set** набор символов; ~**-to-number replacement** замена знаков числами [подстановка чисел вместо знаков].

characteristic *n* (1) характеристика (*различие между двумя открытыми текстами, которое с большой вероятностью приводит к появлению определенного различия в соответствующих им шифрованных текстах*); (2) характеристика (*поля*); (3) (*рабочая*) характеристика (*параметр, характеризующий объем работ, который требуется выполнить для достижения требуемого результата*); (4) отличительный признак; ~**s of polygraphic ciphers** отличительные признаки полиграфических шифров.

characteristical *adj* характеристический; ~ **polynomial** характеристический многочлен.

cheater *n* нарушитель, обманщик.

cheating *n* нарушение, обман.

check *adj* контрольный, проверочный; ~ **decryption** контрольное [проверочное] расшифрование; ~**sum** контрольная сумма; ~**sum generator** генератор контрольной суммы; ~**summing** вычисление контрольной суммы; ~**value** контрольная величина, контрольное значение (*напр., контрольная сумма*); ~ **word** проверочное слово (*шифртекст, генерируемый шифровальным оборудованием в целях обнаружения и исправления погрешностей в процессе шифрования*).

Cheltenham *prop* Челтнем (*курортный город в графстве Глостершир в Англии, где находится штаб-квартира ЦПС*).

chi *n* с (*22-я буква греческого алфавита*); ~**-square distribution** распределение c^2; ~**-square criteria** критерий c^2; ~ **test** с-тест.

chief *adj* главный; ~ **cryptanalyst** главный криптоаналитик (*название одной из руководящих должностей в военном министерстве США во время второй мировой войны*).

children *n pl* потомки; ~ **key** производный [вторичный] ключ.

Chinese *adj* китайский; ~ **Remainder Theorem** китайская теорема об остатках.

chip *n* чип, микросхема, полупроводниковый кристалл.

chosen- *в сложных и сложносоставных словах имеет значение* на основе выбранного.

chosen-ciphertext *adj* на основе выбранного шифртекста; ~ **attack** атака на основе выбранного шифртекста; ~ **attack secure** защищенный от атаки на основе выбранного шифртекста.

chosen-content *adj* на основе выбранного содержимого; ~ **attack** атака на основе выбранного содержимого (*противник составляет ложную криптограмму, определяя ее содержание по своему усмотрению*); ~ **attack secure** защищенный от атаки на основе выбранного содержимого.

chosen-cryptogram *adj* = chosen-ciphertext.

chosen-key *adj* на основе выбранного ключа; ~ **attack** атака на основе выбранного ключа; ~ **attack secure** защищенный от атаки на основе выбранного ключа.

chosen-plaintext *adj* на основе выбранного открытого текста; ~ **attack** атака на основе выбранного открытого текста; ~ **attack secure** защищенный от атаки на основе выбранного открытого текста.

chosen-signature *adj* на основе выбранных подписей; ~ **attack** атака на основе выбранных подписей; ~ **attack secure** защищенный от атаки на основе выбранных подписей.

chosen-text *adj* на основе выбранного (*открытого и шифрованного*) текста; ~ **attack** атака на основе выбранного текста; ~ **attack secure** защищенный от атаки на основе выбранного (*открытого и шифрованного*) текста.

CIA *abbr* ЦРУ (*см. Central Intelligence Agency*).

CIB *abbr* СРР (*см. Communications Intelligence Board*).

cifax *abbr* шифакс (*см. ciphered facsimile communication*).

CIK *abbr* КИКО (*см. cryptoignition key*).

cipher *n* шифр (*преобразование текста для сокрытия его семантического содержания*); ~ **adventure** = cryptothriller; ~ **algorithm** = ciphering algorithm; ~ **alphabet** шифралфавит; ~ **analysis** = cryptanalysis; ~**-analyst** = cryptanalyst; ~ **attachment** приставка-шифратор; ~ **block** блок шифра [шифрблок]; ~ **block chaining** сцепление блоков шифра (*один из режимов использования DES-алгоритма*); ~**-breaker** = cryptanalyst (*букв. взломщик шифров*); ~**-breaking device** устройство для вскрытия [взлома] шифров; ~ **Bureau** Шифровальное бюро (*правительственное криптоаналитическое ведомство в Польше в период между двумя мировыми войнами*); ~ **busting machine** машина для вскрытия шифра; ~ **center** шифровальный центр; ~ **clerk** шифровальщик; ~ **clock** шифровальный циферблат

(*графическое представление шифра простой замены с помощью окружности, с внутренней стороны которой выписываются буквы алфавита открытого текста, а с внешней — шифрованного*); ~ **code** шифровальный код; ~ **construction** конструирование шифров; ~ **control byte** байт контроля шифра; ~ **designer** разработчик шифров; ~ **device** шифровальная машина; ~ **feedback** обратная связь по шифртексту (*один из режимов использования DES-алгоритма*); ~ **feedback mode** режим шифрования с обратной связью по шифртексту; ~**function** шифрфункция; ~ **group** шифргруппа (*шифруемая группа знаков*); ~ **key** ключ к шифру [ключ шифрования]; ~ **machine** шифровальная машина; ~ **material** шифрматериал; ~ **message** шифрованное сообщение, шифровка; ~ **officer** = ~ clerk; ~**'s performance** скорость шифрования; ~**punk** шифрпанк (*человек, который придерживается мнения о том, что любая информация частного характера неприкосновенна и должна быть надежно защищена с помощью стойких криптографических алгоритмов*); ~ **queen** *воен. жарг.* шифровальщица; ~ **security** меры безопасности при работе с шифрами; ~**speak** шифртерминология, шифржаргон; ~ **suit** шифровальный комплект (*набор алгоритмов шифрования, реализованных в данной системе передачи сообщений*); ~ **system** шифрсистема; ~ **system identification** определение шифрсистемы; ~ **taxonomy** таксономия шифров; ~ **throughput** = ~ performance.

cipher *v* шифровать, зашифровывать.

ciphered *adj* шифрованный, зашифрованный; ~ **code** = cipher code; ~ **facsimile communication** (1) криптографическая защита факсимильной связи; (2) шифрованное факсимильное сообщение.

cipherer *n* (1) кодирующее устройство, шифратор; (2) шифровальщик.

ciphering *n* шифрование; ~ **algorithm** алгоритм шифрования; ~ **function** функция шифрования; ~ **generator** генератор шифрующих [ключевых] последовательностей; ~ **method** метод шифрования; ~ **mode** режим шифрования; ~ **step** шаг шифрования.

ciphertext *n* шифртекст (*полученный в результате шифрования текст*); ~ **associated with a given key** шифртекст, полученный на данном ключе; ~ **autokey** автоключ, зависящий от шифртекста; ~ **autokey cipher** шифр с автоключом, зависящим от шифртекста; ~ **character** знак шифртекста; ~ **expansion** экспансия шифртекста (*полученный в результате шифрования текст имеет большую длину по сравнению с исходным открытым текстом*); ~ **feedback** = cipher feedback; ~ **file** файл с шифртекстом; ~ **generator** генератор шифртекста; ~ **letter** буква [буквенный знак] шифртекста; ~ **letter frequencies** частоты встречаемости букв в шифртексте; ~ **message** шифрованное сообщение [шифрованный текст сообщения]; ~**-only attack** атака на основе знания одного только шифртекста; ~**-only attack secure** защищенный от атаки на основе знания одного только шифртекста.

ciphony *n* (1) криптографическая защита телефонной связи; (2) шифрованное речевое сообщение.

circuit *n* (1) сеть; цепь; (2) схема.

circular *adj* = cyclical.

civil *adj* = civilian.

civilian *adj* гражданский; ~ **cryptography** криптография для гражданских нужд (*в отличие от криптографии для военных и дипломатов*).

civision *n* (1) криптографическая защита телевизионных передач; (2) шифрованная телевизионная передача.

CKA *abbr* ЦРК (*см. central keying authority*).

class *n* класс; ~ **NP** класс NP (*класс математических задач, которые не могут быть решены за полиномиальное время на недетерминированной машине Тьюринга*); ~ **P** класс P (*класс математических задач, которые могут быть решены за полиномиальное время на недетерминированной машине Тьюринга*).

classical *adj* классический; ~ **cipher** классический шифр (*использующий перестановку и замену букв алфавита естественного языка*); ~ **cryptography** классическая (*одноключевая*) криптография.

classified *adj* секретный, закрытый; ~ **cryptoalgorithm** секретный криптоалгоритм; ~ **key** секретный ключ; ~ **message** секретное сообщение; ~ **network** сеть засекреченной связи; ~ **work in cryptography** закрытые работы по криптографии.

clear *adj* (1) открытый, незашифрованный; ~ **character** знак открытого текста; ~ **communication** открытая связь; ~**-crypto mode** режим работы (*шифратора*) с переходом с открытой на шифрованную связь и обратно; ~ **key** незашифрованный ключ; ~ **message** открытое [незашифрованное] сообщение; ~ **mode** режим открытой связи; ~ **operation** работа (*шифратора*) в режиме приема-передачи открытого текста [работа в открытом режиме]; ~ **override** переключение (*шифратора*) с открытой на шифрованную связь (*при поступлении на его вход шифртекста*); ~ **to send** разрешенный к передаче; (2) *в грам. знач. сущ.* клер, открытый текст.

clearance *n* допуск, право доступа.

cleartext *n* открытый [исходный] текст (*данные с доступным семантическим содержанием*); ~**-ciphertext expansion** расширение шифртекста относительно открытого [исходного] текста; ~ **message** = clear message; ~ **operation** = clear operation.

Clipper *n* "Клиппер" [*букв.* "Клипер"] (*название чипа для криптографической защиты информации*).

cloak *n* генератор "белого шума" (*прибор для шумовой маскировки*).

clock *n* тактирование; синхронизация; ~**-controlled shift register** регистр сдвига с тактированием.

clustering *n* группирование, кластеризация.

CNCS *abbr* УСКС (*см. cryptonet control station*).

CNK *abbr* ККС (*см. cryptonet key*).

code *n* код (*метод преобразования открытого текста в шифрованный путем использования кодовых таблиц*); ~ **alphabet** алфавит кода; ~**book** кодовая книга; ~**breaker** криптоаналитик, дешифровальщик, *букв.* взломщик кодов; ~**breakers** "Взломщики кодов" (*название книги американского писателя Д. Кана, посвященной истории криптологии с древних времен до наших дней*); ~**-breaking** вскрытие [взлом] шифра; ~**breaking algorithm** алгоритм вскрытия кода; ~**breaking organization** криптоаналитическая организация; ~**breaking exploits** криптоаналитические подвиги; ~**builder** разработчик кодов; ~**change** смена кода; ~ **character** символ кода; ~ **clerk** шифровальщик; ~ **Compilation Company** "Компания по составлению кодов" (*название фиктивного коммерческого предприятия, служившего прикрытием для "Американского черного кабинета"*); ~**cracker** = codebreaker; ~**-cracking** = ~-breaking; ~ **generator** генератор кода; ~**group** кодовая группа; ~ **letter** буква [буквенный знак] кода; ~**like** кодоподобный; ~**maker** составитель [разработчик] кодов; ~**making** составление [разработка] кодов; ~ **material** кодированный материал; ~ **name of an intercept station** кодовое имя станции перехвата; ~**number** кодовое число; ~ **phrase** кодовая фраза; ~ **recovery** восстановление кода; ~ **system** система кодирования; ~**text** кодированный текст; ~ **translation** перекодирование [повторное кодирование в другом коде]; ~ **vocabulary** кодовый словарь (*слова и фразы открытого текста, для которых в данном коде имеются соответствующие кодовые обозначения*); ~**word** кодовое слово, кодообозначение.

code *v* кодировать.

codec·*n* шифратор-дешифратор.

coded *adj* кодированный; ~ **cable** кодированная телеграмма; ~ **communication** кодированная связь; ~ **message** кодированное сообщение; ~ **text** кодированный текст; ~ **transaction** кодированная транзакция.

codename *v* давать кодовое имя.

coder *n* (1) шифратор; (2) кодировщик, шифровальщик.

coding *n* кодирование; ~ **algorithm** кодирующий алгоритм; ~ **office** шифровальная служба; ~ **theory** теория кодирования.

coefficient *n* коэффициент; ~ **matrix of a system of linear equations** матрица коэффициентов системы линейных уравнений.

coin *adj* монетный; ~ **sequence** "монетная" последовательность (*двоичная последовательность, каждый компонент которой рассматривается как результат статистического испытания с двумя равновероятными исходами*).

coincidence *n* совпадение; ~ **test** тест на совпадения; ~**s between two cryptograms** совпадения между двумя криптограммами.

cold *adj* холодный; ~ **start-up** "холодный" запуск (*процедура первоначальной загрузки ключей в криптографическое оборудование*).

collect *v* собирать, получать; ~ **COMINT from satellite** собирать информацию РЭР со спутника; ~ **a large amount of plaintext and ciphertext** получить большое количество шифрованного и открытого текста.

collection *n* сбор, получение (*перехват и накопление сообщений для их дальнейшей обработки с целью получения разведывательной информации*).

Colossum *prop* = Colossus.

Colossus *prop* "Колосс" (*название электронно-механического устройства, созданного в годы второй мировой войны англичанами для автоматизации процесса вскрытия немецкой шифровальной машины "Энигма"*).

columnar *adj* колонный; ~ **transposition** шифр колонной перестановки.

combination *n* комбинация, сочетание; ~ **generator** комбинационный генератор (*порождает псевдослучайную последовательность путем объединения нескольких выходных последовательностей, полученных с помощью набора регистров сдвига, работающих параллельно*); ~ **of linear feedback shift registers** комбинация регистров сдвига с линейной обратной связью.

combined *adj* комбинированный; ~ **encryption** комбинированное шифрование (*межконцевое и канальное*).

combiner *n* смеситель (*функция или устройство для объединения нескольких числовых последовательностей в одну*).

combining function *n* = combiner.

COMINT *abbr тж.* **comint** РЭР (*см. communications intelligence*); ~ **agency** агентство РЭР; ~ **data** данные РЭР; ~ **equipment** оборудование РЭР; ~ **friendly** дружественный для РЭР (*не требует применения нетрадиционных, дорогостоящих или чреватых непредсказуемыми последствиями методов РЭР*); ~ **organization** организация РЭР; ~ **processing** обработка данных РЭР; ~ **purposes** цели РЭР; ~ **reports** сводки РЭР; ~ **satellite** спутник РЭР.

commercial *adj* коммерческий; ~ **channel** коммерческий канал (*связи*); ~ **code** коммерческий код; ~ **Communications Security Endorsement program** Программа аттестации коммерческих средств защиты связи (*США*); ~ **cryptography** коммерческая криптография; ~ **encryption** коммерческие криптосистемы; ~ **Computer Security Center** Центр безопасности коммерческих вычислительных систем (*Англии*); ~ **Enigma** коммерческая модификация немецкой шифровальной машины "Энигма".

common *adj* общий; ~ **carrier** (1) общественная [общедоступная] линия связи; (2) коммерческая [частная] сеть связи; (3) зарегистрированная частная компания-владелец сети связи; ~ **key** общий ключ (*одноключевой криптосистемы*); ~**-key cryptography** = single-key cryptography; ~ **multiple** общий делитель.

communicant *n* абонент (*сети связи*).

communicate *v* обмениваться сообщениями; ~ **privately** обмениваться секретными сообщениями.

communication *n* связь; линия / канал связи; сообщение; ~**s cryptography** криптография для защиты связи; ~**s intelligence** радиоэлектронная разведка; ~**s Intelligence Board** Совет по радиоэлектронной разведке; ~**s intelligence material** материалы радиоэлектронной разведки; ~**s interception** перехват в линиях связи; ~**s intercepts** перехваченные сообщения; ~ **network** сеть связи; ~**s satellite** спутник связи; ~**s Security** (1) Безопасность связи (*подразделения АНБ США, занимающиеся вопросами обеспечения безопасности связи*); (2) обеспечение безопасности связи; ~**s security device** устройство для обеспечения безопасности связи; ~**s Security Establishment** Служба безопасности связи (*канадское правительственное ведомство, занимающееся перехватом и дешифрованием сообщений*); ~ **theory of secrecy systems** "Теория связи в секретных системах" (*статья американского математика К. Шеннона, публикация которой в 1949 г. ознаменовала наступление эры научной криптологии с секретными ключами*).

communicator *n* = communicant.

commutative *adj* коммутативный; ~ **property of encrypting function** коммутативность шифрующей функции; ~ **ring** коммутативное кольцо.

compact *adj* компактный; ~ **knapsack** компактный вектор ранца.

compass *adj* компасный; ~ **cipher** компасный шифр (*разновидность шифра простой замены*); ~ **key** ключ к компасному шифру.

compatibility *n* совместимость.

compound *adj* составной; ~ **encryption** составное шифрование (*последовательно не менее чем двумя различными методами*).

complement *n* дополнение; ~ **key** ключ-дополнение (*получается путем выполнения операции логического отрицания над битами ключа, представленного в двоичном виде*).

complementary *adj* дополняющий, комплиментарный; ~ **key** дополняющий [комплиментарный] ключ (*открытый или секретный ключ в криптосистеме с открытым ключом*).

complete *adj* полный; ~ **protection against impersonation attack** полная защита от попытки имитации; ~ **set of residues modulo n** полное множество вычетов по модулю n; ~ **spread function** полнораспределенная функция (*значение функции существенно зависит от каждого ее аргумента*).

completely *adv* полностью; ~ **random source of messages** полностью случайный источник сообщений.

completeness *n* полнота; ~ **property** свойство полноты шифрования (*каждый выходной бит шифра является функцией всех входных битов*).

complex *adj* сложный, комбинированный; ~ **key** сложный [комбинированный] ключ.

complexity *n* сложность (*алгоритма или математической задачи*).

component *adj* составляющий, входящий в состав; ~ **cipher** составляющий шифр (*шифр, входящий в состав суперпозиции шифров*).

component *n* компонент, составная часть; ~**-by-**~ **addition** покомпонентное сложение (*попарное сложение соответствующих элементов двух последовательностей*); ~**-wise addition** = ~-by-~ addition.

composite *adj* составной (*о числе*).

composite *n* составное число.

composition *n* = product.

compression *n* сжатие (*данных*); ~ **function** = hash function.

compromise *n* компрометация; ~ **of a key** компрометация ключа.

compromise *v* компрометировать; ~ **a secret key** компрометировать секретный ключ.

compromised *adj* скомпрометированный; ~ **cryptosystem** скомпрометированная криптосистема (*потерявшая секретность в результате того, что ее секретные криптопеременные стали известны противнику*); ~ **key** скомпрометированный (*потерявший секретность — вскрытый противником или ставший ему известным в силу каких-либо других обстоятельств*) ключ.

COMPUSEC *abbr* КОМПЬЮБЕЗ (*см. Computer Security*).

computable *adj* вычислимый; ~ **function** вычислимая функция.

computational *adj* вычислительный; ~ **complexity** вычислительная сложность; ~ **complexity-based cryptography** вычислительная криптография [криптография на основе вычислительной сложности]; ~ **cryptography** = ~ complexity based cryptography.

computationally *adv* вычислительно; ~ **feasible** вычислительно осуществимый; ~ **hard problem** труднорешаемая задача (*задача, решение которой требует больших вычислительных затрат*); ~ **infeasible** вычислительно неосуществимый; ~ **secure scheme** вычислительно стойкая схема (*схема шифрования или аутентификации является вычислительно стойкой, если ее стойкость зависит от вычислительного алгоритма, сложность которого такова, что он не может быть реализован на практике, поскольку требует выполнения слишком большого числа операций*).

compute *v* вычислять; ~ **a digital signature on a certain message** вычислить цифровую подпись для некоторого сообщения; ~ **a key** вычислять ключ.

computer *adj* компьютерный; ~ **cryptography** (1) компьютерная криптография; (2) применение криптографии для защиты данных в компьютере и компьютерных сетях; ~ **key** ключ шифрования-расшифрования для компьютера.

computer *n* компьютер; **~-aided cryptanalysis** = ~ cryptanalysis; **~-assisted cryptanalysis** = ~ cryptanalysis; **~-backed cryptanalysis** = ~ cryptanalysis; **~-based message system** компьютерная система передачи сообщений; ~ **cryptanalysis** машинный криптоанализ [криптоанализ с использованием компьютера]; ~ **data encryption** шифрование данных в компьютере; ~ **eavesdropping** перехват электромагнитного излучения работающих компьютеров; **~-performing cryptanalysis** = ~ cryptanalysis; ~ **Security** Компьютерная безопасность (*подразделения АНБ США, занимающиеся вопросами обеспечения компьютерной безопасности*).

COMSAT *abbr* = communications satellite.

COMSEC *abbr* = Communications Security.

concatenated *adj* (1) составной; ~ **encryption** составное шифрование (*последовательно не менее чем двумя различными криптографическими методами*); (2) = chained.

concatenation *n* сцепление; ~ **of a message and an appended authenticator** сцепление сообщения и добавляемого (*к этому сообщению*) аутентификатора.

conceal *v* прятать, скрывать; ~ **the contents of a message** прятать содержание сообщения.

concealment *n* сокрытие, утаивание; ~ **message methodology** методология сокрытия сообщения.

conditional *adj* условный; ~ **distribution** условное распределение.

CONELRAD *abbr* КОНЭЛИЗ (*см. control of electromagnetic radiation*).

conference *n* конференц-связь; ~ **cryptosystem** криптосистема для конференц-связи; ~ **key** конференц-ключ (*общий для всех участников конференции*); ~ **key distribution system** система распределения конференц-ключей.

confidence *n* доверие; ~ **estimation** оценка по доверительному интервалу.

confidential *adj* конфиденциальный; ~ **means of communication** средства конфиденциальной связи; ~ **message** конфиденциальное сообщение.

confidentiality *n* конфиденциальность (*защищенность информации от несанкционированного доступа со стороны пользователей, не имеющих соответствующих полномочий*); ~ **provided by encryption** конфиденциальность, обеспечиваемая с помощью средств шифрования.

confusion *n* перемешивание (*использование шифрующих преобразований, которые усложняют восстановление взаимосвязи статистических свойств открытого и шифрованного текстов*).

congruence *n* конгруэнтность (*числа a и b связаны отношением конгруэнтности по модулю m, если a-b делится на m*).

congruent *adj* конгруэнтный; **a is ~ to b modulo m** число a конгруэнтно числу b (*a-b делится на m*).

congruential *adj* конгруэнтный (*генератор*); ~ **generator** конгруэнтный генератор [генератор конгруэнтных чисел] [конгруэнц-генератор].

consecutive *adj* идущий подряд; ~ **bits of linear feedback shift register sequence** подряд идущие биты последовательности, полученной с помощью регистра сдвига с линейной обратной связью.

consonant *n* буква, обозначающая согласный звук; ~**-vowel relationship** взаимосвязь между буквами, обозначающими согласные и гласные звуки.

constant *adj* постоянный; ~ **sequence** постоянная последовательность (*двоичная последовательность, состоящая из одних нулей или из одних единиц*).

contemporary *adj* современный; ~ **cryptology** современная криптология.

contents *n pl* содержание, содержимое; ~ **of a cryptogram** содержание криптограммы.

contingency *n* непредвиденное обстоятельство, случай; ~ **key** ключ для непредвиденных обстоятельств (*специальный ключ, предназначенный для использования при возникновении непредвиденных обстоятельств*).

continuos *adj* непрерывный; ~ **encryption** непрерывное шифрование.

control *n* контроль; ~ **of electromagnetic radiation** контроль за электромагнитным излучением.

controlled *adj* контролируемый; ~ **cryptographic item** контролируемое криптографическое изделие (*несекретное криптографическое оборудование, использование которого осуществляется под строгим контролем со стороны государства*); ~ **cryptographic item assembly** комплект оборудования, являющийся контролируемым криптографическим изделием; ~ **cryptographic item component** устройство, являющееся частью контролируемого криптографического изделия; ~ **cryptographic item equipment** оборудование, в состав которого входит контролируемое криптографическое изделие; ~ **key distribution** контролируемое распределение ключей.

controlling *adj* контролирующий; ~ **authority** орган контроля (*администрация криптосети*).

conventional *adj* традиционный, стандартный; ~ **cipher** традиционный шифр (*с секретным ключом*); ~ **cryptanalysis** традиционный криптоанализ (*без использования компьютеров*); ~ **cryptographic key** ключ традиционной (*одноключевой*) криптографической системы; ~ **cryptographic means** традиционные криптографические средства; ~ **cryptosystem** = symmetric cryptosystem; ~ **signature scheme** стандартная схема цифровой подписи.

conversational *adj* переговорный; ~ **key** переговорный ключ (*используемый для переговоров при согласовании общего ключа*).

cookie *n* печенье; ~ **factory** "фабрика печенья" (*шутливое название, придуманное для своего ведомства сотрудниками АНБ США*).

cooperative *adj* совместный; ~ **key generation** совместная генерация ключа (*абоненты сети связи генерируют и посылают друг другу случайные последовательности, которые позволяют им сгенерировать ключи для шифрования передаваемых по этой сети сообщений*); ~ **remote rekeying** = manual remote rekeying.

coprime *adj* взаимно простой.

correct *adj* правильный; ~ **decryption** правильное расшифрование; ~ **key** правильный ключ.

correlation *n* корреляция; ~ **attack** корреляционная атака (*криптоанализ на основе выявления статистической зависимости между элементами входной и выходной последовательности шифратора*); ~ **attack secure** защищенный от корреляционной атаки; ~**-immune** корреляционно-стойкий (*характеризующийся отсутствием статистической зависимости между элементами входной и выходной последовательности шифратора*); ~ **immunity** корреляционная стойкость (*степень статистической зависимости между элементами входной и выходной последовательности шифратора*).

corrupt *n* компрометировать; ~ **KDC** компрометировать ЦРК.

corrupted *adj* скомпрометированный; ~ **key** скомпрометированный (*потерявший секретность — вскрытый противником или ставший ему известным каким-либо другим способом*) ключ.

count *n* подсчет, маркировка; ~ **of character frequencies** = character frequency ~.

counter *n* счетчик; ~ **mode** счетный режим (*один из режимов работы синхронного поточного шифра*).

counterfeit *adj* поддельный; ~ **cipher** поддельный шифр.

counting *n* = count.

courier *n* курьер, посыльный; ~**-based key distribution** распределение ключей с помощью курьеров [посыльных]; ~**-based key management** управление ключами с помощью курьеров [посыльных]; ~**-delivered key** доставленный курьером [посыльным] ключ.

course *n* курс; ~ **in cryptanalysis** курс криптоанализа.

cover *n* защита; ~ **time** время защиты (*промежуток времени, в течение которого шифрсистема противостоит попыткам ее вскрытия*).

crack *v* вскрывать; ~ **a cipher** вскрывать шифр; ~ **a cipher machine** вскрывать шифратор.

crackable *adj* вскрываемый; ~ **cipher** вскрываемый шифр.

cracker *n* взломщик, "крэкер" (*человек, взламывающий средства контроля доступа к компьютерным системам как для незаконного обогащения, так и ради удовольствия лишний раз воспользоваться своими навыками*).

cracking *n* вскрытие; ~ **contest** состязания по вскрытию (*криптосистем*); ~ **of a cryptosystem** вскрытие криптосистемы; ~ **program** программа вскрытия.

CRC *abbr* ЦИК (*см. cyclical redundancy code*).

CRG *abbr* ГКГ (*см. Cryptological Readiness Group*).

CRL *abbr* САС (*см. certificate revocation list*).

crib *n* криб, подстрочник (*наиболее вероятный вариант открытого текста для некоторого отрезка шифрованного текста*); ~ **dragging** метод протягивания криба [подстрочника].

cribster *n* крибстер (*криптоаналитик, который вскрывает шифры исключительно с помощью крибов*).

criteria *n* критерий.

CRL *abbr* САС (*см. certificate revocation list*).

CRNG *abbr* КГСЧ (*см. cryptographic random number generator*).

CRT *abbr* КТО (*см. Chinese Remainder Theorem*).

cryppee *n жарг.* криптограф.

cryppie *n* = cryppee.

crypt *n* (1) *разг.* = cryptogram; (2) "Крипт" (*программа шифрования по DES-алгоритму в операционной системе UNIX*).

crypt- = crypto-.

cryptalgorithm *n* = cryptoalgorithm.

cryptanalyse *v* подвергать криптоанализу; проводить криптоанализ; ~ **an algorithm** подвергать алгоритм (*шифрования*) криптоанализу.

cryptanalysing *n* проведение криптоанализа.

cryptanalysis *n* криптоанализ (*анализ криптосистемы с целью определения засекреченных переменных и другой значимой информации об этой криптосистеме, включая открытый текст, шифруемый с ее помощью*); ~ **by frequency counts** криптоанализ на основе маркировок; ~ **results** результаты криптоанализа; ~ **Service** = E-Dienst; ~ **techniques** методы криптоанализа.

cryptanalyst *n* криптоаналитик; дешифровальщик; ~ **hours** время, требуемое криптоаналитику для вскрытия шифра.

crypt-analyst *n* = cryptanalyst.

cryptanalytic *adj* криптоаналитический; ~ **attack** криптоаналитическая атака; ~ **breakthrough** криптоаналитический прорыв; ~ **community** криптоаналитическое сообщество; ~ **defense** защита от криптоанализа; ~ **effort** усилия криптоаналитиков; ~ **information** информация, существенная для криптоанализа; ~ **literature** литература по криптоанализу; ~ **machine** (*вычислительная*) машина, используемая для криптоанализа; ~ **research** криптоаналитические исследования; ~ **results** результаты криптоанализа; ~ **success** криптоаналитический успех; ~ **techniques** криптоаналитические методы; ~ **theory** криптоаналитическая теория; ~ **use of high-speed computers** использование

высокоскоростных компьютеров для криптоанализа; ~ **work** криптоаналитическая работа.

cryptanalytically *adv* криптоаналитически; ~ **equivalent ciphers** криптоаналитически эквивалентные шифры (*если удается вскрыть один из криптоаналитически эквивалентных шифров, то вскрывается и другой*); ~ **secure key** стойкий к криптоанализу ключ.

cryptanalytics *n* методика криптоанализа, теория и техника вскрытия шифров.

cryptanalyze *v* = cryptanalyse.

cryptanalyzed *adj* подвергнутый криптоанализу; ~ **intercept** перехваченное сообщение, подвергнутое криптоанализу.

cryptarithm *n* криптоарифм (*арифметическая задача, в которой вместо цифр используются буквы, причем: (а) каждая цифра заменяется ровно на одну букву и встречается хотя бы раз; (б) самая старшая значащая цифра любого числа не может быть нулем; (в) если основание системы счисления отлично от 10, этот факт оговаривается особо*).

cryptic *adj* (1) записанный в зашифрованном виде [в форме криптограммы]; криптографически закрытый; (2) криптографический; криптологический.

cryptically-written *adj* записанный в зашифрованном виде [в виде криптограммы].

cryption *n* = encryption.

cryptkeyer *n* сотрудник службы безопасности, отвечающий за работу криптографических средств.

CRYPTO *n* гриф на документах, имеющих отношение к криптографической защите информации; категория секретности "КРИПТО".

Crypto *prop* Крипто (*название ежегодной международной конференции по криптологии*).

crypto- *в сложных и сложносоставных словах имеет значение* крипто-.

cryptoaccount *n* учет работы криптографических средств.

crypto-administration *n* администрация криптографической службы.

Crypto AG *prop* "Крипто АГ" (*одна из самых известных в мире фирм по производству коммерческих шифраторов*).

cryptoalgorithm *n* криптоалгоритм, алгоритм шифрования.

cryptoalarm *adj* криптосигнализация (*см. cryptographic alarm*).

cryptoanalysis *n* = cryptanalysis.

cryptoanarchist *n* криптоанархист; ~ **manifesto** манифест криптоанархистов.

cryptoanarchy *n* криптоанархизм (*бесконтрольное, нерегулируемое законом распространение криптотехнологий*).

cryptoancillary *adj* криптовспомогательный; ~ **equipment** вспомогательное криптографическое оборудование (*предназначено для выполнения функций, которые способствуют более эффективной и надежной работе криптографических устройств*).

cryptoanonymity *n* криптоанонимность.

cryptoAPI *abbr* криптоПИП (*см. Cryptographic Application Programming Interface*).

cryptoboard *n* криптоплата.

cryptobully *n* криптогромила, криптобандит.

Cryptobytes *prop* "Криптобайты" (*периодический информационный бюллетень, публикуемый американской компанией "RSA Laboratories"*).

cryptocard *n* криптокарта.

cryptocenter *n* криптоцентр; шифрцентр.

cryptochannel *n* криптоканал (*канал криптографически защищенной связи*).

cryptochip *n* крипточип, криптомикросхема.

cryptocommunications *n* криптографически защищенная связь.

cryptocommunity *n* криптосообщество.

cryptocompany *n* криптокомпания.

cryptocompatibility *n* криптосовместимость.

cryptocomplexity *n* криптографическая сложность [стойкость].

cryptoconference *n* криптоконференция.

crypto-control *n* криптоконтроль.

cryptocorrespondence *n* шифрпереписка.

cryptocriminal *n* криптопреступник.

cryptocustodian *n* ответственный за хранение ключей.

crypto-dedicated chip *n* = cryptochip.

cryptodeveloper *n* крипторазработчик; ~ **kit** набор средств крипторазработчика.

cryptodevice *n* криптографическое устройство, криптоприбор, шифратор.

crypto-engine *n* (1) криптосредство; (2) криптографический алгоритм.

cryproengineer *n* инженер-криптограф.

cryptoequipment *abbr* криптооборудование, шифраппартура (*см. cryptographic equipment*).

cryptofiction *abbr* = cryptographic fiction.

crypto-fill *adj* (*предназначенный*) для загрузки [ввода] ключей; ~ **device** устройство ввода ключей.

cryptogear *n* криптографическое устройство.

cryptogram *n* (1) криптограмма, тайнопись, шифрованное сообщение; ~ **bit** бит криптограммы [шифртекста]; ~ **residue class** класс вычетов криптограмм; ~ **space** пространство криптограмм; (2) "Криптограмма" (*журнал Американской криптологической ассоциации, выходящий раз в два месяца*).

Crypto-Gram *prop* "Крипто-Грамма" (*ежемесячный криптологический информационный бюллетень, распространяемый в электронном виде среди пользователей глобальной компьютерной сети "Интернет"*).

cryptograph *n* шифровальная машина.

cryptographed *adj* = enciphered.

cryptographer *n* криптограф.

cryptographic *adj* криптографический; ~ **aids** криптографические средства; ~ **alarm** криптографическая сигнализация (*специальное устройство, которое отслеживает сбои или изменения в работе криптографического оборудования и сигнализирует обслуживающему персоналу об их возникновении*); ~ криптографическое приложение; ~ **Application Programming Interface** криптографический прикладной программный интерфейс пользователя (*набор криптографических программных средств, разработанных американской корпорацией "Майкрософт" и предоставляемых программистам для разработки собственных приложений, взаимодействующих с этими средствами*); ~ **attack** криптографическая атака; ~ **authentication** криптографическая аутентификация (*с использованием криптографических средств и методов*); ~ **channel** криптографически защищенный канал; ~ **checksum** криптографическая контрольная сумма; ~ **checkvalue** криптографическая контрольная величина (*получается путем криптографического преобразования блока данных и предназначена для контроля целостности информации, передаваемой по каналам связи*); ~ **clerk** шифровальщик; ~ **commodities** криптографическое оборудование [криптографические средства]; ~ **community** криптографическое сообщество; ~ **compatibility** криптографическая совместимость; ~ **component** криптографический узел (*устройство для аппаратной реализации криптографического алгоритма*); ~ **course** курс криптографии; ~ **design vulnerabilities** слабые места криптографических систем; ~ **devices** криптографические устройства; ~ **encryption** криптографическое шифрование; ~ **equipment** криптографическая аппаратура; ~ **equipment room** помещение для криптографического оборудования (*охраняемая экранированная комната для размещения в ней криптографического оборудования*); ~ **expert** эксперт в области криптографии; ~ **facilities** криптографическое оборудование [криптографические средства]; ~ **fiction** криптографическая художественная литература (*литературное произведение, одна из главных сюжетных тем которого связана с криптографией*); ~ **hash function** криптографическая хэш-функция; ~ **information** криптографическая информация (*относится к криптографическим системам и средствам*); ~ **initialization** инициирование криптографических процессов (*напр., вводом клю-*

ча); ~ **integrity** криптографическая целостность [стойкость]; ~ **interception** перехват шифрованной информации; ~ **interface** криптоинтерфейс [интерфейс криптографического устройства]; ~ **jargon** криптографический жаргон; ~ **literature** литература по криптографии; ~ **logic** криптографическая логика (*аппаратная реализация криптографического алгоритма, дополненная криптосигнализацией и другими средствами обеспечения эффективной и надежной работы криптографического оборудования*); ~ **loop-hole** криптографическая слабость; ~ **materials** криптографические материалы; ~ **means** криптографические средства; ~ **mechanism** криптографический механизм; ~ **mistake** криптографическая ошибка; ~ **network** сеть с криптографической защитой; ~ **option** дополнительное криптографическое оборудование (*поставляемое вдобавок к основному*); ~ **parameter** криптографический параметр; ~ **prime** простое число, используемое в криптосистеме; ~ **primitive** криптографический примитив (*элементарное понятие из области криптографии — блочный шифр, алгоритм цифровой подписи и т. д.*); ~ **procedure** криптографическая процедура; ~ **protection** криптографическое закрытие информации [криптографическая защита]; ~ **protocol** криптографический протокол; ~ **randomization** криптографическая рандомизация (*функция, позволяющая случайным образом задавать состояние шифратора*); ~ **random number generator** криптографический генератор случайных чисел; ~ **research** криптографические исследования [исследования в области криптографии]; ~ **sensitivity** криптографическая уязвимость; ~ **separation** криптографическое разделение (*информации с использованием различных ключей шифрования*); ~ **service** криптографическая служба; ~ **service message** сообщение криптографической службы; ~ **Service Provider** средство криптографического сервиса (*программный модуль для выполнения криптографических операций*); ~ **Service Provider Developer Kit** Набор разработчика средств криптографического сервиса; ~ **strength** криптографическая стойкость; ~ **success** успех [достижение] в области криптографии; ~ **system** криптографическая система; ~ **transformation** криптографическое преобразование; ~ **unit** криптографический блок; ~ **variable** криптографическая переменная (*напр., ключ*).

cryptographically *adv* криптографически; ~ **compressed form of a message** криптографически сжатая форма сообщения; ~ **equivalent** эквивалентные по стойкости; ~**-protected communication** связь с криптографическим закрытием информации [шифрсвязь]; ~ **random** криптографически случайный (*псевдослучайная последовательность называется криптографически случайной, если ее следующий элемент вычислительно невозможно предсказать, даже зная алгоритм генерации этой последовательности и все ее предшествующие элементы*); ~ **secure** криптографически стойкий; ~ **secure random number generator** криптографически стойкий генератор случайных чисел; ~ **strong** = ~ secure; ~ **strong random number generator** = ~ secure random number generator; ~ **token** криптографический маркер; ~ **voting** криптографическое голосование (*разновидность криптографических протоколов для анонимного голосования*).

cryptographie *n* = cryptography.

cryptographing *n* разработка и применение средств криптографической защиты.

cryptography *n* криптография (*наука о принципах, средствах и методах преобразования информации для защиты ее от несанкционированного доступа и искажения*); ~ **Application Programming Interface** = Cryptographic Application Programming Interface; ~ **consultant** консультант по криптографии; ~ **research** = cryptographic research; ~ **textbook** учебник [пособие / руководство] по криптографии; ~ **war** криптографическая война (*вечный конфликт между криптографом и криптоаналитиком: первый пытается создать шифр, который может противостоять всем атакам на него, а второй любыми способами старается этот шифр вскрыть*).

cryptoguru *n* криптогуру.

cryptohuddle *n* криптосговор.

cryptoignition *n* инициирование криптографических операций; ~ **key** ключ инициирования криптографических операций.

cryptoinformation *n* (1) шифрованная информация; (2) информация, используемая при криптоанализе; (3) информация, имеющая отношение к криптосистеме.

cryptoinsecurity *n* нарушение криптостойкости (*вследствие ошибки оператора или сбоя аппаратуры*).

cryptointerface *n* криптоинтерфейс (*интерфейс криптографического устройства*).

cryptokey *n* криптоключ; ~ **entry** ввод ключа; ~ **generator** генератор криптоключей.

crypto-like *adj* криптоподобный; ~ **transformation of information** криптоподобное преобразование информации.

crypto-linguist *n* = cryptolinguist.

cryptolinguist *n* криптолингвист.

cryptolinguistics *n* криптолингвистика (*применение лингвистики в криптологии*).

cryptolock *n* криптозамок.

cryptologer *n* = cryptologist.

Cryptologia *prop* "Криптология" (*ежеквартальный криптологический журнал, выходящий в США*).

cryptologic *adj* = cryptological.

cryptological *adj* криптологический; ~ **agency** криптологическое агентство; ~ **circles** криптологические круги; ~ **matters** вопросы, имеющие отношение к криптологии; ~ **Readiness Group** Группа криптологической готовности; ~

service криптологическая служба; ~ **technician** специалист-криптолог; ~ **term** криптологический термин.

cryptologist *n* криптолог (*специалист по криптологии*).

cryptology *n* криптология (*наука о безопасности связи*).

cryptomachine *n* криптомашина; криптографическая [шифровальная] машина.

cryptomapping *n* криптоотображение.

cryptomaterial *n* документы или аппаратура, содержащие криптографическую информацию.

cryptomathematics *n* криптоматематика (*применение математики в криптологии*).

cryptomicroprocessor *n* (1) криптомикропроцессор (*работает по зашифрованной программе*); (2) микропроцессор криптографического устройства.

cryptomode *n* крипторежим (*режим работы с криптографической защитой*).

cryptomodule *n* криптомодуль, модуль с криптографическими средствами.

cryptomonopoly *n* криптомонополия.

cryptonet *n* криптосеть (*сеть криптографически защищенной связи*); ~ **control station** управляющая станция криптосети; ~ **key** ключ криптосети.

crypto-only mode *n* режим работы только с криптографической защитой.

cryptooperation *n* криптооперация; работа в режиме криптографической защиты.

cryptopackage *n* (1) пакет программ для криптографической защиты информации; (2) криптографическое устройство [криптографический прибор] в отдельном корпусе.

cryptopager *n* криптопейджер (*средство персонального конфиденциального радиовызова*).

cryptoparts *n pl* блоки, на которые разбиваются сообщения (*в соответствии с принятым в криптосистеме форматом*).

cryptopatent *n* криптопатент.

cryptoperiod *n* криптопериод; период действия [использования] ключа; период шифрования (*текста*) на одном ключе; ~ **diffusion** размывание криптопериода.

cryptopersonnel *n* персонал [личный] состав криптографической службы.

cryprophonic *adj* криптотелефонный, криптофонический (*относящийся к засекречиванию телефонной связи путем шифрования речи*); ~ **device** криптотелефонное [криптофоническое] устройство; ~ **module** криптотелефонный [криптофонический] модуль.

cryptophonie *n* криптотелефония (*телефонная связь с криптографической защитой от подслушивания*).

cryptopolicy *n* политика в области криптографии.

cryptoprinciple *n* принцип криптографического преобразования информации.

cryptoprocessor *n* (1) криптопроцессор (*работает по зашифрованной программе*); (2) процессор криптографического устройства.

cryptoproof *adj* криптостойкий.

cryptopunk *n* криптопанк (*человек, который придерживается мнения о том, что любая информация частного характера неприкосновенна и должна быть надежно защищена с помощью стойких криптографических алгоритмов*).

cryptor *n* устройство шифрования, шифратор.

cryptoradio *n* крипторадио [оснащенная шифратором радиостанция].

cryptoresearcher *n* криптоисследователь.

cryptosaur *n* "криптозавр" (*маститый специалист-криптолог, достижения которого стали достоянием истории*).

cryptosecure *adj* криптостойкий.

cryptosecurity *n* (1) обеспечение безопасности криптографическими средствами; криптографическая защита; ~ **device** прибор для криптографической защиты; (2) криптостойкость.

cryptoserver *n* криптосервер.

cryptoservice *n* криптографическая служба.

cryptosoft *n* криптософт (*криптографическое программное обеспечение*).

cryptospeak *n* криптотерминология, криптожаргон.

cryptostrength *n* криптостойкость.

cryptosync *abbr* криптосинх (*см. cryptosynchronization*).

cryptosynchronization *n* криптосинхронизация (*синхронизация криптографических устройств / процессов*).

cryptosystem *abbr* криптосистема, шифрсистема (*см. cryptographic system*); ~ **assessment** оценка криптосистемы (*процесс определения применимости криптосистемы на практике*); ~ **designer** разработчик криптосистемы; ~ **evaluation** анализ криптосистемы (*поиск уязвимостей в криптосистеме*); ~ **review** проверка криптосистемы (*рассмотрение криптосистемы органом контроля на предмет полноты реализуемых ею функций и определения области ее применения*); ~ **survey** инспектирование криптосистемы (*пользователи криптосистемы излагают свое мнение относительно того, насколько полно она удовлетворяет их потребностям, и делятся с разработчиками этой криптосистемы своими соображениями о путях ее усовершенствования*); ~ **that would take millions of years to break** криптосистема, для вскрытия которой потребуются миллионы лет; ~ **with 56 bit keys** криптосистема с 56-битным ключом.

crypto-tape *n* криптолента (*бумажная или магнитная лента с шифртекстом или ключом*).

cryptotechnology *n* криптотехнология (*технология проектирования, изготовления и эксплуатации криптографических устройств*).

cryptotelephone *n* криптотелефон (*телефонный аппарат с шифратором или скремблером*).

cryptoterm *n* криптотермин.

cryptotext *n* криптотекст, шифртекст.

cryptothriller *n* криптотриллер (*роман с захватывающим сюжетом, одна из главных тем которого связана с криптологией*).

cryptotoken *abbr* криптомаркер (*см. cryptographic token*).

crypto-to-plain *n* преобразование шифрованного текста в открытый.

cryptotransaction *n* криптооперация.

cryptounit *abbr* криптоблок (*см. cryptographic unit*).

cryptouser *n* криптопользователь.

cryptovariable *abbr* криптопеременная (*см. cryptographic variable*).

cryptovector *n* криптовектор (*вектор ключа или вектор начального заполнения ключевого генератора*).

CS *abbr* КС (*см. cryptosystem*).

CSD *abbr* УЗС (*см. communication security device*).

CSE *abbr* СБС (*см. Communications Security Establishment*).

CSP *abbr* СКС (*см. cryptographic service provider*).

CSPDK *abbr* НРСКС (*см. Cryptographic Service Provider Developer Kit*).

CSRNG *abbr* КСГСЧ (*см. cryptographically secure random number generator*)..

CSS *abbr* ЦСБ (*см. Central Security Service*).

CSSIS *abbr* ЦСБИС (*см. Central Service for Security and Information Systems*).

CTAK *abbr* = cipher text autokey.

CTT *abbr* СКТ (*см. cryptological technician*).

CU *abbr* КБ (*см. cryptographic unit*).

cubic *adj* кубический; ~ **congruential generator** кубический конгруэнтный генератор.

current *adj* действующий, текущий; ~ **key** действующий (*в течение определенного периода времени*) ключ.

custodian *n* хранитель (*лицо, владеющее информацией или отвечающее за ее сохранность*); ~ **of a public directory** хранитель открытого справочника (*ключей или криптоалгоритмов*).

custom *adj* изготовленный, сделанный на заказ; ~ **key** заказной ключ [ключ заказчика (*шифратора*)].

customer *n* заказчик; ~**-specific key** ключ со структурой, определяемой пользователем [заказчиком].

customized *adj* соответствующий требованиям заказчика; ~ **cryptography** криптография, соответствующая требованиям заказчика.

CV *abbr* КП (*см. cryptographic variable*).

cycle *n* цикл (*повторяющийся отрезок последовательности наименьшей длины*); ~ **of length p** цикл длины p.

cyclic *adj* = cyclical.

cyclical *adj* циклический; ~ **redundancy code** циклический избыточный код (*функция хэширования, используемая для контроля за появлением ошибок при передаче и преобразовании данных*); ~ **shift** циклический сдвиг.

cyclicity *n* цикличность; периодичность.

cyclometre *n* циклометр (*специальное электро-механическое устройство, изобретенное польскими криптоаналитиками в 1934 г. для автоматизации процесса вскрытия немецкой шифровальной машины "Энигма"*).

cypher *n* = cipher *n*.

cypher *v* = cipher *v*.

cyphony *n* = ciphony.

D

D *abbr* (1) Д (*см. decipher*); (2) Д (*см. deciphering*); (3) Д (*см. diplomatic*).

DAC *abbr* КАД (*см. data authentication code*).

daily *adj* ежедневный; ~ **key** ежедневный ключ [ключ текущего дня]; ~ **keying element** ежедневно сменяемый элемент [сменяемая часть] ключа.

Dancing Men *n* "Пляшущие человечки" (*рассказ английского писателя А. К. Дойля с описанием оригинального криптоаналитического исследования шифрованного текста*).

dark-side *adj* темный, порочный; ~ **hacker** = cracker.

data *n* данные; ~ **authentication** аутентификация данных; ~ **authentication code** код аутентификации данных; ~ **ciphering operation** операция шифрования данных; ~ **ciphering processor** процессор шифрования данных; ~ **ciphering unit** устройство шифрования данных; ~ **communication equipment** оконечное оборудование канала связи; ~ **compression** сжатие данных; ~ **cryptography** криптография для защиты данных; ~**-dependent encryption** зависящее от вида данных шифрование; ~**-dependent key** зависящий от (*шифруемых*) данных ключ; ~ **element** элемент данных; ~ **encipherment equipment** = ~ encryption equip-

ment; ~ **enciphering key** ключ шифрования данных; ~ **encrypting key** = ~ enciphering key; ~ **Encryption Algorithm** Алгоритм шифрования данных (*стандарт шифрования данных США, введенный в действие в 1981 г. и используемый в американской промышленности*); ~ **encryption device** шифратор данных; ~ **encryption equipment** оборудование шифрования данных; ~ **encryption key** = ~ enciphering key; ~ **encryption processor** = ~ ciphering processór; ~ **Encryption Standard** Стандарт шифрования данных (*США*); ~ **exchange key** ключ для обмена данными; ~ **Link Enciphering Standard** Стандарт шифрования в линиях связи (*введенный в действие в 1981 г. стандарт на шифрование, используемый в промышленности США*); ~ **link encryptor** шифратор канала (*передачи*) данных; ~ **origin authentication** аутентификация источника данных; ~ **scrambling device** скремблер данных; ~ **security device** устройство защиты данных.

DC *abbr* СД (*см. data compression*).

DCE *abbr* ОКС (*см. data communication equipment*).

DCECP *abbr* ЦРВУБС (*см. Development Center for Embedded COMSEC Products*).

DCP *abbr* (1) ПШД (*см. data ciphering processor*); (2) ПОК (*см. dining cryptographers protocol*).

DCPH *abbr* ДШФР (*см. decipher*).

DDN *abbr* ОСД (*см. Defense Data Network*).

DEA *abbr* (1) АШД (*см. Data Encryption Algorithm*); (2) РАШ (*см. Diamond Encryption Algorithm*).

DEC *abbr* РАС (*кнопка / клавиша "расшифровать" на шифраторе*).

deceit *n* обман, мошенничество.

decentralized *adj* децентрализованный; ~ **key distribution** децентрализованное распределение ключей; ~ **key management** децентрализованное управление ключами.

decimated *adj* прореженная; ~ **sequence** прореженная последовательность (*получена в результате отбора элементов другой, исходной последовательности*).

decipher *v* (1) расшифровывать; ~ **key** ключ расшифрования; ~ **on the fly** расшифровывать на лету (*в темпе ввода шифртекста*); (2) вскрывать; дешифровывать; ~ **a code** вскрывать код.

decipherable *adj* поддающийся вскрытию [дешифровке]; ~ **in polynomial time** поддающийся вскрытию [дешифровке] с полиномиальной сложностью.

deciphered *adj* (1) расшифрованный; ~ **message** расшифрованное сообщение; (2) дешифрованный; ~ **message** дешифрованное сообщение.

decipherer *n* (1) устройство расшифрования; (2) дешифратор.

deciphering *n* (1) расшифровывание; ~ **alphabet** алфавит расшифрования; ~ **equation** уравнение расшифрования; ~ **function** функция расшифрования;

~ **key** ключ расшифрования; ~ **process** процесс расшифрования; ~ **rule** правило расшифрования; (2) дешифрование; ~ **process** процесс дешифрования.

decipherment *n* = deciphering.

DECK *abbr* PACK (*см. decipher key*).

decodability *n* декодируемость.

decode *n* декодированное сообщение.

decode *v* декодировать, раскодировать, дешифровывать.

decodement *n* (1) декодирование; (2) декодированный текст.

decoder *n* декодирующее устройство, декодер, дешифратор.

decoding *n* декодирующий; ~ **algorithm** декодирующий алгоритм; ~ **efforts** = decryption efforts; ~ **key** ключ декодирования.

decrypt *n* (1) расшифровка (*расшифрованный текст*); (2) дешифровка (*дешифрованный текст*).

decrypt *v* (1) расшифровывать; ~**-only key** ключ только для расшифрования; (2) дешифровывать.

de-crypt *v* = decrypt.

decrypted *adj* (1) расшифрованный; ~ **message** расшифрованное сообщение; (2) дешифрованный; ~ **message** дешифрованное сообщение.

decrypter *n* = decryptor.

decrypting *adj* дешифровальный; ~ **service** дешифровальная служба; ~ **unit** дешифровальное подразделение.

decryption *n* (1) расшифровывание; ~ **key** ключ расшифрования; ~ **round structure** структура цикла [раунда] расшифрования; (2) дешифрование; ~ **capabilties** дешифровальные возможности; ~ **device** дешифратор; ~ **efforts** усилия дешифровальщиков; ~ **process** процесс дешифрования; (3) открытый текст; ~ **of ciphertext** открытый текст шифровки.

decryptment *n* = decryption.

decryptor *n* (1) получатель шифрованного сообщения; (2) дешифровальщик; (3) устройство для дешифрования сообщений.

decypher *v* = decipher.

DEE *abbr* ОШД (*см. data encryption equipment*).

de-encryption *n* (1) расшифрование; (2) дешифрование.

de facto *adj лат.* фактический; ~ **standard for the encryption of electronic mail** фактический стандарт шифрования электронной почты.

defense *n* (1) оборона; ~ **Data Network** Оборонная сеть (*передачи*) данных; ~ **Intelligence Agency** Разведывательное управление министерства обороны (*США*); ~ **Message System** Система (*передачи*) оборонных сообщений (*США*);

~ **Signals Directorate** Управление безопасности связи (*австралийское правительственное ведомство, занимающееся перехватом и дешифрованием сообщений*); (2) защита.

defined-plaintext *adj* = chosen-plaintext.

defraud *v* вводить в заблуждение; ~ **a decryptor** вводить в заблуждение получателя шифрованного сообщения.

degenerate *adj* вырожденный; ~ **function** вырожденная функция.

degree *n* степень; ~ **of a polynomial** степень полинома; ~ **of security** степень защищенности (*криптосистемы*);.

DEK *abbr* (1) КШД (*см. data enciphering key*); (2) КОД (*см. data exchange key*).

delay *n* задержка; ~ **operator** оператор задержки.

delivery *n* доставка, транспортировка.

dense *adj* плотный, неразреженный; ~ **knapsack** неразреженный вектор ранца.

density *n* плотность; ~ **of knapsack** плотность вектора ранца.

deny *v* отказать; ~ **access** отказать в доступе.

DEP *abbr* ПШД (*см. data encryption processor*).

department *n* (1) управление; **16th ~ of KGB** 16-е управление КГБ (*подразделение КГБ, занимавшееся перехватом и дешифрованием сообщений*); (2) министерство; ~ **of Defense** министерство обороны (*США*); (3) отделение; **"~ 62"** = Federal Information Security Agency; **"~ 8200"** "Отделение 8200" (*подразделение израильской разведывательной организации "Моссад", занимающееся перехватом и дешифрованием сообщений*).

dependence *n* зависимость.

depth *n* мощность.

derive *v* устанавливать; получать; ~ **a key** устанавливать [получать] ключ.

derived *adj* производный; ~ **key** производный ключ (*полученный преобразованием другого ключа*).

DES *abbr* ~-алгоритм [~] (*см. Data Encryption Standard*); **3~** = triple DES; **~-based authentication** аутентификация на основе ~-алгоритма; ~ **chip** микросхема, реализующая ~-алгоритм; ~ **Cracker** "DES-крэкер" (*специализированный вычислитель, предназначенный для взлома ~-алгоритма*); **~-encrypted message** сообщение, зашифрованное по ~-алгоритму; ~ **implementation** реализация ~-алгоритма; **~-key** ~-ключ (*ключ для ~-алгоритма*); **~-like** подобный ~-алгоритму; ~ **protection** криптографическая защита с помощью ~-алгоритма; ~ **translation** перешифрование по ~-алгоритму (*расшифрование текста, зашифрованного с помощью какой-то криптосистемы, и повторное его шифрование посредством ~-алгоритма*).

descramble *v* дескремблировать.

descrambled *adj* дескремблированный.

descrambler *n* дескремблер.

descrambling *n* дескремблирование; ~ **key** ключ дескремблирования.

desensitized *adj* децентрализованный; ~ **authentication** децентрализованная аутентификация.

design *v* разрабатывать; ~ **ciphers** разрабатывать шифры.

designated *adj* назначенный; ~ **confirmer signature** цифровая подпись с назначенным конфирмантом.

designation *n* обозначение.

designer *n* разработчик; ~ **of a cryptosystem** разработчик криптосистемы.

destination *n* получатель (*сообщения*); ~ **cryptor** шифратор получателя (*сообщения*); ~ **key** ключ получателя (*сообщения*).

destinee *n* = destination.

de-tapper *n* прибор для обнаружения несанкционированных подключений к линии связи.

detectable *adj* поддающийся обнаружению; ~ **wiretap** поддающийся обнаружению перехват с подключением к проводной линии связи.

detector *n* критерий, детектор.

determination *n* определение; ~ **of the cipher system used** определение используемой шифрсистемы.

determine *v* определять; ~ **a key** определять ключ; ~ **the period of a sequence** определять период последовательности.

deterministic *adj* детерминированный; ~ **encryption** детерминированное шифрование (*каждому открытому тексту сообщения ставится в соответствие ровно один шифрованный текст*); ~ **key** детерминированный ключ; ~ **key generation** детерминированная генерация ключей.

development *n* разработка; ~ **Center for Embedded COMSEC Products** Центр разработки встраиваемых устройств безопасности связи (*США*).

device *n* устройство, прибор; ~ **key** ключ (*криптографического*) устройства [прибора].

devise *v* разрабатывать; ~ **a cryptosystem** разрабатывать криптосистему.

DFT *abbr* ДПФ (*см. discrete Fourier transform*).

D-H *abbr* Д-Х (*см. Diffie-Hellman*).

DIA *abbr* РУМО (*см. Defense Intelligence Agency*).

Diamond Encryption Algorithm *n* Ромбовый алгоритм шифрования.

dictionary *n* (1) словарь; ~ **attack** "словарная" атака (*вскрытие ключа путем перебора словарного справочника, составленного из наиболее часто используе-*

мых на практике ключей — имен, фамилий, инициалов, номеров телефонов, дат рождения и т. д.); ~ **computer** "словарный" компьютер (*предназначен для отбора сообщений, в которых присутствуют заданные ключевые слова и фразы*); ~ **program** "словарная" программа (*предназначена для отбора сообщений, в которых присутствуют заданные ключевые слова и фразы*); ~ **secure** защищенный от "словарной" атаки; (2) = ~ computer; (3) = ~ program.

difference *n* различие; ~**s in plaintext pairs** попарные различия в открытых текстах.

differential *adj* дифференциальный; ~ **attack** дифференциальная атака; ~ **cryptanalysis** дифференциальный криптоанализ; ~**-linear attack** линейно-дифференциальная атака; ~**-resistant function** дифференциально-стойкая функция (*стойкая против дифференциального криптоанализа*); ~ **techniques** методы дифференциального криптоанализа.

difficulty *n* сложность; ~ **of factoring** сложность разложения на множители.

Diffie *prop* Диффи У. (*американский криптолог, один из родоначальников криптографии с открытым ключом*); ~**-Hellman** = ~-Hellman key exchange algorithm; ~**-Hellman assumption** предположение Диффи-Хеллмана (*в 1976 г. У. Диффи и другой американский криптолог М. Хеллман высказали предположение о том, что невозможно вычислить g^{ab}, зная только g^a и g^b*); ~**-Hellman exponent** показатель экспоненты в алгоритме обмена ключами Диффи-Хеллмана; ~**-Hellman key exchange algorithm** алгоритм обмена ключами Диффи-Хеллмана; ~**-Hellman one-way function** односторонняя функция Диффи-Хеллмана.

diffusion *n* (1) рассеивание, диффузия (*распространение влияния одного знака открытого текста на множество знаков шифртекста*); (2) размывание (*преобразование*) открытого текста для изменения его статистических свойств.

digest *n* (1) сжатие; (2) резюме; ~ **of a message** резюме сообщения.

digital *adj* цифровой; ~ **cash** цифровая наличность; ~ **encryption algorithm** алгоритм цифрового шифрования; ~ **key** цифровой ключ; ~ **signature** цифровая подпись; ~ **Signature Algorithm** Алгоритм цифровой подписи; ~ **Signature Standard** Стандарт цифровой подписи (*США*); ~ **timestamp** цифровая отметка времени.

digraph *n* = bigram.

digraphic *adj* биграммный (*двухбуквенный*); ~ **characteristics** биграммные характеристики; ~ **coincidence test** тест на совпадение биграмм; ~ **frequency counts** подсчет встречаемости биграмм (*в тексте*), биграммная маркировка (*текста*); ~ **Phi test** биграммный ϕ-тест; ~ **substitution** биграммная замена (*с одновременной обработкой сразу двух знаков*).

dimension *n* размерность (*векторного пространства*).

dining *adj* обедающий; ~ **cryptographers protocol** протокол обедающих криптографов (*криптографический протокол, предназначенный для организации об-*

мена сообщениями таким образом, что их авторство остается неизвестным для участников этого протокола, если между ними отсутствует предварительный сговор).

dinome *n* пара цифр, две цифры.

dinomic *adj* двухцифровой; ~ **cipher system** двухцифровая шифрсистема (*при зашифровании каждый символ открытого текста заменяется на две цифры*); ~ **cipher system with variants** вариантная двухцифровая шифрсистема (*при зашифровании каждый символ открытого текста может заменяться на различные пары цифр*).

dinomic *n* двухцифровая шифрсистема.

diplomatic *adj* дипломатический; ~ **cipher** дипломатический шифр.

direct *adj* прямой, непосредственный; ~ **standard cipher alphabet** прямой стандартный шифралфавит; ~ **eavesdropping** прямой перехват (*путем непосредственного подключения к линии связи*); ~ **encryption** непосредственное шифрование (*аппаратными средствами*); ~ **search** = brute-force attack; ~ **symmetry method** = ~ symmetry technique; ~ **symmetry technique** метод прямой симметрии (*применяется для вскрытия шифров периодической многоалфавитной замены*).

directed *adj* направленный; ~ **key search** направленный поиск ключа; ~ **random search algorithm** алгоритм направленного случайного поиска.

directorate *n* = department.

director *n* директор; ~ **of NSA** директор НАБ.

directory *n* справочник.

DIRNSA *abbr* ДИРНАБ (*см. director of NSA*).

disavow *n* отказываться; ~ **communications** отказываться от сообщений.

discard *v* переставать пользоваться, прекращать использовать; ~ **a key** переставать пользоваться ключом [прекращать использовать ключ].

disclosure *n* раскрытие, разглашение (*секретных сведений*); ~ **of sensitive information** раскрытие [разглашение] секретной информации.

discover *v* раскрывать; ~ **a key** раскрывать ключ.

discoverable *adj* раскрываемый; ~ **key** раскрываемый ключ.

discrete *adj* дискретный; ~ **exponential function** функция дискретного возведения в степень; ~ **exponential cryptosystem** криптосистема на основе дискретного возведения в степень; ~ **Fourier transform** дискретное преобразование Фурье; ~ **logarithm** дискретный логарифм; ~ **logarithm problem** задача вычисления дискретного логарифма.

disjunction *n* дизъюнкция.

disjunctive *adj* дизъюнктивный; ~ **normal form** дизъюнктивная нормальная форма.

dispatch *n* донесение, депеша; ~ **in cipher** шифрованное донесение.

dispense *v* распределять, распространять; ~ **keys** распределять ключи.

dispersion *n* дисперсия.

displace *v* передвигать, сдвигать; ~ **a cryptogram by two positions to the left** передвинуть [сдвинуть] криптограмму на две позиции влево.

dispose *v* уничтожать; ~ **of keys** уничтожать ключи.

dissemination *n* распространение (*рассылка радиоразведывательной информации заинтересованным лицам и ведомствам*).

distorted *adj* искаженный; ~ **message** искаженное сообщение.

distribute *v* распределять, распространять, доставлять; ~ **keys** распределять ключи.

distributed *adj* распределенный; ~ **key generation** распределенная генерация ключей; ~ **key infrastructure** распределенная инфраструктура ключей; ~ **key management** распределенное управление ключами; ~ **system security architecture** архитектура безопасности распределенных систем (*разработанная в 1986 г. криптосистема американской фирмы "DEC", использующая две схемы шифрования: (а) симметричную для обеспечения конфиденциальности и целостности данных; (б) асимметричную для аутентификации и распределения ключей*).

distribution *n* (1) распределение, распространение, доставка; ~ **key** ключ для распределения других ключей; ~ **of keys** распределение ключей; (2) (*вероятностное*) распределение.

divide-and-conquer *v* разделяй и властвуй; ~ **attack** криптоанализ методом "разделяй и властвуй".

DL *abbr* ДЛ (*см. discrete logarithm*).

DLE *abbr* ШКД (*см. data link encryptor*).

DLP *abbr* ЗВДЛ (*см. discrete logarithm problem*).

DMS *abbr* СОС (*см. Defense Message System*).

DNF *abbr* ДНФ (*см. disjunctive normal form*).

do *v* делать, заниматься, выполнять; ~ **cryptanalysis** заниматься криптоанализом.

doctored *adj* фальсифицированный, поддельный; ~ **message** фальсифицированное [поддельное] сообщение.

DoD *abbr* МО (*см. Department of Defense*); ~ **5200.28-STD** стандарт (*США*) для оценки уровня защищенности вычислительных систем (*установлен директивой МО США № 5200.28-STD и определяет 4 иерархических класса защищенности вычислительных систем, разрабатываемых и используемых в интересах американского правительства*); ~ **Computer Security Center** Центр защиты компьютеров МО (*США*).

dot *adj* = scalar.

dot *n* точка; ~ **cipher** точечный шифр (*в виде точек на сетке определенного формата*).

double *adj* двойной; ~ **encryption** двойное шифрование; ~ **key** двойной ключ; ~**-length key** ключ двойной длины (*по сравнению с принятой*); ~ **substitution cipher** шифр двойной замены; ~ **transposition cipher** шифр двойной перестановки.

double- *в сложных и сложносоставных словах имеет значение* дважды-, двух-.

double-encrypted *adj* дважды зашифрованный; ~ **data** дважды зашифрованные данные.

double-iterated *adj* дважды итеративный; ~ **knapsack** дважды итеративный вектор ранца.

double-key *adj* двухключевой; ~ **cryptography** двухключевая криптография (*криптография с открытым ключом*); ~ **encryption** шифрование двойным ключом.

downline *adj* отправленный по линии связи из центра; ~ **key load** загрузка [ввод] ключа по линии связи из центра распределения ключей.

download *n* загрузка [ввод] по линии связи (*напр., ключей*).

download *v* загружать [вводить] по линии связи; ~ **a key** загружать [вводить] ключ по линии связи.

downloaded *adj* загружаемый [вводимый] по каналу связи; ~ **key** ключ, загружаемый [вводимый] по линии связи.

Doyle *prop* Дойль А. К. (*английский писатель, автор рассказа "Пляшущие человечки", в котором описано оригинальное криптоаналитическое исследование шифрованного текста*).

Dreyfus *prop* Дрейфус А. (*французский офицер, которому в 1894 г. было предъявлено обвинение в шпионаже в пользу Германии, при этом одной из основных улик послужила частично дешифрованная телеграмма итальянского военного атташе во Франции*); ~ **affair** дело Дрейфуса.

DSA *abbr* АЦП (*см. Digital Signature Algorithm*).

DSCR *abbr* ДСКР (*см. descrambler*).

DSD *abbr* УЗД (*см. data security device*).

DSG *abbr* ЦПД (*см. digital signature*).

DSS *abbr* СЦП (*см. Digital Signature Standard*).

DSSA *abbr* АБРС (*см. distributed system security architecture*).

dual *adj* (1) двойной; ~ **key system** система с двойным ключом (*позволяет одновременно порождать ключи двух типов: (а) для ключевого обмена, (б) для цифровой подписи*); (2) обратный; ~ **shift register** обратный регистр сдвига.

dummy *adj* пустой; ~ **message** фиктивное [пустое] сообщение; ~ **padding** пустое холостое заполнение (*дополнение сообщения ничего не значащей информацией*).

Dunlop *prop* Данлоп Дж. (*американский сержант, служивший курьером в АНБ в 50-е–60-е годы и продававший секретную информацию об американских шифрах советской разведке*).

duplet *n* дуплет (*два одинаковых, идущих подряд знака текста*).

dynamically *adv* динамически; ~ **changing key** динамически [непрерывно] изменяемый (*во времени*) ключ.

E

E *abbr* (1) Ш (*см. encipher*); (2) Ш (*см. enciphering*).

E3 *abbr* = end-to-end encryption cryptosystem.

E^3 *abbr* = E3.

EA *abbr* АШ (*см. encryption algorithm*).

easy *adj* простой, тривиальный; ~ **knapsack** тривиальный ранцевый вектор; ~**-to-break cipher** легко вскрываемый шифр.

eavesdrop *v* подслушивать, перехватывать; ~ **on legitimate messages** перехватывать истинные сообщения.

eavesdropper *n* пассивный нарушитель, перехватчик.

eavesdropping *n* подслушивание; перехват; несанкционированное извлечение информации (*из канала связи*); ~ **by emanation** перехват посредством приема побочных электромагнитных излучений (*несущих информацию*); ~ **by wiretap** перехват с подключением к проводной линии связи [перехват по отводному каналу]; ~ **network** сеть перехвата; ~ **on telephone conversations** подслушивать телефонные разговоры.

EBDA *abbr* ШВОВ (*см. encipher below decipher above*).

E-box *abbr* Р-блок (*см. expansion box*).

ECB *abbr* ЭКК (*см. electronic code book*).

Echelon *prop* "Эшелон" (*комплекс технических средств, предназначенный для перехвата телефонных разговоров, электронной почты и телекоммуникационных сообщений с отбором по заданным ключевым словам и фразам*).

ECPH *abbr* ЗШФР (*см. encipher*).

ECPL *abbr* ССКА (*см. Endorsed Cryptographic Products List*).

e-d *abbr* ш-р (*см. encryption-decryption*).

E-Dienst *abbr* (*от нем. Entzifferungs-Dienst*) "Криптоаналитическая служба" (*военно-морская дешифровальная спецслужба гитлеровской Германии*).

EEC *abbr* НКО (*см. external error control*).

EEPROM *abbr* ЭССПЗУ (*см. electrically erasable programmable read-only memory*).

EES *abbr* СШД (*см. Escrowed Encryption Standard*).

EFD *abbr* ЭУЗ [ЭУВ] (*см. electronic fill device*).

effective *adj* (1) эффективный; ~ **size of a secret key** эффективный размер [длина] секретного ключа; (2) = secure.

effectively *adv* практически; ~ **unbreakable cryptosystem** практически невскрываемая криптосистема (*вследствие чрезмерных затрат времени и средств на ее криптоанализ*).

efficiency *n* эффективность; ~ **of encryption** эффективность шифрования.

effort *n* объем работ; ~ **involved in an attack** вычислительная сложность алгоритма вскрытия (*криптосистемы*).

EFT *abbr* ЭПФ (*см. Electronic Funds Transfer*).

EK *abbr* КШ (*см. encryption key*).

EKMS *abbr* СУЭК (*см. electronic key management system*).

electrically *adv* электрически; ~ **erasable programmable read-only memory** электрически стираемое программируемое постоянное запоминающее устройство.

electromagnetic *adj* электромагнитный; ~ **eavesdropping** перехват посредством приема электромагнитных излучений (*несущих информацию*).

electronic *adj* электронный; ~ **cryptography** криптография с использованием электронной техники; ~ **code book** электронная кодовая книга (*один из режимов использования DES-алгоритма*); ~ **emission** электронное излучение; ~ **espionage** электронный шпионаж; ~ **fill device** электронное устройство загрузки [ввода] (*ключей*); ~ **Funds Transfer** Электронная передача фондов; ~ **intelligence** электронная разведка; ~ **interception** перехват с помощью электронных средств; ~ **key distribution** электронное распределение ключей; ~ **keying** электронный ввод ключей; ~ **key management system** система управления электронными ключами (*предназначена для автоматизации хранения, генерации, распределения, использования и уничтожения электронных ключей*); ~ **mail** электронная почта; ~ **mail message** сообщение электронной почты; ~ **money** электронные деньги; ~ **reconnaissance** электронная разведка; ~ **signature** = digital signature; ~ **surveilance** электронная слежка; ~ **transmission** электронная передача; ~ **warfare** электронные средства для ведения боевых действий.

electronically *adv* электронно; ~ **generated key** ключ, генерируемый электронным устройством; ~ **transferred key** ключ, доставляемый [передаваемый] электронными средствами.

element *n* элемент.

elementary *adj* (1) первичный; ~ **key** первичный ключ (*для выполнения предварительных преобразований*); (2) элементарный; ~ **cryptanalysis** элементарный криптоанализ; ~ **cryptography** элементарная криптография.

Elgar *prop* Элгар Э. (*английский композитор, автор музыкального произведения под названием "Вариации на тему Энигмы", под влиянием которого немецкий инженер А. Шербиус окрестил изобретенную им шифровальную машину "Энигмой"*).

ELINT *abbr* ЭР (*см. electronic intelligence*).

elliptic *adj* эллиптический; ~ **curve cryptosystem** эллиптическая криптосистема (*разновидность криптосистемы с открытым ключом, принципы функционирования которой позаимствованы из теории эллиптических кривых над конечными полями*).

EM *abbr* ЭП (*см. electronic mail*).

E-mail *abbr* Э-почта (*см. electronic mail*).

emanation *n* излучение (*работающей радиоэлектронной аппаратуры*); ~ **integrity** непроницаемость для излучения.

embeddable *adj* встраиваемый.

embedded *adj* встроенный; ~ **cryptography** встроенные криптографические средства; ~ **cryptographic system** встроенная криптографическая система (*является составной частью другой системы, основное назначение которой не связано с криптографией*).

emergency *n* чрезвычайные обстоятельства; ~ **key** ключ, используемый в чрезвычайных обстоятельствах; ~ **key erase button** кнопка [клавиша] экстренного стирания ключей в чрезвычайных обстоятельствах.

emission *n* излучение; ~ **security** безопасность по электромагнитному излучению (*меры предосторожности, предпринимаемые для того, чтобы посторонние лица не смогли получить доступ к открытым текстам и ключевой информации путем анализа электромагнитных излучений от криптографического оборудования*).

EMP *abbr* ПШС (*см. Encrypted Messaging Protocol*).

EMSEC *abbr* БЕИЗ (*см. emission security*).

enable *v* разрешать, включать; ~ **encryption** разрешать [включать] шифрование.

ENC *abbr* (1) ЗАШ (*см. encryption*); (2) ЗАШ (*условное обозначение блока шифрования на схеме*); (3) = encryption key.

encicode *abbr* = enciphered code.

encipher *v* шифровать, зашифровывать; ~ **below decipher above** шифртекст вверху, открытый текст — внизу (*одно из правил шифрования-расширования при работе с шифром Плейфейера*); ~ **function** = encryption function; ~ **in a key** = encrypt in a key; ~ **on the Enigma** шифровать при помощи "Энигмы"; ~ **on the fly** шифровать на лету (*в темпе ввода открытого текста*); ~ **right decipher left** шифртекст справа, открытый текст слева (*одно из правил шифрования-расшифрования при работе с шифром Плейфейера*); ~ **with a binary sequence** шифровать при помощи двоичной последовательности.

encipherability *adv* шифруемость.

enciphered *adj* шифрованный, зашифрованный; ~ **block** блок шифртекста [зашифрованный блок]; ~ **ciphertext** повторно зашифрованный шифртекст; ~ **code** код с перешифровкой; ~ **data** шифрованные данные; ~ **key** зашифрованный ключ; ~ **message** шифрованное сообщение.

encipherer *n* кодирующее устройство, шифратор.

enciphering *n* шифрование, зашифрование; ~ **alphabet** шифрующий алфавит; ~ **bit** бит (*ключа*) шифрования [ключевой последовательности]; ~ **data** шифрование данных; ~ **equation** уравнение зашифрования; ~ **key generator** = key generator; ~ **key sequence generator** генератор ключевой [шифрующей] последовательности; ~ **machine** шифратор; ~ **matrix** матрица шифрования [шифрующая матрица]; ~ **method** метод шифрования; ~ **process** процесс зашифрования; ~ **rule** правило шифрования; ~ **sequence** шифрующая последовательность; ~ **sequence generator** генератор шифрующей последовательности; ~ **transformation** шифрующее преобразование.

encipherment *n* шифрование, зашифрование; ~ **error** ошибка шифрования; ~ **graph** граф шифрования.

enclosure *n* корпус, кожух.

encode *v* кодировать, шифровать.

encoded *adj* кодированный, закодированный; ~ **message** кодированное [закодированное] сообщение.

encodement *n* кодирование, шифрование (*процесс*).

encoder *n* (1) кодирующее устройство, кодер; ~**-decoder** кодер-декодер; (2) кодировщик, шифровальщик.

encoding *n* кодирование, шифрование; ~ **key** ключ кодирования [шифрования]; ~ **machine** шифровальная машина; ~ **matrix** кодирующая [шифрующая] матрица; ~ **rule** правило кодирования [шифрования]; ~ **scheme** схема кодирования [шифрования].

encrypt *v* шифровать, зашифровывать; ~ **information into a cipher** шифровать информацию; ~ **in a key** шифровать на ключе; ~ **a message with a secret key** шифровать сообщение с помощью секретного ключа; ~**-on-demand program** программа шифрования по запросу (*пользователя*); ~ **to a key** = ~ in a key; ~ **under a key** = ~ in a key.

encryptation *n* = encryption.

encrypted *adj* шифрованный, зашифрованный; ~ **channel** канал с криптографической защитой; ~ **file** шифрованный файл; ~ **key** зашифрованный ключ; ~ **key exchange** обмен зашифрованными ключами; ~ **Messaging Protocol** протокол шифрованных сообщений (*используется для защиты от несанкционированного доступа к файлам в файловой системе NFS*); ~ **text** = ciphertext.

encrypter *n* = encryptor.

encrypting *adj* = enciphering.

encryption *n* шифрование, зашифрование; ~ **algorithm** алгоритм шифрования; ~ **algorithm name** название алгоритма шифрования; ~ **application** криптографическое приложение; ~**-based authentication** = cryptographic authentication; ~ **board** плата (*со схемами*) шифрования; ~ **by random grids** шифрование случайными решетками; ~ **calculator** калькулятор с функцией шифрования; ~ **chip** = cryptochip; ~ **circuit** схема шифрования; ~**-decryption** шифрование-расшифрование; ~**-decryption device** шифратор-дешифратор; ~**-decryption key pair** пара ключей шифрования-расшифрования; ~ **equipment manufacturer** производитель шифровального оборудования; ~ **error** = encipherment error; ~ **export controls** контроль за экспортом средств шифрования; ~ **feature** возможность шифрования; ~ **formula** формула шифрования; ~ **function** функция шифрования; ~ **gear** устройство шифрования; ~ **granularity** гранулярность шифрования (*минимальная длина отрезка или блока шифруемого текста*); ~ **integrity** целостность шифрования [шифртекста]; ~ **is easy, but key management is hard** (*old saying in the cryptography business*) "Шифровать легко, трудно ключами управлять" (*старинная поговорка у криптографов*); ~ **issue** вопрос шифрования; ~ **key** ключ шифрования; ~ **key name** имя ключа шифрования; ~ **key weakness** нестойкость ключа шифрования; ~ **machine** = cipher machine; ~ **market** рынок (*сбыта*) средств шифрования; ~ **mechanism** механизм шифрования; ~ **menu** меню шифрования; ~ **module** модуль шифрования; ~**-of-data-only mode** режим шифрования только информационных данных (*без шифрования заголовков, данных управления и контроля*); ~**-only key** ключ только для зашифрования; ~ **option** опция шифрования; ~ **package** шифратор в отдельном корпусе; ~ **procedure** процедура шифрования; ~ **process** процесс шифрования; ~ **product** (1) устройство шифрования [шифратор]; (2) программа шифрования; ~ **program** программа шифрования; ~**-protected** снабженный криптозащитой; ~ **protection** криптографическая защита [криптографическое закрытие информации]; ~ **protocol** протокол шифрования (*определяет порядок применения алгоритма шифрования*); ~ **research** исследования в области шифрования; ~ **round structure** структура цикла [раунда] шифрования; ~ **software module** программный модуль шифрования; ~ **software package** пакет программ шифрования; ~ **standard** стандарт шифрования; ~ **technology** технология шифрования; ~ **under DES** шифрование по DES-алгоритму.

encryptor *n* блок шифрования, схема шифрования; шифратор.

ENDATA *abbr* шифрданные (*см. enciphered data*).

endomorphic *adj* эндоморфный; ~ **cipher** эндоморфный шифр (*шифр, в котором для каждой пары шифрующих преобразований имеется третье, равное их произведению*).

endorsed *adj* аттестованный, сертифицированный; ~ **Cryptographic Products List** Список сертифицированной криптографической аппаратуры; ~ **for un-**

classified cryptographic item аттестованный для использования в несекретном криптографическом изделии (*несекретное криптографическое оборудование, одобренное АНБ США для использования в целях защиты информации, которая имеет значение для обеспечения государственной безопасности США*).

endorsement *n* аттестация; сертификация (*оценка криптографического изделия на предмет его возможного использования для защиты информации, имеющей значение для обеспечения государственной безопасности страны*).

end-to-end *adj* сквозной, межконцевой, оконечный, абонентский; ~ **encryption** сквозное [межконцевое / оконечное / абонентское] шифрование (*производится отправителем шифрованных данных и предполагает их расшифрование получателем*); ~ **encryption cryptosystem** криптосистема межконцевого шифрования.

enemy *n* враг, противник; ~ **cryptanalyst** криптоаналитик противника; ~**'s cipher** шифр противника.

Enigma *n* "Энигма" [*букв.* "Загадка"] (*немецкая шифровальная машина, находившаяся в эксплуатации в 20-е—40-е годы*); ~ **key** ключ "Энигмы"; ~ **secret** секрет "Энигмы"; ~ **simulator** эмулятор "Энигмы"; ~ **variations** модификации "Энигмы".

en route *adv* во время передачи; ~ **to a destination a key should be kept secret** во время передачи получателю ключ должен храниться в секрете.

entrapment *n* ловушка (*намеренное внесение очевидных изъянов в работу криптографического оборудования с тем, чтобы своевременно выявлять все попытки его взлома в процессе эксплуатации*).

entropic *adj* энтропийный; ~ **key** энтропийный ключ.

entropy *n* энтропия.

entry *n* ввод (*напр., ключа*).

EPROM *abbr* ССПЗУ (*см. erasable programmable read-only memory*).

equal *adj* идентичный; ~ **cipher systems** идентичные шифрсистемы (*две шифрсистемы называются идентичными, если имеют одинаковые пространства сообщений и криптограмм, а также совпадающие шифрующие преобразования*).

equally *adv* равно; ~ **likely** = ~ probable; ~ **probable** равновероятный.

equation *n* уравнение.

equipment *n* оборудование, аппаратура; ~ **key** = hardware key.

equiprobable *adj* равновероятный; ~ **keys** равновероятные ключи.

equiprobability *n* равная вероятность.

equivalence *n* эквивалентность; ~ **class** класс эквивалентности.

equivalent *adj* эквивалентный; ~ **keys** эквивалентные ключи.

equivocation *n* надежность; ~ **function** функция надежности ключа (*мера неопределенности ключа для криптоаналитика, анализирующего первые n знаков криптограммы*).

erasable *adj* стираемый; ~ **programmable read-only memory** стираемое программируемое постоянное запоминающее устройство.

erasure *n* стирание (*записи*); уничтожение (*информации*).

ERDL *abbr* ШСОС (*см. encipher right decipher left*).

error *n* ошибка; ~**-correcting code** код, исправляющий ошибки; ~**-free key entry** безошибочный ввод ключа; ~**-propagating cryptosystem** криптосистема с размножением ошибок.

escrow *n* депонирование (*у третьих лиц*); ~ **key** ключ депонирования (*с помощью ключа депонирования шифруются ключи, передаваемые на хранение третьим лицам*).

escrowed *adj* депонированный (*у третьих лиц*); ~ **Encryption Standard** (*американский*) Стандарт шифрования с депонированием (*ключей*).

espionage *n* шпионаж.

establish *v* устанавливать, определять; ~ **a secret key** устанавливать [определять] секретный ключ.

estimation *n* оценка, оценивание.

ETE *abbr* МКШ (*см. end-to-end encryption*).

Euclidean algorithm *n* алгоритм Евклида (*метод нахождения наибольшего общего делителя, названный так по имени древнегреческого математика, который впервые описал его в III веке до нашей эры*).

Euclid's algorithm *n* = Euclidean algorithm.

Euler *prop* Эйлер Л. (*русский математик*); ~ **function** функция Эйлера; ~ **pseudoprime to the base b** псевдопростое число Эйлера по основанию b; ~ **totient function** = ~ function.

Eurocrypt *prop* Еврокрипто (*конференция по криптологии, с 1982 г. ежегодно проводимая в одной из европейских стран*).

even *adj* (1) равномерный; ~ **distribution** равномерное распределение; (2) гладкий; ~ **frequency count** гладкая маркировка (*текста*).

eventually *adj* в конечном счете; ~ **periodic sequence** в конечном счете периодическая последовательность.

exchange *n* обмен; ~ **of keys** обмен ключами.

exclusion *n* запрет; ~ **of a binary function** запрет двоичной функции.

exclusive-OR *n* операция "исключающее ИЛИ" [сложение по mod 2]; ~ **cipher** шифр на основе операции "исключающее ИЛИ" [сложение по mod 2]; ~ **combiner** смеситель по mod 2.

exclusive-OR *v* выполнять операцию "исключающее ИЛИ" [сложение по mod 2]; ~ **ciphertext and plaintext bits** складывать биты шифрованного и открытого текста по mod 2.

exercise *n* тренировка, учение; ~ **key** тренировочный [учебный] ключ (*предназначен для использования во время военных маневров или в процессе обучения гражданского обслуживающего персонала*).

exhaustion *n* полный перебор.

exhaustive *adj* исчерпывающий, тотальный, полный; ~ **break** вскрытие тотальным перебором (*всех возможных вариантов*); ~ **cryptanalysis** переборный [неалгоритмический] криптоанализ; ~ **search** исчерпывающий поиск (*метод поиска неизвестного ключа к шифру полным перебором всех возможных ключей*); ~ **key determination** = ~ cryptanalysis.

exor *abbr* = exclusive-OR.

expanded *adj* расширенный, дополненный; ~ **key** расширенный [дополненный] ключ (*для увеличения его длины*).

expanding *adj* (1) расширяющий; ~ **encipherer** расширяющий шифратор (*увеличивает длину текста при зашифровании*); (2) размножающий (*ошибки*).

expansion *n* расширение; ~ **box** блок расширения; ~ **cipher** расширяющий шифр (*увеличивает длину текста при зашифровании*); ~ **factor** коэффициент расширения (*открытого текста при шифровании*); ~ **permutation** перестановка с расширением.

expected *adj* ожидаемый; ~ **frequency distribution** ожидаемое распределение частот встречаемости букв (*в тексте*).

expert-cryptanalyst *n* эксперт-криптоаналитик.

explicit *adj* открытый, незашифрованный; ~ **key** незашифрованный ключ.

exploration *n* поиск.

exponential *adj* экспоненциальный; ~ **key agreement** = key agreement protocol; ~ **key exchange** = key agreement protocol; ~ **time algorithm** алгоритм с экспоненциальной сложностью.

exponentially *adv* экспоненциально; ~**-derived key** ключ, получаемый при использовании экспоненциальной функции.

exponentiation *n* возведение в степень; ~ **cipher** шифр на основе (*дискретного*) возведения в степень; ~**-based cipher** = ~ cipher.

exportable *n* разрешенный к экспорту; ~ **40-bit session keys** разрешенные к экспорту 40-битовые сеансовые ключи.

exposure *n* раскрытие, разглашение (*зашифрованной информации или ключа*).

extended *adj* расширенный, дополненный; ~ **key** расширенный [дополненный] ключ; ~ **message** расширенное [дополненное] сообщение (*получено путем добавления аутентификатора к исходному тексту сообщения*).

extension *n* расширение; ~ **field** алгебраическое расширение поля.

external *adj* наружный, внешний; ~ **error control for cryptosystems** наружный контроль за ошибками в криптосистемах; ~ **key** внешний ключ [ключ от внешнего источника]; ~ **key entry** ввод ключа извне [ввод ключа от внешнего источника]; ~ **variant cipher system** внешняя вариантная шифрсистема.

externally *adv* извне; ~ **loaded key** = external key; ~ **supplied key** = external key.

extra *adj* дополнительный; ~ **key** дополнительный ключ.

extract *v* извлекать; ~ **logarithms in the field GF(2^n)** извлекать логарифм в поле GF(2^n).

extraction *n* извлечение.

extractor *n* экстрактор (*функция или устройство, выполняющие обратное преобразование по отношению к функции или устройству объединения*).

F

fabricated *adj* = fake.

FAC *abbr* ФАК (*см. factorization*).

facilities *n pl* средства.

facsimile *adj* факсимильный; ~ **encryption** шифрование факсимильных сообщений.

facsimile *n* факсимильная связь.

factor *n* (1) множитель; (2) коэффициент; (3) фактор; ~**-ring** фактор-кольцо.

factor *v* разлагать на множители; ~ **a composite number** разлагать составное число на множители.

factoring *n* = factorization.

factorization *n* разложение на множители; ~ **cryptosystem** криптосистема на основе (*сложности задачи*) разложения больших чисел на (*простые*) сомножители.

factory *n* изготовитель; ~**-installed key** ключ, устанавливаемый при изготовлении криптографической аппаратуры; ~**-programmed key** = ~-installed key.

fail-safe *adj* нечувствительный к сбоям; ~ **circuit** цепь, предотвращающая передачу открытого текста при работе в режиме шифрования.

fail-stop *adj* прекращающий работу при появлении ошибки; ~ **signature scheme** схема цифровой подписи с прекращением работы при появлении ошибки.

failure *n* = fault.

fake *adj* поддельный, фальшивый; ~ **key** поддельный [фальшивый] ключ.

false *adj* = fake.

family *n* семейство; ~ **of cipher machines** семейство шифровальных машин.

FAPSI *abbr* ФАПСИ (*Федеральное агентство правительственной связи и информации*).

fast *n* быстрый; ~**Data Finder** Устройство быстрого поиска данных (*микросхема для ускорения процесса поиска в больших компьютерных базах данных*); ~ **Encryption Algorithm** Быстрый алгоритм шифрования; ~ **Fourier transform** быстрое преобразование Фурье; ~ **Walsh Transform** быстрое преобразование Уолша.

fault *n* сбой; ~ **analysis** анализ сбоев (*для вскрытия ключей в работе криптографического процессора намеренно вызывается возникновение сбоев*); ~ **analysis attack** атака методом анализа сбоев.

fax *abbr* факс (*см. facsimile*).

FBI *abbr* ФБР (*см. Federal Bureau of Investigation*).

FCSR *abbr* РСОСП (*см. feedback with carry shift register*).

FD *abbr* УЗК [УВК] (*см. fill device*).

FDC *abbr* ФДК (*см. French diplomatic communications*).

FDF *abbr* УБП (*см. Fast Data Finder*).

FDM *abbr* МЧД (*см. frequency division multiplexing*).

FEAL *abbr* БАЛШ (*см. Fast Encryption Algorithm*).

feasible *adj* осуществимый, выполнимый; ~ **decryption** осуществимое дешифрование.

federal *adj* федеральный; ~ **Agency for Government Communications and Information** Федеральное агентство правительственной связи и информации (*российское правительственное ведомство, занимающееся перехватом и дешифрованием сообщений*); ~ **Agency of Government Communications and Information** = ~ Agency for Government Communications and Information; ~ **Agency on Government Communications and Information** = ~ Agency for Government Communications and Information; ~ **Bureau of Investigation** Федеральное бюро расследований (*США*); ~ **Government Communications and Information Agency** = ~ Agency for Government Communications and Information; ~ **Information Processing Standard 140-1** Федеральный стандарт (*США*) на обработку информации 140-1 (*соответствие этому стандарту является необходимым требованием, которое предъявляется к криптографическим системам защиты информации в правительственных ведомствах США и Канады*); ~ **Information Security Agency** Федеральное агентство безопасности информации (*немецкое правительственное ведомство, занимающееся перехватом и дешифрованием сообщений*).

feedback *n* обратная связь; ~ **coefficient** коэффициент обратной связи; ~ **function** функция обратной связи; ~ **polynomial** полином обратной связи; ~ **shift**

register регистр сдвига с обратной связью; ~ **tap** точка съема (*регистра сдвига с обратной связью*); ~ **with carry shift register** регистр сдвига с обратной связью и переносом.

Feistel *prop* Фейстел Х. (*американский криптограф, автор многочисленных криптографических патентов, в т.ч. — алгоритма шифрования "Люцифер"*); ~ **cipher** шифр Фейстела (*является разновидностью итеративного блочного шифра*).

Fermat *prop* Ферма П. (*французский математик*); ~ **prime** простое число Ферма (*простое число вида 2^n+1*); ~**'s little theorem** малая теорема Ферма.

Fetterlein *prop* Феттерлейн Э. (*русский криптоаналитик, после Октябрьской революции эмигрировавший в Англию*).

FFT *abbr* БПФ (*см. fast Fourier transform*).

Fibonacci *prop* Фибоначчи (*итальянский математик*); ~ **keystream generator** генератор ключевой последовательности [ключевого потока] на основе чисел Фибоначчи.

field *adj* полевой (*применяемый на поле боя*); ~ **cipher** полевой шифр.

field *n* (1) поле; ~**-mode encryption** шифрование отдельных полей данных; ~ **of characteristic q** поле с характеристикой q; (2) область, сфера (*деятельности*); ~ **of cryptography** область криптографии.

file *n* файл; ~ **grow factor** коэффициент расширения файла (*при его шифровании*).

fill *n* (1) = padding; (2) заполнение (*регистра сдвига*); ввод [загрузка] (*ключа*); ~ **device** = key fill device .

fill *v* заполнять; ~ **out** = pad *v*.

filler *n* = padding.

filter *n* фильтр (*функция, осуществляющая преобразование некоторого входного множества и получающая на выходе только те элементы этого множества, которые удовлетворяют определенному критерию*); ~ **generator** фильтрующий генератор (*линейный регистр сдвига с обратной связью и нелинейным выходом*).

FIMAS *abbr* САСФУ (*см. Financial Institution Message Authentication Standard*).

financial *adj* финансовый; ~ **cryptography** финансовая криптография; ~ **Institution Message Authentication Standard** *n* Стандарт аутентификации сообщений для финансовых учреждений (*США*).

find *v* отыскивать, находить; ~ **a key** находить ключ.

fingerprint *n* = message digest.

finite *adj* конечный; ~ **field logarithm** логарифм над конечным полем; ~ **field with q elements** конечное поле из q элементов; ~ **key** ключ конечной длины.

FIPS 140-1 *abbr* ФСОИ (*см. Federal Information Processing Standard 140-1*).

firewall *n* межсетевой шлюз, брандмауэр, экранирующий фильтр (*аппаратно — программный комплекс управления доступом из одной сети в другую*).

FISA *abbr* (1) = Foreign Intelligence Surveillance Act; (2) ФАБИ (*см. Federal Information Security Agency*).

fixed *adj* стационарный, постоянный; ~ **cipher mode** режим шифрования на постоянном ключе; ~ **coefficient random number generator** генератор последовательности случайных чисел с постоянными коэффициентами; ~**-key cipher** шифр с постоянным ключом; ~ **length key** = finite key; ~ **plaintext message unit** стационарный отрезок открытого текста сообщения (*блок открытого текста, который остается неизменным после применения к нему шифрующего преобразования*).

flat *adj* = even.

flaw *n* дефект; ~ **in the encipherment** дефект шифрования.

flawed *adj* дефектный; ~ **cipher** дефектный шифр.

-fold *в сложных и сложносоставных словах имеет значение* —кратный; **L-~ security** L-кратная стойкость (*невскрываемость при криптоанализе L шифрсообщений*).

forced *adj* принудительная; ~ **key change** принудительная смена [замена] ключа.

foreign *adj* зарубежный; ~ **communications** зарубежные каналы связи; ~ **Intelligence Surveillance Act** Закон о контроле за сбором разведывательной информации о зарубежных странах (*введен в действие в США в 1978 г. в целях обеспечения надзора за деятельностью АНБ со стороны американских правоохранительных органов*); ~ **Intelligence Surveillance Appeals Court** Апелляционная судебная инстанция по контролю за сбором разведывательной информации о зарубежных странах; ~ **Intelligence Surveillance Court** Судебная инстанция по контролю за сбором разведывательной информации о зарубежных странах.

forge *v* подделывать; ~ **a certificate** подделывать сертификат; ~ **coded signals** подделывать кодированные сигналы.

forged *adj* поддельный, подложный; ~ **message** поддельное [подложное] сообщение.

format *n* формат.

formula *n* формула.

Fortezza *prop* "Фортецца" (*название модельного ряда PCMCIA-карт, которые реализуют криптографические алгоритмы, разработанные в рамках американской правительственной программы "Кэпстоун"*).

Fort Meade *prop* Форт-Мид (*город в американском штате Мэриленд, где расположена штаб-квартира АНБ США*).

forward *adj* прямой; ~ **asymmetric encryption** шифрование с прямой асимметрией (*ключ получателя не может быть определен по известному ключу отправителя*).

four- *в сложных и сложносоставных словах имеет значение* четырех-; ~**-square cipher system** шифрсистема "четыре квадрата"; ~ **four-wheel Enigma** четырехроторная [четырехдисковая] "Энигма".

Fourier *prop* Фурье Ж. (*французский математик*); ~ **transform** преобразование Фурье.

fraudulent *adj* поддельный, ложный; ~ **ciphertext** ложный шифртекст; ~ **cryptogram** поддельная криптограмма; ~ **message** ложное [поддельное] сообщение.

freely *adv* свободно; ~**-selected key** свободно выбираемый ключ.

French *adj* французский; ~ **diplomatic communications** французские дипломатические каналы связи.

frequency *n* частота, повторяемость; ~ **distribution** частотное распределение; ~ **division multiplexing** мультиплексирование с частотным делением; ~ **inverter** = inverter; ~ **hop inverter** = hop inverter; ~ **matching** согласование частот (*соотнесение частот встречаемости букв в шифрованном и открытом текстах*); ~ **of key change** частота смены ключа; ~ **test** частотный тест (*тест для проверки случайности двоичной последовательности путем подсчета в ней количества единиц и нулей*).

Friedman *prop* Фридман У. (*американский военный криптолог, незадолго до начала второй мировой войны вскрывший японскую "пурпурную" шифровальную машину*).

friendly *adj* дружественный; ~ **cryptanalyst** дружественный криптоаналитик (*специалист, анализирующий шифры с целью нахождения в них скрытых уязвимых мест, что позволяет вносить в эти шифры необходимые изменения для повышения их стойкости*).

FSR *abbr* РСОС (*см. feedback shift register*).

full *adj* полноценный (*неурезанный, без сокращений*); ~ **DES** полноценный DES.

function *n* функция; ~ **D** (*deciphering*) функция Р (*расшифрование*); ~ **E** (*enciphering*) функция Ш (*шифрование*); ~**'s inverse** обратная функция.

FWT *abbr* БПУ (*см. Fast Walsh Transform*).

G

GA *abbr* ГА (*см. genetic algorithm*).

Gadsby *prop* "Гэдсби" (*роман американского писателя Э. Райта, в котором нет ни единого слова, содержащего букву e английского алфавита*).

gain *v* получить; ~ **access** получить доступ.

Galois *prop* Галуа Э. (*французский математик*); ~ **field** поле Галуа; ~ **shift register** регистр сдвига Галуа.

game *n* игра; ~ **theory cryptography** криптография на основе теории игр.

gamma *n* гамма (*случайная или псевдослучайная битовая последовательность*).

gap *n* "брешь" (*набор подряд идущих нулей в битовой последовательности, которому предшествует и за которым следует единица*).

garbled *adj* искаженный; ~ **message** искаженное сообщение.

gardening *n* "садоводство" (*уловка, с помощью которой криптоаналитик заставляет противника послать криптограмму, открытый текст которой ему известен*).

GC&CS *abbr* (1) ПШКШ (*см. Government Code and Cipher School*); (2) Golf, Cheese and Chess Society (*шутливая расшифровка аббревиатуры GC&CS, придуманная сотрудниками ПШКШ*).

gcd *abbr* НОД (*см. greatest common divisor*).

GCHQ *abbr* ЦПС [ШКПС] (*см. Governmental Communications Head-Quarters*); ~ **base** база ЦПС [ШКПС].

GCSB *abbr* ББПС (*см. Government Communications Security Bureau*).

GDES *abbr* GDES-алгоритм (*см. Generalized DES*).

gear *n* устройство.

general *adj* универсальный, общий; ~ **model of authentication** общая модель аутентификации; ~ **purpose computer** универсальный компьютер.

generalized *adj* обобщенный; ~ **DES** обобщенный DES-алгоритм.

generate *v* генерировать, порождать, формировать; ~ **a ciphertext** формировать шифртекст.

generating *adj* порождающий; ~ **cycle** порождающий цикл; ~ **polynomial** порождающий полином.

generation *n* генерация, порождение; ~ **of keys** генерация [порождение] ключей.

generator *n* генератор, датчик; ~ **polynomial** = generating polynomial.

generic *adj* обобщенный; ~ **key** обобщенный ключ (*используется для получения из него других ключей*).

genetic *adj* генетический; ~ **algorithm** генетический алгоритм (*разновидность алгоритма направленного случайного поиска*).

genuine *adj* подлинный, настоящий; ~ **key** подлинный [настоящий] ключ.

geometric *adj* геометрический; ~ **transposition cipher** шифр с геометрическими перестановками.

German *adj* немецкий; ~ **Book** "немецкий" сборник (*во время второй мировой войны составлялся английскими криптоаналитиками и включал в себя все прочитанные немецкие шифртелеграммы на языке оригинала*).

GF(p) *abbr* GF(p) (*см. Galois field*).

Gifford *prop* Гиффорд Д. (*сотрудник Массачусетского технологического института*); **~'s cipher** шифр Гиффорда (*потоковый шифр, изобретенный Гиффордом в 1984 г.*).

gisting *n* аннотирование, реферирование (*замена содержания перехваченного сообщения текстом, передающим смысл этого сообщения в самых общих чертах*).

global *adj* глобальный; **~ public key infrastructure** глобальная инфраструктура шифрования с открытым ключом.

go *v* делаться, становиться; **~ green** *жарг.* переключаться с открытой на шифрованную связь; **~ secure button** кнопка "включить защиту".

Gold Bug *n* "Золотой жук" (*рассказ американского писателя Э. По с описанием оригинального криптоаналитического исследования шифрованного текста*).

Golomb *prop* Голомб С. (*американский криптолог*); **~'s postulates** = ~'s randomness postulates; **~'s randomness postulates** критерии Голомба проверки случайности (*двоичной последовательности*).

good *adj* стойкий; **~ cipher** стойкий шифр.

GOST *abbr* ГОСТ (*Государственный стандарт*); **~ 28147-89** ГОСТ 28147-89 (*советский / российский государственный стандарт шифрования данных*).

Gouzenko *prop* Гузенко И. (*шифровальщик посольства СССР в Оттаве, в 1945 г. попросивший политического убежища в Канаде и оказавший американцам существенную помощь в работе над вскрытием криптосистем, которые использовались для засекречивания переписки резидентуры советской разведки в Вашингтоне с Москвой*).

government *n* правительство; **~ Code and Cipher School** = Governmental School of Codes and Ciphers; **~ Communications Security Bureau** Бюро безопасности правительственной связи (*Новой Зеландии*); **~-endorsed encryptor** одобренный правительственным ведомством шифратор.

governmental *adj* правительственный; **~ Communications Head-Quarters** Центр правительственной связи [Штаб-квартира правительственной связи] (*английское правительственное ведомство, занимающееся перехватом и дешифрованием сообщений*); **~ School of Codes and Ciphers** Правительственная школа кодов и шифров (*английское правительственное ведомство, занимавшееся перехватом и дешифрованием сообщений с 1919 по 1939 г.*).

GPKI *abbr* ГИОК (*см. global public key infrastructure*).

-gram *в сложных и сложносоставных словах имеет значение* —грамма; **N-~** N-грамма; **N-~ analysis** анализ N-грамм.

G-random *adj* Г-случайный; **~ binary sequence** Г-случайная двоичная последовательность (*двоичная последовательность называется Г-случайной, если она*

удовлетворяет трем критериям случайности, сформулированным американским криптологом С. Голомбом в 1967 г.).

granularity *n* гранулярность.

graph *n* граф.

greatest common divisor *n* наибольший общий делитель.

Green Book *n* "Зеленая книга" (*аналог американской "Оранжевой книги" в Англии*).

grille *n* решетка; ~ **transposition** шифр перестановки на решетке.

group *adj* групповой; ~ **key** групповой ключ (*предназначен для использования группой лиц*); ~ **signature** групповая цифровая подпись; ~ **structure** групповая структура (*про шифр говорят, что он имеет групповую структуру, если результат зашифрования открытого текста сначала на одном ключе, а затем повторного шифрования полученного шифртекста на втором эквивалентен шифрованию того же самого открытого текста на каком-то третьем ключе*).

GRU *abbr* ГРУ (*Главное разведывательное управление*).

GSC&C *abbr* ПШКШ (*см. Governmental School of Codes and Ciphers*).

guard *n* дежурный; ~ **key** дежурный ключ (*постоянно хранимый в криптографическом устройстве*).

guess *v* угадывать; ~ **a key** угадывать ключ.

guessed *adj* угаданный; ~ **plaintext attack** атака на основе угаданного открытого текста (*атака на криптосистему с открытым ключом, в ходе которой атакующий угадывает открытый текст перехваченной криптограммы, шифрует его при помощи открытого ключа получателя и сравнивает с шифрованным текстом*).

H

hack *n* = hacking.

hack *v* заниматься хакерством, хачить.

hackee *n* женщина-"хакер".

hacker *n* "хакер" (*человек, прекрасно разбирающийся в компьютерах и способный вытворять с ними то, что другим кажется чистым волшебством, в том числе — находить пути обхода средств контроля доступа к компьютерным системам*); ~**-proof** защищенный от проникновения "хакеров"; ~**'s trick** хакерский трюк.

hacking *n* хакинг, хак (*получение информации из компьютера за счет обхода средств контроля доступа к ней*).

Hadamard *prop* Адамар Ж. (*французский математик*); ~ **product** произведение Адамара (*поразрядное логическое умножение соответствующих членов двух би-*

товых последовательностей равной длины); ~ **transform** преобразование Адамара.

Hagelin *prop* Хагелин Б. (*основатель и владелец швейцарской компании "Крипто АГ" — одной из самых известных в мире фирм по производству коммерческих шифраторов*); ~ **machine** шифратор Хагелина.

half- *в сложных и сложносоставных словах имеет значение* полу-.

half-rotor *n* полуротор.

Hamming *prop* Хэмминг Р. (*американский математик*); ~ **distance** расстояние Хэмминга; ~ **weight** вес Хэмминга; ~**-weight attack** = ~-weight cryptanalysis; ~**-weight cryptanalysis** криптоанализ с помощью веса Хэмминга (*использует информацию о весе Хэмминга промежуточных значений, вычисляемых в ходе работы криптографического алгоритма*); ~ **weight of a key** вес Хэмминга для ключа.

hand *adj* ручной; ~ **cipher** ручной шифр; ~**-delivered key** ключ, доставляемый посыльным [курьером].

hard *adj* (1) трудный; ~ **integer** "трудное" целое число (*целое, не обладающее большим количеством простых делителей, которые либо сами являются малыми по величине, либо имеют специальный вид, облегчающий их выделение*); ~**-to-break cipher** трудновскрываемый шифр; ~**-to-invert function** = one-way function; (2) труднорешаемый (*о математической задаче, для которой не существует алгоритма ее решения с полиномиальной сложностью*); (3) твердый; ~**-copy key** ключ, записанный на твердом носителе (*перфокарте, магнитном диске и т. д.*); ~**-copy keying** ввод ключа с твердого носителя (*перфокарты, магнитного диска и т. д.*); ~**-drive encryption program** программа шифрования информации на жестком диске.

hardware *n* аппаратные средства; ~ **cryptoalgorithm-cracker** аппаратная система вскрытия криптоалгоритма; ~**-assisted encryption** = ~ encryption; ~**-based key** = ~ key; ~ **cracker** аппаратная система вскрытия (*ключа, шифра*); ~ **encryption** аппаратные средства шифрования [аппаратное шифрование]; ~**-implemented encryption** = ~ encryption; ~ **key** (1) ключ аппаратного шифрования; (2) ключ, вмонтированный в криптографическое устройство; (3) аппаратно генерируемый ключ; ~ **random number generator** аппаратно реализованный генератор случайных чисел.

hardwired *adj* постоянно [жестко] вмонтированный; постоянно запаянный; *проф.* "зашитый"; ~ **key entry** ввод ключа методом постоянного монтажа (*в аппаратуре*).

hash function *n* хэш-функция [функция хэширования] (*позволяет получать фиксированного размера результат на выходе при подаче на вход данных произвольного объема*).

hashing *n* хэширование; ~ **algorithm** алгоритм хэширования.

header *n* заголовок (*сообщения*); ~ **encryption** шифрование заголовка (*сообщения*).

Hellman *prop* Хеллман М. (*американский криптолог, один из родоначальников криптографии с открытым ключом*).

hex *abbr* = hexadecimal.

hexadecimal *adj* шестнадцатеричный; ~ **key** шестнадцатеричный (*представленный в шестнадцатеричной форме*) ключ.

HF *abbr* ВЧ.

hidden-key cryptography *n* = secret-key cryptography.

hierarchy *n* иерархия; ~ **of keys** = key ~.

high- *в сложных и сложносоставных словах имеет значение* высоко-.

highest *superl* наивысший; ~ **frequency letter** буква с наивысшей частотой встречаемости (*в тексте*).

high-frequency *adj* высокочастотный.

high-grade *adj* первоклассный; высокого уровня; ~ **cryptosecurity** криптостойкость высокого уровня; ~ **military communications** сообщения [переписка] высшего военного командования.

high-level *adj* = high-grade.

high-security *adj* высокостойкий, обладающий высокой стойкостью; ~ **cipher** шифр высокой стойкости; ~ **key** высокостойкий ключ.

high-speed *adj* высокоскоростной; ~ **cryptography** высокоскоростная криптография.

Hill *prop* Хилл Л. (*американский криптолог*); ~ **cipher system** шифрсистема Хилла (*с помощью алгебраической процедуры, описанной Л. Хиллом в 1929 г., производится шифрование открытого текста порциями по n букв, где n — произвольное положительное целое*).

historical work characteristic *n* достигнутая оценка рабочей характеристики (*средний объем работы, необходимой для нахождения ключа шифра на основе знания n знаков шифртекста при использовании наилучшего из известных методов анализа данного шифра*).

hobbyist *adj* любительский, непрофессиональный; ~ **cipher** любительский шифр.

homogeneous *adj* однородный; ~ **cipher** однородный шифр.

homomorphism *n* гомоморфизм.

homophone *n* гомофон.

homophonic *adj* гомофонический; ~ **cipher** гомофонический шифр (*при замене одного знака открытого текста выбор осуществляется из нескольких знаков шифртекста*); ~ **substitution** гомофонная замена.

hop *n* скачкообразное изменение; ~ **inverter** скремблер с инверсией частот и скачкообразным изменением частоты инвертирования.

hostile *adj* враждебно настроенный; ~ **outsider** посторонний злоумышленник.

hotline *n* линия экстренной связи; ~ **between Moscow and Washington** линия экстренной связи между Москвой и Вашингтоном.

hours *n pl* время работы.

HT *abbr* ПА (*см. Hadamard transform*).

human intelligence *n* агентурная разведка.

HUMINT *abbr* = human intelligence.

hut *n* барак (*так назывались административные здания на территории поместья Блетчли-Парк, в которых во время второй мировой войны размещались подразделения английской дешифровальной службы, занимавшейся чтением немецкой шифрпереписки*).

HW *abbr* = hardware.

hybrid *adj* гибридный; ~ **cryptosystem** гибридная криптосистема (*криптосистема с открытым ключом задействуется только для управления общими ключами, которые затем используются в традиционных криптосистемах с секретным ключом*); ~ **public-secret key cryptosystem** = ~ cryptosystem.

hypothesis *n* гипотеза; ~**-testing problem** задача проверки гипотез.

I

IACR *abbr* МАКИ (*см. International Association for Cryptologic Research*).

IC *abbr* ИС (*см. index of coincidence*).

ID *abbr* ИД (*см. identity*).

IDEA *abbr* МАЗД (*см. International Data Encryption Algorithm*).

ideal *adj* идеальный; ~ **cipher** идеальный шифр (*с бесконечным расстоянием единственности*).

idempotent *adj* идемпотентный; ~ **element** идемпотентный элемент (*элемент кольца, равный своему квадрату*).

identification *n* (1) идентификация, опознавание; ~ **friend or foe system** система опознавания "свой-чужой"; (2) определение, выявление; ~ **of a cipher system** определение шифрсистемы; ~ **of a plaintext** выявление открытого текста.

identify *v* устанавливать, определять; ~ **a key** устанавливать [определять] ключ.

identity *n* (1) тождество, идентичность; ~ **transformation** тождественное преобразование; (2) идентификация (*личности*); ~**-based cryptosystem** идентификационная криптосистема [криптосистема на основе идентификационной информации].

idiomorph *n* = word pattern.

idle *adj* бездействующий, простаивающий; ~ **mode** холостой режим [режим ожидания].

IEC *abbr* ВКО (*см. internal error control*).

IK *abbr* ОК (*см. interchange key*).

immune *adj* стойкий; ~ **to linear cryptanalysis** стойкий против линейного криптоанализа.

immunization *n* повышение криптостойкости.

impenetrable *adj* сверхстойкий; ~ **cryptosystem** сверхстойкая криптосистема.

imperfect *adj* несовершенный; ~ **cipher** несовершенный шифр.

impersonate *v* выдавать себя за (*кого-то*); ~ **a sender** выдавать себя за отправителя (*сообщения*).

impersonation *n* изменение авторства, имитация; ~ **attack** попытка имитации [атака с изменением авторства]; ~ **of a transmitter** имитация передатчика (*тип угрозы в теории аутентификации информации*).

implant *n* закладка (*электронное устройство для получения несанкционированного доступа к электромагнитным излучениям, которые содержат информацию о ключах, открытых текстах и алгоритмах функционирования криптографического оборудования*).

implementation *n* реализация, внедрение; ~ **of a cryptoalgorithm** реализация криптоалгоритма (*программными или аппаратными средствами*); ~**-specific timing characteristics** зависящие от реализации временные характеристики (*криптосистемы*).

implemented *adj* реализованный.

impossible *adj* неосуществимый, невыполнимый, находящийся за пределами возможного; ~ **cryptanalysis** криптоанализ на основе событий с нулевой вероятностью (*разновидность дифференциального криптоанализа*); ~**-to-break** неподдающийся вскрытию [абсолютно стойкий].

impostor *n* самозванец; человек, выдающий себя за другое лицо.

improper *adj* неправильный, ошибочный; ~ **encryption** неправильное шифрование.

in *prep* (*указывает на способ выражения, форму представления и т. д.*); ~ **black** *жарг.* = ~ cipher; ~ **cipher** шифрованный [зашифрованный]; ~ **clear** клером [открытым текстом]; ~ **the clear** = ~ clear; ~ **code** кодированный [закодированный]; ~ **red** = ~ clear.

in-built *adj* встроенный, вмонтированный.

incidence *n* инцидентность; ~ **matrix of authenticating rules** матрица инцидентности при данных правилах аутентификации.

in-circuit *adj* оперативный, линейный; ~ **encryption device** прибор оперативного [линейного] шифрования.

in-coming *adj* входной; ~ **message** входное сообщение.

indecipherable *adj* = undecipherable.

index *n* индекс; показатель; ~ **of coincidence** индекс совпадения; ~ **of cryptographic strength** показатель криптографической стойкости.

indicator *n* индикатор; ~ **sequence** индикаторная последовательность [вектор-индикатор].

indirect *adj* косвенный; ~ **eavesdropping** косвенный перехват (*напр., с помощью приема электромагнитного излучения работающих радиоэлектронных устройств*); ~ **encryption** шифрование программными средствами; ~ **symmetry method** = ~ symmetry technique; ~ **symmetry technique** метод косвенной симметрии (*применяется для вскрытия шифров периодической многоалфавитной замены*).

indivisible *adj* = prime.

infallible *adj* надежный; ~ **cipher** надежный шифр.

infeasible *adj* неосуществимый, невыполнимый.

infinite *adj* бесконечный; ~ **random key** бесконечный случайный ключ.

information *n* информация; ~ **authentication** аутентификация информации; ~-**bearing emission** излучение (*работающей радиоэлектронной аппаратуры*), несущее информацию; ~ **conveyed by a message** содержание сообщения; ~ **hiding** сокрытие информации; ~ **security** защита [безопасность] информации; ~-**theoretic security** теоретико-информационная стойкость; ~ **theory cryptography** криптография на основе теории информации; ~ **war** информационная война.

INFOSEC *abbr* ИНФОБЕЗ (*см. information security*).

infowar *abbr* инфовойна (*см. information war*).

inherent *adj* вмонтированный, встроенный; ~ **key** встроенный [вмонтированный] (*в схему криптографического устройства*) ключ.

initial *adj* начальный; ~ **fill** начальное заполнение (*регистра сдвига*); ~ **key** (1) начальный ключ; (2) ключ инициализации (*криптосистемы*); ~ **load** начальная загрузка [начальное заполнение]; ~ **permutation** начальная перестановка (*в DES-алгоритме*); ~ **state** начальное состояние (*напр., шифратора*).

initialization *n* (1) инициализация, инициирование; (2) начальное заполнение; начальная загрузка; ~ **vector** вектор начального заполнения.

initiate *v* начинать; ~ **communication** начать сеанс связи.

initiation *n* = initialization.

injection *n* ввод (*напр., ключа*).

inject *v* вводить; ~ **a key** вводить ключ.

injector *n* устройство ввода.

in-line *adj* = on-line.

in-phase *adj* совпадающий по фазе; ~ **autocorrelation** совпадающая по фазе автокорреляция.

input *adj* входной; ~ **alphabet** входной алфавит; ~ **message** входное сообщение; ~ **sequence** входная последовательность; ~ **transformation** входное преобразование (*блока шифртекста*).

input *n* вход.

insecure *adj* (1) нестойкий; ~ **cryptosystem** нестойкая криптосистема; (2) незащищенный; ~ **channel** незащищенный канал (*связи*).

insert *v* (1) подключать; ~ **oneself into a communications link** подключаться к линии связи; (2) вводить, устанавливать; ~ **a key** вводить [устанавливать] ключ.

inserted *adj* введенный, установленный (*в криптографическое устройство*); ~ **key** введенный [установленный] (*в криптографическое устройство*) ключ.

insertion *n* ввод; вставка.

insider *n* свой (*человек*); ~ **cheating** обман со стороны своего; ~ **deceit** = ~ cheating.

install *v* устанавливать, вводить в действие; ~ **individual keys on computers** вводить в действие индивидуальные ключи на компьютерах.

installation *n* установка, инсталляция (*напр., ключа*).

installator *n* лицо, осуществляющее установку [инсталляцию].

installed *adj* = inserted.

instantaneous *adj* мгновенный; ~ **encipherer** шифратор мгновенного действия.

integrity *n* целостность, сохранность, защищенность (*состояние, в котором данные используются только установленным образом*); ~ **of information** целостность [сохранность] информации; ~ **of a key against substitution or alteration** защищенность ключа от подмены или изменения.

intelligence *n* разведка; ~ **cycle** разведывательный цикл (*многоступенчатый процесс, в ходе которого производится поиск, сбор, обработка и доведение до заинтересованных лиц радиоразведывательной информации*).

intended *adj* предполагаемый; ~ **recipient** предполагаемый адресат (*сообщения*).

interactive *adj* интерактивный, диалоговый; ~ **key distribution** интерактивное [диалоговое] распределение ключей.

inter-bit *adj* межбитовый; ~ **dependency** межбитовая зависимость.

intercept *n* перехват (*перехваченное сообщение*); ~ **operator** оператор (*средства*) перехвата; ~ **station** = intercepting station.

intercept *v* перехватить; ~ **a letter, a message, a telegram** перехватить письмо, сообщение, телеграмму.

interceptability *n* доступность для перехвата, уязвимость к перехвату.

intercepted *adj* перехваченный; ~ **communications** = ~ traffic; ~ **data** перехваченные данные; ~ **message** перехваченное сообщение; ~ **traffic** перехваченная переписка.

intercepting *n* перехват; ~ **equipment** аппаратура перехвата; ~ **in Internet** перехват в "Интернет"; ~ **requirements** требования к перехвату; ~ **station** станция перехвата.

interception *n* перехват; ~ **program** программа перехвата.

interceptor *n* перехватчик; тот, кто осуществляет перехват; устройство перехвата.

interchange *adj* обменный; ~ **key** обменный ключ (*предназначен для взаимного обмена сообщениями*).

interface *n* интерфейс; сопряжение.

interloper *n* = interceptor.

intermediate *adj* промежуточный, вспомогательный; ~ **key** промежуточный [вспомогательный] ключ.

internal *adj* внутренний; ~ **error control for cryptosystems** внутренний контроль за ошибками в криптосистемах; ~ **key** = inherent key; ~ **variant cipher system** внутренняя вариантная шифрсистема.

internally *adv* внутри; ~ **generated key** генерируемый внутри (*криптографического устройства*) ключ.

international *adj* международный; ~ **Association for Cryptologic Research** Международная ассоциация криптологических исследований; ~ **Cryptography Experiment** Международная криптографическая инициатива (*неформальная программа, целью которой является проверка соблюдения действующих в США экспортных ограничений на экспорт стойких криптографических средств*); ~ **Data Encryption Algorithm** Международный алгоритм зашифрования данных; ~ **Traffic in Arms Regulations** Законодательство по контролю за экспортом вооружений (*свод законов США, регулирующих продажу американского оружия за рубеж, включая шифраторы и другое криптографическое оборудование*).

Internet *n* "Интернет" (*глобальная компьютерная сеть*); ~ **Secure Protocol** Защищенный протокол для подключения к "Интернет" (*включает средства криптогарфической защиты и аутентификации*).

interruptor *n* прерыватель; ~ **letter** буква-прерыватель; ~ **letter aperiodic** апериодичный шифр многоалфавитной замены с буквой-прерывателем.

intersymbol *adj* межсимвольный; ~ **dependency** межсимвольная зависимость.

inter-user *adj* межпользовательский; ~ **key** межпользовательский ключ (*предназначен для обмена сообщениями между несколькими пользователями*).

intractability *n* неразрешимость (*математической задачи*) с полиномиальной сложностью.

intractable *adj* (1) неразрешимый с полиномиальной сложностью (*о математической задаче, для которой не существует алгоритма решения с полиномиальной сложностью*); (2) стойкий; ~ **cryptosystem** стойкая криптосистема.

introduce *v* ввести в действие, начать использовать; ~ **the four rotor Enigma** ввести в действие четырехроторный вариант "Энигмы"; ~ **a new key** начать использовать новый ключ.

intruder *n* злоумышленник, нарушитель; человек, незаконно присвоивший себе чужие права.

invasion *n* нарушение; ~ **of privacy** нарушение секретности [конфиденциальности].

inverse *adj* обратный; ~ **key** обратный ключ; ~ **keys** взаимно обратные ключи; ~ **mapping** обратное отображение; ~ **Pseudo-Hadamard transform** обратное псевдопреобразование Адамара.

invert *n* обратить; ~ **function f** обратить функцию f [найти функцию, обратную к f].

invertible *adj* обратимый; ~ **mapping** обратимое отображение; ~ **modulo N** обратимый по модулю N.

inverter *n* скремблер с инверсией частот.

invisible *adj* невидимый; ~ **ink** "невидимые" чернила.

involute *adj* сложный; ~ **cipher** сложный шифр.

invulnerable *adj* неуязвимый.

IP *abbr* НП (*см. initial permutation*).

IPHT *abbr* ОППА (*см. Inverse Pseudo-Hadamard transform*).

IPSEC *abbr* = Internet Secure Protocol.

irreducible *adj* неприводимый; ~ **over GF(2)** неприводимый над полем GF(2).

irreversible *adj* = one-way.

isolation *n* разделение, разграничение (*открытой и шифрованной информации*).

isolog *n* изолог (*идентичный или почти идентичный открытый текст, зашифрованный на разных ключах или с помощью разных криптосистем*).

isologous *adj* изологовый; ~ **segments** изологовые сегменты (*различные отрезки шифртекста, которым соответствует идентичный или почти идентичный открытый текст*).

isologue *n* = isolog.

issuer *n* = source.

ITAR *abbr* ЗКЭВ (*см. International Traffic in Arms Regulations*).

iterate *v* выполнять итеративно; ~ **an operation disguising an easy knapsack** выполнять операцию сокрытия тривиального вектора ранца итеративно.

iterated *adj* итеративный, итерационный; ~ **cipher** итеративный шифр (*с многократным выполнением некоторых операций процесса шифрования*); ~ **cryptosystem** итеративная криптосистема (*с многократным выполнением некоторых криптографических операций*).

I-terated *adj* I-итеративный; ~ **knapsack cryptosystem** I-итеративная ранцевая криптосистема (*ранцевая криптосистема, в которой для сокрытия тривиального вектора ранца используются I операций умножения по модулю*).

iteration *n* итерация.

iterative *adj* итеративный, итерационный; ~ **cipher** = product cipher.

IV *abbr* ВНЗ (*см. initialization vector*).

Ivybells *prop* "Вьюнок" (*кодовое название радиоразведывательной операции, в ходе которой в конце 70-х годов АНБ США удалось подключиться к советскому коммуникационному кабелю, проложенному по дну Охотского моря для поддержания связи между материком и полуостровом Камчатка*).

J

Japanese *adj* японский; ~ **Navy code № 25** Японский военно-морской код № 25 (*вскрытие этого кода американцами помогло им провести ряд успешных военных операций на Тихом океане во время второй мировой войны*).

Jefferson *prop* Джефферсон Т. (*третий по счету президент США*); ~ **cipher wheel** дисковый шифр Джефферсона [шифровальное колесо Джефферсона].

Jevons *prop* Джевонс В. (*английский математик*); ~**'s number** число Джевонса (*в своей книге "Трактат о логическом и научном методе", опубликованной в 1870 г., Джевонс заявил о том, что вряд ли кому-то, кроме него самого, удастся разложить на множители число 8616460799*).

JN-25 *abbr* ЯВ-25 (*см. Japanese Navy code № 25*).

joy-riding *n* использование побочных явлений для доступа к информации.

K

k *abbr* к (*см. key*); ~**-64** = key of length 64.

KAC *abbr* ЦАК (*см. key authentication center*).

Kahn *prop* Кан Д. (*американский писатель, автор фундаментальной монографии по истории криптологии под названием "Взломщики кодов"*).

KAK *abbr* КАК (*см. key-autokey*).

kappa *n* к (*11-я буква греческого алфавита*); ~**-test** к-тест.

Kasiski *prop* Казиский Ф. (*немецкий криптолог*); ~ **method** метод Казиского; ~ **test** тест Казиского.

KCA *abbr* ПСК (*см. key certification authority*).

KCC *abbr* ЦСК (*см. key certification center*).

KD *abbr* (1) СК (*см. key directory*); (2) КР (*см. decryption key*).

KDC *abbr* ЦРК (*см. key distribution center*).

KDP *abbr* ПРК (*см. key distribution protocol*).

KDS *abbr* СРК (*см. key distribution system*).

KE *abbr* КШ (*см. encryption key*).

KEA *abbr* АОК (*см. key exchange algorithm*).

keeper *n* держатель, хранитель (*напр., ключа*).

KEK *abbr* КШК (*см. key-encrypting key*).

Kerckhoff *prop* Керкхофф А. (*голландский криптолог*); ~ **assumption** допущение Керкхоффа (*весь механизм шифрования, кроме значения секретного ключа, известен криптоаналитику противника*); ~ **precept** правило Керкхоффа (*стойкость шифра должна определяться только секретностью ключа*); ~ **requirements** требования Керкхоффа (*шесть условий, которым должна удовлетворять универсальная криптосистема*).

KES *abbr* КДК (*см. key escrow system*).

key *n* (*криптографический*) ключ; ~ **agreement protocol** протокол согласования ключа (*одноключевой криптосистемы*); ~ **assignment** назначение [распределение] ключей; ~ **authentication center** центр аутентификации ключей; ~**-autokey** ключ-автоключ; ~ **backup** создание резервных копий ключей; ~ **bank** банк ключей; ~ **bit** бит ключа [ключевой последовательности]; ~**book** книга с ключами; ~**-breaking efforts** усилия по вскрытию ключа; ~ **card** ключевая перфокарта (*бумажная карточка с отверстиями, используемая в качестве носителя ключевой информации*); ~ **carrier** (1) ключевой носитель; устройство транспортировки ключей; (2) владелец [держатель] ключа; ~ **certification** сертификация (*заверение подлинности*) ключей; ~ **certification authority** полномочие на сертификацию ключей; ~ **certification center** центр сертификации ключей; ~ **change** смена [замена] ключа; ~ **change** ~ клавиша смены [замены] ключа; ~ **character** знак ключа; ~ **clustering** группирование ключей (*при криптоанализе*); ~ **compromise** компрометация ключа; ~**-controlled algorithm** управляемый ключом алгоритм (*шифрования*); ~**-cracking computer** компьютер для вскрытия ключей; ~ **creation function** = ~ generation function; ~ **cycle period** = ~ period; ~ **delivery** доставка [транспортировка] ключей; ~ **dependency** (1) зависимость от ключа (*зависимость битов шифртекста от битов ключа*); (2) взаимозависимость ключей; ~**-dependent cryptoalgorithm**

зависящий от ключа алгоритм шифрования; ~ **depth** мощность ключевого множества (*количество ключей*); ~ **diffusion** рассеивание ключа (*распространение влияния ключа на множество знаков шифртекста*); ~ **digits** цифры ключа; ~ **directory** справочник ключей; ~ **disclosure** раскрытие ключа; ~ **disk** (1) диск [дискета] для хранения ключей; (2) ключевой диск [ключевая дискета]; ~ **distribution** распределение [доставка] ключей; ~ **distribution center** центр распределения ключей; ~ **distribution information** информация службы распределения ключей; ~ **distribution mechanism** механизм распределения ключей; ~ **distribution node** узловой центр распределения ключей; ~ **distribution protocol** протокол распределения ключей; ~ **distribution system** система распределения ключей; ~ **encryption** шифрование ключей; ~**-encrypting key** ключ шифрования (*других*) ключей [главный ключ]; ~ **encryption** ~ = ~-encrypting ~; ~ **entropy** энтропия ключей; ~ **entry procedure** процедура ввода ключа; ~ **escrow** депонирование ключей (*передача ключей на хранение третьим лицам*); ~ **escrow system** криптосистема с депонированием ключей; ~ **erasure** стирание ключа (*записанного в памяти*); ~ **exchange** обмен ключами; ~ **exchange algorithm** алгоритм обмена ключами; ~ **exchange system** (1) система обмена ключами; (2) система обмена информацией для выработки ключа; ~ **exchange via third party** ключевой обмен с участием третьей стороны; ~ **exhaustion attack** атака путем перебора всех возможных вариантов ключа; ~ **expansion** увеличение длины ключа; ~ **exposure** раскрытие [разглашение] ключа; ~ **extraction** извлечение [выделение] ключа (*напр., из памяти компьютера или шифратора*); ~ **field** поле ключа (*в сообщении*); ~ **fill device** = ~ load device; ~ **format** формат ключа; ~ **generation** генерация [порождение] ключей; ~ **generation function** функция генерации [порождения] ключа; ~ **generation unit** устройство генерации ключей; ~ **generator** генератор ключей; ~**-guessing attack** атака с угадыванием ключа; ~**-guessing attack secure** защищенный от атаки с угадыванием ключа; ~**-handling procedure** процедура работы с ключом; ~ **hashing** хэширование ключа (*преобразование исходного ключа в более короткий, на котором и производится шифрование*); ~ **hierarchy** иерархия ключей; ~ **identification number** идентификационный номер ключа; ~ **information** ключевая информация; ~ **injection** ввод ключей; ~ **injector** устройство ввода ключей; ~ **input algorithm** алгоритм ввода ключа; ~ **insertion** ввод ключа; ~ **insertion set** комплект [набор] (*устройств*) для ввода ключа; ~ **installation** установка [инсталляция] ключа; ~ **installator** лицо, осуществляющее установку [инсталляцию] ключа; ~ **integrity** целостность ключа; ~**-**~ **pair** = ~-pair; ~**-length** длина ключа; ~**less cipher** бесключевой шифр; ~ **library** библиотека ключей (*собрание ключей одного пользователя или группы пользователей*); ~ **life** продолжительность действия ключа; ~ **list** список [перечень] ключей (*используемых в течение определенного периода времени*); ~ **load** загрузка [ввод] ключей; ~ **load button** кнопка [клавиша] загрузки [ввода] ключа; ~ **load device** устройство загрузки [ввода] ключей (*предназначено для хранения электронного ключа с целью его последующей установки в криптографическое оборудование*); ~ **loader** = ~ load device;

~ **load procedure** процедура загрузки [ввода] ключа; ~ **management** управление ключами; ~ **management center** центр управления ключами; ~ **management facilities** средства управления ключами; ~ **management information** информация службы управления ключами; ~ **management mechanism** механизм управления ключами; ~ **management procedure** процедура управления ключами; ~ **management protocol** протокол управления ключами; ~ **management system** система управления ключами; ~ **manager** администратор [службы] управления ключами; ~ **matrix** матрица ключей [ключевая матрица] (*матрица, используемая в качестве ключа*); ~ **memory** память ключей; ~**-minimal cipher** шифр с ключом минимальной длины [с минимальным числом ключей]; ~ **negotiation** переговоры о выработке (*общего*) ключа [ключевой обмен]; ~ **notarization** нотаризация ключей; ~ **notarization system** система с нотаризацией ключей; ~ **of directory** ключ, содержащийся в справочнике; ~ **of length N** ключ длины N; ~ **pair** пара ключей (*зашифрования и расшифрования*); ~ **parameter** ключевой параметр [параметр ключа]; ~ **period** период (*повторения*) ключа; ~ **phrase** ключевая фраза (*используется в качестве ключа или для формирования ключа*); ~ **predistribution scheme** схема предварительного распределения ключей; ~ **processor** ключевой процессор; ~ **producing unit** = ~ generation unit; ~ **production key** ключ генерации ключей (*используется для инициализации генератора ключевой последовательности*); ~ **program** программа генерации ключей; ~**'s properties** свойства ключа; ~ **protection** защита ключей; ~ **purchase attack** атака с использованием купленного (*напр., у курьера*) ключа; ~ **receiver** получатель ключа; ~ **recovery** восстановление ключей; ~ **recovery database** база данных, используемая для восстановления ключей; ~ **recovery system** система восстановления ключей; ~ **redundancy** избыточность ключа; ~ **register** регистр для хранения ключа; ~**-related information** = keying information; ~ **reload** повторная загрузка [повторный ввод] ключа; ~ **reload procedure** процедура повторной загрузки [повторного ввода] ключа; ~ **reloading** = ~ reload; ~ **repetition** повторное использование ключа; ~ **repository** устройство [прибор] для хранения ключей; ~ **restoration** восстановление ключа; ~ **retrieval** поиск ключа; ~ **safeguarding** = ~ protection; ~ **schedule** (1) ключ-расписание (*в DES-алгоритме*); (2) назначение [смена] ключей; ~**-schedule cryptanalysis** вскрытие ключ-расписания; ~ **search** поиск (*неизвестного*) ключа; ~ **secrecy** секретность ключа; ~ **security** защищенность ключа; ~ **seed** начальное состояние генератора ключей; ~**-selective network** сеть с выбором ключа (*для различных видов связи*) [сеть с избирательностью по ключу]; ~ **sequence** ключевая последовательность; ~ **sequence period** период ключевой последовательности; ~ **server** сервер ключей; ~ **set** множество [набор] ключей; ~ **setting** установка [набор] ключа; ~ **setup** подготовка ключей; ~ **sharing** групповое [коллективное] пользование ключом; ~ **signature** подпись, удостоверяющая ключ [подпись под ключом]; ~ **size** размер [длина] ключа; ~ **source** источник ключа; ~ **space** ключевое пространство (*количество различных ключей для данного шифра*); ~ **space exploration** поиск в пространстве ключей; ~ **splitting** деление ключа на части (*ключ делится на составные*

части, которые распределяются между несколькими лицами так, что ключ может быть восстановлен в полном объеме только в том случае, если каждый из них предоставит для этого свою часть ключа); ~ **starting point** исходная [отсчетная] точка (*используемой*) ключевой последовательности; ~ **storage** хранение ключей (*в памяти компьютера или шифратора*); ~ **storage unit** устройство хранения ключей; ~ **stream** ключевая последовательность [ключевой поток]; ~ **stream generation** генерация [порождение] ключевой последовательности [ключевого потока]; ~ **stream generator** генератор ключевой последовательности [ключевого потока]; ~ **stream period** период ключевой последовательности [ключевого потока]; ~ **string** ключевая строка [строка-ключ]; ~ **structure** структура ключа; ~ **system** ключевая система; ~ **table** таблица ключей [ключевая таблица]; ~ **tag** ключевая бирка (*дополнительная идентификационная информация, которой снабжаются электронные ключи*); ~ **tape** ключевая лента (*перфорированная или магнитная лента, используемая в качестве носителя ключевой информации*); ~**text** текстовая ключевая информация [представление ключа в виде текста]; ~ **theft** кража ключей; ~ **to a cipher** ключ к шифру; ~ **transfer device** устройство передачи ключей; ~ **transfer protocol** протокол передачи ключей; ~ **transport** доставка [транспортировка] ключей; ~ **transport device** устройство транспортировки [доставки] ключей; ~ **trial** проба ключа (*при криптоанализе методом проб и ошибок*); ~ **type** тип ключа; ~ **updating** обновление ключа (*необратимый процесс смены ключа в криптографическом оборудовании*); ~ **variable** ключевая переменная; ~ **variables generator** = ~ stream generator; ~ **wheel** ключевое колесо (*шифровальной машины*); ~**word** ключевое слово; ~**word mixed sequences** смешанные с использованием ключевого слова последовательности; (2) клавиша (*клавиатуры*).

keyed *adj* с ключом; ~ **cryptoalgorithm** алгоритм шифрования с ключом.

Key-Gen *abbr* = key generator (*генератор ключей в криптосхеме РША*).

keying *adj* ключевой; ~ **information** ключевая информация; ~ **material** ключевой материал; ~ **variable** ключевая переменная.

keying *n* ввод ключа.

KG *abbr* ГК (*см. key generator*).

KGB *abbr* КГБ (*Комитет Государственной Безопасности*).

KID *abbr* ИНК (*см. key identification number*).

kiss *n* "поцелуй" (*совпадение открытых текстов двух криптограмм, полученных с помощью различных шифров*).

KK *abbr* КК (*см. key-encrypting key*).

KMC *abbr* ЦУК (*см. key management center*).

KMF *abbr* СУК (*см. key management facility*).

KMP *abbr* ПУК (*см. key management protocol*).

KMS *abbr* СУК (*см. key management system*).

KN *abbr* = knapsack.

knapsack *n* (1) рюкзак, ранец; ~**-based cryptosystem** = ~ cryptosystem; ~ **cryptosystem** ранцевая [рюкзачная] криптосистема; ~ **problem** задача об укладке ранца [о рюкзаке]; (2) вектор ранца.

known-plaintext *adj* на основе знания открытого текста; ~ **attack** атака на основе знания открытого текста; ~ **attack secure** защищенный от атаки на основе знания открытого текста.

known-to-be-ASCII *adj* на основе знания того, что открытый текст имеет кодировку ASCII; ~ **attack** атака на основе знания того, что открытый текст имеет кодировку ASCII.

KNS *abbr* СНК (*см. key notarization system*).

KP *abbr* КП (*см. key processor*).

KPK *abbr* КГК (*см. key production key*).

KPS *abbr* СПК (*см. key predistribution scheme*).

krypto- = crypto-.

KS *abbr* СК (*см. session key*).

KSU *abbr* УХК (*см. key storage unit*).

KVG *abbr* = key variable generator.

L

language *n* язык; ~ **characteristics** характерные черты [особенности] языка.

large *adj* большой; ~ **integer factorization** разложение большого целого числа на множители; ~ **key space** большое ключевое пространство.

latent *adj* скрытый; ~ **repetition** скрытое повторение (*в шифртексте*).

Latin *adj* латинский; ~ **square** латинский квадрат (*матрица размера nxn из n символов, в которой любой столбец и любая строка не содержат повторений*).

lattice *n* решетка; ~ **basis reduction** (*криптоаналитический*) метод редукции базисов решеток; ~ **with basis** решетка с базисом.

law *n* закон; ~ **enforcement access field** поле данных правоохранительных органов (*в алгоритме шифрования "Скипджек"*).

layer *n* (1) уровень, слой; (2) разрез (*операция или преобразование в блочном шифре, выполняемая по всей длине блока*).

LCG *abbr* ЛКГ (*см. linear congruential generator*).

LCM *abbr* НОК (*см. least common multiple*).

LEAF *abbr* ПДПО (*см. law enforcement access field*).

leak *v* просачиваться; ~ **into a transmission line as a side-channel** просачиваться в линию связи, превращая ее в побочный канал.

least *adj* (1) *в грам. знач. сущ.* самое меньшее количество; **at** ~ по меньшей [крайней] мере; (2) наименьший; самый младший; ~ **common multiple** наименьшее общее кратное; ~ **significant bit** самый младший бит (*в двоичном представлении числа*); ~ **square estimation** оценка по методу наименьших квадратов.

legitimate *n* законный; ~ **user** законный пользователь.

length *n* длина; ~ **of a key** длина ключа; ~ **of a shift register** длина регистра сдвига.

lengthen *v* удлинять, увеличивать длину; ~ **a cryptographic key to 256 bits** увеличивать длину криптографического ключа до 256 бит.

letter *n* буква; **N-~ alphabet** алфавит из N букв; ~**-by-**~ побуквенное шифрование (*открытого текста*); ~ **frequency count** подсчет частоты встречаемости букв (*в тексте*); ~**-frequency distribution** распределение частот встречаемости [появления] букв (*в тексте*); ~**-pattern cryptanalysis** криптоанализ на основе изучения сочетания букв в тексте.

level *n* уровень; ~ **of authorized access to information** уровень санкционированного доступа к информации.

LFSR *abbr* РСЛОС (*см. linear feedback shift register*).

license *n* лицензия (*разрешение, выдаваемое государственными органами на право продажи и предоставления услуг*).

life *n* продолжительность действия (*напр., ключа*); ~ **cycle of a key** "жизненный цикл" ключа.

limited *adj* укороченный, имеющий ограниченную длину; ~ **key** ключ ограниченной длины [укороченный ключ].

line *n* линия (*связи*); ~ **eavesdropping** подслушивание на линии (*связи*); ~ **encryption device** линейный шифратор; ~ **level encryption** шифрование на уровне линии (*связи*); ~**-tap** подключение [устройств перехвата] к линии (*связи*); ~**-tapper** перехватчик, подключающийся к линии (*связи*); ~**tapping** = ~-tap.

linear *adj* линейный; ~ **approximation of block ciphers** линейная аппроксимация блочных шифраторов; ~ **attack** линейная атака; ~ **cipher** линейный шифр; ~ **combination** линейная комбинация; ~ **complexity** линейная сложность ключевого потока (*длина самого короткого регистра сдвига с линейной обратной связью*); ~ **complexity profile** профиль линейной сложности (*ключевого потока*); ~ **congruential generator** линейный конгруэнтный генератор; ~ **consistency test** тест на линейную согласованность (*ключевого потока*); ~ **cryptanalysis** линейный криптоанализ (*для описания поведения шифров использует линейные приближения*); ~ **distribution** линейное распределение; ~ **feedback shift register** регистр сдвига с линейной обратной связью; ~ **mapping**

линейное отображение; **~-resistant function** линейно-стойкая функция (*стойкая против линейного криптоанализа*); **~ recurrence** линейная рекуррента; **~ span** = ~ complexity; **~ transformations based cipher** шифр на основе линейных преобразований.

linearity *n* линейность.

linearization *n* линеаризация.

linearize *v* линеаризовать; **~ a system of non-linear equations** линеаризовать систему нелинейных уравнений.

linearly *adv* линейно; **~ dependent** линейно зависимый.

linguistic *adj* лингвистический; **~ attack** лингвистическая атака (*разновидность криптоаналитической атаки*).

link *n* канал; **~-by-~ encryption** поканальное шифрование (*применение шифрования в каждом канале, соединяющем два узла сети связи, которые могут быть промежуточными на пути от отправителя к получателю сообщения*); **~ encryption** канальное шифрование (*засекречивание передач по каналам связи с помощью шифрования*); **~ encryption device** = ~ encryptor; **~ encryption key** ключ канального шифрования; **~ encryptor** канальный шифратор; **~ key** ключ для отдельного канала (*связи*); **~-to-~ encryption** = ~-by-~ encryption.

lipogram *n* липограмма (*содержательный текст, состоящий из слов, в которых не встречается одна или несколько букв алфавита*).

lipstick *n жарг.* = cryptoignition key.

list *n* список, перечень.

listening *adj* перехватывающий, подслушивающий; **~-in** = listening; **~ post** пост [пункт / станция] перехвата [подслушивания]; **~ station** = ~ post.

literal *adj* буквенный; **~ key** буквенный ключ.

LMK *abbr* ЛГК (*см. local master key*).

load *n* загрузка, ввод.

load *v* загружать, вводить; **~ a key** загружать [вводить] ключ.

loading *n* = load.

lobster *n* "омар" (*состояние немецкой шифровальной машины "Энигма", в котором все ее роторы поворачиваются вместе*).

local *adj* локальный, местный; **~ key** локальный ключ (*предназначен для отдельного / конкретного устройства шифрования*); **~ key load** местная загрузка [местный ввод] ключей; **~ master key** локальный главный ключ [мастер-ключ]; **~ randomness** локальная случайность (*случайность короткой подпоследовательности, являющейся частью очень длинной последовательности*).

locally *adv* локально; **~-assigned key** локально назначаемый ключ.

locate *v* определять местоположение; ~ **a transmitter** определять местоположение передатчика.

logarithm *n* логарифм.

logic *n* логика (*наука*); ~ **bomb** логическая бомба (*тайно встроенный в программу код, предназначенный для несанкционированного доступа к открытым текстам, ключам и т. д.*).

long *adj* большой (*по длине*), длинный; ~ **key cipher** шифр с длинным ключом; ~**-lived key** = ~-term key; ~ **period** большой период (*рекуррентной последовательности*); ~**-term key** долговременный ключ.

look into *v* исследовать; ~ **a cipher** исследовать шифр.

lost *adj* утраченный, потерянный; ~ **key** утраченный [потерянный] ключ.

low *adj* низкий, малый; ~ **density attack** попытка вскрытия по линии малой плотности; ~ **frequency letter** редко встречающаяся буква; ~ **grade cipher** шифр низкой стойкости [легко вскрываемый шифр]; ~**-speed encryption** низкоскоростное шифрование.

lower *adj* нижний; ~ **bound** нижняя граница.

lowest *superl* наименьший; ~ **frequency letter** буква с наименьшей частотой встречаемости (*в тексте*).

ltr *abbr* бкв (*см. letter*).

Lucifer *prop* "Люцифер" (*название алгоритма шифрования, положенного в основу DES-алгоритма*).

lug *n* рейтер.

M

MAC *abbr* КАС (*см. message authenticating code*).

machine *n* (1) (*шифровальная*) машина, шифратор; ~ **cryptography** машинная криптография; (2) вычислительная машина, компьютер.

MAC-key *abbr* КАС-ключ (*см. message authenticating code key*).

Magic *prop* "Магия" (*условное наименование источника разведывательной информации, которую американские криптоаналитики добывали путем дешифрования японской шифрпереписки в 30-е—40-е годы*); ~ **intercepts** перехваты "Магии".

maintenance *n* эксплуатация; ~ **key** эксплуатационный ключ (*используется для проведения профилактических работ с криптографическим оборудованием*).

major *adj* главный, основной; ~ **key** главный [основной] ключ.

Makarov *prop* Макаров В. (*сотрудник 16 управления КГБ, с 1985 по 1987 г. работавший на английскую разведку*).

make *v* (1) создавать; ~ **a code** создавать код; (2) подвергать; ~ **an attack on a cipher** подвергать шифр криптоанализу.

man-in-the-middle *n* = meet-in-the-middle.

manipulate *v* = rig.

manual *adj* ручной; ~ **aperiodic polyalphabetic substitution cipher** ручной апериодичный шифр многоалфавитной замены; ~ **cryptosystem** ручная криптосистема (*все операции по зашифрованию и расшифрованию выполняются вручную*); ~ **encryption** ручное шифрование; ~ **key distribution** распределение ключей вручную; ~ **remote rekeying** ручной удаленный ввод ключа (*процедура, применяемая для смены ключа в расположенном на удалении криптографическом оборудовании и требующая участия в ней местного обслуживающего персонала*).

manual *n* руководство; ~ **of cryptography** руководство по криптографии.

manufacturer *n* изготовитель, производитель; ~**-installed key** ключ изготовителя [производителя] (*шифраппаратуры*); ~**'s key** = ~-installed key.

map *n* географическая карта; ~ **cipher** картографический шифр (*разновидность шифра замены, в котором в качестве букв шифралфавита используются символы, обычно применяемые для условных обозначений на географических картах*).

map *v* отображать, устанавливать соответствие.

mapping *n* отображение.

market *n* рынок (*сбыта*); ~ **for cryptography** рынок криптографических средств.

Markov *prop* Марков А. (*русский математик*); ~ **cipher** шифр Маркова.

masking *adj* маскирующий; ~ **cipher** маскирующий шифр.

Maslov *prop* Маслов И. (*начальник 16 управления КГБ*).

masquerade *n* маскировка, подмена (*поведение нарушителя, пытающегося выдать себя за законного пользователя*); ~ **attack** атака с маскировкой [подменой].

masquerade *v* маскироваться, выдавать себя; ~ **as a legitimate user** выдавать себя за легального пользователя.

masquerader *n* имитатор действий законного пользователя.

Massey *prop* Месси Дж. (*американский криптолог*).

master *adj* главный; ~ **crypto-ignition key** главный ключ инициирования криптографических операций (*электронное устройство для добавления новых ключей в набор ключей, используемых для инициирования криптографических операций*); ~ **key** мастер-ключ, главный ключ; ~ **mode** режим работы с использованием мастер-ключа [главного ключа].

matched *adj* = matching.

matching *adj* согласованный; ~ **key** согласованный ключ (*напр., ключ расшифрования, соответствующий ключу шифрования*); ~ **key pair** пара согласованных ключей (*для зашифрования и расшифрования сообщений*).

matching *n* согласование, приведение в соответствие (*ключей*).

mathematical *adj* математический; ~ **cryptanalysis** математический криптоанализ; ~ **cryptography** математическая криптография (*для шифрования сообщений использует сложные математические операции, напр., такие как возведение очень больших чисел в чрезвычайно высокие степени по модулю произведения двух простых чисел*).

mathematically *adv* математически; ~**-related keys** математически связанные ключи.

matrix *n* матрица; ~ **cipher** матричный шифр; ~ **key distribution scheme** матричная схема распределения ключей.

maximal *adj* максимальный; ~ **length sequence** последовательность максимальной длины; ~ **length shift register** регистр сдвига максимальной длины; ~ **period generator** генератор псевдослучайной последовательности с максимальным периодом.

maximally *adv* максимально; ~ **correlation-immune combiner** максимально корреляционно-стойкий смеситель.

maximum *n в грам. знач. прил.* максимальный; ~ **order complexity of a pseudo-random sequence** сложность максимального порядка псевдослучайной последовательности; ~ **a posteriori probability estimation** оценка по методу максимальной апостериорной вероятности; ~ **likelihood detector** критерий [детектор] максимального правдоподобия.

Mbit *abbr* Мбит (*см. megabit*).

Mbyte *abbr* Мбайт (*см. megabyte*).

MD *abbr* (*Message Digest*) алгоритмы сжатия [хэширования] данных Ривеста (*созданная американским криптологом Р. Ривестом серия алгоритмов, которые произвольному набору данных ставят в соответствие некоторое число, при этом вероятность повторного получения точно такого же числа для другого набора данных очень мала*).

meaningful *adj* содержательный (*имеющий смысл*); ~ **decryption** дешифрование, дающее содержательный текст.

mechanism *n* (1) механизм (*физическое устройство*); (2) механизм (*логическая концепция машины, которая может быть реализована на практике как физическое устройство или как последовательность логических команд, выполняемых физическим устройством*).

mechanistic *adj* механический; ~ **cryptography** механическая криптография (*для шифрования сообщений использует машины или механизмы, собранные из*

многих относительно простых компонентов и узлов, которые в совокупности реализуют очень сложное шифрующее преобразование).

medium-speed *adj* среднескоростной; ~ **encryptor** среднескоростной шифратор.

meet-in-the-middle *n* сведение к середине; ~ **attack** атака методом сведения к середине (*предложенный американскими криптологами Р. Мерклем и М. Хеллманом метод вскрытия суперпозиций шифров*).

mega- *в сложных и сложносоставных словах имеет значение* мега- (*превышение основной единицы измерения в миллион раз*).

megabit *n* мегабит; **N ~ data encryption** шифрование данных со скоростью N мегабит в секунду.

megabyte *n* мегабайт; **N ~ data encryption** шифрование данных со скоростью N мегабайт в секунду.

memorize *v* помещать в память [в запоминающее устройство] (*компьютера или шифратора*); ~ **a key** помещать ключ в память [в запоминающее устройство].

memorized *adj* запоминаемый, хранимый в памяти [в запоминающем устройстве] (*компьютера или шифратора*); ~ **key** запоминаемый [хранимый в памяти] ключ.

memory *n* память (*компьютера*); ~ **complexity** сложность по памяти; ~ **requirement** требования по памяти (*к компьютеру со стороны алгоритма*).

MERCURY *prop* = CHALET.

Merkle *prop* Меркль Р. (*американский криптолог, один из авторов метода "сведения к середине"*); ~**'s channels** каналы Меркля (*множество открытых каналов связи — телефонных, телексных, радиовещательных, телевизионных, почтовых и т. д.*); ~**'s puzzle** "шарада" Меркля.

Mersenne *prop* Мерсен М. (*французский физик*); ~ **prime** простое число Мерсена (*простое число вида* 2^n-1).

message *n* сообщение, донесение, послание; ~ **authenticating code** код аутентификации сообщения; ~ **authenticating code key** ключ с кодом аутентификации сообщения; ~ **digest** сжатие [хэширование] сообщения; ~ **entropy** энтропия сообщений; ~ **exhaustion attack** атака путем перебора всех возможных вариантов сообщения; ~ **expansion** расширение сообщения (*при зашифровании*); ~ **format** формат сообщения; ~ **indicator** индикатор сообщения (*битовый набор, передаваемый по каналу связи для синхронизации работы криптографического оборудования*); ~ **key** разовый ключ; ~ **keying element** элемент ключа, изменяемый для каждого сообщения; ~ **preparation for encryption** подготовка сообщения к зашифрованию; ~ **reordering attack** = ordering attack; ~ **residue class** класс вычетов сообщений; ~ **Security Protocol** Прото-

кол безопасности сообщений; ~ **space** пространство сообщений; ~ **substitution** подмена сообщений (*тип угрозы в теории аутентификации информации*); ~ **wiretapping** перехват сообщений с подключением к проводной линии связи [перехват сообщений по отводному каналу].

messaging *n* обмен сообщениями, передача сообщений.

meteorological *adj* метеорологический; ~ **code** метеорологический код (*применяется специально для кодирования сводок и прогнозов погоды*).

method *n* метод; ~ **of encipherment** метод шифрования; ~ **of encryption** = ~ of encipherment.

MH *abbr* Меркль-Хеллман (*Р.Меркль и М.Хеллман — американские криптологи*); ~ **knapsack cryptosystem** ранцевая криптосистема Меркля-Хеллмана.

MIC *abbr* = message integrity check.

microdot *n* микроточка (*многократно уменьшенная фотокопия документа*).

microprocessor *n* микропроцессор; ~**-based cryptosystem** криптосистема с микропроцессором (*для выполнения криптографических преобразований*).

Microsoft *prop* "Майкрософт" (*американская корпорация, крупнейший в мире производитель программного обеспечения*); ~ **Challenge Handshake Authentication Protocol** Протокол аутентификации типа вызов-рукопожатие корпорации "Майкрософт"; ~ **Point-to-Point Encryption protocol** криптографический протокол корпорации "Майкрософт" для одноранговых сетей.

microwave *adj* микроволновый (*относящийся к радиодиапазону с длиной волны менее 10 см*); ~ **receiver** микроволновый приемник.

middle средний; ~**person** = meet-in-the-middle; ~ **wheel** средний шифрдиск (*второй по счету шифрдиск в трехдисковой модификации "Энигмы"*).

military *adj* военный; ~ **cryptanalysis** военный криптоанализ.

MIMD *abbr* МКМД (*см. multiple-instruction multiple-data*); ~ **factorization** разложение на множители на ЭВМ с МКМД-архитектурой.

mimic *adj* имитирующий; ~ **resistance** имитостойкость.

mimic *v* имитировать; ~ **the action of a cipher machine** имитировать работу шифратора.

minimal *adj* минимальный; ~ **cover time** минимальное время защиты (*минимальное время, которое шифрсистема противостоит попыткам ее вскрыть для всех возможных атак против нее*); ~ **key-length** минимальная длина ключа; ~ **polynomial** минимальный полином.

minimum *n* минимум; ~ **disclosure proof** = zero knowledge proof; ~ **mean square error estimation** оценка по минимуму среднеквадратичной ошибки.

minuend cryptosystem *n* = subtractive cryptosystem.

MITM *abbr* СКС (*см. meet-in-the-middle*).

mixed *adj* смешанный; ~ **cipher alphabets** смешанные шифралфавиты; ~ **cipher alphabets recovery** восстановление смешанных алфавитов.

mixing *n* смешение; ~**-key cryptoalgorithm** криптоалгоритм смешения ключевого и информационного потока.

MK *abbr* (1) МК (*см. master key*); (2) РК (*см. message key*).

M-key *abbr* МК (*см. master key*).

ML-sequence *abbr* МД-последовательность (*см. maximal length sequence*).

MLSR *abbr* РСМД (*см. maximal length shift register*).

MPPE *abbr* КОСМ (*см. Microsoft Point-to-Point Encryption protocol*).

mnemonic *adj* легко запоминаемый; ~ **key** легко запоминаемый ключ.

mode *n* режим; ~ **of operation** режим работы (*шифратора*).

modem *n* модем; ~**-encryptor** модем с шифратором.

modern *adj* современный; ~ **cryptanalysis** современный криптоанализ.

modification *n* модификация, изменение; ~ **attack** атака с целью изменения (*содержания*) сообщения; ~ **detection encryption algorithm** алгоритм шифрования с обнаружением изменений (*в шифртексте*); ~ **of a cipher machine** модификация шифратора.

modified *adj* модифицированный, измененный; ~ **one-time key** модифицированный одноразовый ключ.

modify *v* изменять, модифицировать; ~ **messages from a legitimate transmitter** изменять сообщения законного передатчика.

modular *adj* модульный; ~ **arithmetic cipher** шифр на основе модульной арифметики; ~ **exponentiation** модульное возведение в степень.

modulator *n* модулятор.

module *n* модуль, блок.

modulo *adv* по модулю; ~ **2 adder** сумматор по модулю 2; ~**-adding** суммирование по модулю; ~**-exponentiation** возведение в степень по модулю; ~**-L Vernam system** система Вернама по модулю L (*одноразовый шифровальный блокнот*); ~ **2 multiplier** умножитель по модулю 2.

modulus *n* модуль (*числа или конгруэнтности*); ~ **of congruence** модуль конгруэнтности.

mole *n* "крот", "двойной" агент (*перевербованный сотрудник спецслужбы*).

monalphabetic *adj* = monoalphabetic.

monitor *v* наблюдать за радиообменом, вести радиоперехват; ~ **communications** наблюдать за сообщениями.

monitor *n* тот, кто ведет радиоперехват, наблюдает за радиообменом.

monitoring *n* проверка, контроль, мониторинг.

monoalphabetic *adj* моноалфавитный; ~ **cipher** моноалфавитный шифр.

monographic *adj* монографический, однобуквенный; ~ **cipher** монографический шифр (*осуществляет побуквенное преобразование открытого текста в шифрованный*); ~ **Phi test** однобуквенный ф-тест.

monome *n* одна цифра; ~**-dinome** = ~-dinome cipher system; ~**-dinome cipher system** одно-двухцифровая шифрсистема (*при зашифровании одна часть символов открытого текста заменяется на одну цифру, а другая — на две*).

most *adj* (1) *в грам. знач. сущ.* самое большое количество; **at** ~ самое большее; (2) наибольший, самый старший; ~ **significant character** самый старший знак.

most *adv* наиболее, больше всего; ~ **frequently occurring letter in a language** наиболее часто встречаемая буква языка; ~ **secret** = top secret.

mot-key *abbr* МОР-ключ (*см. modified one-time key*).

mount *v* предпринимать, организовывать; ~ **a cryptanalytic attack** предпринимать [организовывать] криптоаналитическую атаку.

MPQS *abbr* МПКР (*см. Multiple-Polynomial Quadratic Sieve*).

MS *abbr* МС (*см. Microsoft*); ~ **-CHAP** МС ПАВР (*см. Microsoft Challenge Handshake Authentication Protocol*).

m-sequence *abbr* м-последовательность (*см. maximal length sequence*).

MSG *abbr* СБЩ (*см. message*).

MSP *abbr* ПБС (*см. Message Security Protocol*).

multi- *в сложных и сложносоставных словах имеет значение* много-; мульти-.

multi-algorithmic *adj* мультиалгоритмический; ~ **encryption** мультиалгоритмическое шифрование (*с использованием нескольких алгоритмов*).

multi-channel *adj* многоканальный; ~ **encryption** многоканальное шифрование.

multi-dimensional *adj* многомерный; ~ **encryption** шифрование с многомерными преобразованиями.

multi-encryption *n* многократное шифрование (*на разных ключах*).

multi-function *adj* многофункциональный; ~ **encryptor** многофункциональный шифратор.

multi-key *adj* многоключевой; ~ **memory** память для хранения ключей различного назначения.

multi-level *adj* многоуровневый; ~ **encryption** многоуровневое шифрование; ~ **key distribution** многоуровневое распределение ключей; ~ **key management** многоуровневое управление ключами.

multiliteral *adj* многозначный, многобуквенный; ~ **cipher alphabet** многозначный шифралфавит (*служит для замены одного знака открытого текста на несколько знаков шифрованного*); ~ **cipher system** многобуквенная шифрсистема (*при зашифровании каждый символ открытого текста заменяется на несколько символов шифрованного*).

multi-nomial *adj* многочленный, полиномиальный.

multi-nomial *n* многочлен, полином.

multi-party *adj* состоящий из нескольких участников; ~ **protocol** протокол обмена сообщениями между несколькими абонентами.

multiple *adj* многократный; ~ **encryption method** метод многократного шифрования (*на различных ключах*); ~ **iterated knapsack cryptosystem** ранцевая криптосистема с многократными итерациями; ~ **mapping** многократное отображение; ~ **substitution** многократная замена; ~ **substitution cipher** шифр многократной замены.

multiple *n* кратное число.

multiple- *в сложных и сложносоставных словах имеет значение* много-, мульти-.

multiple-instruction multiple-data *adj* характеризующийся наличием многих потоков команд и многих потоков данных (*о компьютерной архитектуре*).

multiple-key *adj* многоключевой; имеющий несколько ключей; ~ **cipher** многоключевой шифр; ~ **encryption** шифрование на нескольких ключах.

multiple-polynomial *adj* мультиполиномиальный; ~ **Quadratic Sieve** квадратичное решето (*один из алгоритмов разложения целого числа на множители*).

multiple-stage *adj* многоступенчатый; ~ **encryption** многоступенчатое шифрование.

multiplexer *n* мультиплексор (*многовходовое логическое устройство, которое осуществляет выбор одного значения из нескольких, поданных на его входы*).

multiplication *n* умножение.

multiplicative *adj* мультипликативный; ~ **cipher** мультипликативный шифр; ~ **inverse element** обратный элемент по умножению; ~ **knapsack cryptosystem** мультипликативная ранцевая криптосистема.

multiprocessor *adj* мультипроцессорный, многопроцессорный.

multiprocessor *n* мультипроцессор (*компьютер, имеющий больше одного процессора*).

multiregister *adj* многорегистровый; ~ **generator** многорегистровый генератор (*ключевой / шифрующей последовательности*).

multi-sequence shift register *n* регистр сдвига, порождающий несколько последовательностей.

multisignature *n* (*цифровая*) подпись нескольких лиц [пользователей].

multistep *adj* многошаговый; ~ **key generator** многошаговый генератор ключей.

multisubstitution *abbr* = multiple substitution.

multiuser *adj* многопользовательский; ~ **cryptographic techniques** многопользовательские криптографические методы.

mutual *adj* взаимный; ~ **information** взаимная информация; ~ **key** взаимный ключ (*общий для нескольких участников сеанса связи*).

N

name *n* имя, название, наименование (*алгоритма шифрования*).

name *v* называть; ~ **the enemy ciphers after sea creatures** давать шифрам противника имена морских тварей.

named *adj амер. воен.* безномерной; ~ **key** безномерной ключ (*предназначен для отдельной криптографической операции или для конкретного узла шифратора*).

nastygram *n* мерзограмма (*сообщение, которое используется для взлома криптографического протокола*).

national *adj* национальный; ~ **Bureau of Standards** Национальное бюро стандартов (*США*); ~ **Computer Security Center** Национальный центр защиты компьютеров (*американское государственное ведомство, занимающееся внедрением защищенных компьютерных систем в правительственных учреждениях США, а также координирующее разработку и анализ таких систем*); ~ **Cryptologic Museum** Национальный музей криптологии (*США*); ~ **Institute of Standards and Technology** Национальный институт стандартов и технологии (*США*); ~ **Security Agency** Агентство национальной безопасности [Национальное агентство безопасности] (*правительственное ведомство США, занимающееся перехватом и дешифрованием сообщений*); ~ **Security Agency Scientific Advisory Board** Научно-консультационный совет Агентства национальной безопасности (*США*).

native *adj* внутренний; ~ **key** внутренний (*для криптографического устройства*) ключ.

Navaho *n* = Navajo.

Navajo *n* навахо (*североамериканские индейцы*); ~ **code** код навахо (*использование вооруженными силами США языка индейского племени навахо для засекречивания военных радиопередач во время первой и второй мировых войн*).

naval *adj* военно-морской; ~ **cipher** военно-морской шифр.

NBS *abbr* НБС (*см. National Bureau of Standards*).

NCSC *abbr* НЦЗК (*см. National Computer Security Center*).

NDC *abbr* НДК (*см. non-US diplomatic communications*).

negligible *adj* ничтожный, пренебрежимо малый; ~ **probability** пренебрежимо малая вероятность.

negotiate *v* договариваться; ~ **separate keys** договариваться об использовании различных ключей.

negotiated *adj* = agreed-upon.

network *n* сеть; ~ **attack** сетевая атака (*для вскрытия ключа или шифра используются вычислительные ресурсы распределенной компьютерной сети*); ~**-based cryptosystem** = ~ cryptosystem; ~ **cryptosystem** сетевая криптосистема; ~ **key** сетевой ключ; ~ **subscriber** абонент сети.

new *adj* новый, обновленный; ~ **DES** обновленный DES-алгоритм; ~ **directions in cryptography** "Новые направления в криптографии" (*статья американских криптологов У. Диффи и М. Хеллмана, публикация которой в 1976 г. ознаменовала наступление эры криптологии с открытыми ключами*).

NIST *abbr* НИСТ (*см. National Institute of Standards and Technology*).

NIZK *abbr* НДНР (*см. non-iterative zero-knowledge proof*).

NLFSR *abbr* РСНОС (*см. non-linear feedback shift register*).

no *adj означает запрет*; ~ **foreigners** иностранцам вход воспрещен (*применяется в АНБ США для ограничения доступа иностранных граждан в секретные зоны на американских станциях перехвата, расположенных за рубежом*).

node *n* узел (*сети связи*); ~ **encryption** шифрование в узлах (*сети связи*); ~ **key** узловой ключ; ~**-to-**~ **encryption** межузловое шифрование.

NOFORN *abbr* = no foreigners.

noise *n* шум; ~ **sequence** шумовая последовательность.

nomenclator *n* номенклатор (*гибридная криптографическая система, являющаяся наполовину кодом, наполовину шифром*).

non- *в сложных и сложносоставных словах имеет значение* не-.

non-alphanumeric *adj* небуквенно-цифровой.

non-cooperative remote rekeying = automatic remote rekeying.

non-cryptographic *adj* некриптографический.

non-deterministic *adj* недетерминированный; ~ **Turing machine** недетерминированная машина Тьюринга.

non-disclosure *n* неразглашение (*секретной информации*); невозможность вскрытия (*шифра*).

non-error expanding cipher *n* шифр без размножения ошибок.

non-error propagating cipher *n* = non-error expanding cipher.

non-expanding *adj* не расширяющий; ~ **encipherer** не расширяющий шифратор (*не увеличивает объем текста при шифровании*).

non-interceptible *adj* исключающий перехват; неподдающийся перехвату; ~ **means for data transmission** средства передачи данных, исключающие их перехват.

non-invertible *adj* = one-way.

non-iterative *adj* неитеративный; ~ **zero-knowledge proof** неитеративное доказательство с нулевым разглашением (*конфиденциальной информации*).

non-linear *adj* нелинейный; ~ **algorithm** нелинейный алгоритм; ~ **feedback** нелинейная обратная связь; ~ **feedback shift register** регистр сдвига с нелинейной обратной связью; ~ **mapping** нелинейное отображение; ~ **order of a function** степень нелинейности функции.

non-linearity *n* нелинейность; ~ **criteria for cryptographic functions** критерий нелинейности криптографических функций.

non-prime *adj* непростой; ~ **number** непростое число.

non-recallable *adj* неизвлекаемый (*из криптографического устройства*); ~ **key** неизвлекаемый (*из криптографического устройства*) ключ.

non-redundant *adj* безызбыточный; ~ **source of messages** безызбыточный источник сообщений.

non-repeating *adj* = non-repetitive.

non-repetitive *adj* неповторяющийся; ~ **cipher** неповторяющийся шифр; ~ **key** неповторяющийся ключ; ~ **pseudo-random number generator** генератор неповторяющихся псевдослучайных чисел.

non-secret encryption = public-key encryption.

non-truncated *adj* неусеченный; ~ **congruential generator** неусеченный конгруэнтный генератор.

non-uniform *adj* неравномерный; ~ **distribution** неравномерное распределение.

non-US *adj* неамериканский; ~ **diplomatic communications** неамериканские дипломатические каналы связи.

non-verbal *adj* неречевой; ~ **traffic** неречевой трафик.

non-volatile *adj* хранимый в энергонезависимой памяти; ~ **key** ключ, хранимый в энергонезависимой памяти.

notarization *n* нотаризация (*заверение открытых ключей пользователя цифровой подписью администратора безопасности*).

notarized *adj* заверенный; ~ **key** заверенный ключ.

NP-complete *n* NP-полный; ~ **problem** NP-полная задача.

NP-hard *adj* NP-сложный; ~ **problem** NP-сложная задача.

NSA *abbr* АНБ [НАБ] (*см. National Security Agency*); ~ **algorithm** = ~-provided algorithm; ~ **cryptographer** криптограф из АНБ; ~ **domestic intelligence activity** разведывательная деятельность АНБ внутри США; ~ **employee** сотрудник АНБ; ~**-provided algorithm** алгоритм, предоставленный АНБ; ~ **Research Group** Исследовательская группа АНБ; ~**-sponsored** финансируемый АНБ; ~ **staff** штат служащих АНБ; ~ **supplier** поставщик АНБ.

NSASAB *abbr* НКСАНБ (*см. National Security Scientific Advisory Board*).

null *adj* пустышечный, нулевой; ~ **cipher** пустышечный шифр; ~ **hypothesis** "нулевая" [основная] гипотеза; ~ **sequence** нулевая последовательность (*двоичная последовательность из нулей*).

number *n* число, количество; ~ **of possible keys** количество возможных ключей; ~**-theoretic cryptoalgorithm** криптоалгоритм на основе теории чисел.

numeric *adj* = numerical.

numerical *adj* цифровой; ~ **key** цифровой ключ; ~ **message** цифровое сообщение.

nut *n* помешанный; ~ **file** "файл для помешанных" (*жаргонное название специального файла АНБ США, в котором хранятся сведения о шифрсистемах, представленных их авторами на рассмотрение американского правительства*).

O

OBFB *abbr* ОСВБ (*см. output block feedback*).

O-book *abbr* О-книга (*см. "Orange Book"*).

observation *n* наблюдение; ~ **Service** = B-Dienst.

occurrence *n* встречаемость; ~ **of bigrams in English** встречаемость биграмм в английском языке.

OCSP *abbr* ЛПСС (*см. on-line certificate status protocol*).

octet *n* = byte.

OFB *abbr* ОСВ (*см. output feedback*).

off-line *adj* автономный; предварительный; ~ **cipher device** = ~ encryption device; ~ **cryptanalysis** автономный криптоанализ [не в реальном масштабе времени]; ~ **cryptosystem** автономная криптосистема (*шифрование и расшифрование сообщений производится независимо от их передачи по каналам связи*); ~ **encryption device** прибор предварительного [автономного] шифрования; ~ **mode** автономный режим работы.

off-network *adj* = off-line.

one- *в сложных и сложносоставных словах имеет значение* одно-.

one-key *adj* = single-key.

one-part *adj* состоящий из одной части, одночастевой; ~ **code** одночастевой код (*код, в котором элементы открытого текста и соответствующие им кодовые группы упорядочены по алфавиту или каким-либо другим образом, чтобы их перечисление могло служить одновременно и для кодирования, и для раскодирования*).

one-time *adj* одноразовый; ~ **cipher** одноразовый шифр; ~ **cryptosystem** одноразовая криптосистема; ~ **ecryption** одноразовое шифрование; ~ **hash** = message digest; ~ **key** одноразовый ключ; ~ **key encryption** шифрование на ключе одноразового использования; ~ **pad** одноразовый шифровальный блокнот; ~ **signature scheme** схема одноразовой цифровой подписи; ~ **tape** одноразовая перфолента (*перфорированная бумажная лента, используемая в качестве носителя ключевой информации и предназначенная для однократного применения*).

one-to-many *adj* неоднозначный, многозначный; ~ **mapping** неоднозначное [многозначное] отображение.

one-to-one *adj* взаимно однозначный; ~ **mapping** взаимно однозначное отображение; ~ **substitution cipher** шифр простой замены.

one-way *adj* односторонний; ~ **cipher** односторонний шифр (*шифр на основе односторонней функции*); ~ **function** односторонняя функция; ~ **hash function** односторонняя хэш-функция [функция хэширования] (*вычислительно невозможно найти два разных аргумента, для которых ее значения совпадают*); ~ **key** ключ только для зашифрования или расшифрования.

one-word *adj* состоящий из одного слова; ~ **key** ключ из одного слова [ключ-слово]; ~ **key encryption** шифрование на ключе из одного слова.

on-line *adj* (*работающий*) в реальном масштабе времени [в темпе поступления информации]; оперативный; линейный; ~ **certificate status protocol** линейный протокол статуса сертификата; ~ **cipher device** = ~ encryption device; ~ **cryptanalysis** криптоанализ в реальном масштабе времени; ~ **cryptography configuration** конфигурация, предусматривающая одновременное включение криптографического оборудования на обоих концах канала связи; ~ **cryptosystem** линейная криптосистема (*шифрование и расшифрование сообщений производится согласованно с их передачей по каналам связи*); ~ **encryption device** устройство линейного [оперативного] шифрования; ~ **key management** управление ключами в реальном масштабе времени.

on-the-air *adv* по радио; ~ **key load** загрузка [ввод] ключей по радио.

on-the-fly *adj* оперативный, линейный; ~ **encryption** оперативное [линейное] шифрование.

on the fly *adv* на лету, оперативно.

operation *n* операция; ~ **of decryption** операция расшифрования; ~ **of encryption** операция зашифрования.

operational *adj* (1) действующий; ~ **cipher** действующий шифр; (2) рабочий; ~ **key** рабочий ключ (*используется для шифрования служебной информации,*

передаваемой по каналам связи); (3) практический; ~ **attack** практическая атака (*криптоаналическая атака, которая может быть успешно осуществлена на практике*); ~ **cryptanalyst** криптоаналитик-практик.

operator *n* оператор.

opponent *n* противник; ~ **cryptanalyst** = enemy cryptanalyst.

optimal *adj* оптимальный.

option *n* дополнительное средство.

optional *adj* дополнительный; ~ **modification** дополнительная модификация (*одобренная АНБ США модификация криптографического оборудования, которая может производиться по требованию пользователей этого оборудования*).

orange *adj* оранжевый; ~ **Book** "Оранжевая книга" (*сокращенное название американского стандарта DoD 5200.28-STD*); ~ **machine** "оранжевая" машина (*условное наименование, данное американскими криптоаналитиками японской шифровальной машине, которая являлась разновидностью "красной" машины*).

order *n* порядок, степень; **mth** ~ **correlation immunity** корреляционная стойкость порядка m (*отсутствие статистической зависимости между выходной последовательностью и любыми m элементами входной последовательности*); **kth** ~ **product** произведение порядка k (*произведение k сомножителей*).

ordering *n* определение порядка следования, упорядочение; ~ **attack** атака с целью изменения порядка поступления информации к получателю.

original *adj* исходный, первоначальный; ~ **key** исходный [первоначальный] ключ; ~ **plaintext** исходный открытый текст.

originate *v* отправлять (*сообщение*).

originator *n* отправитель (*сообщения*); ~**'s key** ключ отправителя (*сообщения*).

OTAD *abbr* ЭРК (*см. over-the-air key distribution*).

OTAR *abbr* ЭВК (*см. over-the-air rekeying*).

OTAT *abbr* ЭПК (*см. over-the-air key transfer*).

OTK *abbr* ОРК (*см. one-time key*).

OTP *abbr* ОРШ (*см. one-time pad*).

out-of-date *adj* устаревший, недействительный; ~ **key** устаревший [недействительный] ключ (*с истекшим сроком годности*).

out-of-phase *adj* несовпадающий по фазе; ~ **autocorrelation** несовпадающая по фазе автокорреляция.

output *adj* выходной; ~ **alphabet** выходной алфавит; ~ **block feedback** обратная связь по выходному блоку (*один из режимов использования DES-алгоритма*); ~ **sequence** выходная последовательность; ~ **transformation** выходное преобразование.

output *n* выход; ~ **feedback** обратная связь по выходу (*один из режимов использования DES-алгоритма*).

outsider *n* посторонний, чужой (*человек*).

overall *adj* полный; ~ **diffusion** полная диффузия (*любой бит блока на выходе блочного шифра является функцией каждого бита соответствующего входного блока, при этом изменение одного бита на входе приводит к изменению всех выходных бит с вероятностью 0.5*).

override *n* отмена.

over-the-air *adj* эфирный; ~ **key distribution** эфирное распределение ключей (*ключ, посылаемый по каналу связи, шифруется с помощью того же самого ключа, который используется для шифрования сообщений, передаваемых по этому каналу*); ~ **key transfer** эфирная передача ключей; ~ **rekeying** эфирный ввод ключа (*процедура, применяемая для смены ключа в расположенном на удалении криптографическом оборудовании, при этом новый ключ пересылается по тому же самому каналу связи, что и сообщения, которые предполагается шифровать при помощи этого ключа*).

OWF *abbr* ОСФ (*см. one-way function*).

owner *n* владелец; ~ **key** ключ владельца (*криптографического устройства*).

P

P *abbr* (1) ОТ (*см. plaintext*); (2) П (*см. police*).

package *n* (1) блок, модуль; (2) корпус; (3) пакет (*программ*).

packet *n* пакет; ~ **encryption** пакетное шифрование [шифрование пакетов (*текста*)]; ~ **of encrypted data** пакет с шифрованными данными.

pad *n* (1) шифровальный блокнот; (2) = padding.

pad *v* дополнять холостым заполнением.

padded *adj* дополненный холостым заполнением; ~ **message** сообщение, дополненное холостым заполнением (*ничего не значащей или фальшивой информацией*).

padding *n* холостое заполнение (*ничего не значащая или фальшивая информация, добавляемая в передаваемое сообщение*); ~ **character** знак [символ]-заполнитель.

pager *n* пейджер; ~ **Identification and Message Extraction System** Система перехвата пейджинговых сообщений.

paging *adj* пейджинговый; ~ **signals** пейджинговые сигналы.

painstaking *adj* кропотливый, тщательный; ~ **encryption** кропотливое шифрование.

pair *n* пара.

pairwise *adj* парный; ~ **key** (1) парный ключ (*в криптосистеме с открытым ключом*); (2) ключ, назначенный двум пользователям (*недоступный для других пользователей*); ~ **traffic encryption keys** парный трафик ключей шифрования.

paper *adj* бумажный; ~**-tape key** ключ, записанный на бумажной перфоленте.

parameter *n* параметр.

parametrically *adv* параметрически; ~ **Extensible Algorithmic Key** Параметрически расширяемый алгоритмический ключ (*название криптосистемы*).

Park *prop* = Bletchley Park.

partial *adj* частичный; ~ **key recovery** частичное восстановление ключа.

partially *adj* частично; ~**-known plaintext cryptanalysis** криптоанализ с использованием частично известного открытого текста; ~ **linear** частично линейный.

participant *n* пользователь.

particular *adj* конкретный; ~ **key** конкретный ключ.

pass *v* передавать; ~ **in the clear** передавать (*сообщение*) открытым текстом; ~ **keying information to an enemy** передавать ключевую информацию противнику.

passive *adj* пассивный; ~ **eavesdropping** пассивный перехват (*у противника имеется возможность только наблюдать за обменом сообщениями, не оказывая на него никакого влияния*); ~ **wiretap** = ~ eavesdropping.

passphrase *n* парольная фраза, фраза-пароль (*в отличие от пароля, состоит из нескольких слов*).

password *n* пароль; ~ **encryption** шифрование паролей; ~ **encryption key** ключ шифрования паролей (*используется для дополнительного шифрования информации о пользовательских паролях в операционной системе Windows NT 4.0 американской корпорации "Майкрософт"*); ~**-protected key** защищенный паролем ключ.

patent *adj* явный, очевидный; ~ **repetition** явное повторение (*в шифртексте*).

pattern *n* комбинация, сочетание.

PB *abbr* ШП (*см. plug-board*).

PBC *abbr* СБО (*см. plaintext block chaining*).

PC *abbr* КР (*см. propagation criterion of degree k*).

PCBC *abbr* СБШР (*см. propagating cipher block chaining*).

P-box *abbr* П-блок (*см. permutation box*).

PEAK *abbr* ПРАК (*см. Parametrically Extensible Algorithmic Key*).

PEK *abbr* КШП (*см. password encryption key*).

PEM *abbr* ППС (*см. Privacy Enhanced Mail*).

pending *adj* очередной; ~ **key** очередной ключ (*предназначенный для замены действующего ключа*).

penetrate *v* (1) проникать; ~ **KDC** проникать в ЦРК; (2) вскрывать; ~ **diplomatic codes** вскрывать дипломатические коды.

penetration *n* проникновение (*успешное преодоление средств защиты системы*).

pentagram *n* пентаграмма (*пять символов текста*); ~ **count** маркировка пентаграмм; ~ **counting** = count; ~ **frequency** частота встречаемости [появления] пентаграмм (*в тексте*).

pentagraph *n* = pentagram.

per-call *adj* сеансовый (*предназначенный для одного сеанса связи*); ~ **key** сеансовый ключ.

per-connection *adj* предназначенный для одного соединения; ~ **key** ключ для одного соединения.

perfect *adj* совершенный; ~ **authenticity** совершенная аутентичность; ~ **forward secrecy** совершенная форвардная стойкость (*про криптосистему говорят, что она обладает совершенной форвардной стойкостью, если сгенерированный с ее помощью шифртекст не позволяет противнику получить какую-либо информацию о соответствующем открытом тексте, кроме, может быть, длины последнего*); ~ **nonlinear function** (*двоичная функция от n входных переменных, которая меняет свой выход с вероятностью 1/2 всякий раз, когда ее i входов, $1 \le i \le n$, изменяют свои значения на противоположные*); ~ **protection** совершенная защита; ~ **secrecy** = ~ security; ~ **security** совершенная стойкость; ~ **shuffle** совершенное тасование (*разновидность перестановки*).

perfectly *adv* совершенно; ~ **random key** совершенно случайный ключ.

period *n* период; ~ **determination** определение периода.

periodic *adj* периодический; ~ **cipher** периодический шифр; ~ **key** периодически сменяемый ключ; ~ **sequence** периодическая последовательность; ~ **shifted alphabet cipher** шифр (*замены*) с периодическим смещением алфавита; ~ **polyalphabetic substitution cipher** шифр периодической многоалфавитной замены.

periodic *n* = ~ polyalphabetic substitution cipher.

periodicity *n* периодичность; ~ **of a sequence** периодичность последовательности.

permanent *adj* постоянный; ~ **key** постоянный ключ.

permutation *n* перестановка, подстановка (*математическая операция, изменяющая порядок следования компонентов заданного вектора*); ~ **box** блок перестановок.

permute *v* переставлять, производить перестановку.

permuter *n* блок перестановок, пермутатор (*специализированный криптографический узел, используемый для изменения порядка, в котором содержимое регистра сдвига подается на вход других криптографических узлов*).

per-session *adj* сеансовый; ~ **key** сеансовый ключ.

Pers Z *n* "Отделение Z" (*название криптоаналитического бюро МИД Германии в 20-е—40-е годы*).

personal *adj* персональный; ~ **communications system** система персональной связи (*включает мобильные телефоны, пейджеры и беспроводные линии связи персональных компьютеров*); ~ **identification number** *n* персональный идентификационный номер.

Petsamo *prop* Петсамо (*город в Финляндии, в котором в июне 1941 г. финские войска захватили кодовую книгу и другие криптографические материалы, принадлежавшие советскому консульству*).

PFB *abbr* ОСО (*см. plaintext feedback*).

PGP *abbr* "ДХС" (*см. Pretty Good Privacy*).

phi *n* ϕ (*6-я буква греческого алфавита*); ~ **function** = Euler function; ~ **test** ϕ-тест.

phone *n* телефон; ~**-tapping** перехват разговоров по телефону путем подключения к телефонной линии.

phoney *adj* = phony.

phony *adj* ложный, фальшивый; ~ **public-key** фальшивый открытый ключ.

phrase *n* фраза.

phreak *n* фрик (*специалист в области фриканья*).

phreaking *n* фриканье, фрикинг (*наука и искусство несанкционированного подключения к телефонным и другим сетям*).

PHT *abbr* ППА (*см. Pseudo-Hadamard transform*).

physical *adj* физический; ~ **layer encryption** шифрование на физическом уровне (*семиуровневой модели взаимодействия открытых систем*).

physically *adv* физически; ~ **random sequence** физически случайная последовательность (*получена с помощью генератора, в основе работы которого лежит некое физическое явление*); ~ **secure key** физически защищенный ключ.

picket fence cipher *n* шифр с раздельным шифрованием четных и нечетных букв алфавита.

pin *n* штифт; штырь; контактная иголочка.

PIN *abbr* ПИН (*см. personal identification number*).

pinch *n* любой украденный у противника объект, который помогает при вскрытии вражеской криптосистемы.

pinch *v* похищать криптографические материалы (*ключи, описания шифровальных алгоритмов, схемы шифраторов*).

pinwheel *n* цевочное колесо; ~ **rotor** цевочный ротор.

PKA *abbr* АКОК (*см. public-key algorithm*).

PKCS *abbr* (1) СШОК (*см. Public-Key Cryptography Standard*); (2) КОК (*см. public-key cryptosystem*).

PKC *abbr* КОК (*см. public-key cryptosystem*).

p-key *abbr* о-ключ (*см. public-key*).

PKSD *abbr* ПУХК (*см. programmable key storage device*).

placode *abbr* = plain code.

plain *adj* незашифрованный, открытый; ~ **code** код без перешифровки; ~**-crypto changeover** переключение с открытой на шифрованную связь; ~ **language** = plaintext; ~**-to-crypto** (1) преобразование открытого текста в шифрованный; (2) = ~-crypto changeover.

plaintext *n* открытый текст; ~ **alphabet** алфавит открытого текста; ~ **attack** атака на основе знания открытого текста; ~ **attack secure** защищенный от атаки на основе знания открытого текста; ~ **block chaining** сцепление блоков открытого текста; ~ **character** знак открытого текста; ~**-ciphertext pair** пара открытый-шифрованный текст; ~**-ciphertext relation** зависимость между открытым и шифрованным текстом; ~ **cryptanalysis** криптоанализ на основе знания открытого текста; ~ **cryptanalysis secure** защищенный от криптоанализа на основе знания открытого текста; ~ **feedback** обратная связь по открытому тексту; ~ **letter** буква [буквенный знак] открытого текста; ~ **letter frequencies** частоты встречаемости букв в открытом тексте; ~ **message** открытое [незашифрованное] сообщение; ~ **recognition** распознавание открытого текста; ~ **restoration** восстановление открытого текста (*по имеющемуся шифртексту*); ~ **telegram** незашифрованная телеграмма; ~ **unit** отрезок открытого текста.

plan *v* планировать (*выбирать объекты и цели радиоразведывательной деятельности*).

planning *n* планирование (*выбор объектов и целей радиоразведывательной деятельности*).

plant *v* встраивать, внедрять, вмонтировать; ~ **collection device** встраивать [внедрять, вмонтировать] устройство для сбора информации.

Playfair *prop* Плейфейер (*английский политический деятель, член палаты лордов парламента Англии*); **the** ~ = ~ cipher; ~ **cipher** шифр Плейфейера.

plug-board *n* коммутатор, штепсельная панель (*один из основных узлов немецкой шифровальной машины "Энигма"*); ~ **connections** соединения коммутатора; ~**less Enigma** бескоммутаторная "Энигма"; ~ **settings** установки коммутатора (*в "Энигме"*).

PN *abbr* ПШ (*см. pseudo-noise*).

Poe *prop* По Э. (*американский писатель и криптолог-любитель*).

point *n* точка, пункт; ~**-of-sale encryption** шифрование для обеспечения защиты торговых автоматов; ~**-to-**~ (1) = end-to-end; (2) взаимно однозначный.

poker *n* покер (*карточная игра*); ~ **test** покерный тест (*используется для проверки случайности двоичной последовательности*).

police *adj* полицейский; ~ **channel** полицейский канал связи.

polyalphabetic *adj* многоалфавитный; ~ **substitution cipher** шифр многоалфавитной замены.

polyalphabetic *n* = polyalphabetic substitution cipher.

polycipher *n* = polyalphabetic substitution cipher.

polygram *n* полиграмма (*набор из двух и более символов*); ~ **substitution cipher** шифр полиграммной замены (*производит замену полиграмм открытого текста на полиграммы шифрованного текста*).

polygraphic *adj* полиграфический; ~ **cipher** полиграфический шифр (*осуществляет одновременное преобразование сразу нескольких букв открытого текста в шифрованный*); ~ **substitution cipher system** = полиграфический шифр замены.

polynomial *adj* полиномиальный, многочленный; ~ **time** полиномиально сложный [обладающий полиномиальной сложностью] (*алгоритм*); ~ **transformation of degree n** полиномиальное преобразование степени n.

polynomial *n* полином, многочлен; ~ **multiplication** умножение полиномов; ~ **over GF(2)** полином над полем GF(2); ~ **ring** кольцо полиномов.

polytime *abbr* = polynomial time.

poor *adj* нестойкий; ~ **key** нестойкий ключ.

poorly-selected *adj* = poor.

Porta *prop* Порта Дж. (*итальянский криптолог, который предвосхитил всех других специалистов, описав в своем трактате то, что считается вторым по значимости приемом в современном криптоанализе*); ~ **Table cipher** табличный шифр Порта.

positionally *adv* позиционно; ~ **dependent cipher** шифр, зависящий от положения знаков (*текста*).

post *n* пост, станция, пункт (*перехвата*).

post-encryptor *n* устройство оконечного шифрования.

power *n* (1) степень; ~ **function** степенная функция; (2) энергопотребление; ~ **analysis** анализ энергопотребления (*для вскрытия ключей используется информация об изменениях в энергопотреблении шифратора во время его работы*).

powerful *adj* стойкий; ~ **encryption** стойкое шифрование.

practical *adj* практический; ~ **security** практическая стойкость.

practically *adv* практически; ~ **unbreakable cipher** практически невскрываемый шифр.

pre- (*в сложных и сложносоставных словах имеет значение*) (1) заранее, предварительно; (2) предварительный; предшествующий.

preamble *n* преамбула (*специальный заголовок, предваряющий текст сообщения*).

prearranged *adj* заранее [предварительно] согласованный; ~ **key** заранее [предварительно] согласованный ключ.

pre-ciphering *n* предварительное шифрование.

precomputation *n* предварительные вычисления, предвычисления.

precomputed *adj* заранее вычисленный; ~ **trial encryption** заранее выполненное пробное шифрование.

precomputer *adj* докомпьютерный; ~ **cryptography** докомпьютерная криптография.

predetermined *adj* предварительно назначенный, заранее определенный; ~ **key** предварительно назначенный [заранее определенный] ключ.

pre-encrypt *v* шифровать предварительно.

pre-encryptor *n* устройство предварительного шифрования.

preloaded *adj* заранее [предварительно] загруженный [введенный]; ~ **key** заранее [предварительно] загруженный [введенный] ключ.

pre-loading *n* предварительный ввод (*ключа*); ввод (*ключа*) до начала работы.

prescientific *adj* донаучный; ~ **cryptology** донаучная криптология.

pretty *adv* довольно; ~ **Good Privacy** "Довольно хорошая секретность" (*программа шифрования*).

prevent *v* предотвращать; ~ **an attack** предотвращать атаку.

previous *adj* предыдущий, предшествующий; ~ **key** предыдущий ключ.

primality *n* простота (*числа*); ~ **problem** задача проверки простоты числа; ~ **test** проверка простоты числа.

primary *adj* (1) основной; ~ **key** основной ключ; (2) первичный, исходный; ~ **alphabet** первичный [исходный] алфавит.

prime *adj* простой; ~ **field GF(p)** простое поле GF(p) (*поле GF(p) называется простым, если число p — простое*); ~ **number generation** = prime generation.

prime *n* простое число; ~ **about 1000 bits in length** простое число длиной около 1000 бит; ~ **generation** генерация [порождение] простых чисел; ~ **product ring** кольцо по модулю произведения простых чисел.

primeness *n* = primality.

priming *adj* первичный, начальный; ~ **key** первичный [начальный] ключ.

primitive *adj* примитивный; ~ **element mod p** примитивный элемент по модулю p; ~ **element of the finite field GF(p)** примитивный элемент конечного поля GF(p).

printed *adj* представленный в печатном виде; ~ **key** ключ, представленный в печатном виде.

privacy *n* секретность, конфиденциальность; ~ **homomorphism** обеспечивающий секретность гомоморфизм (*функция зашифрования, которая предоставляет возможность выполнения ряда требуемых операций над открытым текстом путем выполнения соответствующих операций над шифртекстом*); ~ **Enhanced Mail** Почта повышенной секретности (*набор программных средств для обеспечения конфиденциальности сообщений, отправляемых электронной почтой*); ~ **problem** проблема сохранения тайны.

private *adj* (1) секретный, конфиденциальный; ~ **Communication Technology** Технология конфиденциальной связи (*криптографический протокол, разработанный американской корпорацией "Майкрософт" для обеспечения конфиденциальности сообщений, передаваемых по компьютерной сети "Интернет"*); ~ **exponent** секретный показатель степени; ~ **key** секретный ключ (*в криптосистеме с открытым ключом*); ~ **key cryptosystem** = secret-key cryptosystem; ~ **key distribution** распределение секретных ключей; ~ **key encryption** шифрование с помощью секретных ключей; ~ **key operation** операция с открытым ключом; (2) частный; личный; ~ **cryptographic algorithm** частный криптоалгоритм (*алгоритм, принадлежащий частному лицу или фирме*); ~ **key** личный ключ (*отдельного пользователя*); ~ **session key** личный сеансовый ключ (*отдельного пользователя*).

PRN *abbr* ПСЧ (*см. pseudo-random number*).

PRNG *abbr* ГПСЧ (*см. pseudo-random number generator*).

probabilistic *adj* вероятностный; ~ **approach** вероятностный подход; ~ **coding** вероятностное кодирование; ~ **correlation attack** вероятностная корреляционная атака; ~ **encryption** вероятностное шифрование (*открытому тексту сообщения ставится в соответствие множество шифртекстов и делается это таким образом, что получение даже частичной информации об открытом тексте этого сообщения противником, при условии наличия у него одного только шифрованного текста, является труднорешаемой задачей*); ~ **key generation** вероятностная генерация ключей; ~ **proof** вероятностное доказательство; ~ **relaxation** вероятностная релаксация (*метод вскрытия шифра многоалфавитной замены*).

probability *n* вероятность; ~ **distribution** распределение вероятностей; ~ **of successful attack** вероятность успешной атаки; ~ **of deception** вероятность обмана [введения в заблуждение].

probable *adj* вероятный; ~ **message cryptanalysis** криптоанализ с использованием вероятного сообщения; ~ **key** вероятный ключ (*при криптоанализе*); ~ **word cryptanalysis** криптоанализ с использованием вероятных слов.

problem *n* задача.

procedure *n* процедура; ~ **for key generation** процедура генерации ключей.

process *n* процесс.

process *v* обрабатывать; ~ **communications** обрабатывать сообщения (*преобразовывать в форму, пригодную для получения разведывательной информации*).

processing *n* обработка (*преобразование перехваченных сообщений в форму, пригодную для получения разведывательной информации*).

processor *n* процессор; ~**-flag attack** атака со знанием содержимого регистров процессора (*осуществляющего шифрование данных в соответствии с некоторым криптографическим алгоритмом*).

PROD *abbr* = Production.

produce *v* вырабатывать; ~ **ciphertext** вырабатывать шифртекст.

product *adj* составной, продукционный, производный, композиционный; ~ **cipher** суперпозиция шифров [составной / продукционный / производный / композиционный шифр]; ~ **operation** операция повторного шифрования (*шифрованного сообщения*).

product *n* (1) произведение, композиция, суперпозиция (*шифров*); ~ **of two ciphers** произведение двух шифров; (2) изделие, продукт; ~ **with embedded cryptography** продукт со встроенным криптографическим средством.

Production *prop* "Производство" (*подразделение АНБ США, специализирующееся на вскрытии шифрсистем и чтении шифрпереписки*).

professional *adj* профессиональный; ~ **cryptanalyst** криптоаналитик-профессионал.

program *n* программа.

programmable *adj* программируемый; ~ **key storage device** программируемое устройство хранения ключей.

project *n* проект; ~ **Minaret** проект "Минарет" (*согласно этому проекту с 1969 по 1973 г. в США осуществлялся комплекс мероприятий, имевших целью ограничить распространение сведений о том, что АНБ занимается сбором и обработкой информации конфиденциального характера*); ~ **Shamrock** проект "Трилистник" (*с 1952 по 1975 г. позволял АНБ на добровольной основе получать доступ к телеграммам, которые пересылались по каналам связи крупнейших коммерческих телекоммуникационных компаний США*).

proof *n* доказательство; ~ **of identity** установление личности; ~ **of ownership** доказательство права собственности.

propagating *n* размножение, распространение; ~ **cipher block chaining** сцепление блоков шифра с размножением.

propagation *n* размножение, распространение; ~ **criterion of degree k** критерий размножения [распространения] степени k (*двоичная функция удовлетворяет критерию размножения [распространения] степени k, если ее выходное значение меняется с вероятностью 1/2 всякий раз, когда ее i входов, $1 \le i \le k$, изменяют свои значения на противоположные*); ~ **of errors** размножение [распространение] ошибок.

proper *adj* правильный; соответствующий; ~ **key** правильный [соответствующий] ключ.

property *n* свойство.

proprietary *adj* частновладельческий; ~ **cryptoalgorithm** частновладельческий криптоалгоритм (*криптографический алгоритм, право собственности на который находится в частных руках*).

protect *v* защищать; ~ **against a timing attack** защищать от временной атаки; ~ **a private key against disclosure** защищать секретный ключ от разглашения.

protection *n* защита; ~ **against** защита от.

protocol *n* протокол (*последовательность действий, посредством которых две или более стороны совместно выполняют некоторое задание*).

provable *adj* достоверный, доказуемый; ~ **prime** достоверно простое число (*с доказуемой принадлежностью к классу простых чисел*); ~ **security** доказуемая стойкость.

provably *adv* достоверно, доказуемо; ~ **public-key** доказуемо открытый ключ; ~**-secure cipher** достоверно стойкий шифр.

proved *adj* проверенный; ~ **key** проверенный ключ.

provide *v* обеспечивать; ~ **maximum security** обеспечивать максимальную стойкость.

pseudo- *в сложных и сложносоставных словах имеет значение* псевдо-.

Pseudo-Hadamard transform *n* псевдопреобразование Адамара.

pseudokey *n* псевдоключ (*цифровая ключевая последовательность, часть которой используется в качестве ключа*).

pseudo-noise *n* псевдошум; ~ **binary sequence** псевдошумовая двоичная последовательность (*двоичная последовательность называется псевдошумовой, если она удовлетворяет трем критериям случайности, сформулированным американским криптологом С. Голомбом в 1967 г.*); ~ **generator** генератор псевдошума.

pseudoplaintext *n* = pseudotext.

pseudoprime *adj* псевдопростой; ~ **to the base b** псевдопростой по основанию b.

pseudorandom *adj* псевдослучайный; ~ **binary generator** генератор псевдослучайных двоичных последовательностей; ~ **bit generator** = ~ binary generator;

~ **distribution** псевдослучайное распределение; ~ **key** псевдослучайный ключ; ~ **key generator** генератор псевдослучайных ключей; ~ **number** псевдослучайное число; ~ **number generator** генератор псевдослучайных чисел; ~ **number key** ключ в виде набора из псевдослучайных чисел; ~ **permutation** псевдослучайная перестановка; ~ **sequence generator** генератор последовательностей псевдослучайных чисел; ~ **sequencer** генератор псевдослучайных последовательностей; ~ **time division encryption** мозаичное шифрование (*псевдослучайная перестановка сегментов речевых сигналов во времени*).

pseudorandomness *n* псевдослучайность.

pseudotext *n* псевдотекст (*текст, полученный в результате пробного дешифрования*).

p-time *abbr* п-сложный (*см. polynomial time*).

public *adj* открытый; ~ **directory** открытый справочник (*ключей или криптоалгоритмов*); ~ **encryption operation** открытая операция шифрования; ~ **exponent** открытый показатель степени.

public-key *n* открытый ключ; ~ **algorithm** алгоритм криптосистемы с открытым ключом; ~ **certificate** сертификат на открытый ключ; ~ **certification** сертификация (*заверение подлинности*) открытого ключа; ~ **cryptography** криптография с открытым ключом; ~ **Cryptography Standard** Стандарт шифрования с открытым ключом; ~ **cryptosystem** криптосистема с открытым ключом; ~ **distribution system** система открытого распространения ключей; ~ **management** управление открытыми ключами (*в двухключевой криптосистеме*); ~ **protocol** [криптографический] протокол с открытым ключом; ~ **registry** регистрация открытого ключа (*специальный процесс регистрации открытого ключа, обеспечивающий достоверную информацию лицу, которое осуществляет запрос относительно этого ключа*); ~**-secret key pair** пара открытый-секретный ключ (*в криптосистеме с открытым ключом*); ~ **system** = ~ cryptosystem.

publicly-revealed *adj* открытый; ~ **key** открытый ключ зашифрования [расшифрования] (*в двухключевой криптосистеме*).

publish *v* публиковать; ~ **an algorithm in a certified public directory** публиковать алгоритм в заверенном открытом справочнике.

published *adj* опубликованный; ~ **cryptoalgorithm** криптоалгоритм, опубликованный в открытой печати.

punch-card *n* перфокарта; ~ **key** ключ на перфокарте.

purchased *adj* приобретенный; ~ **key** приобретенный (*посредством подкупа*) ключ.

pure *adj* (1) однородный; ~ **cipher** однородный шифр (*шифр, в котором для любых трех преобразований зашифрования, расшифрования и снова зашифрования можно найти шифрующее преобразование, идентичное произведению этих трех преобразований*); (2) однородный, без примесей; ~**-listening attack** атака

на основе одного только прослушивания (*канала речевой связи*); ~**-listening attack secure** защищенный от атаки на основе одного только прослушивания (*канала речевой связи*).

purple *adj* пурпурный; ~ **cipher** = ~ machine; ~ **code** = ~ machine; ~ **machine** "пурпурная" машина (*условное наименование, данное американскими криптоаналитиками японской шифровальной машине, введенной в эксплуатацию в начале 40-х годов*).

purported *adj* предполагаемый; ~ **originator** предполагаемый отправитель (*сообщения*); ~ **transmitter** предполагаемый передатчик (*сообщения*).

puzzle *n* (1) загадка; ~ **Palace** "Дворец загадок" (*иносказательное название, придуманное американским журналистом Д. Бэмфордом для Агентства национальной безопасности США*); (2) "шарада" (*криптограмма, полученная при помощи блочного шифра с малым ключевым пространством и содержащая ключ в заданной стандартной форме*).

Q

QC *abbr* КК (*см. quantum cryptography*).

QRC *abbr* ШКВ (*см. quadratic residue cipher*).

quadratic *adj* квадратичный; ~ **congruential generator** квадратичный конгруэнтный генератор; ~ **function** квадратичная функция; ~ **residue modulo n** квадратичный вычет по модулю n (*целое число a такое, что уравнение* $x^2 \equiv a \bmod n$ *имеет целочисленное решение*); ~ **residue cipher** шифр квадратичных вычетов; ~ **residuosity problem** задача о квадратичных вычетах; ~ **sieve factoring** факторизация методом квадратичного решета.

quantum *adj* квантовый; ~ **cryptography** квантовая криптография (*в квантовой криптографии стойкость криптосистем основывается на принципе неопределенности квантовой механики*).

quasi- *в сложных и сложносоставных словах имеет значение* квази-.

quasi-random *adj* квазислучайный; ~ **number generator** генератор квазислучайных чисел.

quasi-weak *adj* квазинестойкий; ~ **key** квазинестойкий ключ.

quinque- *в сложных и сложносоставных словах имеет значение* пяти-.

quinqueliteral *adj* пятизначный; ~ **cipher alphabet** пятизначный шифралфавит (*служит для замены одного знака открытого текста пятью знаками шифрованного текста*).

R

radiation *n* излучение; ~**-protected** защищенный от воздействия излучений.

radio *n* радио; ~ **broadcast network** радиовещательная циркулярная сеть; ~ **eavesdropping** радиоперехват; ~ **frequencies** радиочастоты; ~ **intelligence** радиоразведка; ~ **intelligence unit** радиоразведывательное подразделение; ~ **intercept** радиоперехват; ~ **reconnaissance service** радиоразведывательная служба [служба радиоразведки].

railway *n* железная дорога; ~ **Enigma** "Железнодорожная Энигма" (*модификация шифровальной машины "Энигма", которая во время второй мировой войны использовалась немцами для шифрования данных, связанных с железнодорожными перевозками*); ~ **key** ключ к "Железнодорожной Энигме".

Rainbow Series *prop* "Радужная серия" (*серия дополнений к "Оранжевой книге", названная так по цвету ее обложек*).

RAM *abbr* ЗУПВ (*см. random access memory*).

random *adj* случайный; произвольный; ~ **access machine** компьютер, оснащенный запоминающим устройством с произвольной выборкой; ~ **access memory** запоминающее устройство с произвольной выборкой; ~ **bit generator** генератор случайных битовых [двоичных] последовательностей; ~ **bit pattern generator** = ~ bit generator; ~ **cipher** случайный шифр; ~ **number** случайное число; ~ **cryptanalysis** криптоанализ методом случайного поиска; ~ **distribution** случайное распределение; ~ **initialization** случайное начальное заполнение (*регистра сдвига*) [случайная начальная загрузка (*шифратора*)]; ~ **key** случайный ключ; ~ **key encryption** шифрование на случайном ключе; ~ **key generator** генератор случайных ключей; ~ **mapping** случайное отображение; ~ **mixed sequences** случайные смешанные последовательности; ~ **number generator** генератор [датчик] случайных чисел; ~ **padding** случайное холостое заполнение (*дополнение сообщения ничего не значащей случайной информацией*); ~ **permutation** случайная перестановка.

randomization *n* рандомизация (*внесение элемента случайности в работу криптографического оборудования*).

randomize *v* рандомизировать, делать случайным.

randomized *adj* рандомизированный; ~ **encipherment system** рандомизированная система шифрования; ~ **key encryption** = random key encryption; ~ **stream cipher** рандомизированный потоковый шифр.

randomizer *n* рандомизатор (*аналоговый или цифровой источник случайных последовательностей*).

randomizing *adj* рандомизирующий; ~ **sequence** рандомизирующая последовательность.

randomly *adv* случайно; ~ **chosen** выбранный случайно; ~**-chosen key** случайно выбранный ключ; ~**-generated** сгенерированный случайным образом; ~**-selected** = ~-chosen.

randomness *n* случайность; ~ **postulate** критерий случайности (*критерий для проверки случайности двоичной последовательности*).

rang *n* = rank.

rank *n* ранг; ~ **of a matrix** ранг матрицы.

rate *n* норма; ~ **of a language** норма языка.

raw *adj* = undigested.

RC *abbr* КР (*см. Rivest's Code*).

reaction *adj* реакционный; ~ **attack** реакционная атака (*атакующий использует реакцию жертвы на свои действия для достижения искомой цели*).

read *v* (1) читать (*дешифровывать*); ~ **cryptograms** читать криптограммы; (2) читать, считывать (*информацию*); ~**-only memory** постоянное запоминающее устройство (*позволяет только считывать из него информацию*).

really *adv* действительно; ~ **random sequence** действительно случайная последовательность (*если генератор действительно случайных последовательностей запустить два раза с одинаковыми входами, полученные выходные последовательности будут разными*).

real-time *adj* (*производимый*) в реальном масштабе времени; ~ **decryption** расшифрование в реальном масштабе времени.

real-world *adj* истинный, действительный; ~ **random sequence** = really-random sequence.

reassigned *adj* назначенный заново, переназначенный; ~ **key** назначенный заново [переназначенный] (*другому пользователю*) ключ.

receive *v* получать (*сообщение*).

received *adj* полученный; ~ **message** полученное сообщение.

receiver *n* получатель, адресат (*сообщения*); ~**'s key** ключ получателя [адресата] (*сообщения*).

recipient *n* = receiver.

reciprocal *adj* возвратный; ~ **polynomial** возвратный полином.

recirculating shift register *n* = feedback shift register.

recomputation *n* повторное вычисление (*напр., криптографической контрольной суммы*).

recompute *v* вычислять повторно.

reconstruct *v* восстановить, реконструировать; ~ **a military code** реконструировать военный код.

reconstructed *adj* восстановленный, реконструированный; ~ **plaintext** восстановленный (*в результате расшифрования или дешифрования шифртекста*) открытый текст.

reconstruction *n* восстановление, реконструкция; ~ **of internal settings of a cipher system** восстановление внутренних установок шифрсистемы; ~ **of the**

plaintext восстановление открытого текста; ~ **of the specific keys to the cipher system** восстановление конкретных ключей шифрсистемы.

recover *v* получать, восстанавливать; ~ **an encryption key from a decryption key** получать ключ зашифрования из ключа расшифрования; ~ **a plaintext** восстанавливать открытый текст (*из шифрованного*); ~ **session keys** восстанавливать сеансовые ключи.

recovered *adj* восстановленный; ~ **plaintext** восстановленный (*в результате расшифрования или дешифрования шифртекста*) открытый текст.

recreate *v* восстанавливать, воссоздавать; ~ **a key** восстанавливать [воссоздавать] ключ (*по отдельным его частям*).

recreation *n* восстановление, воссоздание.

recurrence *n* (1) рекуррентность; ~ **relation** рекуррентное соотношение; (2) рекуррента.

recurring *adj* рекуррентный; ~ **sequence** рекуррентная последовательность.

recursion *n* рекурсия.

recursive *adj* рекурсивный; ~ **function** рекурсивная функция.

red *adj* (1) красный; ~ **machine** "красная" машина (*условное наименование, данное американскими криптоаналитиками японской шифровальной машине, находившейся в эксплуатации с 1932 по 1941 г.*); (2) открытый, незашифрованный; ~**-black concept** концепция разделения открытой и шифрованной информации; ~**-black isolation** разделение открытой и шифрованной информации; ~ **box** блок обработки открытой информации; ~ **designation** обозначение систем, линий и аппаратуры связи для передачи и обработки незашифрованной информации; ~ **data** незашифрованные данные; ~ **side** сторона шифратора, обрабатывающая открытые данные; ~ **signal** открытый сигнал (*любое электромагнитное излучение, которое несет в себе информацию об используемых ключах, обрабатываемых открытых текстах и алгоритмах функционирования криптографического оборудования*); ~ **thread** "красная нить" (*программная закладка, внедряемая в шифровальную программу для понижения ее стойкости*).

reduce *v* уменьшать, упрощать; ~ **a key to a manageable size** уменьшить длину ключа до приемлемой величины.

reduced *adj* уменьшенный, упрощенный; ~ **—size sample of cryptosystem** упрощенный вариант криптосистемы с уменьшенными значениями параметров.

reduction *n* сведение, редукция; ~ **of a polyalphabetic cipher to a monoalphabetic one** сведение многоалфавитного шифра к моноалфавитному.

redundancy *n* избыточность; ~ **of message information contained in a N-digit cryptogram** избыточность текста криптограммы из N знаков.

redundant *adj* избыточный; ~ **authenticating information** избыточная аутентификационная информация.

reencipher *v* перешифровывать; ~ **with the appropriate key** перешифровывать на соответствующем ключе.

reencode *v* перекодировать.

reencodement *n* перекодирование.

reencrypt *v* = reencipher.

reflecting *adj* рефлекторный; ~ **rotor** рефлектор, рефлекторное колесо [ротор].

reflector *n* = reflecting rotor.

register *n* регистр; ~ **of length n** регистр длины [размера] n.

registry *n* = directory.

re-injected shift register *n* = feedback shift register.

reject *v* отвергать; ~ **a message as unauthentic** отвергать сообщение как ложное.

Rejevski *prop* Режевский М. (*польский криптоаналитик, внесший наибольший вклад во вскрытие поляками немецкой шифровальной машины "Энигма" перед второй мировой войной*).

rekey *v* вводить ключ повторно.

rekeying *n* повторный ввод ключа.

related *adj* родственный; имеющий отношение; ~**-data cryptanalysis** криптоанализ с использованием дополнительных данных; ~ **key** "родственный" ключ (*производный от другого ключа или имеющий какое-либо еще к нему отношение*); ~**-key cryptanalysis** криптоанализ на основе "родственных" ключей (*исследуется влияние различий между ключами на работу шифратора*).

relation *n* соотношение; зависимость.

relatively *adv* относительно; ~ **prime to** относительно простой по отношению к.

release *n* редакция, версия; ~ **prefix** редакционный префикс (*пометка в заголовке документа с ключевой информацией, свидетельствующая о том, могут ли ближайшие союзники США получить доступ к содержимому этого документа*).

reload *v* загружать [вводить] заново; ~ **a key** загружать [вводить] ключ заново.

remote *adj* удаленный; ~ **control key erasure** дистанционно управляемое стирание ключа (*записанного в памяти*); ~ **key entry** удаленный ввод ключа; ~ **keying** = ~ key entry; ~ **key load** удаленная загрузка [удаленный ввод] ключа; ~ **rekeying** = ~ key entry.

repeated *adj* повторный, повторяющийся, многократный; ~ **encryption** повторное [повторяющееся / многократное] шифрование; ~ **key cipher** шифр с повторяющимся ключом (*шифртекст вырабатывается при помощи побитового сложения по модулю 2 знаков открытого текста и периодической ключевой последовательности, полученной путем повторения исходного ключа*); ~ **key load** повторная загрузка [повторный ввод] ключа; ~ **plaintext digraphs** повторяющиеся биграммы открытого текста.

repeating *adj* = repeated.

repetitive *adj* = repeated.

repetition *n* повторение; **~s in a ciphertext** повторения в шифртексте.

replacement *n* замена, подстановка.

required *adj* требуемый; **~ key** требуемый ключ.

re-radiation *n* повторное излучение; переизлучение.

rescrambling *n* повторное скремблирование.

reserve *adj* запасной, резервный; **~ key** запасной [резервный] ключ; **~ keying material** запасной ключевой материал (*ключи, предназначенные для использования при возникновении непредвиденных обстоятельств*).

residence *n* резидентура.

resident *n* резидент.

residue *n* (1) вычет; **~ class** класс вычетов; (2) остаток (*от деления*); **r-th ~ cryptosystem** криптосистема с остатком порядка r.

resistance *n* стойкость; **~ against ciphertext-only attack** стойкость против атаки на основе знания одного только шифрованного текста; **~ to cryptanalysis** стойкость против криптоанализа; **~ to cryptanalytic attack** = ~ to cryptanalysis.

resistant *adj* стойкий; **~ against plaintext attack** стойкий против атаки со знанием открытого текста; **~ to plaintext attack** = ~ against plaintext attack.

resolution *n* решение (*математической задачи*); **~ of decryption** решение задачи дешифрования.

resource *n* ресурс; **~-limited cryptanalyst** криптоаналитик с ограниченными ресурсами.

restoration *n* восстановление.

restore *v* восстанавливать.

restored *adj* восстановленный; **~ message** восстановленное (*после зашифрования*) сообщение.

restricted *adj* суженный; **~ cryptoalgorithm** суженный криптоалгоритм (*стойкость определяется секретностью самого алгоритма*).

resulting *adj* результирующий, получающийся; **~ cryptogram** результирующая криптограмма; **~ key** результирующий ключ.

retired *adj* использованный; **~ key** использованный ключ.

retrieval *n* поиск.

reuse *n* повторное использование; **~ of a key** повторное использование ключа.

reuse *v* повторно использовать; **~ a pad** повторно использовать шифрблокнот.

reused *adj* повторно использованный; ~ **key** повторно использованный ключ.

reveal *v* раскрывать; ~ **a secret key to an enemy** раскрывать перед противником секретный ключ.

revealed *adj* раскрытый, разглашенный; ~ **key** раскрытый ключ (*вскрытый противником или ставший ему известным в силу каких-либо других обстоятельств*).

reverse *adj* обратный; ~ **standard cipher alphabet** обратный стандартный шифр-алфавит.

reversed *adj* обращенный, расположенный в обратном порядке; ~ **alphabet** обращенный [расположенный в обратном порядке] алфавит.

reverse-encode *v* = decode.

reverse-encrypt *v* = decrypt.

reverse-engineer *v* восстанавливать средствами обратного проектирования; ~ **a cryptographic algorithm from executable code** восстановить криптографический алгоритм средствами обратного проектирования по исполняемому коду (*программы*); ~ **a smart card** восстанавливать алгоритм функционирования интеллектуальной карты средствами обратного проектирования.

reversible *adj* обратимый; ~ **encryption** обратимое шифрование; ~ **transformation** обратимое преобразование.

revoke *v* аннулировать, объявлять недействительным; ~ **a key** аннулировать, [объявлять недействительным] ключ;.

revoked *v* аннулированный, объявленный недействительным; ~ **a key** аннулированный, [объявленный недействительным] ключ;.

RF *abbr* РЧ (*см. radio frequences*); ~ **signals** РЧ-сигналы.

rig *v* искусственно занижать стойкость; ~ **a cipher machine** искусственно занижать стойкость шифратора.

rigged *adj* характеризующийся искусственно заниженной стойкостью; ~ **Crypto AG cipher machine** шифратор фирмы "Крипто АГ" с искусственно заниженной стойкостью.

ring *n* кольцо.

Rip van Winkle *prop* Рип ван Винкль (*герой одноименного рассказа американского писателя В. Ирвинга*); ~ **cipher** шифр Рипа ван Винкля.

Rivest *prop* Ривест Р. (*американский криптолог, один из авторов алгоритма шифрования РША*); ~**'s Cipher** = ~'s Code; ~**'s Code** Код Ривеста (*общее название серии алгоритмов шифрования, разработанных Ривестом*).

RKG *abbr* ГКП (*см. running-key generator*).

RN *abbr* СЧ (*см. random number*).

RNG *abbr* ГСЧ (*см. random number generator*).

Rocket *prop* = Railway Enigma.

rolling *adj* чередующийся, изменяющийся; ~ **bandscrambler** спектрально — полосный скремблер с чередующимся кодом перестановки полос; ~ **key** = running key; ~ **mode** режим работы с изменяющимся ключом.

ROM *abbr* ПЗУ (*см. read-only memory*).

Room 40 *n* "Комната 40" (*подразделение английского адмиралтейства, в годы первой мировой войны занимавшееся криптоанализом перехваченных шифрсообщений противника*).

Room Forty *n* = Room 40.

rot13 *abbr* = rotate alphabet 13 places.

rotary *adj* роторный; ~ **wheel** роторное колесо.

rotate *v* вращать; ~ **alphabet 13 places** вращать алфавит на 13 позиций (*простейший шифр, который заменяет каждую букву английского алфавита на другую, расположенную в нем через 13 позиций*).

Rotation Cipher *n* = rottate alphabet 13 places.

rotor *n* ротор, колесо; ~ **initial settings** = starting positions; ~ **machine** роторная [колесная] шифровальная машина; ~ **movement** движение роторов; ~ **order** порядок следования роторов; ~ **starting positions** начальные положения роторов.

rough *adj* неравновероятный; ~ **frequency count** маркировка (*текста*), показывающая значительные отличия от равновероятного распределения.

roughness *n* неравновероятность.

round *n* цикл, раунд (*один из последовательных шагов обработки в алгоритме блочного шифрования*); ~ **function** функция цикла [раунда]; ~ **key** ключ цикла [раунда].

route *n* путь, маршрут; ~ **transposition** шифр путевой перестановки; ~ **transposition sequences** путевые перестановочные последовательности.

routine *n* (1) программа; (2) алгоритм.

routing *n* маршрутизация; ~ **information** сведения маршрутизации.

RSA *abbr* (*R. Rivest, A. Shamir and L. Adelman*) РША (*Р. Ривест, А. Шамир и Л. Адельман*); ~ **algorithm** алгоритм РША; ~**-encrypted message** сообщение, зашифрованное по алгоритму РША; ~ **implementation** реализация алгоритма РША; ~ **key pair** пара открытый-секретный ключ в криптосистеме РША; ~ **protocol for mental poker** протокол РША для игры в покер "вслепую" (*криптографический протокол, дающий возможность честно играть в покер, не прибегая к картам*); ~ **scheme** схема РША; ~**-signed message** сообщение, заверенное (*цифровой*) подписью по алгоритму РША; ~ **system** система РША; ~ **trapdoor one-way function** односторонняя функция РША с потайным ходом.

RSADSI *abbr* = RSA Data Security Incorporated (*название американской компании, производящей средства шифрования данных*).

RSALABS *abbr* = RSA Laboratories (*подразделение американской компании RSA Data Security Incorporated*).

RSAREF *abbr* библиотека криптографических процедур на языке программирования "Си", разработанная американской фирмой RSA Laboratories .

rule *n* правило; ~**-based cryptanalysis** криптоанализ на основе принятых правил; ~ **D** = deciphering ~; ~ **E** = enciphering ~; ~ **of encipherment** = enciphering ~.

run *n* "полоска" (*набор одинаковых, подряд идущих компонентов битовой последовательности, которому предшествует и за которым следует компонент, отличный от повторяющегося в наборе компонента*); ~**s test** "полосковый" тест (*используется для проверки случайности двоичной последовательности*).

run *v* работать; ~ **about one thousandth as fast as DES** работать примерно в тысячу раз медленнее, чем DES-алгоритм.

running *adj* переменный, пробегающий ряд значений; ~ **key** ключевой поток; динамический [бегущий] ключ; ключевая последовательность; ~ **key cipher** (1) шифр с динамическим [бегущим] ключом; (2) шифр с одинаковой длиной ключа и сообщения; ~ **key generator** генератор ключевого потока [ключевой последовательности].

Russian *adj* российский; ~ **encryption algorithm** "российский алгоритм шифрования" (*ГОСТ 28147-89*).

S

SAC *abbr* СКЛИ (*см. strict avalanche criterion*).

safe *adj* безопасный; ~ **cryptoalgorithm** безопасный криптоалгоритм (*затраты на вскрытие этого криптоалгоритма значительно превышают ценность шифруемых с его помощью данных*).

safeguard *v* защищать; ~ **communications** защищать каналы связи.

SAFER *abbr* СИБАШ (*см. Secure And Fast Encryption Routine*).

salt *n* = seed.

sample *n* (1) выборка; ~ **estimation** оценка по выборке, выборочная оценка; (2) проба; ~ **key** пробный ключ (*предназначен для целей наладки криптографического оборудования*).

satellite *n* спутник; ~ **interception system** система спутникового перехвата; ~ **personal communications system** система персональной спутниковой связи; ~ **scrambler** скремблер для спутниковой связи.

satscrambler *abbr* спутскремблер (*см. satellite scrambler*).

S-box *abbr* S-блок (*см. substitution box*).

scalable *adj* масштабируемый; ~ **cipher** масштабируемый шифр (*позволяет изменять свои параметры – длину ключа, размер блока и т. д.*).

scalar *adj* скалярный; ~ **product** скалярное произведение.

SCE *abbr* СПСИ (*см. Special Collection Element*).

scheme *n* схема, система.

Scherbius *prop* Шербиус А. (*немецкий инженер, изобретатель шифровальной машины "Энигма"*).

Schmidt *prop* Шмидт Г. (*немецкий офицер, с 1931 по 1943 г. продававший ключи и схемы шифровальной машины "Энигма" сначала французам, а затем англичанам*).

scientific *adj* научный; ~ **secret-key cryptology** научная криптология с секретным ключом.

SCORE *abbr* СККИ (*см. Special Committee on Compromising Emanation*).

SCR *abbr* СКР (*см. scrambler*).

scramble *v* скремблировать (*засекречивать речевое сообщение*).

scrambled *adj* скремблированный; ~ **message** сообщение, защищенное с помощью скремблирования.

scrambler *n* скремблер (*устройство засекречивания речевых сообщений*).

scrambling *n* скремблирование (*преобразование речевых данных по определенному закону с целью исключения подслушивания*); ~ **equipment** аппаратура скремблирования; ~ **generator** генератор кода скремблирования; ~ **key** ключ скремблирования; ~ **matrix** матрица скремблирования.

scratch *adj* рабочий; ~ **pad store** временная ключевая память (*используется для временного хранения ключевой информации в криптографическом оборудовании*).

SCS *abbr* ССС (*см. Special Collection Service*).

scytale *n* = skytale.

SEAL *abbr* ОПАШ (*см. Software-optimized Encryption Algorithm*).

seal *v* запечатывать.

sealed *adj* запечатанный; ~**-authenticator authentication scheme** схема аутентификации с запечатанным аутентификатором.

sealing *adj* запечатывающий; ~ **encryption algorithm** запечатывающий алгоритм шифрования.

searched *adj* искомый; ~ **key** искомый ключ (*при криптоанализе*).

second *adj* = secondary.

secondary *adj* вторичный; ~ **key** вторичный ключ (*доступ к которому открывает другой ключ*).

secrecy *n* секретность; ~ **channel** канал обеспечения секретности.

secret *adj* секретный, тайный; ~ **exponent** секретный показатель степени; ~ **key** секретный ключ; ~**-key agreement** соглашение [договоренность] о секретном ключе (*между участниками обмена сообщениями*); ~**-key cipher** шифр с секретным ключом; ~**-key cryptography** криптография с секретным ключом (*одноключевая криптография*); ~ **key distribution** (1) секретное распределение ключей; (2) распределение секретных ключей; ~ **writing** тайнопись.

secure *adj* стойкий; надежный; защищенный; засекреченный; ~ **against** обеспечивающий защиту от [защищенный от]; ~ **And Fast Encryption Routine** Стойкий и быстрый алгоритм шифрования (*алгоритм блочного шифрования, изобретенный американским криптологом Дж. Месси*); ~ **channel** защищенный канал; ~ **electronic exchange of keys** секретный электронный [безопасный] обмен ключами; ~ **Hash Algorithm** Стойкий алгоритм хэширования; ~ **Hash Standard** Стандарт (*США*) на стойкое хэширование; ~ **Hypertext Transfer Protocol** Защищенный протокол для передачи гипертекста (*криптографический протокол для передачи мультимедийных документов в компьютерной сети "Интернет"*); ~ **Multipurpose Internet Mail Extensions** Дополнительные средства защиты электронной почты в "Интернет" (*криптографический протокол для шифрования сообщений, передаваемых через компьютерную сеть "Интернет", а также для работы с цифровыми подписями*); ~ **key exchange** = ~ exchange of keys; ~ **key generator** генератор стойких ключей; ~ **Key Management Protocol** Защищенный протокол управления ключами (*разработанная корпорацией "Ай-Би-Эм" система депонирования ключей*); ~ **key passing channel** защищенный канал для передачи ключей; ~ **message** защищенное [засекреченное] сообщение; ~ **Sockets Layer** Защищенный гнездовой уровень (*криптографический протокол американской фирмы "Нетскейп Коммьюникейшнс", предназначенный для защиты информации при ее передаче по глобальной компьютерной сети "Интернет"*); ~ **Wide Area Network** Защищенная глобальная компьютерная сеть (*набор аппаратных и программных средств для обеспечения конфиденциальности при работе с протоколами TCP/IP*).

secure *v* (1) обеспечивать; ~ **integrity of information** обеспечивать целостность информации; (2) защищать; ~ **against substitution and modification** защищать от подмены и модификации.

security *n* (1) стойкость; надежность; безопасность; ~ **auditing** проверка на стойкость; ~ **estimate** оценка стойкости (*криптосистемы*); ~ **of a key** стойкость ключа; ~ **period** период сохранения стойкости (*критосистемы*); (2) засекречивание, сохранение в секрете; ~ **label** гриф секретности (*пометка на документе, обозначающая степень секретности содержащейся в нем информации*); ~ **period** период сохранения (*информации*) в секрете.

seed *n* начальное заполнение; величина рассеивания [рандомизации] (*данные, используемые для инициализации генератора псевдослучайной последовательности, напр., линейного регистра сдвига*); ~ **key** исходный [первичный] ключ; ~ **number** порождающее число; ~ **value** = ~.

SEEK *abbr* СЭОК (*см. secure electronic exchange of keys*).

selected *adj* выбранный; ~ **key** выбранный ключ.

selective *adj* избирательный; ~ **encryption** избирательное шифрование (*отдельных отрезков открытого текста сообщения*); ~ **key distribution** избирательное распределение ключей; ~ **packet encryption** избирательное шифрование пакетов (*сообщения*).

self- *в сложных и сложносоставных словах имеет значение* само-.

self-dual *adj* самодвойственный, самосдвоенный; ~ **key** самодвойственный [самосдвоенный] ключ (*в DES-алгоритме*).

self-shrinking *adj* самосверточный; ~ **generator** самосверточный генератор.

self-synchronous *adj* самосинхронизирующийся; ~ **key generator** самосинхронизирующийся генератор ключей; ~ **stream cipher** самосинхронизирующийся поточный шифр.

self-synchronizing *adj* = self-synchronous.

semantic *adj* семантический; ~ **security** семантическая стойкость.

semi- *в сложных и сложносоставных словах имеет значение* полу-.

semi-infinite *adj* полубесконечный; ~ **sequence** полубесконечная последовательность (*генерируется конечная случайная последовательность длины n и затем повторяется нужное число раз для получения периодической последовательности с периодом n*).

semiweak *adj* полустойкий, полувырожденный; ~ **key** полустойкий [полувырожденный] ключ.

send *v* посылать, передавать (*сообщение*); ~ **a message over a link** посылать сообщение по каналу (*связи*); ~ **a message under an existing protocol** посылать сообщение в соответствии с существующим протоколом.

sender *n* отправитель (*сообщения*).

sensitive *adj* (1) секретный, конфиденциальный (*об информации*); (2) уязвимый.

sensitivity *n* (1) секретность, конфиденциальность; (2) уязвимость.

separation *n* разделение; ~ **of capacities for encryption and decryption in public-key cryptosystems** разделение возможностей (*пользователей*) по зашифрованию и расшифрованию в криптосистемах с открытым ключом (*многие могут зашифровать сообщение, которое в состоянии расшифровать только один человек, или наоборот, один человек может зашифровать сообщение, которое в состоянии прочесть многие*).

sequence *n* последовательность; ~ **generation** генерация [порождение] последовательности.

sequencer *n* генератор последовательностей.

serial *adj* последовательный; ~ **test** последовательный тест (*служит для проверки случайности двоичной последовательности путем сравнения вероятностей совпадения и различия ее двух подряд идущих элементов*).

service *n* (1) служба; ~ **№ 1** Служба № 1 (*подразделение 16 управления КГБ, занимавшееся разработкой и установкой закладок в шифровальное оборудование*); (2) военная служба; ~ **Enigma** военная "Энигма" (*одна из модификаций шифровальной машины "Энигма", которые использовались в вооруженных силах Германии во время второй мировой войны*).

session *n* сеанс; ~ **key** сеансовый ключ (*действующий только в течение ограниченного промежутка времени, напр., в одном сеансе передачи сообщений*); ~ **cryptography** криптография со сменой сеансового ключа.

set *n* (1) множество; (2) набор, комплект; ~ **of keys** комплект ключей.

set *v* устанавливать, вводить; ~**-key operation** операция установки [ввода] ключа; ~**key** команда "ввести ключ".

settings *n pl* установки (*параметры шифрсистемы, определяющие состояние, в котором она находится*); ~ **of a cipher system** установки шифрсистемы.

set up *v* сделать начальные установки; ~ **a cipher machine** сделать начальные установки в шифровальной машине.

SHA *abbr* САХ (*см. Secure Hash Algorithm*).

Shamir *prop* Шамир А. (*американский криптолог, один из авторов алгоритма шифрования РША*); ~**'s three-pass protocol** трехэтапный протокол Шамира (*криптографический протокол, не требующий предварительного распределения ключей перед обменом сообщениями*).

Shannon *prop* Шеннон К. (*американский математик*); ~**'s bound for perfect secrecy** граница Шеннона для совершенно стойких систем; ~**'s five criteria** пять критериев Шеннона (*критерии оценки шифрсистем, предложенные Шенноном в 1940 г.*); ~**'s theory approach** подход на основе теории Шеннона.

share *v* пользоваться совместно; ~ **a key** пользоваться ключом совместно; ~ **keys with a key distribution center** получать ключи из центра распределения ключей.

shared *adj* находящийся в совместном владении; ~ **key** ключ, находящийся в совместном владении нескольких лиц; ~ **common key** = ~ key.

Shark *prop* "Акула" (*модификация немецкой шифровальной машины "Энигма", использовавшаяся подводным флотом Германии во время второй мировой войны*).

shielded *adj* экранированный, экранирующий; ~ **enclosure** экранированный [экранирующий] корпус [кожух]; ~ **room** экранированная комната (*помещение, защищенное от электромагнитных излучений*).

shift *n* сдвиг; ~**-by-thirteen cipher** = rot13; ~**-by-three cipher** = Caesar cipher; ~ **register** регистр сдвига; ~ **register cascade** каскад регистров сдвига

(*множество регистров сдвига, соединенных между собой таким образом, что работа одного отдельно взятого регистра зависит от функционирования предыдущего регистра каскада*); ~ **register with linear feedback** регистр сдвига с линейной обратной связью; ~ **register with non-linear feedback** регистр сдвига с нелинейной обратной связью; ~ **register with non-linear logic** = ~ register with non-linear feedback; ~ **transformation** сдвиговое преобразование.

shift *v* сдвигать; ~ **a register one position to the left** сдвигать содержимое регистра на одну позицию влево.

short *adj* (1) короткий; ~ **key** короткий ключ ; ~ **shift register** короткий регистр сдвига; (2) краткий; ~ **signal cipher** краткий сигнальный шифр (*во время второй мировой войны применялся для засекречивания эфирных позывных немецких военных кораблей и подводных лодок*); ~ **signal codebook** = ~ signal cipher; ~ **weather cipher** краткий погодный шифр (*во время второй мировой войны применялся немецкими военными кораблями и подводными лодками для засекречивания прогнозов и сводок погоды*); ~ **weather codebook** = ~ weather cipher.

short-cut *adj* упрощенный; ~ **cryptanalysis** упрощенный криптоанализ.

short-term *adj* краткосрочный; ~ **key** краткосрочный ключ.

shrinking *adj* сверточный; ~ **generator** сверточный генератор.

SHS *abbr* ССХ (*см. Secure Hash Standard*).

S-HTTP *abbr* ЗППГ (*см. Secure Hypertext Transfer Protocol*).

shuffle *n* тасование (*разновидность перестановки*).

side *adj* побочный; ~**-channel attack** атака через побочный канал (*вместо криптоалгоритма анализу подвергается его аппаратная или программная реализация — каким образом изменяется энергопотребление шифратора в различных режимах его работы, как он функционирует, если температура окружающей среды значительно превышает установленную норму, сколько времени занимает выполнение той или иной криптоопераций и т. д.*); ~**-channel cryptanalysis** = ~-channel attack.

side *n* сторона (*шифратора*).

SIG *abbr* СИГ (*см. signature*).

SIGINT *abbr* РТР (*см. signals intelligence*); ~ **performance** эффективность РТР; ~ **satellite** спутник РТР; ~ **service** служба РТР; ~ **summary** суммарная сводка РТР.

sigmage *n* определение величины стандартного отклонения [уклонения].

sign *n* подписывать, заверять; ~ **a digital message** подписать цифровое сообщение..

signal *n* сигнал; ~ **related information** информация о сигнале; ~**s analyst** специалист в области радиотехнической разведки; ~**s intelligence** радиотехническая разведка.

signature *n* сигнатура, подпись; ~ **authentication key** ключ аутентификации [цифровой] подписи; ~ **block** блок подписи (*присоединяемый к сообщению*); ~ **channel** канал обмена сообщениями с подписью; ~ **generation key** ключ генерации [цифровой] подписи; ~ **key** ключ [цифровой] подписи.

significance *n* значимость; ~ **level of a test** уровень значимости теста.

SIGSUM *abbr* = SIGINT summary .

simple *adj* простой; ~ **columnar transposition** шифр простой колонной перестановки; ~ **Key for IP** Простой ключ для IP-протокола (*упрощенная система управления ключами при работе с IP-протоколом, созданная корпорацией "Сан Майкросистемс"*); ~ **substitution** простая замена.

simplified *adj* упрощенный; ~ **DES** упрощенный DES-алгоритм.

simultaneous *adj* совместный; ~ **diophantine approximation** совместные диофантовы приближения.

single *adj* однократный, одинарный; единый; ~ **encryption** однократное шифрование; ~ **length key** ключ единой установленной длины; ~ **transposition** шифр одинарной перестановки.

single- *в сложных и сложносоставных словах имеет значение* одно-.

single-bit *adj* однобитовый; ~ **cipher feedback** однобитовая обратная связь по шифртексту.

single-iterated *adj* одноитерационный (*с использованием единственной итерации*).

single-key *adj* одноключевой; ~ **cryptography** одноключевая криптография (*криптография с секретным ключом*); ~ **network** сеть (*засекреченной связи*) с использованием одного ключа.

single-letter *adj* однобуквенный; ~ **frequency cryptanalysis** криптоанализ, основанный на частотах встречаемости [появления] отдельных букв (*в тексте*).

single-use *adj* одноразовый; ~ **pad** одноразовый шифрблокнот; ~ **pad encryption** шифрование при помощи одноразового шифрблокнота.

single-valued *adj* однозначный; ~ **encryption** однозначное шифрование.

size *n* размер, длина (*ключа*); ~**-limited key** ограниченный по размерам [длине] ключ; ~ **of a key** размер [длина] ключа (*количество бит в ключе*).

SK *abbr* (1) СК (*см. secret key*); (2) СК (*см. session key*).

s-key *abbr* (1) с-ключ (*см. secret key*); (2) с-ключ (*см. session key*).

skip *n* прыжок, скачок; ~ **code** "прыжковый" код (*зашифрование осуществляется вставкой букв открытого текста сообщения в другой текст невинного содержания*); ~**jack** "Скипджек" [*букв.* "Попрыгунчик"] (*название алгоритма шифрования, аппаратно реализованного в виде чипа "Клиппер"*).

SKMP *abbr* ЗПУК (*см. Secure Key Management Protocol*).

skytale *n* сцитала (*примитивное шифровальное устройство у древних греков*).

small *adj* короткий; ~ **key** короткий ключ.

S/MIME *abbr* ДСЗЭПИ (*см. Secure Multipurpose Internet Mail Extensions*).

smooth *adj* гладкий; ~ **integer** "гладкое" целое число (*обладает только малыми по величине простыми делителями*).

sneaker *n* взломщик средств защиты компьютерных систем.

software *n* программные средства; ~ **cracker** программа вскрытия (*ключа, шифра*); ~ **encryption** программные средства шифрования; ~**-based encryption** = ~ encryption; ~ **implementation of DES** программная реализация DES-алгоритма; ~**-implemented encryption algorithm** программная реализация алгоритма шифрования; ~**-optimized Encryption Algorithm** Оптимизированный для программной реализации алгоритм шифрования.

solution *n* (1) решение (*математической задачи*); ~ **space** пространство решений; ~ **to a knapsack problem** решение задачи об укладке ранца; (2) вскрытие; ~ **by analogy** вскрытие по аналогии; ~ **by frequency matching** вскрытие методом согласования частот; ~ **by probable word method** вскрытие методом вероятного слова; ~ **efforts on a machine** усилия по вскрытию шифровальной машины; ~ **of a cipher** вскрытие шифра.

solvable *adj* вскрываемый; ~ **cipher** вскрываемый.

solve *v* (1) решать (*математическую задачу*); ~ **a system of linear equations** решать систему линейных уравнений; (2) вскрывать, подвергать криптоанализу (*напр., шифратор*); ~ **by cryptanalysis** вскрыть с помощью криптоанализа; ~ **a cipher** вскрывать шифр; ~ **the Enigma** вскрывать "Энигму"; ~ **a key** вскрывать ключ.

solved *adj* вскрытый; ~ **cipher** вскрытый шифр.

solving *n* вскрытие.

sort *v* сортировать, упорядочивать; ~ **cryptograms by frequency [by length]** сортировать [упорядочивать] криптограммы по частоте [длине].

source *n* источник, отправитель (*сообщения*); ~ **alphabet** исходный [входной] алфавит; ~ **coding** первоначальное кодирование (*уменьшение избыточности открытого текста сообщения перед его зашифрованием*); ~ **cryptor** шифратор источника [отправителя] (*сообщения*); ~ **key** ключ источника [отправителя] (*сообщения*); ~**-to-destination encryption** = end-to-end encryption.

Soviet *adj* советский; ~ **encryption algorithm** "советский алгоритм шифрования" (*ГОСТ 28147-89*).

space *n* (1) пространство; ~**-efficient** эффективный по памяти (*алгоритм*); (2) пробел.

spare *adj* запасной; ~ **key** запасной ключ.

SPCS *abbr* СПСС (*см. satellite personal communications system*).

special *adj* специальный; специализированный; ~ **Collection Element** Специальное подразделение по сбору информации (*станция перехвата АНБ США, расположенная в американском посольстве или консульстве*); ~ **Collection Service** Специальная служба сбора информации (*занимается разработкой и изготовлением оборудования для тайных технических операций по сбору информации для АНБ и ЦРУ США, а также готовит квалифицированный персонал для обслуживания этого оборудования*); ~ **Committee on Compromising Emanation** Специальный комитет (*АНБ США*) по компрометирующему излучению; ~ **Compartmented Information** специально скомпонованная информация (*обозначение, применяемое в АНБ США для особо секретных разведывательных данных, которые, как правило, добываются путем перехвата и дешифрования сообщений*); ~ **purpose machine** специализированная вычислительная машина; ~ **Section** = Spets-Otdel; ~ **Signals Division** Специальное подразделение связи (*британское военное радиоразведывательное подразделение*); ~ **signature scheme** специальная схема цифровой подписи.

specific *adj* специальный; ~ **key** специальный ключ (*выбранный из некоторого множества ключей для конкретного применения*).

speech *n* речь; ~ **encryption** шифрование речи; ~ **scrambler device** речевой скремблер; ~ **security device** прибор засекречивания речи.

speed up *v* ускорять; ~ **an attack** ускорять атаку.

Spets-Otdel *prop* Спецотдел.

spiral *adj* спиральный; ~ **cipher clock** спиральный шифровальный циферблат (*графическое представление шифра многоалфавитной замены в виде спирали, с внутренней стороны которой выписываются буквы алфавита открытого текста, а с внешней – алфавитов шифрованного*).

split *adj* разделенный; ~ **key** разделенный (*на несколько частей*) ключ; ~ **key cryptosystem** система с разделенным ключом; ~**-key entry** ввод ключа по частям; ~**-key technique** метод распределения (*секретного*) ключа по частям между отдельными лицами.

split *v* делить на части; ~ **a key** делить ключ на части (*ключ делится на составные части, которые распределяются между несколькими лицами так, что ключ может быть восстановлен в полном объеме только в том случае, если каждый из них предоставит для этого свою часть ключа*).

splitting *adj* ветвящийся; ~ **strategy** ветвящаяся стратегия.

splitting *n* ветвление; ~ **does not occur in any encoding rule** ветвления не происходит ни в одном правиле кодирования.

SPN *abbr* ППС (*см. substitution-permutation network*).

spread *n* распространение; ~ **of cryptography** распространение криптографии.

SPS *abbr* ВКП (*см. scratch pad store*).

spy *v* следить, шпионить; ~ **on a communication channel** следить за тем, что передается по каналу связи.

square *n* квадрат (*матрица размера nxn, элементами которой являются символы n-буквенного алфавита*).

square *v* возводить (*число*) в квадрат; ~**-and-multiply trick** метод возведения в произвольную степень путем многократного возведения в квадрат и умножения.

SR *abbr* РС (*см. shift register*).

SRI *abbr* ИОС (*см. signal related information*).

SSD *abbr* СПС (*см. Special Signals Division*).

SSL *abbr* ЗГУ (*см. Secure Sockets Layer*).

stage *n* разряд, ступень, каскад; **n-~ shift register** n-разрядный регистр сдвига.

stale *adj* устаревший, потерявший значение; ~ **message** устаревшее [потерявшее значение] сообщение.

stand-alone *adj* автономный; ~ **encryption device** автономное устройство шифрования.

standard *adj* стандартный; ~ **cipher alphabet** стандартный шифралфавит; ~ **cryptanalysis** стандартный криптоанализ; ~ **encryption algorithm** стандартный алгоритм шифрования; ~ **key** стандартный ключ (*напр., имеющий стандартную длину*); ~ **mode of operation** стандартный режим работы (*шифратора*).

start-up *n* пуск, запуск; ~ **key encryption key** пусковой ключ шифрования ключей (*распространяется среди потенциальных абонентов сети связи*).

state *n* состояние; ~ **of a cryptosystem** состояние криптосистемы; ~ **of a shift register** состояние регистра сдвига.

static *adj* статический; ~ **key** статический (*не изменяющийся во времени*) ключ.

station *n* (1) станция (*перехвата*); ~**-to-~ protocol** = authenticated key agreement protocol; (2) резидентура.

statistical *adj* статистический; ~ **attack** статистическая атака; ~ **approach** статистический подход; ~ **cryptanalysis** статистический криптоанализ; ~ **properties of a sequence** статистические свойства последовательности; ~ **test of a sequence** статистический тест последовательности.

statistically *adv* статистически; ~ **even** статистически равновероятный; ~ **flat** = ~ even; ~ **independent keys** статистически независимые ключи.

statistics *n pl* статистика, статистические свойства; ~ **of a ciphertext** статистические свойства шифртекста; ~ **of a plaintext** статистические свойства открытого текста.

steckerboard *n* = plug-board.

steganalysis *n* стеганоанализ (*аналитические методы, которые применяются для обнаружения и прочтения сообщений, засекреченных при помощи стеганографии*).

steganogram *n* стеганограмма.

steganographic *adj* стеганографический; ~ **methods** методы стеганографии; ~ **program** стеганографическая [компьютерная] программа.

steganography *n* стеганография.

step *n* шаг; ~ **encryption** пошаговое шифрование.

stereotype *adj* стереотип, шаблон.

stereotyped *adj* стереотипный, шаблонный; ~ **message** стереотипное [шаблонное] сообщение.

still-sensitive *adj* по-прежнему конфиденциальный; ~ **ciphertext** недостаточно стойкий шифртекст (*требует повторного зашифрования*).

Stimson *prop* Стимсон Г. (*государственный секретарь США, автор крылатой фразы "Джентльмены не читают переписку друг друга", произнесенной в оправдание принятого им решения прекратить в 1929 г. финансирование "Американского черного кабинета"*).

stochastic *n* случайный; ~ **distribution** случайное распределение.

storage *n* хранение; ~ **of keys** хранение ключей.

strategic *adj* стратегический; ~ **communication interception** перехват стратегических сообщений; ~ **cryptography** криптография стратегического уровня.

stream *n* поток; ~ **encryption** поточное шифрование.

strength *n* криптостойкость; ~ **of cryptographic algorithm** стойкость криптографического алгоритма; ~ **of encryption** криптостойкость шифрования.

strengthen *v* повышать криптостойкость; ~ **a cipher** повышать криптостойкость шифра.

strict *adj* строгий; ~ **avalanche criterion** строгий критерий лавинообразных изменений (*двоичная функция удовлетворяет этому критерию, если ее выходное значение меняется на противоположное с вероятностью 1/2 при изменении ровно одного из ее входов*).

strictly *adv* строго; ~ **periodic** строго периодический.

strip *v* (1) отделять; ~ **an authenticator off one message to authenticate another** отделять аутентификатор от одного сообщения, чтобы аутентифицировать другое; (2) снимать перешифровку.

strong *adj* (1) стойкий; ~ **cryptoalgorithm** стойкий криптоалгоритм; ~ **encryption** стойкое шифрование; ~**est encryption level allowed by the US government** самый высокий уровень стойкости, разрешенный правительством США; ~ **key** стойкий ключ; ~ **primes** стойкие простые числа (*два простых числа p и*

q называются стойкими, если их произведение n таково, что, если p и q неизвестны, разложение n на множители является очень трудоемкой задачей); (2) строгий; ~ **definition of security** строгое определение стойкости (*криптосистема является стойкой, если любой алгоритм атаки этой криптосистемы, обладающий не пренебрежимо малой вероятностью ее успешного вскрытия, связан с неприемлемо большим объемом вычислений*); ~ **prime** строгое простое число; ~ **pseudoprime to the base b** строгое псевдопростое число по основанию b; (3) сильный; ~ **intersymbol dependence** сильная межсимвольная зависимость.

strongly *adv* сильно; ~**-randomized cipher** шифр с высокой степенью случайности.

structure *n* структура; ~ **of a cryptosystem** структура криптосистемы.

subject *v* подвергать, испытывать; ~ **a cryptosystem to cryptanalytic attack by experts** испытывать криптосистему методом экспертного криптоанализа.

subkey *n* подключ (*ключ более низкого иерархического уровня*).

subscriber *n* абонент (*сети*); ~**'s public-key** открытый ключ абонента.

subsequence *n* подпоследовательность.

subset *n* подмножество; ~ **problem** = ~ sum problem; ~ **sum problem** задача о сумме подмножеств [задача о рюкзаке].

subspace *n* подпространство.

substitute *v* заменять; ~ **fraudulent messages for legitimate ones** заменять истинные сообщения на ложные.

substitution *n* (1) замена, подстановка (*криптографическая операция, связанная с заменой каждого символа открытого текста на другой символ из того же алфавита, при этом конкретный вид перестановки определяет секретный ключ*); ~ **box** блок замены; ~ **cipher** шифр замены; ~ **cipher module** модуль [блок] шифрования заменой; ~ **key** ключ шифра замены; ~**-permutation network** заменно-перестановочная сеть; ~ **table** таблица замен (*одномерный массив без повторений*); ~**-transposition cipher** шифр замены-перестановки (*замена предшествует перестановке*); (2) подмена; ~ **attack** попытка подмены [атака с подменой].

subtractive *n* разностный; ~ **cryptosystem** разностная криптосистема (*шифрование выполняется вычитанием ключа из открытого текста, а расшифрование — сложением шифртекста с ключом*).

successful *adj* успешный; ~ **attack** успешная (*криптоаналитическая*) атака.

succumb *v* быть уязвимым; ~ **to a chosen-ciphertext attack** быть уязвимым по отношению к атаке на основе выбранного шифртекста.

summation *n* суммирование; ~ **cipher** суммарный шифр; ~ **generator** суммирующий генератор (*объединяет выходные последовательности, полученные с*

использованием нескольких регистров сдвига, путем выполнения операции сложения).

superenciphering *n* наложение шифров, перешифрование, избыточное шифрование (*шифрование уже зашифрованного текста*).

superencipherment *n* = superenciphering.

superencrypting *n* = superenciphering.

superencryption *n* = superenciphering.

superimpose *n* накладывать; ~ **cryptograms** накладывать одну криптограмму на другую.

superincreasing *adj* сверхвозрастающий; ~ **key** сверхвозрастающий (*в процессе шифрования*) ключ; ~ **sequence** сверхвозрастающая последовательность.

superpolynomial *adj* сверхполиномиальный; ~ **cryptanalytic algorithm** сверхполиномиальный криптоаналитический алгоритм (*имеет вычислительную сложность* $O(C^{f(n)})$, *где C является константой,* $f(n)=o(n)$).

susceptible *adj* уязвимый; ~ **to a dictionary attack** уязвимый по отношению к словарной атаке.

S/WAN *abbr* ЗГКС (*см. Secure Wide Area Network*).

SWORD *prop* "Булат" (*суперкомпьютер, который использовался в 16 управлении КГБ для решения дешифровальных задач*).

syllabary *adj* слоговый; ~ **cipher** слоговый шифр; ~ **code** слоговый код (*содержит отдельные буквы и слоги вместе с соответствующими кодовыми группами, которые используются для кодирования слов и имен собственных, отсутствующих в основном коде*); ~ **square** шифр типа "слоговый квадрат" (*при зашифровании в качестве единиц открытого текста используются как отдельные буквы, так и слоги*).

syllabary *n* (1) = ~ code; (2) = ~ cipher.

syllable *n* слог; ~ **frequency** частота встречаемости [появления] слогов (*в тексте*).

symbol *n* символ; ~ **frequency** частота встречаемости [появления] символов (*в тексте*).

symmetric *adj* симметричный; ~ **cryptosystem** симметричная криптосистема; ~ **encryption** симметричное шифрование; ~ **key** симметричный ключ (*используется как для зашифрования, так и для расшифрования*).

symmetrically *adv* симметрично; ~ **encrypted** зашифрованный по одноключевому алгоритму.

synchronous *adj* синхронный; ~ **cryptooperation** синхронная криптооперация (*выполняется в реальном масштабе времени и требует синхронной работы криптографического оборудования*); ~ **stream cipher** синхронный поточный шифр.

synthesis *n* синтез; ~ **of pseudorandom number generators** синтез генераторов псевдослучайных чисел.

system *adj* системный; ~ **key** системный ключ (*используется для шифрования других ключей*); ~ **session key** системный сеансовый ключ.

system *n* система; ~**-generated key** ключ, генерируемый системой; ~ **master key** главный ключ [мастер-ключ] системы; ~ **of cryptography** криптографическая система; ~ **of linear equations in n unknowns** система линейных уравнений с n неизвестными.

systematically *adv* методично; ~ **mixed sequences** методично смешанные последовательности.

Szek *prop* Сцек А. (*чешский инженер, во время первой мировой войны работавший на радиостанции в Брюсселе и выкравший оттуда немецкий дипломатический код, который впоследствии был использован англичанами, чтобы прочесть одну из шифртелеграмм министра иностранных дел Германии Циммермана*).

T

table *n* таблица.

tactical *adj* тактический; ~ **encryption device** шифратор тактического назначения.

tamper *v* подделывать; ~**-free channel** физически защищенный канал; ~**proof** защищенный от воровства или от неумелого обращения; ~**resistant** имитостойкий; ~ **with information** подделывать информацию.

Tanneberg *prop* сражение при Таннеберге (*город в Восточной Пруссии*) между Германией и Россией в августе 1914 г. (*первая битва в мировой истории, решающее влияние на исход которой оказала несостоятельность в вопросах криптографии*).

tap *n* отвод, ответвление (*от линии связи*).

tap *v* делать отвод (*от линии связи*), подключаться (*к линии связи*).

tapped *adj* = wiretapped.

TAREX *abbr* = Target Exploitation.

target *n* цель; ~ **Exploitation** разработка цели (*поиск несекретной информации, касающейся страны, которая является объектом радиоразведывательной деятельности, а также периодическое посещение этой страны специально обученным персоналом с целью сбора открытых данных о ее коммуникациях*).

Tavares *prop* Таварес С. (*один из разработчиков блочного алгоритма шифрования KACT*).

taxonomy *n* классификация; ~ **for authentication schemes** классификация схем аутентификации.

TCSEC *abbr* КОЗВС (*см. Trusted Computing System Evaluation Criteria*).

TCU *abbr* КТ (*см. terminal cryptographic unit*).

TDM *abbr* МВД (*см. time division multiplexing*).

TEA *abbr* КАШ (*см. Tiny Encryption Algorithm*).

TECHINS *abbr* ТЕХИН (*см. technical instructions*).

technical *adj* технический; ~ **extracts of traffic** технические отрывки из переписки (*сгенерированный компьютером АНБ США отчет, содержащий характеристику всех коммуникационных сетей в мире — кому, каким образом и что именно они передают*); ~ **instructions** технические инструкции.

technique *n* метод, способ.

technology *n* технология.

TEK *abbr* КОС (см. *traffic encryption key*).

telecommunications *n pl* телекоммуникации, средства телекоммуникационной связи; ~ **systems accessible to wiretaps** телекоммуникационные системы, доступные для перехвата путем подключения к проводным линиям связи.

telegram *n* телеграмма; ~ **in cipher** шифрованная телеграмма [шифровка]; ~ **in clear** открытый текст телеграммы [незашифрованная телеграмма]; ~ **in code** кодированная телеграмма.

telegraph *n* телеграф.

telegraph *v* телеграфировать, передавать по телеграфу.

telegraphic *adj* телеграфный; ~ **code** телеграфный код; ~ **communication** телеграфная связь.

telemetry *n* телеметрия.

telex *n* телекс; ~ **tape encryptor** шифратор телексных сообщений с записью на ленту.

telltale *adj* сигнальный; ~ **emanation** сигнальное излучение (*работающей радиоэлектронной аппаратуры, несущее конфиденциальную информацию*).

temporary *adj* временный; ~ **key** временный (*временно действующий*) ключ.

terminal *adj* терминальный; ~ **key** терминальный ключ.

terminal *n* терминал (*оконечное абонентское устройство*); ~ **cryptographic unit** криптографический терминал.

Tessera *adj* "Тессера" (*первоначальное название модельного ряда PCMCIA-карт "Фортецца"*).

test *n* проверка; ~ **key** проверочный ключ.

test *v* проверять; ~ **on a bombe** проверять (*ключ немецкой шифровальной машины "Энигма"*) с помощью "бомбы".

tetragram *n* тетраграмма (*четыре символа текста*); ~ **count** маркировка тетраграмм; ~ **counting** = ~ count; ~ **frequency** частота встречаемости [появления] тетраграмм (*в тексте*).

tetragraph *n* = tetragram.

text *n* текст; ~**-dependent encryption** шифрование, зависящее от текста [шифрование на ключе, зависящем от текста]; ~**-MAC pair** пара текст-КАС.

TEXTA *abbr* ТОП (*см. technical extracts of traffic*).

theoretical *adj* теоретический; ~ **attack** теоретическая атака (*криптоаналитическая атака на шифр, которая хотя и приносит желаемый успех при наличии определенных условий, но тем не менее не реализуема на практике, поскольку в реальной жизни эти условия невозможно соблюсти*); ~ **cryptanalysis** теоретический криптоанализ; ~ **security** теоретическая стойкость.

theoretically *adv* теоретически; ~ **unbreakable** теоретически невскрываемый.

threat *n* угроза; ~ **analysis** анализ (*возможных*) угроз.

three *num* три; ~ **key method** метод трех ключей (*передача секретного ключа шифрования с использованием криптосистемы с открытым ключом*).

three- *в сложных и сложносоставных словах имеет значение* трех-; ~**-wheel Enigma** трехроторная [трехдисковая] "Энигма".

threshold *n* порог, пороговая величина; ~ **cryptosystem** пороговая криптосистема; ~ **RSA signature scheme** пороговая РША-схема цифровой подписи.

third-party *adj* третья сторона (*в дополнение к отправителю и получателю сообщений*); ~ **key** ключ, полученный от третьей стороны (*напр., из центра распределения ключей*).

time *n* время; ~ **and frequency sequential permutation** последовательная частотно-временная перестановка; ~ **complexity** временная сложность (*алгоритма*); ~**-dated message** сообщение с отметкой времени; ~ **division multiplexing** мультиплексирование с временным делением; ~**-expired key** ключ с ограничением срока действия; ~**-lock puzzle** "шарада" с временным замком; ~ **segment permutation** перестановка сегментов сигнала во времени; ~**stamp** отметка времени; ~**-varying cipher** шифр с изменяющимся во времени ключом [меняющийся во времени шифр].

timing *n* хронометраж, замер времени; ~ **attack** временная атака (*позволяет вскрывать ключи путем замера времени, которое требуется для выполнения криптографических операций*); ~ **cryptanalysis** = ~ attack.

tiny *adj* крошечный; ~ **Encryption Algorithm** Крошечный алгоритм шифрования.

TKDC *abbr* ДЦРК (*см. trusted key distribution center*).

Tolstoy *prop* Толстой С. (*советский криптоаналитик, в 1941 г. вскрывший японский "пурпурный" шифр*).

top-secret *adj* совершенно секретный.

total *adj* тотальный; ~ **encryption** тотальное шифрование (*всего потока информации*).

totally *adv* абсолютно; ~**-random key** абсолютно случайный ключ.

tractable *adj* разрешимый с полиномиальной сложностью (*о математической задаче, для которой существует алгоритм решения с полиномиальной сложностью*).

trade *adj* торговый; ~ **cipher system** "торговый" шифр (*шифр, применявшийся для засекречивания переписки между Москвой и советским торговым представительством в США*).

traffic *n* переписка, трафик, обмен сообщениями; ~ **analysis** анализ переписки; ~ **encryption key** = traffic key; ~ **padding** холостое заполнение трафика (*генерация ничего не значащих / фальшивых блоков данных и / или отдельных ничего не значащих / фальшивых элементов данных внутри подлинных блоков данных*); ~ **key** ключ обмена сообщениями.

training *adj* учебный, тренировочный; ~ **key** учебный [тренировочный] ключ (*используется для целей обучения обслуживающего персонала*).

transaction *n* транзакция, операция; ~ **key** ключ (*одной*) транзакции.

transfer *n* передача; ~ **of a secret key** передача секретного ключа.

transform *n* преобразование.

transformation *n* преобразование (*функция, отображающая пространство сообщений в пространство криптограмм*).

transformed *adj* преобразованный; ~ **ciphertext** преобразованный шифртекст.

transient *adj* (1) переменный, изменяемый; ~**-key cryptography** криптография с переменным ключом; (2) промежуточный; ~ **key** промежуточный ключ.

transition *n* переход, переключение; ~ **from clear to cipher** переход [переключение] с открытой на шифрованную связь; ~ **from clear to crypto** = ~ from clear to cipher.

translate *v* преобразовывать, переводить; ~ **into cipher** зашифровывать; ~ **out of cipher** расшифровывать.

translation *n* преобразование, перевод, переход; ~ **from one key to another** переход от одного ключа к другому.

transmission *n* передача; ~ **line** линия связи.

transmit *v* передавать; ~ **a coded message** передавать кодированное сообщение; ~ **in encrypted form** передавать в зашифрованном виде.

transmitted *adj* переданный; ~ **key** переданный ключ; ~ **message** переданное сообщение.

transmitter *v* (1) отправитель (*сообщения*); **~'s key** ключ отправителя (*сообщения*); (2) передатчик.

transparent *adj* прозрачный; **~ cryptography** "прозрачная" криптография (*не требует от пользователя знания ее принципов, не зависит от типа шифруемых данных, характеристик системы и входящих в нее устройств, не влияет на их нормальную работу*).

transparency *n* прозрачность (*инвариантность по отношению к характеристикам различных систем и устройств, а также к типу шифруемых данных*).

transport *adj* транспортный; **~ encryption** транспортное шифрование (*транзитное шифрование информации, поступающей по внешним каналам, на транспортном уровне*); **~ key** транспортный ключ (*используется для шифрования другого ключа при его передаче*).

transposition *n* (1) перестановка (*специальная криптографическая операция, при которой символы открытого текста перемешиваются, причем конкретный вид перемешивания определяет секретный ключ*); **~ cipher** перестановочный шифр; **~ mixed sequences** перестановочные смешанные последовательности; **~-substitution cipher** шифр перестановки — замены (*перестановка предшествует замене*); (2) = ~ cipher.

trapdoor *n* лазейка, потайной ход (*слабое место в шифрсистеме*); **~ cipher** шифр с потайным ходом; **~ knapsack** ранец с лазейкой; **~ one-way function** односторонняя функция с лазейкой.

trapping *n* организация лазеек [потайных ходов] (*в шифрсистеме*).

treatise *n* трактат; **~ on cryptography** трактат по криптографии; **~ on the "Enigma"** Трактат об "Энигме" (*работа А. Тьюринга, в которой на примере немецкой шифровальной машины "Энигма" поясняются различные методы, используемые в криптоанализе*).

treble *adj* = triple.

trial *n* проба; **~-and-error cryptanalysis** криптоанализ методом проб и ошибок; **~ decipherment** = ~ decryption; **~ decryption** пробное дешифрование; **~ encryption** пробное шифрование; **~ key** пробный ключ (*при криптоаналитическом исследовании криптосистемы методом проб и ошибок*).

trick *v* добиваться обманным путем; **~ subscribers into encrypting their messages with phony keys** обманным путем добиваться, чтобы абоненты шифровали свои сообщения на ложных ключах.

trigram *n* триграмма (*три символа текста*); **~ count** маркировка триграмм; **~ counting** = ~ count; **~ frequency** частота встречаемости [появления] триграмм (*в тексте*).

trigraph *n* = trigram.

trilateral *adj* трехсторонний; ~ **frequency count** трехсторонняя маркировка (*для каждой буквы текста подсчитываются три частоты встречаемости — ее самой, а также двух букв, расположенных от нее слева и справа*).

triliteral *adj* трехбуквенный; ~ **cipher system** трехбуквенная шифрсистема (*каждый символ открытого текста заменяется тремя символами шифрованного*).

triliteral *n* трехбуквенная шифрсистема.

trinome *n* три цифры, тройка цифр.

trinomic *adj* трехцифровой; ~ **cipher system** трехцифровая шифрсистема (*при зашифровании каждый символ открытого текста заменяется на три цифры*).

trinomic *n* трехцифровая шифрсистема.

triple *adj* тройной, утроенный, трехкратный; ~ **DES** "тройной" DES (*утроенное шифрование по DES-алгоритму с разными ключами*); ~ **encryption** трехкратное шифрование.

triplet *n* триплет (*три одинаковых, идущих подряд знака текста*).

Triton *prop* "Тритон" (*четырехроторная модификация немецкой шифровальной машины "Энигма", использовавшаяся ВМС Германии во время второй мировой войны*).

Trojan *adj* троянский; ~ **horse software** компьютерные программы типа "троянский конь" (*резидентные программы, предназначенные для перехвата ключей и открытых текстов, принадлежащих пользователям компьютерных систем*).

true *adj* истинный, подлинный, настоящий; ~ **key** подлинный ключ; ~ **message** настоящее сообщение; ~ **rate of a language** истинная норма языка.

truly *adv* действительно; ~ **random sequence** = really random sequence; ~ **unbreakable cipher** действительно невскрываемый шифр.

truncated *adj* усеченный; ~ **congruential generator** усеченный конгруэнтный генератор.

trunk *n* линия магистральной связи; ~ **encryption device** устройство шифрования для линий магистральной связи.

trusted *adj* доверенный, трастовый; ~ **Computing System Evaluation Criteria** Критерий оценки защищенности вычислительной системы (*американский стандарт, установленный директивой МО США № 5200.28-STD и определяющий 4 иерархических класса защищенности вычислительных систем, разрабатываемых и используемых в правительственных интересах*); ~ **key distribution center** доверенный [трастовый] центр распределения ключей; ~ **Third Party** доверенная третья сторона.

truth *n* истинность; ~ **table** таблица истинности (*двоичной функции*).

try *v* испытывать, проверять; **~-all-possible-keys cryptanalysis** криптоанализ методом перебора всех возможных вариантов ключа.

TTP *abbr* ДТС (*см. Trusted Third Party*).

tuple *n* кортеж (*набор взаимосвязанных величин*); **k-~** кортеж из k элементов.

Turing *prop* Тьюринг А. (*английский математик*); **~ machine** машина Тьюринга.

turn off *v* отключать; **~ encryption** отключать шифрование.

two- *в сложных и сложносоставных словах имеет значение* двух-.

two-key *adj* двухключевой; **~ cryptography** двухключевая криптография.

two-letter *adj* двухбуквенный.

two-part *adj* состоящий из двух частей, двухчастевой; **~code** двухчастевой код (*состоит из двух частей, одна из которых используется для кодирования исходящих сообщений, а другая — для раскодирования входящих*); **~ key** ключ из двух частей (*постоянной и сменяемой*).

two-party *adj* состоящий из двух участников; **~ key distribution protocol** протокол распределения ключей между двумя участниками обмена сообщениями.

two-way *adj* двухсторонний; **~ key** двухсторонний ключ (*используется и для зашифрования, и для расшифрования*).

type *n* тип; **~ 1 cryptographic product** криптографический продукт Типа 1 (*по классификации АНБ США*); **~ 2 cryptographic product** криптографический продукт Типа 2 (*по классификации АНБ США*).

U

UBE *abbr* НВШ (*см. unbreakable encryption*).

UGSDA *abbr* АТСДП (*см. unusually good simultaneous diophantine approximation*).

UHF *abbr* СВЧ (*см. ultra-high frequency*).

UKUSA *abbr* СКСША (*см. United Kingdom & United States of America*); **~ agreement** соглашение между СК и США.

Ultra *prop* "Ультра" (*условное наименование источника разведывательной информации, которую английские криптоаналитики добывали путем чтения немецкой шифрпереписки в годы второй мировой войны*).

ultra- *в сложных и сложносоставных словах имеет значение* сверх-.

ultra-high *adj* сверхвысокий; **~ frequency** сверхвысокая частота.

unambiguous *adj* однозначный; **~ decoding** однозначное декодирование.

unauthentic *adj* неаутентичный, фальсифицированный; **~ message** неаутентичное [фальсифицированное] сообщение.

unauthorized *adj* несанкционированный (*о действии, предпринятом без соответствующих полномочий*); ~ **access** несанкционированный доступ.

unbreakable *adj* невскрываемый; ~ **by brute force** невскрываемый с помощью атаки с применением грубой силы; ~ **cipher** невскрываемый шифр; ~ **encryption** невскрываемое шифрование; ~ **key** невскрываемый ключ.

unbreakability *n* невскрываемость (*шифра*).

unbroken *adj* невскрытый; ~ **code** невскрытый код.

uncertainty *n* неопределенность; ~ **of a secret key** неопределенность секретного ключа.

unchangeable *adj* неизменяемый, постоянный; ~ **key** неизменяемый [постоянный] ключ.

unclassified *adj* несекретный; ~ **research in cryptography** несекретные исследования в области криптографии.

uncompromised *adj* нескомпрометированный [нераскрытый]; ~ **key** нескомпрометированный [нераскрытый] ключ (*неизвестный противнику*).

unconditionally *adv* безусловно; ~ **secure authentication scheme** безусловно стойкая схема аутентификации (*ее стойкость не зависит ни от вычислительной мощности, ни от времени, которыми может̀ располагать противник*); ~ **secure cryptosystem** безусловно стойкая криптосистема (*ее стойкость не зависит ни от вычислительной мощности, ни от времени, которыми может располагать противник*).

uncrackable *adj* = unbreakable.

undecidable *adj* нерешаемый, неразрешимый (*о математической задаче*).

undecipherable *adj* (1) не поддающийся расшифрованию [расшифровке]; (2) не поддающийся дешифрованию [дешифровке].

undeciphered *adj* (1) нерасшифрованный; (2) недешифрованный.

undecrypted *adj* = undeciphered.

undeniable *adj* неоспоримый; ~ **signature scheme** схема неоспоримой цифровой подписи.

underlying *adj* исходный, соответствующий; ~ **plaintext** исходный [соответствующий] открытый текст.

undigested *adj* неотредактированный; ~ **decrypts** неотредактированные дешифровки.

unduplicatable *adj* неподдающийся копированию; ~ **key** неподдающийся копированию ключ.

unencrypted *adj* незашифрованный.

unfactorable *adj* неразлагаемый на (*простые*) сомножители.

unicity *n* единственность; ~ **distance** расстояние единственности (*количество знаков шифртекста, необходимое криптоаналитику для вскрытия шифра*); ~ **point** = ~ distance.

unidirectional *adj* = one-way.

uniform *adj* (1) равномерный; ~ **distribution** равномерное распределение; (2) однородный; ~ **protection** однородная защита.

uniformly *adv* равномерно; ~ **distributed numbers** равномерно распределенные числа.

unilateral *adj* однобуквенный; ~ **frequency distribution** распределение частот встречаемости букв (*в тексте*).

unique *adj* однозначный; ~ **decryption** (1) однозначное расшифрование; (2) однозначное дешифрование.

uniquely *adv* однозначно; ~ **reversible** однозначно обратимый.

unit *n* (1) устройство; функциональный блок; (2) подразделение.

united *adj* соединенный; ~ **Kingdom & United States** Соединенное Королевство и Соединенные Штаты; ~ **States SIGINT System** Система РТР Соединенных Штатов (*включает АНБ, ЦСБ и некоторые подразделения других разведывательных ведомств США*).

universal *adj* универсальный; ~ **key** универсальный ключ.

unkeyed *adj* не имеющий ключей; ~ **device is unclassified** без ключей устройство несекретно.

unknown *adj* неизвестный; ~ **key** неизвестный ключ.

unlock *v* (1) вскрывать; ~ **a cipher** вскрывать шифр; (2) дешифровывать; ~ **information guarded by cryptography** дешифровывать защищенную криптографическими средствами информацию.

unorthodox *adj* оригинальный, нешаблонный; ~ **linear transform** оригинальное линейное преобразование (*другое название псевдопреобразования Адамара*).

unpredictable *adj* непредсказуемый; ~ **key** непредсказуемый ключ.

unpublished *abbr* секретный; ~ **algorithm** секретный алгоритм; ~ **key** секретный ключ (*в криптосистеме с открытым ключом*).

unreadable *adj* несчитываемый, невызываемый (*из памяти компьютера или криптографического устройства*); ~ **key** несчитываемый ключ.

unscrambled *adj* расшифрованный.

untappable *adj* защищенный от несанкционированного подключения; ~ **channel** канал с защитой от несанкционированного подключения.

untrustworthy *adj* незаслуживающий доверия; ~ **participant** не заслуживающий доверия пользователь.

unusually *adv* аномально; ~ **good simultaneous diophantine approximation** аномально точные совместные диофантовы приближения.

update *v* обновлять; ~ **a key** обновлять ключ.

updated *adj* обновленный; ~ **key** обновленный ключ.

updation *n* обновление; ~ **of cryptographic keys** обновление криптографических ключей.

upgrade *v* модернизировать; ~ **cipher communications** модернизировать оборудование шифрсвязи.

upper *adj* верхний; ~ **bound** верхняя граница.

use *n* использование; ~ **of codes and ciphers** использование кодов и шифров; ~ **of encryption** использование шифрования.

use *v* использовать; ~ **the same key in both directions** использовать один и тот же ключ для осуществления связи в обоих направлениях; ~ **strong cryptography** использовать стойкую криптографию.

used *adj* использованный; ~ **key** использованный ключ.

user *n* пользователь; ~**-assigned key** назначенный пользователем ключ; ~ **authentication** аутентификация пользователя; ~**-authorized key** = ~-assigned key; ~**-defined key** = ~-assigned key; ~**-entered key** вводимый пользователем ключ; ~ **key** ключ пользователя; ~**-oriented encryption** ориентированное на пользователя [учитывающее требования пользователя] шифрование; ~**-programmable structure key** ключ с программируемой пользователем структурой; ~**-specific encryption** = ~-oriented encryption; ~**-selected key** выбранный пользователем ключ; ~**-to-**~ = end-to-end.

USSS *abbr* СРСШ (*см. United States SIGINT System*).

V

vacuum-cleaner *n* пылесос; ~ **method** метод "пылесоса" (*позволяет перехватывать спутниковые радиопередачи с помощью наземного приемного устройства*).

valid *adj* допустимый; ~ **cryptogram** допустимая криптограмма.

validate *v* объявлять подлинным.

validation *n* проверка подлинности.

variable *adj* переменный; ~**-length key** ключ с переменной длиной; ~**-size block cipher** блочный шифр с переменной длиной блока; ~**-width cipher** блочный шифр с переменной длиной блока.

variable *n* переменная, параметр.

variant *adj* непостоянный; ~ **cipher system** (1) непостоянная шифрсистема (*с неравновероятными ключами*); (2) вариантная шифрсистема (*при зашифро-*

вании каждый символ открытого текста может заменяться на разные символы шифрованного).

variant *n* = modification.

variation *n* = modification.

vector *n* вектор; ~ **space** векторное пространство.

Venona *prop* Венона (*условное наименование источника разведывательной информации, которую американские и английские криптоаналитики добывали путем чтения шифрпереписки советской разведсети на Западе в 40-е—50-е годы*); ~ **decrypts** дешифровки Веноны.

verification *n* проверка (*подлинности*), верификация.

verifier *n* верификатор, устройство верификации.

verify *v* проверять; ~ **an authenticator** проверять аутентификатор; ~ **a key** проверять подлинность ключа.

Vernam *prop* Вернам Г. (*американский инженер*); ~ **cipher** шифр Вернама.

version *n* версия; ~ **rollback attack** атака методом возврата к старой версии (*разновидность криптоаналитической атаки против шифрсистемы, в которой для устранения ошибок были сделаны необходимые исправления, однако в целях совместимости оставлена возможность использования алгоритмов и протоколов более ранних версий*).

vertical *adj* вертикальный; ~ **substitution cipher** шифр вертикальной [колонной] замены.

very large-scale integration logic *n* сверхбольшая интегральная схема.

video *n* видеосигнал; ~ **encryption** шифрование видеосигналов.

Vigenere *prop* Вижинер Б. (*французский криптограф*); ~ **cipher** шифр Вижинера; ~ **square** квадрат Вижинера; ~ **table** = ~ square.

violate *v* нарушать; ~ **secrecy or privacy** нарушать секретность или конфиденциальность.

virtual *adj* виртуальный; ~ **Matrix Encryption** виртуально-матричное шифрование.

visual *adj* визуальный, видимый; ~ **cryptography** криптография для защиты изображений.

VLSI *abbr* СБИС (*см. very large-scale integration logic*); ~ **implementation of a cryptoalgorithm** СБИС-реализация криптоалгоритма.

VME *abbr* ВМШ (*см. Virtual Matrix Encryption*).

vocoder *n* вокодер.

vocoding *n* кодирование речевых сигналов.

voice *n* телефонная [радиотелефонная] связь; ~ **cryptography** криптография для защиты телефонных переговоров; ~ **encryption device** телефонный шифратор; ~**-mail encryption** шифрование речевых сообщений в системе электронной почты; ~ **scrambler device** телефонный скремблер; ~ **security device** устройство засекречивания телефонных переговоров.

volume *n* объем; ~ **of traffic** объем трафика.

VORTEX *prop* = CHALET.

vowel *n* буква, обозначающая гласный звук; ~**-consonant relationship** взаимосвязь между буквами, обозначающими гласные и согласные звуки.

Voynich *prop* Войнич У. (*американский торговец редкими книгами, муж известной английской писательницы Э. Войнич, автора романа "Овод"*); ~ **manuscript** рукопись Войнича (*манускрипт, приобретенный Войничем в 1912 г. в Италии и содержащий шифрованный текст на неизвестном языке, который не поддается прочтению*).

vulnerability *n* уязвимость, слабость; ~ **of a cipher system** уязвимость шифрсистемы.

vulnerable *adj* уязвимый, слабый; ~ **to break-ins** уязвимый против взлома; ~ **to a chosen-ciphertext attack** уязвимый по отношению к атаке на основе выбранного шифртекста.

W

WA *abbr* ВС (*см. Wassenaar Arrangement*).

WAKE *abbr* ПШАК (*см. word auto key encryption*).

Walker *prop* Уокер Дж. (*офицер ВМС США, на протяжении 17 лет продававший советской разведке криптографические ключи и схемы американских шифраторов*).

Walsh *prop* Уолш Дж. (*американский математик*); ~**-Hadamard transform** преобразование Уолша-Адамара; ~ **transform** преобразование Уолша.

Wassenaar *prop* Вассенаар (*город в Голландии*); ~ **Arrangement** Вассенаарское соглашение (*договор, подписанный 33 странами-участницами в июле 1996 г. в Вассенаарe, о мерах по контролю над экспортом вооружений, а также технологий и товаров двойного назначения, включая стойкие средства шифрования*).

Wassernaar *prop* = Wassenaar.

weak *adj* слабый, нестойкий, вырожденный; ~ **key** нестойкий [вырожденный] ключ.

weaken *v* понижать (*стойкость шифра*); ~ **a cipher unit** понижать стойкость криптоблока.

weakened *adj* ослабленный (*по стойкости*); ~ **DES** ослабленный DES-алгоритм.

weakening *n* снижение стойкости.

weakness *n* слабость (*снижающая стойкость шифра*); ~ **of a cipher** слабость в шифре.

weather *adj* относящийся к погоде; ~ **cipher** шифр для засекречивания прогноза погоды (*применялся немецкими военными кораблями и подводными лодками во время второй мировой войны*); ~ **codebook** = ~ cipher.

weight *n* вес, весовой коэффициент.

weighted *adj* выполненный с учетом весов; ~ **addition** суммирование, выполненное с учетом весов.

well *adv* хорошо; ~**-resourced attack** атака со стороны хорошо оснащенного противника.

Wehrmacht *n нем.* "вермахт" (*сухопутные вооруженные силы Германии в 20-е—40-е годы*); ~ **Enigma** "Энигма" "вермахта".

WF *abbr* РХ (*см. work factor*).

wheel *n* колесо, ротор, шифрдиск.

whispers *n pl* шуршание (*звуки, издаваемые радиопередатчиком непосредственно перед посылкой шифрованного сообщения*).

white *adj* белый; ~ **noise** "белый шум" [зашумление]; ~**-noise-based random generator** генератор случайных чисел на основе источника "белого шума" [зашумления]; ~ **noise encryption** шифрование "белым шумом" [зашумлением].

WHT *abbr* ПУА (*см. Walsh-Hadamard transform*).

wideband *adj* широкополосный; ~ **encryption** широкополосное шифрование речи [речевых сигналов].

wired *adj* проволочный; ~ **rotor** проволочный ротор.

wireless *adj* радио-; ~ **communication** радиосвязь; ~ **conversations** переговоры по радио; ~ **interception** радиоперехват; ~ **interceptor** оператор радиоперехвата; ~ **link** радиолиния; ~ **silence** радиомолчание.

wiretap *n* перехват, прослушивание (*посредством подключения к проводному каналу связи, несанкционированного его пользователями*); ~ **attack** атака с перехватом информации посредством подключения к проводной линии связи; ~ **attack secure** защищенный от атаки с перехватом информации посредством подключения к проводной линии связи; ~ **law** закон о прослушивании.

wiretap *v* перехватывать, прослушивать (*посредством подключения к проводному каналу связи, несанкционированного его пользователями*).

wiretapped *adj* перехваченный посредством подключения к проводной линии связи; ~ **ciphertext** шифртекст, перехваченный посредством подключения к проводной линии связи.

wiretapping *n* = wiretap.

withstand *v* противостоять; ~ **cryptanalysis** противостоять криптоанализу; ~ **a targeted attack** противостоять целенаправленной атаке.

word *n* слово; ~ **auto key encryption** пословное шифрование на автоключе (*шифрование при помощи сложения автоматически генерируемых 32-битовых последовательностей со словами открытого текста по модулю 2*); ~**-length aperiodic** апериодичный шифр многоалфавитной замены с пословной сменой алфавита; ~ **pattern** распределение слов (*в тексте*); ~**-usage frequency** частота употребления слов (*в тексте*).

work *n* работа; ~ **characteristic** характеристика криптостойкости шифра [рабочая характеристика / коэффициент трудоемкости] (*объем работы, которую надо выполнить для вскрытия шифра*); ~ **characteristic reduction field** поле снижения рабочей характеристики (*часть криптограммы, где хранится информация, которая позволяет снизить криптостойкость используемого шифра*); ~ **factor** = ~ characteristic; ~ **factor reduction field** = ~ characteristic reduction field; ~ **function** = ~ characteristic; ~ **key** = working key; ~ **on a cipher system** работа над вскрытием шифрсистемы.

work *v* работать, заниматься; ~ **on the cryptanalysis of a cipher** заниматься криптоанализом шифра, работать над вскрытием шифра.

working *adj* рабочий, действующий; ~ **key** рабочий [действующий] ключ.

World-Wide Web *n* Всемирная информационная система ["Всемирная паутина"].

worst *adj в грам. знач. сущ.* наихудший, самый плохой; **at** ~ в [самом] худшем случае; ~ **case conditions** наихудшие условия (*наихудшие для разработчика шифрсистем условия, при которых может производиться ее криптоанализ: (а) полное знание этой шифрсистемы противником; (б) наличие у противника достаточного объема перехваченного шифртекста; (в) знание противником отдельных отрезков открытого и соответствующего ему шифрованного текстов*).

WRF *abbr* = work factor reduction field.

Wright *prop* Райт Э. (*американский писатель, автор романа "Гэдсби", в котором нет ни единого слова, содержащего букву "е" английского алфавита*).

wrong *adj* неправильный, несоответствующий; ~ **key** неправильный [несоответствующий] ключ.

wrongly *adv* неправильно; ~**-entered key** неправильно введенный ключ.

WT *abbr* ПУ (*см. Walsh transform*).

WWW *abbr* ВИС (*см. World-Wide Web*).

X

X-distribution *n* = chi-distribution.

XOR *abbr* = exclusive-OR.

X-text *n* зашифрованный текст, шифртекст.

Y

Yardley *prop* Ярдли Г. (*американский криптоаналитик*).

Yellow Book *prop* "Желтая книга" (*определяет правила применения критериев американской "Оранжевой книги"*).

Yurchenko *prop* Юрченко В. (*высокопоставленный сотрудник КГБ, в 1985 г. убежавший на Запад и предположительно выдавший нескольких агентов советской разведки в США, в том числе — Р. Пелтона, бывшего сотрудника АНБ*).

Z

Zendia *prop* Зендиа (*вымышленное государство, вскрытием шифров которого занимаются сотрудники АНБ США на курсах повышения квалификации*).

Zendian *adj* зендийский (*см. Zendia*); ~ **cipher** зендийский шифр.

zero *adj* нулевой; ~**-knowledge iterative proof** итеративное доказательство с нулевым разглашением (*конфиденциальной информации*); ~**-knowledge proof** доказательство с нулевым разглашением (*конфиденциальной информации*); ~**-knowledge protocol** протокол обмена с нулевым разглашением (*конфиденциальной информации*).

zero *n* нуль; ~**-one distribution** распределение нулей и единиц.

zeroization *n* обнуление (*процесс удаления ключа из криптографического оборудования или устройства ввода ключей*).

zeroize *v* обнулять (*удалять ключ из криптографического оборудования или устройства ввода ключей*).

Zimmermann *prop* (1) Циммерман А. (*германский министр иностранных дел в годы первой мировой войны*); ~ **telegram** телеграмма Циммермана (*прочитанная англичанами немецкая шифртелеграмма, содержание которой оказало существенное влияние на решение США принять участие в первой мировой войне на стороне Антанты*); (2) Циммерман Ф. (*автор программы шифрования "PGP"*).

ZKIP abbr ИДНР (см. zero-knowledge iterative proof).

Книги издательства "БХВ-Петербург" в продаже:

Серия "В подлиннике"

Айвенс К. Эксплуатация Windows NT. Проблемы и решения	592 с.
Андреев А. и др. Новые технологии Windows 2000	592 с.
Андреев А. и др. Microsoft Windows 2000 Server. Русская версия	960 с.
Андреев А. и др. Microsoft Windows 2000 Professional. Русская версия	752 с.
Андреев А. и др. Microsoft Windows 2000 Server и Professional. Русские версии	1056 с.
Бейн С. Adobe Illustrator 7.0	704 с.
Беленький Ю., Власенко С. Microsoft Word 2000	992 с.
Браун М. HTML 3.2 (с компакт-диском)	1040 с.
Вебер Дж. Технология Java (с компакт-диском)	1104 с.
Гофман В. Delphi 5 в подлиннике	800 с.
Дженнингс Р. Microsoft Access 97 (2 тома)	1270 с.
Колесниченко О, И. Шишигин. Аппаратные средства PC. 3-е издание	800 с.
Д. О'Доннел, Лэдд Э. Microsoft Internet Explorer 4	720 с.
Матросов А. и др. HTML 4.0	672 с.
Михеева В., Харитонова И. Microsoft Access 2000	1088 с.
Новиков Ф., Яценко А. Microsoft Office 2000 в целом	728 с.
Нортон П. Персональный компьютер: аппаратно-программная организация. Книга 1	848 с.
Нортон П. Windows 98	592 с.
Пилгрим А. Персональный компьютер: модернизация и ремонт. Книга 2	528 с.
Питц М., Кирк Ч. XML	736 с.
Пономаренко С. Adobe Illustrator 8.0	576 с.
Пономаренко С. CorelDRAW 9	560 с.
Пономаренко С. Adobe Photoshop 4.0	416 с.
Пономаренко С. Adobe Photoshop 5.0	512 с.
Пономаренко С. Macromedia FreeHand 7	320 с.
Тайц А. Adobe InDesign	704 с.
Тихомиров Ю. Microsoft SQL Server 7.0	720 с.
CD-ROM с примерами к книгам серии "В подлиннике" Access 2000, Excel 2000, Word 2000, Office 2000 в целом	

Серия "Мастер"

Айзекс С. Dynamic HTML (с компакт-диском)	496 с.
Анин Б. Защита компьютерной информации	384 с.
Березин С. Факсимильная связь в Windows	304 с.
Борн Г. Реестр Windows 98 (с дискетой)	496 с.

Габбасов Ю. Internet 2000	448 с.
Гарбар П. Novell GroupWise 5.5: система электронной почты и коллективной работы	480 с.
Гарнаев А. Visual Basic 6.0: разработка приложений (с дискетой)	448 с.
Гарнаев А. Microsoft Excel 2000: разработка приложений	576 с.
Гордеев О. Программирование звука в Windows (с дискетой)	384 с.
Гофман В. Работа с базами данных в Delphi	656 с.
Грей Д. Photoshop Plug-Ins. Расширение программы Adobe Photoshop (с компакт-диском)	432 с.
Дарахвелидзе П., Марков Е., Котенок О. Программирование в Delphi 5 (с дискетой)	784 с.
Дубина А. Машиностроительные расчеты в среде Excel 97/2000 (с дискетой)	416 с.
Зима В. Безопасность глобальных сетевых технологий	320 с.
Киммел П. Borland C++ 5	876 с.
Кокорева О. Реестр Windows 2000	352 с.
Краснов М. Open GL в проектах Delphi (с дискетой)	352 с.
Кулагин Б. 3D Studio Max 3.0: от объекта до анимации (с компакт-диском)	480 с.
Мешков А., Тихомиров Ю. Visual C++ и MFC, том II (с дискетой)	464 с.
Мешков А., Тихомиров Ю. Visual C++ и MFC, 2-е издание (с дискетой)	1040 с.
Миронов Д. Создание Web-страниц в MS Office 2000	320 с.
Михеева В., Харитонова И. Microsoft Access 2000: разработка приложений	832 с.
Мюррей У. Создание переносимых приложений для Windows	816 с.
Новиков Ф. и др. Microsoft Office 2000: разработка приложений	656 с.
Нортон П. Разработка приложений в Access 97 (с компакт-диском)	656 с.
Олифер В., Олифер Н. Новые технологии и оборудование IP-сетей	500 с.
Пономаренко С. InDesign: дизайн и верстка	544 с.
Приписнов Д. Моделирование в 3D Studio MAX 3.0 (с компакт-диском)	352 с.
Рудометов В. PC: настройка, оптимизация и разгон, 2-е издание	336 с.
Тайц А. Каталог Photoshop Plug-Ins	464 с.
Тихомиров Ю. Программирование трехмерной графики в Visual C++ (с дискетой)	256 с.
Чекмарев А. Средства проектирования на Java (с компакт-диском)	400 с.
Шапошников И. Интернет-программирование	224 с.
Шапошников И. Web-сайт своими руками	224 с.
Шилдт Г. Теория и практика C++	416 с.
Ресурсы Microsoft Windows NT Server 4.0	752 с.
Ресурсы BackOffice: Exchange Server и SMS (с компакт-диском)	1008 с.
Ресурсы BackOffice: SQL Server и SNA Server (с компакт-диском)	768 с.
Сетевые средства Microsoft Windows NT Server 4.0	880 с.
Создание intranet. Официальное руководство Microsoft (с компакт-диском)	672 с.
Visual Basic 6.0	992 с.

Серия "Изучаем вместе с BHV"

Березин С., Раков С. Internet у вас дома, 2-е издание	752 с.
Гарнаев А. Использование Microsoft Excel и VBA в экономике и финансах	336 с.
Культин Н. Макрокоманды Microsoft Word (с дискетой)	256 с.
Куперштейн В. Современные информационные технологии в делопроизводстве и управлении	256 с.
Тайц А. Adobe Photoshop 5.0 (с дискетой)	448 с.

Серия "Самоучитель"

Gala2000Group. Самоучитель 3D Studio Max 3.0 (с компакт-диском)	272 с.
Ананьев А., Федоров А. Самоучитель Visual Basic 6.0	624 с.
Бекаревич Ю., Пушкина Н. Самоучитель Microsoft Access 2000	480 с.
Васильев В., Малиновский А. Основы работы на ПК	448 с.
Гарнаев А. Самоучитель VBA	512 с.
Долженков В. Самоучитель Excel 2000 (с дискетой)	368 с.
Культин Н. Программирование на Object Pascal в Delphi 5 (с дискетой)	464 с.
Культин Н. Самоучитель. Программирование в Turbo Pascal 7.0 и Delphi, 2-е издание (с дискетой)	416 с.
Матросов А., Чаунин М. Самоучитель Perl	432 с.
Омельченко Л., Федоров А. Самоучитель FrontPage 2000	512 с.
Омельченко Л. Самоучитель Visual FoxPro 6.0	512 с.
Омельченко Л., Федоров А. Самоучитель Windows 2000 Professional	528 с.
Полещук Н. Самоучитель AutoCAD 2000 (с компакт-диском)	560 с.
Секунов Н. Самоучитель Visual C++ 6 (с дискетой)	960 с.
Тайц А. Самоучитель CorelDRAW 9.0	688 с.
Тайц А. Самоучитель Adobe Photoshop 5.5 (с дискетой)	544 с.
Тихомиров Ю. Самоучитель MFC (с дискетой)	640 с.
Усаров Г. Самоучитель Microsoft Outlook 2000	336 с.
Хомоненко А. Самоучитель Microsoft Word 2000	560 с.
Шилдт Г. Самоучитель C++, 3-е издание (с дискетой)	688 с.

Серия "В вопросах и ответах"

Власенко С., Маленкова А. Microsoft Word 97	336 с.

Серия "Одним взглядом"

Пономаренко С. PageMaker 6.0 для Windows 95	144 с.
Серебрянский И. Novell NetWare 4.1	160 с.

Серия "Компьютер и творчество"

Людиновсков С. Музыкальный видеоклип своими руками	320 с.
Петелин Р., Петелин Ю. Аранжировка музыки на PC	272 с.
Петелин Р., Петелин Ю. Звуковая студия в PC	256 с.
Петелин Р., Петелин Ю. Персональный оркестр в PC	240 с.
Петелин Р., Петелин Ю. Музыка на PC. Cakewalk Pro Audio 9	512 с.

Внесерийные книги

Мультимедийный учебник на CD-ROM "Изучаем Windows 98"	
Андрианов В. Автомобильные охранные системы	272 с.
Байков В. Интернет: поиск информации и продвижение сайтов	288 с.
Бекаревич Ю., Пушкина Н. MS Access 2000 за 30 занятий	512 с.
Бэйверсток Э. Книжный маркетинг	400 с.
Живайкин П. 600 звуковых и музыкальных программ	624 с.
Закарян И., Филатов И. Интернет как инструмент для финансовых инвестиций	256 с.
Культин Н. Turbo Pascal в задачах и примерах	256 с.
Мамаев Е. MS SQL Server 7.0: проектирование и реализация баз данных	416 с.
Мамаев Е. Администрирование SQL Server 7.0	496 с.
Мещеряков М. Linux: инсталляция и основы работы (с компакт-диском)	144 с.
Робачевский А. Операционная система Unix	528 с.
Рудометов В., Рудометов Е. PC: настройка, оптимизация и разгон	256 с.
Соломенчук В. Интернет: поиск работы, учеба, гранты	288 с.
Солонина А. Цифровые процессоры обработки сигналов фирмы Motorola	512 с.
Синицын И. Основы Microsoft Outlook 97	432 с.
Угрюмов Е. Цифровая схемотехника	528 с.
Успенский И. Интернет как инструмент маркетинга	256 с.
Шарыгин М. Сканеры и цифровые камеры	384 с.

Серия "Техника в Вашем доме"

Андрианов В. Средства мобильной связи	256 с.
Бородин В. Бытовые стиральные машины	224 с.
Левченко В. Спутниковое телевидение	288 с.
Миклашевский Н. Чистая вода. Бытовые фильтры и системы очистки воды	238 с.

Медицина

Карпов О., Зайцев А. Риск применения лекарств при беременности и лактации	352 с.
Корешкин И. Все способы похудения	304 с.

Книги издательства "БХВ-Петербург" можно приобрести:

Москва

1. "Библио-Глобус"	ул. Мясницкая, 6	(095) 928 8744
2. "Дом книги"	ул. Новый Арбат, 8	(095) 203 8242
3. "Дом технической книги"	Ленинский пр., 40	(095) 137 6019
4. "Кнорус"	ул. Б. Переяславская, 46	(095) 280 9106
5. "Мидикс"	Ленинский пр., 29	(095) 958 0265
6. "Мир"	Ленинградский пр., 78	(095) 152 8282
7. "Молодая Гвардия"	ул. Большая Полянка, 28	(095) 238 0032
8. "Ридас"	Новоданиловская наб., 9	(095) 954 3044
9. ТД "Москва"	ул. Тверская, 8	(095) 229 7355

Санкт-Петербург

1. Магазин при издательстве "БХВ-Петербург"	ул. Бобруйская, 4, офис 26	(812) 541 8551
2. "Веком"	пр. Славы, 15	(812) 109 0391
3. "Волшебная формула"	В.О., Наличная ул., 41	(812) 350 0324
4. "Гелиос"	пр. Большевиков, 19	(812) 5885707
5. "Дом военной книги"	Невский пр., 20	(812) 312 4936
6. "Дом книги"	Невский пр., 28	(812) 312 0184
7. Книжный клуб "Снарк"	Загородный пр., 21	(812) 1649366
8. "Книжный мир на Петроградской"	П. С., Большой пр., 34	(812) 230 9966
9. "Ланк-Маркет «Владимирский»"	Владимирский пр., 15	(812) 327 2060
10. "Ланк-Маркет «Московский»"	Бассейная ул., 41	(812) 327 0400
11. Магазин № 1 СПбГУ	В.О. Университетская наб. (Главное здание университета)	(812) 328 9691
12. "Недра"	В.О., Средний пр., 61	(812) 321 4315
13. "Подписные издания"	Литейный пр., 57	(812) 273 5053
14. "Прометей"	ул. Народная, 16	(812) 446 2209
15. "Рена"	Лесной пр., 65, корп. 13	(812) 2457039
16. "Родина"	Ленинский пр., 127	(812) 254 2104
17. "Терус"	Кондратьевский пр., 33	(812) 540 0852
18. "Техническая книга"	ул. Пушкинская, 2	(812) 164 6565
19. ЦФТ "Нарвский" Книжная ярмарка	Промышленная ул., 6	
20. "Шанс на Садовой"	ул. Садовая, 40	(812) 315 3117
21. "Энергия"	Московский пр., 189	(812) 443 4534

Города России

1. Архангельск	"Техническая книга"	ул. Воскресенская, 105	(8582) 46 3028
2. Барнаул	"Русское слово"	ул. Малахова, 61	(3852) 44 2446
3. Брянск	"Мысль"	пр. Ленина, 11	(0832) 74 1326
4. Белгород	"Школьник"	Театральный проезд, 9	(0722) 22 4322
5. Воронеж	"Светлана"	пр. Революции, 33	(0732) 55 4507
6. Вологда	"Дом книги"	ул. Мира, 38	(8172) 72 1743
7. Вологда	"Источник"	ул. Мира, 14	(8172) 72 4238
8. Гатчина	"Книги"	ул. Советская, 4/9	(271) 11259
9. Екатеринбург	"Дом книги"	ул. А. Валика, 11	(3432) 59 4200
10. Екатеринбург	"Книжный магазин № 14"	ул. Челюскинцев, 23	(3432) 53 2490
11. Ижевск	"Техника"	ул. Пушкинская, 242	(3412) 22 5764
12. Иркутск	"Иркутск книга"	ул. Лыткина, 75А	(3952) 24 5526 24 9620
13. Калининград	"ДОК"	ул. Барнаульская, 4	(0112) 43 4522
14. Калуга	"Кругозор"	ул. Калинина, 11	(0842) 57 6060
15. Красноярск	"Эрудит"	ул. Ленина, 28	(3912) 27 6250
16. Лениногорск	"Книги"	ул. Тукая, 25	(85515) 22961
17. Мурманск	"Техноцентр"	ул. Егорова, 14	(8152) 45 5568
18. Нижний Новгород	"Дом книги"	ул. Советская, 14	(8312) 44 2273
19. Нижний Новгород	"Знание"	пр. Ленина, 3	(8312) 42 6589
20. Новгород	"Прометей"	ул. Б. С.-Петербуржская, 13	(81622) 73021
21. Новомосковск	"Книги"	ул. Комсомольская, 36/14	(08762) 61265
22. Новосибирск	ТОО "Эмбер"	ул. Спартака, 16	(3832) 69 3650
23. Омск	"Магазин № 12"	пр. Маркса, 89	(3812) 40 0400
24. Пермь	"Знание"	ул. Крупской, 42	(3422) 48 1564
25. Петрозаводск	"Эхо"	пр. Ленина, 24	(8142) 77 3601
26. Псков	"Сказ"	ул. Пушкина, 1	(8112) 16 5001
27. Ростов-на-Дону	"Магазин № 26"	ул. Пушкинская, 123/67	(8632) 66 6237
28. Ростов-на-Дону	"Магазин № 4"	ул. Б. Садовая, 41	(8632) 66 8040
29. Севастополь	"Дом книги"	ул. Коминтерна, 12	(8652) 27 0956
30. Тверь	"Кириллица"	ул. Советская, 56	(0822) 33 0568
31. Тюмень	"Новинка"	ул. Республики, 155	(3452) 22 7226
32. Уфа	"Азия"	ул. Гоголя, 62	(3472) 22 5662
33. Ярославль	"Дом книги"	ул. Кирова, 18	(0852) 30 4751

Книги серии **"мастер"** написаны экспертами в области разработки и эксплуатации программных и аппаратных комплексов. Только в книгах этой серии Вы найдете систематическое и полное изложение специальных аспектов современных компьютерных технологий и сможете выбрать оптимальные и эффективные решения проблем прикладного программирования. Серия **"мастер"** предлагает незаменимый инструмент в работе, дает уникальный шанс расширить границы Ваших профессиональных возможностей, передает Вам опыт и знания специалистов высшей квалификации.

МАСТЕР

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ

Б. Анин

ЗАЩИТА компьютерной информации

Современные технологии защиты информации, детально описанные в книге, дают возможность противостоять попыткам получить несанкционированный доступ к Вашим конфиденциальным компьютерным данным. Прочитав книгу, Вы сможете:

- Оценить уровень компьютерной преступности в России и взглянуть на компьютер глазами хакера
- Научиться находить и удалять программные закладки (клавиатурные шпионы, троянские программы и т. п.)
- Усилить парольную защиту операционной системы для успешного предотвращения ее взлома
- Защитить Вашу компьютерную систему от тайного проникновения через компьютерную сеть, к которой она подключена
- Применяя технологии шифрования, сделать так, чтобы к хранимым на Вашем компьютере данным доступ имели только Вы и никто другой

ISBN 5-8206-0104-1

9 785820 601040