Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Российский государственный торгово-экономический университет»
Оренбургский филиал

В.А. ТРИПКОШ, А.Г. МАТВЕЕВ

ЭЛЕКТРОННАЯ ЦИФРОВАЯ ПОДПИСЬ В ДЕЯТЕЛЬНОСТИ ПРЕДПРИЯТИЙ И ОРГАНИЗАЦИЙ

Учебное пособие

Оренбург
2012

УДК 658 (075)
ББК 65.290с51я73
Т 67

Рекомендовано к печати Советом Оренбургского филиала РГТЭУ (протокол № 15 от 03.07.2012 г.).

Рецензенты:

В.Н. Шепель – доктор экономических наук, кандидат технических наук, профессор, зав. кафедрой управления и информатики в технических системах ФГБОУ ВПО ОГУ;

Л.Ф. Насейкина – кандидат педагогических наук, доцент, доцент кафедры программного обеспечения вычислительной техники автоматизированных систем ФГБОУ ВПО ОГУ

Трипкош, В.А.

Т 67 Электронная цифровая подпись в деятельности предприятий и организаций: учебное пособие / В.А. Трипкош, А.Г. Матвеев. – Оренбург: Издательский центр ОГАУ, 2012. – 172 с.

ISBN 978-5-88838-781-8

В учебном пособии рассмотрены правовые и организационные основы использования электронной цифровой подписи в деятельности предприятий и организаций, общие вопросы криптографии и методы шифрования, алгоритмы хеширования и электронной цифровой подписи, проблемы идентификации и аутентификации пользователей и документов в информационных системах, современные программные комплексы электронной цифровой подписи и системы защиты информации от несанкционированного доступа. Каждый раздел учебного пособия завершается перечнем контрольных вопросов и упражнений. Завершает пособие лабораторный практикум с заданиями и методикой выполнения лабораторных работ, что окажется весьма полезным при самостоятельном закреплении материала.

Учебное пособие соответствует требованиям Федерального государственного образовательного стандарта высшего профессионального образования для направления 080100 «Экономика» (профили: «Финансы и кредит», «Бухгалтерский учет, анализ и аудит» и «Экономика предприятий и организаций»). В учебном пособии приводятся компетенции, которыми должны обладать студенты, обучающиеся по этому направлению, по завершении изучения дисциплины «Электронная цифровая подпись».

ISBN 978-5-88838-781-8

Содержание

Введение……. 6
1 Правовые и организационные основы использования электронной цифровой подписи в деятельности предприятий и организаций…….. 8
1.1 Правовое регулирование отношений в области использования электронной цифровой подписи…… 8
1.1.1 Электронная цифровая подпись – понятие, правовой режим…..... 8
1.1.2 Деятельность удостоверяющего центра……. 12
1.2 Организационные основы использования электронной цифровой подписи в деятельности предприятий и организаций……. 14
1.3 Контрольные вопросы……. 19
2 Общие вопросы криптографии……. 20
2.1 Основные понятия и определения криптографии…… 20
2.1.1 Понятие криптографии и ее задачи…… 20
2.1.2 Основные определения криптографии……... 21
2.1.3 Характеристика криптографических атак…….. 23
2.2 Основы построения криптографических систем защиты информации……. 25
2.2.1 Способы реализации криптографических алгоритмов……. 25
2.2.2 Требования к криптографическим системам защиты информации…….. 25
2.2.3 Криптографические протоколы…….. 26
2.3 Контрольные вопросы……. 27
3 Общие сведения о методах шифрования……. 29
3.1 Методы шифрования с закрытым ключом…… 29
3.1.1 Пример симметричного шифра древности…… 33
3.1.2 Примеры симметричных шифров современности…… 34
3.1.3 Контрольные вопросы…….. 36
3.1.4 Упражнения……... 36
3.2 Методы шифрования с открытым ключом……... 37
3.2.1 Причины создания асимметричной криптографии…….. 37
3.2.2 Реализация асимметричных алгоритмов шифрования данных….. 38
3.2.3 Использование асимметричных алгоритмов для ключевого управления в симметричных криптосистемах…….. 40
3.2.4 Требования к алгоритмам шифрования с открытым ключом…….. 41
3.2.5 Примеры асимметричных шифров…… 42
3.2.6 Контрольные вопросы…….. 43
4 Электронная цифровая подпись на основе алгоритмов с открытым ключом……..…….. 44
4.1 Функции и свойства электронной цифровой подписи…… 44
4.2 Технология формирования и сверки электронной цифровой подписи……. 45
4.3 Контрольные вопросы……. 49

5 Алгоритмы хеширования……..……..... 50
5.1 Однонаправленные хеш-функции…….. 50
5.1.1 Основы построения хеш-функций…….. 50
5.1.2 Хеш-функции на основе симметричных блочных алгоритмов...… 52
5.2 Примеры алгоритмов хеширования……... 54
5.2.1 Алгоритмы MD2, MD4, MD5…….. 54
5.2.2 Алгоритм безопасного хеширования SHA……..... 59
5.2.3 Отечественный алгоритм хеширования ГОСТ Р34.11-94……..... 60
5.3 Контрольные вопросы……. 60
5.4 Упражнения…….. 60
6 Алгоритмы электронной цифровой подписи……... 62
6.1. Основные положения теории чисел…….. 62
6.1.1 Основная теорема арифметики……... 63
6.1.2 Взаимно простые числа и функция Эйлера……... 63
6.1.3 Арифметика остатков и теория сравнений…… 65
6.1.4 Малая теорема Ферма и теорема Ферма-Эйлера……... 67
6.1.5 Наибольший общий делитель……. 68
6.1.6 Обобщенный алгоритм Евклида……. 68
6.1.7 Инверсия по модулю m…… 69
6.2 Алгоритм цифровой подписи RSA…… 70
6.2.1 Общие сведения о системе RSA…….. 70
6.2.2 Подготовка ключей в системе RSA…… 71
6.2.3 Протокол RSA для шифрования данных…… 72
6.2.4 Вопросы практического использования алгоритма RSA……. 73
6.3 Алгоритм цифровой подписи Эль Гамаля…… 75
6.3.1 Общие сведения о системе Эль Гамаля…….. 75
6.3.2 Подготовка ключей в системе Эль Гамаля…… 75
6.3.3 Протокол Эль Гамаля для шифрования данных…… 76
6.4 Алгоритм цифровой подписи DSA…… 78
6.5 Алгоритмы ГОСТ Р34.10-94 и ГОСТ Р34.10-2001……... 80
6.6 Возможные атаки при использовании алгоритмов асимметричного шифрования……. 82
6.6.1 Атака «человек-в-середине»…… 82
6.6.2 Атака на основе выбранного открытого текста……. 83
6.7 Контрольные вопросы……. 84
6.8 Упражнения…….. 85
7 Проблемы идентификации и аутентификации пользователей и документов в информационных системах……... 86
7.1 Основные понятия и концепции защиты пользователей и документов в информационных системах…… 86
7.2 Идентификация и механизмы подтверждения подлинности пользователей в компьютерных системах…… 87
7.2.1 Механизмы парольной аутентификации…… 87
7.2.2 Механизмы аутентификации на базе аппаратных средств……. 89

7.2.3 Механизмы биометрической идентификации и аутентификации…….. 90
7.3 Идентификация и аутентификация пользователей в виртуальном пространстве сетей…….. 92
7.3.1 Взаимная проверка подлинности пользователей…….. 92
7.3.2 Протоколы идентификации с нулевой передачей знаний……... 95
7.4 Проблема аутентификации данных и электронная цифровая подпись..... 97
7.5 Контрольные вопросы……. 98
8 Современные программные комплексы электронной цифровой подписи и системы защиты информации от несанкционированного доступа…… 99
8.1 Современные системы идентификации документов и аутентификации пользователей с использованием электронной цифровой подписи……... 99
8.1.1 Технология работы с программным комплексом Pretty Good Privacy (PGP)……... 99
8.1.2. Установка, настройка, запуск и работа с программой PGP…… 102
8.2 Системы защиты информации от несанкционированного доступа…….. 114
8.2.1 Система криптографической защиты информации от несанкционированного доступа КРИПТОН-ВЕТО…….. 114
8.2.2 Комплекс КРИПТОН-ЗАМОК для ограничения доступа к компьютеру…….. 118
8.2.3 Система защиты данных Crypton Sigma……. 121
8.3 Контрольные вопросы……. 124
9 Лабораторный практикум…….. 125
9.1 Лабораторная работа № 1. Защита данных с использованием шифра Цезаря……... 125
9.2 Лабораторная работа № 2. Работа с криптосистемой PGP: генерация ключей…… 130
9.3 Лабораторная работа № 3. Работа с криптосистемой PGP: экспорт и импорт ключей……... 136
9.4 Лабораторная работа № 4. Работа с криптосистемой PGP: зашифрование и расшифрование сообщений……... 142
9.5 Лабораторная работа № 5. Работа с криптосистемой PGP: формирование и сверка электронной цифровой подписи……... 148
9.6 Лабораторная работа № 6. Работа с криптосистемой PGP: подписание ключей……. 155
9.7 Лабораторная работа № 7. Работа с криптосистемой PGP: зашифрование и расшифрование сообщений различными способами……. 159
Заключение……. 169
Литература…….. 170

ВВЕДЕНИЕ

В современных компьютерных информационных системах, используемых в сфере экономической и производственной деятельности, важную роль играют средства и системы криптографической защиты информации. Это обусловлено не только тенденциями перевода экономических и государственно-правовых отношений на «электронную основу», но и все более широким использованием во всех сферах деятельности телекоммуникационных сетей, прежде всего, открытого типа.

Среди многообразных современных средств защиты информации важное место занимают программные комплексы электронной цифровой подписи, с помощью которых существенно облегчаются процессы организации электронного документооборота, пересылки документов, подписания контрактов, организации электронных аукционов и т.п.

Рассмотрению всего комплекса вопросов связанных с электронной цифровой подписью и ее использованием в деятельности предприятий и организаций посвящено настоящее учебное пособие.

Содержание учебного пособия соответствует требованиям Федерального государственного образовательного стандарта высшего профессионального образования для направления «080100 Экономика» (профили «Финансы и кредит», «Бухгалтерский учет, анализ и аудит» и «Экономика предприятий и организаций») и будет полезным для студентов изучающих учебную дисциплину «Электронная цифровая подпись».

Целью изучения дисциплины «Электронная цифровая подпись» является освоение знаний, приобретение умений и формирование компетенций в области электронной цифровой подписи и подтверждения соответствия для профессиональной деятельности бакалавров по направлению «Экономика» и профилям: финансы и кредит; бухгалтерский учет, анализ и аудит; экономика предприятий и организаций.

Задачами дисциплины являются:

1) усвоение теоретических основ построения системы электронной цифровой подписи;
2) овладение методологическим инструментарием обеспечения целостности электронных документов и подтверждения их подлинности при обработке и передаче по каналам теледоступа в единых информационно-телекоммуникационных системах;
3) изучение правовых и организационных средств и методов использования электронной цифровой подписи в виртуальном пространстве экономической деятельности предприятий и организаций;
4) изучение методов формирования электронных документов и электронного документооборота, идентификации и аутентификации пользователей и документов в информационных инфраструктурах на основе электронной цифровой подписи, а также методов управления контролем доступа, необходимых для построения защищенных информационных систем различного назначения;

5) овладение практическими навыками работы с международной системой электронной цифровой подписи в составе программного комплекса Pretty Good Privacy (PGP);
6) ознакомление с современными программными комплексами электронной цифровой подписи и системами защиты информации от несанкционированного доступа, а также внедрения систем защиты информации в деятельность предприятий и организаций.

В результате изучения дисциплины «Электронная цифровая подпись» студенты должны овладеть следующими общекультурными и профессиональными компетенциями (ОК и ПК):

ОК-12 – способен понимать сущность и значение информации в развитии современного информационного общества, сознавать опасности и угрозы, возникающие в этом процессе, соблюдать основные требования информационной безопасности, в том числе защиты государственной тайны;

ОК-13 – владеет основными методами, способами и средствами получения, хранения, переработки информации, имеет навыки работы с компьютером как средством управления информацией, способен работать с информацией в глобальных компьютерных сетях;

ПК-10 – способен использовать для решения аналитических и исследовательских задач современные технические средства и информационные технологии;

ПК-12 – способен использовать для решения коммуникативных задач современные технические средства и информационные технологии.

Для достижения целей дисциплины «Электронная цифровая подпись» учебное пособие содержит девять разделов, в которых в логической последовательности рассматриваются: правовые и организационные основы использования электронной цифровой подписи в деятельности предприятий и организаций; общие вопросы криптографии и методы шифрования; алгоритмы хеширования и электронной цифровой подписи; проблемы идентификации и аутентификации пользователей и документов в информационных системах; современные программные комплексы электронной цифровой подписи и системы защиты информации от несанкционированного доступа.

Каждый раздел учебного пособия завершается перечнем контрольных вопросов и упражнений. Завершает пособие лабораторный практикум с заданиями и методикой выполнения лабораторных работ, что окажется весьма полезным при самостоятельном закреплении материала.

Для написания учебного пособия были использованы труды российских ученых по криптографическим методам и средствам защиты информации: Басаловой Г.В., Варлатой С.К., Шахановой М.В., Молдовян Н.А., Молдовян А.А., Рябко Б.Я., Фионова А.Н. и др.

1 ПРАВОВЫЕ И ОРГАНИЗАЦИОННЫЕ ОСНОВЫ ИСПОЛЬЗОВАНИЯ ЭЛЕКТРОННОЙ ЦИФРОВОЙ ПОДПИСИ В ДЕЯТЕЛЬНОСТИ ПРЕДПРИЯТИЙ И ОРГАНИЗАЦИЙ

1.1 Правовое регулирование отношений в области использования электронной цифровой подписи

1.1.1 Электронная цифровая подпись – понятие, правовой режим

Электронная цифровая подпись (ЭЦП) предназначена для идентификации лица, подписавшего электронный документ и является полноценной заменой (аналогом) собственноручной подписи в случаях, предусмотренных законом.

Использование электронной подписи позволяет осуществить:

– *контроль целостности передаваемого документа* – при любом случайном или преднамеренном изменении документа подпись станет недействительной, потому что вычислена она на основании исходного состояния документа и соответствует лишь ему;

– *защиту от изменений (подделки) документа* – гарантия выявления подделки при контроле целостности делает подделывание нецелесообразным в большинстве случаев;

– *невозможность отказа от авторства* – так как создать корректную подпись можно, лишь зная закрытый ключ, а он должен быть известен только владельцу, то владелец не может отказаться от своей подписи под документом;

– *доказательное подтверждение авторства документа* – так как создать корректную подпись можно, лишь зная закрытый ключ, а он должен быть известен только владельцу, то владелец пары ключей может доказать свое авторство подписи под документом. В зависимости от деталей определения документа могут быть подписаны такие поля, как «автор», «внесенные изменения», «метка времени» и т.д.

Правовое регулирование применения ЭЦП в России осуществляется на основе Гражданского Кодекса РФ и, принятого 06.04.2011 г., Федерального закона № 63-ФЗ «Об электронной подписи».

Ранее правовое регулирование отношений связанных с ЭЦП осуществлялось исключительно Федеральным законом от 10.01.2002 года № 1-ФЗ «Об электронной цифровой подписи», который перестал действовать с 01.07.2012 года (действие Федерального закона № 1-ФЗ продлено до 01.07.2013 года), однако сертификаты электронных цифровых подписей, выданные в соответствии с этим нормативным актом, можно будет использовать и дальше.

Федеральный закон № 63-ФЗ существенным образом изменяет правовое регулирование отношений, связанных с подписанием электронных документов. Изменение было вызвано тем, что Закон № 1-ФЗ не позволял ис-

пользовать в электронном документообороте иные технологии подтверждения аутентичности текста документа, кроме указанных в этом Законе. Более простые виды электронных подписей (не основанные на технологии ассиметричного шифрования, как ЭЦП) никак в правовом отношении не регулировались. В связи с этим возникал определенный правовой риск, связанный с признанием соглашения незаключенным, если итоговый электронный документ не был подписан ЭЦП.

Кроме того, изменениям способствовала неутешительная статистика, свидетельствующая о слабой распространенности ЭЦП в российском деловом обороте. По состоянию на февраль 2007 года в России было выдано около 200000 сертификатов ключа ЭЦП, что составляет лишь 0,2 процента от населения страны. В Европе же за аналогичный период времени от введения в действие Директивы ЕС от 13.12.1999 года № 1999/93/ЕС «Об общих принципах электронных подписей» (DIRECTIVE 1999/93/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 13 December 1999 on a Community framework for electronic signatures) усиленные электронные подписи использовало около 70 процентов населения.

Раннее действовавший Закон № 1-ФЗ в принципе не позволял выдавать сертификаты ЭЦП юридическим лицам – при том, что именно эти участники оборота чаще используют электронные документы. Новый Закон № 63-ФЗ устраняет это ограничение, что должно способствовать более широкому распространению электронных подписей в России.

Назначение Федерального закона № 63-ФЗ состоит в том, что он регулирует отношения в области использования электронных подписей при совершении гражданско-правовых сделок, оказании государственных и муниципальных услуг, исполнении государственных и муниципальных функций, при совершении иных юридически значимых действий.

Для целей Федерального закона № 63-ФЗ используется целый ряд *понятий и определений*, основные из которых приведены ниже.

Электронная подпись – информация в электронной форме, которая присоединена к другой информации в электронной форме (подписываемой информации) или иным образом связана с такой информацией и которая используется для определения лица, подписывающего информацию.

Ключ электронной подписи – уникальная последовательность символов, предназначенная для создания электронной подписи.

Ключ проверки электронной подписи – уникальная последовательность символов, однозначно связанная с ключом электронной подписи и предназначенная для проверки подлинности электронной подписи.

Средства электронной подписи – шифровальные (криптографические) средства, используемые для реализации хотя бы одной из следующих функций – создание электронной подписи, проверка электронной подписи, создание ключа электронной подписи и ключа проверки электронной подписи.

Участники электронного взаимодействия – осуществляющие обмен информацией в электронной форме государственные органы, органы местного самоуправления, организации, а также граждане.

Корпоративная информационная система – информационная система, участники электронного взаимодействия в которой составляют определенный круг лиц;

Информационная система общего пользования – информационная система, участники электронного взаимодействия в которой составляют неопределенный круг лиц и в использовании которой этим лицам не может быть отказано.

Другие понятия и определения из Федерального закона № 63-ФЗ будут приведены позже.

Закон № 63-ФЗ устанавливает следующие *принципы использования электронной подписи*:

1) право участников электронного взаимодействия использовать электронную подпись любого вида по своему усмотрению, если требование об использовании конкретного вида электронной подписи в соответствии с целями ее использования не предусмотрено федеральными законами или принимаемыми в соответствии с ними нормативными правовыми актами либо соглашением между участниками электронного взаимодействия;

2) возможность использования участниками электронного взаимодействия по своему усмотрению любой информационной технологии и (или) технических средств, позволяющих выполнить требования настоящего Федерального закона применительно к использованию конкретных видов электронных подписей;

3) недопустимость признания электронной подписи и (или) подписанного ею электронного документа не имеющими юридической силы только на основании того, что такая электронная подпись создана не собственноручно, а с использованием средств электронной подписи для автоматического создания и (или) автоматической проверки электронных подписей в информационной системе.

Закон № 63-ФЗ устанавливает следующие *виды электронной подписи*:

– простая электронная подпись;

– усиленная электронная подпись, которая, в свою очередь, может быть усиленной неквалифицированной электронной подписью и усиленной квалифицированной электронной подписью.

Простой электронной подписью является электронная подпись, которая посредством использования кодов, паролей или иных средств подтверждает факт формирования электронной подписи определенным лицом.

Неквалифицированной электронной подписью является электронная подпись, которая:

– получена в результате криптографического преобразования информации с использованием ключа электронной подписи;

– позволяет определить лицо, подписавшее электронный документ;

– позволяет обнаружить факт внесения изменений в электронный документ после момента его подписания;

– создается с использованием средств электронной подписи.

Квалифицированной электронной подписью является электронная подпись, которая соответствует всем признакам неквалифицированной электронной подписи и следующим дополнительным признакам:

– ключ проверки электронной подписи указан в квалифицированном сертификате;

– для создания и проверки электронной подписи используются средства электронной подписи, получившие подтверждение соответствия требованиям, установленным в соответствии с Федеральным законом № 63-ФЗ.

Этот Закон предоставляет участникам правоотношений по своему усмотрению использовать любой вид электронных подписей из числа предусмотренных Законом.

Особо оговаривается в новом Законе, что электронные подписи, созданные в соответствии с нормами иностранного права или международными соглашениями, признаются на территории России как электронные подписи того вида, под признаки которых они подпадают.

Таким образом, в Законе № 63-ФЗ установлены такие же виды электронных подписей, как и в Директиве ЕС № 1999/93/ЕС, которая предусматривает разделение электронной подписи на простой и усиленный варианты.

Закон № 63-ФЗ предполагает существование удостоверяющих центров, как это было предусмотрено и старым Законом об ЭЦП. При этом новый Закон определяет, что *удостоверяющий центр* – это юридическое лицо или индивидуальный предприниматель, осуществляющие функции по созданию и выдаче сертификатов ключей проверки электронных подписей, а также иные функции, предусмотренные настоящим Федеральным законом.

Однако новый Закон предполагает возможным создание отдельных видов электронных подписей без использования услуг удостоверяющих центров, тогда как получить ЭЦП без обращения к таким организациям ранее было невозможно.

К примеру, без обращения к услугам удостоверяющих центров можно получить и использовать простую электронную подпись. Таким же образом может быть использована неквалифицированная электронная подпись, соответствующая всем требованиям надежности, которые предъявляет новый Закон.

В Законе № 63-ФЗ предусмотрено разделение удостоверяющих центров на имеющие и не имеющие аккредитацию. Последние смогут выдавать только сертификаты ключа, а квалифицированные сертификаты будут выдавать исключительно аккредитованные удостоверяющие центры.

Таким образом, для получения квалифицированной электронной подписи необходимо обращаться в аккредитованный удостоверяющий центр.

Вместе с Законом об электронной подписи был принят и Федеральный закон от 06.04.2011 года № 65-ФЗ «О внесении изменений в отдельные законодательные акты Российской Федерации в связи с принятием Федерального закона «Об электронной подписи».

В частности в Федеральный закон от 27.07.2010 года № 210-ФЗ «Об организации предоставления государственных и муниципальных услуг» были внесены изменения в части, регулирующей использование электронной подписи при обращении в органы государственной власти и органы местного самоуправления за государственными и муниципальными услугами.

1.1.2 Деятельность удостоверяющего центра

Федеральный закон № 63-ФЗ предполагает существование *удостоверяющих центров*, как это было в старом Законе об ЭЦП. При этом в новом Законе предусмотрено разделение удостоверяющих центров *на имеющие* и *не имеющие аккредитацию*. Последние смогут выдавать только сертификаты ключа, а квалифицированные сертификаты будут выдавать исключительно аккредитованные удостоверяющие центры.

В Законе определено, что:

– *сертификат ключа проверки электронной подписи* – электронный документ или документ на бумажном носителе, выданные удостоверяющим центром либо доверенным лицом удостоверяющего центра и подтверждающие принадлежность ключа проверки электронной подписи владельцу сертификата ключа проверки электронной подписи;

– *квалифицированный сертификат ключа проверки электронной подписи* – сертификат ключа проверки электронной подписи, выданный аккредитованным удостоверяющим центром или доверенным лицом аккредитованного удостоверяющего центра либо федеральным органом исполнительной власти, уполномоченным в сфере использования электронной подписи;

– *владелец сертификата ключа проверки электронной подписи* – лицо, которому в установленном настоящим Федеральным законом порядке выдан сертификат ключа проверки электронной подписи.

Удостоверяющим центром может быть юридическое лицо (в том числе и созданное по иностранному праву) или индивидуальный предприниматель. Ограничений относительно организационно-правовой формы такого юридического лица Закон № 63-ФЗ не содержит.

Аккредитацию удостоверяющих центров проводит уполномоченный федеральный орган на добровольной основе. Аккредитация осуществляется сроком на пять лет.

Головной удостоверяющий центр, функции которого осуществляет уполномоченный федеральный орган, не подлежит аккредитации в соответствии с настоящим Федеральным законом.

Деятельность удостоверяющего центра заключается в том, что он:

1) создает сертификаты ключей проверки электронных подписей и выдает такие сертификаты лицам, обратившимся за их получением (заявителям);

2) устанавливает сроки действия сертификатов ключей проверки электронных подписей;

3) аннулирует выданные этим удостоверяющим центром сертификаты ключей проверки электронных подписей;

4) выдает по обращению заявителя средства электронной подписи, содержащие ключ электронной подписи и ключ проверки электронной подписи (в том числе созданные удостоверяющим центром) или обеспечивающие возможность создания ключа электронной подписи и ключа проверки электронной подписи заявителем;

5) ведет реестр выданных и аннулированных этим удостоверяющим центром сертификатов ключей проверки электронных подписей (далее – реестр сертификатов), в том числе включающий в себя информацию, содержащуюся в выданных этим удостоверяющим центром сертификатах ключей проверки электронных подписей, и информацию о датах прекращения действия или аннулирования сертификатов ключей проверки электронных подписей и об основаниях таких прекращения или аннулирования;

6) устанавливает порядок ведения реестра сертификатов, не являющихся квалифицированными, и порядок доступа к нему, а также обеспечивает доступ лиц к информации, содержащейся в реестре сертификатов, в том числе с использованием информационно-телекоммуникационной сети Интернет;

7) создает по обращениям заявителей ключи электронных подписей и ключи проверки электронных подписей;

8) проверяет уникальность ключей проверки электронных подписей в реестре сертификатов;

9) осуществляет по обращениям участников электронного взаимодействия проверку электронных подписей;

10) осуществляет иную связанную с использованием электронной подписи деятельность.

При этом *удостоверяющий центр обязан*:

1) информировать в письменной форме заявителей об условиях и о порядке использования электронных подписей и средств электронной подписи, о рисках, связанных с использованием электронных подписей, и о мерах, необходимых для обеспечения безопасности электронных подписей и их проверки;

2) обеспечивать актуальность информации, содержащейся в реестре сертификатов, и ее защиту от неправомерного доступа, уничтожения, модификации, блокирования, иных неправомерных действий;

3) предоставлять безвозмездно любому лицу по его обращению в соответствии с установленным порядком доступа к реестру сертификатов информацию, содержащуюся в реестре сертификатов, в том числе информацию об аннулировании сертификата ключа проверки электронной подписи;

4) обеспечивать конфиденциальность созданных удостоверяющим центром ключей электронных подписей.

В соответствии с Законом № 63-ФЗ сертификат ключа проверки электронной подписи должен содержать следующую информацию:

– даты начала и окончания срока его действия;

– фамилию, имя и отчество (если имеется) – для физических лиц, наименование и место нахождения – для юридических лиц или иную информацию, позволяющую идентифицировать владельца сертификата ключа проверки электронной подписи;

– ключ проверки электронной подписи;

– наименование используемого средства электронной подписи и (или) стандарты, требованиям которых соответствуют ключ электронной подписи и ключ проверки электронной подписи;

– наименование удостоверяющего центра, который выдал сертификат ключа проверки электронной подписи;

– иную информацию, предусмотренную настоящим Федеральным законом, для квалифицированного сертификата.

В случае выдачи сертификата ключа проверки электронной подписи юридическому лицу в качестве владельца сертификата ключа проверки электронной подписи наряду с указанием наименования юридического лица указывается физическое лицо, действующее от имени юридического лица на основании учредительных документов юридического лица или доверенности.

Сертификат ключа проверки электронной подписи прекращает свое действие:

1) в связи с истечением установленного срока его действия;

2) на основании заявления владельца сертификата ключа проверки электронной подписи, подаваемого в форме документа на бумажном носителе или в форме электронного документа;

3) в случае прекращения деятельности удостоверяющего центра без перехода его функций другим лицам;

4) в иных случаях, установленных настоящим Федеральным законом, другими федеральными законами, принимаемыми в соответствии с ними нормативными правовыми актами или соглашением между удостоверяющим центром и владельцем сертификата ключа проверки электронной подписи.

1.2 Организационные основы использования электронной цифровой подписи в деятельности предприятий и организаций

Электронная цифровая подпись может применяться предприятиями, организациями и частными лицами для:

– сдачи налоговой отчетности (налоговых деклараций) в электронном виде;

– сдачи отчетности в пенсионный фонд России (ПФР) и в фонд социального страхования (ФСС);

– получения необходимых справок и ответов на вопросы из государственных органов и организаций;

– участия в конкурсах государственного и муниципального заказов и электронных торгах, вне зависимости от места их проведения и места нахождения участника;

– для организации собственного (внутри предприятия) электронного документооборота;

– при взаимодействии с партнерами;

– решения других прикладных задач.

Электронная цифровая подпись применяется также для идентификации в системах санкционированного доступа в помещения, в информационных системах и т.п.

Возможность *сдачи отчетности* предприятиями, организациями и частными лицами по каналам связи закреплена законодательно с 2002 года Федеральным законом «Об электронной цифровой подписи» и обеспечивается использованием электронной цифровой подписи, которая является аналогом собственноручной. Вся передаваемая информация защищена от несанкционированного доступа путем использования сертифицированных средств криптографической защиты информации.

Система представления отчетности в электронном виде по телекоммуникационным каналам связи рекомендована различными органами государственной власти и открывает качественно новый этап во взаимодействии налогоплательщиков и налоговых органов, обеспечивая надежный двусторонний информационный обмен между ними.

Система позволяет передавать налоговые декларации, бухгалтерскую отчетность и иные документы, необходимые для исчисления и уплаты налогов и сборов, в электронном виде без дублирования на бумаге, используя обычный доступ к телекоммуникационным каналам связи, в том числе Интернет.

В случае представления отчетности по телекоммуникационным каналам связи предприятия, организации и частные лица (пользователи) получают следующие преимущества:

– экономия времени (нет необходимости посещать налоговую инспекцию или иное государственное учреждение и дублировать отчетность на бумажных носителях);

– снижение количества технических ошибок (перед отправкой файлы отчетности проходят обязательный контроль на правильность заполнения в соответствии с требованиями формата);

– оперативное обновление форм отчетности и средств контроля по каналам связи в случае изменения форматов;

– возможность получения общедоступной информации от органов государственной власти;

– подтверждение доставки отчетности;

– конфиденциальность (документы передаются по телекоммуникационным каналам связи в зашифрованном виде с использованием средств криптографической защиты информации (далее СКЗИ), сертифицированных уполномоченным государственным органом, ФСБ);

– оперативность обработки информации и избежание технических ошибок при вводе данных.

С развитием Интернет-технологий и законодательства о размещении госзаказа стало возможным значительно повысить эффективность всего закупочного процесса за счет использования механизма *электронных торгов*.

Вступивший в силу 1 января 2006 года Федеральный закон Российской Федерации № 94-ФЗ от 21 июля 2005 года «О размещении заказов на поставку товаров, выполнение работ, оказание услуг для государственных и муниципальных нужд» закрепил роль Интернета как одного из основных каналов взаимодействия между заказчиком и участниками размещения заказа.

С 1 января 2011 года вступили в силу изменения в Федеральный закон от 21 июля 2005 года № 94-ФЗ «О размещении заказов на поставки товаров, выполнение работ, оказание услуг для государственных и муниципальных нужд», которые устанавливают новый порядок проведения открытых аукционов в электронной форме.

Согласно названного законодательства – электронный аукцион – это аукцион, в котором поставщики осуществляют многократную подачу ценовых предложений на специализированном Интернет-сайте – электронной торговой площадке.

При этом следует указать следующие особенности электронных аукционов:

– подача ценовых предложений осуществляется через Интернет в режиме он-лайн;

– возможна многократная подача ценовых предложений;

– каждый участник аукциона может подавать собственные ценовые предложения через специальную форму и наблюдать предложения конкурентов;

– победителем аукциона на заключение государственного контракта признается участник размещения заказа, подавший наиболее низкое ценовое предложение последним (до закрытия торгов).

Юридическая значимость электронного аукциона обеспечивается электронной цифровой подписью, которую применяют участники электронного аукциона (и заказчики, и поставщики). ЭЦП означает обязательство победителя аукциона заключить государственный контракт по предложенной им цене на заранее оговоренных заказчиком в аукционной документации условиях.

Развитие информационных технологий обеспечило возможность организации собственного (внутри предприятия или организации) *электронного документооборота*.

Электронный документооборот – система ведения документации, при которой весь массив создаваемых, передаваемых и хранимых документов поддерживается с помощью информационно-коммуникационных технологий на компьютерах, объединенных в сетевую структуру, предусматривающую возможность формирования и ведения распределенной базы данных. При этом не отрицается использование бумажных документов, но приоритетным признается электронный документ, создаваемый, корректируемый и хранимый в компьютере.

Таким образом, электронный документооборот – это в первую очередь система, позволяющая решать все типовые задачи для работы с документами: регистрация и ввод документов, поиск документов, маршрутизация, создание отчетов, ведение архива и т.д. Системы электронного документооборота применяются на крупных и средних предприятиях, в государственных структурах, при работе на электронных торговых площадках.

Электронная цифровая подпись в настоящее время широко используется при ведении *защищенной переписки* и *защиты персональных данных*.

Персональные данные – любая информация, относящаяся к определенному или определяемому на основании такой информации физическому лицу, в том числе его фамилия, имя, отчество, год, месяц, дата и место рождения, адрес, семейное, социальное, имущественное положение, образование, профессия, доходы и т.д.

Защита персональных данных – комплекс мер технического, организационного и организационно-технического характера, направленных на защиту сведений, относящихся к субъекту персональных данных.

В соответствии требованиям Федерального Закона от 27 июля 2006 года № 152-ФЗ «О персональных данных» должны быть приведены как новые, так и уже существующие информационные системы, обрабатывающие персональные данные. Под требования этого закона попадают практически все организации – коммерческие, государственные, общественные, – которые имеют кадровые службы и владеют персональными данными своих клиентов и сотрудников.

Согласно требованиям Федерального Закона № 152-ФЗ и постановлению Правительства РФ от 17.11.2007 года № 781 в информационных системах персональных данных должно быть обеспечено:

– предотвращение несанкционированного доступа к персональным данным и (или) передача их лицам, не имеющим права доступа к такой информации;

– недопущение воздействия на технические средства автоматизированной обработки персональных данных, в результате которого может быть нарушено их функционирование;

– возможность незамедлительного восстановления персональных данных, модифицированных или уничтоженных вследствие несанкционированного доступа к ним;

– постоянный контроль за обеспечением уровня защищенности персональных данных.

Услуги по защите персональных данных, в том числе в части использования ЭЦП, направлены на снижение рисков карательных санкций со стороны регулирующих органов, определенных в законе № 152-ФЗ – ФСБ России, ФСТЭК России и Роскомнадзора.

В соответствии с названными сферами применения электронной цифровой подписи используются следующие виды ЭЦП:

– ЭЦП для электронных торгов;

– ЭЦП для бухгалтерской и налоговой отчетности;

– ЭЦП для личного пользования (зашифрованная переписка);

– ЭЦП для отчетности в ПФР;

– ЭЦП для отчетности в ФСС.

При этом комплект ЭЦП при выдаче ее удостоверяющим центром выглядит следующим образом:

– открытый ключ на бумажном носителе;

– закрытый и открытый ключи на защищенном носителе – портативном устройстве, выполненном в форме USB-брелка, обеспечивающем хранение конфиденциальной ключевой информации и аутентификацию пользователя;

– дистрибутив программы Крипто-Про CSP – Криптопровайдер, позволяющий осуществлять криптографические операции в операционных системах Microsoft. Он предназначен для авторизации и обеспечения юридической значимости электронных документов при обмене ими между пользователями посредством использования процедур формирования и проверки ЭЦП в соответствии с отечественными стандартами ГОСТ. Данное программное обеспечение устанавливается на одно рабочее место. Также выдается лицензия на одно рабочее место.

Срок действия ЭЦП всех удостоверяющих центров составляет один год. Поэтому каждый год ЭЦП необходимо перевыпускать.

Если закрытый ключ генерируется в удостоверяющем центре, то уполномоченный сотрудник в обязательном порядке предъявляет документ, удостоверяющий личность (как правило, паспорт) и передает администратору центра полный пакет документов, требуемый его регламентом. После этого администратор генерирует закрытый и открытый ключи клиента в специально оборудованном помещении. Далее администратор выдает единственный ключ, который должен обязательно выдаваться на защищенном носителе, и регистрирует сертификат, то есть вносит его в реестр действующих ключей.

Для обеспечения безопасного хранения ключей ЭЦП защищенный носитель оснащен системой безопасности:

– пин-кодом;

– защитой от взлома;

– защитой от несанкционированного копирования изнутри, например, во время подписания.

Но часто используются незащищенные носители, с которых ключи легко копируются, также есть возможность использования ЭЦП с компьютера, если она установлена на нем. С точки зрения безопасности, это в корне не правильно, возникает риск завладения ЭЦП третьими лицами. Кроме того, владелец ЭЦП может скопировать свою подпись и с защищенного носителя.

Как показывает практика, руководители предприятий и организаций предпочитают получать ЭЦП на свое имя, но, как правило, самостоятельно не работают на торговых площадках и копируют подпись для своих сотрудников. Следует учитывать, что, если подпись копируется для нескольких подчиненных, в случае судебного разбирательства документы, подписанные данными ЭЦП, будут иметь полную юридическую силу. Соответственно, если кто-то подписал какой-либо документ ЭЦП руководителя, руководитель

несет полную ответственность за действия, которые совершаются с помощью этой подписи. Например, если секретарь или иной сотрудник по незнанию или по злому умыслу подписали какой-либо договор или контракт, то он имеет полную юридическую силу.

В этом случае:

во-первых, непонятно, кто и где подписывает документы;

во-вторых, появляется возможность действий со стороны третьих лиц от имени организации без ведома руководства.

Правильнее выдать обычную бумажную доверенность уполномоченному лицу на получение еще одной ЭЦП, после получения которой он работает на торговой площадке от своего имени. Если допущена какая-либо ошибка, сразу понятно, кто ее совершил. В доверенности указываются все полномочия, которыми наделен сотрудник: например, он может совершать все действия по подаче заявки и участию в аукционе, но не имеет права подписывать государственный контракт. Таким образом, сотрудник может совершать все действия по участию в размещении заказа, но право подписи контракта остается только у руководителя предприятия.

1.3 Контрольные вопросы

1) Назначение электронной цифровой подписи и ее свойства.

2) Юридическое значение ЭЦП.

3) Правовые акты, по которым осуществляется регулирование применения ЭЦП в России.

4) Назначение и основное содержание Федерального закона № 63-ФЗ.

5) В чем причины отмены Федерального закона № 1-ФЗ?

6) Основные принципы использования электронной подписи согласно Закона № 63-ФЗ.

7) Виды электронной подписи согласно Закона № 63-ФЗ.

8) Виды и назначение удостоверяющего центра.

9) Требования, предъявляемые к удостоверяющему центру.

10) Понятие и виды сертификатов ключа подписи.

11) Функции удостоверяющего центра.

12) Обязательства удостоверяющего центра.

13) Сферы применения ЭЦП предприятиями и организациями.

14) Сущность и преимущества применения ЭЦП для сдачи отчетности предприятиями, организациями и частными лицами по каналам связи.

15) Сущность и преимущества применения ЭЦП при организации электронных торгов.

16) Сущность и преимущества применения ЭЦП при организации электронного документооборота.

17) Сущность и преимущества применения ЭЦП при ведении защищенной переписки и защиты персональных данных

18) Виды ЭЦП в зависимости от сфер применения.

19) Комплект ЭЦП при выдаче ее удостоверяющим центром.

2 ОБЩИЕ ВОПРОСЫ КРИПТОГРАФИИ

Процессы формирования и сверки ЭЦП в электронных документах осуществляются с помощью криптографических методов и, реализующих эти методы, средств.

Об этом прямо отмечается в Законе № 63-ФЗ, например:

«Средства электронной подписи – шифровальные (криптографические) средства, используемые для реализации хотя бы одной из следующих функций – создание электронной подписи, проверка электронной подписи, создание ключа электронной подписи и ключа проверки электронной подписи»;

«Неквалифицированной электронной подписью является электронная подпись, которая получена в результате криптографического преобразования информации с использованием ключа электронной подписи…»;

«Квалифицированной электронной подписью является электронная подпись, которая соответствует всем признакам неквалифицированной электронной подписи и…».

Следовательно, для изучения ЭЦП и процессов связанных с ней необходимо рассмотреть общие вопросы криптографии, ее задачи и методы.

2.1 Основные понятия и определения криптографии

2.1.1 Понятие криптографии и ее задачи

На протяжении всей истории человечества актуальными были и остаются сейчас вопросы защиты информации при ее передаче между абонентами. Особенно важна защита информации для экономических и финансовых систем использующих методы и средства электронной коммерции.

До последнего времени изобретено множество способов, позволяющих в той или иной мере скрыть смысл передаваемых сообщений от посторонних лиц.

На практике разработано несколько групп методов защиты секретных посланий. Среди таких методов, как *физическая защита материального носителя информации*, *стеганографическая защита информации*, особое место занимает *криптографический метод* как наиболее надежный и распространенный в наши дни.

Криптографический метод защиты информации предполагает преобразование информации для сокрытия ее смысла от третьих лиц.

Криптография в переводе с греческого означает «тайнопись». В настоящее время криптография занимается поиском и исследованием математических методов преобразования информации.

Защита информации очень важна для финансовых систем использующих методы и средства электронной коммерции. Поэтому для понимания принципов применения технологий Интернета для защиты сетей необходимо рассмотреть основы криптографии.

Вместе с криптографией развивается и совершенствуется *криптоанализ* – наука о преодолении криптографической защиты информации. Криптоаналитики (их часто называют взломщиками) исследуют возможности расшифровывания информации без знания ключей. Успешно проведенный криптоанализ позволяет получить ключ шифрования, или открытый текст, или то и другое вместе.

Часто криптографию и криптоанализ объединяют в одну науку – *криптологию* (kryptos - тайный, logos - наука), занимающуюся вопросами обратимого преобразования информации с целью защиты от несанкционированного доступа, оценкой надежности систем шифрования и анализом стойкости шифров.

В настоящее время криптография применяется во многих сферах, таких как:

– обслуживание банковских пластиковых карт;

– сдача бухгалтерских и иных отчетов через удаленные каналы связи;

– банковское обслуживание предприятий через локальную или глобальную сеть;

– шифрование данных при передаче по открытым каналам связи (например, зашифрование в целях безопасности таких сведений как адрес, телефон, номер кредитной карты при совершении покупки в Интернет-магазине);

– хранение и обработка паролей пользователей в сети;

– безопасное от несанкционированного доступа хранение данных на жестком диске компьютера и др.

Долгое время криптографические методы применялись лишь для шифрования данных с целью защиты от несанкционированного доступа. В последние два-три десятилетия, в связи с развитием телекоммуникационных систем и технологий, интерес к криптографии значительно возрос. Благодаря достижениям новых криптографических методов расширился и спектр задач криптографии, это:

– собственно шифрование данных с целью защиты от несанкционированного доступа;

– проверка подлинности сообщений: получатель сообщения может проверить его источник;

– проверка целостности передаваемых данных: получатель может проверить, не было ли сообщение изменено или подменено в процессе пересылки;

– обеспечение невозможности отказа, то есть невозможности как для получателя, так и для отправителя отказаться от факта передачи.

2.1.2 Основные определения криптографии

Криптография использует целый ряд терминов и определений. Рассмотрим основные из них.

Открытым текстом называют сообщение, которое может быть прочитано, осмыслено и понято без каких-либо специальных мер.

Исходные сообщения обычно являются открытыми текстами. В иностранной литературе для открытого текста используют термин plaintext.

Любое сообщение (документ) может быть записано с помощью двоичного алфавита, то есть с использованием только нулей и единиц. При такой записи на машинный носитель говорят об электронном документе.

Зашифрованием называют процесс искажения открытого текста в целях его недоступности для понимания.

Шифртекстом (шифрованным сообщением, закрытым текстом, криптограммой, криптографическим образованием) называют результат зашифрования открытого текста.

В иностранной литературе для закрытого текста используют термин ciphertext.

Расшифрованием называют противоположный процесс по обращению шифртекста в его исходный вид открытый текст.

В англоязычной литературе терминам «зашифрование / расшифрование» соответствуют термины «enciphering/deciphering».

На рисунке 2.1 с помощью этих понятий поясняется содержание одной из задач криптографии – шифрование данных с целью защиты от несанкционированного доступа при передаче по открытым каналам связи.

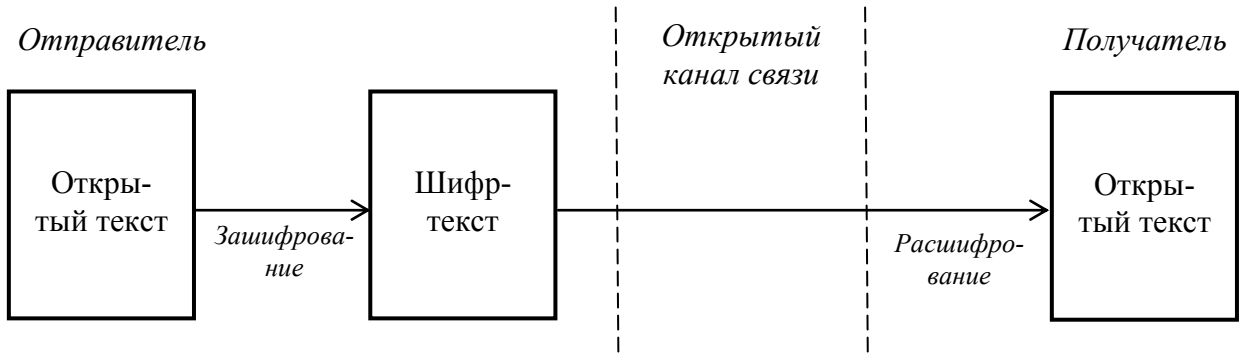


Рисунок 2.1 – К пояснению содержания задачи шифрования данных с целью защиты от несанкционированного доступа

Центральными в криптологии являются понятия *«шифр»* и *«ключ»*.

Шифр (криптографический алгоритм) – это совокупность заранее оговоренных способов преобразования исходного открытого сообщения с целью его защиты и, наоборот, преобразования шифртекста в исходный открытый текст.

Ключ – информация, необходимая для зашифрования и расшифрования сообщений.

Для зашифрования открытого текста шифр используется в сочетании с ключом. Одним шифром, но разными ключами одно и то же сообщение будет преобразовываться в разные шифртексты.

Криптографические образования могут быть стойкими или слабыми.

Криптостойкостью называется характеристика шифра, определяющая его стойкость к взлому (дешифрованию) без знания ключа (т.е. способность противостоять криптоанализу).

Криптографическая стойкость измеряется тем, сколько понадобится времени и ресурсов, чтобы из шифртекста восстановить исходный открытый текст без предварительного знания шифра и ключа.

При стойкой криптографии образуются шифртексты, которые исключительно сложно взломать без обладания определенными инструментами по расшифрованию.

Защищенность шифртекста зависит от секретности ключа и стойкости криптоалгоритма (ниже будет показано, что в большей мере важна секретность ключа). Вместе с тем и стойкость криптоалгоритма и секретность ключа сегодня в значительной степени регулируются вычислительными возможностями компьютеров.

Таким образом, с учетом сделанных выше определений можно дать более точное определение науке «криптография» – *криптография* изучает построение и использование систем шифрования, в том числе их стойкость, слабости и степень уязвимости относительно различных методов взлома.

Криптосистема (система шифрования, шифрсистема) – это любая система, которую можно использовать для обратимого изменения текста сообщения с целью сделать его непонятным для всех, кроме тех, кому оно предназначено.

В настоящее время криптосистемы реализуют с помощью средств вычислительной техники. В состав таких криптосистем входят криптоалгоритмы, ключи и увязывающие их протоколы, реализованные в программах для вычислительных машин.

2.1.3 Характеристика криптографических атак

В процессе обработки (хранения, передачи и преобразования) информация подвергается воздействию различных *атак* со стороны противников (оппонентов, конкурентов, врагов и т.д.).

При этом основными нарушениями безопасности информации являются:

– потеря конфиденциальности, то есть раскрытие информационных ценностей, не подлежащих огласке;

– потеря целостности, то есть модификация без разрешения автора;

– потеря доступности, то есть неавторизованная потеря доступа к этим ценностям.

Различают пассивные и активные атаки.

Атака, при которой противник не имеет возможности изменять передаваемые сообщения, называется *пассивной*. В этом случае возможно лишь прослушивание передаваемых сообщений, анализ трафика и их дешифрование.

Атака, при которой противник имеет возможность модифицировать передаваемые сообщения, а также добавлять свои сообщения, называется *активной*.

Атака со стороны противника предпринимается для того, чтобы осуществить криптоанализ и взлом зашифрованных данных. Это невозможно

сделать без учета особенностей текстов сообщений, которые подлежат зашифрованию.

Обычно в криптоанализе используются такие характеристики, как:

– повторяемость букв, пар букв (биграмм) и *n*-грамм;

– сочетаемость букв друг с другом;

– чередование гласных и согласных букв и др.

Названные характеристики определяются на основе изучения достаточно объемных текстов.

Поэтому криптографические атаки принято классифицировать по количеству и типу информации, доступной для криптоанализа. Данный признак определяет следующие виды атак:

– *атака по шифртексту*. В этом случае противник имеет для анализа различные шифртексты, зашифрованные с помощью одного и того же ключа. Цель аналитиков состоит в добыче открытого текста как можно большего числа сообщений для вычисления ключа, использованного при зашифровании. Вычисленный ключ затем будет использован для дешифрования других сообщений;

– *атака по известному открытому тексту*. В этом случае, получив в свое распоряжение какие-либо открытые тексты раннее переданных шифртекстов, криптоаналитик сопоставляет их, пытаясь узнать тайный ключ, чтобы с его помощью дешифровать все последующие сообщения. Реальность такой атаки обусловлена тем, что на практике всегда можно получить кусочки открытого текста и шифртекста, например, по формату перехваченного зашифрованного файла (форматы содержат определенные стандартные заголовки или фрагменты);

– *атака по выбранному открытому тексту*. В этом случае имеется возможность не только использовать полученные пары «открытый текст – шифртекст", но и сформировать нужные тексты и зашифровать их с помощью того ключа, который требуется узнать.

С учетом особенностей названных видов атак понятно стремление разработчиков первых криптосистем делать свои алгоритмы шифрования стойкими только к атакам по шифртексту, а стойкость к атакам по открытому или выбранному тексту обеспечивать организационными мерами. Например, держать в секрете устройство шифровальной техники, алгоритмы шифрования, проверять на надежность специалистов, имеющих доступ к криптосистемам.

По мере развития криптологии специалисты поняли, что надежная криптосистема должна оставаться защищенной и в условиях, когда противнику стал известен алгоритм шифрования. В этом случае секретность ключа должна быть достаточной, чтобы обеспечить стойкость шифртекста к попыткам взлома. Этот принцип впервые сформулировал в 1883 году А. Керкхоффс и он обычно называется *принципом Керкхоффса*.

Именно такой подход используют разработчики современных криптосистем, обеспечивая возможность этим системам противостоять атакам по выбранному тексту.

2.2 Основы построения криптографических систем защиты информации

2.2.1 Способы реализации криптографических алгоритмов

До изобретения шифровальной техники зашифрование и расшифрование данных выполнялось вручную. Специалист-шифровальщик обрабатывал исходное сообщение посимвольно и таким образом получал зашифрованный текст. Однако такой подход допускает возможность ошибки шифрования. После изобретения шифровальной техники (механических шифровальных машин) процесс шифрования данных стал автоматизированным.

Кроме того, это снизило вероятность ошибок и ускорило процесс зашифрования и расшифрования. В дальнейшем появились сначала электромеханические, а затем электронные криптографические устройства.

Если процесс зашифрования и расшифрования данных осуществляется специальными электронными схемами по установленным логическим правилам, то такой способ реализации криптографического алгоритма называется *аппаратным*. Следует отметить, что аппаратная реализация криптографического алгоритма характеризуется высокой производительностью, простотой в использовании, надежностью.

Причиной появления *программных* реализаций криптографических алгоритмов явилось массовое внедрение средств вычислительной техники, и в первую очередь персональных компьютеров.

Основным преимуществом программных реализаций систем шифрования является их гибкость, то есть возможность быстрой настройки и замены криптоалгоритмов. Кроме того, программные реализации систем криптографической защиты характеризуются меньшей стоимостью. Однако, они не лишены и недостатков, в частности им свойственно значительно меньшее быстродействие по сравнению с системами, реализованными аппаратно.

Существуют и *комбинированные* системы шифрования, так называемые программно-аппаратные средства. В этом случае ЭВМ дополняется аппаратным вычислительным блоком, специально ориентированным на выполнение криптографических операций. Этот блок принято называть криптографическим сопроцессором. Меняя программное обеспечение криптографического сопроцессора, можно изменять алгоритм шифрования.

Обычно комбинированные системы шифрования объединяют в себе достоинства программных и аппаратных средств.

2.2.2 Требования к криптографическим системам защиты информации

К криптосистемам защиты информации в настоящее время предъявляются следующие наиболее общие требования:

– шифртекст должен расшифровываться только при наличии ключа;

– знание третьей стороной алгоритма шифрования не должно как-либо влиять на стойкость шифртекста;

– надежная защита информации должна обеспечиваться при зашифровании любым из допустимых ключей;

– криптографический алгоритм должен реализовываться как программно, так и аппаратно.

На практике не всегда удается выполнить все эти требования полностью. Однако характеристики современных криптосистем, в основном, удовлетворяют перечисленным требованиям.

2.2.3 Криптографические протоколы

В настоящее время в криптологии, кроме создания и исследования шифров, особое внимание уделяется разработке криптографических протоколов. *Криптографический протокол* – это процедура взаимодействия двух или более абонентов с использованием криптографических средств, в результате которой легитимные абоненты, в отличие от противников, достигают своей цели – обмена конфиденциальной информацией. В основе такого протокола лежит совокупность правил, определяющих использование криптографических методов и алгоритмов по преобразованию информации при ее зашифровании и расшифровании.

Любой криптографический протокол предназначен для решения определенной задачи и характеризуется рядом свойств:

– в ходе реализации протокола требуется строго соблюдать порядок действий;

– протокол должен быть конкретным и непротиворечивым;

– протокол должен быть всецело полным, то есть в нем должны быть предусмотрены все действия для любой возможной ситуации.

Исходя из свойств криптографического протокола следует, что участники информационного обмена конфиденциальной информацией должны знать протокол и выполнять полностью его этапы. При этом не исключается, что участники протокола могут не доверять друг другу, следовательно, криптографические протоколы должны обеспечивать защиту не только от третьей стороны, но и от ошибочных действий или мошенничества партнеров.

Все криптографические протоколы условно можно разделить на прикладные протоколы и примитивные. *Прикладной протокол* решает конкретную задачу защиты данных, которая может возникнуть на практике. Обычно разработку прикладных протоколов осуществляют из *примитивных* протоколов, приблизительно так же, как программисты создают свои программы с использованием библиотечных функций.

Наиболее распространенными примитивными протоколами являются:

1) *Протоколы конфиденциальной передачи сообщений*. Они решают задачу передачи секретной информации между двумя абонентами сети связи в условиях, когда эту сеть может прослушивать противник. Протоколы этого типа появились раньше других, так как названная задача является исторически первой задачей, которая решалась криптографией.

Примером простейшей реализации такого протокола может быть следующая последовательность действий:

– участники обмена конфиденциальной информацией выбирают систему шифрования;

– участники обмена договариваются о ключе шифрования;

– один абонент зашифровывает исходное открытое сообщение с помощью ключа выбранным методом и получает шифртекст;

– шифртекст пересылается второму абоненту по открытому каналу связи;

– второй абонент расшифровывает шифртекст с помощью ключа, получая при этом открытое сообщение.

2) *Протоколы аутентификации и идентификации*. Они предназначены для предотвращения доступа к некоторой конфиденциальной информации нелегитимных лиц, а также предотвращения доступа легитимных пользователей к тем ресурсам, на которые у них нет полномочий.

3) *Протоколы распределения ключей*. Они необходимы для обеспечения ключами участников обмена секретной зашифрованной информацией.

4) *Протоколы электронной цифровой подписи*. Они предназначены для внедрения в электронные документы цифрового реквизита, по свойствам аналогичного рукописной подписи или печати на бумажных документах. Электронная цифровая подпись позволяет проверить авторство и целостность (отсутствие фальсификации) электронного документа.

5) *Протоколы обеспечения неотслеживаемости* («электронные деньги»). Они предназначены для работы с электронными платежными средствами и обеспечивают невозможность отслеживания источника пересылки информации.

Следует иметь в виду, что чаще всего на практике криптографические атаки направлены не на алгоритм шифрования, а на протокол. Так при наличии надежного криптоалгоритма нет гарантии полной безопасности участникам секретного обмена, если применяется криптографический протокол, содержащий ошибки, допускающие возможность взлома противником или нечестных действий участвующих сторон.

Именно поэтому при разработке современных криптографических протоколов они тщательно анализируются на предмет отсутствия изъянов и стойкости к криптоатакам.

2.3 Контрольные вопросы

1) Причины реализации системы ЭЦП на основе криптографических методов.

2) Определения криптографии, криптоанализа, криптологии.

3) Сферы применения криптографии.

4) Основные понятия и определения криптографии: открытый текст, шифртекст, зашифрование, расшифрование.

5) Основные понятия и определения криптологии: шифр, ключ, криптостойкость, криптоситема.

6) Понятие и разновидности криптографических атак.

7) Классификация и характеристика криптографических атак.

8) Формулировка принципа Керкхоффса.

9) Способы реализации криптографических алгоритмов и их характеристика.

10) Основные требования к криптографическим системам защиты информации.

11) Понятие криптографического протокола, свойства и разновидности протоколов.

12) Характеристика протокола конфиденциальной передачи сообщений.

13) Характеристика протокола аутентификации и идентификации.

14) Характеристика протокола электронной цифровой подписи.

3 ОБЩИЕ СВЕДЕНИЯ О МЕТОДАХ ШИФРОВАНИЯ

В криптографии все методы преобразования информации с целью защиты от несанкционированного доступа делятся на две большие группы:

– методы шифрования с закрытым ключом (симметричные методы);

– методы шифрования с открытым ключом (асимметричные методы).

3.1 Методы шифрования с закрытым ключом

Шифрование с закрытым ключом (шифрование с секретным ключом или симметричное шифрование) используется с далекой древности.

Это классическая или одноключевая криптография. Она опирается на использование симметричных алгоритмов шифрования, в которых зашифрование и расшифрование отличаются только порядком выполнения и направлением некоторых этапов. Так симметричные алгоритмы используют один и тот же секретный ключ как для прямого действия – зашифрования, так и для второго действия – расшифрования, которое является простым обращением первого.

Схема процессов симметричного шифрования приведена на рисунке 3.1.

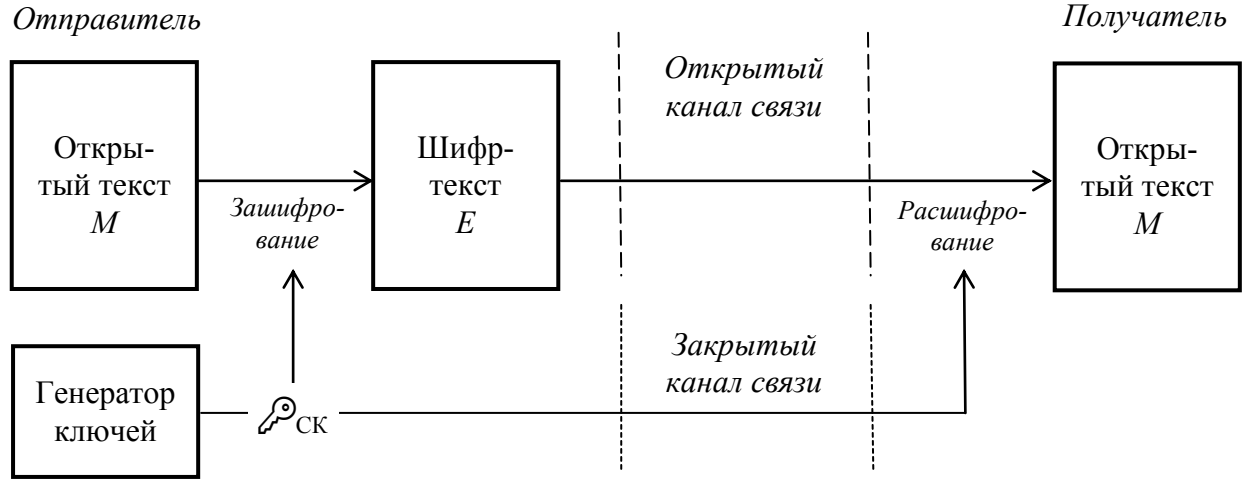


Рисунок 3.1 – К пояснению процессов симметричного шифрования данных

Отправитель имеет в своем распоряжении генератор ключей. После генерации и выбора конкретного секретного ключа $\mathcal{K}_{СК}$, среди всех возможных ключей данной системы, этот ключ на предварительном этапе передается Получателю. Причем ключ $\mathcal{K}_{СК}$ передается по закрытому каналу связи (например, специальным курьером). Отправитель формирует некоторое сообщение (открытый текст *M*), которое затем зашифровывается с использованием выбранного секретного ключа. В результате зашифрования получается криптограмма (шифртекст *E*). Далее криптограмма *E* передается Получателю по открытому, незащищенному каналу связи (например, радиоканал или

компьютерная сеть). Получатель криптограмму E с помощью ключа $\mathcal{K}_{СК}$ расшифровывает и получает исходное сообщение в открытом тексте M.

Если M – сообщение, K – секретный ключ $\mathcal{K}_{СК}$, а E – шифртекст, то математически рассмотренный процесс описывается выражением

$$E=f(M,K).$$

То есть зашифрованное сообщение E является некоторой функцией от исходного сообщения M и ключа K.

Известны разные методы симметричного шифрования (рисунок 3.2). Так на практике часто используются алгоритмы замены, перестановки, комбинированные и другие методы.

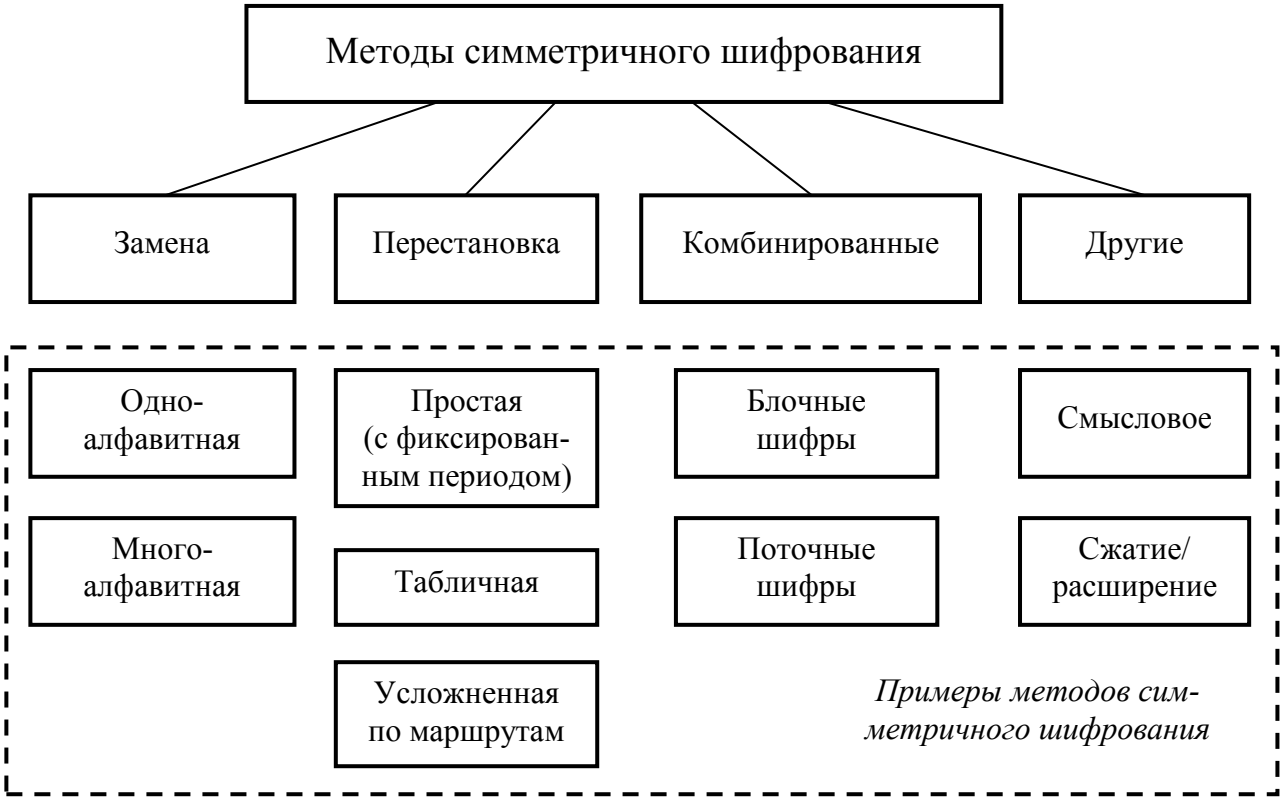


Рисунок 3.2 – Методы симметричного шифрования

В *методах замены* (или подстановки) символы исходного открытого текста заменяются некоторыми эквивалентами зашифрованного текста.

Так одним из важных подклассов методов замены являются *одноалфавитная* (или моноалфавитная) замена. В этом случае устанавливается однозначное соответствие между каждым i-м знаком исходного алфавита сообщения M и соответствующим i-м знаком шифртекста E. Одноалфавитная подстановка иногда называется также простой заменой, так как является самым простым алгоритмом замены. Примером одноалфавитной замены является шифр Цезаря, который будет рассмотрен ниже.

К одноалфавитным методам замены относятся также *пропорциональные* (монофонические) шифры, в которых уравнивается частота появления символов в шифртексте для его защиты от раскрытия с помощью частотного

анализа. Для символов, встречающихся часто (например, в русском языке – буквы О, Е, А, Т, Н, С), используется относительно большое число возможных эквивалентов. Для реже используемых исходных символов (например, буквы Ъ, Ф) может оказаться достаточным один или два эквивалента. Эквиваленты символов для открытого текста при его зашифровании выбираются либо случайному закону, либо определенным образом (например, по порядку).

Для маскирования естественной частотной статистики исходного языка на практике применяется *многоалфавитная* замена, которая также бывает нескольких видов. В многоалфавитных подстановках для замены символов исходного открытого текста используется не один, а несколько алфавитов. Обычно в алфавитах для замены символы записываются в другом порядке. Примером многоалфавитной замены может служить алгоритм, основанный на использовании таблицы Вижинера (эта таблица представляет собой квадратную матрицу с числом элементов NxN, где N – количество символов в алфавите). Этот метод был описан французом Блезом Вижинером в «Трактате о шифрах», вышедшем в 1585 году.

Еще одним примером метода многоалфавитной замены является *шифр с бегущим ключом* или *книжный шифр*. В этом случае один текст используется в качестве ключа для шифрования другого текста. При ручной криптографии в качестве ключа для шифра с бегущим ключом обычно выбирали какую-нибудь толстую книгу. При механической криптографии для автоматизации процесса выполнения многоалфавитных подстановок широко применялись роторные шифровальные машины. Среди таких машин самым известным устройством являлась немецкая шифровальная роторная машина Энигма (лат. Enigma – загадка), использовавшаяся во время второй мировой войны.

Другим частным случаем многоалфавитной замены является *гаммирование*. В этом методе шифрование выполняется путем сложения символов исходного текста и ключа по модулю, равному числу букв в алфавите. Если в исходном алфавите, например, русском – *33* символа, то сложение производится по модулю *33*. Такой процесс сложения исходного текста и ключа называется в криптографии наложением гаммы.

Правило гаммирования обычно определяют выражением

$$z = x + k \ (mod\ N),$$

где z – закодированный символ, x – исходный символ, k – ключ, N - количество символов в алфавите, а сложение по модулю N – операция, аналогичная обычному сложению, с тем отличием, что если обычное суммирование дает результат, больший или равный N, то значением суммы считается остаток от деления его на N.

Например, если сложить по модулю 33 символы *Д* (его значение *5*) и *Э* (значение *31*), то получится

$$3 = 5 + 31 \ (mod\ 33).$$

То есть в результате получают символ *В*, соответствующий числу *3*.

На практике, в условиях электронных криптосистем, наиболее часто встречается двоичное гаммирование. В этом случае используется двоичный алфавит (символы *0* и *1*), а сложение производится по модулю два. Операция сложения по модулю *2* часто обозначается знаком ⊕, то есть можно записать

$$z = x + k \ (mod\ 2) = x \oplus k.$$

Операция сложения по модулю два в алгебре логики также называется «*исключающее ИЛИ*» (по-английски *XOR*).

Особенностью гаммирования по модулю *2* является то, что нужно использовать одну и ту же операцию как для зашифрования, так и для расшифрования. Следовательно, можно использовать один и тот же алгоритм, а соответственно и одну и ту же компьютерную программу, как для шифрования, так и для расшифрования данных. Кроме того, операция сложения по модулю *2* очень быстро выполняется на ЭВМ, поэтому гаммирование даже очень больших исходных текстов осуществляется практически мгновенно. Названные достоинства обусловили широкое применение метода гаммирования в современных криптосистемах.

В *методах перестановки* символы исходного открытого текста меняются местами друг с другом по определенному правилу.

Примером простейшей перестановки является *перестановка с фиксированным периодом d*. В этом случае сообщение *M* делится на блоки по *d* символов и в каждом блоке производится одна и та же перестановка. Правило, по которому производится перестановка, является ключом и обычно задается некоторой перестановкой первых *d* натуральных чисел. В результате сами буквы исходного сообщения не изменяются, но в шифртексте передаются в другом порядке. Например, для *d = 5* в качестве ключа перестановки можно взять *34152*. Тогда в каждом блоке из *5* символов 3-й символ становится на первое место, 4-й – на второе, 1-й – на третье, 5-й – на четвертое и 2-й – на пятое.

Другим примером алгоритмов перестановки является *перестановка по таблице*. В этом случае осуществляется запись исходного открытого текста по строкам некоторой таблицы и чтение его по столбцам этой же таблицы. Последовательность заполнения строк и чтения столбцов может быть любой и обычно задается ключом.

Программным и аппаратным путем можно реализовать и другие методы перестановки. Например, при шифровании двоичных данных удобно использовать аппаратный блок, который перемешивает определенным образом с помощью электрического монтажа цепей биты исходного n-разрядного сообщения. Получатель для расшифрования сообщения применяет другой блок, который восстанавливает порядок электрических цепей.

С целью повышения криптостойкости шифртекст, зашифрованный с помощью одного метода (например, замены), может быть еще раз зашифрован с помощью другого метода (например, перестановки). В этом случае получается *комбинированный* шифр. В настоящее время применяемые на практике *блочные* или *поточные* симметричные шифры также относятся к ком-

бинированным, так как в них используется несколько операций для зашифрования сообщений.

3.1.1 Пример симметричного шифра древности

Примером симметричного криптоалгоритма древности является *шифр Цезаря*, который представляет собой простейшую систему одноалфавитной замены. Считается, что знаменитый римский император, живший в первом веке до нашей эры, использовал этот метод в своей переписке.

Шифр Цезаря применительно к русскому языку реализуется следующим образом. Каждая буква сообщения заменяется на другую, которая в русском алфавите отстоит от исходной на три позиции дальше. Например, буква *А* заменяется на букву *Г*, другие буквы – аналогично. Последние буквы алфавита заменяются циклическим переходом к его началу, например, буква *Э* заменяется на букву *А*. Таким образом, ключом в данном методе является сам алгоритм.

К примеру, слово *ЛЕГИОН* после зашифрования методом Цезаря превратится в слово *ОЗЁЛСР*.

Это достаточно простой метод и шифртексты, полученные таким образом, легко взламываются, так как можно получить информацию по количеству слов в исходном сообщении и по анализу повторов букв в зашифрованном сообщении. Тем не менее, шифр Цезаря служит примером одной из первых криптосистем.

Для усовершенствования шифра Цезаря можно расширить алфавит за счет включения знаков препинания и пробелов. Такое увеличение алфавита позволяет замаскировать длину каждого отдельного слова. Кроме того, усовершенствование шифра Цезаря можно осуществить за счет применения правила Керкхоффса, то есть использовать секретный ключ. В этом случае надо буквы сдвигать не на три знака вправо, а на n знаков, причем для русского алфавита $0 < n < 33$. О ключе, то есть о значении n, абоненты могут каким-либо образом договариваться (например, лично) и иногда менять значение ключа. Так как n может принимать разные значения, знание одного только шифра не позволит противнику расшифровать секретное сообщение.

Шифр Цезаря даже с учетом названных усовершенствований является слабым, с точки зрения криптостойкости, методом. Шифртексты, полученные таким образом, легко вскрываются путем последовательного перебора всех возможных ключей (так называемый, метод «грубой силы»). Один из вариантов расшифрования будет содержать исходный текст. Если это единственное слово, имеющее смысл, то это и есть исходное сообщение.

Однако, вывод о единственности решения может быть обоснован только когда исходное сообщение составлено на одном из естественных языков (в нашем примере – русском) и содержит более пяти-шести знаков. Если же сообщение очень короткое или состоит из цифр, то возможных решений может быть несколько, и найти правильное из них достаточно трудно.

Возможность взлома зашифрованных сообщений на естественном языке обусловлена тем, что они подчиняются многочисленным правилам. Так разные буквы и их сочетания имеют различные вероятности и, к тому же многие наборы букв вообще запрещены. Это свойство называется избыточностью текста. Во-многом, при атаках по шифротексту, именно избыточность позволяет взламывать криптографические образования, созданные с использованием шифра Цезаря или других симметричных алгоритмов. Если же имеется возможность проводить атаку по открытому тексту, то есть взломщик располагает парами «открытый текст – шифртекст», то раскрытие становится совсем простым как в случае текстов на естественном языке, так и в случае сообщений из наборов цифр.

Таким образом, вероятность успешного взлома зависит от многих факторов:

– от метода шифрования;

– от длины перехваченного шифртекста;

– от языка и алфавита исходного сообщения.

Технология шифрования с закрытым ключом (симметричная криптография) требует, чтобы участники зашифрованной переписки имели доступ к одному и тому же ключу. Это необходимо, так как отправитель использует ключ для зашифрования, а получатель применяет этот же ключ для расшифрования сообщения.

Следовательно, возникает проблема безопасной передачи этого ключа. Поэтому глобальная проблема всех симметричных криптосистем состоит в сложности выполнения процедуры, которую принято называть *ключевым управлением* – как доставить ключ получателю без риска, что его перехватят?

Всем системам симметричного шифрования присущи следующие основные недостатки:

– во-первых, принципиальной является надежность канала передачи ключа второму участнику секретных переговоров. Другими словами, ключ должен передаваться по секретному каналу;

– во-вторых, к службе генерации ключей предъявляются повышенные требования, обусловленные тем, что при схеме взаимодействия абонентов «каждый с каждым» зависимость числа ключей от числа абонентов является квадратичной.

Однако, несомненным достоинством симметричных криптосистем является то, что алгоритмы шифрования с закрытым ключом используют ключи не очень большой длины и могут быстро шифровать большие объемы данных.

3.1.2 Примеры симметричных шифров современности

Наиболее известным и широко распространенным симметричным шифром является шифр DES (*Data Encryption Standard*), разработанный фирмой IBM в 1976 году. Этот шифр рекомендован Национальным бюро стандартов США к использованию в открытых секторах экономики.

В самом общем виде криптографический протокол шифрования данных в соответствие с *алгоритмом DES* работает следующим образом:

– данные представляются в цифровом виде и разбиваются на блоки длиной 64 бита, затем поблочно шифруются;

– каждый блок разбивается на левую и правую части;

– на первом этапе шифрования вместо левой части блока записывается правая, а вместо правой – сумма по модулю 2 (операция XOR) левой и правой частей;

– на втором этапе шифрования по определенной схеме выполняются побитовые замены и перестановки.

Ключ шифра DES имеет длину 64 бита, из которых 56 битов - случайные, а 8 - служебные, используемые для контроля ключа.

Алгоритм DES реализуется в двух режимах:

– ECB (Electronic Code Book) и

– CBC (Cipher Block Chaining).

Режим CBC отличается от обычного (режим ECB) тем, что перед шифрованием очередного блока к нему применяется операция «исключающее ИЛИ» с предыдущим блоком.

Для повышения надежности алгоритма DES используется его модификация – *Triple DES* (тройной DES). Существует несколько вариантов Triple DES. Наиболее простой – перешифрование. В этом случае открытый текст зашифровывается на первом ключе, полученный шифртекст – на втором, и, в заключение, данные, полученные после второго шага, – на третьем ключе. Все три ключа выбираются независимо друг от друга.

В настоящее время шифр DES сменяет шифр AES (*Advanced Encryption Standard*).

Алгоритм IDEA (*International Data Encryption Algorithm*) – еще один пример симметричного блочного шифра. Длина ключа шифра IDEA 128 бит. Этот европейский стандарт предложен в 1990 году. Алгоритм IDEA по скорости и стойкости к криптоанализу не уступает алгоритму DES.

Алгоритм CAST – это симметричный блочный шифр, использующий 128-битовый ключ в США и 40-битный – в экспортном варианте. Алгоритм CAST применяется компанией Northern Telecom (Nortel).

Шифр *Skipjack* разработан Агентством национальной безопасности США (National Security Agency – NSA) в рамках проекта Capstone. Шифр использует 80-битовые ключи. Проект Capstone включает четыре основных компонента:

– шифр Skipjack;

– алгоритм цифровой подписи на базе стандарта DSS (*Digital Signature Standard*);

– хеш-функцию на базе алгоритма SHA (*Secure Hash Algorithm*);

– микросхему, реализующую все вышеизложенное.

Шифры *RC2* и *RC4* разработаны Роном Рейвестом – одним из основателей компании RSA Data Security, и запатентованы этой компанией. Шифр RC2 – блочный, с длиной блока 64 бита, а шифр RC4 – поточный.

Эти шифры применяют ключи разной длины, а в экспортируемых продуктах заменяют алгоритм DES. Считается, что производительность RC2 и RC4 не меньше, чем у алгоритма DES.

3.1.3 Контрольные вопросы

1) Поясните общую схему симметричного шифрования.

2) Что общего имеют все методы шифрования с закрытым ключом?

3) Назовите основные группы методов шифрования с закрытым ключом.

4) Сформулируйте общие принципы для методов шифрования заменой (подстановкой).

5) В чем заключаются многоалфавитные подстановки?

6) Приведите пример шифра одноалфавитной замены.

7) Что такое монофонические шифры?

8) Приведите примеры шифров перестановки.

9) Опишите алгоритм любого метода шифрования перестановкой. Приведите пример шифрования некоторого сообщения этим методом. Каков алгоритм расшифрования в этом методе?

10) К какой группе методов шифрования с закрытым ключом относится метод с использованием таблицы Вижинера? Каковы алгоритмы шифрования и расшифрования в этом методе? Приведите пример шифрования некоторого сообщения этим методом.

11) Каким образом можно зашифровать и расшифровать сообщение методом табличной перестановки, если размер шифруемого сообщения не кратен размеру блока?

12) Поясните сущность шифра Цезаря. Приведите пример шифрования некоторого сообщения этим методом.

13) Дайте краткую характеристику шифрам Data Encryption Standard (DES), Triple DES (тройной DES).

14) Дайте краткую характеристику шифрам International Data Encryption Algorithm (IDEA), Skipjack, RC2 и RC4.

3.1.4 Упражнения

1) Пусть исходный алфавит содержит следующие символы:

АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ

Зашифруйте с помощью шифра Вижинера и ключа ЯБЛОКО сообщения КРИПТОСТОЙКОСТЬ и ГАММИРОВАНИЕ.

2) Пусть исходный алфавит состоит из следующих знаков (символ «_» (подчеркивание) будем использовать для пробела):

АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ_

Расшифруйте сообщения ШВМБУЖНЯ и ЯБХЪШЮМХ, зашифрованные с помощью шифра Вижинера и ключа ОРЕХ.

3) Первый байт фрагмента текста в шестнадцатеричном виде имеет вид A5. На него накладывается по модулю два 4-х битовая гамма 0111 (в двоичном виде). Что получится после шифрования?

4) Первый байт фрагмента текста, зашифрованного методом гаммирования (по модулю 2), в шестнадцатеричном виде имеет вид 9A. До шифрования текст имел первый байт, равный 74 (в шестнадцатеричном виде). Какой ключ использовался при шифровании?

5) Зашифруйте методом перестановки с фиксированным периодом d = 6 с ключом 436215 сообщения ЖЕЛТЫЙ_ОГОНЬ и МЫ_НАСТУПАЕМ.

6) Расшифруйте сообщения СЛПИЬНАЕ и РОИАГДВН, зашифрованные методом перестановки с фиксированным периодом d = 8 с ключом 64275813.

7) Определите ключи в системе шифрования, использующей перестановку с фиксированным периодом d = 5 по парам открытых и зашифрованных сообщений МОЙ ПАРОЛЬ – ЙПМ ООЬАЛР и СИГНАЛ БОЯ – НИСАГО ЛЯБ.

8) Зашифруйте сообщения ШИРОКОПОЛОСНЫЙ УСИЛИТЕЛЬ (ключ: 41235) и ПЕРЕДАЧА ИЗОБРАЖЕНИЯ (ключ: 24513) методом перестановки по таблице 5*5. Ключ указывает порядок считывания столбцов при шифровании.

9) Расшифруйте сообщения ЕАУПД_КЕАЗАРЧВ (ключ: 4123) и А_НСЫИЛБСАЛЙГ (ключ: 3142), зашифрованные методом перестановки по таблице 4*4 (символ подчеркивания заменяет пробел). Ключ указывает порядок считывания столбцов при шифровании.

10) Расшифруйте указанные сообщения:

353214764134136759136762849754128212350354035767106216753211

351 761756130532128759353134758105757213101752352763211762

3.2 Методы шифрования с открытым ключом

3.2.1 Причины создания асимметричной криптографии

На практике использование симметричных криптосистем сталкивается с двумя достаточно серьезными проблемами. Первая проблема заключается в сложности осуществления процедуры безопасной доставки секретных ключей всем участникам информационного обмена. Второй проблемой является сложность идентификации и обеспечения подтверждения подлинности абонентов при электронном общении.

В условиях широкого распространения электронного документооборота и электронной коммерции требуются эффективные методы подтверждения авторства и подлинности электронных документов, одновременно позволяющие решить проблему безопасного распространения ключей.

Во второй половине семидесятых годов XX века были разработаны принципиально новые подходы, позволяющие решить названные выше задачи. В основе этих подходов лежит использование так называемых *асиммет-*

ричных криптоалгоритмов или методов шифрования с открытым ключом. Эти методы реализуют процедуры зашифрования и расшифрования данных разными ключами.

Считается, что асимметричные методы первыми разработали Уитфилд Диффи (W. Diffie) и Мартин Хеллман (M. Hellman) в 1975 году. Однако, существуют доказательства того, что Британской секретной службой CESG (Национальным центром Объединенного Королевства по официальному применению криптографии) эта проблема была решена несколько раньше У. Диффи и М. Хеллмана, но это хранилось под грифом «совершенно секретно» и никак не использовалось. Первой же гражданской реализацией концепции криптографии с открытым ключом было изобретение Ральфа Меркла (R. Merkle), получившее название «Головоломок Меркла» (было описано им в качестве курсовой работы в 1974 году), но к практическим целям эта схема была неприменима.

В ассиметричных алгоритмах один из двух ключей является *открытым* (public key) и должен быть доступен всем участникам информационной системы, а второй – *закрытым* (private key) и должен держаться в секрете. Ключи хотя и математически связаны между собой, но не дают возможности по одному ключу определить другой.

Назначение криптографической системы определяет, какой из двух ключей (открытый или закрытый) и для каких процессов (зашифрования или расшифрования) используется.

Так в задачах связанных с передачей конфиденциальной информации между абонентами сети связи открытые ключи используются для зашифрования информации, а закрытые – для расшифрования сообщений. В задачах связанных с подтверждением авторства и подлинности электронных документов с помощью ЭЦП закрытые ключи используются для создания электронной подписи, а открытые – для ее сверки.

Кроме названных задач, в настоящее время асимметричные алгоритмы широко применяются на практике:

– для обеспечения информационной безопасности телекоммуникационных сетей;

– для обеспечения защиты информации в глобальной сети Интернет;

– в банковских и платежных системах использующих различные схемы платежей.

3.2.2 Реализация асимметричных алгоритмов шифрования данных

Асимметричная криптография позволяет решить проблему снабжения участников информационного обмена ключами шифрования, так как закрытый ключ должен использоваться только для расшифрования данных и должен быть известен только одному лицу – его владельцу.

Согласно асимметричной схеме, любой кто владеет открытым ключом (а этот ключ распространяется свободно) может зашифровать сообщение. Расшифровать же сообщение сможет только владелец закрытого ключа.

Протокол обмена конфиденциальными электронными сообщениями между двумя пользователями (Отправителем и Получателем) по схеме открытого шифрования предполагает выполнение следующих действий:

1) На предварительном этапе Получатель посылает Отправителю свой открытый ключ $\mathcal{O}_{\text{ОК-П}}$ по любому (в том числе, открытому) каналу связи, например, по электронной почте.

2) Отправитель зашифровывает свое сообщение *M* открытым ключом Получателя $\mathcal{O}_{\text{ОК-П}}$, при этом формируется шифртекст *E*.

3) Отправитель пересылает криптограмму *E* Получателю.

4) Получатель расшифровывает полученное сообщение *E* своим закрытым ключом $\mathcal{O}_{\text{ЗК-П}}$, при этом формируется исходный открытый текст сообщения *M*.

Схема реализации этого протокола приведена на рисунке 3.3.

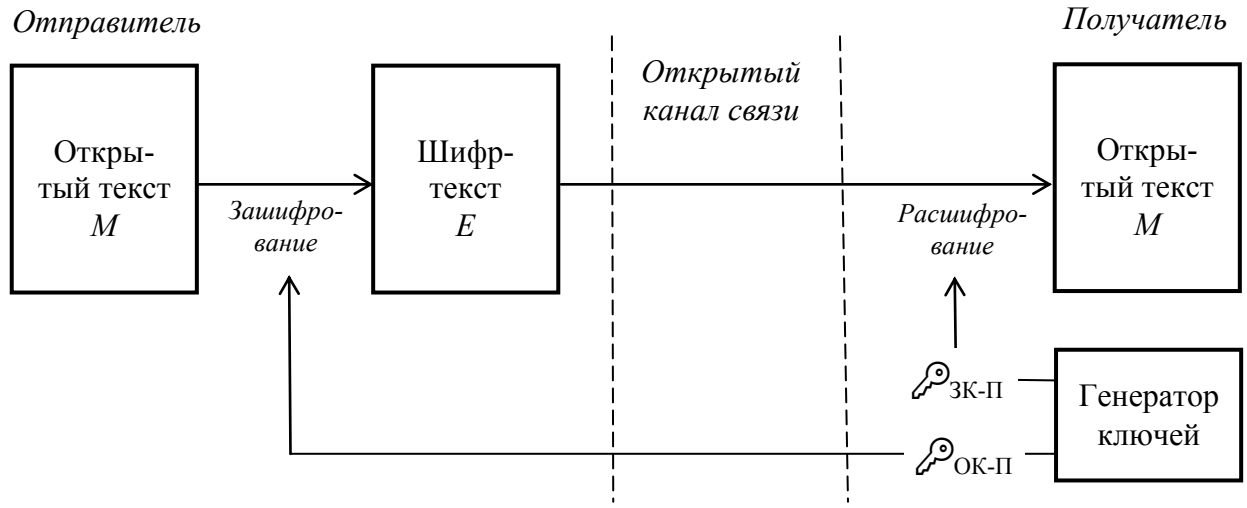


Рисунок 3.3 – К пояснению процессов асимметричного шифрования данных

Как было отмечено выше использование ассиметричных криптосистем снимает проблему распределения ключей. При симметричной криптографии пользователи перед обменом шифрограммами должны были каким-либо надежным образом (лично или через курьера) по закрытому каналу связи согласовывать используемый секретный ключ. Если один из участников информационного обмена решил изменить ключ, он должен был как-то сообщить об этом и передать новый ключ своему абоненту. При асимметричной криптографии абоненты не должны заботиться о возможности компрометации ключа. Участники системы связи могут свободно обмениваться открытыми ключами и зашифрованными с их помощью сообщениями. Более того, сам факт получения от другого абонента его открытого ключа является приглашением к секретным переговорам. Необходимость в передаче закрытого ключа вовсе отсутствует.

В информационных системах на основе компьютерных сетей для упрощения процедуры распределения ключей обычно используется база данных (например, на сервере), в которой хранятся открытые ключи всех пользователей. При необходимости любой участник системы может запросить из базы открытый ключ нужного абонента и использовать этот ключ для зашифрования ему сообщений.

3.2.3 Использование асимметричных алгоритмов для ключевого управления в симметричных криптосистемах

При всех названных выше преимуществах алгоритмов с открытым ключом им присущ и недостаток – относительно невысокая скорость зашифрования и расшифрования больших объемов данных.

Основная причина такого положения дел состоит в том, что основной операцией в асимметричных криптосистемах является операция возведения в степень по большому модулю (обычно, 500-1000 разрядному двоичному числу). ЭВМ выполняет эту операцию намного медленнее, чем операцию замены или перестановки при шифровании того же объема данных симметричными способами.

Поэтому на практике асимметричные алгоритмы редко используются для непосредственного шифрования сообщений.

Однако, в случае работы с короткими блоками данных (в том числе, ключами определенной длины), алгоритмы шифрования с открытым ключом используются достаточно эффективно.

Так может быть реализована *комбинированная* или *гибридная схема* – асимметричный алгоритм применяют в процессе создания участниками обмена согласованного сеансового секретного ключа, а затем этот ключ используют для зашифрования и расшифрования сообщений симметричным методом.

Протокол формирования сеансового секретного ключа $\mathcal{K}_{ССК}$ в этом случае может быть следующим:

1) Пользователь А получает открытый ключ Пользователя Б $\mathcal{K}_{ОК\text{-}ПБ}$ из центра распределения ключей или непосредственно от Пользователя Б.

2) Пользователь А генерирует случайным образом сеансовый секретный ключ $\mathcal{K}_{ССК}$ и зашифровывает его полученным открытым ключом.

3) Зашифрованный сеансовый секретный ключ $\mathcal{K}_{Зш\ ССК}$ пересылается пользователю Б.

4) Пользователь Б расшифровывает его своим закрытым ключом $\mathcal{K}_{ЗК\text{-}ПБ}$.

5) Пользователи А и Б используют согласованный сеансовый секретный ключ $\mathcal{K}_{ССК}$ для обмена шифрованными сообщениями.

Схема реализации этого протокола приведена на рисунке 3.4.

Процесс шифрования небольшого по размеру блока данных (сеансового секретного ключа) выполняется достаточно быстро и не замедляет работу телекоммуникационных сетей.

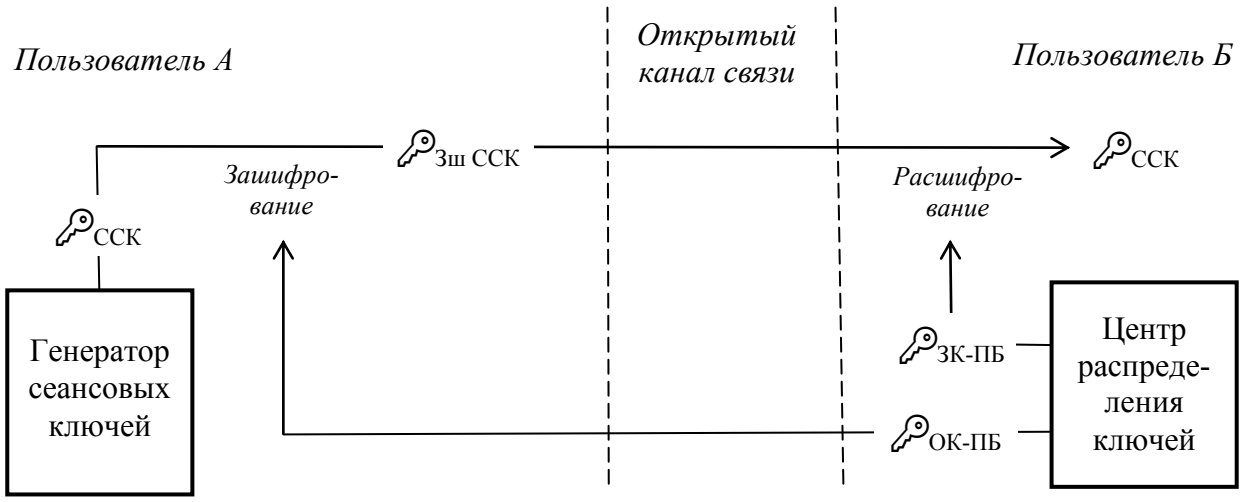


Рисунок 3.4 – К пояснению процесса формирования согласованного сеансового секретного ключа

На практике существуют и более сложные протоколы распределения ключей, обеспечивающие, кроме собственно распределения ключей, еще и взаимное подтверждение подлинности участников сеанса связи и другие требования.

3.2.4 Требования к алгоритмам шифрования с открытым ключом

Основные качества, которым должны отвечать алгоритмы шифрования с открытым ключом, определяются следующими требованиями:

– возможностью вычислительно легко создавать ключевую пару (открытый ключ, закрытый ключ);

– возможностью вычислительно легко зашифровывать сообщение открытым ключом;

– возможностью вычислительно легко расшифровывать сообщение, используя закрытый ключ;

– невозможностью вычислительно определить закрытый ключ, зная соответствующий открытый ключ;

– невозможностью вычислительно из зашифрованного сообщения восстановить исходное сообщение, зная только открытый ключ.

С учетом этих требований следует отметить, что алгоритмы шифрования с открытым ключом в большей степени базируются на свойствах математических функций, чем алгоритмы симметричных криптосистем, использующие в основном операции замены и перестановки.

Математический аппарат асимметричной криптографии, в основном, разработали американские ученые У. Диффи (W. Diffie), М. Хеллман (M. Hellman), Р. Меркль (R. Merkle).

Асимметричные методы шифрования основаны на использовании так называемых односторонних функций. *Односторонней функцией* (one-way function) называется математическая функция, которую относительно легко вычислить, но практически невозможно найти по значению функции соот-

ветствующее значение аргумента. Другими словами, зная *x* легко вычислить *f(x)*, но по известному *f(x)* трудно найти подходящее значение *x*.

Реализация конкретного асимметричного алгоритма зависит от соответствующей односторонней функции.

До последнего времени специалистами предложено большое количество алгоритмов шифрования с открытым ключом, основанных на различных односторонних функциях, наиболее известные из этих алгоритмов будут рассмотрены ниже.

3.2.5 Примеры асимметричных шифров

Среди шифров, разработанных на принципах асимметричного шифрования, первым является *шифр RSA*. Свое название этот шифр получил по первым буквам фамилий его создателей: Рональда Райвеста, Ади Шамира и Леонарда Элдемана, которые основали компанию *RSA Data Secutity*. Фактически RSA это самый известный и самый популярный из асимметричных шифров.

Математически шифр RSA основывается на поиске делителей очень большого натурального числа, являющегося произведением двух простых. Такой поиск – крайне трудоемкая процедура, в том числе и для вычислительных систем. Поэтому по открытому ключу (числу, являющемуся сложной функцией делителя большого числа) очень сложно вычислить парный ему закрытый ключ (в свою очередь являющийся сложной функцией этого большого числа).

Считается, что шифр RSA является криптостойким при достаточной длине ключей, например, приемлемый вариант 1024 битов. Однако, со временем этого может оказаться не достаточно в условиях роста мощности процессоров и атак полного перебора. Но, в свою очередь, применение более длинных ключей повысит стойкость шифра.

Другим примером асимметричного шифра является *шифр DSS* (Digital Signature Standard). Для этого шифра длина используемого ключа варьируется в пределах от 512 до 1024 битов.

Основное назначение шифра DSS – создание цифровой подписи. Поэтому для закрытия информации данный шифр не используется. В настоящее время в алгоритме DSS найдены некоторые слабые места защиты, поэтому он распространен менее широко.

Среди наиболее популярных шифров программная реализация асимметричного алгоритма RSA гораздо сложнее, чем симметричного алгоритма DES. Поэтому асимметричные шифры, как правило, используются при передаче небольшого объема сообщений.

В целом следует отметить, что появление асимметричной схемы шифрования явилось технологической революцией, предоставившей стойкую криптографию широкому кругу пользователей.

3.2.6 Контрольные вопросы

1) Основные причины создания асимметричной криптографии, ее первые разработчики.

2) Сущность асимметричных методов шифрования.

3) Сферы практического применения асимметричных алгоритмов.

4) Реализация протокола обмена конфиденциальными электронными сообщениями по асимметричной схеме.

5) Использование асимметричных алгоритмов для ключевого управления в симметричных криптосистемах.

6) Требования к алгоритмам шифрования с открытым ключом.

7) Примеры асимметричных шифров и их краткая характеристика.

4 ЭЛЕКТРОННАЯ ЦИФРОВАЯ ПОДПИСЬ НА ОСНОВЕ АЛГОРИТМОВ С ОТКРЫТЫМ КЛЮЧОМ

4.1 Функции и свойства электронной цифровой подписи

В современных условиях во всех сферах человеческой деятельности, наряду с традиционными «бумажными» документами и обычным документооборотом, все большее значение приобретают электронные документы и электронный документооборот. В этих условиях необходим механизм, обеспечивающий подтверждение авторства и целостность электронных документов. В традиционном документообороте такой механизм есть – это собственноручная подпись и печать. Следовательно, и электронные документы должны иметь свой аналог обычной физической подписи.

Внедрение системы *электронной цифровой подписи* становится особенно актуальной в связи с развитием электронной коммерции и возможностью оплаты услуг через Интернет. Поэтому во многих системах связи и, прежде всего, в телекоммуникационных сетях участники коммерческих сделок должны иметь возможность удостовериться в подлинности полученного электронного документа и при необходимости доказать свое авторство получателю или третьей стороне (например, суду).

Любая подпись, будь то обычная или цифровая, всегда выполняет, по крайней мере, следующие функции:

– удостоверение того, что подписавшийся является тем, за которого мы его принимаем;

– подтверждение того, что подписавшийся не может отказаться от документа, который он подписал;

– удостоверение того, что отправитель подписал именно тот документ, который отправил, а не какой-либо иной. Другими словами ему нельзя навязать другой или похожий документ, поскольку у него есть подписанная копия оригинала.

Заметим, что две первые функции обеспечивают защиту интересов лица, для которого предназначен документ (получателя), третья функция защищает интересы подписывающего (отправителя).

Во всех этих случаях «работает» свойство подписи, называемое *аутентичность*, то есть подлинность. Это свойство переносится на документ, под которым стоит подпись.

Подпись должна обладать следующими свойствами:

1) подпись должна создаваться только одним лицом, а подлинность ее может быть удостоверена многими;

2) подпись неразрывно связывается с конкретным документом и не может быть перенесена на другой документ;

3) после того, как документ подписан, его невозможно изменить;

4) от поставленной подписи невозможно отказаться, то есть лицо, подписавшее документ, не сможет потом утверждать, что не ставило подпись.

На основании рассмотренных функций и свойств можно сформулировать следующие требования к ЭЦП:

1) подпись должна быть битовым (двоичным) образцом, который зависит от подписываемого электронного сообщения (тоже битового образца);

2) подпись должна использовать некоторую уникальную информацию отправителя для предотвращения подделки или отказа (его закрытого ключа);

3) создавать цифровую подпись должно быть относительно легко, как и обычный автограф;

4) вычислительно невозможно подделать цифровую подпись как созданием нового сообщения для существующей цифровой подписи, так и созданием ложной цифровой подписи для некоторого сообщения;

5) цифровая подпись должна быть достаточно компактной и не занимать много памяти.

Таким образом, ЭЦП в электронном документе должна служить той же цели, что и собственноручный автограф или печать на бумажном листе. ЭЦП должна не только подтверждать авторство (то есть являться средством авторизации), но также должна обеспечить определение того было ли содержание подписанного электронного документа изменено в процессе передачи (то есть обеспечить контроль его целостности).

Более того, вследствие своей цифровой природы ЭЦП, безусловно, должна превосходить ручную подпись и печать.

Для формирования электронной цифровой подписи – уникального числового дополнения к передаваемой информации, позволяющего проверить ее авторство – могут использоваться алгоритмы шифрования с открытым ключом. При этом ЭЦП представляет собой последовательность бит фиксированной длины, которая вычисляется определенным образом с помощью содержимого подписываемой информации и закрытого ключа.

При формировании ЭЦП специальным образом шифруется или все электронное сообщение целиком, или результат вычисления хеш-функции от сообщения. Последний способ на практике оказывается предпочтительнее, так как подписываемое сообщение может иметь разный размер, чаще всего довольно большой, а хеш-значение (хеш-код, дайджест, сигнатура, свертка, контрольная сумма, код аутентичности) всегда имеет постоянную обычно короткую длину.

Рассмотрим подробнее оба варианта формирования цифровой подписи.

4.2 Технология формирования и сверки электронной цифровой подписи

Первый, наиболее простой способ формирования и сверки ЭЦП основывается, так же как и при асимметричном шифровании, на использовании пары связанных между собой ключей – открытого и закрытого. Но, в отличие от шифрования, роли ключей меняются – закрытый ключ становится ключом подписывания, а открытый – ключом сверки. Так как по открытому ключу

нельзя практически найти закрытый ключ, то в качестве подписи может выступать само сообщение, зашифрованное закрытым ключом.

Таким образом подписать электронный документ может только владелец закрытого ключа, но каждый, кто имеет его открытый ключ, может сверить подпись.

Протокол формирования и сверки ЭЦП, когда Отправитель хочет передать Получателю подписанное сообщение, состоит из следующих шагов:

1) На предварительном этапе Отправитель посылает Получателю свой открытый ключ $\mathcal{P}_{\text{ОК-О}}$ по любому каналу связи, в том числе открытому, например, по электронной почте.

2) Отправитель зашифровывает сообщение *M* своим закрытым ключом $\mathcal{P}_{\text{ЗК-О}}$, получая при этом криптограмму *C*, являющееся в данном случае цифровой подписью.

3) Зашифрованное сообщение *C* пересылается Получателю.

4) Получатель расшифровывает полученное сообщение *C*, используя открытый ключ Отправителя $\mathcal{P}_{\text{ОК-О}}$. Если сообщение расшифровалось, значит, оно действительно подписано Отправителем.

Схема реализации этого протокола приведена на рисунке 4.1.

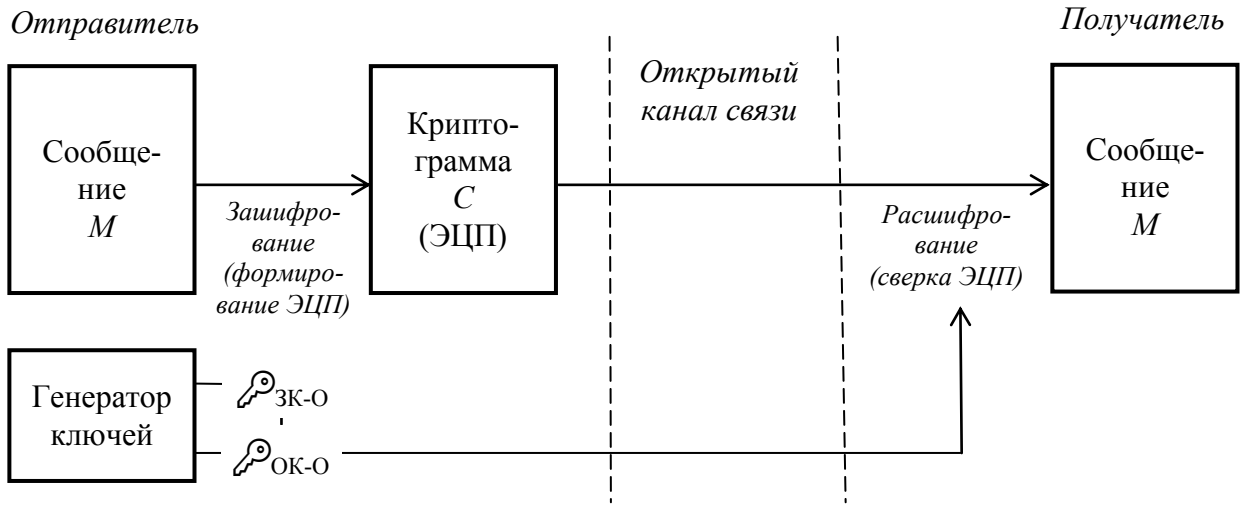


Рисунок 4.1 – Схема протокола простого способа формирования и сверки ЭЦП

Если Отправитель надежно хранит свой закрытый ключ, его подписи будут достоверны. Третьей стороне не будет возможности изменить сообщение (функция обеспечения целостности), не имея доступа к закрытому ключу Отправителя. Кроме того, в ключе обычно содержится дополнительная информация, например, информация о владельце ключа, о сроке действия ключа и др., поэтому при сверке подписи сразу же становится ясно, кто подписал то или иное сообщение (функция установления авторства). Тем самым обеспечиваются все перечисленные функции и свойства ЭЦП.

Представленная на рисунке 4.1 схема обеспечивает формирование так называемой *электронной цифровой подписи с восстановлением документа*. Такие подписи как бы содержат в себе подписываемый электронный доку-

мент – в процессе сверки ЭЦП автоматически вычисляется и тело самого документа. Если при расшифровывании сообщение воспроизведено правильно, то это значит, что подпись была верной.

Однако, в настоящее время на практике так обычно не делается. Это обусловлено следующими причинами:

во-первых, алгоритмы асимметричной криптографии достаточно медленные и для формирования ЭЦП путем зашифровывания всего (обычно большого) документа требуется слишком много времени;

во-вторых, для подтверждения целостности сообщения требуется много памяти, так как вычисленная таким образом ЭЦП будет неоправданно большой (значительно больше самого документа).

Поэтому для экономии времени и памяти ЭВМ, а также для удобства работы при формировании и сверке ЭЦП асимметричный алгоритм обычно используется вместе с какой-нибудь *однонаправленной хеш-функцией*. Одностороняя стойкая хеш-функция – математический алгоритм, преобразовывающий сообщение произвольной длины в хеш-значение (всегда в строку фиксированной длины).

Такая хеш-функция обладает применительно к целям преобразования двумя свойствами:

– из хеш-значения такой функции невозможно вычислить исходное сообщение или любую его часть;

– вероятность повторения хеш-значения для двух произвольных сообщений ничтожно мала.

Другими словами, не существует никакого искусственно созданного поддельного сообщения, которое после преобразования повторило бы хеш-значение исходного сообщения.

В случае использования однонаправленной хеш-функции на первом этапе с ее помощью из сообщения произвольной длины вычисляется хеш-значение нужного размера, а на втором этапе для вычисления ЭЦП производится зашифровывание с помощью закрытого ключа полученного на предыдущем этапе хеш-значения от сообщения.

Обычно ЭЦП, вычисленные по хеш-значению электронного документа, называют *присоединяемыми цифровыми подписями*. Такие цифровые подписи представляют собой некоторый уникальный числовой код, который необходимо присоединять (внедрять) к подписываемому сообщению. Тело документа при этом не зашифровывается и передается в открытом виде вместе с электронной цифровой подписью отправителя.

Протокол формирования и сверки ЭЦП, когда Отправитель хочет передать Получателю сообщение, дополненное присоединенной цифровой подписью, состоит из следующих шагов:

1) На предварительном этапе Отправитель посылает Получателю свой открытый ключ $\mathcal{P}_{\text{ОК-О}}$ по любому каналу связи, в том числе открытому, например, по электронной почте.

2) Отправитель сообщения M с помощью некоторой надежной хеш-функции H вычисляет хеш-значение своего сообщения, причем $h = H(M)$.

3) Затем Отправитель зашифровывает хеш-значение *h* сообщения *M* своим закрытым ключом $\mathcal{P}_{\text{ЗК-О}}$, получая при этом цифровую подпись *C*.

4) Исходное сообщение *M* вместе с цифровой подписью *C* пересылаются Получателю.

5) Получатель двумя разными способами вычисляет хеш-значение:

– непосредственно хешируя полученное сообщение *M*, то есть *h = H(M)*;

– расшифровывая цифровую подпись *C*, используя открытый ключ Отправителя $\mathcal{P}_{\text{ОК-О}}$.

6) Получатель сравнивает полученные разными способами хеш-значения и, если они совпадают принимает решение, что сверка прошла. В противном случае принимается решение: либо подпись *C* не верна, либо сообщение *M* фальсифицировано.

Схема реализации этого протокола приведена на рисунке 4.2.

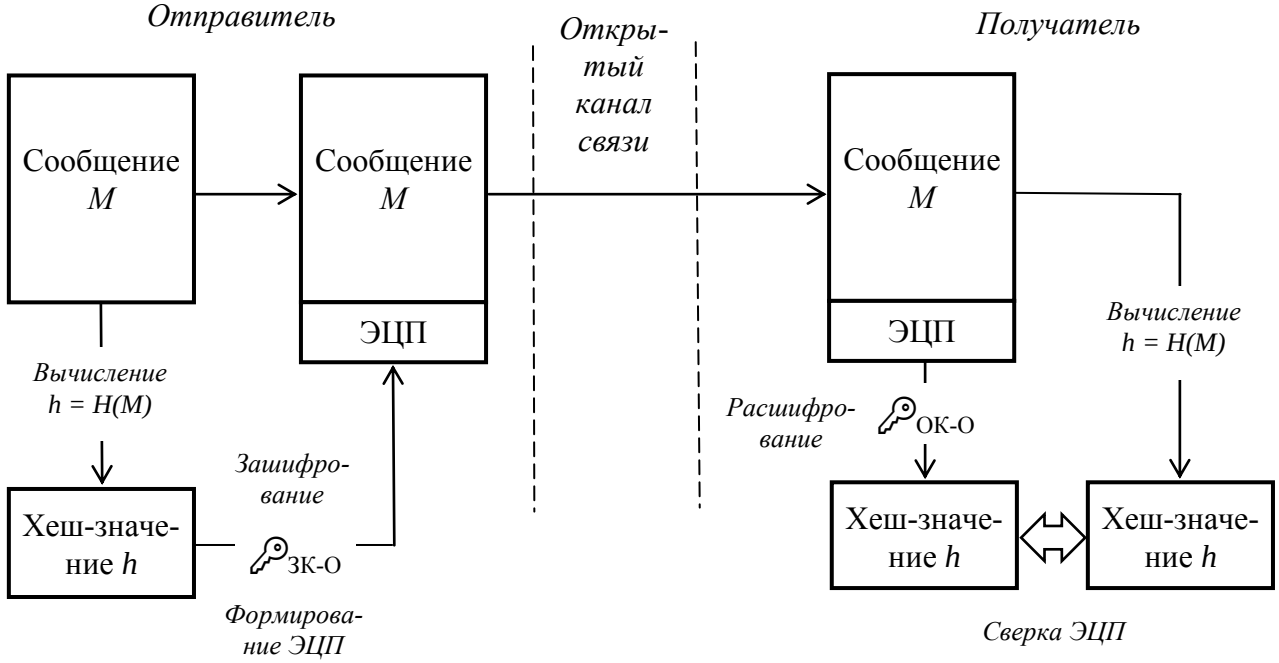


Рисунок 4.2 – Схема протокола способа формирования и сверки присоединенной к сообщению ЭЦП

В схеме рисунка 4.2 хеш-функция не является частью алгоритма ЭЦП, поэтому может быть использована любая надежная хеш-функция.

Следует отметить, что в случае присоединенной к сообщению ЭЦП не обеспечивается конфиденциальность. То есть сообщение, хотя и невозможно изменить, но можно прочитать. Даже если не использовать хеш-функцию, а применять электронную цифровую подпись с восстановлением документа, конфиденциальность не обеспечивается, так как любой может расшифровать сообщение, используя открытый ключ отправителя.

На практике, во многих ситуациях приведенной на рисунке 4.2 схемы создания и использования цифровой подписи оказывается вполне достаточно. Однако бывают случаи, когда Получатель может сжульничать. Особенно,

если пересылаемым документом является чек за оказанные услуги от Отправителя. Так Получатель может удостовериться, что ЭЦП на чеке верная и, после получения денег, снять несколько копий с подписанного документа. После чего периодически предъявлять их в банк для получения денег.

Для недопущения такого мошенничества в ЭЦП часто включают *метки времени*. Дата и время подписания документа добавляются к его телу и подписываются вместе со всем документом. Метка времени может быть зафиксирована банком при оплате чека и занесена в базу данных. Попытка повторного предъявления чека в этом случае будет замечена банком и мошенничество станет невозможным.

На практике используют и более сложные разновидности ЭЦП, например, так называемую, *неотрицаемую цифровую подпись*. В отличие от обычной ЭЦП неотрицаемая подпись не может быть проверена без разрешения подписавшего. Таким образом, получатель документа не сможет сверить подпись без согласия лица, подписавшего сообщение.

Особенностью всех разновидностей ЭЦП является то, что любая ЭЦП может быть сгенерирована тем, кто обладает копией нужного закрытого ключа.

Таким образом, цифровая подпись свидетельствует не о том, что электронный документ заверило конкретное физическое лицо, а о том, что этот документ заверен конкретным криптографическим ключом. По этой причине на практике криптографический ключ «связывается» с физическим лицом введением дополнительных механизмов защиты ключа, основанных на личностных данных его владельца. Одним из таких механизмов является *сертификация ключей* специально созданными федеральными, региональными и муниципальными органами – удостоверяющими (сертифицирующими, регистрирующими и т. п.) центрами. Организационные и правовые нормы деятельности удостоверяющих центров рассмотрены в первом разделе учебного пособия.

4.3 Контрольные вопросы

1) Определение, функции и свойства электронной цифровой подписи.

2) Технологические процедуры применения электронной цифровой подписи: простой способ формирования и сверки ЭЦП с восстановлением документа.

3) Назначение однонаправленной хеш-функции и требования к ней.

4) Технологические процедуры применения электронной цифровой подписи: способ формирования и сверки присоединенной к сообщению ЭЦП.

5) Способы недопущения мошенничества с ЭЦП.

5 АЛГОРИТМЫ ХЕШИРОВАНИЯ

5.1 Однонаправленные хеш-функции

5.1.1 Основы построения хеш-функций

Хеш-функцией (*hash function* от англ. hash – мелко измельчать и перемешивать) называется математическая или иная функция, которая для строки произвольной длины вычисляет некоторое целое значение или некоторую другую строку фиксированной длины.

Математически это можно записать так:

$$h = H(M),$$

где M – исходное сообщение, называемое иногда *прообразом*;

h – результат, называемый значением хеш-функции, а также *хеш-кодом* или *дайджестом сообщения* (от англ. *message digest*).

Основной смысл хеш-функции состоит в определении характерного признака прообраза – значения хеш-функции. Это значение обычно имеет определенный фиксированный размер, например, 64 или 128 битов. Хеш-код может быть в дальнейшем проанализирован для решения какой-либо задачи.

Например, хеширование может применяться для сравнения данных:

– если у двух массивов данных хеш-коды разные, массивы гарантированно различаются;

– если одинаковые – массивы, скорее всего, одинаковы.

Очевидно, что однозначного соответствия между исходными данными и хеш-кодом нет из-за того, что количество значений хеш-функций всегда меньше, чем вариантов входных данных. Следовательно, существует множество входных сообщений, дающих одинаковые хеш-коды (такие ситуации называются *коллизиями*). Вероятность возникновения коллизий играет немаловажную роль в оценке качества хеш-функций.

В современной криптографии хеш-функции нашли широкое применение. Например, отправитель строит хеш-сообщение, зашифровывает его и отправляет вместе с сообщением (также зашифрованным). Получатель расшифровывает сообщение и хеш, строит новый хеш для расшифрованного сообщения и сравнивает его с полученным. Если они одинаковы, то вероятность того, что сообщение пришло без изменений весьма высока. Обычно хеш-функции являются однонаправленными. Слово «однонаправленная» обозначает, что практически невозможно воспроизвести исходную текстовую строку на основании ее хеш-значения. В свою очередь слова «практически невозможно» обозначают, что для нахождения такой строки потребуются вычислительные ресурсы недостижимой на данный момент мощности из-за ограниченности современных технологий, а также то, что время нахождения такой строки с использованием современных вычислительных средств настолько велико (может измеряться годами), что даже если такая строка будет найдена, то ее информационный смысл уже утратит свою значимость.

Простейшая хеш-функция может быть составлена с использованием операции «сумма по модулю 2» следующим образом:

– сначала получают входную строку и складывают все байты по модулю 2;

– затем байт-результат возвращают в качестве значения хеш-функции.

В этом случае длина значения хеш-функции составит 8 бит независимо от размера входного сообщения. Например, пусть исходное сообщение, переведенное в цифровой вид и представленное в шестнадцатеричном формате, имеет вид

3F AB 2C 45 D1

Для перевода сообщения в двоичный вид, запишем байты друг под другом и сложим биты в каждом столбике по модулю 2:

$$\begin{array}{r} 0011\ 1111 \\ 1010\ 1011 \\ 0010\ 1100 \\ 0100\ 0101 \\ 1101\ 0001 \\ \hline 0010\ 1100 \end{array}$$

Результат $0010\ 1100_2$ или $2C_{16}$ и будет значением хеш-функции.

Рассмотренная простая хеш-функция не годится для криптографических применений, так как достаточно легко изменить содержание подписанного сообщения, не меняя значения контрольной суммы (это легко сделать самостоятельно). Поэтому в криптографии хеш-функция будет считаться хорошей, если трудно создать два прообраза с одинаковым значением хеш-функции, а также, если у выхода функции нет явной зависимости от входа.

Можно сформулировать основные требования, предъявляемые к криптографическим хеш-функциям:

– хеш-функция должна быть применима к сообщению любого размера;

– значение хеш-функции должно вычисляться достаточно быстро;

– при известном значении хеш-функции должно быть трудно (практически невозможно) найти подходящий прообраз *M*;

– при известном сообщении *M* должно быть трудно найти другое сообщение *M'* с таким же значением хеш-функции, как у исходного сообщения;

– должно быть трудно найти какую-либо пару случайных различных сообщений с одинаковым значением хеш-функции.

Создание хеш-функции, которая удовлетворяет всем перечисленным требованиям, непростая задача. Следует также учитывать, что на вход хеш-функции могут поступать данные произвольного размера, а длина хеш-результата должна получаться всегда одинакового размера.

В современной криптографии в качестве хеш-функций обычно применяются функции, обрабатывающие входное сообщение блок за блоком и вычисляющие хеш-значение h_i для каждого блока M_i входного сообщения по зависимостям вида

$$h_i = H(M_i, h_{i-1}),$$

где h_{i-1} – результат, полученный при вычислении хеш-функции для предыдущего блока входных данных.

В результате выход хеш-функции h_n является функцией от всех n блоков входного сообщения.

5.1.2 Хеш-функции на основе симметричных блочных алгоритмов

В качестве хеш-функции можно использовать блочный алгоритм симметричного шифрования. Действительно, если используемый блочный алгоритм является криптографически стойким, то и хеш-функция на его основе будет надежной.

Простейшим способом использования блочного алгоритма для получения хеш-кода является шифрование сообщения в режиме CBC (этот режим упоминался в разделе 3 учебного пособия при рассмотрении симметричного шифра DES). Режим CBC (Cipher Block Chaining) отличается от обычного (режима ECB) тем, что перед шифрованием очередного блока к нему применяется операция «исключающее ИЛИ» с предыдущим блоком.

В режиме CBC сообщение представляется в виде последовательности блоков, длина которых равна длине блока алгоритма шифрования. При необходимости последний блок дополняется справа нулями, чтобы получился блок нужной длины. Хеш-значением будет последний зашифрованный блок текста. При условии использования надежного блочного алгоритма шифрования полученное хеш-значение будет обладать следующими свойствами:

– практически невозможно без знания секретного ключа шифрования вычислить хеш-значение для заданного открытого массива информации;

– практически невозможен без знания секретного ключа шифрования подбор открытых данных под заданное значение хеш-функции.

Сформированное таким образом хеш-значение обычно называют *имитовставкой* или *аутентификатором* и используется для проверки целостности сообщения. Таким образом, имитовставка – это контрольная комбинация, зависящая от открытых данных и секретной ключевой информации. Целью использования имитовставки является обнаружение всех случайных или преднамеренных изменений в массиве информации.

Значение, полученное хеш-функцией при обработке входного сообщения, присоединяется к сообщению в тот момент, когда стало известно, что сообщение корректно. Получатель проверяет целостность сообщения путем вычисления имитовставки полученного сообщения и сравнения его с полученным хеш-кодом, который должен быть передан безопасным способом.

Одним из таких безопасных способов может быть шифрование имитовставки закрытым ключом отправителя, то есть создание цифровой подписи. Возможно также шифрование полученного хеш-кода алгоритмом симметричного шифрования, если отправитель и получатель имеют общий ключ симметричного шифрования.

Указанный процесс получения и использования имитовставки описан в отечественном стандарте ГОСТ 28147-89 (http://protect.gost.ru). Стандарт предлагает использовать младшие 32 бита блока, полученного на выходе операции зашифрования всего сообщения в режиме сцепления блоков шифра для контроля целостности передаваемого сообщения. Таким же образом для формирования имитовставки можно использовать любой блочный алгоритм симметричного шифрования.

Еще одним возможным способом применения блочного шифра для выработки хеш-кода является следующий способ. Исходное сообщение обрабатывается последовательно блоками. Последний блок при необходимости дополняется нулями, иногда в последний блок приписывают длину сообщения в виде двоичного числа. На каждом этапе зашифровывается хеш-значение, полученное на предыдущем этапе, с помощью текущего блока сообщения взятого в качестве ключа. Последнее полученное зашифрованное значение будет окончательным хеш-результатом.

Таким образом, если стандартную схему шифрования сообщения M с помощью блочного шифра f на ключе K обычно записывают как

$$E = f(M, K),$$

то схему получения хеш-кода h по описанному выше алгоритму можно представить как

$$h_i = f\left(h_{i-1}, M\right).$$

В качестве начального хеш-кода h_0 обычно берут некоторую константу. Зашифрование производится в режиме простой замены. При использовании указанного способа размер блока совпадает с длиной ключа и размером хеш-значения будет длина блока.

Возможен также другой способ использования блочного шифра в режиме простой замены – элементы сообщения зашифровываются хеш-значениями, полученными на предыдущем этапе, то есть

$$h_i = f\left(M, h_{i-1}\right).$$

Возможны еще несколько схем использования блочного шифра для формирования хеш-функции. Например, пусть M_i – блок исходного сообщения, h_i – значение хеш-функции на i-ом этапе преобразования, f – блочный алгоритм шифрования, используемый в режиме простой замены, $\oplus$ – операция сложения по модулю 2. Тогда в числе других возможны следующие схемы формирования хеш-функции:

$$h_i = f\left(M_i, h_{i-1}\right) \oplus M_i;$$
$$h_i = f\left(M_i, h_{i-1}\right) \oplus h_{i-1} \oplus M_i;$$
$$h_i = f\left(h_{i-1}, M_i\right) \oplus h_i;$$
$$h_i = f\left(h_{i-1} \oplus M_i, M_i\right) \oplus h_i.$$

Во всех этих схемах длина формируемого хеш-значения равна длине блока при шифровании. Кроме перечисленных, также некоторые другие схемы использования блочного алгоритма шифрования для вычисления хеш-значений могут применяться на практике.

Основным недостатком хеш-функций, спроектированных на основе блочных алгоритмов, является относительно низкая скорость работы. Необходимую криптостойкость можно обеспечить и за меньшее количество операций над входными данными. Существуют более быстрые алгоритмы хеширования, спроектированные исходя из требований криптостойкости. Наиболее распространенные из них – MD, SHA и ГОСТ Р34.11-94, будут рассмотрены ниже.

5.2 Примеры алгоритмов хеширования

Рассмотрим алгоритмы криптостойкого хеширования, получившие наибольшее распространение в настоящее время:

– алгоритмы MD2, MD4, MD5;

– алгоритм безопасного хеширования SHA;

– отечественный стандарт хеш-функции.

5.2.1 Алгоритмы MD2, MD4, MD5

Структура алгоритмов MD соответствует схеме Меркла – Дамгаарда. Данная схема была предложена в 1988-1989 годах независимо двумя известными криптологами: Ральфом Мерклом (Ralph Merkle) и Айвеном Дамгаардом (Ivan Damgard).

Сущность схемы состоит в следующем:

– хешируемое сообщение *M* разбивается на блоки определенной длины $m_0 \cdots m_i \cdots m_N$. Выполняется (по-разному в различных алгоритмах) дополнение сообщения *M* до размера, кратного данной длине блока (при этом количество блоков может увеличиться);

– каждый *i*-й блок сообщения обрабатывается функцией $H(\cdot)$, являющейся криптографическим ядром алгоритма хеширования, причем данная функция накладывает результат этой обработки на текущее хеш-значение h_{i-1} (то есть на результат обработки предыдущих блоков сообщения)

$$h_i = H(h_{i-1}, m_i);$$

– результатом работы алгоритма хеширования *hash*(), то есть хеш-значением сообщения *M*, является значение h_N

$$hash(M) = h_N;$$

– начальное значение h_{-1} обычно является константным и различным в различных алгоритмах хеширования. Оно часто обозначается как IV (от *Initialization Vector* – вектор инициализации).

Рассмотренная схема хеширования Меркла – Дамгаарда представлена на рисунке 5.1.

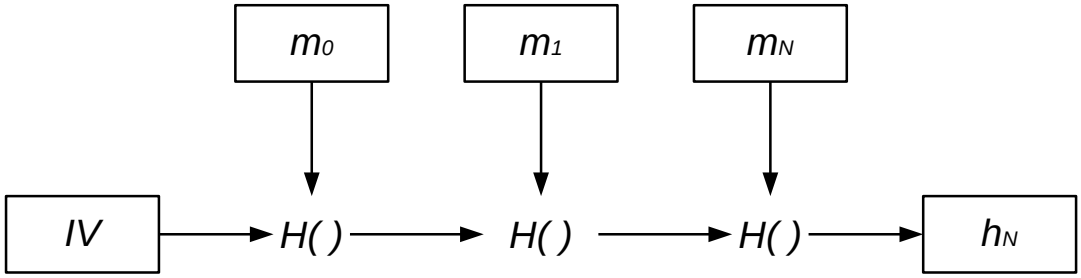


Рисунок 5.1 – Схема Меркла – Дамгаарда

Алгоритмы *MD2, MD4, MD5* разработаны профессором Массачусетского университета Рональдом Райвестом. Символы в названии «*MD*» означают *Message Digest* – краткое изложение сообщения. Существует и алгоритм *MD1*, который, однако, является закрытым и его спецификация не опубликована.

Алгоритм MD2 был разработан в 1988 г. для использования в качестве одного из криптографических алгоритмов, входящих в стандарт защищенной электронной почты PEM (*Privacy-Enhanced Mail*) и из рассматриваемых считается наиболее простым.

Процесс хеширования по алгоритму MD2 осуществляется по следующему упрощенному протоколу:

1) Выполняется дополнение сообщения *M* до размера, кратного 16 байтам (дополнение выполняется и в том случае, когда исходный размер входных данных кратен 16 байтам), следующим образом:

– в конец сообщения *M* добавляется определенное количество *n* байтов (от 1 до 16), чтобы результат составлял 32 байта. Например, дополнение 27-байтного сообщения состоит из 5 байтов, каждый из которых содержит значение 5;

– вычисляется 16-байтная контрольная сумма сообщения, которая также дополняет сообщение *M* перед его дальнейшей обработкой. При вычислении контрольной суммы используются операции табличной замены, данные исходного сообщения и операция сложения по модулю 2. Таким образом, формируется 48 байтный вектор инициализации, называемый внутренним состоянием, начальное значение которого равно 0.

2) Каждый 16-байтный m_i-й блок дополненного сообщения, включая блок контрольной суммы, последовательно накладывается на вектор состояния так, что после копирования блока данных второй из фрагментов состояния *X(16-31 бит)* содержит обрабатываемый блок данных, а третий *X(32-47 бит)* – результат применения побитовой операции «исключающее или» (XOR) к текущему содержимому первого фрагмента *X(0-15 бит)* и блока данных. Графически это поясняется на рисунке 5.2.

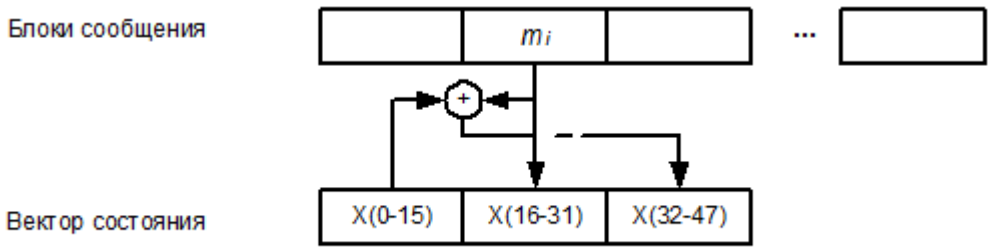


Рисунок 5.2 – Копирование блока данных в состояние

3) Выполняется обработка вектора состояния, которая состоит из 18 раундов преобразований, в рамках каждого из которых выполняется обновление каждого байта состояния путем его сложения по модулю 2 с содержимым специальной таблицы, заполненной псевдослучайными значениями дробной части числа π.

4) После обработки всех блоков сообщения в первом 16-байтном фрагменте состояния *X(0-15)* находится результирующее хеш-значение алгоритма.

Алгоритм MD4 вычисляет 128-битовое хеш-значение от входного сообщения *M* переменного и неограниченного размера.

Процесс хеширования по алгоритму MD4 осуществляется по следующему упрощенному протоколу:

1) Входное сообщение разбивается на блоки размером по 512 битов. Последний блок всегда дополняется до 512-битового размера следующим образом:

– сначала добавляется единичный бит;

– затем – нулевые биты до размера 448 битов (то есть 512 - 64); нулевые биты не добавляются, если после добавления единичного бита блок имеет 448-битовый размер;

– к последовательности добавляется 64-битовое значение *L*, равное размеру в битах исходной последовательности входных данных до дополнения. Если размер сообщения в битах *L* превышает $2^{64} - 1$ (что можно считать крайне маловероятным), то значение размера становится невозможно записать в 64 бита. Поэтому в качестве размера записывается не *L*, а значение $L \bmod 2^{64}$. Если изначально размер последнего блока равен 448 битам (или больше), дополнение выполняется все равно, при этом количество блоков увеличивается на один.

2) Для хранения промежуточных значений алгоритм использует 4 регистра по 32 бита, которые обозначаются как *A*, *B*, *C* и *D*. Они инициализируются перед началом вычислений согласно таблице 5.1, в которой указаны их шестнадцатеричные значения.

Таблица 5.1 – Исходные значения регистров

Регистр	Исходное значение
A	*01234567*
B	*89ABCDEF*
C	*FEDCBA98*
D	*76543210*

Согласно описанной выше схеме Меркла – Дамгаарда, совокупность регистров *A*…*D* содержит значение H_i после обработки *i*-го блока сообщения, а указанное в таблице начальное заполнение регистров является вектором инициализации алгоритма.

3) Каждый из 512-битовых блоков входных данных M_i поочередно обрабатывается следующим образом:

– блок входных данных представляется в виде 16 слов *M[0]…M[15]*, каждое из которых является 32-битовым;

– содержимое регистров *A*…*D* копируется во временные переменные *a*…*d* , как показано на рисунке 5.3;

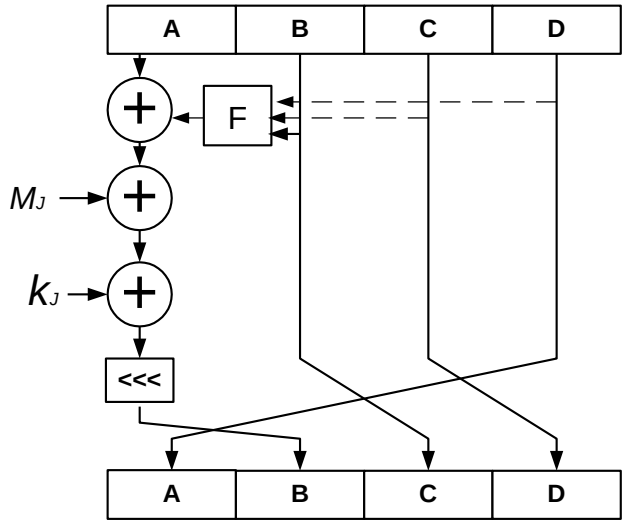


Рисунок 5.3 – Итерация алгоритма MD4

Выполняются 48 итераций таких преобразований, в каждой из которых модифицируются переменные *a*…*d* с участием одного из слов блока сообщения $M[0]\cdots M[15]$ в соответствии со следующим правилом:

$$t = \left(a + f_j(b,c,d) + M[x_j] + K_j \bmod 2^{32}\right) <<< s_j$$
$$a = d;\ d = c;\ b = t,$$

где *j* – номер итерации, *j = 0…47*;

t – временная 32-битовая переменная;

$f_j(\cdot)$ – одна из раундовых функций, определенных в таблице 5.2;

Таблица 5.2 – Раундовые функции

Номер итерации j	$f_j(\cdot)$
0…15	$f_j(x,y,z)=(x\wedge y)\vee(\overline{x}\wedge z)$
16…31	$f_j(x,y,z)=(x\wedge y)\vee(x\wedge z)\vee(y\wedge z)$
32…47	$f_j(x,y,z)=x\oplus y\oplus z$

– индекс используемого слова блока сообщения x_j определяется из таблицы 5.3;

Таблица 5.3 – Соответствия номера итерации и номера блока сообщения

Номер итерации j	x_j	Номер итерации j	x_j
0, 16, 32	*0*	*8, 18, 33*	*8*
1, 20, 40	*1*	*9, 22, 41*	*9*
2, 24, 36	*2*	*10, 26, 37*	*10*
3, 28, 44	*3*	*11, 30, 45*	*11*
4, 17, 34	*4*	*12, 19, 35*	*12*
5, 21, 42	*5*	*13, 23, 43*	*13*
6, 25, 38	*6*	*14, 27, 39*	*14*
7, 29, 46	*7*	*15, 31, 47*	*15*

– K_j – модифицирующие константы. В таблице 5.4 указаны их шестнадцатеричные значения;

Таблица 5.4 – Модифицирующие константы

Номер итерации j	K_j
0…15	*0*
16…31	*5A827999*
32…47	*6ED9EBA1*

– представленная на рисунке 5.3 операция <<< – это операция побитового циклического сдвига влево; количество битов сдвига s_j определяется в зависимости от номера итерации согласно таблице 5.5.

4) Значения регистров *A*…*D* складываются по модулю 2^{32} с полученными значениями переменных *a*…*d* соответственно.

5) Выходное значение алгоритма, то есть 128-битовое хеш-значение сообщения – результат конкатенации регистров *A*…*D* после обработки последнего блока расширенного сообщения.

Таблица 5.5 – Количество битов сдвига

Номер итерации j	s_j
0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44	*3*
17, 21, 25, 29	*5*
1, 5, 9, 13	*7*
18, 22, 26, 30, 33, 37, 41, 45	*9*
2, 6, 10, 14, 34, 38, 42, 46	*11*
19, 23, 27, 31	*13*
35, 39, 43, 47	*15*
3, 7, 11, 15	*19*

Алгоритм MD5 появился в начале 90-х годов XX века в результате усовершенствования алгоритма формирования хеш-функции MD4. В результате использования MD5 для произвольного сообщения формируется 128-битное хеш-значение. Входные данные обрабатываются блоками по 512 бит. В алгоритме используются элементарные логические операции (инверсия, конъюнкция, сложение по модулю 2, циклические сдвиги и др.), а также обыкновенное арифметическое сложение. Комплексное повторение этих элементарных функций алгоритма обеспечивает то, что результат после обработки хорошо перемешан. Поэтому маловероятно, чтобы два сообщения, выбранные случайно, имели одинаковый хеш-код.

Алгоритм MD5 имеет следующее свойство – каждый бит полученного хеш-значения является функцией от каждого бита входа. Считается, что MD5 является наиболее сильной хеш-функцией для 128-битного хеш-значения.

5.2.2. Алгоритм безопасного хеширования SHA

Алгоритм SHA (Secure Hash Algorithm – безопасный хеш-алгоритм) был разработан национальным институтом стандартов и технологии (NIST) США и опубликован в качестве американского федерального информационного стандарта в 1993 году.

Стандарт SHA-1, как и MD5, основан на алгоритме MD4. SHA-1 формирует 160-битное хеш-значение на основе обработки исходного сообщения блоками по 512 бит. В алгоритме SHA-1 также используются простые логические и арифметические операции.

Наиболее важным отличием SHA-1 от MD5 является то, что хеш-код SHA-1 на 32 бита длиннее, чем хеш-код MD5. Если предположить, что оба алгоритма одинаковы по сложности для криптоанализа, то SHA-1 является более стойким алгоритмом. Используя атаку методом грубой силы (лобовую атаку), труднее создать произвольное сообщение, имеющее данный хеш-код, а также труднее создать два сообщения, имеющие одинаковый хеш-код.

В 2001 году национальный институт стандартов и технологии США принял в качестве стандарта три хеш-функции с большей длиной хеш-кода,

чем у SHA-1. Часто эти хеш-функции называют SHA-2 или SHA-256, SHA-384 и SHA-512 (в названии указывается длина создаваемого алгоритмами хеш-кода).

Эти алгоритмы отличаются не только длиной создаваемого хеш-кода, но и используемыми внутренними функциями и длиной обрабатываемого блока:

– у SHA-256 длина блока – 512 битов;

– у SHA-384 и SHA-512 длина блока – 1024 бита.

Постепенные усовершенствования алгоритма SHA привели к увеличению его криптостойкости. Несмотря на отличия рассматриваемых алгоритмов друг от друга, все они являются дальнейшим развитием SHA-1 и MD4 и имеют похожую структуру.

5.2.3. Отечественный алгоритм хеширования ГОСТ Р34.11-94

В России принят ГОСТ Р34.11-94, который является отечественным стандартом для хеш-функций. Его структура довольно сильно отличается от структуры алгоритмов SHA-1,2 или MD5, в основе которых лежит алгоритм MD4. Длина хеш-кода, создаваемого алгоритмом ГОСТ Р 34.11-94, равна 256 битам.

Алгоритм последовательно обрабатывает исходное сообщение блоками по 256 бит справа налево. Параметром алгоритма является стартовый вектор хеширования – произвольное фиксированное значение длиной также 256 бит. В алгоритме ГОСТ Р34.11-94 используются операции перестановки, сдвига, арифметического сложения, сложения по модулю 2. В качестве вспомогательной функции в ГОСТ Р34.11-94 используется алгоритм по ГОСТ 28147-89 в режиме простой замены.

5.3 Контрольные вопросы

1) Что в криптографии называется хеш-функцией?

2) Для каких целей используются хеш-функции?

3) Перечислите основные требования, предъявляемые к хеш-функциям.

4) Назовите примеры криптографических хеш-функций.

5) Каков российский стандарт на алгоритм формирования криптографической хеш-функции?

6) Каким образом можно использовать блочный алгоритм шифрования для формирования хеш-функции?

5.4 Упражнения

Пусть хеш-функция $y = h(x_1\, x_2\, \ldots\, x_n)$ определяется как результат выполнения побитовой операции «сумма по модулю 2» для всех байтов сообщения, представленного в двоичном виде. Длина хеш-кода равна *8* битам. Для каждого из шести сообщений, записанных в левом столбце таблицы 5.6,

найдите соответствующий результат вычисления хеш-функции из правого столбца этой таблицы. Все сообщения и значения хеш-функции представлены в шестнадцатеричном формате.

Таблица 5.6 – Сообщения и их хеш-функции (вразброс)

Сообщения	Значения хеш-функции
• 0A3 69 2C • 82 0F B5 • 0DA 14 90 • 32 01 BF • 9E A6 23 • 10 BE 57	• 38 • 1B • 0F9 • 8C • 0E6 • 5E

6 АЛГОРИТМЫ ЭЛЕКТРОННОЙ ЦИФРОВОЙ ПОДПИСИ

6.1. Основные положения теории чисел

Каждое натуральное число, большее единицы, делится по крайней мере на два числа – на *1* и на само себя. Если число не имеет делителей, кроме самого себя и единицы, то оно называется *простым*, а если у числа есть еще делители, то *составным*. Единица же не считается ни простым числом, ни составным. Например, числа *7, 29* – простые; числа *9, 15* – составные (*9* делится на *3, 15* делится на *3* и на *5*).

Интересный факт – если два простых числа отличаются на *2*, то их называют *числами-«близнецами»*. Чисел-«близнецов» не очень много. Например, «близнецами» являются *5* и *7, 29* и *31, 149* и *151*, а также $242206083 \cdot 2^{38880} \pm 1$ (наибольшая найденная на момент написания учебного пособия пара чисел-«близнецов»).

Не о всяком числе можно сразу сказать, простое оно или составное. Если число меньше ста, то, скорее всего сразу можно ответить на этот вопрос. Однако с большими числами дело сложнее. Возьмем, например, число *2009*. Простое оно или составное? Попробуем найти возможные делители этого числа среди первых простых чисел. Число *2009* определенно не делится на *2* (так как оно нечетное), на *3* (так как сумма его цифр *2 + 9 = 11* не делится на *3*), на *5*. Попытка же разделить число *2009* на *7*, показывает, что получается целый результат – *287*. Таким образом, получен ответ: число *2009* – составное. В данном случае ответ получен достаточно быстро. Бывает, что проверка на простоту производиться гораздо дольше, а для работы с большими целыми числами требуются даже специальные компьютерные программы.

Поиск больших простых чисел имеет важное значение для математики и не только. Например, в криптографии большие простые числа используются в алгоритмах шифрования с открытым ключом. Для обеспечения надежности шифрования там используются простые числа длиной до *1024* бит.

Перемножить два числа сравнительно нетрудно, особенно если применить калькулятор, а числа не слишком велики. Существует и обратная задача – *задача факторизации* – нахождение двух или более чисел, дающих при перемножении заданное число. Эта задача гораздо труднее, чем перемножение чисел, и любому, кто пытался ее решить, об этом известно. Например, если требуется умножить *67* на *113*, то результат, *7571*, будет получен достаточно быстро. Если же требуется найти два числа, произведение которых равно *7571*, то, скорее всего, это займет гораздо больше времени.

Поиск сомножителей числа *n* может вестись, например, перебором всех простых чисел до $\sqrt{n}$ как в рассмотренном выше примере с числом *2009*. Однако, если множители – большие простые числа, то на их поиск уйдет достаточно много времени.

Таким образом, факторизация большого числа требует значительных затрат времени даже в том случае, когда известно, что оно является произведением двух больших простых чисел.

Сложность задачи факторизации используется в некоторых криптографических алгоритмах, например, в системе шифрования RSA.

6.1.1 Основная теорема арифметики

Любое составное число можно составить из некоторого количества простых чисел с помощью умножения. Например, составное число *2009* можно получить так

$$2009 = 7 \cdot 7 \cdot 41$$

В математике рассматривается так называемая *основная теорема арифметики*, которая утверждает, что любое натуральное число $n > 1$ либо само является простым, либо может быть разложено на произведение простых делителей, причем единственным способом (если не обращать внимания на порядок следования сомножителей).

Воспользовавшись обозначением степени, разложение числа *2009* на простые множители можно записать так

$$2009 = 7^2 \cdot 41$$

Разложение на множители называется *каноническим*, если все множители являются простыми и записаны в порядке возрастания.

Например, каноническое разложение числа *150* на множители запишется так

$$150 = 2 \cdot 3 \cdot 5^2$$

6.1.2 Взаимно простые числа и функция Эйлера

Два числа называются *взаимно простыми*, если они не имеют ни одного общего делителя кроме единицы.

Например, числа *11* и *12* взаимно просты (у них нет общих делителей кроме единицы), числа *30* и *35* – нет (у них есть общий делитель *5*).

Исследованием закономерностей, связанных с целыми числами, долго занимался швейцарский математик Леонард Эйлер (Leonard Euler). Одним из вопросов, которым он интересовался, был следующий: сколько существует натуральных чисел, не превосходящих n и взаимно простых с n? Ответ на этот вопрос был получен Эйлером в 1763 году и этот ответ связан с каноническим разложением числа n на простые множители.

Так, если

$$n = p_1^{a_1} \cdot p_2^{a_2} \cdot \cdots \cdot p_n^{a_n},$$

где $p_1, p_2, \cdots, p_n$ – разные простые множители, то число φ натуральных чисел, не превосходящих n и взаимно простых с n можно определить по формуле

$$\varphi(n) = n \times \left(1 - \frac{1}{p_1}\right) \cdot \left(1 - \frac{1}{p_2}\right) \cdot \ldots \cdot \left(1 - \frac{1}{p_n}\right)$$

Число натуральных чисел, не превосходящих n и взаимно простых с n, называется *функцией Эйлера* и обозначается $\varphi(n)$.

Например, найдем количество натуральных чисел, не превосходящих *12* и взаимно простых с *12*.

Из ряда натуральных чисел

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

взаимно простыми (не имеющими общих делителей) с *12* будут только числа *1, 5, 7, 11*. Их количество равно четырем.

Таким образом

$$\varphi(12) = 4.$$

Теперь можно подсчитать $\varphi(12)$ по формуле, предложенной Эйлером. Для этого вначале запишем каноническое разложение числа *12*

$$12 = 2^2 \cdot 3.$$

Теперь подсчитаем функцию Эйлера $\varphi(12)$

$$\varphi(12) = 12 \cdot \left(1 - \frac{1}{2}\right) \cdot \left(1 - \frac{1}{3}\right) = 12 \cdot \frac{1}{2}\frac{2}{3} = 4$$

Значения, вычисленные путем простого перебора взаимно простых чисел и по формуле Эйлера, совпали. Это неудивительно, так как формула для вычисления функции Эйлера может быть доказана строго математически.

Формулу Эйлера удобно использовать для больших n, если известно разложение числа n на простые множители. Для криптографии формула Эйлера важна тем, что она позволяет легко получить число $\varphi(n)$ для простых и некоторых других чисел.

В криптографии используются два следующих следствия формулы Эйлера.

Следствие 1. Если p – простое число, то

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

Действительно, если p – простое число, то его каноническое разложение состоит только из него самого.

Тогда

$$\varphi(p) = p \cdot \left(1 - \frac{1}{p}\right) = \frac{p(p-1)}{p} = p - 1$$

Следствие 2. Пусть p и q – два различных простых числа, причем $p \neq q$

Тогда

$$\varphi(p \cdot q) = (p-1)(q-1).$$

Эта формула объясняется следующим образом.

Пусть

$$p \cdot q = N,$$

где p и q – два различных простых числа $(p \neq q)$.

Тогда

$$\varphi(p \cdot q) = \varphi(N) = N \cdot \left(1 - \frac{1}{p}\right) \cdot \left(1 - \frac{1}{q}\right) = \frac{p \cdot q \cdot (p-1) \cdot (q-1)}{p \cdot q} = (p-1) \cdot (q-1)$$

Рассмотрим несколько примеров использования следствий формулы Эйлера.

Пример 1. Найдем $\varphi(13)$. Число *13* – простое, значит, используя первое следствие

$$\varphi(13) = 13 - 1 = 12,$$

можно выписать все числа, меньшие *13*, и подсчитать все взаимно простые с ним. Их действительно *12*.

Пример 2. Найдем $\varphi(35)$. Число *35* – составное число, значит, первое следствие не подходит. Однако *35* является произведением двух простых чисел *35 = 5 · 7*. Используя второе следствие, можно вычислить $\varphi(35)$

$$\varphi(35) = (5-1) \cdot (7-1) = 4 \cdot 6 = 24.$$

Проверяем, выписывая все числа, меньшие *35* и не имеющие с ним общих делителей:

1, 2, 3, 4, 6, 8, 9, 11, 12, 13, 16, 17, 18, 19, 22, 23, 24, 26, 27, 29, 31, 32, 33, 34.

Их действительно оказалось *24*.

На последнем примере видно, что использовать формулу Эйлера гораздо удобнее, чем рассматривать все числа из довольно большого диапазона и проверять на взаимную простоту.

6.1.3 Арифметика остатков и теория сравнений

Немецкий математик Карл Фридрих Гаусс предложил запись

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

для двух чисел a и b, если они имеют одинаковые остатки от деления на m (читается a сравнимо с b по модулю m).

Например,

$$1997 \equiv 1 \pmod{4},$$

$$7k + 1 \equiv 1 \pmod{7},$$

где k – любое целое число.

Сравнения обнаруживают полезные для математиков и криптографов свойства, во многом похожие на свойства равенств. Эти свойства позволяют

существенно упрощать арифметические вычисления, если важен только остаток от деления на некоторое число *m*.

Так, например, свойства сравнений полезны при расчетах в алгоритмах шифрования с открытым ключом.

Простейшими свойствами сравнений являются следующие свойства.

Свойство 1. Если *a* - *b* делится на *m*, то

$$a \equiv b(\mathrm{mod}\, m).$$

Например, $15 \equiv 1(\mathrm{mod}\, 7)$

так как *15 - 1 = 14*, а *14* кратно *7*.

Свойство 2. Если

$$a \equiv b(\mathrm{mod}\, m)$$

и

$$c \equiv d(\mathrm{mod}\, m),$$

то

$$a + c \equiv b + d(\mathrm{mod}\, m),$$

$$ac \equiv bd(\mathrm{mod}\, m).$$

Например, так как $13 \equiv 5(\mathrm{mod}\, 8)$ и $11 \equiv 3(\mathrm{mod}\, 8)$,

то

$$13 + 11 \equiv 5 + 3 \equiv 0(\mathrm{mod}\, m),$$

а также

$$13 \cdot 11 \equiv 5 \cdot 3 \equiv 7(\mathrm{mod}\, 8).$$

Свойство 3. Если

$$a \equiv b(\mathrm{mod}\, m),$$

то

$$a^k \equiv b^k(\mathrm{mod}\, m), k \subset N.$$

Например, так как

$$25 \equiv 4(\mathrm{mod}\, 7),$$

то

$$25^{40} \equiv 4^{40} \equiv \left(4^2\right)^{20} \equiv 2^{20} \equiv \left(2^4\right)^5 \equiv 4(\mathrm{mod}\, m).$$

Свойство 4. Если

$$ac \equiv bc(\mathrm{mod}\, m)$$

и *c* взаимно простое с *m*, то

$$a \equiv b(\mathrm{mod}\, m).$$

Например, известно, что

$$1200 \equiv 45(\mathrm{mod}\, 7),$$

а так как

$$1200 \equiv 15 \cdot 80$$

и

$$45 \equiv 15 \cdot 3,$$

то

$$80 \equiv 3 (\mathrm{mod}\, 7).$$

Свойство 5. Если

$$ac \equiv bc (\mathrm{mod}\, mc),$$

то

$$a \equiv b (\mathrm{mod}\, m).$$

Например, если

$$44 \equiv 4 (\mathrm{mod}\, 4 \cdot 10),$$

то

$$11 \equiv 1 (\mathrm{mod} 10).$$

6.1.4 Малая теорема Ферма и теорема Ферма-Эйлера

В основе алгоритма шифрования по системе RSA лежит теорема, сформулированная в начале семнадцатого столетия без доказательства французским математиком Пьером Ферма (Pierre Fermat). Ее часто называют «Малой теоремой Ферма», и ее не надо путать с известной «Великой теоремой Ферма» – ее он также сформулировал без доказательства, а доказана она была только в 1993-94 годах.

Леонард Эйлер в 1760 году опубликовал доказательство «Малой теоремы Ферма» и получил ее обобщение, известное под названием «Теоремы Ферма-Эйлера». Именно эта теорема используется в алгоритме зашифрования / расшифрования RSA.

Малая теорема Ферма формулируется следующим образом.

Если p – простое число, а m – любое число, которое не делится на p, то

$$m^{p-1} \equiv 1 (\mathrm{mod}\, p),$$

то есть число m^{p-1} при делении на p дает остаток 1.

Например, пусть $p = 11$, $m = 3$. Проверим, будет ли $3^{10} \bmod 11$ равно одному

$$3^{10} \bmod 11 = 144 \bmod 11 = 1.$$

Обобщение, сформулированное и доказанное Эйлером, справедливо для любого модуля, но в системе RSA используется частный случай, когда модуль является произведением только двух различных простых чисел. Поэтому рассмотрим формулировку теоремы для этого случая.

Теорема Ферма-Эйлера (для случая системы RSA).

Если p и q – два различных простых числа, а m – любое число, которое не делится на p и q, то

$$m^{(p-1)(q-1)} \equiv 1 (\mathrm{mod}\, pq).$$

Например, пусть $p = 11$, $q = 5$ (соответственно $pq = 55$), $m = 3$.

Проверим, будет ли

$$3^{40} \equiv 1 (\mathrm{mod}\, 55)$$

равно одному

$$3^{40} \bmod 55 = \left(3^5\right)^4 \bmod 55 = 23^4 \bmod 55 = 279841 \bmod 55 = 1.$$

6.1.5 Наибольший общий делитель

Пусть a и b – два целых положительных числа. Наибольший общий делитель (НОД) чисел a и b есть наибольшее число c, которое делит и a, и b:

$$c = \text{НОД}(a, b).$$

Например, $\text{НОД}(25, 35) = 5.$

Для нахождения наибольшего общего делителя можно использовать следующий алгоритм, известный как *алгоритм Евклида*.

Алгоритм NOD (целые a, b, c);
Начало
1) Пока $a \neq b$ выполнять:
   если $a>b$ то $a:=a-b$, иначе $b:= b-a$;
2) $c:=a$;

Конец.

После выполнения алгоритма результат будет содержаться в переменной c.

С помощью алгоритма Евклида НОД*(18, 9)* вычисляется следующим образом:

a:	18	9
b:	9	9
c:	9	9

Здесь каждый столбец представляет собой очередную итерацию алгоритма. Процесс продолжается до тех пор, пока b не станет равным a. Тогда в переменную c записывается ответ, в данном случае 9. Это и будет значение НОД*(18, 9)*.

6.1.6 Обобщенный алгоритм Евклида

Для многих криптографических систем, рассмотренных в данном учебном пособии, актуален так называемый *обобщенный алгоритм Евклида*, с которым связана следующая теорема.

Теорема. Пусть a и b – два целых положительных числа. Тогда существуют целые (не обязательно положительные) числа x и y, такие, что

$$ax + by = \text{НОД}(a, b).$$

Обобщенный алгоритм Евклида (ОАЕ) служит для отыскания НОД*(a,b)* и x, y, удовлетворяющих записанному выше уравнению.

Введем три строки:

$$U = (u_1, u_2, u_3),$$

$$V = (v_1, v_2, v_3),$$

$$T = (t_1, t_2, t_3).$$

Алгоритм записывается следующим образом (во входных параметрах должно соблюдаться условие $a \geq b$).

Алгоритм ОАЕ (целые a, b);
Начало
1) $U = (a,1,0)$, $V = (b,0,1)$.
2) Пока $v_1 \neq 0$ выполнять:
   $q = u_1 \text{ div } v_1$;
   $T = (u_1 \bmod v_1, u_2 - qv_2, u_3 - qv_3)$;
   $U = V$, $V = T$.
3) $U = (НОД(a, b), x, y))$.
Конец.

После окончания алгоритма результат будет содержаться в строке U. Операция *div* в алгоритме – это операция целочисленного деления.

Пример. Пусть $a = 18$, $b = 9$. Найдем числа x и y, удовлетворяющие уравнению

$$18x + 9y = \text{НОД}(18, 9).$$

Выполним алгоритм по шагам, как показано в таблице 6.1.

Таблица 6.1 – Алгоритм ОАЕ

q	U			V			T		
	u_1	u_2	u_3	v_1	v_2	v_3	t_1	t_2	t_3
$18 \text{ div } 9 = 2$	18	1	0	9	0	1	$18 \bmod 9 = 0$	$1 - 2 \cdot 0 = 1$	$0 - 2 \cdot 1 = -2$
	9	0	1	0	1	-2			

В результате получили

$$U = (\text{НОД}(a, b), x, y)) = (9, 0, 1).$$

Выполним проверку

$$18 \cdot 0 + 9 \cdot 1 = 9 = \text{НОД}(18, 9).$$

6.1.7 Инверсия по модулю m

Во многих задачах криптографии для заданных чисел c, m требуется найти такое число $d < m$, что

$$cd \bmod m = 1.$$

Такое d существует тогда и только тогда, когда числа c и m взаимно простые.

Число d, удовлетворяющее равенству $cd \bmod m = 1$, называется *инверсией c по модулю m* и часто обозначается

$$c^{-1} \bmod m.$$

Данное обозначение для инверсии связано с тем, что равенство

$$cd \bmod m = 1$$

можно переписать в виде

$$cc^{-1} \bmod m = 1.$$

Таким образом, умножение на c^{-1} соответствует делению на c при вычислениях по модулю m.

Инверсию по модулю m также можно вычислять с помощью обобщенного алгоритма Евклида.

Это делается следующим образом.

Равенство, приведенное ниже, означает, что для некоторого целого k имеет место равенство $cd - km = 1$. Учитывая, что c и d взаимно просты, это равенство можно преобразовать

$$m(-k) + cd = \text{НОД}(m, c).$$

Значит, можно вычислить $c^{-1} \bmod m$ (или найти число d) с помощью обобщенного алгоритма Евклида. При этом значение переменной k нас не интересует. Если число d получается отрицательным, то нужно прибавить к нему m, так как по определению число $a \bmod m$ берется из множества $\{0,1,..., m-1\}$.

Пример. Пусть $m = 9$, $c = 5$. Найдем $5^{-1} \bmod 9$. Будем выполнять вычисления по обобщенному алгоритму Евклида, записывая все вычисления по шагам, как показано в таблице 6.2.

Таблица 6.2 – Инверсия по модулю m

q	U			V			T		
	u_1	u_2	u_3	v_1	v_2	v_3	t_1	t_2	t_3
9 div 5 = 1	*9*	*1*	*0*	*5*	*0*	*1*	*9 mod 5 = 4*	*1 - 1·0 = 1*	*0 - 1·1 = -1*
5 div 4 = 1	*5*	*0*	*1*	*4*	*1*	*-1*	*5 mod 4=1*	*0 - 1·1 = -1*	*1 - 1·(-1)= 2*
4 div 1 = 4	*4*	*1*	*-1*	*1*	*-1*	*2*	*4 mod 1=0*	*1 - 4·(-1) =5*	*-1 -4·2 = -9*
	1	*-1*	*2*	*0*	*1*	*7*			

Таким образом, получили, что $5^{-1} \bmod 9 = 2$.

Проверим

$$5 \cdot 2 \bmod 9 = 10 \bmod 9 = 1.$$

6.2 Алгоритм цифровой подписи RSA

6.2.1 Общие сведения о системе RSA

Алгоритм шифрования с открытым ключом *RSA* был предложен одним из первых в конце 70-х годов XX века. Его название составлено из первых букв фамилий авторов – Р.Райвеста (R.Rivest), А.Шамира (A.Shamir) и

Л.Адлемана (L.Adleman). Алгоритм RSA является, наиболее популярным и широко применяемым асимметричным алгоритмом в криптографических системах.

Алгоритм основан на использовании того факта, что задача разложения большого числа на простые сомножители является очень трудоемкой.

Криптографическая система RSA базируется на следующих двух фактах из теории чисел:

1) задача проверки числа на простоту является сравнительно легкой;

2) задача разложения чисел вида $n = pq$, где p и q – простые числа, на множители является очень трудной, если известно только число n, а p и q – большие числа (это так называемая задача факторизации).

Алгоритм RSA представляет собой блочный алгоритм шифрования, где зашифрованные и незашифрованные данные должны быть представлены в виде целых чисел между 0 и n-1 для некоторого n.

6.2.2 Подготовка ключей в системе RSA

Первый этап – *подготовка ключей*. Для того, чтобы один абонент (абонент А) смог передать зашифрованное сообщение другому абоненту (абоненту Б), то абонент Б должен подготовить пару (открытый ключ и закрытый ключ) и отправить свой открытый ключ пользователю А.

Первым этапом является генерация открытого и закрытого ключей. Для этого вначале выбираются два больших простых числа P и Q. Затем вычисляется произведение N

$$N = PQ.$$

После этого определяется вспомогательное число f

$$f = (P - l)(Q - 1).$$

Затем случайным образом выбирается число $d < f$ и взаимно простое с числом f.

Далее необходимо найти число e, такое, что

$$ed \bmod f = 1.$$

Числа d и N будут открытым ключом пользователя, а значение e – закрытым ключом.

Таким образом, на этом этапе у пользователя должна быть информация, указанная в таблице 6.3.

Таблица 6.3 – Открытый и закрытый ключи

	Открытый ключ	Закрытый ключ
Пользователь системы	N, d	e

Так как пользователь Б хочет получить зашифрованное сообщение от пользователя А, значит пользователь Б должен отправить свой открытый

ключ (d, N) пользователю А. Числа P и Q больше не нужны, однако их нельзя никому сообщать.

На этом этап подготовки ключей закончен и можно использовать основной протокол RSA для шифрования данных.

6.2.3 Протокол RSA для шифрования данных

Второй этап – *шифрование данных*. Если абонент А хочет передать некоторые данные абоненту Б, он должен представить свое сообщение в цифровом виде и разбить его на блоки $M_1, M_2, M_3, \ldots$, где $M_i < N$. Зашифрованное сообщение будет состоять из блоков c_i.

Абонент А зашифровывает каждый блок своего сообщения по формуле

$$c_i = M_i^d \bmod N,$$

используя открытые параметры пользователя Б, и пересылает зашифрованное сообщение $C = (c_1, c_2, c_3, \ldots)$ по открытым каналам связи.

Абонент Б, получивший зашифрованное сообщение, расшифровывает все блоки полученного сообщения по формуле

$$M_i = c^e \bmod N.$$

Все расшифрованные блоки будут точно такими же, как и исходящие от пользователя А.

Злоумышленник, перехватывающий все сообщения и знающий всю открытую информацию, не сможет найти исходное сообщение при больших значениях P и Q.

Пример вычислений по алгоритму RSA

Пусть пользователь А хочет передать сообщение пользователю Б. В этом случае вначале пользователь Б должен подготовить открытый и закрытый ключи. Пусть им выбраны, например, следующие параметры:

$$P = 3,$$

$$Q = 11,$$

тогда

$$N = 3 \cdot 11 = 33.$$

Далее он вычисляет

$$f = (P - 1)(Q - 1) = (3-1)(11-1) = 20.$$

Затем пользователь Б выбирает любое число d, не имеющее общих делителей с f (это необходимо для того, чтобы зашифрованное сообщение можно было потом однозначно восстановить). Пусть $d = 13$. Это число будет одним из компонентов открытого ключа.

Далее необходимо найти число e, которое можно будет использовать в качестве закрытого ключа для расшифрования сообщения. Значение e должно удовлетворять соотношению

$$ed \bmod f = 1.$$

Для малых значений f число e можно найти подбором. В общем случае для поиска e можно использовать обобщенный алгоритм Евклида. В данном случае подходит $e = 17$. Действительно,

$$13 \cdot 17 \bmod 20 = 221 \bmod 20 = 1.$$

Теперь пользователь Б должен запомнить свой закрытый ключ *17*, отправить открытый ключ (*13, 33*) пользователю А и уничтожить числа $P = 3$ и $Q = 11$.

Пользователь А, получивший открытый ключ (*13, 33*), увидев, что $N=33$, разбивает исходное сообщение на три блока, причем значение каждого меньше N. Например, пусть имеется три блока $m_1 = 8$, $m_2 = 27$, $m_3 = 5$. Затем пользователь А зашифровывает каждый блок:

$$c_1 = 8^{13} \bmod 33 = 17;$$

$$c_2 = 27^{13} \bmod 33 = 15;$$

$$c_3 = 5^{13} \bmod 33 = 26.$$

Зашифрованное сообщение, состоящее из трех блоков (*17, 15, 26*), передается пользователю Б, который, используя свой закрытый ключ $e = 17$ и $N=33$, расшифровывает сообщение:

$$m_1 = 17^{17} \bmod 33 = 8;$$

$$m_2 = 15^{17} \bmod 33 = 27;$$

$$m_3 = 26^{17} \bmod 33 = 5.$$

Таким образом, абонент Б расшифровал сообщение от абонента А.

6.2.4 Вопросы практического использования алгоритма RSA

Долгое время алгоритм RSA активно используется как в виде самостоятельных криптографических продуктов, так и в качестве встроенных средств в популярных приложениях. Открытое шифрование на базе алгоритма RSA применяется:

– в популярном пакете шифрования PGP;

– в операционной системе Windows;

– в различных Интернет-браузерах;

– банковских компьютерных системах.

Кроме того, различные международные стандарты шифрования с открытым ключом и формирования цифровой подписи используют RSA в качестве основного алгоритма.

Для обеспечения высокой надежности шифрования необходимо, чтобы выступающее в качестве модуля число N было очень большим – несколько сотен или тысяч бит. Только в этом случае будет практически невозможно по открытым параметрам определить закрытый ключ. Так, известно, что в конце 1995 года удалось практически реализовать раскрытие шифра RSA для 500-значного модуля.

Для этого с помощью сети Интернет было задействовано более тысячи компьютеров.

Авторы RSA рекомендовали использовать следующие размеры модуля N:

– 768 бит – для частных лиц;

– 1024 бит – для коммерческой информации;

– 2048 бит – для особо секретной информации.

Так как с момента получения этих рекомендаций прошло определенное время, поэтому современные пользователи должны делать поправки в сторону увеличения размера ключей. Однако, чем больше размер ключей, тем медленнее работает система. Поэтому увеличивать размер ключа без необходимости не имеет смысла.

С размером ключей связан и другой (*вычислительный*) аспект реализации RSA. При использовании алгоритма вычисления необходимы как при создании ключей, так и при зашифровании/расшифровании, при этом, чем больше размер ключей, тем труднее производить расчеты. Для работы с громадными числами приходится использовать аппарат длинной арифметики. Числа, состоящие из многих сотен бит, не умещаются в регистры большинства микропроцессоров и их приходится обрабатывать по частям. При этом как зашифрование, так и расшифрование включают возведение большого целого числа в целую степень по модулю N. При прямых расчетах промежуточные значения были бы невообразимыми.

Чтобы упростить процесс вычислений используют специальные алгоритмы для работы с большими числами, основанные на свойствах модульной арифметики, а также оптимизацию при возведении в степень.

Алгоритм RSA реализуется как программным, так и аппаратным путем. Многие мировые фирмы выпускают специализированные микросхемы, производящие шифрование алгоритмом RSA. Программные реализации значительно медленнее, чем аппаратные. К достоинствам программного шифрования RSA относится возможность:

– гибкой настройки параметров;

– интеграции в различные программные пакеты.

В целом, и программная, и аппаратная реализации RSA обычно требуют для выполнения в тысячи раз большего времени по сравнению с симметричными алгоритмами, например ГОСТ 28147-89.

Алгоритм RSA может использоваться для формирования электронной цифровой подписи, а также и для обмена ключами.

Возможность применения алгоритма RSA для получения электронной подписи связана с тем, что закрытый и открытый ключи в этой системе равноправны. Каждый из ключей, d или e, могут использоваться как для зашифрования, так и для расшифрования.

Это свойство выполняется не во всех криптосистемах с открытым ключом.

6.3 Алгоритм цифровой подписи Эль Гамаля

6.3.1 Общие сведения о системе Эль Гамаля

Асимметричный алгоритм, предложенный в 1985 году Эль Гамалем (T.ElGamal), является универсальным. Он может быть использован для решения всех трех основных криптографических задач:

– для шифрования данных;

– для формирования цифровой подписи;

– для согласования общего ключа.

Кроме того, возможны модификации алгоритма для схем проверки пароля, доказательства идентичности сообщения и другие варианты. Безопасность этого алгоритма, так же как и алгоритма Диффи-Хеллмана, основана на трудности вычисления дискретных логарифмов. Этот алгоритм фактически использует схему Диффи-Хеллмана, чтобы сформировать общий секретный ключ для абонентов, передающих друг другу сообщение, и затем сообщение зашифровывается путем умножения его на этот ключ.

6.3.2 Подготовка ключей в системе Эль Гамаля

Во всех случаях, и при шифровании, и при формировании электронной цифровой подписи каждому пользователю необходимо сгенерировать пару ключей. Для этого, так же как и в схеме Диффи-Хеллмана, выбираются некоторое большое простое число P и число A, такие, что различные степени A представляют собой различные числа по модулю P. Числа P и A могут передаваться в открытом виде и быть общими для всех абонентов сети.

Затем каждый абонент группы выбирает свое секретное число X_i, причем $1 < x_i < P-1$, и вычисляет соответствующее ему открытое число

$$Y_i = A^{x_i} \bmod P.$$

Таким образом, каждый пользователь может сгенерировать закрытый ключ X_i и открытый ключ Y_i.

Информация о необходимых параметрах системы приведена в таблице 6.4.

Таблица 6.4 – Параметры системы Эль Гамаля

	Общие параметры	Открытый ключ	Закрытый ключ
Пользователь 1	*P, A*	Y_1	X_1
…		...	...
Пользователь i		Y_i	X_i

6.3.3 Протокол Эль Гамаля для шифрования данных

Рассмотрим, как производится шифрование данных в системе Эль Гамаля.

Сообщение, предназначенное для зашифрования, должно быть представлено в виде одного числа или набора чисел, каждое из которых меньше *P*. Пусть пользователь 1 хочет передать пользователю 2 сообщение *M*. В этом случае выполняется следующий криптографический протокол.

1) Первый пользователь выбирает случайное число *k*, взаимно простое с *P-1*, и вычисляет числа

$$r = A^k \bmod P,$$

$$e = M \cdot Y_2^k \bmod P,$$

где Y_2 – открытый ключ пользователя 2. Число *k* держится в секрете.

2) Пара чисел (*r, e*), являющаяся шифртекстом, передается второму пользователю.

3) Второй пользователь, получив (*r, e*), для расшифрования сообщения вычисляет

$$M = e \cdot r^{P-1-X_2} \bmod P,$$

где X_2 – закрытый ключ пользователя 2.

В результате у второго пользователя получается исходное сообщение *M*.

Если злоумышленник узнает или перехватит *P, A,* Y_2, *r, e*, то он не сможет по ним раскрыть *M*. Это связано с тем, что противник не знает параметр *k*, выбранный первым пользователем для зашифрования сообщения *M*.

Вычислить каким-либо образом число *k* практически невозможно, так как это задача дискретного логарифмирования. Следовательно, злоумышленник не может вычислить и значение *M*, так как *M* было умножено на неизвестное ему число. Противник также не может воспроизвести действия законного получателя сообщения (второго абонента), так как ему не известен закрытый ключ X_2 (вычисление X_2 на основании Y_2 – также задача дискретного логарифмирования).

По аналогичному алгоритму может производиться и согласование ключа, используемого для симметричного шифрования больших объемов данных. Более того, алгоритм Эль Гамаля на практике целесообразно использовать именно для согласования общего ключа сессии, а не прямого шифрования больших сообщений. Это связано с тем, что в алгоритме используются операции возведения в степень и умножения по большому модулю. Так же как и в алгоритмах RSA и Диффи-Хеллмана, операции производятся над большими, состоящими из нескольких сотен или тысяч бит, числами. Поэтому шифрование больших сообщений производится крайне медленно.

Пример шифрования. Пусть два абонента, обменивающиеся через Интернет зашифрованными сообщениями, имеют следующие общие параметры:

$$P = 11, \quad A = 7.$$

Кроме того, пользователи 1 и 2 имеют пары закрытых и открытых ключей, вычисляемые по системе Эль Гамаля:

– пользователь 1 имеет закрытый ключ $X_1 = 3$ и открытый ключ

$$Y_1 = 7^3 \bmod 11 = 2;$$

– пользователь 2 имеет закрытый ключ $X_2 = 9$ и открытый ключ

$$Y_2 = 7^9 \bmod 11 = 8.$$

Первый абонент желает передать второму сообщение. Для этого первый абонент запрашивает из центра распределения ключей открытый ключ второго абонента $Y_2 = 8$. Теперь он может зашифровать свое сообщение, которое в числовом виде пусть имеет значение $M = 9$.

Первый абонент выбирает случайно число k, например $k = 7$. Число k должно быть взаимно простым с $P\text{-}1$. Значение $k = 7$ не имеет общих делителей с $P\text{-}1 = 10$, значит, оно вполне подходит.

Первый абонент зашифровывает свое сообщение по формулам:

$$r = A^k \bmod P = 7^7 \bmod 11 = 6;$$

$$e = M \cdot Y_2^k \bmod P = 9 \cdot 8^7 \bmod 11 = 7.$$

Пара чисел (*6, 7*) будет представлять собой шифртекст и передается второму пользователю. Второй пользователь, получив (*6, 7*) и используя свой закрытый ключ $X_2 = 9$ для расшифрования сообщения, вычисляет

$$M = e \cdot r^{P-1-X_2} \bmod P = 7 \cdot 6^{11-1-9} \bmod 11 = 7 \cdot 6^1 \bmod 11 = 9.$$

В результате он действительно получает исходное сообщение *M*.

Следует отметить, что схема Эль Гамаля является характерным примером подхода, который допускает пересылку сообщения M в открытой форме вместе с присоединенным аутентификатором (*a, b*). В таких случаях процедура установления подлинности принятого сообщения состоит в проверке соответствия аутентификатора сообщению.

Схема электронной цифровой подписи Эль Гамаля имеет ряд преимуществ по сравнению со схемой цифровой подписи RSA:

1) При заданном уровне стойкости алгоритма цифровой подписи целые числа, участвующие в вычислениях, имеют запись на 25% короче, что уменьшает сложность вычислений почти в два раза и позволяет заметно сократить объем используемой памяти.

2) При выборе модуля *P* достаточно проверить, что это число является простым и что у числа (*P-1*) имеется большой простой множитель (то есть всего два достаточно просто проверяемых условия).

3) Процедура формирования подписи по схеме Эль Гамаля не позволяет вычислять цифровые подписи под новыми сообщениями без знания закрытого ключа (как в RSA).

Однако алгоритм электронной цифровой подписи Эль Гамаля имеет и некоторые недостатки по сравнению со схемой подписи RSA. В частности, для него длина цифровой подписи получается в 1,5 раза больше.

6.4 Алгоритм цифровой подписи DSA

Алгоритм цифровой подписи *DSA (Digital Signature Algorithm)* предложен в 1991 году в НИСТ США для использования в стандарте цифровой подписи DSS (Digital Signature Standard). Алгоритм DSA является развитием алгоритмов цифровой подписи Эль Гамаля и К. Шнорра.

Отправитель и получатель электронного документа используют при вычислении большие целые числа:

– G и P – простые числа, длина L бит каждого из них лежит в пределах $512 \le L \le 1024$;

– q – простое число длиной 160 бит (делитель числа (P-1)).

Числа G, P, q являются открытыми и могут быть общими для всех абонентов информационной системы.

Отправитель выбирает случайное целое число X, причем $1 < X < q$. Число X является закрытым ключом отправителя для формирования электронной цифровой подписи.

Затем отправитель вычисляет значение $Y = G^X \bmod P$. Число Y является открытым ключом для проверки подписи отправителя и передается всем получателям документов.

Этот алгоритм также предусматривает использование односторонней функции хеширования $h(\cdot)$. В стандарте DSS определен алгоритм безопасного хеширования SHA (Secure Hash Algorithm).

Для того, чтобы подписать документ M, отправитель хеширует его в целое хеш-значение

$$m = h(M),$$

где $1 < m < q$.

Затем отправитель генерирует случайное целое число K, причем $1 < K < q$, и вычисляет число

$$r = \left(G^K \bmod P\right) \bmod q.$$

На следующем шаге отправитель вычисляет с помощью закрытого ключа X целое число

$$s = \left((m + r \cdot X)/K\right) \bmod q.$$

Пара чисел (r, s) образует цифровую подпись

$$S = (r, s)$$

под документом M.

Таким образом, подписанное сообщение представляет собой тройку чисел (M, r, s).

Получатель подписанного сообщения (M, r, s) проверяет выполнение условий

$$0 < r < q, \quad 0 < s < q$$

и отвергает подпись, если хотя бы одно из этих условий не выполнено.

Затем получатель вычисляет значение

$$w = (1/s) \bmod q,$$

хеш-значение

$$m = h(M)$$

и числа:

$$u_1 = (m \cdot w) \bmod q;$$

$$u_2 = (r \cdot w) \bmod q.$$

Далее получатель с помощью открытого ключа Y вычисляет значение

$$v = \left(\left(G_1^u \cdot Y_2^u \right) \bmod P \right) \bmod q$$

и проверяет выполнение условия

$$v = r.$$

Если условие $v = r$ выполняется, тогда подпись $S=(r, s)$ под документом M признается получателем подлинной.

Можно строго математически доказать, что последнее равенство будет выполняться тогда, и только тогда, когда подпись $S=(r, s)$ под документом M получена с помощью именно того закрытого ключа X, из которого был получен открытый ключ Y. Таким образом, можно надежно удостовериться, что отправитель сообщения владеет именно данным закрытым ключом X (не раскрывая при этом значения ключа X) и что отправитель подписал именно данный документ M.

По сравнению с алгоритмом цифровой подписи Эль Гамаля алгоритм DSA имеет следующие основные преимущества:

1) При любом допустимом уровне стойкости, то есть при любой паре чисел G и P (от 512 до 1024 бит), числа q, X, r, s имеют длину по 160 бит, сокращая длину подписи до 320 бит.

2) Большинство операций с числами K, r, s, X при вычислении подписи производится по модулю числа q длиной 160 бит, что сокращает время вычисления подписи.

3) При проверке подписи большинство операций с числами u_1, u_2, v, w также производится по модулю числа q длиной 160 бит, что сокращает объем памяти и время вычисления.

Недостатком алгоритма DSA является то, что при подписывании и при проверке подписи приходится выполнять сложные операции деления по модулю q:

$$s = ((m + rX)/K) (\bmod q), \quad w = (1/s) (\bmod q),$$

что не позволяет получать максимальное быстродействие.

Следует отметить, что реальное исполнение алгоритма DSA может быть ускорено с помощью выполнения предварительных вычислений. Заметим, что значение r не зависит от сообщения M и его хеш-значения m. Можно заранее создать строку случайных значений K и затем для каждого из этих значений вычислить значения r. Можно также заранее вычислить

обратные значения K^{-1} для каждого из значений K. Затем, при поступлении сообщения M, можно вычислить значение s для данных значений r и K^{-1}.

Эти предварительные вычисления значительно ускоряют работу алгоритма DSA.

6.5 Алгоритмы ГОСТ Р34.10-94 и ГОСТ Р34.10-2001

Отечественный стандарт цифровой подписи обозначается как *ГОСТ Р 34.10-94*. Алгоритм цифровой подписи, определяемый этим стандартом, концептуально близок к DSA.

В нем используются следующие параметры:

p – большое простое число длиной от 509 до 512 бит либо от 1020 до 1024 бит;

q – простой сомножитель числа (p-1), имеющий длину 254...256 бит;

a – любое число, меньшее (p-1), причем такое, что $a^q \bmod p = 1$;

x – некоторое число, меньшее q;

y – некоторое число, причем $y = a^x \bmod p$.

Кроме того, этот алгоритм использует однонаправленную хеш-функцию $H(x)$, стандарта ГОСТ Р 34.11-94, которая основана на использовании стандартного симметричного алгоритма ГОСТ 28147-89.

Первые три параметра p, q, a являются открытыми и могут быть общими для всех пользователей информационной системы. Число x является закрытым ключом. Число y является открытым ключом.

Чтобы подписать некоторое сообщение M, а затем проверить подпись, выполняются следующий криптографический протокол:

1) Пользователь А генерирует случайное число k, причем $k < q$.

2) Пользователь А вычисляет значения:

$$r = \left(a^k \bmod p\right) \bmod p,$$

$$s = \left(x \cdot r + k\left(H\left(M\right)\right)\right) \bmod p.$$

Если $H\left(M\right) \bmod q = 0$, то значение $H\left(M\right) \bmod q$ принимают равным единице. Если $r=0$, то выбирают другое значение k и начинают снова. Цифровая подпись представляет собой два числа:

$$r \bmod 2^{256}; \quad s \bmod 2^{256}.$$

Пользователь А отправляет эти числа пользователю Б.

3) Пользователь Б проверяет полученную подпись, вычисляя:

$$v = H\left(M\right)^{q-2} \bmod q;$$

$$z_1 = \left(s \cdot v\right) \bmod q;$$

$$z_2 = \left(\left(q - r\right) \cdot v\right) \bmod q;$$

$$u = \left(\left(a_1^z \cdot y_2^z\right) \bmod p\right) \bmod p.$$

4) Если $u = r$, то подпись считается верной.

Различие между этим алгоритмом и алгоритмом DSA заключается в том, что в DSA

$$s = \left(k^{-1}\left(x \cdot r + \left(H\left(M\right)\right)\right)\right) \bmod q,$$

что приводит к другому уравнению верификации.

Следует также отметить, что в отечественном стандарте ЭЦП параметр q имеет длину 256 бит. Хотя западных криптографов вполне устраивает q длиной примерно 160 бит.

Различие в значениях параметра q является отражением стремления разработчиков отечественного стандарта к получению более безопасной подписи. Этот стандарт вступил в действие с начала 1995 года.

Однако в 2001 году был принят новый отечественный стандарт на алгоритм формирования и проверки ЭЦП. Его полное название «ГОСТ Р34.10-2001. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи».

Данный алгоритм был разработан главным управлением безопасности связи Федерального агентства правительственной связи и информации при Президенте Российской Федерации при участии Всероссийского научно-исследовательского института стандартизации. Новый стандарт разрабатывался с целью обеспечения большей стойкости алгоритма генерации ЭЦП.

В основе ГОСТ Р34.10-2001 лежат алгоритмы с использованием операций на *эллиптических кривых*. Стойкость ГОСТ Р34.10-2001 основывается на сложности взятия дискретного логарифма в группе точек эллиптической кривой, а также на стойкости хеш-функции по ГОСТ Р34.11-94. Размер формируемой цифровой подписи – 512 битов.

В целом алгоритм вычислений по алгоритму ГОСТ Р34.10-2001 аналогичен применяемому в предыдущем стандарте ГОСТ Р34.10-94. Сначала генерируется случайное число k, с его помощью вычисляется компонента r подписи. Затем на основе компоненты r, числа k, значения секретного ключа и хеш-значения подписываемых данных формируется s-компонента ЭЦП. При проверке же подписи аналогичным образом проверяется соответствие определенным соотношениям $r, s,$ открытого ключа и хеш-значения информации, подпись которой проверяется. Подпись считается неверной, если соотношения неверны.

Старый ГОСТ Р34.10-94 не отменен, и в настоящее время параллельно действуют два отечественных стандарта на ЭЦП. Однако необходимо отметить, что для прежнего ГОСТа принято ограничение – при реализации ЭЦП по стандарту ГОСТ Р34.10-94 разрешено использовать только 1024-битные значения параметра p.

Использование математического аппарата группы точек эллиптической кривой в новом ГОСТ Р34.10-2001 позволяет существенно сократить порядок модуля p без потери криптостойкости. Так, в стандарте указано, что длина числа p может быть 256 или больше бит.

6.6 Возможные атаки при использовании алгоритмов асимметричного шифрования

6.6.1 Атака «человек-в-середине»

Проанализируем простейший протокол шифрования с открытым ключом с точки зрения возможности проведения противником различных атак.

Этот протокол предусматривает следующие действия пользователей:

– если пользователь А желает передать секретное сообщение пользователю Б так, чтобы никто другой не смог его прочитать, он должен получить от пользователя Б открытый ключ $U_Б$ и зашифровать свое сообщение этим открытым ключом;

– зашифрованное сообщение может пересылаться по любому каналу связи, например, по электронной почте;

– получив сообщение от пользователя А, пользователь Б может расшифровать его своим закрытым ключом $R_Б$.

Такая процедура обмена зашифрованными сообщениями с использованием асимметричного алгоритма не позволит противнику, контролирующему открытый канал связи, по перехваченным открытым ключам и зашифрованным сообщениям восстановить исходные сообщения. Это обеспечивается свойствами односторонней функции, а именно, сложностью вычисления обратной функции.

Однако такая схема уязвима для атак типа «man-in-the-middle» («человек-в-середине»).

Эта атака заключается в следующем. Допустим, противник может не только перехватывать сообщения, но и заменять их другими, то есть имеет возможность осуществлять активную атаку. Это вполне возможно в современных сетях передачи данных, например в Интернете, где информация от одного пользователя передается другому через множество промежуточных узлов, не контролируемых этими пользователями. Злоумышленником может быть, например, системный администратор сети. Такой нарушитель может не только перехватывать сообщения пользователей, но изменять, удалять или заменять их своими. Он может выдавать себя за одного из участников сеанса связи. Вот как может производиться атака «man-in-the-middle»:

1) Пользователь Б посылает пользователю А свой открытый ключ $U_Б$. Противник перехватывает этот ключ, сохраняет его и заменяет его своим открытым ключом $U_П$.

2) Пользователь А шифрует свое сообщение M полученным открытым ключом $U_П$, предполагая, что использует открытый ключ абонента Б, и пересылает зашифрованное сообщение пользователю Б.

3) Злоумышленник перехватывает это сообщение, расшифровывает его своим закрытым ключом $R_П$, читает или меняет, а затем зашифровывает открытым ключом пользователя Б и посылает пользователю Б.

Аналогично взломщик перехватит и открытый ключ пользователя А, чтобы читать ответы пользователя Б. В результате нарушитель сможет чи-

тать (а, возможно, и изменять) всю корреспонденцию абонентов. Пользователи А и Б, скорее всего, ничего не заподозрят, так как у них нет способа проверить, действительно ли они общаются друг с другом.

На практике разработано несколько способов предотвращения атаки «man-in-the-middle». Один из способов заключается в разделении каждого зашифрованного сообщения на две части, каждая из которых бесполезна без другой. Части сообщения пересылаются по очереди и не могут быть расшифрованы по отдельности. Вот как может выглядеть этот криптографический протокол для обмена сообщениями между двумя пользователями А и Б:

1) Пользователи А и Б обмениваются открытыми ключами.

2) Пользователь А зашифровывает свое сообщение открытым ключом пользователя Б и пересылает половину зашифрованного сообщения пользователю Б.

3) Пользователь Б зашифровывает свое сообщение открытым ключом пользователя А и пересылает половину зашифрованного сообщения пользователю А.

4) Пользователь А пересылает вторую половину зашифрованного сообщения пользователю Б.

5) Пользователь Б соединяет обе полученные половины сообщения от пользователя А и расшифровывает его своим закрытым ключом. Затем посылает вторую половину своего зашифрованного сообщения пользователю А.

6) Пользователь А складывает полученные от пользователя Б половины сообщения и расшифровывает его своим закрытым ключом.

Этот усовершенствованный протокол не позволяет противнику читать или изменять корреспонденцию пользователей А и Б. Нарушитель, как и раньше, может подменить открытые ключи абонентов, а также перехватить передаваемые между ними данные. Однако, получив на втором шаге протокола в свое распоряжение первую половину зашифрованного сообщения от А к Б, он не сможет расшифровать ее своим закрытым ключом и снова зашифровать открытым ключом абонента Б. Абоненты А и Б также не смогут прочитать сообщения до окончания протокола (шагов 5 и 6), но в этом нет ничего плохого, так как в результате они получат корректную корреспонденцию.

Для осуществления протокола процесс разделения сообщения на две части может производиться разными способами, например, каждый нечетный байт помещается в первое сообщение, а каждый четный – во второе или как-то иначе.

Атаки «человек-в-середине» можно избежать и другими способами, например, добавляя к передаваемым открытым ключам цифровые подписи специального удостоверяющего центра.

6.6.2 Атака на основе выбранного открытого текста

Алгоритмы с открытым ключом чувствительны к атакам по выбранному открытому тексту. Известно, что такая атака имеет место, если криптоаналитик имеет возможность не только использовать предоставленные ему

пары «текст-шифртекст», но и сам формировать нужные ему тексты и зашифровывать их.

Факт возможности проведения атаки по выбранному открытому тексту объясняется следующим образом.

Предположим, используется асимметричный алгоритм F для согласования общего секретного ключа. Пусть один из абонентов отправил другому 64-битовый сеансовый ключ K, зашифрованный открытым ключом y другого абонента $C=F(K, y)$. Злоумышленник, перехватив зашифрованное сообщение C, не сможет его, конечно, дешифровать, так как не имеет закрытого ключа x. Однако нарушитель может поступить по-другому, а именно, попытаться подобрать подходящее значение K. Для этого нужно зашифровать все возможные 64-битовые комбинации открытых текстов открытым ключом y и сравнить результаты с C. Это возможно, так значения y и C передавались в открытом виде. Особенно актуальна угроза такой атаки, если число возможных исходных сообщений не очень велико, например, если длина исходного сообщения мала или если не все исходные тексты допустимы на практике.

Для того, чтобы избежать возможности такой атаки, используют *рандомизированные* (или *вероятностные*) алгоритмы шифрования и формирования ЭЦП с открытым ключом. Такие алгоритмы зашифровывают одно и то же сообщение при наличии одинакового ключа каждый раз по-разному, так как используют некоторый случайный элемент.

Примерами рандомизированных алгоритмов с открытым ключом могут служить алгоритмы Эль Гамаля и алгоритмы формирования ЭЦП по ГОСТ Р34.10.

Другим вариантом предотвращения атаки на основе выбранного открытого текста является добавление в зашифровываемое сообщение некоторой дополнительной «случайной» информации, например, метки даты времени.

6.7 Контрольные вопросы

1) Дайте определение простого и составного числа. Приведите по три примера простых и составных чисел.

2) Дайте определение понятия «взаимно простые числа». Приведите примеры взаимно простых чисел и чисел, не являющихся взаимно простыми.

3) Сформулируйте основную теорему арифметики.

4) В чем заключается задача факторизации?

5) Дайте определение наибольшего общего делителя.

6) Сформулируйте алгоритм Евклида для нахождения наибольшего общего делителя двух чисел.

7) Сформулируйте малую теорему Ферма.

8) Сформулируйте теорему Ферма-Эйлера (для случая системы RSA).

9) Сформулируйте обобщенный (расширенный) алгоритм Евклида.

10) Сформулируйте принципы выполнения «операции взятия по модулю». Приведите примеры выполнения этой операции и поясните их.

11) Что такое инверсия по модулю n?

12) Для каких целей может применяться алгоритм RSA?

13) Опишите процесс шифрования с использованием алгоритма RSA.

14) Для каких целей может применяться алгоритм Эль Гамаля?.

15) Опишите последовательность действий при использовании алгоритма Эль Гамаля.

16) Какие атаки возможны при использовании алгоритмов шифрования с открытым ключом?

6.8 Упражнения

1) Являются ли простыми числа: *37, 59, 67, 93, 101, 111, 231*?

2) Являются ли взаимно простыми числа:

16 и *37*;

16, *37* и *38*;

5, *9*, *27* и *54*;

4, *7*, *15*, *59*.

3) Определите число натуральных чисел, не превосходящих *59* и, взаимно простых с *59*.

4) Определите число натуральных чисел, не превосходящих *143* и, взаимно простых с *143*.

5) Определите наибольший общий делитель чисел *187* и *153*.

6) Вычислите:

38 по модулю 10

$38 \cdot 57$ по модулю 11.

7) С помощью обобщенного алгоритма Евклида найдите числа x и y, удовлетворяющие уравнению $33x + 16y = НОД(33,16)$.

8) Вычислите:

$7^{-1} \bmod 10$;

$3^{-1} \bmod 11$.

9) Пусть пользователь А хочет передать пользователю Б сообщение $M = 10$, зашифрованное с помощью алгоритма RSA. Пользователь Б имеет следующие параметры: $P = 7$, $Q = 11$, $d = 47$. Опишите процесс передачи сообщения M пользователю Б.

10) Пользователю системы RSA с параметрами $N = 33$, $d = 3$ передано зашифрованное сообщение $c = 13$. Расшифруйте это сообщение, взломав систему RSA пользователя.

11) В системе связи, применяющей шифр Эль Гамаля, пользователь 1 желает передать сообщение M пользователю 2. Найдите недостающие параметры при следующих заданных параметрах $P = 19$, $A = 2$, $X_2 = 3$, $k = 5$, $M = 10$.

7 ПРОБЛЕМЫ ИДЕНТИФИКАЦИИ И АУТЕНТИФИКАЦИИ ПОЛЬЗОВАТЕЛЕЙ И ДОКУМЕНТОВ В ИНФОРМАЦИОННЫХ СИСТЕМАХ

7.1 Основные понятия и концепции защиты пользователей и документов в информационных системах

На практике с каждым объектом компьютерной информационной системы (компьютер, пользователь, документ и т.д.) связана некоторая информация, однозначно идентифицирующая его. Обычно это число, строка символов, алгоритм, определяющий данный объект. Эту информацию называют *идентификатором объекта*.

При наличии у объекта некоторого идентификатора, зарегистрированного в компьютерной системе, он называется *законным (легальным) объектом*; остальные объекты относятся к *незаконным (нелегальным)*.

Идентификация объекта – одна из функций подсистемы защиты по распознаванию объекта. Эта функция выполняется в первую очередь, когда объект делает попытку войти в компьютерную систему. Если процедура идентификации завершается успешно, данный объект считается законным для данной системы.

Аутентификация объекта (проверка подлинности объекта) – это следующая функций подсистемы защиты. Она устанавливает, является ли данный объект именно таким, каким он себя объявляет.

Авторизация (предоставление полномочий) – третья функция подсистемы защиты. После того как объект идентифицирован и подтверждена его подлинность, можно установить сферу его действия и доступные ему ресурсы компьютерной системы.

Названные три процедуры инициализации являются *процедурами защиты* и относятся к одному объекту компьютерной системы.

При защите каналов передачи данных подтверждение подлинности (*аутентификация*) объектов означает взаимное установление подлинности объектов, связывающихся между собой по линиям связи. Процедура подтверждения подлинности выполняется обычно в начале сеанса в процессе установления соединения абонентов.

Цель процедуры аутентификации – обеспечить уверенность, что соединение установлено с законным объектом и вся информация дойдет до места назначения.

После того как соединение установлено, необходимо обеспечить выполнение *требований защиты* при обмене сообщениями (документами):

а) получатель должен быть уверен в подлинности источника данных;

б) получатель должен быть уверен в подлинности передаваемых данных;

в) отправитель должен быть уверен в доставке данных получателю;

г) отправитель должен быть уверен в подлинности доставленных данных.

Для выполнения требований (а) и (б) средством защиты является *цифровая подпись.*

Для выполнения требований (в) и (г) отправитель должен получить *уведомление о вручении* с помощью удостоверяющей почты (certified mail). Средством защиты в такой процедуре является *цифровая подпись* подтверждающего ответного сообщения, которое в свою очередь является доказательством пересылки исходного сообщения (документа).

Если эти четыре требования реализованы в компьютерной системе, то гарантируется защита данных при их передаче по каналу связи и обеспечивается функция защиты, называемая *функцией подтверждения* (неоспоримости) передачи.

В этом случае отправитель не может отрицать ни факта посылки сообщения (документа), ни его содержания, а получатель не может отрицать ни факта получения сообщения, ни подлинности его содержания.

7.2 Идентификация и механизмы подтверждения подлинности пользователей в компьютерных системах

Для получения доступа к ресурсам компьютерной системы, пользователь должен пройти *процесс представления* компьютерной системы, который включает две стадии:

– идентификацию – пользователь сообщает системе по ее запросу свое имя (идентификатор);

– аутентификацию – пользователь подтверждает идентификацию, вводя в систему уникальную, не известную другим пользователям информацию о себе (например, пароль).

Различают следующие механизмы аутентификации:

– механизмы парольной аутентификации;

– механизмы аутентификации на базе аппаратных средств;

– механизмы биометрической идентификации и аутентификации.

Рассмотрим эти механизмы более подробно.

7.2.1 Механизмы парольной аутентификации

Метод подтверждения подлинности с использованием пароля является простейшим. Он основан на сравнении представляемого пользователем пароля P_A с исходным значением P_A', хранящимся в компьютерном центре (рисунок 7.1).

Так как пароль должен храниться в тайне, он должен зашифровываться перед пересылкой по незащищенному каналу

$$E_K = f(P_A, K)$$

и расшифровываться в компьютерном центре

$$D_K = f\,(E_K,\, K) = P_A,$$

где E_K – зашифрованный пароль;

$f\,(P_A,\, K)$ – функция зашифрования пароля P_A с использованием ключа K;

$f\,(E_K,\, K)$ – функция расшифрования пароля E_K с использованием ключа K.

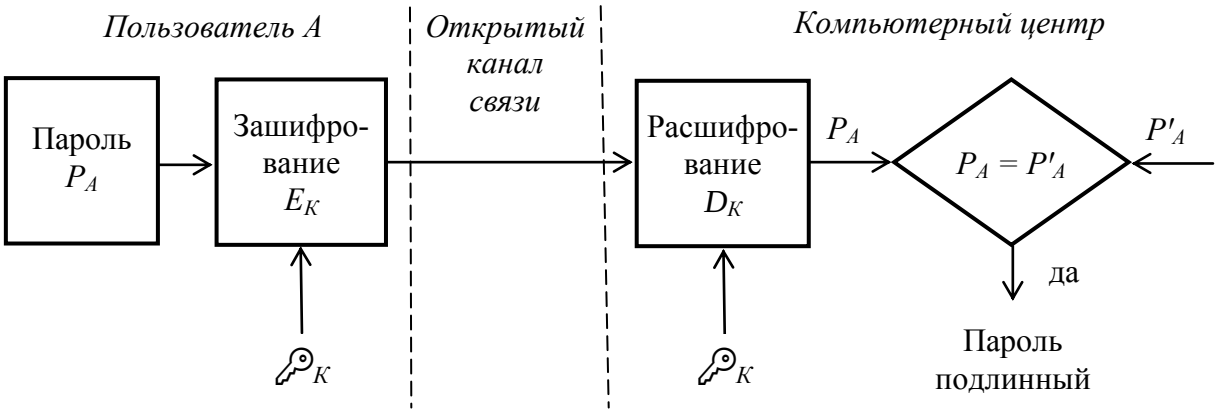


Рисунок 7.1 – Схема простой аутентификации с помощью пароля

Если значения P_A и P_A' совпадают, то пароль P_A считается подлинным, а пользователь – законным.

Если кто-нибудь, не имеющий полномочий для входа в систему, узнает каким-либо образам пароль и идентификационный номер законного пользователя, он получает доступ в систему. Это существенный недостаток парольной аутентификации.

Для устранения этого недостатка получатель не должен раскрывать исходную открытую форму пароля. В этом случае отправитель должен пересылать вместо открытой формы пароля *отображение пароля*, получаемое с использованием односторонней функции $\alpha(\cdot)$ пароля. Это преобразование должно гарантировать невозможность раскрытия противником пароля по его отображению, так как противник наталкивается на неразрешимую числовую задачу.

Например, функция $\alpha(\cdot)$ может быть определена следующим образом:

$$\alpha(P)E_P(ID),$$

где P – пароль отправителя;

ID – идентификатор отправителя;

E_P – процедура шифрования, выполняемая с использованием пароля P в качестве ключа.

Такие функции особенно удобны, если длина пароля и ключа одинаковы.

В этом случае подтверждение подлинности с помощью пароля состоит из пересылки получателю отображения $\alpha(P)$ и сравнения его с предварительно вычисленным и хранимым эквивалентом $\alpha'(P)$.

На практике пароли состоят только из нескольких букв, чтобы дать возможность пользователям запомнить их. Короткие пароли уязвимы к атаке

полного перебора всех вариантов. Для того, чтобы предотвратить такую атаку, функцию $\alpha(P)$ определяют иначе, а именно:

$$\alpha(P) = E_{P+K}(ID),$$

где K и ID – соответственно, ключ и идентификатор отправителя.

Очевидно, значение $\alpha(P)$ вычисляется заранее и хранится в виде $\alpha'(P)$ в *идентификационной таблице* у получателя (рисунок 7.2).

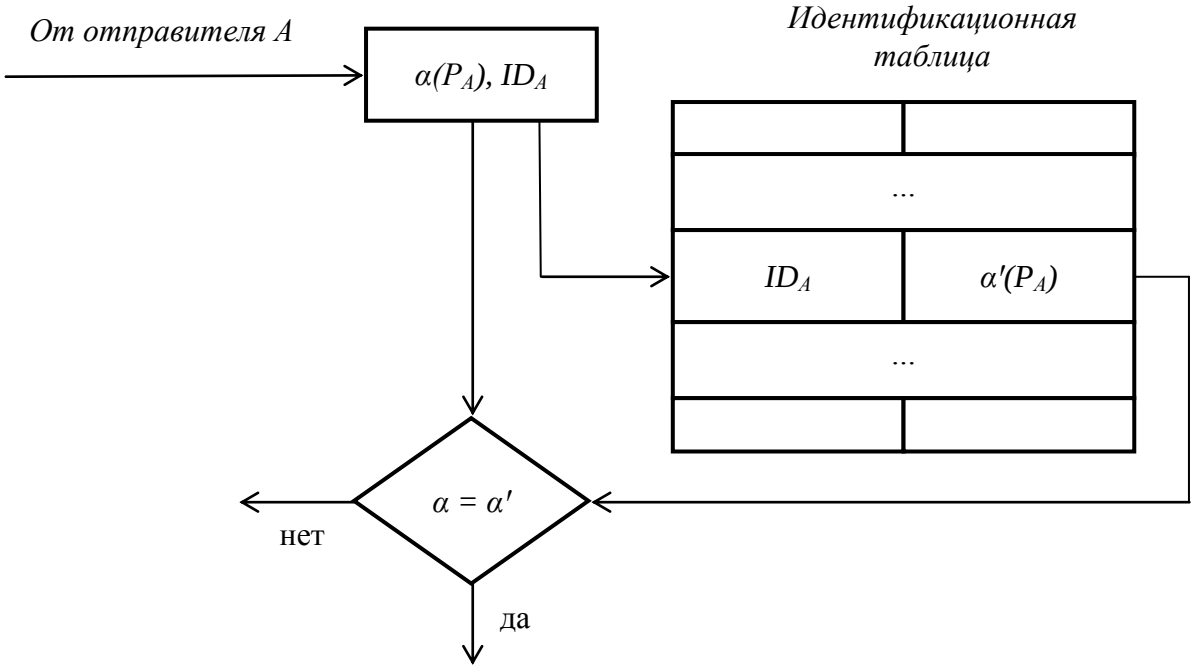


Рисунок 7.2 – Схема аутентификации с помощью пароля с использованием идентификационной таблицы

Подтверждение подлинности состоит из сравнения двух отображений пароля $\alpha(P_A)$ и $\alpha'(P_A)$ и признания пароля P_A, если эти отображения равны.

Однако, любой, кто получит доступ к идентификационной таблице, может незаконно изменить ее содержимое, не опасаясь, что эти действия будут обнаружены. В этом состоит основной недостаток парольной аутентификации.

7.2.2 Механизмы аутентификации на базе аппаратных средств

Более надежным, чем парольная аутентификация, но в то же время более дорогим и потому менее распространенным решением, является *аутентификация на базе аппаратных средств*.

Для удостоверения личности используются специализированные аппаратные средства хранения и генерации идентификационных данных пользователя.

Наиболее распространены *смарт-карты* и *токены*.

По сравнению с парольной, данный вид аутентификации является более защищенным за счет того, что аутентифицироваться может только тот

пользователь, который обладает аппаратным носителем идентификационной информации. Кроме того, отпадает необходимость в запоминании (записывании) сложных для пользователей паролей, что создает возможность несанкционированного доступа.

При аутентификации путем использования *смарт-карты* пользователь вставляет либо прислоняет карту к специальному устройству, считывающему данные с нее. Информация на карте может быть защищена паролем, что обеспечивает повышенный уровень безопасности.

Токены – аппаратные средства, состоящие из микросхемы на некоем компактном носителе и предназначенные для хранения идентификационных данных.

Данные, записанные на токене, считываются с помощью *адаптера*, который подсоединяется к компьютеру пользователя или непосредственно через USB-порт компьютера.

Во время аутентификации пользователь должен подключить токен к USB-порту (или адаптеру) своего компьютера и ввести пароль доступа к токену. Применение токенов, работающих через USB-порт, экономически более выгодно, поскольку не требует приобретения дополнительного оборудования для чтения информации.

7.2.3 Механизмы биометрической идентификации и аутентификации

Среди рассмотренных выше методов, методы *биометрической идентификации и аутентификации* пока наименее распространены. Но в последнее время эти методы позволяют уверенно идентифицировать потенциального пользователя путем измерения физиологических параметров и характеристик человека, особенностей его поведения.

Основными преимуществами биометрических методов идентификации и аутентификации пользователя являются:

– высокая степень достоверности идентификации по биометрическим признакам из-за их уникальности;

– неотделимость биометрических признаков от дееспособной личности;

– трудность фальсификации биометрических признаков.

В качестве *биометрических признаков*, которые могут быть использованы при идентификации потенциального пользователя, можно выделить следующие:

– узор радужной оболочки и сетчатки глаз;

– отпечатки пальцев;

– геометрическая форма руки;

– форма и размеры лица;

– особенности голоса;

– биомеханические характеристики рукописной подписи;

– биомеханические характеристики «клавиатурного почерка».

При регистрации пользователь должен продемонстрировать один или несколько раз свои характерные биометрические признаки. Эти признаки

(известные как подлинные) регистрируются системой как *контрольный образ* законного пользователя.

Этот образ пользователя хранится в электронной форме и используется для проверки идентичности каждого, кто выдает себя за соответствующего законного пользователя. В зависимости от совпадения или несовпадения совокупности предъявленных признаков с зарегистрированными в контрольном образе их предъявивший признается законным пользователем (при совпадении) или нет (при несовпадении).

Системы идентификации *по узору радужной оболочки и сетчатки глаз* могут быть разделены на два класса:

– использующие рисунок радужной оболочки глаза,

– использующие рисунок кровеносных сосудов сетчатки глаза.

Поскольку вероятность повторения данных параметров равна 10^{-78}, эти системы являются наиболее надежными среди всех биометрических систем.

Такие средства идентификации применяются там, где требуется высокий уровень безопасности (например, в зонах военных и оборонных объектов).

Системы идентификации *по отпечаткам пальцев* являются самыми распространенными. Одна из основных причин широкого распространения таких систем заключается в наличии больших банков данных по отпечаткам пальцев. Основными пользователями подобных систем во всем мире являются полиция, различные государственные и некоторые банковские организации.

Системы идентификации *по геометрической форме руки* используют сканеры формы руки, обычно устанавливаемые на стенах. Следует отметить, что подавляющее большинство пользователей предпочитают системы именно этого типа, а не описанные выше.

Системы идентификации *по лицу и голосу* являются наиболее доступными из-за их дешевизны, поскольку большинство современных компьютеров имеют видео- и аудиосредства. Системы данного класса широко применяются при удаленной идентификации субъекта доступа в телекоммуникационных сетях.

Системы идентификации личностей *по динамике рукописной подписи* учитывают интенсивность каждого усилия подписывающего, частотные характеристики написания каждого элемента подписи и начертание подписи в целом.

Системы идентификации *по биомеханическим характеристикам «клавиатурного почерка»* основываются на том, что моменты нажатия и отпускания клавиш при наборе текста на клавиатуре существенно различаются у разных пользователей. Этот динамический ритм набора («клавиатурный почерк») позволяет построить достаточно надежные средства идентификации. В случае обнаружения изменения клавиатурного почерка пользователя ему автоматически запрещается работа на ЭВМ.

Следует отметить, что применение биометрических параметров при идентификации субъектов доступа автоматизированных систем пока *не полу-*

чило надлежащего нормативно-правового обеспечения, в частности в виде стандартов.

Поэтому применение систем биометрической идентификации допускается только в автоматизированных системах, обрабатывающих и хранящих персональные данные, составляющие коммерческую и служебную тайну.

7.3 Идентификация и аутентификация пользователей в виртуальном пространстве сетей

7.3.1 Взаимная проверка подлинности пользователей

Обычно стороны, вступающие в информационный обмен, нуждаются во взаимной проверке подлинности (аутентификации) друг друга. Этот процесс *взаимной аутентификации* выполняют в начале сеанса связи.

Для взаимной проверки подлинности применяют следующие способы:

– механизм запроса-ответа;

– механизм отметки времени («временной штемпель»).

Механизм *запроса-ответа* состоит в следующем.

Если пользователь А хочет быть уверенным, что сообщения, получаемые им от пользователя Б, не являются ложными, он включает в посылаемое для Б сообщение непредсказуемый *элемент-запрос X* (например, некоторое случайное число).

При ответе пользователь Б должен выполнить над этим элементом некоторую операцию (например, вычислить некоторую функцию *f(X)*). Это невозможно осуществить заранее, так как пользователю Б неизвестно, какое случайное число *X* придет в запросе. Получив ответ с результатом действий Б, пользователь может быть уверен, что Б – подлинный.

Недостаток этого метода – возможность установления закономерности между запросом и ответом.

Механизм *отметки времени* подразумевает регистрацию времени для каждого сообщения. В этом случае каждый пользователь сети может определить, насколько «устарело» пришедшее сообщение, и решить не принимать его, поскольку оно может быть ложным.

При использовании отметок времени возникает проблема допустимого временного интервала задержки для подтверждения подлинности сеанса. Ведь сообщение с «временным штемпелем» в принципе не может быть передано мгновенно. Кроме того, компьютерные часы получателя и отправителя не могут быть абсолютно синхронизированы. Какое запаздывание «штемпеля» является подозрительным?

В обоих случаях для защиты механизма контроля следует применять шифрование, чтобы быть уверенным, что ответ послан не злоумышленником.

Для взаимной проверки подлинности обычно используют *процедуру «рукопожатия»*. Эта процедура базируется на указанных выше механизмах контроля и заключается во взаимной проверке ключей, используемых сторо-

нами. Иначе говоря, стороны признают друг друга законными партнерами, если докажут друг другу, что обладают правильными ключами.

Процедуру рукопожатия обычно применяют в компьютерных сетях при организации сеанса связи между пользователями, пользователем и хост-компьютером, между хост-компьютерами и т.д.

Рассмотрим в качестве примера процедуру рукопожатия для двух пользователей А и Б. Пусть приценяется симметричная криптосистема. То есть пользователи А и Б разделяют один и тот же секретный ключ $K_{АБ}$. Вся процедура показана на рисунке 7.3.

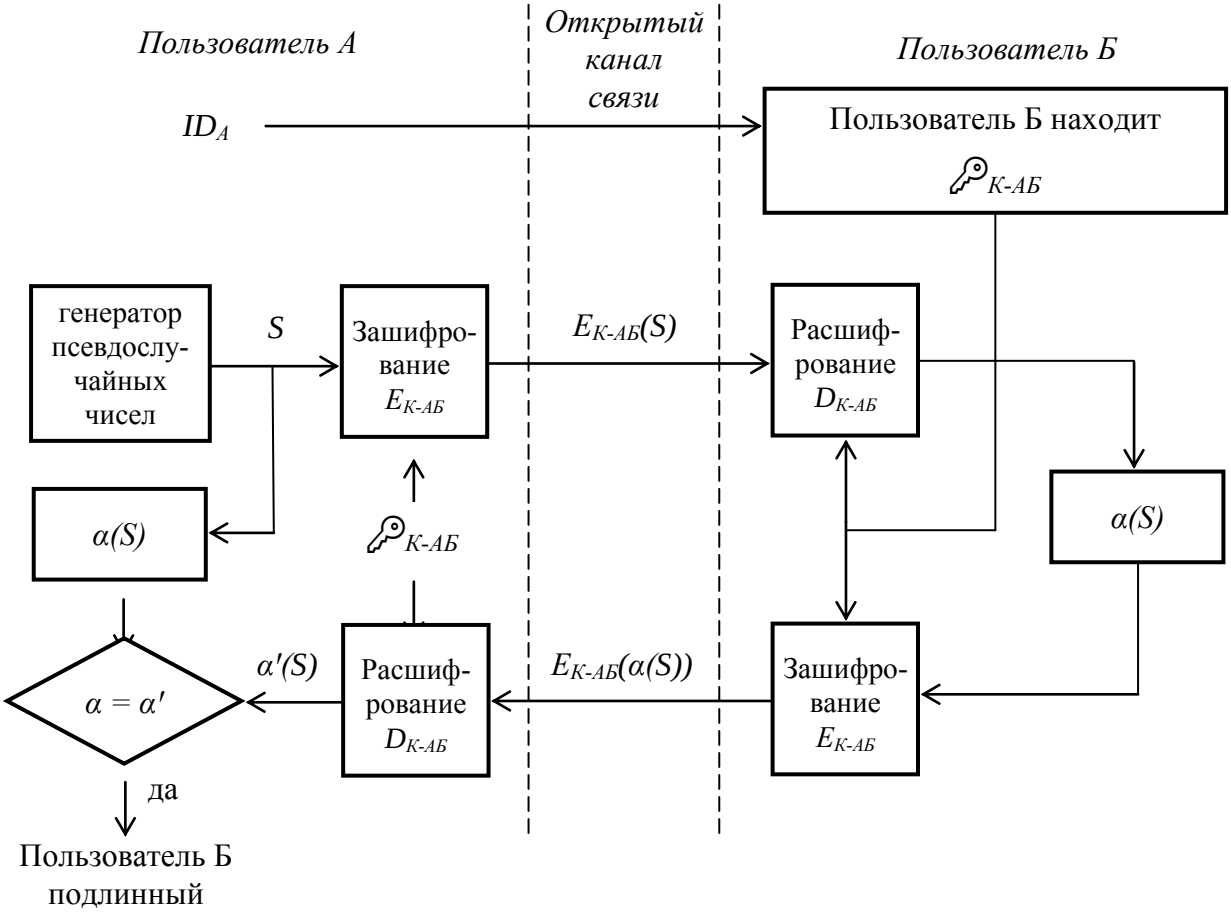


Рисунок 7.3 – Схема процедуры рукопожатия (пользователь А проверяет подлинность пользователя Б)

Процедура рукопожатия для двух пользователей А и Б состоит в следующем:

– пользователь А инициирует процедуру рукопожатия, отправляя пользователю Б свой идентификатор ID_A в открытой форме;

– пользователь Б. получив идентификатор ID_A, находит в базе данных секретный ключ $K_{К-АБ}$ и вводит его в свою криптосистему;

– пользователь А генерирует случайную последовательность S с помощью генератора псевдослучайных чисел и отправляет ее пользователю Б в виде криптограммы $E_{К-АБ}(S)$;

– пользователь Б расшифровывает эту криптограмму и раскрывает исходный вид последовательности S;

– затем оба пользователя А и Б преобразуют последовательность S, используя открытую одностороннюю функцию $\alpha(\cdot)$;

– пользователь Б шифрует сообщение $\alpha(S)$ и отправляет эту криптограмму пользователю А;

– пользователь А расшифровывает эту криптограмму и сравнивает полученное сообщение $\alpha'(S)$ с исходным $\alpha(S)$. Если эти сообщения равны, пользователь А признает подлинность пользователя Б.

Очевидно, пользователь Б проверяет подлинность пользователя А таким же способом. Обе эти процедуры образуют *процедуру рукопожатия*, которая обычно выполняется в самом начале любого сеанса связи между любыми двумя сторонами в компьютерных сетях.

Достоинством модели рукопожатия является то, что ни один из участников сеанса связи не получает никакой секретной информации во время процедуры подтверждения подлинности.

Иногда пользователи хотят иметь *непрерывную проверку подлинности* отправителей в течение всего сеанса связи. Один из простейших способов непрерывной проверки подлинности показан на рисунке 7.4.

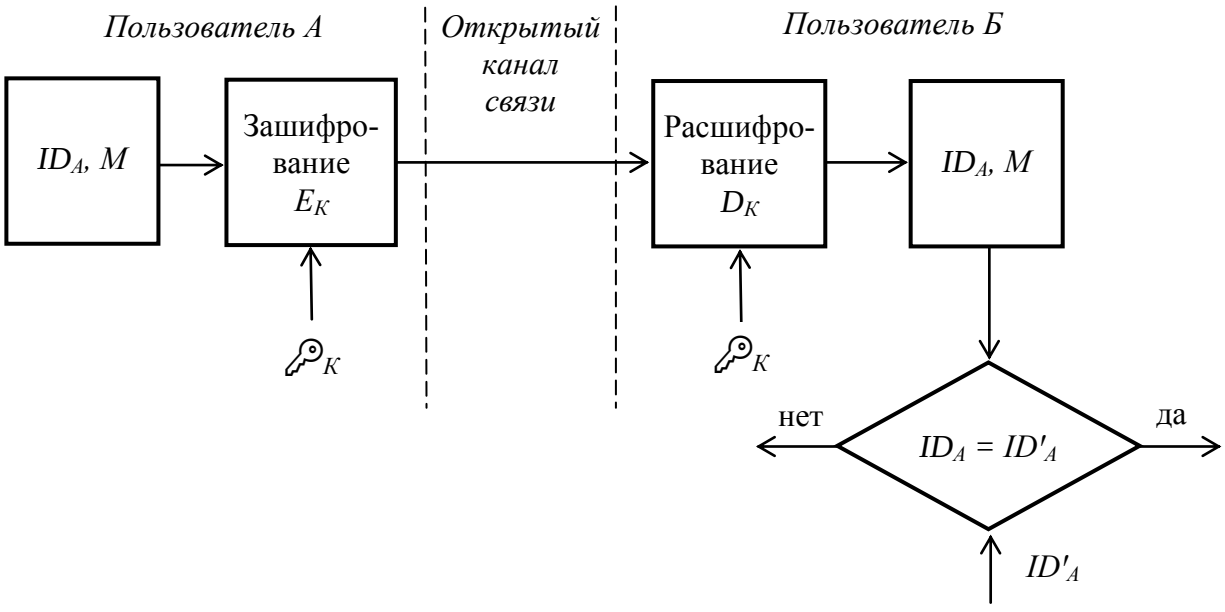


Рисунок 7.4 – Схема непрерывной проверки подлинности отправителя

Передаваемая криптограмма имеет вид $E_K(ID_A, M)$, где ID_A – идентификатор отправителя А; M – сообщение.

Получатель Б, принявший эту криптограмму, расшифровывает ее и раскрывает пару (ID_A, M). Если принятый идентификатор ID_A совпадает с хранимым значением ID_A, получатель Б признает эту криптограмму.

Другой вариант непрерывной проверки подлинности использует вместо идентификатора отправителя его секретный пароль. Заранее подготовленные пароли известны обеим сторонам.

Пусть P_A и $P_Б$ – пароли пользователей А и Б соответственно. Тогда пользователь А создает криптограмму

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

Получатель криптограммы расшифровывает ее и сравнивает пароль, извлеченный из этой криптограммы, с исходным значением. Если они равны, получатель признает эту криптограмму.

Процедура рукопожатия была рассмотрена в предположении, что пользователи А и Б уже имеют общий секретный сеансовый ключ. Реальные процедуры предназначены для распределения ключей между подлинными партнерами и включают как этап распределения ключей, так и этап собственно подтверждения подлинности партнеров по информационному обмену.

7.3.2 Протоколы идентификации с нулевой передачей знаний

Широкое распространение интеллектуальных карт (например, смарт-карт) для разнообразных коммерческих, гражданских и военных применений (кредитные карты, карты социального страхования, карты доступа в охраняемое помещение, компьютерные пароли и ключи, и т.п.) потребовало обеспечения безопасной идентификации таких карт и их владельцев.

Во многих приложениях главная проблема заключается в том, чтобы при предъявлении интеллектуальной карты оперативно обнаружить обман и отказать обманщику в допуске, ответе или обслуживании.

Для безопасного использования интеллектуальных карт разработаны *протоколы идентификации с нулевой передачей знаний*. Секретный ключ владельца карты становится неотъемлемым признаком его личности.

Доказательство знания этого секретного ключа с нулевой передачей этого знания служит доказательством подлинности личности владельца карты.

Схему идентификации с нулевой передачей знаний предложили в 1986 году У. Фейге, А. Фиат и А. Шамир. Эта схема является наиболее известным доказательством идентичности с нулевой передачей конфиденциальной информации. Для выявления ее основной концепции рассмотрим *упрощенный вариант схемы идентификации с нулевой передачей знаний*.

Прежде всего, выбирают случайное значение модуля n, который является произведением двух больших простых чисел.

Модуль n должен иметь длину 512...1024 бит. Это значение n может быть представлено группе пользователей, которым придется доказывать свою подлинность.

В процессе идентификации участвуют две стороны:

– сторона А, доказывающая свою подлинность;

– сторона Б, проверяющая представляемое стороной А доказательство.

Для того чтобы сгенерировать открытый и закрытый ключи для стороны А, доверенный арбитр (Центр) выбирает некоторое число V, которое является квадратичным вычетом по модулю n.

Иначе говоря, выбирается такое число V, что сравнение

$$x^2 = V(\text{mod } n)$$

имеет решение и существует целое число

$$V^{-1} \text{ mod } n.$$

Выбранное значение V является открытым ключом для А.

Затем вычисляют наименьшее значение S, для которого

$$S = sqrt(V^{-1})(\text{mod } n).$$

Это значение S является *закрытым ключом* для А.

Теперь можно приступить к выполнению *протокола идентификации*.

1) Сторона А выбирает некоторое случайное число r, причем $r < n$. Затем она вычисляет

$$x = r^2 \text{ mod } n$$

и отправляет x стороне Б.

2) Сторона Б посылает А случайный бит b.

3) Если $b = 0$, тогда А отправляет r стороне Б. Если b = 1, то А отправляет стороне Б

$$y = r \cdot S \text{ mod } n.$$

4) Если $b = 0$, сторона Б проверяет, что

$$x = r^2 \text{ mod } n,$$

чтобы убедиться, что А знает $sqrt(x)$. Если $b = 1$, сторона Б проверяет, что

$$x = y^2 \cdot V \text{ mod } n,$$

чтобы быть уверенной, что А знает $sqrt(V^1)$.

Эти шаги образуют один цикл протокола, называемый *аккредитацией*.

Стороны А и Б повторяют этот цикл t раз при разных случайных значениях r и b до тех пор, пока Б не убедится, что А знает значение S.

Если сторона А не знает значения S, она может выбрать такое значение r, которое позволит ей обмануть сторону Б, если Б отправит ей $b=0$, либо А может выбрать такое r, которое позволит обмануть Б, если Б отправит ей $b=1$.

Но этого невозможно сделать в обоих случаях. Вероятность того, что А обманет Б в одном цикле, составляет 1/2. Вероятность обмануть Б в t циклах равна $(1/2)^t$.

Для того, чтобы этот протокол работал, сторона А никогда не должна повторно использовать значение r. Если А поступила бы таким образом, а сторона Б отправила бы стороне А на шаге 2 другой случайный бит b, то Б имела бы оба ответа А. После этого Б может вычислить значение S, и для А все закончено.

Лучшими характеристиками обладает *параллельная схема идентификации с нулевой передачей знаний*, которая позволяет увеличить число аккредитаций, выполняемых за один цикл, и тем самым уменьшить длительность процесса идентификации.

7.4 Проблема аутентификации данных и электронная цифровая подпись

При электронном документообороте существенно снижаются затраты на обработку и хранение документов, убыстряется их поиск. Но при этом возникает *проблема аутентификации* автора документа и самого документа, то есть установления подлинности автора и отсутствия изменений в полученном документе.

В обычной (бумажной) информатике эти проблемы решаются за счет того, что информация в документе и рукописная подпись автора жестко связаны с физическим носителем (бумагой). В электронных документах на машинных носителях такой связи нет.

Целью аутентификации электронных документов является их защита от возможных видов злоумышленных действий, к которым относятся:

– *активный перехват* – нарушитель, подключившийся к сети, перехватывает документы (файлы) и изменяет их;

– *маскарад* – абонент В посылает документ абоненту Б от имени абонента А;

– *ренегатство* – абонент А заявляет, что не посылал сообщения абоненту Б, хотя на самом деле послал;

– *подмена* – абонент Б изменяет или формирует новый документ и заявляет, что получил его от абонента А;

– *повтор* – абонент В повторяет ранее переданный документ, который абонент А посылал абоненту Б.

Эти виды злоумышленных действий могут нанести существенный ущерб банковским и коммерческим структурам, государственным предприятиям и организациям, а также частным лицам, применяющим в своей деятельности компьютерные информационные технологии.

При обработке документов в электронной форме совершенно непригодны традиционные способы установления подлинности по рукописной подписи и оттиску печати на бумажном документе.

Принципиально новым решением является *электронная цифровая подпись*.

Электронная цифровая подпись используется для аутентификации текстов, передаваемых по телекоммуникационным каналам. Как было показано выше, функционально она аналогична обычной рукописной подписи и обладает ее основными достоинствами:

– удостоверяет, что подписанный текст исходит от лица, поставившего подпись;

– не дает самому этому лицу возможности отказаться от обязательств, связанных с подписанным текстом;

– гарантирует целостность подписанного текста.

При этом *цифровая подпись* представляет собой относительно небольшое количество дополнительной цифровой информации, передаваемой вместе с подписываемым текстом.

Система электронной цифровой подписи включает две процедуры:

1) процедуру постановки подписи;

2) процедуру проверки подписи.

В процедуре постановки подписи используется закрытый ключ отправителя сообщения, в процедуре проверки подписи – открытый ключ отправителя. При формировании электронной цифровой подписи отправитель, прежде всего, вычисляет хеш-функцию $h(M)$ подписываемого текста M. Вычисленное значение хеш-функции $h(M)$ представляет собой один короткий блок информации m, характеризующий весь текст M в целом. Затем число m зашифровывается закрытым ключом отправителя. Получаемая при этом пара чисел представляет собой электронную цифровую подпись для данного текста M.

При проверке электронной цифровой подписи получатель сообщения снова вычисляет хеш-функцию $m = h(M)$ принятого по каналу текста M, после чего при помощи открытого ключа отправителя проверяет, соответствует ли полученная подпись вычисленному хеш-значению m.

Принципиальным моментом в системе электронной цифровой подписи является невозможность подделки электронной цифровой подписи пользователя без знания его закрытого ключа подписывания.

В качестве подписываемого документа может быть использован любой файл. *Подписанный файл* создается из неподписанного путем добавления в него одной или более электронных подписей.

Каждая подпись обычно содержит следующую информацию:

– дату подписи и срок окончания действия ключа данной подписи;

– информацию о лице, подписавшем файл (Ф.И.О., должность, краткое наименование фирмы);

– идентификатор подписавшего (имя открытого ключа);

– собственно цифровую подпись.

7.5 Контрольные вопросы

1) Дайте и поясните определения: идентификация, аутентификация и авторизация объекта.

2) Перечислите основные требования защиты при обмене сообщениями.

3) Поясните механизмы парольной аутентификации.

4) Поясните механизмы аутентификации на базе аппаратных средств.

5) Поясните механизмы биометрической идентификации и аутентификации.

6) Поясните способы взаимной проверки подлинности пользователей.

7) Поясните сущность процедуры «рукопожатия».

8) Поясните сущность протокола идентификации с нулевой передачей знаний.

9) Поясните цели аутентификации электронных документов.

10) Назовите преимущества электронной цифровой подписи при аутентификации данных.

8 СОВРЕМЕННЫЕ ПРОГРАММНЫЕ КОМПЛЕКСЫ ЭЛЕКТРОННОЙ ЦИФРОВОЙ ПОДПИСИ И СИСТЕМЫ ЗАЩИТЫ ИНФОРМАЦИИ ОТ НЕСАНКЦИОНИРОВАННОГО ДОСТУПА

8.1 Современные системы идентификации документов и аутентификации пользователей с использованием электронной цифровой подписи

8.1.1 Технология работы с программным комплексом Pretty Good Privacy (PGP)

В соответствии законодательством Российской Федерации при создании ключей электронных цифровых подписей для использования в информационной системе общего пользования должны применяться только сертифицированные средства электронной цифровой подписи.

При этом использование несертифицированных средств электронной цифровой подписи и созданных ими ключей в корпоративных информационных системах федеральных органов государственной власти, органов государственной власти субъектов РФ и органов местного самоуправления не допускается. Сертификация средств электронной цифровой подписи осуществляется в соответствии с законодательством Российской Федерации о сертификации продукции и услуг.

Вместе с тем для обеспечения конфиденциальности обмена сообщениями между частными лицами, не претендующими на правовую поддержку в разрешении возможных конфликтов, могут применяться любые средства.

Выбор PGP для такого применения, кроме надежности, обусловлен также доступностью для работы на персональных вычислительных средствах.

Программа Pretty Good Privacy, как следует из названия, должна обеспечивать «довольно хорошую секретность». Кроме того, на примере PGP проще освоить технологию защиты обмена сообщениями.

Программа PGP в виде исполняемого кода запрещена к экспорту из США, поэтому во всем остальном мире используются «международные релизы» этой системы. Сегодня в России известны десять ее версий.

Программа PGP – это гибридная криптосистема, объединяющая лучшие стороны симметричной и асимметричной криптографии.

В PGP исходное сообщение M вначале сжимается, образуя сжатое сообщение $M_{сж}$. Кроме того, что сжатие сокращает время передачи данных по компьютерной сети и экономит дисковое пространство ЭВМ, оно, что более важно, снижает избыточность данных, повышая криптографическую стойкость.

Затем в PGP создается одноразовый симметричный ключ, применяемый только для одной операции.

Этот *сеансовый ключ* $\mathcal{P}_{СК}$ представляет собой псевдослучайное число, сгенерированное от случайных движений мыши компьютера и нажатий клавиш на его клавиатуре. Сеансовый ключ работает на основе очень надежного, быстрого симметричного алгоритма, которым сжатое сообщение преобразовывается в шифртекст *E*.

На следующем шаге зашифровывается сеансовый ключ, но уже открытым ключом получателя $\mathcal{P}_{ОК\text{-}П}$. Этот зашифрованный открытым ключом сеансовый ключ $\mathcal{P}_{ЗШ\text{-}СК}$ прикрепляется к шифртексту *E* и передается вместе с ним получателю (рисунок 8.1).

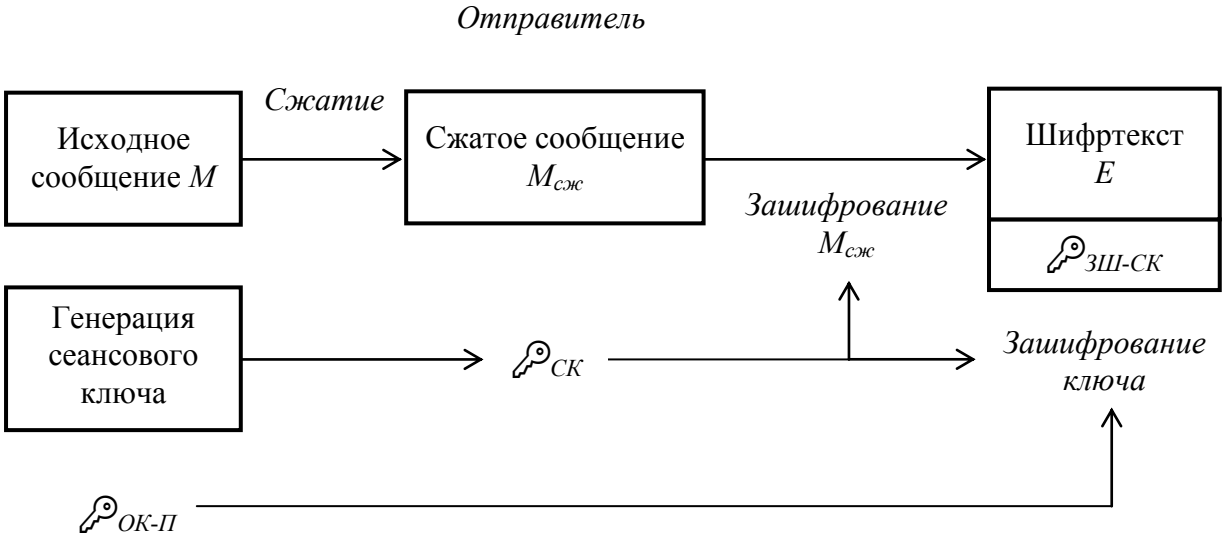


Рисунок 8.1 – Схема зашифрования сообщения в PGP

Расшифрование в PGP происходит в обратном порядке (рисунок 8.2).

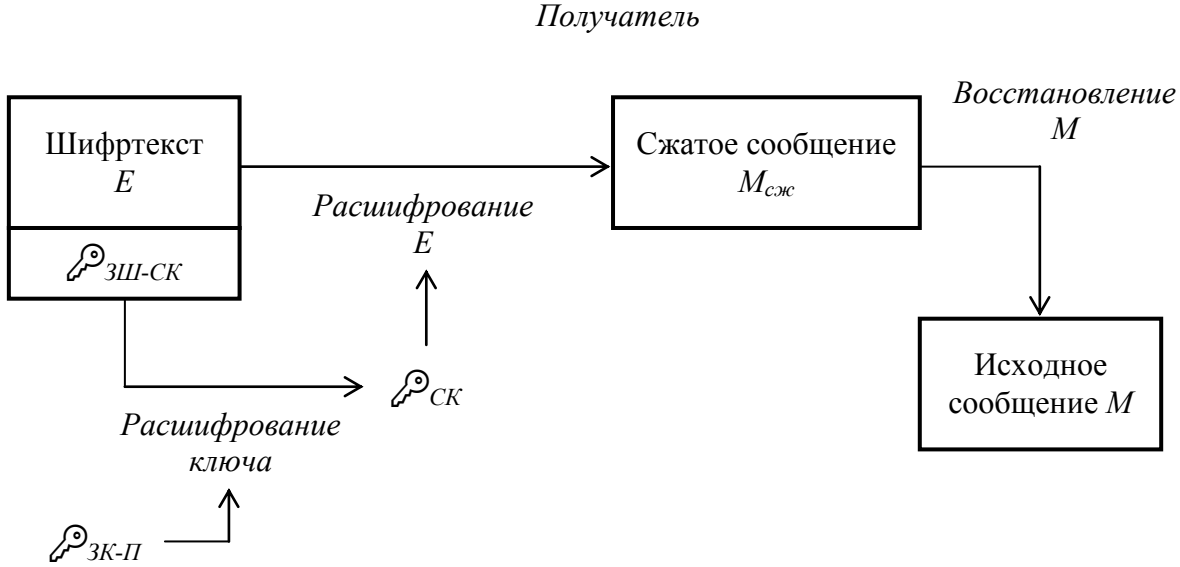


Рисунок 8.2 – Схема расшифрования сообщения в PGP

При расшифровании:

– закрытый ключ получателя $\mathcal{P}_{ЗК\text{-}П}$ используется для расшифровывания сеансового ключа;

– сеансовый ключ $\mathcal{P}_{СК}$ применяется для расшифрования сжатого сообщения $M_{сж}$;

– из сжатого сообщения восстанавливается исходное сообщение *M*.

Таким образом, комбинация этих двух криптографических методов в PGP объединяет удобство зашифрования открытым ключом со скоростью работы симметричного алгоритма. Симметричное зашифрование в тысячи раз быстрее асимметричного. Зашифрование открытым ключом, в свою очередь, предоставляет простое решение проблем управления ключами и передачи данных.

Ключи в PGP хранятся в зашифрованном виде в двух файлах на жестком диске:

– файл для открытых ключей (pabring.pkr);

– файл для закрытых ключей (secring.skr).

Эти файлы называются *связками* и хранятся чаще всего в папке PGP.

В связку открытых ключей будут добавляться открытые ключи будущих корреспондентов, а в связке закрытых ключей – всегда хранятся закрытые ключи владельца компьютера. Потеря (удаление) связки закрытых ключей приводит к невозможности расшифровывания сообщений, зашифрованных корреспондентами с помощью соответствующих открытых ключей. Следовательно, полезной практикой следует считать сохранение резервных копий этой связки.

Основные шаги при использовании программы PGP:

1) Установить программу на свой компьютер.

2) Создать закрытый и открытый ключ.

3) Распространить свой открытый ключ среди своих корреспондентов в обмен на их ключи.

4) Удостовериться в верности открытых ключей корреспондентов и занести их себе на связку открытых ключей.

5) Шифрование и удостоверение корреспонденции своей цифровой подписью.

После генерации пары ключей и обмена открытыми ключами можно начать шифрование и удостоверение своих сообщений и файлов цифровой подписью. Если используется почтовая программа, которая поддерживается программой PGP, то можно зашифровывать и расшифровывать всю корреспонденцию, находясь прямо в этой программе. Если же почтовая программа не поддерживается программой PGP, то можно шифровать корреспонденцию другими способами (через буфер обмена или шифрованием файлов целиком).

6) Расшифрование поступающих к сообщений и проверка подлинности отправителя. При получении зашифрованного сообщения можно расшифровать его, проверить подлинность отправителя этого сообщения и целостность самого сообщения. Если почтовая программа не поддерживается PGP, то можно сделать это через буфер обмена.

7) Уничтожение файлов. Когда необходимо полностью удалить какой-либо файл можно исполнить команду *wipe* (стереть). Таким образом, удаленный файл уже невозможно будет восстановить.

8.1.2. Установка, настройка, запуск и работа с программой PGP

Установка PGP

Программа Pretty Good Privacy, как следует из названия, должна обеспечивать «довольно хорошую секретность», впрочем, многое зависит как от правильной первоначальной установки, так и от разумного использования программы в дальнейшем.

Инсталляция программы PGP 8.0.2 мало чем отличается от аналогичных процедур по установке нового программного обеспечения.

Этапы таковы:

1) закрыть все открытые приложения и запустить инсталляционный файл PGP8.exe (рисунок 8.3);

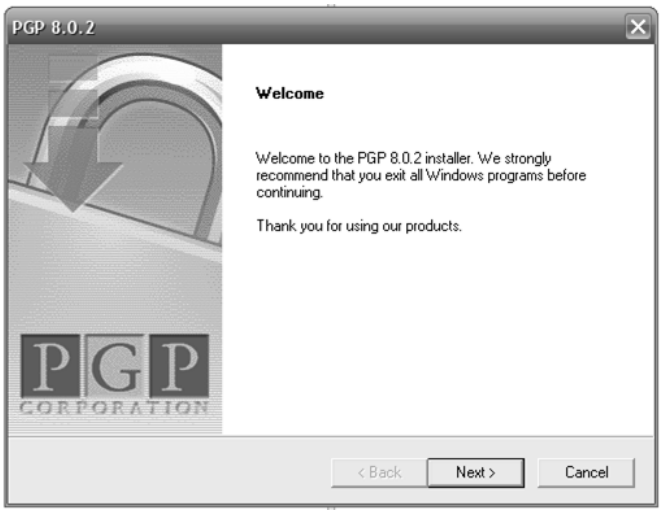


Рисунок 8.3 – Окно приглашения к установке PGP 8.0.2

2) согласиться со стандартным лицензионным соглашением, опубликованным компанией-продавцом (рисунок 8.4);

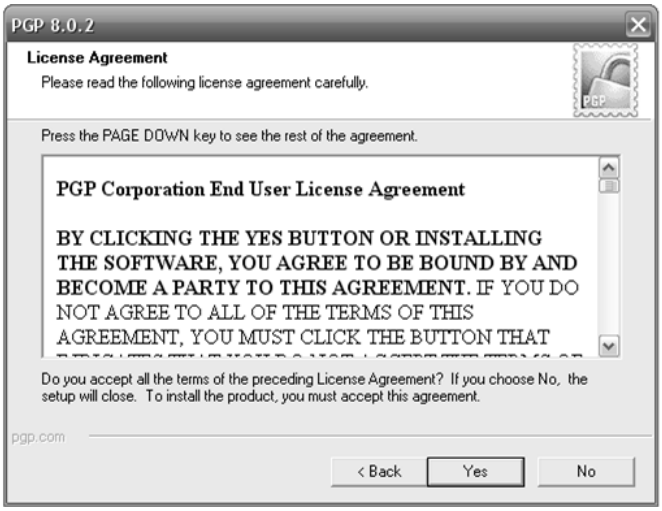


Рисунок 8.4 – Окно со стандартным лицензионным соглашением компании-продавца

3) при необходимости можно прочитать readme файл (рисунок 8.5);

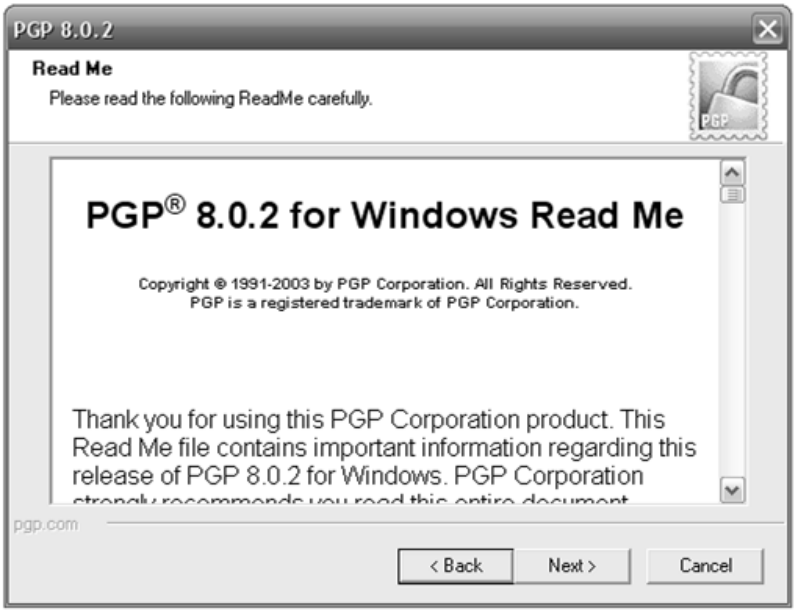


Рисунок 8.5 – Окно с файлом readme

4) запустить процесс установки программы на компьютере. Программа установки через некоторое время сделает запрос об уже имеющихся ключах или их отсутствии, если вы новый пользователь (рисунок 8.6). Если вы новый пользователь PGP, то никаких старых ключей нет;

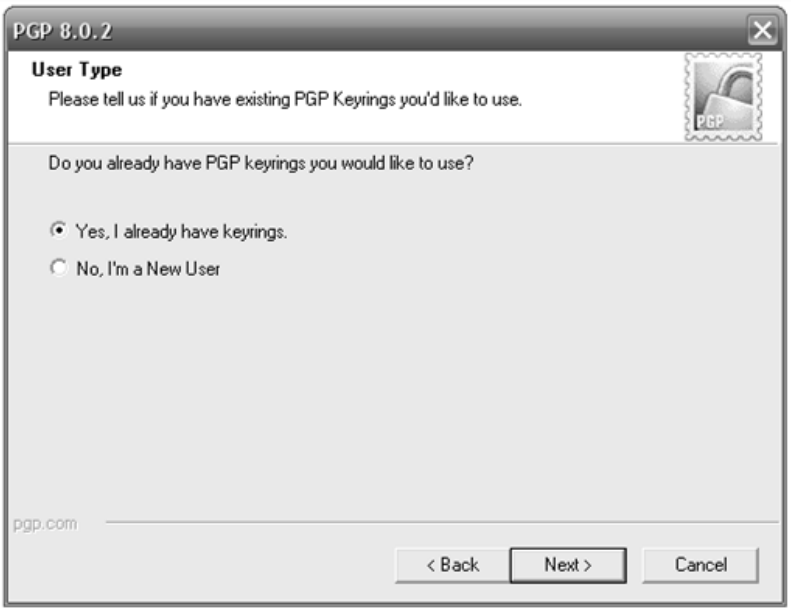


Рисунок 8.6 – Запрос на наличие ключей PGP

5) согласиться с предлагаемым местом размещения программы на диске компьютера или предложить свое собственное место (рисунок 8.7);

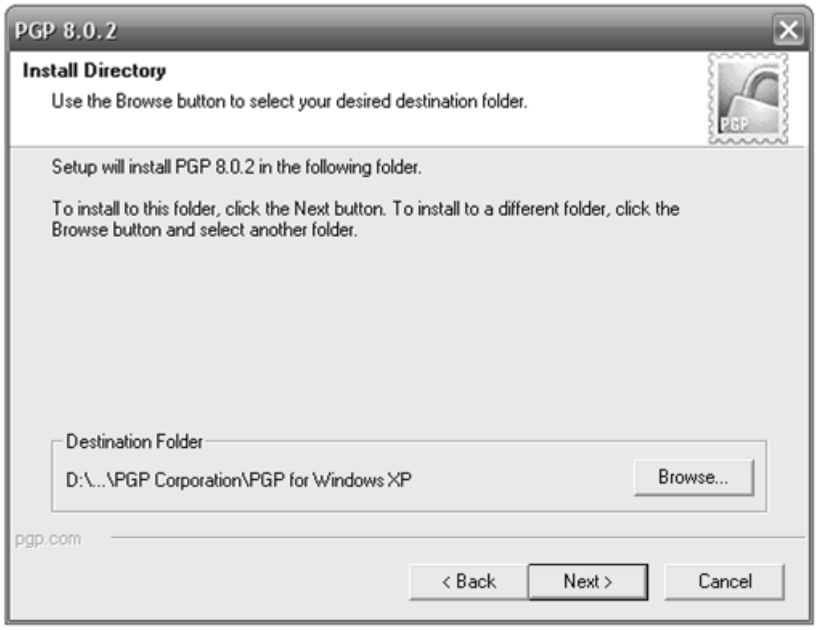


Рисунок 8.7 – Окно с предлагаемым местом размещения программы

5) выбрать необходимые для установки на персональном компьютере компоненты и подтвердить свой выбор (рисунок 8.8);

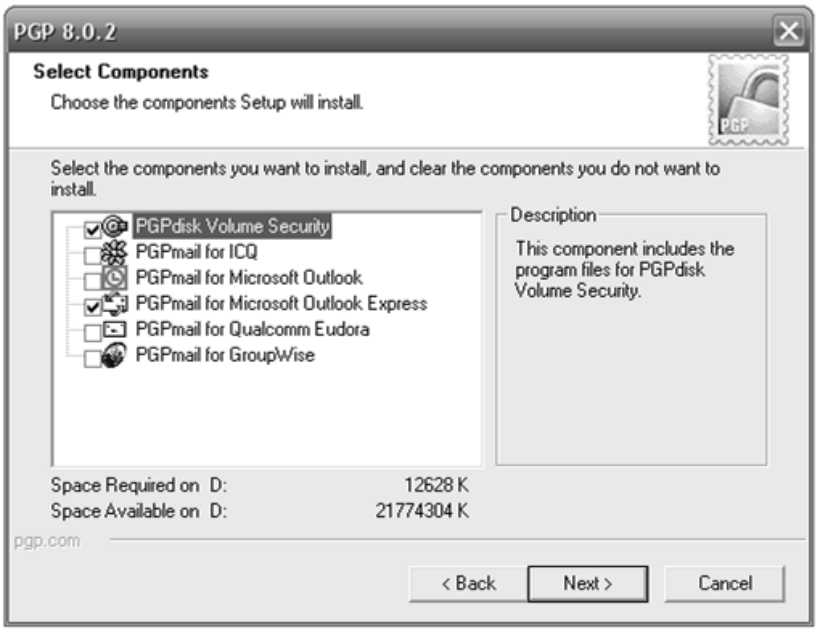


Рисунок 8.8 – Выбор необходимых компонентов PGP 8.0.2

6) далее, нажать на кнопку Finish, после чего последует перезагрузка компьютера (рисунок 8.9).

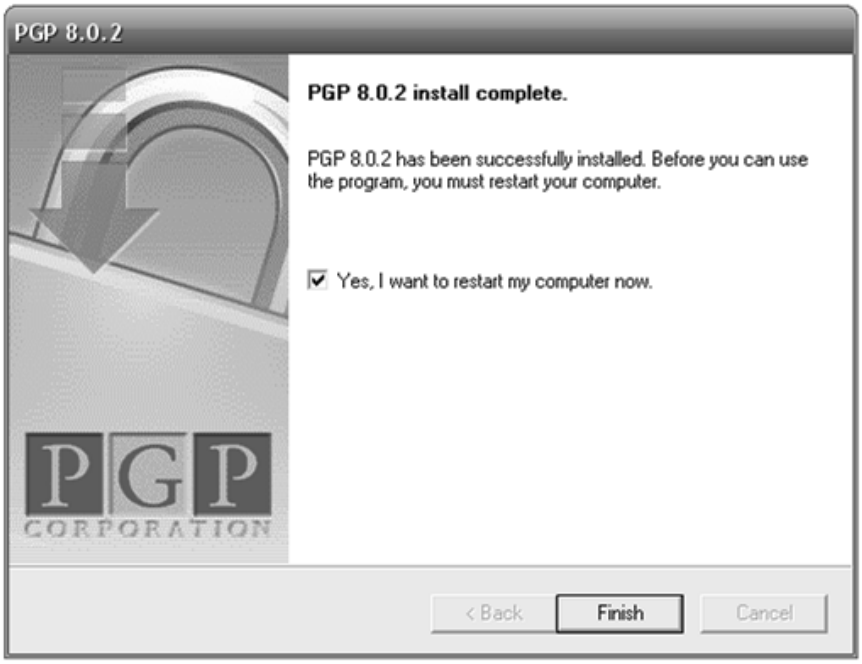


Рисунок 8.9 – Окно завершения установки PGP 8.0.2

Настройка программы

Для того, чтобы иметь возможность обмениваться конфиденциальными сообщениями со своими абонентами необходимо сгенерировать пару ключей: общественный (открытый) и личный (закрытый). Эта пара будет использована в дальнейшем и для создания электронной цифровой подписи. Следует помнить, что общественным ключом можно делиться с кем угодно, но личный ключ необходимо хранить в секрете от всех.

Порядок действий при генерации ключей будет рассмотрен в разделе 9 «Лабораторный практикум». Однако, следует знать некоторые теоретические вопросы процесса генерации ключей:

1) при генерации ключей необходимо указать собственное имя (Full name) и адрес электронной почты (E-mail address), именно эти данные будут ассоциированы программой с вашими ключами;

2) выбор типа ключа (Key Pair Type): ключ RSA архаичнее и медленнее Diffie-Hellman/DSS, но он поддерживается версиями ранее PGP 5.0;

3) выбор длины общественного ключа (Key Pair Size): по умолчанию, при использовании метода Diffie-Hellman/DSS, предлагается выбрать 2048-разрядный ключ;

4) можно установить даты действия ключевой пары, но не стоит без необходимости ограничивать «время жизни» ключей, так как после не будет возможности их изменить;

5) следует сформировать дополнительный ключ для расшифровки данных (Additional Decryption Key), так называемый «спасительный ключ» (Recovery Keys), он необходимым для восстановления зашифрованных данных в случае потери личного ключа;

6) необходимо сгенерировать корпоративный ключ (Corporate Signing Key), которым будет пользоваться администратор сервера сети для автоматической подписи и установления доверия по отношению к вашим сообщениям;

7) для администратора формируется также ключ для аннулирования вашего ключа (Designated Revocation Key), в случае обнаружения подмены;

8) необходимо сочинить ключевую фразу (passphrase); существует единственное ограничение – во фразе должно присутствовать не менее восьми символов. Ключевую фразу следует хорошо запомнить;

9) в процессе генерации ключей, которая может занять несколько минут, рекомендуется беспорядочно нажимать на клавиши клавиатуры компьютера; тем самым облегчается процесс генерации случайных данных;

10) когда ключи сгенерированы, пользователю программы предлагается выйти в Интернет и отправить общественный ключ на сервер, где уже сложены ключи других работников компании (root server); к этой процедуре можно вернуться позже или предпочесть индивидуальную рассылку.

После того как ключи созданы, можно приступать к работе. Однако, программа PGP требует ответственного подхода, поэтому окно «PGPkeys» удастся закрыть не сразу (рисунок 8.10).

Рисунок 8.10 – Предупреждение о необходимости сохранения резервной копии связки ключей

На случай если что-либо произойдет с жестким диском компьютера, нужна резервная копия связки ключей. Потеря связки ключей немедленно влечет за собой недоступность предназначаемой для вас конфиденциальной информации.

Поэтому необходимо сохранить на машинном носителе связку с общественными и личным ключами (файлы pubring.pkr и secring.scr, соответственно), как показано на рисунке 8.11.

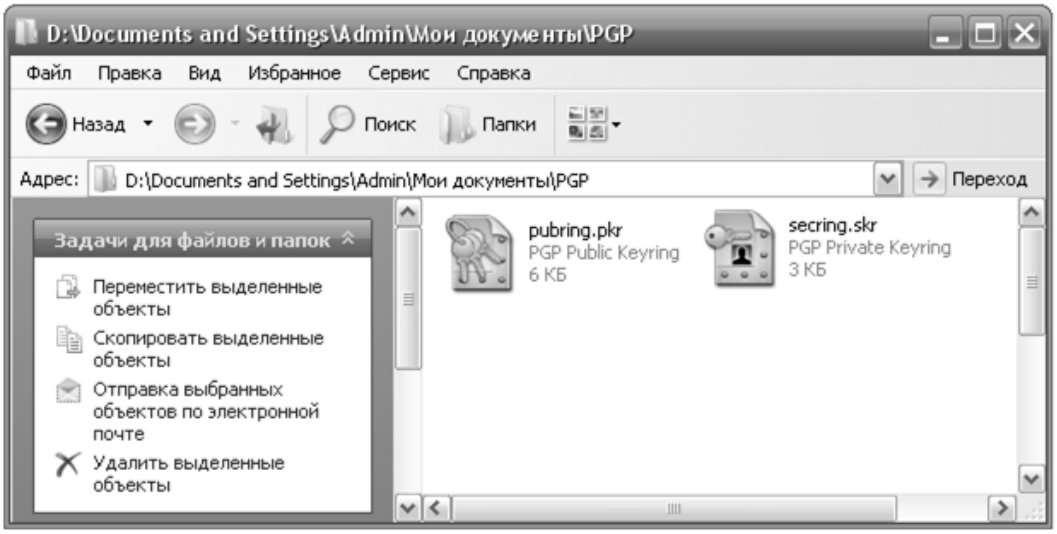


Рисунок 8.11 – Сохранение резервной копии связки ключей

Совокупный размер связки ключей составляет примерно 5 Кбайт. В пропорции 3:1 в пользу ключа общественного. Поэтому хранение этих файлов на машинном носителе не вызовет каких-либо проблем. Если кража некой зашифрованной информации ни в коем случае не допустима, то необходимо удалить файл secring.skr с жесткого диска и хранить его только на машинном носителе, например, в сейфе. Для открытия корреспонденции, в этом случае, потребуется вставить «ключевой диск». Можно поступить и иначе: произвольным образом переименовать файл secring.skr и хранить его вдали от папки PGP.

Для работы с PGP необходимо распространить свой общественный ключ среди всех потенциальных получателей вашей конфиденциальной корреспонденции. Способы распространения общественных ключей (экспорт и импорт ключей) будут рассмотрены в разделе 6 «Лабораторный практикум».

Следует знать, что при распространении общественных ключей можно использовать некий сервер и в подписи своего письма необходимо указывать адрес этого сервера. Если переписка происходит внутри одной большой организации, именно так, как правило, и поступают.

Адрес сервера может выглядеть, например, так:

http://swissnet.ai.mit.edu:11371/pks/lookup?op=get&search=0x5DC10B44

Последняя комбинация <0x5DC10B44> - это идентификатор вашего ключа. Он записан в разделе Key Properties (в окне PGPkeys после правого клика мышью по строке ключа выбрать Key Properties), в окошке ID в разделе General.

Запуск программы

Существует, по крайней мере, четыре способа запустить программу PGP на компьютере:

1) С помощью меню Пуск (Start).

2) Из области индикаторов панели задач (замочек в правом нижнем углу экрана).

3) Непосредственно из программы, используемой для передачи данных (например, Outlook).

4) Через строку PGP после правого клика на иконке файла или папки.

Первый способ предоставляет доступ к основным ресурсам PGP, плюс возможность обращения к вспомогательной документации и к подпрограмме удаления криптографического пакета с компьютера.

Второй способ чаще всего используется при необходимости криптования данных, помещенных в буфере обмена (clipboard).

Третий способ предпочитает большинство пользователей, то есть работу с шифрованием в привычном окне почтовой программы.

Последний из перечисленных способов является самым коротким и быстрым.

Рассмотрим эти четыре способа подробнее.

Первый способ запуска программы PGP: Пуск (Start) > Программы (Programs) > PGP (рисунок 8.12).

Рисунок 8.12 – Запуск программы PGP первым способом

Предназначение рабочей документации (Documentation) и подпрограммы удаления (Uninstall) очевидно. Разберемся со служебными полномочиями других подпрограмм.

Подпрограмма PGPadmin позволяет построить криптографическую систему для целой организации. Добавляет в конфигурацию ключевой пары пользователя понятие дополнительного корпоративного ключа (Additional Decryption Key – ADK).

Подпрограмма PGPdisk (рисунок 8.13) фактически создает защищенную область на жестком диске компьютера, доступ в которую открыт только обладателю пароля (passphrase).

Рисунок 8.13 – Окно подпрограммы PGPdisk

Даже после восстановления информации на диске (потерянной после заражения вирусом или форматирования диска), данные из PGPdisk останутся закрытыми «на ключевую фразу».

Можно создать новый PGPdisk (New) необходимого объема и присвоить ему метку. После чего созданный логический диск открыт для размещения данных (Mount). Рекомендуется, попользовавшись, закрывать секретную область на диске (Unmount). Для того, чтобы снова открыть PGPdisk потребуется ввести ключевую фразу. В разделе свойств диска (Prefs) можно произвести дополнительную настройку:

– установить время, по истечении которого доступ к секретной области будет закрыт автоматически (по умолчанию – 15 минут);

– отменить установление защиты при выключении компьютера;

– выбрать «горячую клавишу» для быстрого запирания на ключ.

Подпрограмма PGPkeys позволяет получить доступ к таблице личных и общественных ключей, а также открытых ключей корреспондентов (рисунок 8.14).

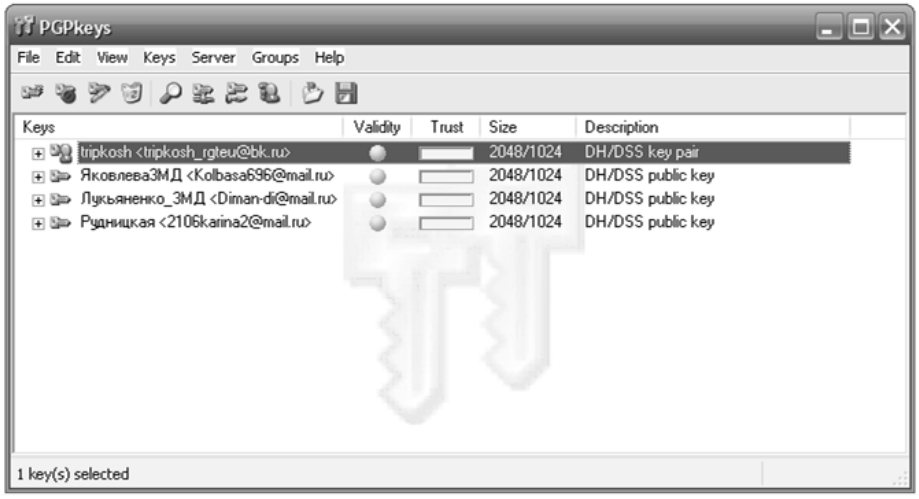


Рисунок 8.14 – Окно подпрограммы PGPkeys

Полномасштабное описание этой таблицы на языке оригинала можно найти в описании разработчика (PGP Manual или Help). Здесь же ограничимся лишь их кратким комментарием. Основные кнопки:

– инициация создания новой ключевой пары;

– отмена данного ключа;

– подпись избранного ключа;

– уничтожение ключа, соответствующего выделенной строке;

– поиск нужного ключа в списке;

– отправка открытого ключа на сервер;

– получение ключа с сервера;

– посмотреть основные параметры ключа;

– вытащить ключи из некоего файла на компьютере;

– вставить избранные ключи в файл.

Верхняя строка меню в таблице PGPkeys предоставляет дополнительные возможности. Например, можно добавить новые пункты в описания ключей (View):

– идентификатор ключа (Key ID);

– уровень доверия (Trust);

– дата создания (Creation Date);

– дата завершения функционирования ключа (Expiration Date);

– ассоциированность с дополнительным ключом (ADK).

Из верхней строки меню PGPkeys есть доступ к таблице основных свойств PGP (Edit > Preferences).

Подпрограмма PGPmail активизирует таблицу инструментов PGP (рисунок 8.15).

Рисунок 8.15 – Окно подпрограммы PGPmail

Основные инструменты подпрограммы:

– переход к таблице PGPkeys;

– выбор файла для шифрования;

– выбор файла документа «на подпись»;

– зашифровать подписанный файл;

– расшифровать данные с подписью;

– удалить так, чтобы невозможно было восстановить;

– создание невосстановимой области.

Подпрограмму PGPtray активизирует значок . Запуск этой подпрограммы, предоставляющей легкий путь криптования информации, содержащейся в буфере обмена (clipboard), будет рассмотрен подробнее при описании второго способа обращения к PGP.

Второй способ запуска программы PGP: в правом нижнем углу экрана имеется замочек . Любой (правый или левый) клик мышью на этой иконке приводит к появлению нового окна с длинным списком возможностей программы PGP (рисунок 8.16).

Рисунок 8.16 – Второй способ запуска программы PGP

Особый интерес в этом списке вызывает строка настройки предпочтений – Options (рисунок 8.17).

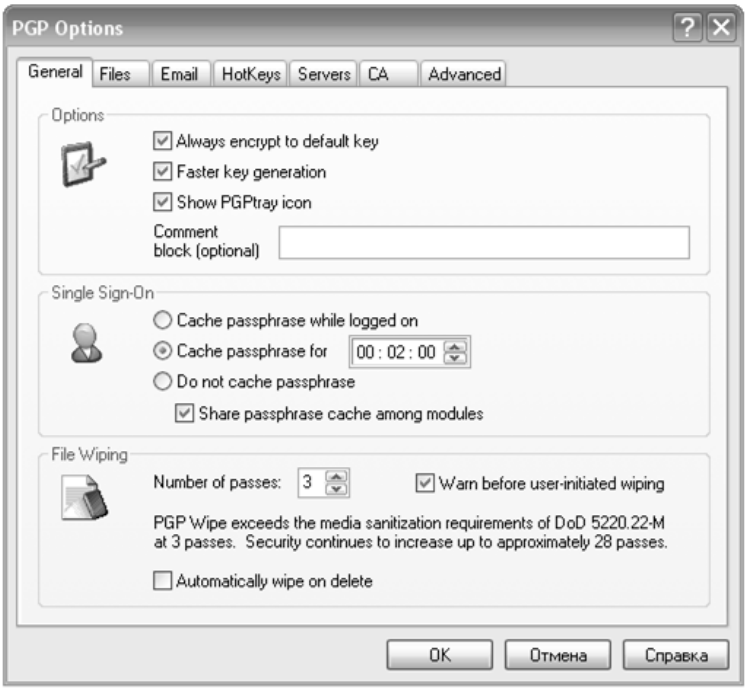


Рисунок 8.17 – Окно настройки предпочтений PGP

Рассмотрим основные варианты (раздел General):

– Always encrypt to default key (всегда шифровать ключом, определенным как основной) – если эта опция включена, все данные, зашифрованные открытым ключом получателя, будут шифроваться также и вашим основным ключом.

– Faster key generation (ускоренное создание ключа) – позволяет несколько сэкономить время, хотя теоретически и снижает надежность ключа.

– Show PGPtray icon (показывать иконку PGPtray) – если эта опция включена, то в правой нижней части экрана появляется замочек  .

– Cache passphrases for (кэшировать пароль в течение указанного времени) – полезно при чтении огромного вороха корреспонденции установить время кэширования побольше, тогда не придется всякий раз перенабирать ключевую фразу.

В разделе Files можно менять место расположения связки с ключами. Хотя без особого труда то же самое делается и вручную.

На вкладке установления предпочтений при работе с электронной почтой Email можно изменять параметры работы с почтой для программ, поддерживаемых с помощью встраиваемых модулей. Если отметить галочками все строки, это сократит в дальнейшем перечень процедур при работе с шифрованными почтовыми сообщениями ровно на четыре щелчка мышью.

Раздел Servers, как правило, содержит адрес корпоративного сервера, на котором хранятся открытые ключи всех пользователей компании. Однако, в новой версии PGP любой пользователь может произвольно дополнять список подобных серверов.

Начинающим пользователям не стоит изменять настройки в разделе Advanced. В нем алгоритмы шифрования и модели доверия подобраны оптимальным образом.

Все остальные команды в PGPtray предназначены для работы с данными, размещенными в буфере обмена (clipboard). Поместить данные в буфер можно, выделив их маркером мыши и одновременно нажав на клавиатуре клавиши Ctrl и C, или выбрав в меню данной программы Edit (Правка) и Copy (Копировать).

Вот список этих команд:

– Current Window – использовать выделенный текст в текущем окне;

– Empty Clipboard – очистить буфер обмена от старого содержимого;

– Edit Clipboard – простейший текстовый редактор (один клик и текст в буфере);

– Decrypt & Verify Clipboard – расшифровать и идентифицировать информацию из буфера;

– Encrypt & Sign Clipboard – зашифровать и подписать данные, находящиеся в буфере;

– Sign Clipboard – подписать сообщение в буфере;

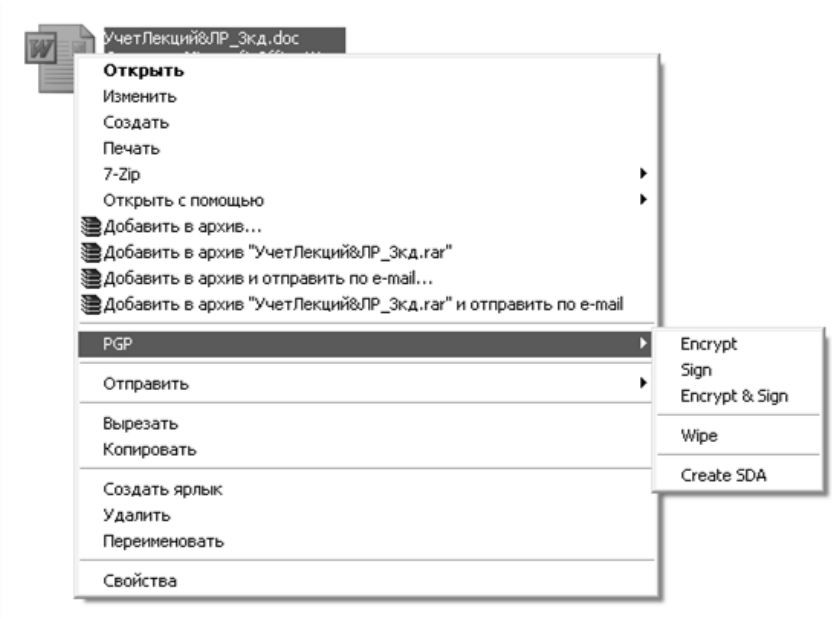
– Encrypt Clipboard – зашифровать сообщение в буфере.

Третий способ запуска программы PGP: из рабочего окна избранной почтовой программы (например, Outlook).

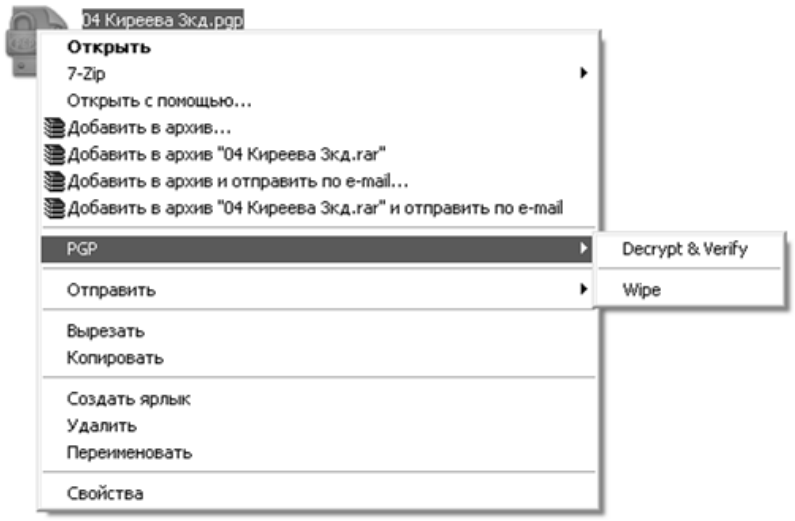
В строке меню почтовой программы появляется PGP с переходами на PGPkeys и Preferences.

Дополнительных кнопок при создании нового сообщения всего три: Encrypt, Sign и PGPkeys. Назначение их можно уже не комментировать.

Четвертый способ запуска программы PGP: после правого клика мышью на любом файле или папке, в нижней строке перечня возможностей возникает сообщение PGP (рисунок 8.18).



а)



б)

Рисунок 8.18 – Четвертый способ запуска программы PGP

На рисунке 8.18а:
– Encrypt – зашифровать данные;
– Sign – подписать шифровку;
– Encrypt & Sign – проделать то же самое, но в одно действие;
– Wipe – удалить информацию так, чтобы ее уже никто не восстановил.

На рисунке 8.18б:

– Decrypt & Verify – расшифровать и подтвердить подлинность.

Работа с программой

PGP – довольно простая в употреблении программа, хотя и использует чрезвычайно сложные математические алгоритмы шифрования данных.

Основные действия при работе программы рассмотрим в разделе 9 «Лабораторный практикум».

8.2 Системы защиты информации от несанкционированного доступа

8.2.1 Система криптографической защиты информации от несанкционированного доступа КРИПТОН-ВЕТО

Система КРИПТОН-ВЕТО предназначена для защиты персонального компьютера с процессором не ниже 386, работающего под управлением MS DOS 5.0 и выше, Windows 95 и выше.

Персональный компьютер при этом может использоваться в качестве:

– абонентского пункта;

– центра коммутации пакетов;

– центра выработки ключей.

Система КРИПТОН-ВЕТО ограничивает круг лиц и их права доступа к информации на персональном компьютере. Ее реализация основана на технологиях «прозрачного» шифрования логических дисков по алгоритму ГОСТ 28147-89 и электронной цифровой подписи по ГОСТ Р34.10/11-94.

В состав основных функций системы КРИПТОН-ВЕТО включены следующие:

– обеспечение секретности информации в случае кражи жесткого диска или компьютера целиком;

– обеспечение защиты от несанкционированного включения компьютера;

– разграничение полномочий пользователей по доступу к ресурсам персонального компьютера;

– проверка целостности используемых программных средств компьютерной системы в момент включения системы;

– проверка целостности программы в момент ее запуска на выполнение;

– запрещение запуска на выполнение посторонних программ;

– ведение системного журнала, регистрирующего события, возникающие в системе;

– обеспечение «прозрачного» шифрования информации при обращении к защищенному диску;

– обнаружение искажений, вызванных вирусами, ошибками пользователей, техническими сбоями или действиями злоумышленника.

Основным аппаратным элементом системы являются серийно выпускаемые аттестованные ФСТЭК России платы серии КРИПТОН, с помощью которых проверяется целостность системы и выполняется шифрование по ГОСТ 28147-89.

Система предполагает наличие администратора безопасности, который определяет взаимодействие между управляемыми ресурсами:

– пользователями;

– программами;

– логическими дисками;

– файлами (дискреционный и мандатный доступ);

– принтером;

– дисководами.

Система обеспечивает защиту следующим образом.

Жесткий диск разбивается на логические диски:

– первый логический диск (С:) отводится для размещения системных программ и данных;

– последний логический диск – для размещения системы защиты информации от несанкционированного доступа (СЗИ НСД) и доступен только администратору;

– остальные логические диски предназначены для хранения информации и программ пользователей. Эти диски можно разделить по пользователям и/или по уровню секретности размещаемой на них информации.

Можно выделить отдельные диски с информацией различного уровня секретности (доступ к таким дискам осуществляется с помощью специальной программы, проверяющей допуск пользователя к документам-файлам).

Сначала администратор устанавливает уровень секретности диска, а затем определяет круг лиц, имеющих доступ к этому диску. По форме хранения информации диски подразделяются на открытые и шифруемые.

По уровню доступа – на доступные для чтения и записи, доступные только для чтения, недоступные (заблокированные).

Недоступный диск делается невидимым в операционной системе и, следовательно, не провоцирует пользователя на несанкционированный доступ к нему. Доступный только для чтения диск можно использовать для защиты не только от целенаправленного, но также от непреднамеренного (случайного) искажения (удаления) информации.

Открытый диск ничем не отличается от обычного логического диска. Очевидно, что системный диск должен быть открыт.

Для шифруемых дисков используется шифрование информации в прозрачном режиме. При записи информации на диск она автоматически шифруется, при чтении с диска автоматически расшифровывается. Каждый шифруемый диск имеет для этого соответствующий ключ. Последнее делает бесполезными попытки улучшения своих полномочий пользователями, допущенными на персональный компьютер, поскольку они не имеют ключей доступа к закрытым для них дискам.

Структура системы КРИПТОН-ВЕТО приведена на рисунке 8.19.

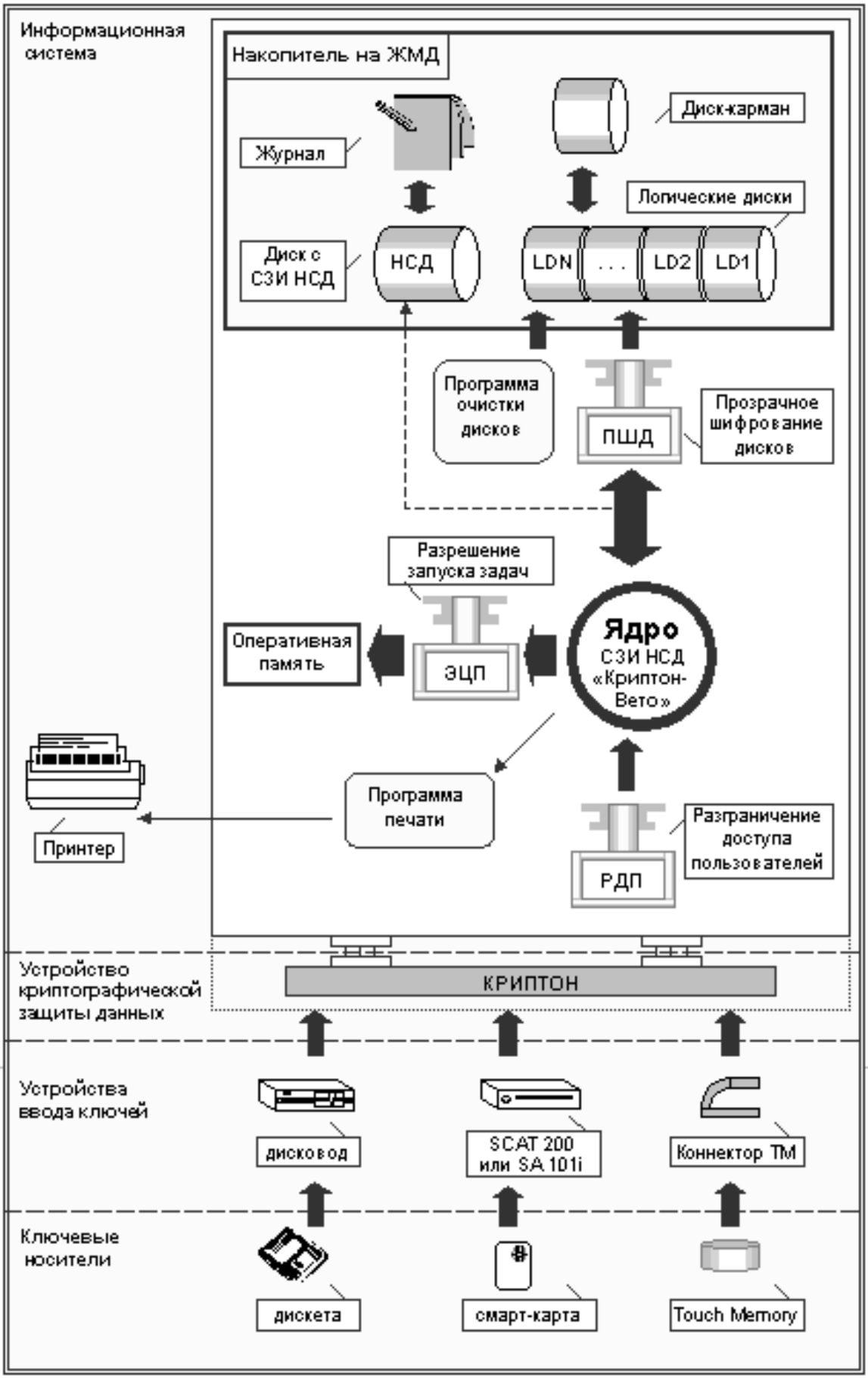


Рисунок 8.19 – Структура системы КРИПТОН-ВЕТО

Наличие шифрования обеспечивает секретность информации даже в случае кражи жесткого диска.

Для допуска к работе на персональном компьютере администратором формируется список пользователей, в котором:

– указывается идентификатор и пароль пользователя;

– определяется уровень допуска к секретной информации;

– определяются права доступа к логическим дискам.

В дальнейшем только администратор может изменить список пользователей и их полномочия.

Для исключения возможности установки на персональный компьютер посторонних программ с целью взлома защиты администратор определяет перечень программ, разрешенных к запуску на данном компьютере. Разрешенные программы подписываются администратором электронной цифровой подписью.

Только эти программы могут быть запущены в системе.

Использование электронной цифровой подписи одновременно с наличием разрешения позволяет отслеживать целостность запускаемых программ. Последнее исключает возможность запуска измененной программы, в том числе и произошедшего в результате непредвиденного воздействия «вируса».

Для входа в компьютер используются ключи, записанные на ключевой дискете, смарт-карте или на устройстве Touch-Memory. Ключи изготавливаются администратором системы и раздаются пользователям под расписку.

Для исключения загрузки компьютера в обход СЗИ НСД загрузка осуществляется только с жесткого диска. При включении персонального компьютера (до загрузки операционной системы) с жесткого диска аппаратно проверяется целостность ядра системы безопасности КРИПТОН-ВЕТО, системных областей жесткого диска, таблицы полномочий пользователей. Затем управление передается проверенному ядру системы безопасности, которая проверяет целостность операционной системы.

Расшифрование полномочий пользователя, ключей зашифрованных дисков и дальнейшая загрузка операционной системы производятся лишь после заключения о ее целостности.

В процессе работы в компьютере загружены ключи только тех дисков, к которым пользователю разрешен доступ.

Для протоколирования процесса работы ведется журнал. В нем регистрируются следующие события:

– установка системы КРИПТОН-ВЕТО;

– вход пользователя в систему (имя, дата, время);

– попытка доступа к запрещенному диску (дата, время, диск);

– зашифрование диска;

– расшифрование диска;

– перешифрование диска;

– добавление нового пользователя;

– смена полномочий пользователя;

– удаление пользователя из списка;

– сброс причины останова системы;

– попытка запуска запрещенной задачи;

– нарушение целостности разрешенной задачи и т.д.

Журнал может просматриваться только администратором.

Для проверки работоспособности системы используются программы тестирования.

При необходимости пользователь может закрыть информацию на своем диске и от администратора, зашифровав последнюю средствами абонентского шифрования.

8.2.2 Комплекс КРИПТОН-ЗАМОК для ограничения доступа к компьютеру

Комплекс КРИПТОН-ЗАМОК предназначен для построения аппаратно-программных средств ограничения доступа к компьютеру с использованием устройств комплексной защиты данных (УКЗД) серии КРИПТОН.

Комплекс позволяет организовать на базе персонального компьютера рабочее место с ограничением круга лиц, имеющих доступ к содержащейся в нем информации.

Для работы комплекса КРИПТОН-ЗАМОК необходим персональный компьютер IBM PC с процессором не ниже i386 и операционной системой – MS DOS, Windows 95/98/NT, UNIX и другими, для которых имеется соответствующий драйвер, позволяющий под управлением MS DOS понимать формат установленной на компьютере файловой системы.

Комплекс служит для защиты компьютеров с жесткими дисками, с файловыми системами в форматах FAT12, FAT16, FAT32, NTFS, UNIX и т.д.

Работа с дисками с файловыми системами FAT12, FAT16 и FAT32 обеспечивается средствами комплекса без дополнительных драйверов. Работа с дисками с нестандартными файловыми системами NTFS, HTFS, UNIX и т.д., может производиться только при наличии на компьютере соответствующих драйверов.

Комплекс КРИПТОН-ЗАМОК выпускается в двух исполнениях:

– для жестких дисков объемом менее 8 Гбайт;

– для жестких дисков объемом более 8 Гбайт.

В базовый состав аппаратно-программных средств ограничения доступа к компьютеру входят:

1) УКЗД серии КРИПТОН, поддерживающие режим работы комплекса ЗАМОК;

2) комплект драйверов и библиотек УКЗД;

3) комплекс ЗАМОК, включающий:

- микросхему с программным обеспечением комплекса, устанавливаемую в УКЗД серии КРИПТОН;
- инсталляционный дистрибутивный носитель с программным обеспечением комплекса.

Установленный в персональный компьютер комплекс ограничения доступа КРИПТОН-ЗАМОК выполняет следующие функции:

– ограничивает доступ пользователей к компьютеру путем их идентификации и аутентификации;

– разделяет доступ пользователей к ресурсам компьютера в соответствии с их полномочиями;

– контролирует целостность ядра комплекса, программ операционной среды, прикладных программ и областей памяти в момент включения компьютера до загрузки его операционной системы;

– регистрирует события в защищенном электронном журнале;

– передает управление и параметры пользователя программному обеспечению (RUN-файлам), указанному администратором (например, программному обеспечению защиты от несанкционированного доступа).

В соответствии с выполняемыми функциями комплекс КРИПТОН-ЗАМОК содержит следующие основные подсистемы:

1) Подсистему управления доступом, состоящую из устройства КРИПТОН и программы обслуживания CRLOCK.EXE.

2) Подсистему регистрации и учета, включающую два журнала:

– аппаратный – на устройстве КРИПТОН, фиксирующий попытки входа в компьютер до запуска его операционной системы;

– полный – на жестком диске, в котором после удачного входа в комплекс отображаются все события, в том числе и содержимое аппаратного журнала), управление которыми осуществляется программой обслуживания комплекса CRLOCK.EXE.

3) Подсистему обеспечения целостности, состоящую из устройства КРИПТОН и программы CHECKOS.EXE, проверяющей целостность главной операционной системы при работе комплекса.

При этом комплекс КРИПТОН-ЗАМОК обеспечивает выполнение следующих задач:

– в компьютер может войти только санкционированный пользователь;

– загружается достоверное ядро комплекса;

– загружается достоверная операционная система;

– проверяется целостность прикладного программного обеспечения, указанного администратором;

– производится запуск программ, указанных администратором.

Рассмотрим штатную работу комплекса КРИПТОН-ЗАМОК.

В начале работы с комплексом устройство КРИПТОН при инициализации его ключами с ключевого носителя (дискеты, смарт-карты или Touch Memory) загружает три файла:

– UZ.DB3 (он один для всех пользователей данного компьютера);

– GK.DB3 (он уникален для каждого и может быть зашифрован на пароле пользователя) и

– файл-паспорт пользователя INIT.NSD.

Первые два файла обеспечивают выполнение устройством КРИПТОН криптографических процедур в соответствии с ГОСТ 28147-89 и формиру-

ются при помощи любой из программ генерации криптографических ключей, выпускаемых фирмой АНКАД для средств серии КРИПТОН (например, Crypton Soft, Crypton Tools или Cr Mng). Файл INIT.NSD уникален для каждого пользователя и используется при входе в комплекс для загрузки и проверки его ядра, поиска пользователя в файле полномочий, его аутентификации и расшифровки его записи. Файл INIT.NSD формируется на ключевом носителе пользователя: для администратора – автоматически программой INSTAL.EXE при установке комплекса на компьютер, а для всех остальных пользователей - администратором при помощи программ CRLOCK.EXE.

Алгоритм работы комплекса КРИПТОН-ЗАМОК включает следующие шаги:

1) УКЗД КРИПТОН инициализируется файлами UZ.DB3 и GK.DB3.

2) КРИПТОН загружает файл INIT.NSD и проверяет его целостность по имитовставке. В случае нарушения целостности этого файла или при его отсутствии дальнейшая загрузка компьютера не производится.

3) КРИПТОН производит поиск имени вошедшего пользователя в списке пользователей. В случае отсутствия пользователя в списке дальнейшая загрузка компьютера не производится.

4) КРИПТОН производит аутентификацию пользователя – проверяет имитовставку его ключа. В случае несовпадения имитовставки пользователь считается несанкционированным и дальнейшая загрузка компьютера не производится.

5) КРИПТОН производит загрузку операционной системы комплекса ЗАМОК с Flash-диска. При загрузке автоматически стартует программа проверки целостности защищаемой операционной системы – CHECKOS.EXE.

6) CHECKOS.EXE получает параметры вошедшего пользователя от устройства КРИПТОН и:

– разблокирует клавиатуру;
– проверяет целостность файл-списка;
– проверяет целостность системных областей и файлов защищаемой операционной системы;
– при наличии RUN-файлов проверяет их целостность и запускает на выполнение;
– по запросу пользователя меняет пароль ключей на его носителе; по запросу администратора запускает программу обслуживания комплекса CRLOCK.EXE;
– при успешном завершении всех проверок CHECKOS.EXE запускает защищаемой операционной системы.

После загрузки защищаемой операционной системы компьютера комплекс ограничения доступа к компьютеру прекращает свою деятельность и не вмешивается в дальнейшую работу компьютера (до следующей загрузки). Далее устройство КРИПТОН может использоваться как обычный шифратор.

Механизм RUN-файлов позволяет в процессе работы комплекса КРИПТОН-ЗАМОК запускать любые программы с предварительной проверкой их целостности. В частности, механизм RUN-файлов может быть исполь-

зован при проверке файлов, находящихся на логических дисках с нестандартными файловыми системами (NTFS, HPFS, UNIX и т.д.).

Другой вариант использования – запуск из под комплекса КРИПТОН-ЗАМОК любого другого программного обеспечения:

– системы защиты от несанкционированного доступа;

– криптомаршрутизатора;

– операционной системы и т.д.

На этой основе может быть построена система защиты персонального компьютера с требуемыми свойствами.

8.2.3 Система защиты данных Crypton Sigma

Система Crypton Sigma – это программный комплекс, предназначенный для защиты данных на персональном компьютере.

Будучи установленной на компьютере, система Crypton Sigma хранит конфиденциальные данные в зашифрованном виде, не допуская несанкционированный доступ и утечку данных.

Для шифрования данных в системе Crypton Sigma используется алгоритм шифрования ГОСТ 28147-89.

Система защиты конфиденциальных данных Crypton Sigma ориентирована на широкий круг пользователей компьютеров-бизнесменов, менеджеров, бухгалтеров, адвокатов и др., то есть всех тех, кто нуждается в защите профессиональной и личной информации. Система Crypton Sigma легко устанавливается, проста и надежна в использовании, а также полностью «прозрачна» для всех программ и системных утилит операционной системы.

При установке системы Crypton Sigma на компьютере создаются новые логические диски. При записи на эти диски информация автоматически шифруется, а при считывании – расшифровывается.

Этот метод прозрачного шифрования позволяет полностью снять с пользователя заботу о защите данных. Работа с защищенными дисками незаметна для пользователя и равносильна встраиванию процедур зашифрования/расшифрования в запускаемые приложения.

Защищенные системой диски на вид ничем не отличаются от обычных и могут использоваться в локальной или глобальной сети.

Поддерживаемые файловые системы - FAT16, FAT32 и NTFS.

Система Crypton Sigma может работать как с УКЗД КРИПТОН, так и с его программным эмулятором. Криптографические ключи для защиты диска хранятся на съемном носителе (дискете), а при использовании УКЗД КРИПТОН возможно хранение ключевой информации на устройстве Touch Memory или смарт-карте. Кроме того, можно использовать устройство eToken (ключевой носитель для USB-порта). Применение УКЗД КРИПТОН не позволит злоумышленнику перехватить ключи пользователя с помощью внедренных программ.

Для работы системы Crypton Sigma требуется следующая минимальная конфигурация:

1) Компьютер:

– IBM PC/AT, PS/2 (с процессором i80486 или выше) или полностью совместимый;

– минимум 8 Мбайт оперативной памяти;

– минимум 3 Мбайт свободного дискового пространства для установки и запуска системы Crypton Sigma.

2) Программно-аппаратное обеспечение:

– операционная система Windows 95/98 или Windows NT 4.0;

– интерфейс Crypton API v.2.2 или выше;

– УКЗД КРИПТОН с соответствующим драйвером или его программный эмулятор.

Система Crypton Sigma специально разрабатывалась так, чтобы сделать все процедуры управления максимально простыми и ясными. Все, что должен уметь пользователь, – это создать специальный файл (контейнер) для хранения зашифрованных данных и открыть его для доступа через логический диск системы Crypton Sigma.

Контейнер – это специальный файл, создаваемый при помощи панели управления системы Crypton Sigma. Контейнер можно открыть для доступа через логический диск, обслуживаемый драйвером системы Crypton Sigma. Все файлы, находящиеся на этом логическом диске, хранятся в зашифрованном виде. Пользователь может создать любое количество контейнеров. Каждый контейнер имеет собственный пароль. Пользователь должен ввести этот пароль при создании контейнера и использовать его для получения доступа к тем данным, которые будут храниться в данном контейнере. Используя панель управления Crypton Sigma, пользователь может сменить пароль для выбранного контейнера при условии, что ему известен прежний пароль.

Схема работы системы Crypton Sigma показана на рисунке 8.20.

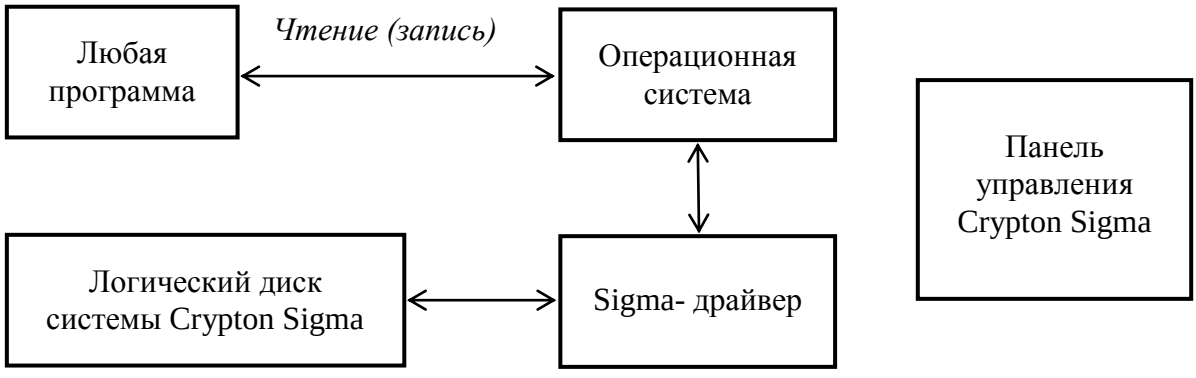


Рисунок 8.20 – Схема выполнения операций чтения (записи) с логических дисков системой Crypton Sigma

Логический диск системы Crypton Sigma создается и управляется драйвером этой системы. Этот логический диск используется для записи (чтения) данных в контейнер.

Работа пользователя с таким логическим диском не отличается от работы с любыми другими дисками компьютера. Драйвер системы Crypton Sigma обрабатывает запросы операционной системы на чтение (запись) с логических дисков, при этом драйвер автоматически производит зашифрование/расшифрование данных. Следует отметить, что драйвер системы Crypton Sigma обрабатывает не все запросы на чтение/запись. Система Crypton Sigma создает и обслуживает собственные логические диски. Драйвер системы обслуживает операции чтения (записи) только с этих логических дисков.

Эти диски доступны точно так же, как и остальные диски на компьютере, и могут обозначаться любыми незанятыми на данный момент буквами, например D:, E:, K:, Z:.

Данные, записываемые на логический диск, фактически записываются драйвером системы в контейнер системы. Естественно, размер доступной памяти логического диска равен размеру соответствующего контейнера.

Максимальный размер контейнера, создаваемого:

– на жестком диске с файловой системой FAT16, равен 2 Гбайта;

– на жестком диске с файловой системой FAT32. равен 4 Гбайта;

– на жестком диске с файловой системой NTFS, равен 512 Гбайт;

– на сетевом диске, равен 2 Гбайта.

Минимальный размер контейнера системы равен 5 Кбайт.

Для доступа к зашифрованным данным, хранящимся в контейнере, следует присоединить этот контейнер к выбранному логическому диску, например E:, и открыть его для доступа, введя соответствующий пароль. После завершения работы с данными следует закрыть этот логический диск для доступа. При этом данные, сохраненные в контейнере, станут недоступными.

Следует заметить, что если пользователь забудет пароль для доступа к данным, хранящимся в контейнере системы Crypton Sigma, то он полностью теряет возможность доступа к этим данным. Высокостойкие алгоритмы шифрования, используемые в системе Crypton Sigma, не позволяют восстановить информацию без знания пароля. Если существует опасность того, что пароль может быть забыт или утрачен, пользователь должен записать его и спрятать в надежном месте.

Отметим основные преимущества системы Crypton Sigma:

1) Надежная защита. Практически ни одна из существующих универсальных программ со встроенной защитой документов не имеет такой надежной защиты как Crypton Sigma. Компания Access Data (США) продает программный пакет, который вскрывает защиту данных в WordPerfect, Lotus 1-2-3, MS Excel, Symphony, Quattro Pro, Paradox, MS Word. Эта программа не просто перебирает все возможные комбинации паролей она проводит математически обоснованный криптографический анализ и тратит на вскрытие защищенных данных всего лишь несколько секунд. Система Crypton Sigma выгодна отличается использованием стойких и надежных алгоритмов шифрования и не содержит встроенных программных блоков, позволяющих злоумышленнику совершить несанкционированный доступ к зашифрованным данным.

2) Высокая степень секретности. После того как данные записываются на логический диск системы Crypton Sigma, они уже никогда не хранятся на компьютере в открытом (расшифрованном) виде. Расшифрование данных происходит только в момент доступа к ним пользователей, знающих пароль. Система Sigma нигде не хранит паролей, необходимых для доступа к данным. Она лишь проверяет, подходит ли пароль для расшифрования данных, на которые претендует пользователь, точно так же, как замок нигде не хранит дубликат ключа, а только «проверяет», может ли данный ключ открыть его или нет.

3) Использование системы в локальных сетях. Программное обеспечение Crypton Sigma для Windows 95/98/NT позволяет использовать любой сетевой диск для создания на нем контейнеров и доступа к хранящимся на них данным. Эти сетевые диски могут быть выделены для доступа компьютерами с любой другой, отличной от Windows, операционной системой, например системой семейства UNIX (OSF/1, LINUX, BSD, Sun OS, AIX и др.), а также Novell, Windows и др. Логические диски Crypton Sigma с точки зрения операционной системы или любого другого программного обеспечения выглядят точно так же, как обыкновенные локальные диски компьютера. Поэтому логические диски Crypton Sigma могут быть открыты для доступа через локальную компьютерную сеть. Таким образом, зашифрованная информация при необходимости может быть доступна для коллективного использования.

4) Удобство в использовании. Система Crypton Sigma проста в использовании и, следовательно, практически не позволяет совершать случайных действий, приводящих к появлению секретной информации на компьютере в открытом виде. Пользователю необходимо только ввести правильный пароль – об остальном позаботится система. После верификации пароля доступ к зашифрованной информации становится прозрачным для всех запускаемых пользователем программ. Все зашифрованные данные автоматически расшифровываются перед тем, как появиться перед пользователем, и автоматически зашифровываются перед записью их на диски, обслуживаемые системой Crypton Sigma.

8.3 Контрольные вопросы

1) Дайте общую характеристику программного комплекса PGP (Pretty Good Privacy).

2) Дайте общую характеристику системе криптографической защиты информации от несанкционированного доступа КРИПТОН-ВЕТО.

3) Дайте общую характеристику комплексу КРИПТОН-ЗАМОК для ограничения доступа к компьютеру.

4) Дайте общую характеристику системе защиты данных Crypton Sigma.

9 Лабораторный практикум

9.1 Лабораторная работа № 1. Защита данных с использованием шифра Цезаря

Цель работы: Освоить технологию шифрования и расшифрования информации в среде Excel с использованием шифра Цезаря

А. Порядок выполнения лабораторной работы

1) Войти в среду Excel и создать свою симметричную криптосистему (по системе Цезаря):
   – перейти на второй лист книги Excel. Начиная с ячейки A1 до A40 набрать алфавит, при этом в первые восемь ячеек столбца А внести знаки препинания и пробел, как это показано на рисунке 9.1.1. Выделить весь диапазон алфавита от A1 до A40 и назначить ему имя АВС;
   – на первом листе этой книги Excel (см. рисунок 9.1.2) в ячейке B1 набрать текст, который необходимо зашифровать, например: Гай Юлий Цезарь:"Пришел, увидел, победил!". При наборе текста необходимо использовать только те символы, которые входят в алфавит;
   – в ячейке B3 записать формулу =ПРОПИСН(B1), функция ПРОПИСН переводит буквенные символы в строке в прописные буквы;
   – в ячейке D3 записать формулу =ДЛСТР(B3), функция ДЛСТР позволяет определить длину строки, что необходимо пользователю, для кодировки исходной строки;
   – в ячейку D4 записать значение ключа *k*, например, 5;
   – в столбце А, начиная с ячейки A6, пронумеровать ячейки числами последовательного ряда от 1 до N, где N – число символов в тексте, включая пробелы. Значение N рассчитано в ячейке D3;
   – в ячейку B6, записать формулу =ПСТР(B$3;A6;1), которая разделяет кодируемый текст на отдельные символы. Скопировать эту формулу в диапазон ячеек от B7 до B47;
   – в ячейку C6 записать формулу =ПОИСКПОЗ(B6;АВС;0). Функция ПОИСКПОЗ производит поиск индекса (номера позиции) символа в массиве АВС, который был определен на листе 2. Скопировать содержимое ячейки C6 в диапазон ячеек от C7 до C47;
   – получив номер символа в алфавите АВС, произвести сдвиг нумерации алфавита для кодируемой последовательности символов. В ячейку D6 записать формулу:

=ЕСЛИ(ПОИСКПОЗ(B6;АВС;0)+$D$4>40;ПОИСКПОЗ(B6;АВС;0)+ $D$4-40;ПОИСКПОЗ(B6;АВС;0)+$D$4) (1)

   Эта формула производит сдвиг номеров символов алфавита на величину *k* и определяет номер заменяющего символа из алфавита АВС;
   – содержимое D6 скопировать в диапазон ячеек от D7 до D47;

– выбрать символы из алфавита АВС в соответствии с новыми номерами. В ячейку Е6 записать формулу =ИНДЕКС(АВС;D6). Скопировать содержимое ячейки Е6 в диапазон ячеек от Е7 до Е47;
– для получения строки зашифрованного текста необходимо в ячейку F6 записать =E6, в ячейку F7 соответственно – =F6&E7. Далее скопировать содержимое ячейки F7, в диапазон ячеек от F8 до F47. В ячейке F47 прочитать зашифрованный текст.

2) Для проверки шифрования произвести расшифрование полученного текста и сравнить его с исходным:
– для этого на третьем листе книги Excel (см. рисунок 9.1.3) выполнить все действия аналогично первому пункту лабораторной работы. При этом необходимо учесть следующие особенности:
  а) в ячейке В1 третьего листа книги Excel набрать зашифрованный текст;
  б) в ячейку D6 третьего листа книги Excel вместо формулы (1) записать формулу:

=ЕСЛИ(ПОИСКПОЗ(B6;АВС;0)-$D$4<0;ПОИСКПОЗ(B6;АВС;0)-$D$4+40;ПОИСКПОЗ(B6;АВС;0)-$D$4) (2)

– получение исходного текста (без ошибок) в ячейке F47 третьего листа свидетельствует о корректном выполнении этой части лабораторной работы.

3) Для зашифрования и расшифрования своего сообщения:
– на первом листе книги Excel в ячейке В1 заменить текст: Гай Юлий Цезарь:"Пришел, увидел, победил!" на свой оригинальный текст общей длиной не более 42 символов. В ячейке F47 прочитать зашифрованный текст;
– на третьем листе книги Excel в ячейке В1 набрать этот зашифрованный текст вместо того, который там был;
– получение своего оригинального текста (без ошибок) в ячейке F47 третьего листа книги Excel свидетельствует о корректном выполнении этой части лабораторной работы.

4) Отправить по электронной почте одному из студентов группы зашифрованный текст своего сообщения. ВАЖНО! Сообщить ему по закрытому каналу связи свой КЛЮЧ (значение ключа *k*).

5) Для расшифрования сообщений от других абонентов:
– скопировать на четвертый лист книги Excel весь ее третий лист;
– получить по электронной почте от студента своей группы его шифртекст и ввести в ячейку В1 четвертого листа книги Excel вместо того шифртекста, который там был;
– ВАЖНО! Получить по закрытому каналу связи от этого студента КЛЮЧ (значение *k*) и ввести его ячейку D4 четвертого листа книги Excel;
– получение исходного текста (без ошибок) сообщения вашего абонента в ячейке F47 четвертого листа книги Excel свидетельствует о корректном выполнении этой части лабораторной работы.

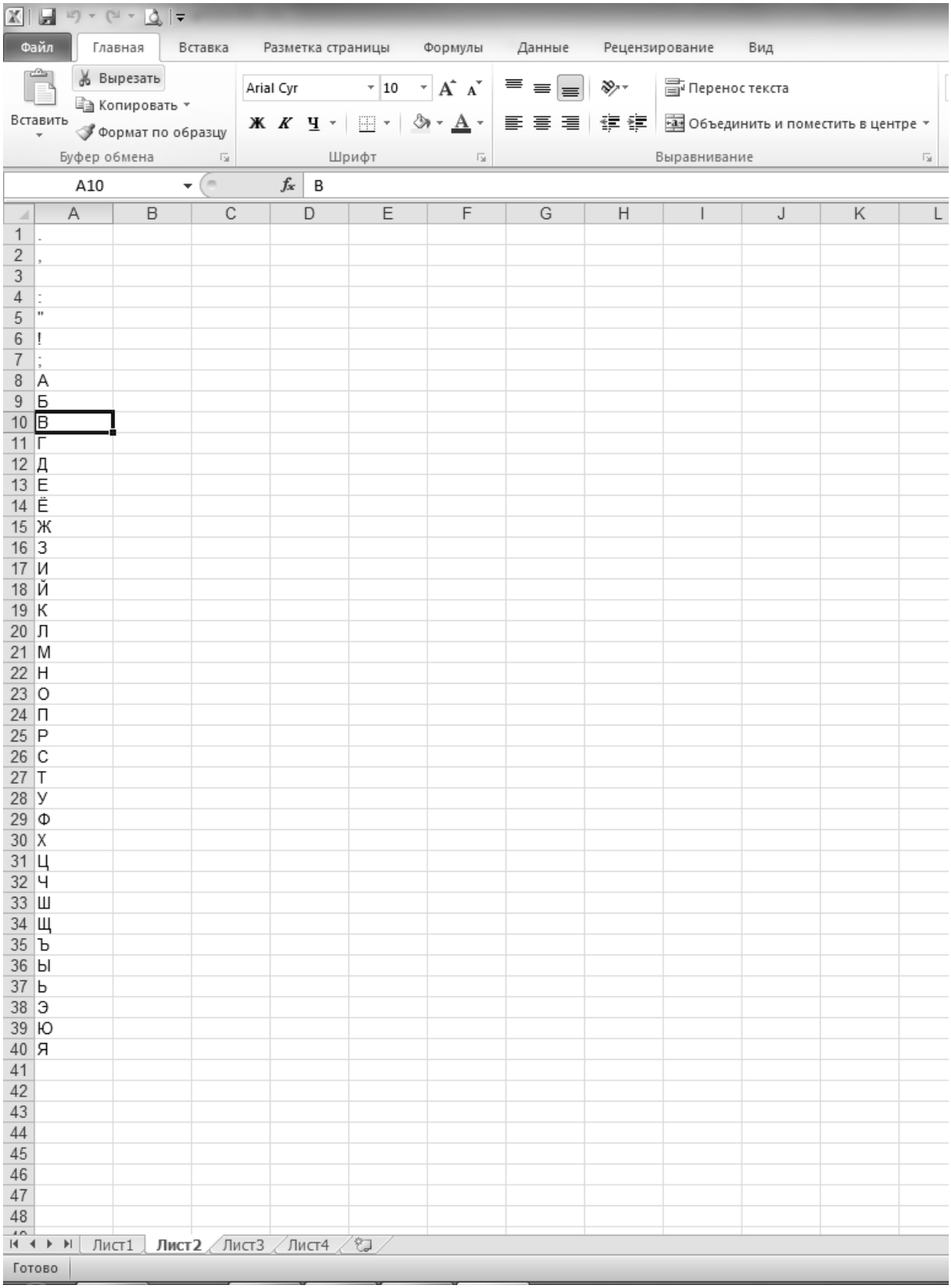
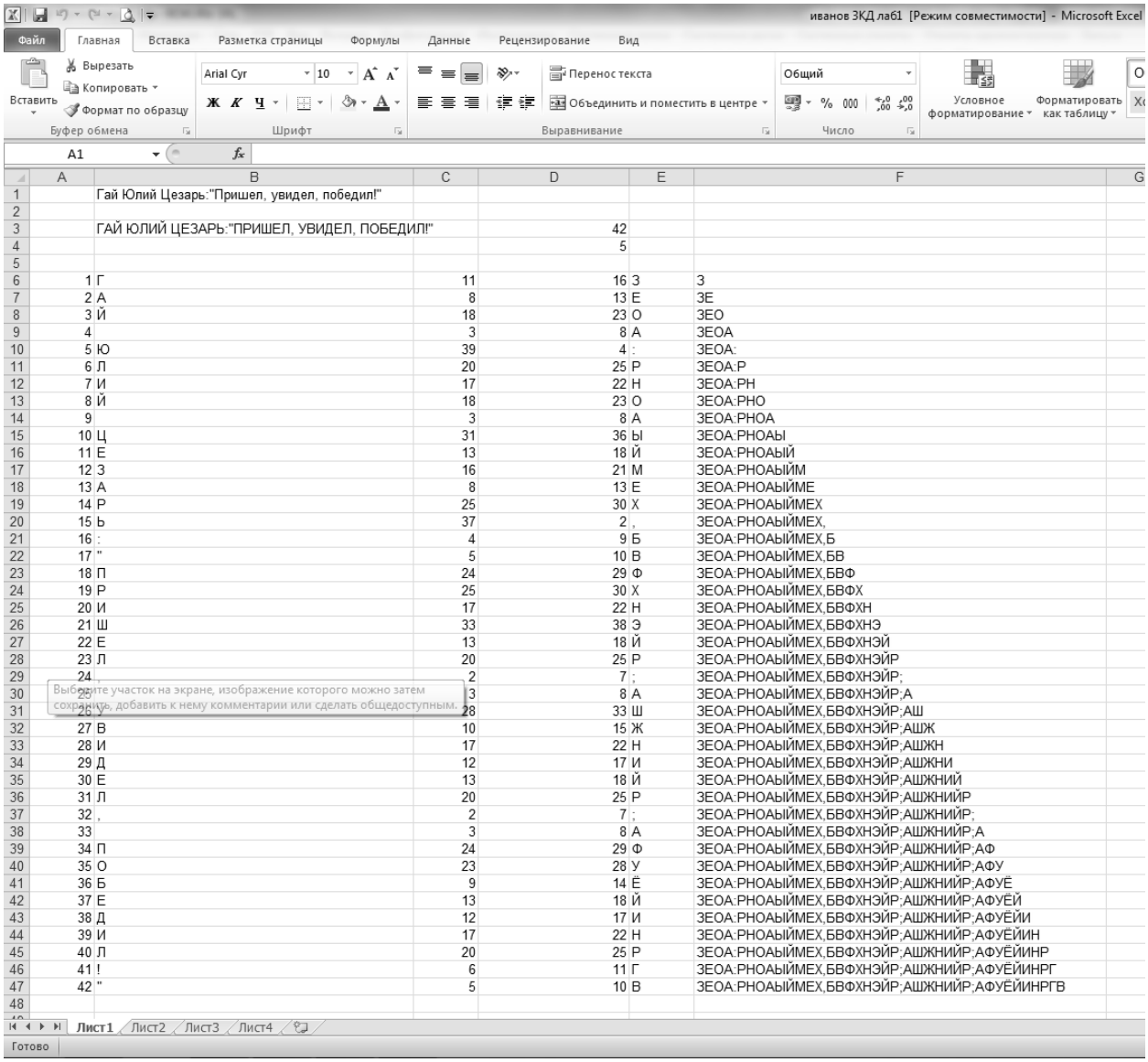


Рисунок 9.1.1 – Пример второго листа криптосистемы Цезаря в среде Excel

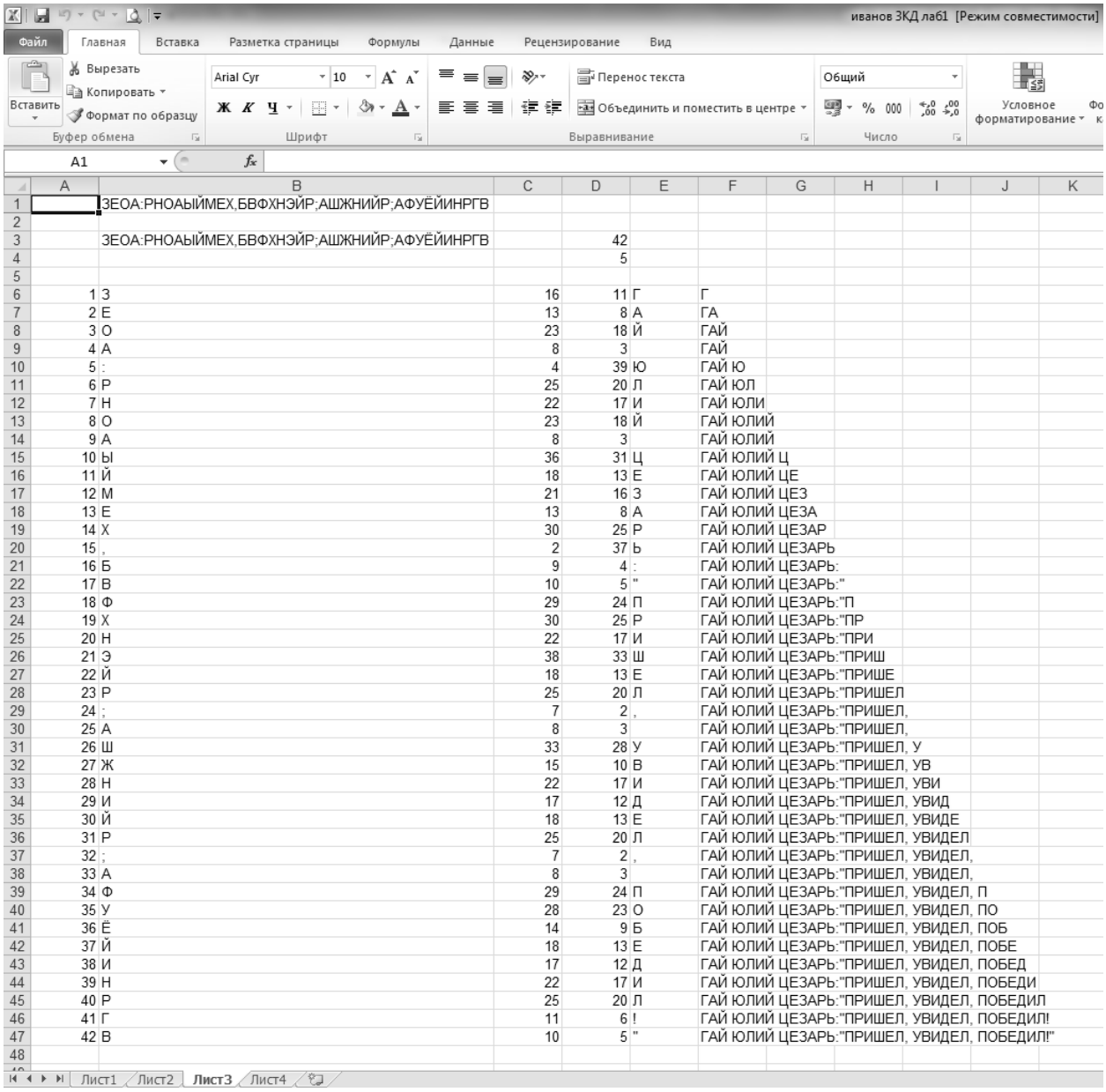


	A	B	C	D	E	F	G
1		Гай Юлий Цезарь:"Пришел, увидел, победил!"					
2							
3		ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ, ПОБЕДИЛ!"		42			
4				5			
5							
6	1	Г	11	16	З	З	
7	2	А	8	13	Е	ЗЕ	
8	3	Й	18	23	О	ЗЕО	
9	4		3	8	А	ЗЕОА	
10	5	Ю	39	4	:	ЗЕОА:	
11	6	Л	20	25	Р	ЗЕОА:Р	
12	7	И	17	22	Н	ЗЕОА:РН	
13	8	Й	18	23	О	ЗЕОА:РНО	
14	9		3	8	А	ЗЕОА:РНОА	
15	10	Ц	31	36	Ы	ЗЕОА:РНОАЫ	
16	11	Е	13	18	Й	ЗЕОА:РНОАЫЙ	
17	12	З	16	21	М	ЗЕОА:РНОАЫЙМ	
18	13	А	8	13	Е	ЗЕОА:РНОАЫЙМЕ	
19	14	Р	25	30	Х	ЗЕОА:РНОАЫЙМЕХ	
20	15	Ь	37	2	,	ЗЕОА:РНОАЫЙМЕХ,	
21	16	:	4	9	Б	ЗЕОА:РНОАЫЙМЕХ,Б	
22	17	"	5	10	В	ЗЕОА:РНОАЫЙМЕХ,БВ	
23	18	П	24	29	Ф	ЗЕОА:РНОАЫЙМЕХ,ББВФ	
24	19	Р	25	30	Х	ЗЕОА:РНОАЫЙМЕХ,ББВФХ	
25	20	И	17	22	Н	ЗЕОА:РНОАЫЙМЕХ,ББВФХН	
26	21	Ш	33	38	Э	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭ	
27	22	Е	13	18	Й	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙ	
28	23	Л	20	25	Р	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР	
29	24	,	2	7	;	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;	
30	25		3	8	А	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;А	
31	26	У	28	33	Ш	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШ	
32	27	В	10	15	Ж	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖ	
33	28	И	17	22	Н	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖН	
34	29	Д	12	17	И	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИ	
35	30	Е	13	18	Й	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙ	
36	31	Л	20	25	Р	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР	
37	32	,	2	7	;	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;	
38	33		3	8	А	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;А	
39	34	П	24	29	Ф	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;АФ	
40	35	О	23	28	У	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;АФУ	
41	36	Б	9	14	Ё	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;АФУЁ	
42	37	Е	13	18	Й	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;АФУЁЙ	
43	38	Д	12	17	И	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;АФУЁЙИ	
44	39	И	17	22	Н	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;АФУЁЙИН	
45	40	Л	20	25	Р	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;АФУЁЙИНР	
46	41	!	6	11	Г	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;АФУЁЙИНРГ	
47	42	"	5	10	В	ЗЕОА:РНОАЫЙМЕХ,ББВФХНЭЙР;АШЖНИЙР;АФУЁЙИНРГВ	

Рисунок 9.1.2 – Пример первого листа криптосистемы Цезаря в среде Excel

Таким образом, создана симметричная криптосистема (по системе Цезаря), при этом:

– первый лист книги Excel предназначен для зашифрования вашего произвольного сообщения;
– второй лист книги Excel содержит алфавит и знаки препинания;
– третий лист книги Excel предназначен для расшифрования шифртекста вашего сообщения;
– четвертый лист книги Excel предназначен для расшифрования шифртекстов сообщений ваших абонентов.



	A	B	C	D	E	F	G	H	I	J	K
1		ЗЕОА:РНОАЫЙМЕХ,ББФХНЭЙР;АШЖНИЙР;АФУЁЙИНРГВ									
2											
3		ЗЕОА:РНОАЫЙМЕХ,ББФХНЭЙР;АШЖНИЙР;АФУЁЙИНРГВ		42							
4				5							
5											
6	1	З	16	11	Г	Г					
7	2	Е	13	8	А	ГА					
8	3	О	23	18	Й	ГАЙ					
9	4	А	8	3		ГАЙ					
10	5	:	4	39	Ю	ГАЙ Ю					
11	6	Р	25	20	Л	ГАЙ ЮЛ					
12	7	Н	22	17	И	ГАЙ ЮЛИ					
13	8	О	23	18	Й	ГАЙ ЮЛИЙ					
14	9	А	8	3		ГАЙ ЮЛИЙ					
15	10	Ы	36	31	Ц	ГАЙ ЮЛИЙ Ц					
16	11	Й	18	13	Е	ГАЙ ЮЛИЙ ЦЕ					
17	12	М	21	16	З	ГАЙ ЮЛИЙ ЦЕЗ					
18	13	Е	13	8	А	ГАЙ ЮЛИЙ ЦЕЗА					
19	14	Х	30	25	Р	ГАЙ ЮЛИЙ ЦЕЗАР					
20	15	,	2	37	Ь	ГАЙ ЮЛИЙ ЦЕЗАРЬ					
21	16	Б	9	4	:	ГАЙ ЮЛИЙ ЦЕЗАРЬ:					
22	17	В	10	5	"	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"					
23	18	Ф	29	24	П	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"П					
24	19	Х	30	25	Р	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПР					
25	20	Н	22	17	И	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИ					
26	21	Э	38	33	Ш	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШ					
27	22	Й	18	13	Е	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕ					
28	23	Р	25	20	Л	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ					
29	24	;	7	2	,	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ,					
30	25	А	8	3		ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ,					
31	26	Ш	33	28	У	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, У					
32	27	Ж	15	10	В	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВ					
33	28	Н	22	17	И	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИ					
34	29	И	17	12	Д	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИД					
35	30	Й	18	13	Е	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕ					
36	31	Р	25	20	Л	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ					
37	32	;	7	2	,	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ,					
38	33	А	8	3		ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ,					
39	34	Ф	29	24	П	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ, П					
40	35	У	28	23	О	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ, ПО					
41	36	Ё	14	9	Б	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ, ПОБ					
42	37	Й	18	13	Е	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ, ПОБЕ					
43	38	И	17	12	Д	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ, ПОБЕД					
44	39	Н	22	17	И	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ, ПОБЕДИ					
45	40	Р	25	20	Л	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ, ПОБЕДИЛ					
46	41	Г	11	6	!	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ, ПОБЕДИЛ!					
47	42	В	10	5	"	ГАЙ ЮЛИЙ ЦЕЗАРЬ:"ПРИШЕЛ, УВИДЕЛ, ПОБЕДИЛ!"					
48											

Рисунок 9.1.3 – Пример третьего листа криптосистемы Цезаря в среде Excel

Б. Содержание отчета

1) Сохранить книгу Excel с вашей криптосистемой (содержащей четыре листа) в виде файла MS Excel (например, *иванов_3_кд_лаб1.xls*).
2) Прикрепить этот файл к письму и по электронной почте отправить преподавателю.

9.2 Лабораторная работа № 2. Работа с криптосистемой PGP: генерация ключей

Цель работы: Освоить технологию применения системы PGP. Научиться генерировать ключи

А. Порядок выполнения лабораторной работы

Криптосистема PGP позволяет иметь любое количество ключевых пар, однако ни одна из них не может быть надежнее защищающей ее ключевой фразы (пароля). Это полезно помнить, если появится желание иметь несколько ключевых пар. Лучше научиться работать с ключами одной пары.

Чтобы сгенерировать новую ключевую пару, необходимо сделать следующее:

1) Откройте пункт меню PGPkeys, для чего щелкните по иконке (замок) PGPtray на панели задач и выберите PGPkeys. Если программа только установлена, то открывшееся окно будет пустым (рисунок 9.2.1). Впоследствии в нем будут перечислены ваши ключевые пары, а также открытые ключи корреспондентов, добавленные вами на связку.

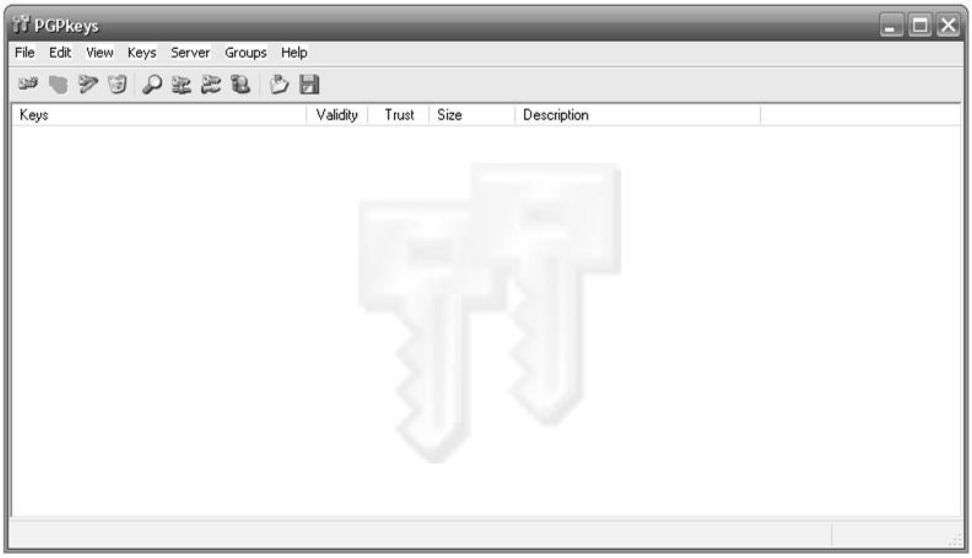


Рисунок 9.2.1 – Окно подпрограммы PGPkeys

В верхней части окна PGPkeys расположены:

– строка меню, предоставляющая доступ к функциям PGPkeys;

– панель инструментов с пиктограммами (иконками) наиболее часто выполняемых операций. В их числе:

– ключик (Generate new keypair) – создайте новую ключевую пару (keypair);

– красный крестик (Revoke the selected item) – отмените выделенный ключ (изделие);

– рука с карандашом (Sign the selected item) – подпишите выделенный ключ (изделие);

– корзина (Delete the selected item) – удалите выделенный ключ (изделие);
– лупа (Open key search window) – откройте ключевое окно поиска;
– ключ на проводе (Send key to server) – пошлите выделенный ключ на сервер;
– ключ на подносе (Update key from server) – модернизируйте ключ из сервера;
– профиль со знаком вопроса (Show key certificate properties) – откройте окно параметров выделенного ключа;
– открытая папка (Import keys from file) – импорт выбранных ключей из файла на связку;
– дискета (Export selected keys to a file) – экспорт выбранных ключей со связки в файл.

2) Щелкните в панели инструментов по иконке (Generate new keypair – произведите новый keypair). Появится окно мастера генерации ключа (рисунок 9.2.2).

Рисунок 9.2.2 – Окно мастера генерации ключа

3) Нажмите кнопку Expert**,** чтобы перейти к расширенным параметрам создаваемого ключа. На экране появится окно с предложением назначить параметры ключевой пары (рисунок 9.2.3):

– Поле Full name (Полное название (имя)) – поле для ввода имени, которое будет узнаваться вашими корреспондентами.

– Поле E-mail address (Адрес электронной почты) – поле для ввода адреса, указание которого позволит корреспондентам находить на связке открытых ключей те, которые принадлежат вам.

Рисунок 9.2.3 – Окно для ввода имени и адреса электронной почты

– Параметры Key type (Тип ключа), Key size (Размер ключа) лучше принять по умолчанию.

– Параметр Key Expiration (Истечение срока ключа) – может иметь установленное по умолчанию Never (никогда), либо принимать значение даты, с которой ключевая пара не сможет применяться для новых криптографических операций.

– Кнопка More Information (Подробная информация) обеспечивает вывод на экран разделов документации с пояснением назначения указанных в окне параметров.

4) Нажмите Далее. На экране появится окно (рисунок 9.2.4) для ввода ключевой фразы Pass phrase (фразы прохода или пароля).

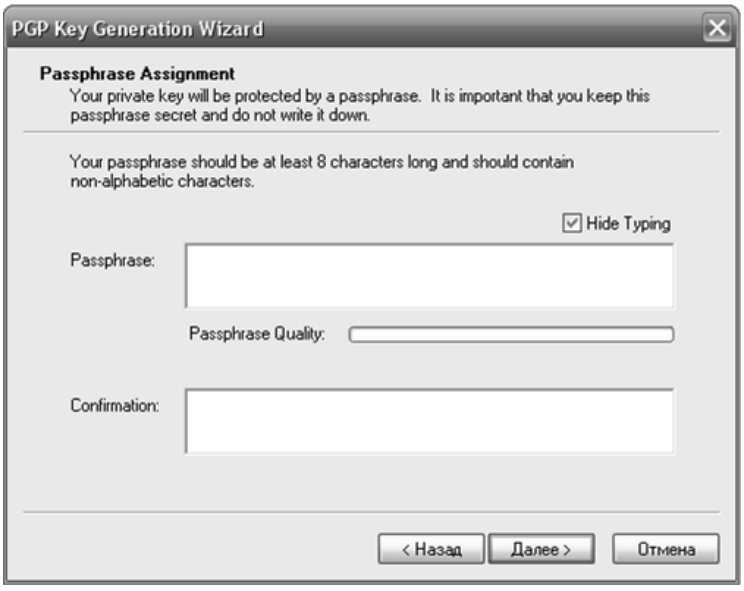


Рисунок 9.2.4 – Окна для ввода и контроля ключевой фразы

В окне есть напоминание о том, что ключевая фраза должна содержать не менее 8 символов (желательно не буквенных) и что ключевую фразу (синонимы: пароль, ключ) необходимо хранить в тайне (дословно – не записывая ее). Ключевая фраза вводится дважды. «Галочка» Hide Typing предлагает выполнять набор без отображения (Скройте Печатание). После первого набора окно Pass phrase Quality (Качество фразы прохода) будет частично закрашено, напоминая о необходимости второго набора. Если введенная при втором наборе ключевая фраза отличается от фразы первого набора, PGP выдаст предупреждение. Необходимо вернуться для повторения наборов.

Если введенные ключевые фразы совпадут, на экране появится окно (рисунок 9.2.5) процесса генерации ключевой пары (Done (сделано) в строке Current status – Текущий статус).

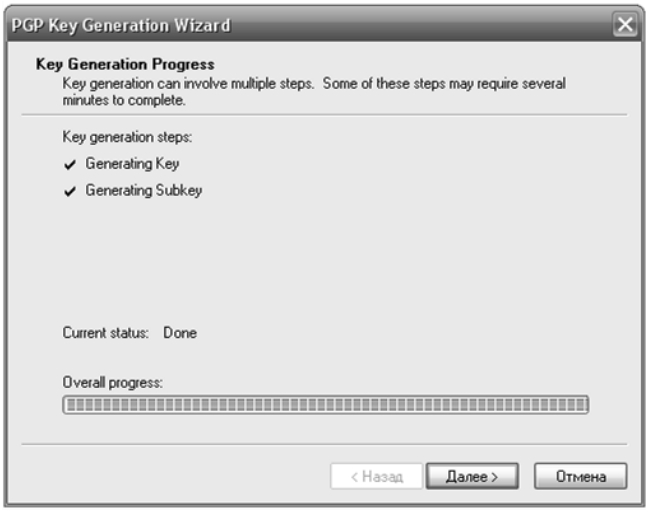


Рисунок 9.2.5 – Окно процесса генерации ключевой пары

5) Нажать кнопку Далее. На экране появится окно завершения работы мастера генерации ключевой пары (рисунок 9.2.6).

Рисунок 9.2.6 – Окно завершения работы мастера генерации ключевой пары

6) Нажать кнопку Готово**.** На экране появится окно PGPkeys (рисунок 9.2.7), которое раньше было пустым. Теперь в окне представлены сведения о сгенерированной ключевой паре.

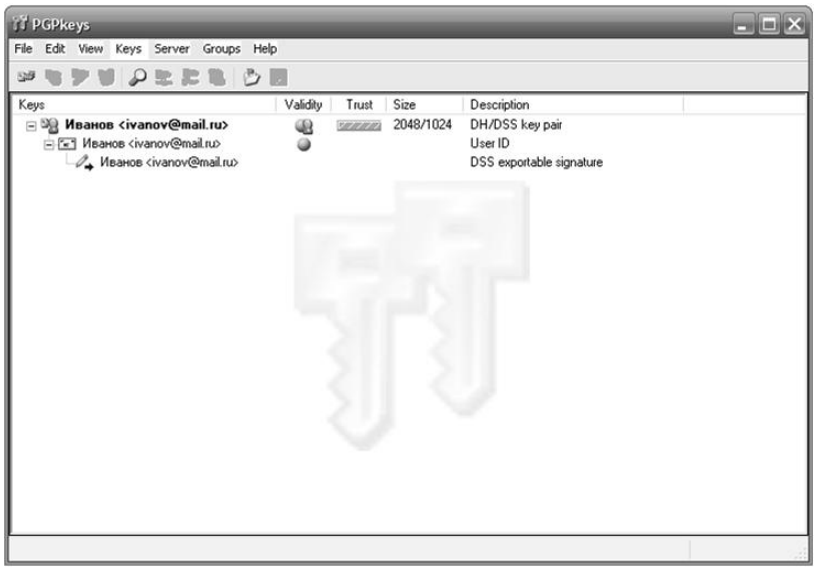


Рисунок 9.2.7 – Сведения о сгенерированной ключевой паре

Окно PGPkeys может отображать следующие параметры ключей:
– Keys (имена) – это ваше имя и e-mail-адрес, введенные при генерации;
– Validity (достоверность) – применительно к чужим ключам обозначает степень убежденности в том, что данный открытый ключ действительно принадлежит предполагаемому владельцу. Зависит от состава подписей, заверяющих данный ключ, и уровней доверия пользователя поручителям (людям, подписавшим ключ). Ключ, не имеющий подписей, считается частично достоверным, о чем программа будет напоминать всякий раз при попытке зашифровать информацию данным ключом;
– Size (размер). Для DH/DSS: 2048 бит – ключ шифрования, а 1024 бит – ключ подписания;
– Key ID (идентификационный номер);
– Trust (уровень доверия);
– Creation Date (дата создания);
– Expiration Date (дата окончания действия);
– ADK (наличие дополнительных ключей расшифрования);
– Description (описание).

Каждый из параметров может иметь несколько признаков (атрибутов), почти каждому из которых соответствует пиктограмма.

Таким образом, в окне PGPkeys после генерации ключевой пары отображаются:
– сгенерированная пользователем асимметричная пара «открытый ключ/ закрытый ключ» типа DH/DSS. (на практике чаще говорят просто о ключе);
– имя простого сертификата PGP (так называемый Open PGP);
– экспортируемая со связки подпись.

Эта пара, пока единственная, является одновременно и основной. Основная пара (основной ключ), или ключ по умолчанию, используется криптосистемой PGP при расшифровании поступивших сообщений, а также при подписании отправляемых сообщений или при подписании открытого ключа корреспондента, для которого вы выступаете доверителем. В окне PGPkeys основной ключ выделен жирным шрифтом, дабы отличить его от остальных.

Если используется более одной ключевой пары, выделение основной пары упрощает работу с PGP. Чтобы выбрать основной ключ:

– в окне PGPkeys надо выделить ключ, который должен стать основным;
– в строке меню окна PGPkeys выбрать функцию Keys, а в ней – опцию Set Default.

Имя ключа станет жирным, обозначая, что теперь он основной.

PGP автоматически направит открытый ключ в файл pubring.pkr, а закрытый ключ – в файл secring.skr. Файлы первоначально размещаются, скорее всего, в каталоге с программой PGP или там, куда определено при настройке (закладка Files).

После генерации ключевой пары необходимо сделать несколько ее резервных копий. Игнорирование этой рекомендации приводит к неоправданному риску потери в последующем всех данных поступивших сообщений. К примеру, если что-то случится с единственным файлом связки закрытых ключей, никто во всем мире не поможет расшифровать адресованные вам сообщения.

Кроме резервного копирования pkr- и skr-файлов связки ключей, необходимо уделить особое внимание хранению закрытого ключа. И хотя закрытый ключ защищен ключевой фразой, известной только вам, недоброжелатель, случайно узнав ее, сможет воспользоваться закрытым ключом, чтобы расшифровывать вашу информацию и делать от вашего имени подписи. Поэтому закрытый ключ рекомендуется хранить на дискете (или на смарт-карте), а перед применением копировать его в адрес, который был указан при настройке криптосистемы (закладка Files).

Размещение и хранение своих ключей, а также открытых ключей ваших корреспондентов связано с выполнением операций их экспортирования и импортирования.

Б. Содержание отчета

1) После того как вы потренируетесь генерировать ключевые пары, сгенерируйте ОДНУ ЕДИНСТВЕННУЮ свою ключевую пару, выбрав в качестве ее имени свою фамилию, а в качестве адреса свой реальный адрес электронной почты. Хорошенько запомните свою КЛЮЧЕВУЮ ФРАЗУ.
2) Сохраните резервную копию ключевой пары (файлы pubring.pkr и secring.skr) на свой флэш-диск.

9.3 Лабораторная работа № 3. Работа с криптосистемой PGP: экспорт и импорт ключей

Цель работы: Освоить технологию применения системы PGP. Научиться экспортировать и импортировать открытые ключи, активировать и дезактивировать ключи

А. Порядок выполнения лабораторной работы

Экспортирование и импортирование открытых ключей

Наиболее распространенным способом обмена ключами является их пересылка через сервер-депозитарий. Частным же лицам удобнее оперировать файлами, решая таким способом и проблемы хранения своих ключей.

1) Чтобы сохранить свой ключ в виде отдельного файла (*экспорт ключа*) необходимо:
   – в окне PGPkeys выделить ключ, который намечено экспортировать;
   – в строке меню окна PGPkeys выбрать функцию Keys, а в ней – опцию Export. На экране появится окно Export Key to File (рисунок 9.3.1) для указания места хранения экспортируемого ключа;

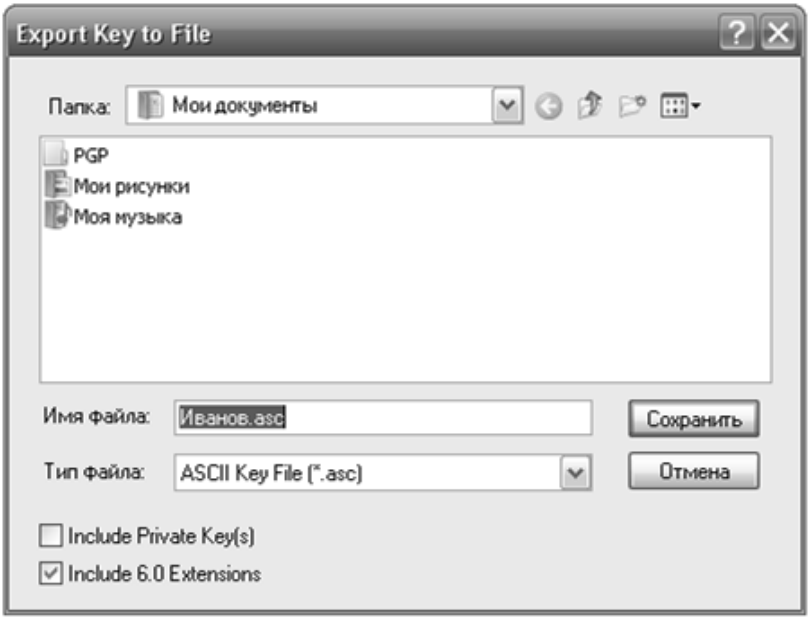


Рисунок 9.3.1 – Окно для указания места хранения экспортируемого ключа

   – в окне Папка необходимо указать каталог хранения экспортируемого ключа, а в окне Имя файла — имя и расширение файла. По умолчанию PGP предлагает для файла расширение .asc, что соответствует текстовой форме представления открытого ключа (вариантами расширения могут быть: pkr, pubkr);
   – опция Include 6.0 Extensions включается при необходимости экспортирования вместе с ключом и фото (в этом случае сам ключ будет несовместим с версиями PGP ниже 6.0);

– опция Include Private Keys включается при необходимости включения в файл и закрытого ключа. Понятно, что полученный при этом файл не подлежит передаче!!! и используется только пользователем;
– нажать клавишу Сохранить.

2) При необходимости копирования ключа со связки (разумеется, кроме закрытого ключа) в любой текстовый файл или в сообщение (*экспорт ключа*) необходимо:
– в окне PGPkeys выделить ключ, который намечено экспортировать;
– в строке меню окна PGPkeys выбрать функцию Edit, а в ней - опцию Copy.

Текст ключа представляет собой набор символов, ограниченный фразами: ---- BEGIN и ---- END.

Например:

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP 8.0.2 - not licensed for commercial use: www.pgp.com

mQGiBFDrGzwRBADCEC4z9eJyJ6HrIp3QtqosS+zFWx8m2aHvWHoWMtPsrnte490x
JMF8JdDKhrMviqgxir3k2z4fCKBuJ+9MoDduzhPwlMFwjoDc6WVQumRiT2zW2AGY
0ofyufZkr9uUbJmeSg+GSiZ+83iPoHfmAzBtGVANWQG47atFE5urIZu09wCg/yvo
2Cgfr2dmH6t98d5U/Y47vg0D/jXtprV9ifDPSnVvy868zlGd9n5mM1tbDim7vVqN
Stx+w9MBXNinfr3nCTWmb/K6kbePiA+CSeh35s2JmeCjmPAYxMTiUIhkTtqY3EKv
XqMKXLYZQK03f8CTyKi1qjbdDGaljSzs+HCo108PUgh1HnUO9kLOrw4vY7yLLSNJ
ttYFA/9f6s0vXEsmpKXxK93fu67Afkt3znBf1KT7L63+imBKG+inKwcv6+k0j1QR
t5to+QajomaZ7s8mK9W8/7bvDzK+kJAVl830OTdRLHLesvPzsFoqFGN/ILXMvDlo
rksM2Vna1kTm5KgDWN9oDeanqWE3yttLDzBKY/CyBytAM7dWabQd0JjQstCw0L
vtCyIDxpdmFub3ZAbWFpbC5ydT6JAFcEEBECABcFAlDrGzwHCwkIBwMCCgIZAQU
AwAAAAKCRBDeKPmD9puwVTaAKCsaSJucijbkT8iAfpsQZPl1CM0vgCeLNxa8Oe
=e/Nd
-----END PGP PUBLIC KEY BLOCK-----

Этот текст находится в буфере обмена, откуда его можно вставить в сообщение простой опцией Paste функции Edit или комбинацией клавиш Ctrl+V.

3) Для того чтобы добавить в связку ключ, принятый от корреспондента в виде файла, то есть выполнить операцию *импорта ключа*, необходимо:
– в строке меню окна PGPkeys выбрать функцию Keys, а в ней – опцию Import. На экране появится окно Select File Containing Key (рисунок 9.3.2) для указания папки хранения принятого файла с ключом (или ключами, если корреспондент прислал файл с несколькими открытыми ключами);
– в строке Папка указать директорию размещения принятого файла с открытым ключом корреспондента;

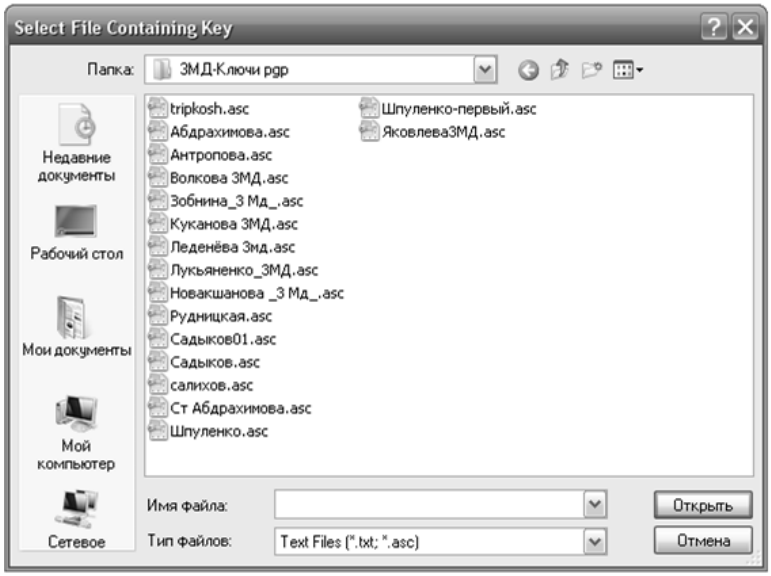


Рисунок 9.3.2 – Окно для указания папки хранения принятого файла с ключом

– в строке Тип файла указать расширение принятого файла (вариантами расширения могут быть: txt, asc, pkr, pubkr);
– в появившемся списке выделить принятый файл с ключом корреспондента и нажать кнопку Открыть. На экране появится окно Select Keys с параметрами ключа (ключей) (рисунок 9.3.3);

Рисунок 9.3.3 – Окно с параметрами ключа (ключей)

– выделить строку с ключом (или строки с ключами, если их в файле несколько) и нажать кнопку Import. Ключ будет присоединен к списку в окне PGPkeys.

4) Если открытый ключ вставлен в сообщение, полученное от корреспондента, то для добавления его на связку (*импорт ключа*) необходимо:
– в полученном сообщении выделить блок, начиная с заголовка
---- BEGIN PGP PUBLIC KEY BLOCK ----
и заканчивая строкой
---- END PGP PUBLIC KEY BLOCK ----

и скопировать выделенный материал в буфер обмена (обычно можно просто нажать Ctrl+C);

– в строке меню PGPkeys выбрать функцию Edit, а в ней Paste. На экране появится окно Select Keys с параметрами ключа;
– выделить строку с ключом и нажать кнопку Import. Ключ будет присоединен к списку в окне PGPkeys.

Активирование и дезактивирование ключей

Если количество ключей на связке велико, и поиск нужного ключа для зашифрования сообщения затруднен, можно временно дезактивировать часть ключей, чтобы сделать их в окне менее заметными.

1) Для дезактивирования чужого открытого ключа необходимо:
   – в окне PGPkeys выделить ключ, который выбран для дезактивирования;
   – в строке меню окна PGPkeys выбрать функцию Keys, а в ней Disable. Пиктограмма ключа потускнеет, обозначая, что он временно дезактивирован.
2) Для активирования чужого открытого ключа необходимо:
   – в окне PGPkeys выделить ключ, который выбран для активирования;
   – в строке меню окна PGPkeys выбрать функцию Keys, а в ней Enable. Пиктограмма станет обычной, а ключ – готовым к работе.

Сертификация ключей

При работе с криптосистемой с открытым ключом необходима уверенность в том, что открытый ключ, которым вы собираетесь что-то зашифровать – не искусная имитация, а истинная собственность вашего корреспондента. Процедура взаимного заверения корреспондентами друг друга в подлинности открытых ключей, принадлежащих им (или их общим знакомым), – это краеугольный камень распределенной модели доверия Web of Trust, лежащей в основе PGP. Один из приемов этой процедуры основан на использовании цифрового отпечатка ключа.

1) Цифровой отпечаток открытого ключа (fingerprint) – это хеш-значение ключа, столь же уникальное, сколь и сам ключ.

   Чтобы просмотреть отпечаток ключа, в окне PGPkeys необходимо:
   – нажать правой кнопкой на имя ключа, а в появившемся меню выбрать опцию Key Properties (Ключевые Свойства). На экране появится окно свойств выделенного ключа (рисунок 9.3.4);
   – закладке General этого окна имеется раздел Fingerprint (Отпечаток пальца как символ неповторимости), в котором отпечаток может быть представлен в двух формах: в виде уникального списка слов (левый отпечаток) или в виде шестнадцатеричного числа (правый отпечаток). По умолчанию отпечаток отображается в виде списка слов.

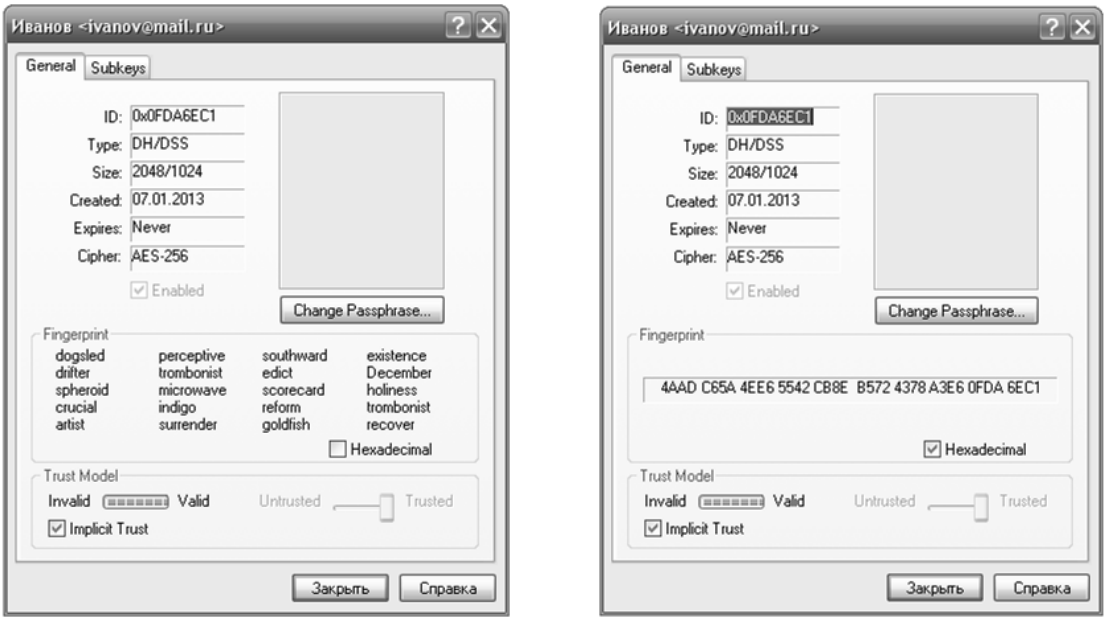


Рисунок 9.3.4 – Окна свойств выделенного ключа

Шестнадцатеричное число отпечатка ключа (в данном случае это 4AAD C65A 4EE6 5542 CB8E B572 4378 A3E6 0FDA 6EC1) можно увидеть, отметив «галочкой» опцию Hexadecimal.

Если вы увидели отпечаток своего ключа, то это число можно выделить и скопировать, а затем вставить его в любой документ для передачи вашему корреспонденту.

Проделав такую же процедуру с импортированным ключом вашего корреспондента и сравнив отпечаток с его копией в полученном от него же документе, можно принять решение о подписании ключа. Этим вы заверите других пользователей, что, по вашему мнению, данный открытый ключ действительно принадлежит человеку, чье имя указано на этом ключе, а сам ключ не является злоумышленной подменой. Иными словами, вы станете поручителем этого ключа, и третьи лица, которые вам доверяют, будут расценивать подписанный ключ априорно достоверным (то есть им не надо самим проделывать такую же процедуру). Это и есть модель Web of Trust в действии.

2) Сведения об открытом ключе, заверенные цифровой подписью, понимаются как цифровой сертификат ключа.

Сертификат характеризуется форматом. Один из форматов, поддерживаемых PGP (Open PGP), содержит, в частности, следующие сведения:

– открытый ключ владельца сертификата – открытая часть
– ключевой пары и ее алгоритм (RSA или DH/ DSS);
– сведения о владельце сертификата – информация, идентифицирующая личность пользователя (его имя, адрес электронной почты, фотография и т.д);
– ЭЦП владельца сертификата – подпись ключевой пары, связанной с сертификатом (так называемая автоподпись);

– период действия сертификата – дата начала действия сертификата и дата окончания его действия; указывает на то, когда сертификат станет недействительным;
– предпочтительный алгоритм шифрования – указывает на то, зашифрованную каким алгоритмом информацию предпочитает получать владелец сертификата (CAST, AES, IDEA, Triple-DES и Twofish).

Сертификат Open PGP может быть представлен в виде открытого ключа с одной или несколькими привязанными к нему подписями («бирками»).

На одной из «бирок» указана информация, идентифицирующая владельца ключа (ID), а также его цифровая подпись, подтверждающая, что ключ и идентификационные сведения взаимосвязаны. (Этот вид подписи называется автоподписью (self-signature); ее содержит каждый PGP-сертификат).

В переписке хорошо знакомых корреспондентов можно обойтись использованием простых сертификатов. Небольшим группам людей, нуждающимся в защищенной связи, не составит труда просто передать друг другу дискеты или отправить электронные письма, содержащие копии их ключей. Это ручное распространение открытых ключей, и оно эффективно только до определенного этапа.

Дальнейшее – за пределами возможностей данного способа, и тогда возникает необходимость развертывания системы, которая бы обеспечивала достаточную надежность и безопасность, предоставляла бы возможности хранения и обмена ключами. Такая система может создаваться в форме простого хранилища-депозитария, называемого сервером-депозитарием открытых ключей или в виде специальных удостоверяющих центров.

Б. Содержание отчета

1) После того как вы изучите содержание предыдущих пунктов и потренируетесь в их выполнении:
   – экспортируйте СВОЮ КЛЮЧЕВУЮ ПАРУ на свой флэш-диск;
   – распространите (экспортируйте) СВОЙ ОТКРЫТЫЙ КЛЮЧ среди студентов своей группы;
   – добавьте на свою связку ключей открытые ключи студентов своей группы.
2) Экспортируйте СВОЙ ОТКРЫТЫЙ КЛЮЧ на флэш-диск преподавателя.
3) Экспортируйте СВОЙ ОТКРЫТЫЙ КЛЮЧ в текстовый формат и вставьте его в документ MS Word. Назовите файл этого документа следующим образом: *фамилия_курс_специальность_лаб3.doc* (например, *иванов_3_мд_лаб3.doc*).
4) Создайте отпечаток своего ключа в виде списка слов. Вставьте screenshot этого отпечатка в файл с ключом.
5) Создайте отпечаток своего ключа в виде шестнадцатеричного числа. Вставьте screenshot этого отпечатка в файл с ключом.
6) Прикрепите этот файл к письму электронной почты и отправьте его на почтовый адрес преподавателя.

9.4 Лабораторная работа № 4. Работа с криптосистемой PGP: зашифрование и расшифрование сообщений

Цель работы: Освоить технологию применения системы PGP. Научиться зашифровывать и расшифровывать сообщения

А. Порядок выполнения лабораторной работы

Для подготовки зашифрованного сообщения, предназначенного для передачи по электронной почте, в криптосистеме PGP применяются два способа:

– подготовка зашифрованного сообщения через буфер обмена;

– подготовка зашифрованного сообщения из файла.

Первый способ

1) Для подготовки зашифрованного сообщения в криптосистеме PGP через буфер обмена необходимо:

– вызвать в окно текст нужного сообщения и простейшим способом скопировать этот текст в буфер обмена (Clipboard) (через меню Правка – Копирование или сочетанием клавиш Ctrl + C);

– щелкнуть по изображению замочка PGPtray на панели задач и в появившемся меню выбрать функцию Clipboard, а в ней – Engrypt (зашифровать). На экране появится окно PGPtray – Key Selection Dialog (рисунок 9.4.1) со списком сертификатов доступных открытых ключей;

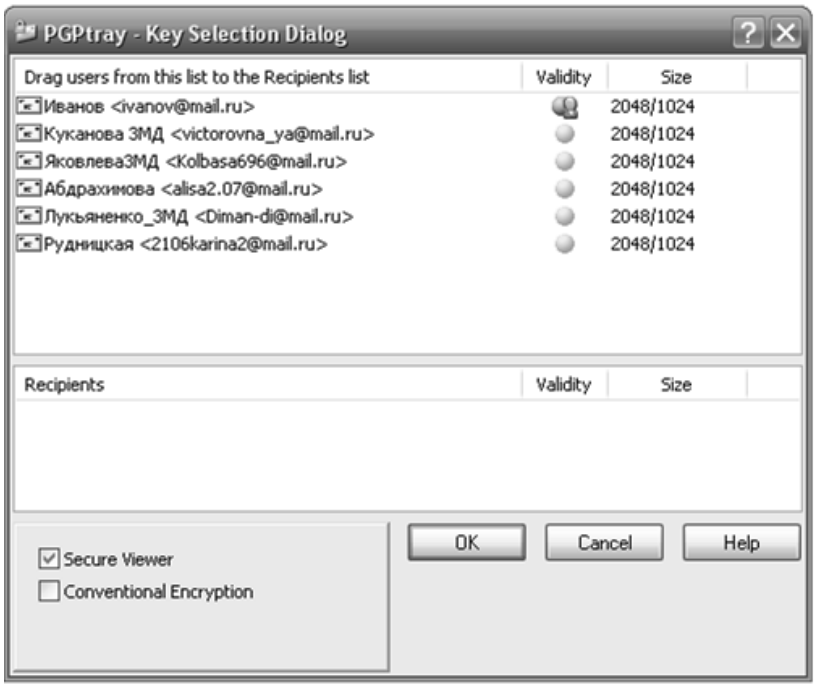


Рисунок 9.4.1 – Список сертификатов доступных открытых ключей

– указать вариант зашифрования Secure Viewer (Безопасный зритель) – если необходимо отправить получателю сообщение, которое и после

расшифрования его закрытым ключом не может быть прочитано, пока не будет выполнена процедура снятия дополнительной защиты;

– в верхней части окна Drag users from this list to the Recipients list (Перенесите пользователя (его ключ) из этого списка в список получателя) дважды щелкнуть по ключу корреспондента. Ключ окажется (рисунок 9.4.2) в нижнем окне (Recipients – Получатели);

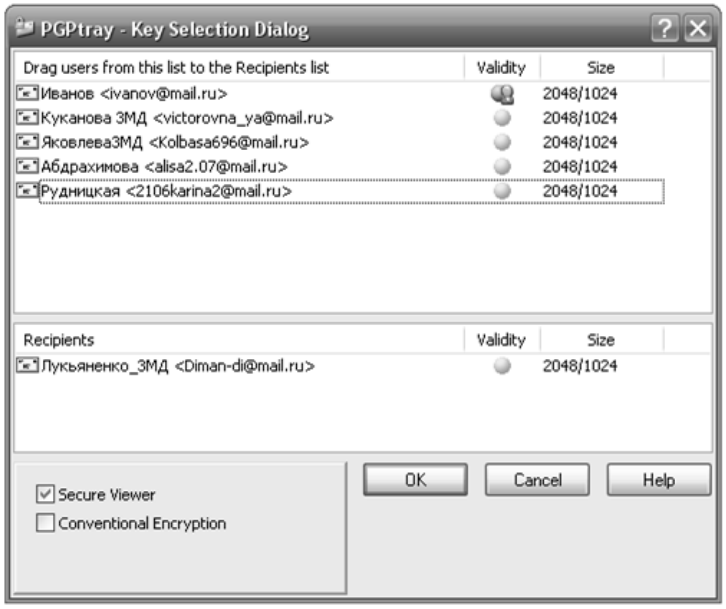


Рисунок 9.4.2 – Перенос ключа пользователя в список получателя

– щелкнуть по кнопке ОК, после чего в буфере обмена появится не открытый, а зашифрованный текст, который имеет вид:

-----BEGIN PGP MESSAGE-----
Version: PGP 8.0.2 - not licensed for commercial use: www.pgp.com

qANQR1DBwU4D/OB/6PTJp7AQCACNPZr0D5+o2xHFSMaGUUmEXbrwIE5rVPCycl
zGaLzOSRRjZ9fm0UzM2OG/ZV/cPy8+t83ot7wePCi43dG02ESmSwc7UHbKR0koqD
Ue0yBSbIhilukt6fCQZ4SJ4ztahx4wGr8e7fPYUtdlm69AhDzvSb2bx7+pjr9BLx
ikXq+Jq6fXl+VQqrSpjkR/ENnnaNorF9gY5k4n3hxglVxmMhEGw1EWcOjGe2UC11
DnwpVB3UP79b2r8bmNPPTeuP0YxrLwVUqnkq2Q8iLUwZdwWt9Ti0p9rP/atvup6
y9l8252cy6Wv0+jI7bcvdK6ZuvegYyhvQMHX8G73BkLKsntzB/4nxKWaaPp6NYaa
lOh9+tnzN5zXt7l1FqIckSaXAQFiGuTLmJaXyhYQ5h2FfdYD/rCZLZwHx32xfYrE
EyC3tVESzhg+sgktF8NEMaz2T1Rzncvmzqwi+lb7E1FETcCOnlHywCtqG0n/NpJF
h8WtC4HfIWYdmzJrvSwAuA4MqOwnQD8heY/8YguxvQlawV+aemn+zp3TBkLx8v
rXQ0qDVE5hUNch2wu7CsUBLfsfPsHfoJEYcMW9VtVqHoF612A2uQJBwFhSKVhxE
kCHZJluYMPot1Mflz+lfsNXJJc4dF8JxR+cJSg35WSWDB4g51R3g4w3vkBoO0j3A
5CDPhz5n0kYBnNIyRJKDAYCE5wzBRWQFu8yjUq7VQqlgALBv5WZLYBeeijzQKY0d
V1WwliQ0/EvCdYDY+KU0pQIDtt1I30VudygNzien
=ty3R
-----END PGP MESSAGE-----

Такой текст можно вставить из буфера обмена (через меню Правка – Вставить или сочетанием клавиш Ctrl +V) в редактор используемой почтовой программы или в специально создаваемый файл для передачи. Фраза sews в шифртексте после слова Comment идентифицирует отправителя (если эта фраза вводилась в закладке General окна настройки PGP Options).

2). Для расшифрования переданного получателю сообщения необходимо:

– открыть текст сообщения в текущем окне;

– щелкнуть по изображению замочка PGPtray на панели задач и в появившемся меню выбрать функцию Current Window (или Clipboard, если получатель предварительно пожелает скопировать шифртекст в буфер обмена), а в ней – Derypt & Verify (расшифровка и сверка). На экране появится окно (рисунок 9.4.3) с указанием на то, что сообщение зашифровано открытым ключом получателя и с предложением ввести свой закрытый ключ (пароль);

Рисунок 9.4.3 – Информация о появлении зашифрованного сообщения

– щелкнуть по кнопке ОК, после чего на экране появится окно (рисунок 9.4.4), напоминающее о том, что сообщение было зашифровано в варианте обеспечения безопасности просмотра и о необходимости соблюдения условий безопасности;

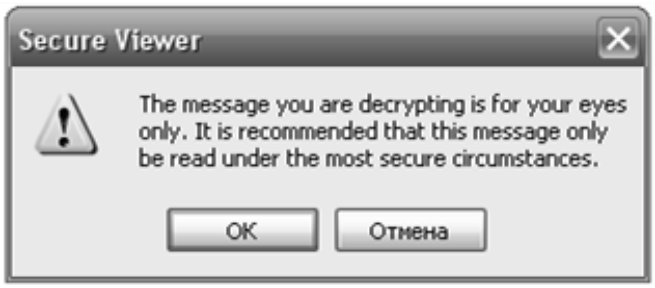
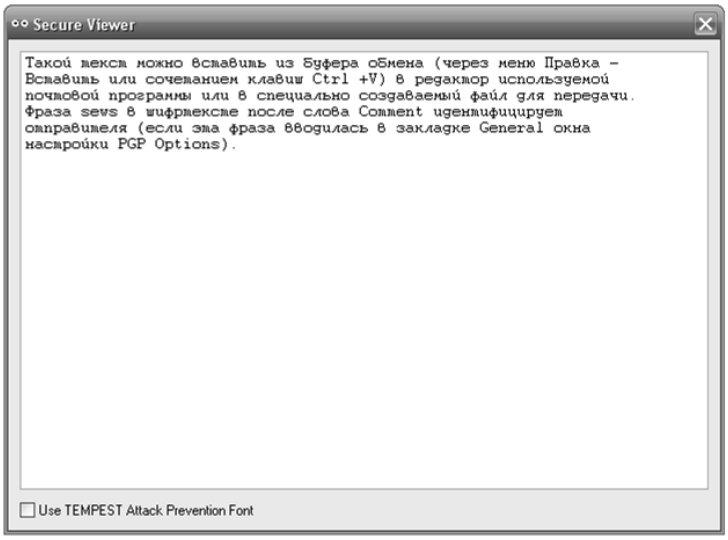


Рисунок 9.4.4 – Информация о необходимости соблюдения условий безопасности

– щелкнуть по кнопке ОК, после чего на экране появится окно Secure Viewer (рисунок 9.4.5а), в котором сообщение остается нечитаемым до момента снятия «галочки» с опции Use TEMPEST Attack Prevention Fort в нижней области окна (рисунок 9.4.5б). После прочтения сообщения с соблюдением мер ограничения доступа достаточно восстановить «галочку», чтобы вновь сделать сообщение нечитаемым;

а)



б)

Рисунок 9.4.5 – Окно с нечитаемым (а) и читаемым (б) сообщением

– закрыть окно Secure Viewer.

Второй способ

1) Для подготовки зашифрованного сообщения в криптосистеме PGP из файла необходимо:

– щелкнуть по изображению замочка PGPtray на панели задач и в появившемся меню выбрать функцию PGPmail. На экране появится меню (рисунок 9.4.6).

Рисунок 9.4.6 – Меню

– в меню щелкнуть по пиктограмме (Конверт с замком – Engrypt (зашифровать). На экране появится окно Select File(s) to Encrypt (Выбор файла для зашифрования) (рисунок 9.4.7);

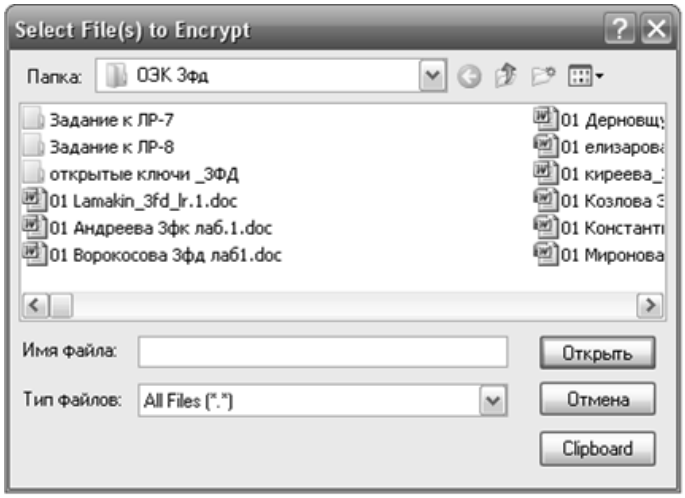


Рисунок 9.4.7 – Выбор файла для зашифрования

– выделить файл, подлежащий зашифрованию и щелкнуть по кнопке Открыть. На экране появится окно PGPtray – Key Selection Dialog со списком сертификатов доступных открытых ключей. Далее, как и при работе с буфером обмена, необходимо выделить ключ получателя и перенести его в окно (Recipients – Получатели);
– щелкнуть по кнопке ОК, после чего в папке, где хранится исходный файл, появится файл типа PGP Encrypted File с именем исходного файла, но с расширением .pgp. Этот файл можно отправлять получателю по электронной почте.

2). Для расшифрования файла получателю необходимо:
– вызвать папку с принятым файлом;
– правой кнопкой мыши щелкнуть по имени файла;
– в появившемся окне выбрать PGP, а в ней – Decrypt & Verify (Расшифровка и Сверка). На экране появится окно (рисунок 9.4.8) с указанием на то, что сообщение зашифровано открытым ключом получателя, и с предложением ввести свой закрытый ключ (пароль);

Рисунок 9.4.8 – Предложение ввести свой закрытый ключ

– после ввода ключа щелкнуть по кнопке ОК, после чего в папке с принятым файлом появится расшифрованный файл.

3) Расшифрование файла можно выполнить и другим способом:
   – щелкнуть по изображению замочка PGPtray на панели задач и в появившемся меню выбрать функцию PGPmail;
   – в появившемся меню щелкнуть по пиктограмме (Конверт с открытым замком – Decrypt/Verify (Расшифровка и Сверка). На экране появится окно Select File(s) to Decrypt/Verify (Выбор файла для расшифрования);
   – выделить файл, подлежащий расшифрованию и щелкнуть по кнопке Открыть. На экране появится окно (рисунок 9.4.8) с предложением ввести свой закрытый ключ (пароль);
   – после ввода ключа щелкнуть по кнопке ОК, после чего в папке с принятым файлом появится расшифрованный файл.

Б. Содержание отчета

1) После того как вы изучите содержание пунктов работы и потренируетесь в их выполнении:
   – зашифруйте произвольный текстовый файл первым способом для одного из студентов группы и отправьте его по электронной почте этому студенту (он должен работать за другим компьютером);
   – зашифруйте другой текстовый файл вторым способом для другого студента группы и отправьте его по электронной почте этому студенту (он должен работать за другим компьютером);
   – получите от других студентов по электронной почте зашифрованные ответные файлы и расшифруйте их первым и вторым способом.
2) Получите от преподавателя зашифрованный для вас файл, расшифруйте его, ознакомьтесь с его содержанием и сохраните расшифрованный вариант текста в виде файла MS Word (например, *иванов_3_мд_лаб4.doc*).
3) Зашифруйте этот файл одним из способов для преподавателя и зашифрованную версию файла прикрепите к письму и по электронной почте отправьте преподавателю.

9.5 Лабораторная работа № 5. Работа с криптосистемой PGP: формирование и сверка электронной цифровой подписи

Цель работы: Освоить технологию применения системы PGP. Научиться формировать и проверять ЭЦП.

А. Порядок выполнения лабораторной работы

Электронной цифровой подписью можно заверять:

– свои сообщения для передачи корреспондентам;

– открытые ключи корреспондентов в целях их удостоверения;

– свой открытый ключ перед передачей в сертификационный (удостоверяющий) центр.

Для подписания сообщения необходимо выполнить три предварительные действия:

– найти папку хранения сообщения (рисунок 9.5.1) и выделить подлежащий подписанию файл;

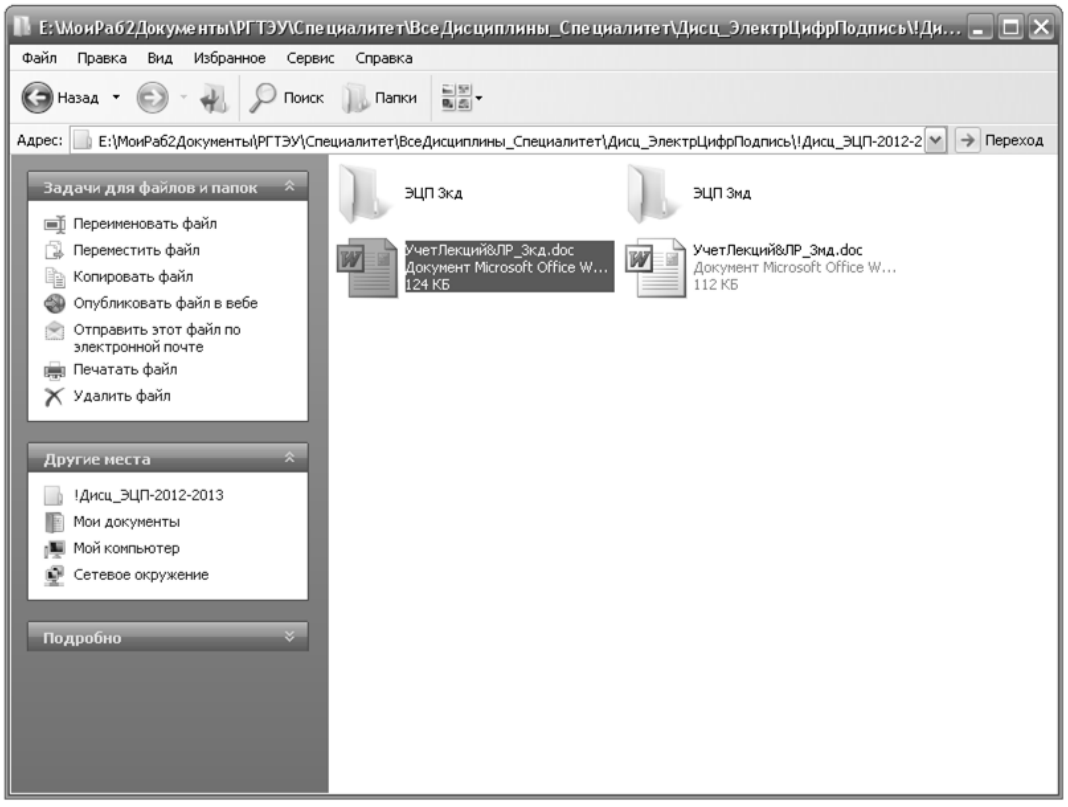


Рисунок 9.5.1 – Выделенный файл, подлежащий подписанию

– щелкнуть правой кнопкой по выделенному файлу и в появившемся окне выбрать строку PGP, а в ней Sign. На экране появится окно PGPshell – Enter Passphrase (рисунок 9.5.2) с именем ключевой пары отправителя и окном для ввода закрытого ключа (пароля);

– в окно Enter passphrase for above key ввести основной (закрытый) ключ отправителя;

– в окне с опциями Detached Signature (Отдельная Подпись) и Text Output (Вывод текста) выбрать вариант формирования результирующего файла и нажать кнопку ОК.

Рисунок 9.5.2 – Имя ключевой пары отправителя

Вариант 1. Подписание сообщения в варианте Detached Signature (Отдельная Подпись) сопровождается формированием файла подписи, отдельного от файла исходного сообщения.

1) Для подписания (после выполнения трех предварительных действий) необходимо:
   – поставить «галочку» в окне опции Detached Signature (Отдельная Подпись);
   – щелкнуть по кнопке ОК, после чего в папке, где хранится исходный файл (рисунок 9.5.3), появится файл с именем исходного файла, но с расширением .sig.

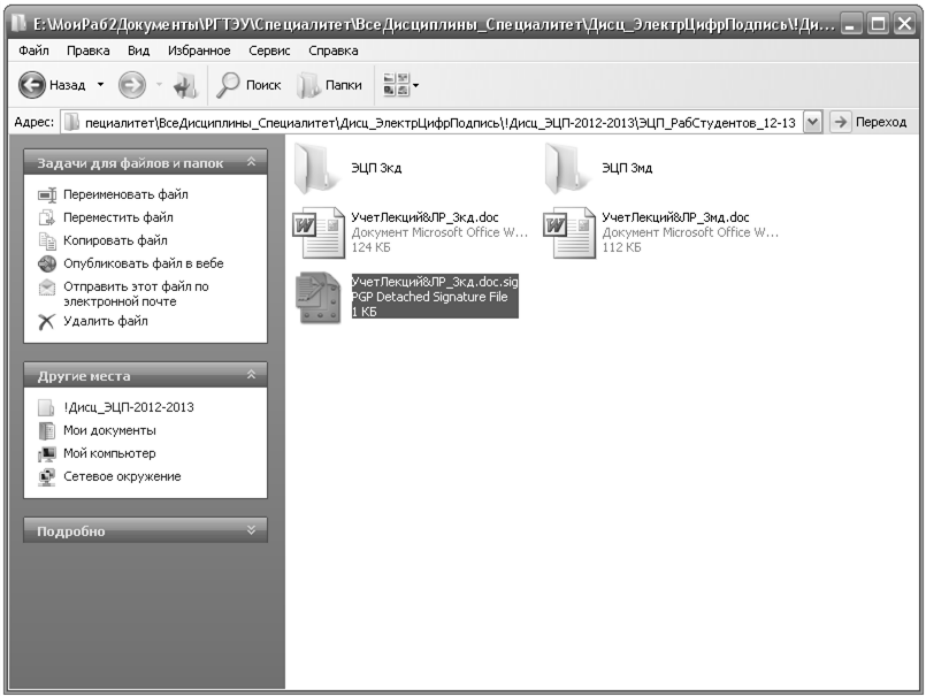


Рисунок 9.5.3 – Имя исходного файла

Этот файл электронной цифровой подписи имеет вид:

% ? - @ Ж в Р С f ч ™] Г g j н ь к э·I ” Т b | I s B G й J 9 ц V %· j 1 ц Э @ k й ”м ” D # щ щ

При попытке открыть этот файл у получателя на экран будет выведено окно (рисунок 9.5.4) Select Signed File (Выбор подписанного файла) с предложением открыть файл с исходным сообщением.

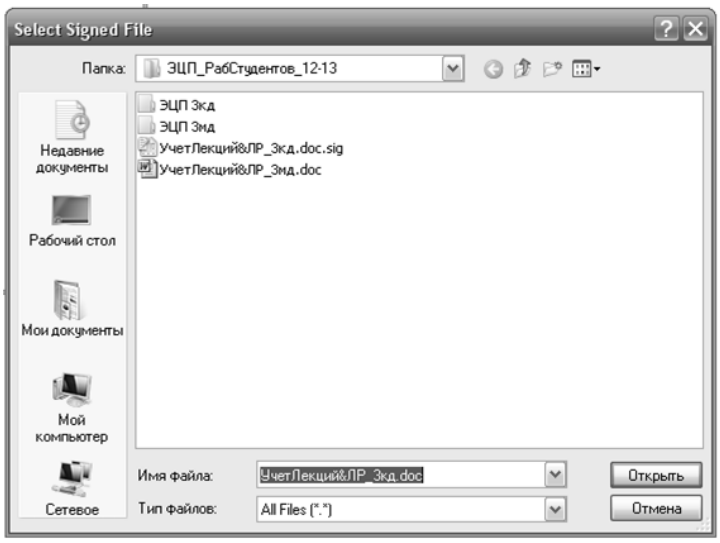


Рисунок 9.5.4 – Выбор подписанного файла

Это означает, что в этом варианте подписания файл с расширением .sig нужно отправлять получателю по электронной почте вместе с файлом исходного сообщения. Получатель, имея эти файлы, может ограничиться только знакомством с исходным сообщением.

2) При желании получить подтверждение подлинности и целостности принятого сообщения (сверки подписи) следует найти и открыть файл с расширением .sig, для чего:
   – выделить файл с расширением .sig;
   – щелкнуть правой кнопкой по выделенному файлу и в появившемся окне выбрать строку PGP, а в ней Verify Signature.

Если файл с исходным сообщением и файл с электронной цифровой подписью не фальсифицированы, на экране появится окно PGPlog (рисунок 9.5.5);

Рисунок 9.5.5 – Признак целостности исходного файла и ЭЦП

Если в исходное сообщение или в файл с подписью при передаче по сети были внесены изменения, на экране появится окно PGPlog (рисунок 9.5.6), с признаком некондиционности сообщения и ЭЦП.

Рисунок 9.5.6 – Признак некондиционности сообщения и ЭЦП

Если у получателя нет ключа отправителя, то на экране появится окно PGPlog (рисунок 9.5.7) с сообщением Unklown Key (Неизвестный ключ) и ссылкой на идентификатор ID его владельца.

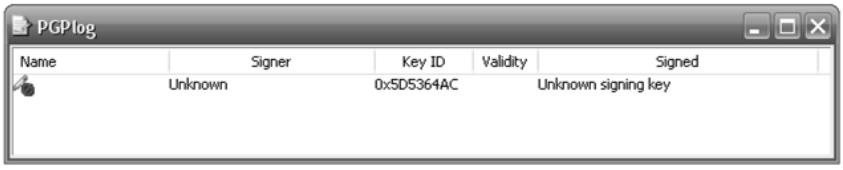


Рисунок 9.5.7 – Неизвестный ключ

В этом случае получателю необходимо найти по ID отправителя, узнать место хранения его открытого ключа, импортировать отсутствующий ключ и вновь проверить авторство и подлинность принятого сообщения.

Вариант 2. Подписание сообщения в варианте Text Output (Вывод текста) сопровождается формированием файла, в котором исходное сообщение и подпись представлены в зашифрованном виде.

1) Для подписания (после выполнения трех предварительных действий) необходимо:
   – поставить «галочку» в окне опции Text Output (Вывод текста);
   – щелкнуть по кнопке ОК, после чего в папке, где хранится исходный файл (рисунок 9.5.8), появится файл с именем исходного файла, но с расширением .asc.

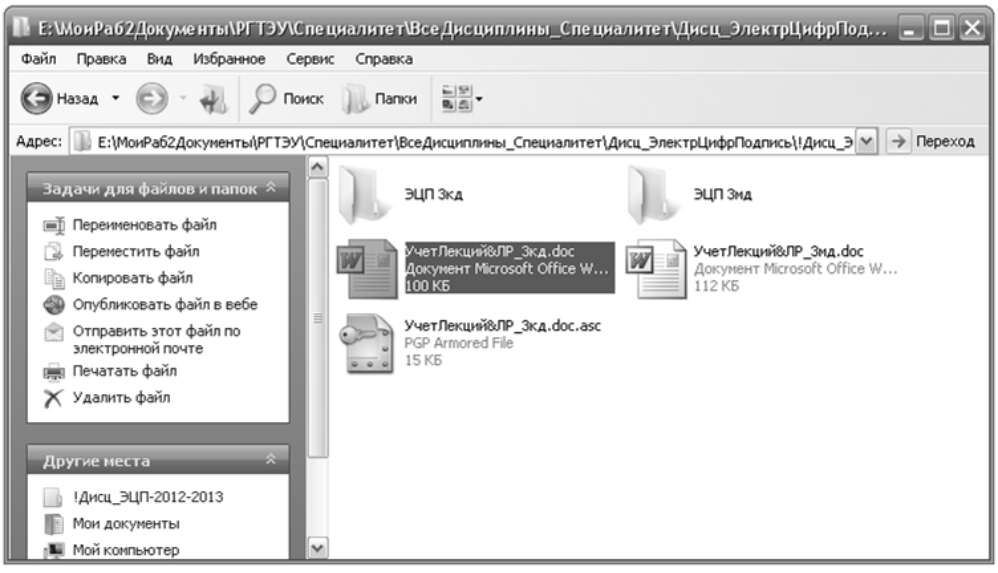


Рисунок 9.5.8 – Имя исходного файла с расширением .asc

Этот файл имеет вид:

BEGIN PGP MESSAGE
Version: PGP 8.0.2
Comment: sews

Зашифрованный текст сообщения и электронной цифровой подписи

END PGP MESSAGE

Этот файл можно отправлять получателю по электронной почте.

2) Для раскрытия файла с расширением .asc получателю необходимо:
   – вызвать на экран папку и выделить файл с расширением .asc;
   – щелкнуть правой кнопкой по выделенному файлу и в появившемся окне выбрать строку PGP, а в ней Verify Signature.

Если файл с электронной цифровой подписью не фальсифицирован, на экране появится окно PGPlog (рисунок 9.5.9), а в папке рядом с принятым файлом появится файл с исходным сообщением.

Рисунок 9.5.9 – Признак кондиционности ЭЦП

Фраза Invalid Key в разделе Signed окна PGPlog свидетельствует о том, что авторство и целостность сообщения подтверждены, но сама подпись отправителя никем не заверена, то есть отправитель не имеет гарантов. Об этом же свидетельствует серый шарик в разделе Validity этого же окна.

Вариант 3. Подписание сообщения в варианте с обеими опциями (Detached Signature (Отдельная Подпись) и Text Output (Вывод текста) сопровождается формированием файла электронной цифровой подписи, в читаемой части которого можно увидеть ссылку на имя отправителя (если оно вводилось при настройке криптосистемы).

1) Для подписания (после выполнения трех предварительных действий) необходимо:
   – поставить «галочки» в окнах опций Detached Signature (Отдельная Подпись) и Text Output (Вывод текста);
   – щелкнуть по кнопке ОК, после чего в папке где хранится исходный файл (см. рисунок 9.5.3), появится файл с именем исходного файла, но с расширением .sig.

Этот файл имеет вид:

BEGIN PGP SIGNATURE
Version: PGP 8.0.2 Comment: sews
iQA/AwUAQMllptFm95kAXYFnEQKJkQCfUwtJa5mCZ7gOl 7gowwds8zA6cM4 Ao Jiz
=XexT
END PGP SIGNATURE

Так же, как в варианте 1, этот файл необходимо отправлять получателю вместе с файлом исходного сообщения.

Таким образом, при подписании файла следует помнить, что если не выполнены правила передачи, то получатель может и не прочитать сообщение.

Вариант 4. Другой способ подписания своего файла.

1) Для подписания необходимо:

– вызвать на экран текст исходного сообщения и скопировать его в буфер;

– щелкнуть по изображению замочка PGPtray на панели задач и в появившемся меню выбрать функцию Clipboard, а в ней Sign. На экране появится окно PGPtray – Enter Passphrase (рисунок 9.5.10) с именем секретного ключа отправителя и окном для ввода закрытого ключа (пароля);

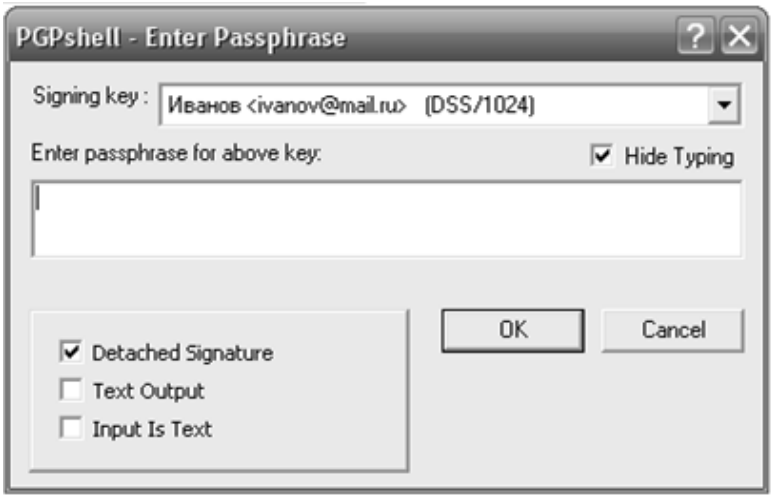


Рисунок 9.5.10 – Окно для ввода закрытого ключа

– после ввода ключа щелкнуть по кнопке ОК, после чего в буфере обмена будет сформирован файл, который имеет вид:

```
        BEGIN PGP SIGNED MESSAGE
Hash: SHA1
Исходное сообщение
        BEGIN PGP SIGNATURE
Version: PGP 8.0.2
Comment: sews
iQA/AwUBQMlspdFm95kAXYFnEQKLyQCgjJL+M6GEvo
RZ2EBIBbUnZtpyg3EAoK6t
=JCwl
        END PGP SIGNATURE
```

– скопировать файл из буфера обмена в обычный файл с расширением .doc и переслать его получателю по электронной почте.

Таким образом, способ подписания сообщения через буфер обмена дает возможность получить для передачи по электронной почте один файл, в котором объединены исходное сообщение и электронная цифровая подпись.

2) Открытие такого файла у получателя дает возможность прочитать сообщение сразу. Для сверки подписи необходимо:
   – скопировать файл в буфер обмена;
   – щелкнуть по изображению замочка PGPtray на панели задач и в появившемся меню выбрать функцию Clipboard, а в ней Decrypt & Verify (расшифровка и сверка). На экране появится окно Text Viewer с текстом (рисунок 9.5.11).

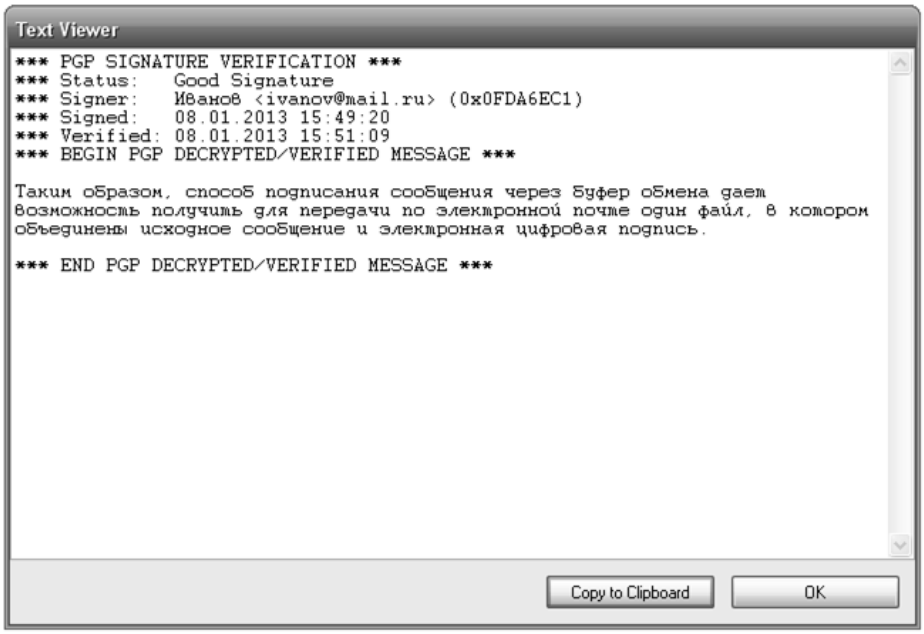


Рисунок 9.5.11 – Окно Text Viewer с текстом

На этом рисунке строки текста означают:

*** PGP SIGNATURE VERIFICATION ***	
*** Status: Good Signature	Статус подписи: хороший
*** Signer: Иванов <ivanov@mail.ru> (0x0FDA6EC1)	Подписывающее лицо
*** Signed: 08.01.2013 15:49:20	Подписано: дата, время
*** Verified: 08.01.2013 15:51:09	Проверено: дата, время
*** BEGIN PGP DECRYPTED/ VERIFIED MESSAGE ***	Начало проверенного сообщения
Исходное сообщение	
*** END PGP DECRYPTED/VERIFIED MESSAGE ***	Конец проверенного сообщения

Текст свидетельствует о том, что целостность и авторство проверкой подтверждены. Вместе с тем ключ подписывающего лица не имеет доверительной подписи.

Сохранить этот текст можно, записав его в буфер с помощью кнопки Copy to Clipboard, а затем переписать в файл. Щелчок по кнопке ОК приведет к закрытию окна и потере текста.

Б. Содержание отчета

1) После того как вы изучите содержание пунктов работы и потренируетесь в их выполнении:
   – подпишите своей ЭЦП произвольный текстовый файл первым вариантом и отправьте файлы (текстовый и подписи) по электронной почте одному из студентов группы (он должен работать за другим компьютером);
   – подпишите своей ЭЦП другой текстовый файл вторым (третьим, четвертым) вариантом и отправьте его по электронной почте другому студенту (он должен работать за другим компьютером);
   – получите от других студентов по электронной почте подписанные ими ответные файлы и сверьте их разными вариантами.
2) Подпишите своей ЭЦП произвольный текстовый файл первым вариантом, а затем внесите изменения в исходный файл. Отправьте этот файл вместе с файлом ЭЦП по электронной почте одному из студентов группы и убедитесь у него, что сверка не прошла.
3) Создайте произвольный текстовый файл для преподавателя, сохраните его в формате MS Word (например, *иванов_3_мд_лаб5А.doc*). Подпишите этот файл своей ЭЦП любым из вариантов и отправьте по электронной почте преподавателю для сверки.
4) Получите от преподавателя подписанные для вас два файла и сверьте их. Тот файл, для которого сверка не прошла, удалите. Ознакомьтесь с содержанием другого файла (для которого сверка прошла), сохраните его в формате MS Word (например, *иванов_3_мд_лаб5Б.doc*).
5) Подпишите своей ЭЦП этот файл любым из вариантов и по электронной почте отправьте преподавателю для сверки.

9.6 Лабораторная работа № 6. Работа с криптосистемой PGP: подписание ключей

Цель работы: Освоить технологию применения системы PGP. Научиться подписывать чужие и свои ключи

А. Порядок выполнения лабораторной работы

Подписание чужого ключа

1) Для подписания (удостоверения) чужого ключа необходимо:
   – щелкнуть по значку PGPtrey и выбрать из меню PGPkeys;
   – выделить подлежащий подписанию открытый ключ;
   – на инструментальной панели выбрать пиктограмму Рука с карандашом (Sign the selected item). Появится окно PGP Sign Key (рисунок 9.6.1):

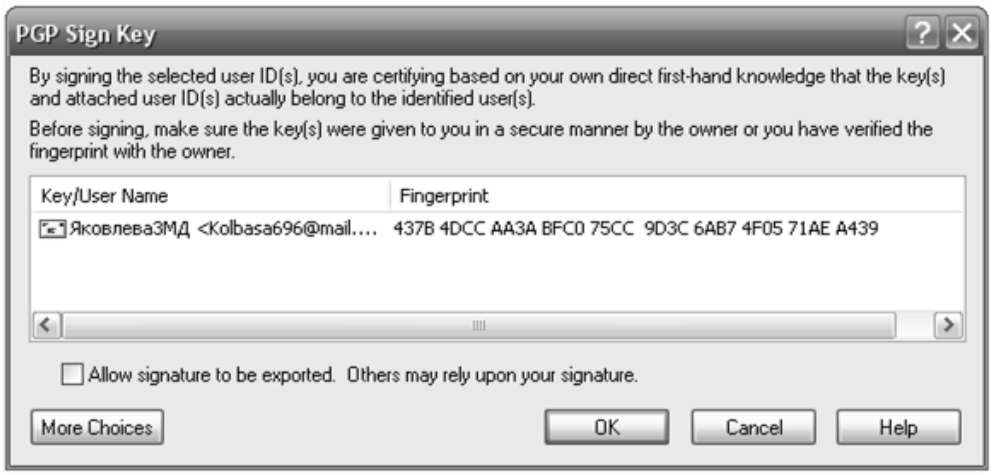


Рисунок 9.6.1 – Подписание чужого ключа

Текст предупреждает: «Удостоверяя ID (s) отобранного пользователя, вы на основании своего личного убеждения подтверждаете, что ключ (и) и приложенный к нему ID (s) фактически принадлежат идентифицированному пользователю (ям). Перед подписанием удостоверьтесь, что ключ (и) давался вам владельцем в безопасной манере или вы проверили слепок его ключа».

2) Если вы убеждены в том, что ключ действительно принадлежит лицу, которое вам известно, то:

– проставьте галочку в окне Allow signature to by exported. Others may rely your signature (Разрешите экспорт подписи. Другие могут полагаться на вашу подпись) и нажмите OK. На экране появится окно (рисунок 9.6.2) Enter Passphrase for Selected key (Ввод пароля выбранного ключа);

Рисунок 9.6.2 – Ввод пароля выбранного ключа

– выбрать с помощью скроллинга окна Signing key (Закрытый ключ) свой ключ для подписания, ввести его пароль в окно Passphrase of signing key и нажать кнопку OK. Параметр Validity подписанного ключа в окне PGPkeys приобретает вид зеленого кружка (рисунок 9.6.3), то есть ключ становится достоверным.

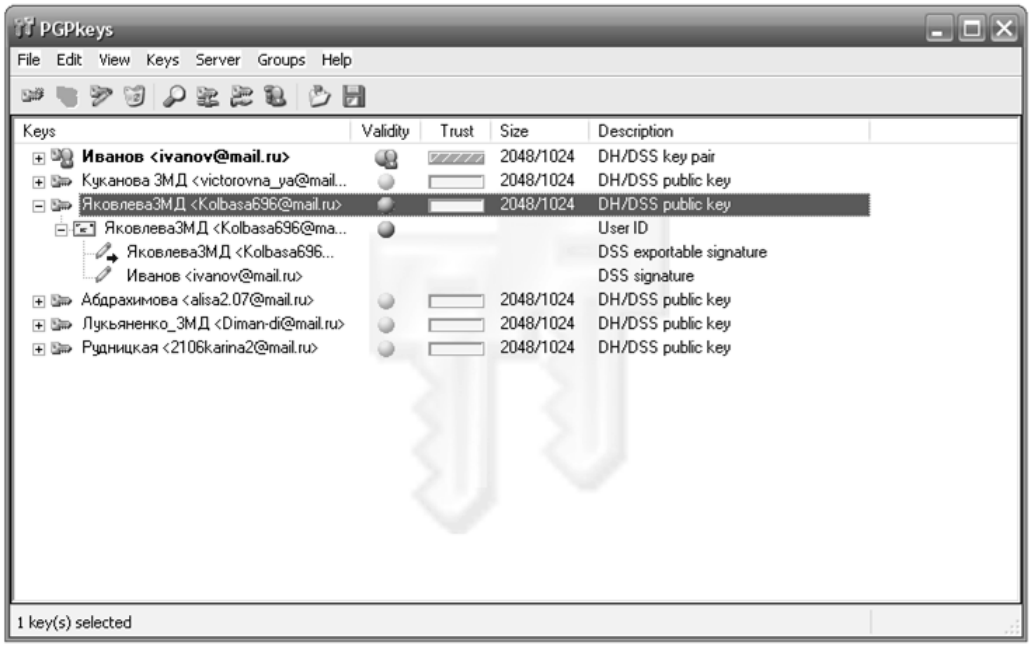


Рисунок 9.6.3 – Признаки достоверности ключа

Тот факт, что ключ удостоверен вами, отображается в окне PGPkeys строчкой экспортируемой со связки подписью – с вашим именем и вашим адресом электронной почты. Далее подписанный ключ целесообразно передать тем корреспондентам, которые заинтересованы в удостоверении данного ключа.

Если окно Enter Passphrase for Selected key (Ввод пароля выбранного ключа) вместо отображенного на рисунке 9.6.2 имеет вид, представленный на рисунке 9.6.4, это свидетельствует о том, что пароль выбранного ключа был уже задействован для какой-то операции, и в настоящее время cached (находится в активной памяти и будет задействован без дополнительного вызова).

Рисунок 9.6.4 – Признак того, что пароль выбранного ключа был уже задействован для какой-то операции

Щелчок по кнопке ОК завершит подписание выделенного ключа.

Подписание своего ключа

Необходимость подписания своего ключа возникает в том случае, когда по той или иной причине пользователь создает себе новую ключевую пару, но хочет продолжить переписку со своими абонентами. В этом случае, чтобы у абонентов не возникли какие-либо подозрения, он подписывает свою новую ключевую пару своим старым ключом. После чего распространяет свой новый подписанный открытый ключ среди абонентов.

Для подписания своего ключа необходимо:

– щелкнуть по изображению замочка PGPtray на панели задач и в появившемся меню выбрать функцию PGPkeys. На экране появится одноименное окно (рисунок 9.6.5);

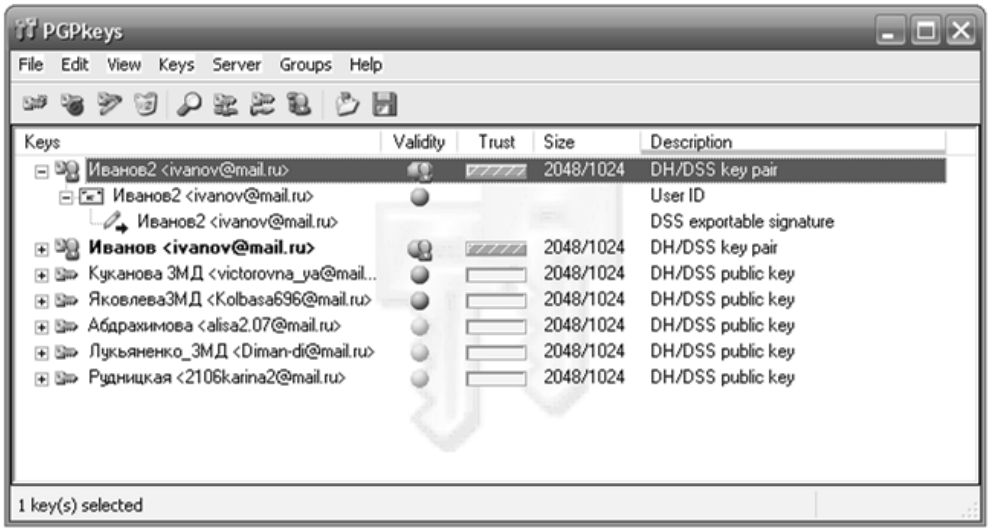


Рисунок 9.6.5 – Подписание своего ключа

– выделить ключ, подлежащий подписанию, после чего выбрать из меню Keys, а в ней Sign. На экране появится окно PGP Sign Key (рисунок 9.6.6), в котором есть разъяснение о необходимости убедиться в правильности цифрового отпечатка вашего ключа;

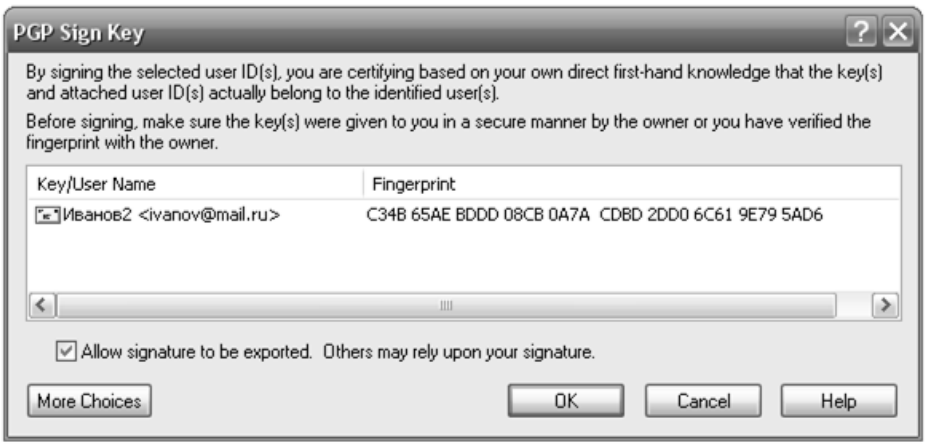


Рисунок 9.6.6 – Предупреждение

– поставить галочку в поле Allow signature to be exportd (Разрешить экспорт подписи вместе с файлом) и нажать ОК;
– в появившемся окне (рисунок 9.6.7) ввести пароль и нажать кнопку ОК.

Рисунок 9.6.7 – Окно для ввода пароля

Подписанный ключ можно передавать в сертифицирующий центр, а открытые ключи – абонентам.

Б. Содержание отчета

1) После того как вы изучите содержание пунктов работы и потренируетесь в их выполнении:
   – подпишите (удостоверьте) открытый ключ одного из студентов вашей группы;
   – экспортируйте этот ключ в текстовый вид и отправьте его в формате MS Word по электронной почте преподавателю (имя файла должно быть, например, *иванов_3_мд_лаб6А.doc*).
2) Сгенерируйте НОВУЮ ключевую пару, выбрав в качестве ее имени свою фамилию с добавлением цифры «2», а в качестве адреса свой реальный адрес электронной почты. Запомните свою НОВУЮ КЛЮЧЕВУЮ ФРАЗУ. После чего:
   – подпишите (удостоверьте) свою новую ключевую пару с помощью своего старого ключа;
   – экспортируйте свой новый открытый ключ в текстовый вид и отправьте его в формате MS Word по электронной почте преподавателю (имя файла должно быть, например, *иванов_3_мд_лаб6Б.doc*).

9.7 Лабораторная работа № 7. Работа с криптосистемой PGP: зашифрование и расшифрование сообщений различными способами

Цель работы: Освоить технологию применения системы PGP. Научиться зашифровывать и расшифровывать сообщения.

А. Порядок выполнения лабораторной работы

Вариант 3. Conventional Encryption (Обычное зашифрование) – если необходимо просто зашифровать и передать сообщение корреспонденту, который знает ключевую фразу. Другими словами, вариант реализован по схеме симметричного шифрования и не связан с использованием асимметричных ключей сгенерированных ключевых пар.

1) Для зашифрования по этому варианту необходимо:

– после копирования сообщения в буфер и вызова через меню PGPtray функции Clipboard, а в ней – Engrypt (зашифровать) в появившемся окне PGPtray – Key Selection Dialog выбрать опцию Conventional Encryption (поставить «галочку»). Окно видоизменится (рисунок 9.7.1);

Рисунок 9.7.1 – Предложение зашифровать сообщение

– щелкнуть по кнопке ОК, после чего на экране появится окно (рисунок 9.7.2) с предложением ввести ключ для зашифрования сообщения, помещенного в буфер. Это может быть любой ключ, о котором заранее договорились отправитель и получатель;

Рисунок 9.7.2 – Предложение ввести ключ для зашифрования сообщения

– после ввода ключа щелкнуть по кнопке ОК, после чего в буфере обмена появится зашифрованный текст.

Этот текст можно отправлять получателю средствами почтовой программы, а можно записать в файл для последующей передачи.

2) Расшифрование принятого получателем сообщения осуществляется в следующем порядке:
   – открыть текст в текущем окне;
   – щелкнуть по изображению замочка PGPtray на панели задач и в появившемся меню выбрать функцию Current Window (или Clipboard, если получатель предварительно пожелал скопировать шифртекст в буфер обмена), а в ней – Derypt & Verify (расшифровка и сверка). На экране появится окно (рисунок 9.7.3, левое) с предложением ввести ключевое слово. Если после ввода ключа и щелчка по кнопке ОК на экране появится это же окно, но с сообщением о некорректности ключа (рисунок 9.7.3, правое), ввод ключевого слова надо повторить.

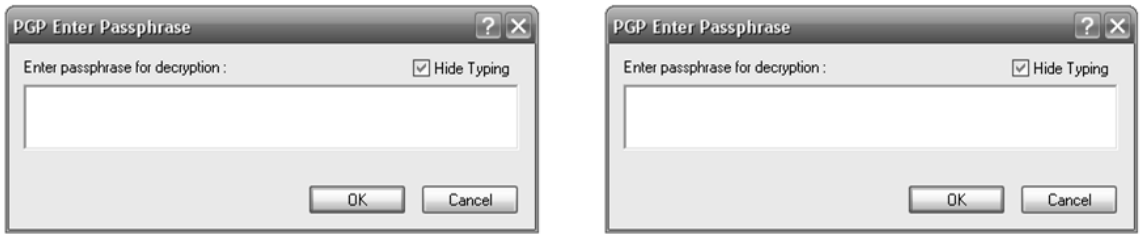


Рисунок 9.7.3 – Окна для введения ключевого слова

Если ключ введен правильно, на экране появится окно Text Viewer (рисунок 9.7.4) с расшифрованным сообщением.

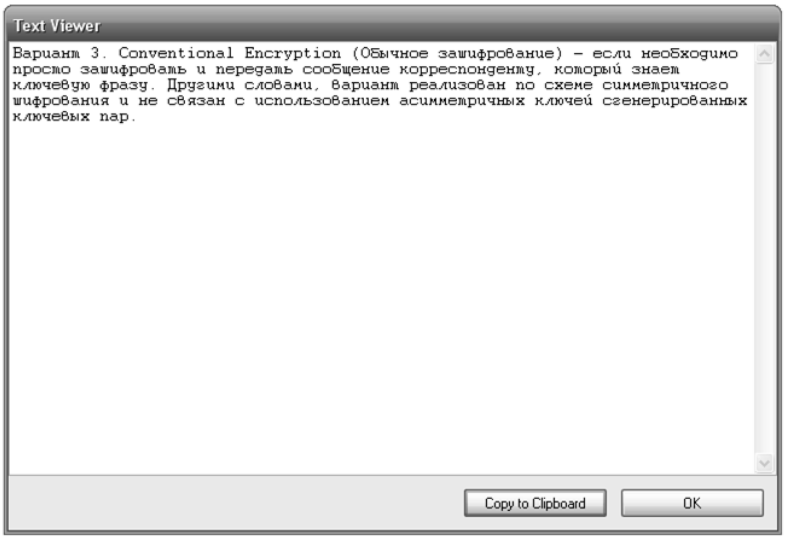


Рисунок 9.7.4 – Расшифрованное сообщение

Сохранить расшифрованное сообщение можно, записав его в буфер с помощью кнопки Copy to Clipboard, а затем переписать в файл. Щелчок по кнопке ОК приведет к потере расшифрованного сообщения.

Вариант 4. Для зашифрования сообщения из файла необходимо:

– вызвать на экран папку с файлом исходного сообщения и выделить этот файл (рисунок 9.7.5);

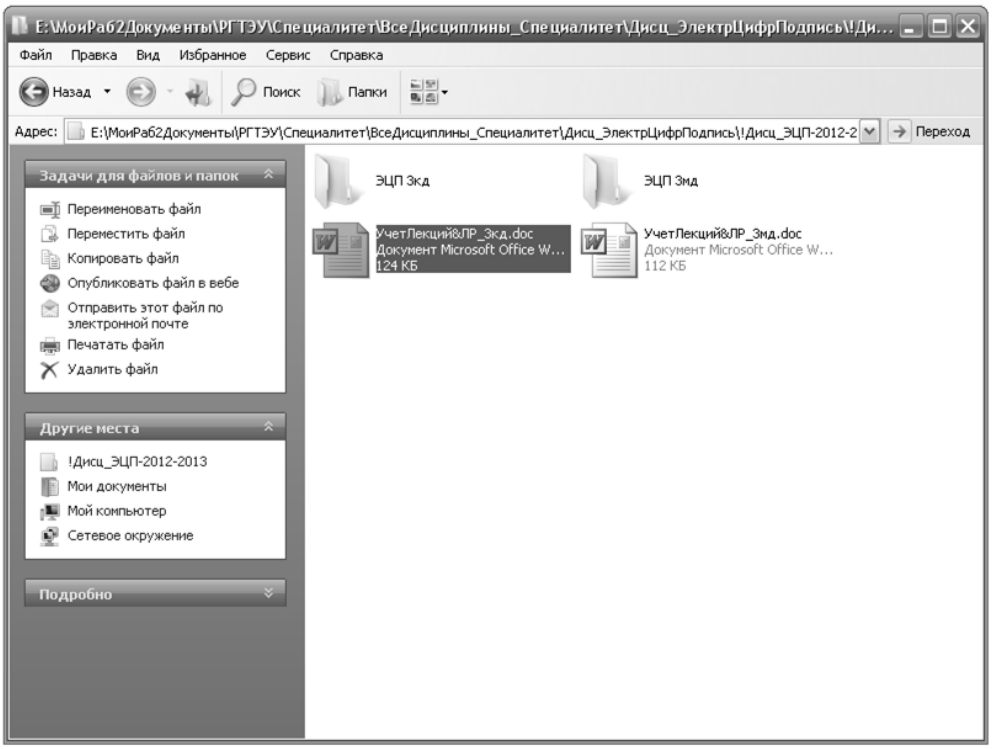


Рисунок 9.7.5 – Выделение файла исходного сообщения

– щелкнуть правой кнопкой мышки по строчке с выделенным файлом. В появившемся меню выбрать строчку PGP, а в ней Encript;

– на экране появится окно PGPshell – Key Selection Dialog (рисунок 9.7.6) со списком сертификатов доступных открытых ключей;

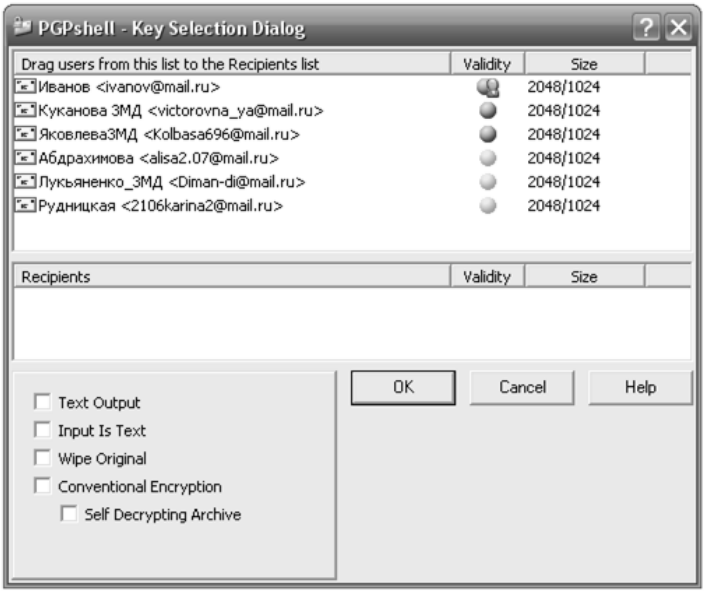


Рисунок 9.7.6 – Список сертификатов доступных открытых ключей

– оставить пустыми места для «галочек» опций (внизу слева) и дважды щелкнуть по ключу предполагаемого корреспондента. Ключ переместится в окно Recipients – Получатели;
– щелкнуть по кнопке ОК, после чего в папке, где хранится исходный файл, появится файл типа PGP Encrypted File с именем исходного файла, но с расширением .pgp. Этот файл можно отправлять получателю.

Вариант 5. Зашифрование по этому варианту выполняется при необходимости передать корреспонденту сообщение в кодах ASCII.

1) Для зашифрования необходимо:
– выполнить первые три операции варианта 4;
– поставить «галочку» в окне опции Text Output (Вывод текста) и дважды щелкнуть по ключу предполагаемого корреспондента. Ключ переместится в окно Recipients – Получатели;
– щелкнуть по кнопке ОК, после чего в папке, где хранится исходный файл, появится файл типа PGP Armored File с именем исходного файла, но с расширением .asc. Этот файл можно отправлять получателю по электронной почте.
2) Расшифрование выполняется по любому из способов, изложенных ранее. Файл, зашифрованный по варианту 5, по объему больше почти на треть, а поэтому вариант 5 применяется только в случаях, когда корреспондент сам заказывает именно такое представление.

Вариант 6. Зашифрование по этому варианту выполняется, если необходимо удалить передаваемое сообщение со своего компьютера.

1) Для зашифрования необходимо:
– выполнить первые три операции варианта 4;
– поставить «галочку» в окне опции Wipe Original (Удалить оригинал) и дважды щелкнуть по ключу предполагаемого корреспондента. Ключ переместится в окно Recipients – Получатели;
– щелкнуть по кнопке ОК, после чего на экране появится окно (рисунок 9.7.7) PGP Warning (Предупреждение).

Рисунок 9.7.7 – Предупреждение

В окне вопрос: Вы уверены, что хотите зашифровать сообщение и удалить оригинал, хотя прочитать зашифрованное сообщение не можете?

Если щелкнуть по кнопке ДА, то в папке, где хранится исходный файл, сам исходный файл будет удален, но появится файл типа PGP Encrypted File с именем исходного файла, но с расширением .pgp. Этот файл можно отправлять корреспонденту.

2) Расшифрование переданного файла получателем выполняется по любому из способов, изложенных ранее.

Вариант 7. Зашифрование по этому варианту выполняется по схеме симметричного зашифрования и не связано с использованием асимметричных ключей сгенерированных ключевых пар.

1) Для зашифрования необходимо:

– выполнить первые три операции варианта 4;

– поставить «галочку» в окне опции Conventional Encryption (Обычное зашифрование), после чего окно PGPshell – Key Selection Dialog примет вид, изображенный на рисунке 9.7.8;

Рисунок 9.7.8 – Выделение опции «Обычное зашифрование»

– щелкнуть по кнопке ОК, после чего на экране появится окно (рисунок 9.7.9) PGPshell – Enter Passphrase с предложением дважды ввести оговоренный корреспондентами ключ;

Рисунок 9.7.9 – Предложение дважды ввести оговоренный корреспондентами ключ

– после ввода ключей и нажатия кнопки ОК в папке, где хранится исходный файл, появится файл типа PGP Encrypted File с именем исходного файла, но с расширением pgp. Этот файл можно отправлять корреспон-

денту. Объем этого файла будет меньше, чем во всех ранее описанных вариантах, но это за счет применения другого алгоритма.

2) Для расшифрования переданного получателю зашифрованного этим вариантом файла необходимо:
   – вызвать папку с принятым файлом и выделить его;
   – правой кнопкой мыши щелкнуть по имени файла;
   – в появившемся окне выбрать PGP, а в ней Decrypt & Verify (Расшифровка и Сверка). На экране появится окно PGP Enter Passphrase (рисунок 9.7.10, левое) с предложением ввести согласованный с отправителем ключ;

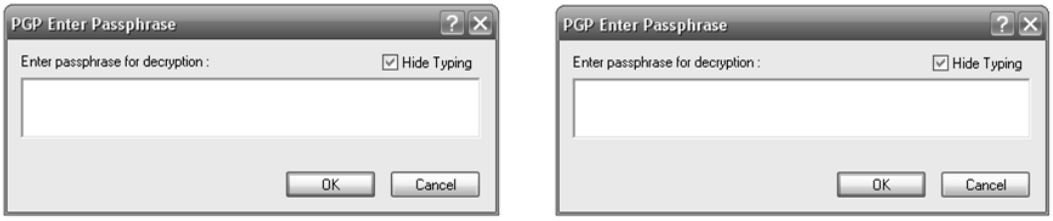


Рисунок 9.7.10 – Предложения о необходимости введения ключей

   – если после ввода ключа и нажатия кнопки ОК на экране появится окно PGP Enter Passphrase (рисунок 9.7.10, правое), то это значит, что введен неверный ключ, а ввод необходимо выполнить заново;
   – если после ввода ключа и нажатия кнопки ОК на экран будет выведено окно Enter output filename (рисунок 9.7.11) с предложением определить имя и место хранения расшифрованного файла, то это значит, что файл расшифрован и его надо сохранить или отменить запоминание до следующего удобного момента.

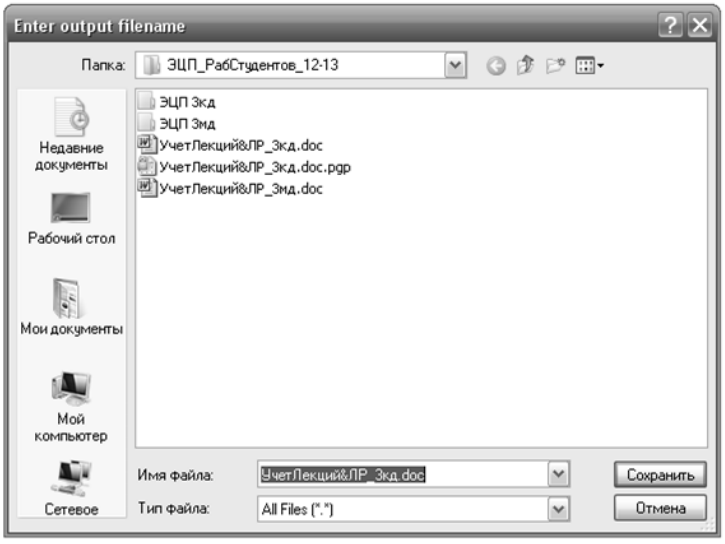


Рисунок 9.7.11 – Предложение определить имя и место хранения расшифрованного файла

Файл при этом останется зашифрованным.

Вариант 8. Зашифрование по этому варианту выполняется при необходимости передать зашифрованное сообщение корреспонденту, у которого на компьютере нет PGP.

1) Для зашифрования необходимо:

– выполнить первые три операции варианта 4;

– поставить «галочку» в окне опции Self Decrypting Archive (Самораспаковывающийся архив), после чего программа сама проставит галочку в опции Conventional Encryption (Обычное зашифрование), а окно PGPshell – Key Selection Dialog примет вид, изображенный на рисунке 9.7.12;

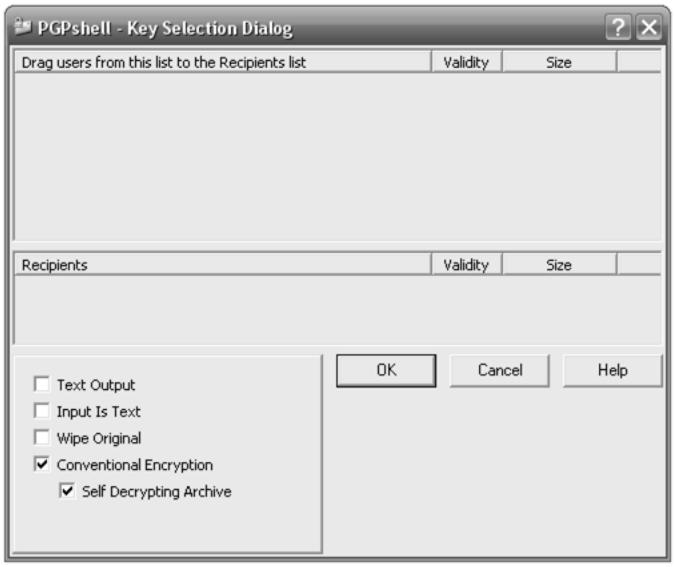


Рисунок 9.7.12 – Признак «Обычное зашифрование»

– щелкнуть по кнопке OK, после чего на экране появится окно (рисунок 9.7.13) PGPshell – Enter Passphrase с предложением дважды ввести оговоренный с корреспондентом ключ;

Рисунок 9.7.13 – Предложение дважды ввести оговоренный с корреспондентом ключ

– после ввода ключей и нажатия кнопки ОК на экране появится окно (рисунок 9.7.14) Please confirm the file name of this SDA (Пожалуйста, подтвердите имя этого файла SDA).

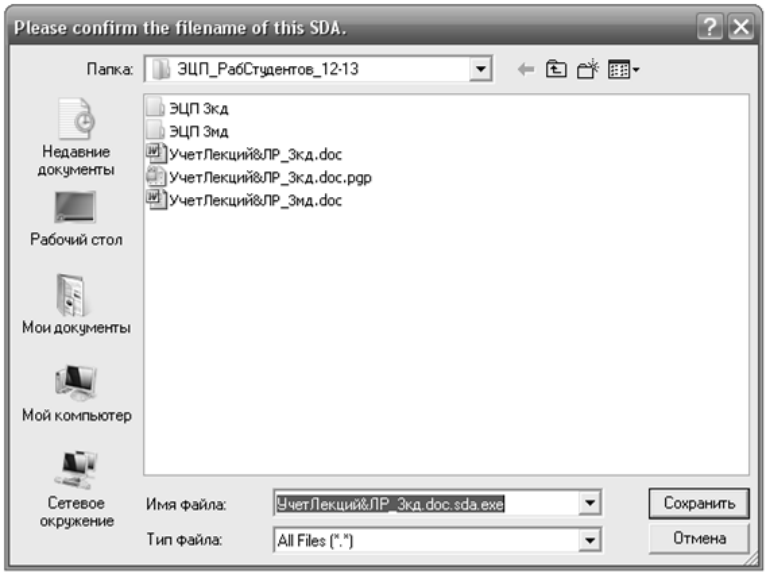


Рисунок 9.7.14 – Предложение подтвердить имя файла SDA

Созданный файл является файл-приложением с именем исходного файла, но с расширением .sda.exe. SDA – это аббревиатура самораспаковывающегося файла-приложения. Объем этого файла будет намного больше исходного, потому что исходный текст сообщения вставляется в исполняемый файл, то есть в специальную программу, которая может быть выполнена без PGP.

После уточнения имени и места хранения файла щелкнуть по кнопке Сохранить. Сохраненный файл можно пересылать получателю.

2) Для расшифрования файла с расширением .sda.exe. необходимо:

– вызвать папку с принятым файлом и щелкнуть по имени файла. Файл запустится на исполнение и выведет на экран окно (рисунок 9.7.15) PGP Self Decrypting Archive – Enter Passphrase;

Рисунок 9.7.15 – Информация о запуске файла на исполнение

– сообщение в окне о том, что файл зашифрован по технологии PGP сопровождается приглашением на веб-сайт PGP для более детального изучения этой технологии. Ниже отображения директории размещения файла расположено окно для ввода ключа, которым по договоренности с получателем зашифровано сообщение;
– ввести ключ и щелкнуть по кнопке ОК, после чего в папку, где хранится файл-приложение, будет записано расшифрованное сообщение.

Б. Содержание отчета

1) Придумать ПАРОЛЬ и записать его в документе MS Word с именем (например, *иванов_3мд_лаб7_пароль.doc*).
2) Зашифровать этот файл по варианту № 4, получится файл *иванов_3мд_лаб7_пароль.pgp*).
3) Взять любой документ Word, переименовать его в файл *иванов_3мд_лаб7.doc.*
4) Зашифровать его по варианту № 8, получится файл *иванов_3мд_лаб7.sda.exe.*
5) Заархивировать полученные файлы и отправить их преподавателю.

ЗАКЛЮЧЕНИЕ

Данное учебное пособие содержит актуальный материал теоретического и практического характера в области электронной цифровой подписи по следующим темам: правовые и организационные основы использования электронной цифровой подписи в деятельности предприятий и организаций; общие вопросы криптографии; общие сведения о методах шифрования; электронная цифровая подпись на основе алгоритмов с открытым ключом; аалгоритмы хеширования; алгоритмы электронной цифровой подписи; проблемы идентификации и аутентификации пользователей и документов в информационных системах; современные программные комплексы электронной цифровой подписи и системы защиты информации от несанкционированного доступа.

Ценность содержания учебного пособия определяется бурным развитием информационных и телекоммуникационных технологий, их активным внедрением в деятельность предприятий и организаций, и вытекающей из этого потребностью защиты информации.

Весь материал, изложенный в учебном пособии, структурирован по разделам, каждый из которых снабжен контрольными вопросами и упражнениями для самоконтроля.

Глубокое изучение рассматриваемых в учебном пособии вопросов реализации и использования средств электронной цифровой подписи будет способствовать становлению студента как специалиста в выбранной им области.

ЛИТЕРАТУРА

1 Об электронной подписи: ФЗ РФ № 63-ФЗ от 6 апреля 2011 г.

2 Об электронной цифровой подписи: ФЗ РФ № 1-ФЗ от 10 января 2002 г.

3 ГОСТ 28147-89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования данных.

4 ГОСТ Р34.10-94. Информационная технология. Криптографическая защита информации. Процедуры выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма.

5 ГОСТ Р34.11-94. Информационная технология. Криптографическая защита информации. Функция хеширования.

6 ГОСТ Р34.10-2001. Процессы формирования и проверки электронной цифровой подписи.

7 Басалова, Г.В. Основы криптографии / Г.В. Басалова. – Интернет-университет информационных технологий – ИНТУИТ.ру, 2010.

8 Басалова, Г.В. Основы криптографии / Г.В. Басалова. – Тула: Изд-во Тульского государственного университета, 2009.

9 Программно-аппаратная защита информации: учеб. пособие /С.К. Варлатая, М.В. Шаханова. – Владивосток: Изд-во ДВГТУ, 2007.

10 Кобелев, О.А. Электронная коммерция: учебное пособие / под ред. С.В.Пирогова. – 2-е изд., перераб. и доп. – М.: Издательско-торговая корпорация «Дашков и Ко», 2006. – 684 с.

11 Молдовян, Н.А. Введение в криптосистемы с открытым ключом /Н.А.Молдовян, А.А.Молдовян. – Санкт-Петербург: Издательство BHV-СПб, 2005.

12 Рябко, Б.Я. Криптографические методы защиты информации / Б.Я. Рябко, А.Н. Фионов. – М.: Горячая линия-Телеком, 2005.

13 www.intuit.ru – учебный комплекс INTUIT.RU Интернет-университета Информационных технологий: Основы криптографии.

14 www.iecp.ru – Единый портал ЭЦП в РФ.

Учебное издание

Трипкош Владимир Алойсович
Матвеев Александр Геннадьевич

ЭЛЕКТРОННАЯ ЦИФРОВАЯ ПОДПИСЬ В ДЕЯТЕЛЬНОСТИ ПРЕДПРИЯТИЙ И ОРГАНИЗАЦИЙ

Подписано в печать 28.12.2012. Формат 60×84/16.
Печать трафаретная. Усл. печ. л. 10,0.
Тираж 300 экз. (1-й завод 100 экз.) Заказ № 4693.

Издательский центр ОГАУ.
460014, г. Оренбург, ул. Челюскинцев, 18.
Тел. (3532) 77-61-43.