ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ
Государственное образовательное учреждение высшего профессионального образования
«Поволжская государственная академия телекоммуникаций и информатики»

Кафедра ПДС

Методические разработки к лабораторным работам по дисциплине

«СРЕДСТВА ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ В СЕТЯХ ПЕРЕДАЧИ ДАННЫХ»

для студентов специальностей 200900, 201000, 201800

Самара, 2010

Методические разработки к лабораторным работам по дисциплине «Средства обеспечения информационной безопасности в сетях передачи данных» / Сост. к.т.н., доц. А.В.Крыжановский, д.т.н.,проф.А.А. Нерсесянц -Самара, 2005.-140 с. , ил.

Приведены методические указания к лабораторным работам, относящимся к основным разделам криптографии: симметричные и асимметричные криптосистемы, электронная цифровая подпись, управление криптографическими ключами и идентификация. Все методические указания снабжены краткими теоретическими сведениями.

Методические разработки утверждены на заседании кафедры ПДС 7.06.2005 г.,протокол № 8.

Редактор- д.т.н., проф. Б.Я.Лихтциндер
Рецензент-д.т.н. проф. В.Г. Карташевский

Лабораторная работа №1

ИЗУЧЕНИЕ ОТЕЧЕСТВЕННОГО СТАНДАРТА ШИФРОВАНИЯ ГОСТ 28147-89

1. Цель работы

Изучить основные принципы и особенности симметричного алгоритма шифрования по стандарту ГОСТ 28147-89.

2. Рекомендуемые источники

1. ГОСТ 28147-89. Система обработки информации. Защита крипотографическая. Алгоритм криптографического преобразования информации.
2. Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях /Под ред. В.Ф. Шаньгина – 2-е изд. перераб. и доп. – М.:Радио и связь,2001. с. 98-111.
3. Соколов А.В., Шаньгин В.Ф. Защита информации в распределенных корпоративных сетях и системах. – М.: ДМК Пресс, 2002, с. 181 – 194.
4. Б. Шнайер. Прикладная криптография. – М.: Триумф, 2002, с. 373 – 377.
5. Шеннон К.Э. Теория связи в секретных системах. В кн. К.Э. Шеннона «Работы по теории информации и кибернетике».- М.: ИЛ, 1963, - с. 243 – 332.

3. Подготовка к работе

1. Ознакомиться с алгоритмом шифрования ГОСТ 28147-89 по одному из рекомендованных источников /1 – 4/.
2. Ознакомиться с содержанием данной методической разработки.
3. Подготовить бланк отчета, который должен содержать:
   - цель работы;

- блок-схему алгоритма в режиме простой замены;
- заготовки таблиц по п.8 задания;
- объяснение результатов шифрования и дешифрования.

4. Контрольные вопросы

1. Объясните сущность основных понятий: криптография, криптоанализ, конфиденциальность и целостность информации.
2. Обобщенная схема симметричной криптосистемы.
3. Обобщенная схема асимметричной криптосистемы.
4. Понятие криптостойкости системы шифрования.
5. Основные принципы криптоаналитических атак..
6. Сущность криптоаналитических атак при наличии открытого и зашифрованноготекстов.
7. Сущность криптоаналитических атак при возможности выбора открытого текста.
8. Сущность криптоаналитических атак с использованием выбранного шифротекста.
9. Сущность криптоаналитических атак методом полного перебора всехвозможных ключей (методом «грубой силы»).
10. Сущность принципа рассеивания.
11. Сущность принципа перемешивания.
12. Принцип шифрования по стандарту ГОСТ 28147-89.
13. Принцип расшифрования по стандарту ГОСТ 28147-89.
14. Процедуры подстановки и перестановки в стандарте ГОСТ 28147-89.
15. Достоинства и недостатки шифрования по стандарту ГОСТ 28147-89.

5. Содержание работы

1. Ознакомиться с алгоритмом и реализационными основами шифрования по ГОСТ 28147-89.

2. Изучить основные этапы шифрования на компьютерной модели.
3. Изучить основные особенности алгоритма по стандарту ГОСТ 28147-89.

6. Содержание отчета

1. Цель работы.
2. Структурную схему алгоритма шифрования по стандарту ГОСТ 28147-89 в режиме простой замены.
3. Таблицы шифрования и дешифрования.
4. Выводы о технических возможностях, особенностях, преимуществах и областях применения алгоритма по стандарту ГОСТ 28147-89.

7. Методические указания к выполнению работы

Лабораторная работа выполняется на ПЭВМ в диалоговом режиме.

После запуска программы Zinf на экране монитора возникает главное меню, на котором нужно выбрать пункт GOST 28147. Возврат в главное меню и выход из него осуществляется кнопкой EXIT. Программа Zinf не контролирует ввод некорректных данных и ошибочных действий пользователя, поэтому требуется внимательность, а для выхода из тупиковых ситуаций нужно воспользоваться кнопкой EXIT.

Лабораторная программа ГОСТ имитирует процедуры, установленные стандартом для шифрования в режиме *простой замены***.**

Шифруемый 64-х разрядный блок информации вводится в окна «Блок №1» и «Блок №2» в шестнадцатеричном коде, по 8 шестнадцатеричных цифр в каждое окно. Ввод большего числа цифр программа воспринимает как ошибку.

Соответствие десятичных, двоичных и шестнадцатеричных чисел представлено в табл. 1-1.

Таблица 1.1-Соответствие чисел в различных кодах

Десят.	Двоичн.	Шестн.	Десят.	Двоичн.	Шестн.
0	0000	0	8	1000	8
1	0001	1	9	1001	9
2	0010	2	10	1010	A
3	0011	3	11	1011	B
4	0100	4	12	1100	C
5	0101	5	13	1101	D
6	0110	6	14	1110	E
7	0111	7	15	1111	F

Ключ, состоящий из 256 разрядов, вводится в 8 окон также в шестнадцатеричном коде по 8 цифр в каждое окно (Кл. XO, X!, …,X7).

В программе для наблюдения за процессом обработки данных, реализованы 2 варианта работы:

—основной вариант – клавиши «Зашифровать» обеспечивают штатный режим, при котором основной цикл (зашифрование и расшифрование одним из ключей) выполняется непрерывно 32 раза;

—учебный вариант (шаговый режим) – цикл разбит на 5 этапов: суммирование по mod 32, подстановка, сдвиг, суммирование по mod 2, перепись. Нажимая на соответствующие клавиши, можно последовательно наблюдать за поэтапным процессом шифрования.

Для упрощения программы в учебном варианте используется только подключ XO, что соответствует варианту 8 одинаковых подключей.

Другой особенностью программы является вариант реализации подстановок. Стандарт рекомендует для

каждой тетрады 32-х разрядного слова S использовать различные варианты секретных таблиц-подстановок (8 таблиц).

В данной учебной программе для ее упрощения реализована одна таблица замен:0→F, 1→E, 2→D, 3→C, 4→B, 5→A, 6→9, 7→8, 8→7, 9→6, A→5, B→4, C→3, D→2, E→1, F→0.

В лабораторной программе в целях изучения влияния на качество шифрования подстановок и перестановок (сдвигов), есть возможность отключать любой из этих этапов с помощью флажков.

Еще раз обратите внимание на то, что в окна информационных блоков и ключей нельзя вводить больше 8 цифр (меньше – можно) и нельзя делать пробелы в словах.

8. Лабораторное задание

1.Нажатием клавиши ТЕСТ произвести тестовую загрузку информации и ключей.

Освоить работу программы шифрования во всех ее вариантах.

2.Проверить реализацию в данном алгоритме принципа: «Изменение одного символа (бита) в информации должно распространиться на большое число символов криптограммы». Для этого:

- ввести тестовую комбинацию;
- зашифровать информацию и записать криптограмму в двоичном коде (64 разряда);
- изменить один бит в одном из информационных блоков;
- провести шифрование в четырех вариантах в зависимости от включения или выключения флажков-этапов подстановки и сдвига. Записать полученные результаты в двоичном коде.

3.Проверить аналогично, реализацию принципа «Изменение одного символа ключа должно распространяться на большое число знаков криптограммы».

4.Проверить работу алгоритма шифрования при «плохих» ключах.

Например, ключи из одних нулей (64 шестнадцатеричных «0») или из одних единиц (64 цифры «F»). Записать для каждого случая варианта (по включению флажков) криптограммы в шестнадцатеричном и двоичном кодах.

9. Общие сведения

9.1. Принципы криптографической защиты информации

Криптография представляет собой совокупность методов преобразования данных, направленных на то, чтобы сделать эти данные бесполезными для противника. Такие преобразования позволяют решить две главные проблемы защиты данных: *проблему конфиденциальности* (путем лишения противника возможности извлечь информацию из канала связи) и *проблему целостности* (путем лишения противника возможности изменить сообщение так, чтобы изменился его смысл, или ввести ложную информацию в канал связи).

Проблемы конфиденциальности и целостности информации тесно связаны между собой, поэтому методы решения одной из них часто применимы для решения другой.

Обобщенная схема криптографической системы, обеспечивающей шифрование передаваемой информации, показана на рис.1.1. *Отправитель* генерирует *открытый текст* исходного сообщения M, которое должно быть передано законному *получателю* по незащищенному

каналу. За каналом следит *перехватчик* с целью перехватить и раскрыть передаваемое сообщение. Для того чтобы перехватчик не смог узнать содержание сообщения M, отправитель шифрует его с помощью обратимого преобразования E_K и получает *шифртекст* (или *криптограмму*) $C = E_K(M)$, который отправляет получателю.

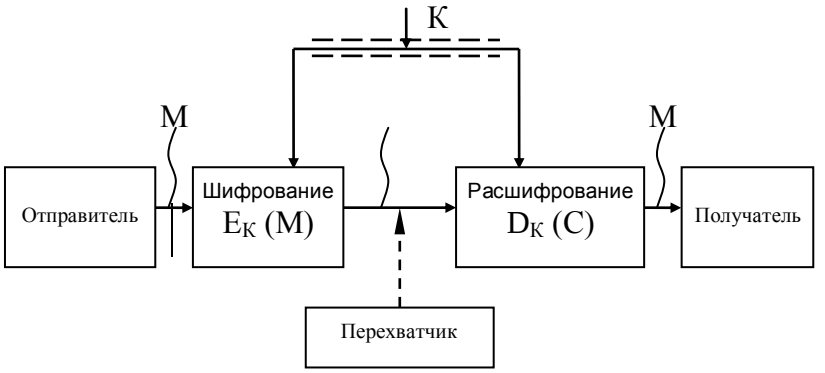

Рисунок1.1- Обобщенная схема криптосистемы

Законный получатель, приняв шифртекст C, расшифровывает его с помощью обратного преобразования $D = E_K^{-1}$ и получает исходное сообщение в виде открытого текста M:

$$D_K(C) = E_K^{-1}(E_K(M)) = M.$$

Преобразование E_K выбирается из семейства криптографических преобразований, называемых криптоалгоритмами,. Параметр, с помощью которого выбирается отдельное используемое преобразование, называется криптографическим ключом K. Криптосистема имеет разные варианты реализации: набор инструкций, аппаратные средства, комплекс программ компьютера, которые позволяют зашифровать открытый текст и

расшифровать шифр-текст различными способами, один из которых выбирается с помощью конкретного ключа К.

Говоря более формально, криптографическая система – это однопараметрическое семейство $(E_K)_{K\in\overline{K}}$ обратимых преобразований

$$E_K : \overline{M} \to \overline{C}$$

из пространства $\overline{M}$ сообщений открытого текста в пространство $\overline{C}$ шифрованных текстов. Параметр К (ключ) выбирается из конечного множества $\overline{K}$, называемого *пространством ключей.*

Вообще говоря, преобразование шифрования может быть симметричным или асимметричным относительно преобразования расшифрования. Это важное свойство функции преобразования определяет два класса криптосистем:

- симметричные (одноключевые) криптосистемы;
- асимметричные (двухключевые) криптосистемы (с открытым ключом).

Схема симметричной криптосистемы с одним секретным ключом была показана на рис.1.1. В ней используются одинаковые секретные ключи в блоке шифрования и блоке расшифрования.

Обобщенная схема асимметричной криптосистемы с двумя разными ключами $К_1$ и $К_2$ показана на рис.1.2. В этой криптосистеме один из ключей является открытым, а другой – секретным.

В симметричной криптосистеме секретный ключ надо передавать отправителю и получателю по защищенному каналу распространения ключей, например такому, как курьерская служба. На рис.1.1 этот канал показан "экранированной" линией. Существуют и другие способы распределения секретных ключей, они будут рассмотрены позднее. В асимметричной криптосистеме передают по

незащищенному каналу только открытый ключ, а секретный ключ сохраняют на месте его генерации.

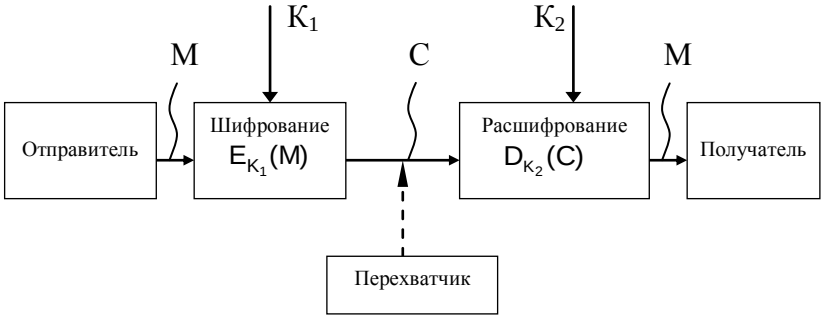

Рисунок 1.2- Обобщенная схема асимметричной криптосистемы с открытым ключом

На рис.1.3 показан поток информации в криптосистеме в случае активных действий перехватчика. Активный перехватчик не только считывает все шифртексты, передаваемые по каналу, но может также пытаться изменять их по своему усмотрению.

Любая попытка со стороны перехватчика расшифровать шифртекст С для получения открытого текста М или зашифровать свой собственный текст М’ для получения правдоподобного шифртекста С’, не имея подлинного ключа, называется *крипто-аналитической атакой.*

Если предпринятые криптоаналитические атаки не достигают поставленной цели и криптоаналитик не может, не имея подлинного ключа, вывести М из С или С’ из М’, то полагают, что такая криптосистема является *криптостойкой*.

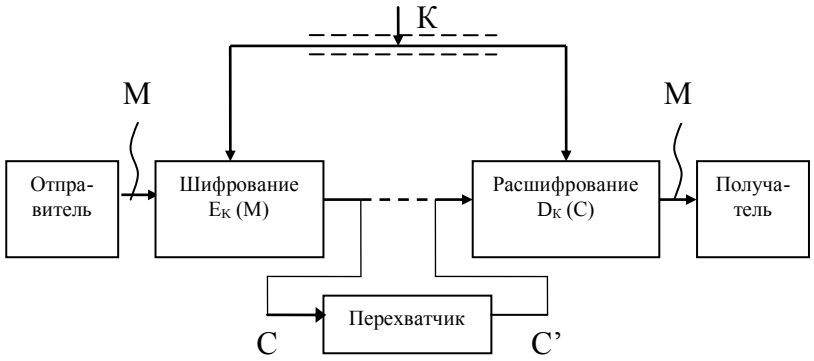

Рисунок1. 3- Поток информации в криптосистеме при активном перехвате сообщений

Криптоанализ – это наука о раскрытии исходного текста зашифрованного сообщения без доступа к ключу. Успешный анализ может раскрыть исходный текст или ключ. Он позволяет также обнаружить слабые места в криптосистеме, что, в конечном счете, ведет к тем же результатам.

Фундаментальное правило криптоанализа, впервые сформулированное голландцем А.Керкхоффом еще в XIX веке заключается в том, что стойкость шифра (криптосистемы) должна определяться только секретностью ключа. Иными словами, правило Керкхоффа состоит в том, что весь алгоритм шифрования, кроме значения секретного ключа, известен криптоаналитику противника. Это обусловлено тем, что криптосистема, реализующая семейство криптографических преобразований, обычно рассматривается как открытая система. Такой подход отражает очень важный принцип технологии защиты информации: защищенность системы не должна зависеть от секретности чего-либо такого, что

невозможно быстро изменить в случае утечки секретной информации. Обычно криптосистема представляет собой совокупность аппаратных и программных средств, которую можно изменить только при значительных затратах времени и средств, тогда как ключ является легко изменяемым объектом. Именно поэтому стойкость криптосистемы определяется только секретностью ключа.

Другое почти общепринятое допущение в криптоанализе состоит в том, что криптоаналитик имеет в своем распоряжении шифртексты сообщений.

Существует четыре основных типа криптоаналитических атак. Конечно, все они формулируются в предположении, что криптоаналитику известны применяемый алгоритм шифрования и шифртексты сообщений. Перечислим эти криптоаналитические атаки.

1. Криптоаналитическая атака при наличии только известных шифртекстов

Криптоаналитик имеет только шифртексты C_1, C_2, …, C_i нескольких сообщений, причем все они зашифрованы с использованием одного и того же алгоритма шифрования E_K. Работа криптоаналитика заключается в том, чтобы раскрыть исходные тексты M_1, M_2, …, M_i по возможности большинства сообщений или, еще лучше, вычислить ключ К, использованный для шифрования этих сообщений, с тем, чтобы расшифровать и другие сообщения, зашифрованные этим ключом.

2. Криптоаналитическая атака при наличии известных открытых текстов

Криптоаналитик имеет доступ не только к шифртекстам C_1, C_2, …, C_i нескольких сообщений, но также к открытым текстам M_1, M_2, …, M_i этих сообщений. Его работа

заключается в нахождении ключа К, используемого при шифровании этих сообщений, или алгоритма расшифрования D_K любых новых сообщений, зашифрованных тем же самым ключом.

3. Криптоаналитическая атака при возможности выбора открытых текстов

Криптоаналитик не только имеет доступ к шифртекстам C_1, C_2, …, C_i и связанным с ними открытым текстам M_1, M_2, …, M_i нескольких сообщений, но и может по желанию выбирать открытые тексты, которые затем получает в зашифрованном виде. Такой криптоанализ получается более мощным по сравнению с криптоанализом с известным открытым текстом, потому что криптоаналитик может выбрать для шифрования такие блоки открытого текста, которые дадут больше информации о ключе. Работа криптоаналитика состоит в поиске ключа К, использованного для шифрования сообщений, или алгоритма расшифрования D_K новых сообщений, зашифрованных тем же ключом.

4. Криптоаналитическая атака с адаптивным выбором открытого текста

Это – особый вариант атаки с выбором открытого текста. Криптоаналитик может не только выбирать открытый текст, который затем шифруется, но и изменять свой выбор в зависимости от результатов предыдущего шифрования. При криптоанализе с простым выбором открытого текста криптоаналитик обычно может выбирать несколько крупных блоков открытого текста для их шифрования; при криптоанализе с адаптивным выбором открытого текста он имеет возможность выбрать сначала более мелкий пробный блок открытого текста, затем выбрать следующий блок в зависимости от результатов

первого выбора, и т.д. Эта атака предоставляет криптоаналитику еще больше возможностей, чем предыдущие типы атак.

Кроме перечисленных основных типов криптоаналитических атак, можно отметить, по крайней мере, еще два типа.

5. Криптоаналитическая атака с использованием выбранного шифртекста

Криптоаналитик может выбирать для расшифрования различные шифртексты C_1, C_2, …, C_i и имеет доступ к расшифрованным открытым текстам M_1, M_2, …, M_i. Например, криптоаналитик получил доступ к защищенному от несанкционированного вскрытия блоку, который выполняет автоматическое расшифрование. Работа криптоаналитика заключается в нахождении ключа. Этот тип криптоанализа представляет особый интерес для раскрытия алгоритмов с открытым ключом.

6. Криптоаналитическая атака методом полного перебора всех возможных ключей

Эта атака предполагает использование криптоаналитиком известного шифртекста и осуществляется посредством полного перебора всех возможных ключей с проверкой, является ли осмысленным получающийся открытый текст. Такой подход требует привлечения предельных вычислительных ресурсов и иногда называется силовой атакой.

Существуют и другие, менее распространенные, криптоаналитические атаки.

9.2. Основные виды шифрования

Большинство средств защиты информации базируется на использовании криптографических шифров и процедур шифрования-расшифрования. В соответствии со

стандартом ГОСТ 28147-89 под шифром понимают совокупность обратимых преобразований множества открытых данных на множество зашифрованных данных, задаваемых ключом и алгоритмом криптографического преобразования.

Ключ – это конкретное секретное состояние некоторых параметров алгоритма криптографического преобразования данных, обеспечивающее выбор только одного варианта из всех возможных для данного алгоритма /1/.

Основной характеристикой шифра является *криптостойкость*, которая определяет его стойкость к раскрытию методами криптоанализа. Обычно эта характеристика определяется интервалом времени, необходимым для раскрытия шифра.

К шифрам, используемым для криптографической защиты информации, предъявляется ряд требований:

- достаточная криптостойкость (надежность закрытия данных);
- простота процедур шифрования и расшифрования;
- незначительная избыточность информации за счет шифрования;
- нечувствительность к небольшим ошибкам шифрования и др.

В той или иной мере этим требованиям отвечают:

- шифры перестановок;
- шифры замены;
- шифры гаммирования;
- шифры, основанные на аналитических преобразованиях шифруемых данных.

Шифрование перестановкой заключается в том, что символы шифруемого текста переставляются по определенному правилу в пределах некоторого блока этого текста. При достаточной длине блока, в пределах которого осуществляется перестановка, и сложном

неповторяющемся порядке перестановки можно достигнуть приемлемой для простых практических приложений стойкости шифра.

Шифрование заменой (подстановкой) заключается в том, что символы шифруемого текста заменяются символами того же или другого алфавита в соответствии с заранее обусловленной схемой замены.

Шифрование гаммированием заключается в том, что символы шифруемого текста складываются с символами некоторой случайной последовательности, именуемой *гаммой шифра*. Стойкость шифрования определяется в основном длиной (периодом) неповторяющейся части гаммы шифра. Поскольку с помощью ЭВМ можно генерировать практически бесконечную гамму шифра, то данный способ является одним из основных для шифрования информации в автоматизированных системах.

Шифрование аналитическим преобразованием заключается в том, что шифруемый текст преобразуется по некоторому аналитическому правилу (формуле).

Например, можно использовать правило умножения вектора на матрицу, причем умножаемая матрица является ключом шифрования (поэтому ее размер и содержание должны храниться в секрете), а символами умножаемого вектора последовательно служат символы шифруемого текста. Другим примером может служить использование так называемых однонаправленных функций для построения криптосистем с открытым ключом.

По мнению К. Шеннона /5/, в практических шифрах необходимо использовать два основных общих принципа: рассеивание и перемешивание.

Рассеивание представляет собой распространение влияния одного знака открытого текста на много знаков

шифртекста, что позволяет скрыть статистические свойства открытого текста.

Перемешивание предполагает использование таких шифрующих преобразований, которые усложняют восстановление взаимосвязи статистических свойств открытого и шифрованного текстов. Однако шифр должен не только затруднять раскрытие, но и обеспечивать легкость зашифрования и расшифрования при известном пользователю секретном ключе.

Распространенным способом достижения эффектов рассеивания и перемешивания является использование *составного шифра*, т.е. такого шифра, который может быть реализован в виде некоторой последовательности простых шифров, каждый из которых вносит свой вклад в значительное суммарное рассеивание и перемешивание.

В составных шифрах в качестве простых шифров чаще всего используются простые перестановки и подстановки. При *перестановке* просто перемешивают символы открытого текста, причем конкретный вид перемешивания определяется секретным ключом. При *подстановке* каждый символ открытого текста заменяют другим символом из того же алфавита, а конкретный вид подстановки также определяется секретным ключом. Следует заметить, что в современном блочном шифре блоки открытого текста и шифртекста представляют собой двоичные последовательности обычно длиной 64 бита. В принципе каждый блок может принимать 2^{64} значений. Поэтому подстановки выполняются в очень большом алфавите, содержащем до $2^{64} \approx 10^{19}$ "символов".

При многократном чередовании простых перестановок и подстановок, управляемых достаточно длинным секретным ключом, можно получить очень стойкий шифр с хорошим рассеиванием и перемешиванием. Рассмотренные ниже криптоалгоритмы DES, IDEA и отечественный

стандарт шифрования данных построен в полном соответствии с указанной методологией.

9.3. Отечественный стандарт шифрования данных

В нашей стране установлен единый алгоритм криптографического преобразования данных для систем обработки информации в сетях ЭВМ, отдельных вычислительных комплексах и ЭВМ, который определяется ГОСТ 28147-89 /1/. Стандарт обязателен для организаций, предприятий и учреждений, применяющих криптографическую защиту данных, хранимых и передаваемых в сетях ЭВМ, в отдельных вычислительных комплексах и ЭВМ.

Этот алгоритм криптографического преобразования данных предназначен для аппаратной и программной реализации, удовлетворяет криптографическим требованиям и не накладывает ограничений на степень секретности защищаемой информации. Алгоритм шифрования данных представляет собой 64-битовый блочный алгоритм с 256-битовым ключом.

При описании алгоритма используются следующие обозначения:

L и R – последовательности битов;

LR – конкатенация последовательностей L и R, в которой биты последовательности R следуют за битами последовательности L;

$\oplus$ *– операция побитового сложения по модулю 2;*

$\boxplus$ *– операция сложения по модулю* 2^{32} *двух 32-разрядных двоичных чисел;*

$\boxplus'$ *– операция сложения двух 32-разрядных чисел по модулю* $2^{32}-1$.

Два целых числа a, b, где $0 \le a, b \le 2^{32}-1$,

$a = (a_{32}a_{31} \ldots a_2a_1)$, $b = (b_{32}, b_{31}, \ldots, b_2, b_1)$,

представленные в двоичном виде, т.е.

$$a = a_{32}\cdot 2^{31} + a_{31}\cdot 2^{30} + ... + a_2\cdot 2^1 + a_1,$$
$$b = b_{32}\cdot 2^{31} + b_{31}\cdot 2^{30} + ... + b_2\cdot 2^1 + b_1,$$

суммируются по модулю 2^{32} (операция $\boxplus$) по следующему правилу:

$$a \boxplus b = a + b, \text{ если } a + b < 2^{32},$$
$$a \boxplus b = a + b - 2^{32}, \text{ если } a + b \geq 2^{32}.$$

Правила суммирования чисел по модулю $2^{32} - 1$:

$$a \boxplus' b = a + b, \text{ если } a + b < 2^{32} - 1,$$
$$a \boxplus' b = a + b - (2^{32} - 1), \text{ если } a + b \geq 2^{32} - 1.$$

Алгоритм предусматривает четыре режима работы:

- шифрование данных в режиме простой замены;
- шифрование данных в режиме гаммирования;
- шифрование данных в режиме гаммирования с обратной связью;
- выработка имитовставки.

Режим простой замены

Для реализации алгоритма шифрования данных в режиме простой замены используется только часть блоков общей криптосистемы (рис.1.4). Обозначения на схеме:

N_1, N_2 – 32-разрядные накопители;

CM_1 – 32-разрядный сумматор по модулю 2^{32} ($\boxplus$);

CM_2 – 32-разрядный сумматор по модулю 2 ($\oplus$);

R – 32-разрядный регистр циклического сдвига;

КЗУ – ключевое запоминающее устройство на 256 бит, состоящее из восьми 32-разрядных накопителей X_0, X_1, X_2, ..., X_7;

S – блок подстановки, состоящий из восьми узлов замены (S-блоков замены) S_1, S_2, S_3, ..., S_7, S_8.

9.3.1. Зашифрование открытых данных в режиме простой замены

Открытые данные, подлежащие зашифрованию, разбивают на 64-разрядные блоки T_0. Процедура зашифрования 64-разрядного блока T_0 в режиме простой замены включает 32 цикла (j = 1…32). В ключевое запоминающее устройство вводят 256 бит ключа К в виде восьми 32-разрядных подключей (чисел) K_i:

$$K=K_7K_6K_5K_4K_3K_2K_1K_0.$$

Последовательность битов блока $T_0=(a_1(0), a_2(0), ..., a_{31}(0), a_{32}(0), b_1(0), b_2(0),..., b_{31}(0), b_{32}(0))$ разбивают на две последовательности по 32 бита: b(0) a(0), где b(0) – левые или старшие биты, a(0) – правые или младшие биты.

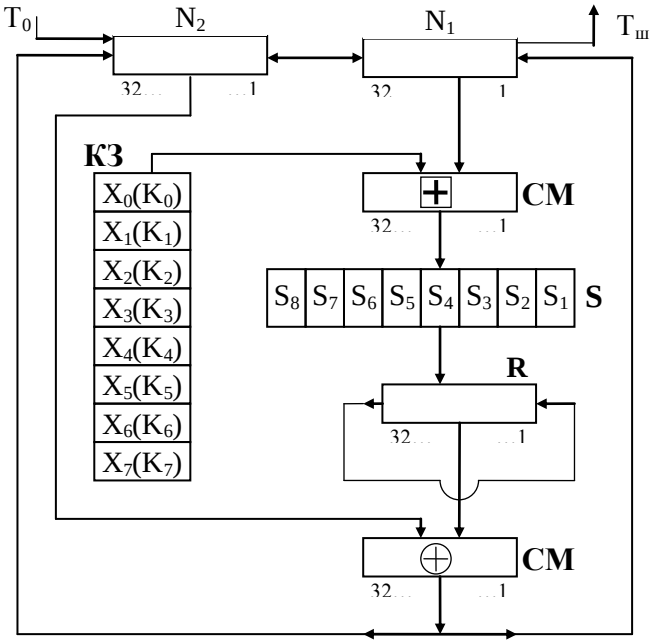

Рисунок1. 4- Схема реализации режима простой замены

Эти последовательности вводят в накопители N_1 и N_2 перед началом первого цикла зашифрования. В результате начальное заполнение накопителя N_1

$a(0) = (a_{32}(0), a_{31}(0), ..., a_2(0), a_1(0))$,

32, 31, ... 2, 1 ← номер разряда N_1

начальное заполнение накопителя N_2

$b(0) = (b_{32}(0), b_{31}(0), ..., b_2(0), b_1(0))$.

32, 31, ... 2, 1 ← номер разряда N_2

Первый цикл (j=1) процедуры зашифрования 64-разрядного блока открытых данных можно описать уравнениями:

$$\begin{cases} a(1) = f(a(0) \boxplus K_0) \oplus b(0), \\ b(1) = a(0). \end{cases}$$

Здесь a(1) – заполнение N_1 после 1-го цикла зашифрования; b(1) – заполнение N_2 после 1-го цикла зашифрования; f – функция шифрования.

Аргументом функции f является сумма по модулю 2^{32} числа a(0) (начального заполнения накопителя N_1) и числа K_0 – подключа, считываемого из накопителя X_0 КЗУ. Каждое из этих чисел равно 32 битам.

Функция f включает две операции над полученной 32-разрядной суммой $(a(0) \boxplus K_0)$.

Первая операция называется *подстановкой (заменой)* и выполняется блоком подстановки S. Блок подстановки S состоит из восьми узлов замены (S-блоков замены) $S_1, S_2, ..., S_8$ с памятью 64 бит каждый. Поступающий из CM_1 на блок подстановки S 32-разрядный вектор разбивают на восемь последовательно идущих 4-разрядных векторов, каждый из которых преобразуется в четырехразрядный вектор соответствующим узлом замены. Каждый узел замены можно представить в виде таблицы-перестановки шестнадцати четырехразрядных двоичных чисел в диапазоне 0000…1111. Входной вектор указывает адрес строки в таблице, а число в этой строке является выходным

вектором. Затем четырехразрядные выходные векторы последовательно объединяют в 32-разрядный вектор. Узлы замены (таблицы-перестановки) представляют собой ключевые элементы, которые являются общими для сети ЭВМ и редко изменяются. Эти узлы замены должны сохраняться в секрете.

Вторая операция – *циклический сдвиг влево* (на 11 разрядов) 32-разрядного вектора, полученного с выхода блока подстановки S. Циклический сдвиг выполняется регистром сдвига R.

Далее результат работы функции шифрования f суммируют поразрядно по модулю 2 в сумматоре CM_2 с 32-разрядным начальным заполнением b(0) накопителя N_2. Затем полученный на выходе CM_2 результат (значение a(1)) записывают в накопитель N_1, а старое значение N_1 (значение a(0)) переписывают в накопитель N_2 (значение b(1) = a(0)). Первый цикл завершен.

Последующие циклы осуществляются аналогично, при этом во втором цикле из КЗУ считывают заполнение X_1 – подключ K_1, в третьем цикле – подключ K_2 и т.д., в восьмом цикле – подключ K_7. В циклах с 9-го по 16-й, а также в циклах с 17-го по 24-й подключи из КЗУ считываются в том же порядке: K_0, K_1, K_2,...,K_6, K_7. В последних восьми циклах с 25-го по 32-й порядок считывания подключей из КЗУ обратный: K_7, K_6, ..., K_2, K_1, K_0. Таким образом, при зашифровании в 32 циклах осуществляется следующий порядок выборки из КЗУ подключей:

$K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7,$

$K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0.$

В 32-м цикле результат из сумматора CM_2 вводится в накопитель N_2, а в накопителе N_1 сохраняется прежнее заполнение. Полученные после 32-го цикла зашифрования заполнения накопителей N_1 и N_2 являются блоком

зашифрованных данных $T_ш$, соответствующим блоку открытых данных T_0.

Уравнения зашифрования в режиме простой замены имеют вид:

$$\begin{cases} a(j) = f(a(j-1) \boxplus_{j-1(mod8)}) \oplus b(j-1) \\ b(j) = a(j-1) \end{cases} \quad \text{при } j=1\ldots24,$$

$$\begin{cases} a(j) = f(a(j-1) \boxplus_{32-j}) \oplus b(j-1) \\ b(j) = a(j-1) \end{cases} \quad \text{при } j=25\ldots31,$$

$$\begin{cases} a(32) = a(31) \\ b(32) = f(a(31) \boxplus_{0}) \oplus b(31) \end{cases} \quad \text{при } j=32,$$

где a(j) = (a_{32}(j), a_{31}(j), ..., a_1(j)) – заполнение N_1 после j-го цикла зашифрования;

b(j) = (b_{32}(j), b_{31}(j), ..., b_1(j)) – заполнение N_2 после j-го цикла зашифрования, j=1…32.

Блок зашифрованных данных $T_ш$ (64 разряда) выводится из накопителей N_1, N_2 в следующем порядке: из разрядов 1…32 накопителя N_1, затем из разрядов 1…32 накопителя N_2, т.е. начиная с младших разрядов:

$T_ш$ = (a_1(32), a_2(32), ..., a_{32}(32), b_1(32), b_2(32), ..., b_{32}(32)).

Остальные блоки открытых данных зашифровываются в режиме простой замены аналогично.

9.3.2.Расшифрование в режиме простой замены

Криптосхема, реализующая алгоритм расшифрования в режиме простой замены, имеет тот же вид, что и при зашифровании (рис. 4).

В КЗУ вводят 256 бит ключа, на котором осуществлялось зашифрование. Зашифрованные данные, подлежащие расшифрованию, разбиты на блоки $T_ш$ по 64 бита в каждом. Ввод любого блока

$T_ш$ = (a_1(32), a_2(32), ..., a_{32}(32), b_1(32), b_2(32), ..., b_{32}(32))

в накопители N_1 и N_2 производят так, чтобы начальное значение накопителя N_1 имело вид

(a_{32}(32), a_{31}(32), ..., a_2(32), a_1(32)),

32, 31, ..., 2, 1 ← номер разряда N_1

а начальное заполнение накопителя N_2 – вид

(b_{32}(32), b_{31}(32), ..., b_2(32), b_1(32)).

32, 31, ..., 2, 1 ← номер разряда N_2

Расшифрование осуществляется по тому же алгоритму, что и зашифрование, с тем изменением, что заполнения накопителей X_0, X_1, ..., X_7 считываются из КЗУ в циклах расшифрования в следующем порядке:

K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0,

K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0.

Уравнения расшифрования имеют вид:

$$\begin{cases} a(32-j) = f(a(32-j+1) \boxplus_{j-1}) \oplus b(32-j+1) \\ b(32-j) = a(32-j+1) \end{cases} \quad \text{при } j=1\ldots8;$$

$$\begin{cases} a(32-j) = f(a(32-j+1) \boxplus_{32-j(mod8)}) \oplus b(32-j+1) \\ b(32-j) = a(32-j+1) \end{cases} \quad \text{при } j=9\ldots31;$$

$$\begin{cases} a(0) = a(1) \\ b(0) = f(a(1) \boxplus_{7}) \oplus b(1) \end{cases} \quad \text{при } j=32.$$

Полученные после 32 циклов работы заполнения накопителей N_1 и N_2 образуют блок открытых данных

T_0 = (a_1(0), a_2(0), ..., a_{32}(0), b_1(0), b_2(0),..., b_{32}(0)),

соответствующий блоку зашифрованных данных $T_ш$. При этом состояние накопителя N_1

(a_{32}(0), a_{31}(0), ..., a_2(0), a_1(0)),

32, 31, ..., 2, 1 ← номер разряда N_1

состояние накопителя N_2

(b_{32}(0), b_{31}(0), ..., b_2(0), b_1(0)).

32, 31, ..., 2, 1 ← номер разряда N_2

Аналогично расшифровываются остальные блоки зашифрованных данных.

Если алгоритм зашифрования в режиме простой замены 64-битового блока T_0 обозначить через A, то

$A(T_0) = A(a(0), b(0)) = (a(32), b(32)) = T_ш$.

Следует иметь в виду, что режим простой замены допустимо использовать для шифрования данных только в ограниченных случаях – при выработке ключа и зашифровании его с обеспечением имитозащиты для передачи по каналам связи или для хранения в памяти ЭВМ.

Лабораторная работа №2

ИЗУЧЕНИЕ АЛГОРИТМА ШИФРОВАНИЯ С ОТКРЫТЫМ КЛЮЧОМ RSA

1. Цель работы

Изучить принципы и процедурные аспекты криптосистемы RSA.

2. Рекомендуемые источники

1.Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях /Под ред. В.Ф. Шаньгина.- 2-е изд.. перераб. и доп..-М.: Радио и связь, 2001, с. 131-138.

2.Соколов А.В., Шаньгин В.Ф. Защита информации в распределенных корпоративных сетях и системах.- М.: ДМК Пресс, 2002, с. 204-206.

3.Б. Шнайер. Прикладная криптография.-М.: Триумф, 2002, с. 521-530.

3. Подготовка к работе

1.Ознакомиться с алгоритмом шифрования по одному из рекомендованных источников/1-3/.

2.Ознакомиться с содержанием данной методической разработки.

3.Подготовить бланк отчета, который должен содержать:

- цель работы;
- основные математические соотношения формирования ключевой информации.

4. Контрольные вопросы

1.Изобразите обобщенную схему асимметричной криптосистемы.

2.Характерные особенности асимметричных криптосистем.
3.Основные требования, обеспечивающие безопасность асимметричной криптосистемы.
4.Понятие однонаправленных функций.
5.Сущность алгоритма шифрования RSA.
6.Сущность процедуры шифрования в системе RSA.
7.Сущность процедуры расшифрования в системе RSA.
8.Требования, предъявляемые к исходным параметрам системы RSA для формирования ключевой информации.

5. Содержание работы

1.Ознакомиться с алгоритмом RSA и его основными особенностями.
2.Изучить основные этапы шифрования на компьютерной модели.
3.Произвести установку открытых и закрытых ключей для двух абонентов.
4.Произвести двухсторонний обмен информацией.

6. Содержание отчета

Отчет должен содержать:
1.Цель работы.
2.Структурную схему асимметричной криптосистемы с открытым ключом.
3.Основные математические соотношения формирования ключевой информации.
4.Таблицы шифрования и дешифрования.
5.Выводы о технических возможностях, особенностях, преимуществах и областях применения алгоритма RSA.

7. Методические указания к выполнению работы

Лабораторная работа выполняется на ПЭВМ в режиме диалога. Все операции программа выполняет в 64 разрядном формате со знаком, т.е. допускается

максимальный размер операндов $2^{63} - 1 = 9{,}2 \cdot 10^{18}$. Однако, с учетом того, что в вычислениях имеются операции умножения, значения операндов-сомножителей не должно превышать значений $3 \cdot 10^{9}$. В лабораторной программе используются числа, содержащие не более 7-8 разрядов.

Лабораторная работа выполняется в двух рабочих окнах. Сначала в окне, совмещенном с главным меню, определяются численные значения параметров криптосистемы RSA для двух участников обмена информацией. Затем в окне «Передача сообщений по методу RSA», которое вызывается из главного меню по пункту «Шифр RSA»→«Обмен сообщениями» задаются цифровые сообщения, устанавливаются значения ключей и осуществляется обмен засекреченными сообщениями.

8. Лабораторное задание

1.Вывести таблицу простых чисел. Для этого в главном меню выбрать пункт «Опции», а затем – пункт «Простые числа», после запуска которого на экране появится рабочее окно «Последовательность простых чисел». После нажатия клавиши «Сформировать ряд простых чисел на экране появится таблица, содержащая 1000 простых чисел от 3 до 7927.

2.Для двух участников обмена зашифрованными сообщениями А (Анна) и Б (Боб) выбрать из таблицы две пары простых чисел p и q, причем $p \neq q$.

3.Для обоих участников вычислить значения модуля N_a и N_b и функции Эйлера
fEa и fEb по соотношениям:

$$N = pq$$
$$fE = (p-1)(q-1)$$

4.Для обоих участников выбрать значения закрытых ключей Da и Db, которые должны быть взаимно простыми с соответствующими функциями Эйлера, т.е. Da с fEa и Db

с fEb. Проще всего это сделать, если в качестве D выбрать из таблицы простое число

1< D < fE и проверить, что fE не делится на D.

5.Выбранные и вычисленные значения Na, fEa, Da, Nb, fEb, Db заносятся в соответствующие позиции главного окна.

6.Нажатием клавиши «Найти открытый ключ» запускается программа вычисления открытых ключей Ea и Eb, удовлетворяющих условию ED = 1 (mod fE).

7.После заполнения всех 8 позиций главного окна процедура формирования ключей считается законченной.

8.Проверить, что полученное значение открытого ключа E и значение модуля N -взаимно простые числа по методике п. 4.

9.Для обмена сообщениями перейти во второе окно, выбрав пункт «Обмен сообщениями» основного меню.

10.Для пользователя A (Анна) записать в цифровой форме передаваемое сообщение в позицию M (Message) (4 знака) при условии M<N. Ввести значения Nb, Eb, полученные ранее. Зашифрованное сообщение программа помещает в позицию Sh (Shifr), которое затем из перой строки (строка Анны) пересылается в позицию Sh второй строки (строка Боба) и расшифровывается. Если все процедуры выполнены правильно, то в позициях M в обеих строках должны появиться одинаковые числа.

Внимание! Распространенная ошибка пользователей криптосистемы RSA заключается в путанице, где чьи ключи использовать. Поэтому при работе во втором окне позиции N,E, и D указаны без индексов a и b. Нужно иметь в виду, что для зашифрования у отправителя (Анны) используется открытый ключ (N и E), а для расшифрования у получателя (Боба) используется тайный ключ (N и D)/

11.Произвести процесс выбора ключей для передачи сообщений от Анны к Бобу и ответа от Боба к Анне

трижды для 6-и, 7-и и 8 значных значений модуля N. Все значения ключей и сообщений (в открытом и зашифрованном виде) зафиксировать и занести в таблицу и отметить продолжительность процедур.

12.Провести наблюдение за процессом засекреченной передачи при несоблюдении следующих условий:

- выбор в качестве ключа D не простого числа;
- (ED) ≠1 (mod fE).

9. Общие сведения

9.1. Концепция криптосистемы с открытым ключом

Эффективными системами криптографической защиты данных являются асимметричные криптосистемы, называемые также криптосистемами с открытым ключом. В таких системах для зашифрования данных используется один ключ, а для расшифрования – другой ключ (отсюда и название – асимметричные). Первый ключ является открытым и может быть опубликован для использования всеми пользователями системы, которые зашифровывают данные. Расшифрование данных с помощью открытого ключа невозможно.

Для расшифрования данных получатель зашифрованной информации использует второй ключ, который является *секретным*. Разумеется, ключ расшифрования не может быть определен из ключа зашифрования.

Обобщенная схема асимметричной криптосистемы с открытым ключом показана на рис.2.1. В этой криптосистеме применяют два различных ключа: $K_в$ – открытый ключ отправителя A; $k_в$ – секретный ключ получателя B. Генератор ключей целесообразно располагать на стороне получателя B (чтобы не пересылать секретный ключ $k_в$ по незащищенному каналу). Значения

ключей $К_в$ и $k_в$ зависят от начального состояния генератора ключей.

Раскрытие секретного ключа $k_в$ по известному открытому ключу $К_в$ должно быть вычислительно неразрешимой задачей.

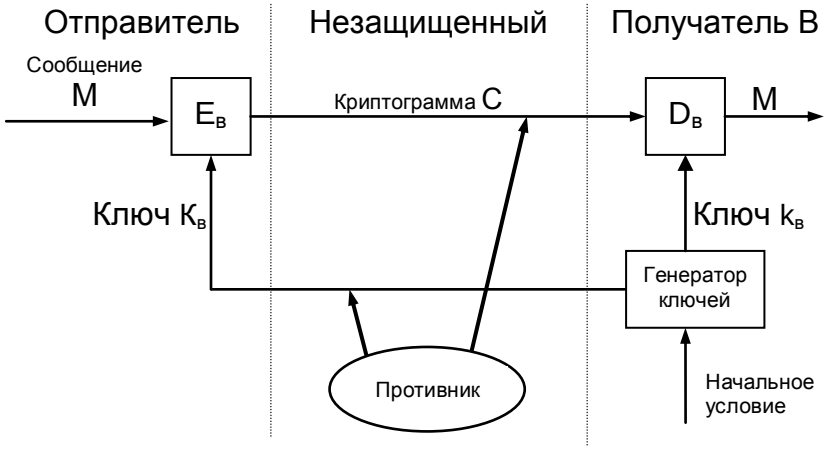

Рисунок 2.1- Обобщенная схема асимметричной криптосистемы с открытым ключом

Характерные особенности асимметричных криптосистем:

1. Открытый ключ $К_в$ и криптограмма С могут быть отправлены по незащищенным каналам, т.е. противнику известны $К_в$ и С.
2. Алгоритмы шифрования и расшифрования

$E_в : M \to C$,

$D_в : C \to M$

являются открытыми.

Защита информации в асимметричной криптосистеме основана на секретности ключа $k_в$.

У.Диффи и М.Хеллман сформулировали требования, выполнение которых обеспечивает безопасность асимметричной криптосистемы:

1. Вычисление пары ключей ($К_в$, $k_в$) получателем В на основе начального условия должно быть простым.
2. Отправитель А, зная открытый ключ $К_в$ и сообщение М, может легко вычислить криптограмму

$$C = E_{K_в}(M) = E_в(M) \qquad (2.1)$$

3. Получатель В, используя секретный ключ $k_в$ и криптограмму С, может легко восстановить исходное сообщение

$$M = D_{k_в}(C) = D_в(C) = D_в[E_в(M)] \qquad (2.2)$$

4. Противник, зная открытый ключ $К_в$, при попытке вычислить секретный ключ $k_в$ наталкивается на непреодолимую вычислительную проблему.
5. Противник, зная пару ($К_в$, С), при попытке вычислить исходное сообщение М наталкивается на непреодолимую вычислительную проблему.

9.2. Однонаправленные функции

Концепция асимметричных криптографических систем с открытым ключом основана на применении однонаправленных функций. Неформально однонаправленную функцию можно определить следующим образом. Пусть X и Y – некоторые произвольные множества. Функциf : $X \to Y$ является однонаправленной, если для всех $x \in X$ можно легко вычислить функцию

$y = f(x)$, где $y \in Y$.

И в то же время для большинства $y \in Y$ достаточно сложно получить значение $x \in X$, такое, что $f(x)=y$ (при этом полагают, что существует по крайней мере одно такое значение x).

Основным критерием отнесения функции f к классу однонаправленных функций является отсутствие эффективных алгоритмов обратного преобразования
$Y \rightarrow X$.

В качестве первого примера однонаправленной функции рассмотрим целочисленное умножение. Прямая задача – вычисление произведения двух очень больших целых чисел P и Q, т.е. нахождение значения

$$N = P*Q \qquad (2.3)$$

является относительно несложной задачей для ЭВМ.

Обратная задача – разложение на множители большого целого числа, т.е. нахождение делителей P и Q большого целого числа $N = P*Q$, является практически неразрешимой задачей при достаточно больших значениях N. По современным оценкам теории чисел при целом $N \approx 2^{664}$ и $P \approx Q$ для разложения числа N потребуется около 10^{23} операций, т.е. задача практически неразрешима на современных ЭВМ.

Следующий характерный пример однонаправленной функции – это модульная экспонента с фиксированными основанием и модулем. Пусть A и N – целые числа, такие, что $1 \leq A < N$. Определим множество Z_N:
$Z_N = \{0, 1, 2, ..., N-1\}$.

Тогда модульная экспонента с основанием A по модулю N представляет собой функцию
$f_{A,N} : Z_N \rightarrow Z_N$,
$f_{A,N}(x) = A^x \pmod N$ (2.4),
где X – целое число, $1 \leq x \leq N-1$.

Существуют эффективные алгоритмы, позволяющие достаточно быстро вычислить значения функции $f_{A,N}(x)$.
Если $y = A^x$, то естественно записать $x = \log_A(y)$.

Поэтому задачу обращения функции $f_{A,N}(x)$ называют задачей нахождения дискретного логарифма или задачей дискретного логарифмирования.

Задача дискретного логарифмирования формулируется следующим образом. Для известных целых A, N, y найти целое число x, такое, что
$A^x \bmod N = y$.

Алгоритм вычисления дискретного логарифма за приемлемое время пока не найден. Поэтому модульная экспонента считается однонаправленной функцией.

По современным оценкам теории чисел при целых числах $A \approx 2^{664}$ и $N \approx 2^{664}$ решение задачи дискретного логарифмирования (нахождение показателя степени x для известного y) потребует около 10^{26} операций, т.е. эта задача имеет в 10^3 раз большую вычислительную сложность, чем задача разложения на множители. При увеличении длины чисел разница в оценках сложности задач возрастает.

Следует отметить, что пока не удалось доказать, что не существует эффективного алгоритма вычисления дискретного логарифма за приемлемое время. Исходя из этого, модульная экспонента отнесена к однонаправленным функциям условно, что, однако, не мешает с успехом применять ее на практике.

Вторым важным классом функций, используемых при построении криптосистем с открытым ключом, являются так называемые однонаправленные функции с "потайным ходом" (с лазейкой). Дадим неформальное определение такой функции. Функция
$f : X \rightarrow Y$
относится к классу однонаправленных функций с "потайным ходом" в том случае, если она является однонаправленной и, кроме того, возможно эффективное вычисление обратной функции, если известен "потайной ход" (секретное число, строка или другая информация, ассоциирующаяся с данной функцией).

В качестве примера однонаправленной функции с "потайным ходом" можно указать используемую в криптосистеме RSA модульную экспоненту с фиксированными модулем и показателем степени. Переменное основание модульной экспоненты используется для указания числового значения сообщения M либо криптограммы C.

9.3. Криптосистема шифрования данных RSA

Алгоритм RSA предложили в 1978 г. три автора: Р.Райвест (Rivest), А.Шамир (Shamir) и А.Адлеман (Adleman). Алгоритм получил свое название по первым буквам фамилий его авторов. Алгоритм RSA стал первым полноценным алгоритмом с открытым ключом, который может работать как в режиме шифрования данных, так и в режиме электронной цифровой подписи.

Надежность алгоритма основывается на трудности факторизации больших чисел и трудности вычисления дискретных логарифмов.

В криптосистеме RSA открытый ключ $K_в$, секретный ключ $k_в$, сообщение M и криптограмма C принадлежат множеству целых чисел

$$Z_N = \{0, 1, 2, ..., N-1\}, \qquad (2.5)$$

где N – модуль:

$$N = P*Q \qquad (2.6)\ .$$

Здесь P и Q – случайные большие простые числа. Для обеспечения максимальной безопасности выбирают P и Q равной длины и хранят в секрете.

Множество Z_N с операциями сложения и умножения по модулю N образует арифметику по модулю N.

Открытый ключ $K_в$ выбирают случайным образом так, чтобы выполнялись условия:

$$1< K_в \le \varphi(N), \quad НОД(K_в, \varphi(N)) = 1, \qquad (2.7)$$

$$\varphi(N)=(P-1)(Q-1), \qquad (2.8)$$

где $\varphi(N)$ – функция Эйлера.

Функция Эйлера $\varphi(N)$ указывает количество положительных целых чисел в интервале от 1 до N, которые взаимно просты с N.

Второе из указанных выше условий означает, что открытый ключ $K_в$ и функция Эйлера $\varphi(N)$ должны быть взаимно простыми.

Далее, используя расширенный алгоритм Евклида, вычисляют секретный ключ $k_в$, такой, что

$$k_в * K_в \equiv 1 \pmod{\varphi(N)} \qquad (2.9)$$

или

$$k_в = K_в^{-1} \pmod{(P-1)(Q-1)}.$$

Это можно осуществить, так как получатель B знает пару простых чисел (P,Q) и может легко найти $\varphi(N)$. Заметим, что $k_в$ и N должны быть взаимно простыми.

Открытый ключ $K_в$ используют для шифрования данных, а секретный ключ $k_в$ – для расшифрования.

Преобразование шифрования определяет криптограмму C через пару (открытый ключ $K_в$, сообщение M) в соответствии со следующей формулой:

$$C = E_{K_в}(M) = E_B(M) = M^{K_в} \pmod{N}. \qquad (2.10)$$

В качестве алгоритма быстрого вычисления значения C используют ряд последовательных возведений в квадрат целого M и умножений на M с приведением по модулю N.

Обращение функции $C = M^{K_в} \pmod{N}$, т.е. определение значения M по известным значениям C, $K_в$ и N, практически не осуществимо при $N \approx 2^{512}$.

Однако обратную задачу, т.е. задачу расшифрования криптограммы C, можно решить, используя пару (секретный ключ $k_в$, криптограмма C) по следующей формуле:

$$M = D_{k_в}(C) = D_B(C) = C^{k_в} \pmod{N}. \qquad (2.11)$$

Процесс расшифрования можно записать так:

$$D_B(E_B(M)) = M. \qquad (2.12)$$

Подставляя в (2.12) значения (2.10) и (2.11), получаем:

$(M^{K_в})^{k_в} = M \pmod N$

или

$$M^{K_в k_в} = M \pmod N. \qquad (2.13)$$

Величина $\varphi(N)$ играет важную роль в теореме Эйлера, которая утверждает, что если НОД (x, N) =1, то

$x^{\varphi(N)} \equiv 1 \pmod N$,

или в несколько более общей форме

$$x^{n\cdot\varphi(N)+1} \equiv x \pmod N. \qquad (2.14)$$

Сопоставляя выражения (2.13) и (2.14), получаем

$K_в * k_в = n * \varphi(N) + 1$

или, что то же самое,

$K_в * k_в \equiv 1 \pmod{\varphi(N)}$.

Именно поэтому для вычисления секретного ключа $k_в$ используют соотношение (2.9).

Таким образом, если криптограмму

$C = M^{K_в} \pmod N$

возвести в степень $k_в$, то в результате восстанавливается исходный открытый текст M, так как

$(M^{K_в})^{k_в} = M^{K_в k_в} = M^{n\cdot\varphi(N)+1} \equiv M \pmod N$.

Таким образом, получатель B, который создает криптосистему, защищает два параметра: 1) секретный ключ $k_в$ и 2) пару чисел (P,Q), произведение которых дает значение модуля N. С другой стороны, получатель B открывает значение модуля N и открытый ключ $K_в$.

Противнику известны лишь значения $K_в$ и N. Если бы он смог разложить число N на множители P и Q, то он узнал бы "потайной ход" – тройку чисел $\{P,Q,K_в\}$, вычислил значение функции Эйлера

$\varphi(N) = (P-1)(Q-1)$

и определил значение секретного ключа $k_в$.

Однако, как уже отмечалось, разложение очень большого N на множители вычислительно не осуществимо (при условии, что длины выбранных P и Q составляют не менее 100 десятичных знаков).

9.4. Процедуры шифрования и расшифрования в криптосистеме RSA

Предположим, что пользователь A хочет передать пользователю B сообщение в зашифрованном виде, используя криптосистему RSA. В таком случае пользователь A выступает в роли отправителя сообщения, а пользователь B – в роли получателя. Как отмечалось выше, криптосистему RSA должен сформировать получатель сообщения, т.е. пользователь B. Рассмотрим последовательность действий пользователя B и пользователя A.

1. Пользователь B выбирает два произвольных больших простых числа $P \neq Q$.
2. Пользователь B вычисляет значение модуля $N = P * Q$.
3. Пользователь B вычисляет функцию Эйлера

$\varphi(N) = (P-1)(Q-1)$

и выбирает случайным образом значение открытого ключа $K_в$ с учетом выполнения условий:

$1 < K_в \le \varphi(N)$, НОД $(K_в, \varphi(N)) = 1$.

4. Пользователь B вычисляет значение секретного ключа $k_в$, используя расширенный алгоритм Евклида при решении сравнения

$k_в \equiv K_в^{-1} \pmod{\varphi(N)}$.

5. Пользователь B пересылает пользователю A пару чисел $(N, K_в)$ по незащищенному каналу.

Если пользователь A хочет передать пользователю B сообщение M, он выполняет следующие шаги.

6. Пользователь А разбивает исходный открытый текст М на блоки, каждый из которых может быть представлен в виде числа
$M_i = 0, 1, 2, ..., N - 1$.
7. Пользователь А шифрует текст, представленный в виде последовательности чисел M_i по формуле

$$C_i = M_i^{K_в} \pmod N$$

и отправляет криптограмму
$C_1, C_2, C_3, ..., C_i, ...$
пользователю В.
8. Пользователь В расшифровывает принятую криптограмму
$C_1, C_2, C_3, ..., C_i, ...,$
используя секретный ключ $k_в$, по формуле

$$M_i = C_i^{k_в} \pmod N.$$

В результате будет получена последовательность чисел M_i, которые представляют собой исходное сообщение М. Чтобы алгоритм RSA имел практическую ценность, необходимо иметь возможность без существенных затрат генерировать большие простые числа, уметь оперативно вычислять значения ключей $K_в$ и $k_в$.

Лабораторная работа №3

ИЗУЧЕНИЕ ПОСИМВОЛЬНОГО ШИФРОВАНИЯ НА ОСНОВЕ КРИПТОСИСТЕМЫ RSA

1. Цель работы

Изучить особенности посимвольного шифрования на основе криптосистемы RSA.

2. Рекомендуемые источники

1. Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях /Под ред. В.Ф. Шаньгина.- 2-е изд.. перераб. и доп..-М.: Радио и связь, 2001, с. 131-138.
2. Соколов А.В., Шаньгин В.Ф. Защита информации в распределенных корпоративных сетях и системах.- М.: ДМК Пресс, 2002, с. 204-206.
3. Б. Шнайер. Прикладная криптография.-М.: Триумф, 2002, с. 521-530.
4. Методические указания к проведению лабораторной работы «Обмен зашифрованными сообщениями с помощью программы PGP» сост. Алексеев А.П., ПГАТИ. 2002.

3. Подготовка к работе

1. Ознакомиться с алгоритмом шифрования по одному из рекомендованных источников /1-3/.
2. Ознакомиться с содержанием данной методической разработки.
3. Подготовить бланк отчета, который должен содержать:
   - цель работы;
   - основные математические соотношения формирования ключевой информации.

4. Контрольные вопросы

1. Сущность посимвольного шифрования на основе криптосистемы RSA.
2. Особенности посимвольного шифрования на основе RSA с использованием кода ASCII.
3. Достоинства посимвольного шифрования.
4. Недостатки посимвольного шифрования.
5. Сущность поточного шифрования.
6. Особенности блочного шифрования.
7. Сущность систем шифрования с обратной связью.
8. Достоинства блочного шифрования.

5. Содержание работы

1. Ознакомиться с основными особенностями посимвольного шифрования.
2. Изучить особенности кодирования буквенно-цифровых текстов на основе международного стандарта ASCII.
3. Произвести установку ключей в криптосистеме RSA.
4. Произвести шифрование и расшифрование произвольного текста посимвольным методом в криптосистеме RSA.

6. Содержание отчета

1. Цель работы.
2. Основные математические соотношения формирования ключевой информации в криптосистеме RSA.
3. По результатам выполнения п.5.4 представьте строки открытого и шифрованного текстов, располагая соответствующие буквы обоих текстов одну под другой.
4. Выводы о свойствах и эффективности асимметричного посимвольного шифрования.

7. Методические указания к выполнению работы

В лабораторной работе используется метод посимвольного шифрования открытого буквенно-цифрового текста произвольной длины. Этот текст представляется в коде ASCII, который преобразуется в зашифрованный текст также в коде ASCII той же длины.

В кодовой таблице международного стандарта для кодирования текстовой информации ASCII (American Standard Code for Information Interchange) зарезервировано 128 7-ми разрядных кодов для кодирования символов латинского алфавита, цифр, знаков препинания. Для кодирования символов национальных алфавитов используются расширения кодовой таблицы ASCII в виде 8-ми разрядных кодов от 128 до 255. Отсутствие согласованных стандартов привело к появлению различных кодовых таблиц для кодирования русскоязычных текстов:

- альтернативная кодовая таблица CP-866;
- международный стандарт ISO 8859;
- кодовая таблица фирмы Microsoft CP-1251 (кодировка Windows);
- кодовая таблица, применяемая в ОС Unix KOI 8.

В табл.3-1 представлены кодировки по стандарту CP-1251 (Windows). Реализованная в настоящей работе программа использует эту таблицу.

Реализованная для данной работы программа, учитывает тот факт, что клавиши типовой русифицированной клавиатуры не используют начальную часть кода ASCII (от 0 до 31). За счет этого общее количество используемых кодов расширенной таблицы ASCII снижается с 256 до 224.

Таблица 3.1 – Кодовая таблица Windows (CP-1251)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0				©	Ё	§	Є	·		°						
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1																
	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2		!	"	#	$	%	&		(	)	*	+				/
	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}]	~	□
	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
8	Ъ	Ѓ	‚	ѓ	„	…	†	‡		‰	Љ	‹	Њ	Ќ	Ћ	Џ
	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	ђ	‘	’	"	"		–	—		™	љ	›	њ	ќ	ћ	џ
	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
A		Ў	ў	J	¤	Ґ	¦	§	Ё	©	Є	«		-	®	Ї
	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
B	°	±	І	і	ґ	µ		·	ё	№	є	»	j	S	s	ї
	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
C	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
D	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	▪ 223
E	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
F	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Искусственно (программно) сдвигая область этих кодов в сторону малых значений (до зашифрования) и восстанавливая истинное значение после расшифрования, а также при выводе на экран шифрованных значений, можно использовать более широкий диапазон значений для n. В

этом случае n может принимать любое значение от 225 до 256, причем n – это произведение двух простых чисел.

Однако, такое решение приводит к некоторой потере информативности. Каждый раз, когда символы шифрованного текста принимают значения в диапазоне от 0 до 31-го, (они не предусмотрены в кодовых таблицах для клавиатуры и дисплея) в окне шифртекста появляются “жирные” вертикальные линии.

8. Лабораторное задание

1.Лабораторная работа выполняется в двух рабочих окнах. Сначала в окне, совмещенном с главным меню, определяются численные значения параметров криптосистемы RSA: модуля **n**, тайного **d** и открытого **e** ключей. Полученные значения нужно записать или запомнить и перейти в окно RSA – ASCII.

2.В этом окне представлены три строки для исходного, зашифрованного и расшифрованного текстов и две кнопки «Расшифровать» и «Зашифровать».

3.В соответствии с номером варианта выберите из табл.3.2 /4/ текст и зашифруйте его для двух различных наборов **n, d** и **e** при **n**= 225 ÷255. Рекомендуемые параметры:

а) n=253, d=13;e=17, p=23, q=11

б) n=247,d=19, e=91, p=19, q=13.

4.Расшифруйте текст и сравните его с исходным.

5.Запишите строки открытого и шифрованного текстов, располагая буквы шифрованного текста под соответствующими буквами открытого текста.

Таблица 3.2-Открытые тексты для шифрования

Вариант	Открытый текст
1	Бородатый человек много рассказывал про этот **ключик**, но я все забыла. Помню только, что нужно отворить им какую-то дверь и это принесет счастье…
2	Пьеро вскочил, размахивая руками. Веди меня к ней… Если ты мне поможешь отыскать Мальвину, я тебе открою тайну золотого **ключика**…
3	-Как! – закричал Буратино радостно.- Ты знаешь тайну золотого **ключика**? -Знаю, где **ключик** лежит, как его достать, знаю, что им нужно открыть одну дверцу…
4	Черепаха позеленела от злости и сказала мне: - На дне пруда лежит волшебный **ключик**…Я знаю одного человека, - он готов сделать все на свете, чтобы получить этот **ключик**…
5	-**Ключик** на дне озера…Мы никогда не увидим счастья…-А это ты видел?- крикнул ему в ухо Буратино. И, вытащив из кармана **ключик**, повертел им перед носом Пьеро.-Вот он!
6	Буратино сказал: - Я все-таки хочу,во что бы то ни стало, узнать у Карабаса Барабаса, где эта дверца, которую открывает золотой **ключик**. За дверцей хранится что-нибудь замечательное, удивительное…И оно должно принести нам счастье.
7	Он только теперь понял, как дороги ему друзья. Пусть Мальвина занимается воспитанием, пусть Пьеро хоть тысячу раз подряд читает стишки, - Буратино отдал бы даже золотой **ключик**, чтобы увидеть снова
	друзей.
8	-Эта дверца и этот золотой **ключик,**- проговорил Карло, - сделаны очень давно каким-то искусным мастером. Посмотрим, что спрятано за дверцей.

9. Общие сведения

9.1. Блочные и поточные шифры

Проектирование алгоритмов шифрования данных основано на рациональном выборе функций, преобразующих исходные (незашифрованные) сообщения в шифртекст. Идея непосредственного применения такой функции ко всему сообщению реализуется очень редко. Практически все применяемые криптографические методы связаны с разбиением сообщения на большое число фрагментов (или знаков) фиксированного размера, каждый из которых шифруется отдельно. Такой подход существенно упрощает задачу шифрования, так как сообщения обычно имеют различную длину.

Различают три основных способа шифрования: поточные шифры, блочные шифры и блочные шифры с обратной связью. Для классификации методов шифрования данных следует выбрать некоторое количество характерных признаков, которые можно применить для установления различий между этими методами. Будем полагать, что каждая часть или каждый знак сообщения шифруется отдельно в заданном порядке.

Можно выделить следующие характерные признаки методов шифрования данных.

- Выполнение операций с отдельными битами или блоками. Известно, что для некоторых методов шифрования знаком сообщения, над которым

производят операции шифрования, является отдельный бит, тогда как другие методы оперируют конечным множеством битов, обычно называемым блоком.

- Зависимость или независимость функции шифрования от результатов шифрования предыдущих частей сообщения.
- Зависимость или независимость шифрования отдельных знаков от их положения в тексте. В некоторых методах знаки шифруются с использованием одной и той же функции независимо от их положения в сообщении, а в других методах, например при поточном шифровании, различные знаки сообщения шифруются с учетом их положения в сообщении. Это свойство называют позиционной зависимостью или независимостью шифра.
- Симметрия или асимметрия функции шифрования. Эта важная характеристика определяет существенное различие между обычными симметричными (одноключевыми) криптосистемами и асимметричными (двухключевыми) криптосистемами с открытым ключом. Основное различие между ними состоит в том, что в асимметричной криптосистеме знания ключа шифрования (или расшифрования) недостаточно для раскрытия соответствующего ключа расшифрования (или шифрования).

В табл.3.3 приведены типы криптосистем и их основные характеристики.

Таблица 3.3- Основные характеристики криптосистем

Тип крипто-системы	Операции с битами или блоками	Зависимость от предыдущих знаков	Позиционная зависимость	Наличие симметрии функции шифрования
Поточного шифрования	Биты	Не зависит	Зависит	Симметричная
Блочного шифрования	Блоки	Не зависит	Не зависит	Симметричная или несимметричная
С обратной связью по шифртексту	Биты или блоки	Зависит	Не зависит	Симметричная

Поточное шифрование состоит в том, что биты открытого текста складываются по модулю 2 с битами псевдослучайной последовательности. К достоинствам поточных шифров относятся высокая скорость шифрования, относительная простота реализации и отсутствие размножения ошибок. Недостатком является необходимость передачи информации синхронизации перед заголовком сообщения, которая должна быть принята до расшифрования любого сообщения. Это обусловлено тем, что если два различных сообщения шифруются на одном и том же ключе, то для расшифрования этих сообщений требуется одна и та же псевдослучайная последовательность. Такое положение может создать угрозу криптостойкости системы. Поэтому часто используют дополнительный, случайно выбираемый ключ сообщения, который передается в начале сообщения и применяется для модификации ключа шифрования. В результате разные сообщения будут шифроваться с помощью различных последовательностей.

Поточные шифры широко применяются для шифрования преобразованных в цифровую форму речевых сигналов и цифровых данных, требующих оперативной доставки потребителю информации. До недавнего времени такие применения были преобладающими для данного метода шифрования. Это обусловлено, в частности, относительной простотой проектирования и реализации генераторов хороших шифрующих последовательностей.

Но самым важным фактором, конечно, является отсутствие размножения ошибок в поточном шифре. Стандартным методом генерирования последовательностей для поточного шифрования является метод, применяемый в стандарте шифрования DES в режиме обратной связи по выходу (режим OFB).

При *блочном шифровании* открытый текст сначала разбивается на равные по длине блоки, затем применяется зависящая от ключа функция шифрования для преобразования блока открытого текста длиной m бит в блок шифртекста такой же длины. Достоинством блочного шифрования является то, что каждый бит блока шифртекста зависит от значений всех битов соответствующего блока открытого текста, и никакие два блока открытого текста не могут быть представлены одним и тем же блоком шифртекста. Алгоритм блочного шифрования может использоваться в различных режимах. Четыре режима шифрования алгоритма DES фактически применимы к любому блочному шифру: режим прямого шифрования или шифрования с использованием электронной книги кодов ECB (Electronic code Book), шифрование со сцеплением блоков шифртекста CBC (Cipher block chaining), шифрование с обратной связью по шифртексту CFB (Cipher feedback) и шифрование с обратной связью по выходу OFB (Output feedback).

Основным достоинством прямого блочного шифрования ECB является то, что в хорошо спроектированной системе блочного шифрования небольшие изменения в шифртексте вызывают большие и непредсказуемые изменения в соответствующем открытом тексте, и наоборот. Вместе с тем применение блочного шифра в данном режиме имеет серьезные недостатки. Первый из них заключается в том, что вследствие детерминированного характера шифрования при

фиксированной длине блока 64 бита можно осуществить криптоанализ шифртекста "со словарем" в ограниченной форме. Это обусловлено тем, что идентичные блоки открытого текста длиной 64 бита в исходном сообщении представляются идентичными блоками шифртекста, что позволяет криптоаналитику сделать определенные выводы о содержании сообщения. Другой потенциальный недостаток этого шифра связан с размножением ошибок. Результатом изменения только одного бита в принятом блоке шифртекста будет неправильное расшифрование всего блока. Это, в свою очередь, приведет к появлению искаженных битов (от 1 до 64) в восстановленном блоке исходного текста.

Из-за отмеченных недостатков блочные шифры редко применяются в указанном режиме для шифрования длинных сообщений. Однако в финансовых учреждениях, где сообщения часто состоят из одного или двух блоков, блочные шифры широко используют в режиме прямого шифрования. Такое применение обычно связано с возможностью частой смены ключа шифрования, поэтому вероятность шифрования двух идентичных блоков открытого текста на одном и том же ключе очень мала.

Криптосистема с открытым ключом также является системой блочного шифрования и должна оперировать блоками довольно большой длины. Это обусловлено тем, что криптоаналитик знает открытый ключ шифрования и мог бы заранее вычислить и составить таблицу соответствия блоков открытого текста и шифртекста. Если длина блоков мала, например 30 бит, то число возможных блоков не слишком большое (при длине 30 бит это $2^{30} = \sim 10^9$), и может быть составлена полная таблица, позволяющая моментально расшифровать любое сообщение с использованием известного открытого ключа.

Асимметричные криптосистемы с открытым ключом подробно разбираются в следующей главе.

Наиболее часто блочные шифры применяются *в системах шифрования с обратной связью.* Системы шифрования с обратной связью встречаются в различных практических вариантах. Как и при блочном шифровании, сообщения разбивают на ряд блоков, состоящих из m бит. Для преобразования этих блоков в блоки шифртекста, которые также состоят из m бит, используются специальные функции шифрования. Однако если в блочном шифре такая функция зависит только от ключа, то в блочных шифрах с обратной связью она зависит как от ключа, так и от одного или более предшествующих блоков шифртекста.

Практически важным шифром с обратной связью является шифр со сцеплением блоков шифртекста CBC. В этом случае m бит предыдущего шифртекста суммируются по модулю 2 со следующими m битами открытого текста, а затем применяется алгоритм блочного шифрования под управлением ключа для получения следующего блока шифртекста. Еще один вариант шифра с обратной связью получается из стандартного режима CFB алгоритма DES, т.е. режима с обратной связью по шифртексту.

Достоинством криптосистем блочного шифрования с обратной связью является возможность применения их для обнаружения манипуляций сообщениями, производимых активными перехватчиками. При этом используется факт размножения ошибок в таких шифрах, а также способность этих систем легко генерировать код аутентификации сообщений. Поэтому системы шифрования с обратной связью используют не только для шифрования сообщений, но и для их аутентификации. Криптосистемам блочного шифрования с обратной связью свойственны некоторые

недостатки. Основным из них является размножение ошибок, так как один ошибочный бит при передаче может вызвать ряд ошибок в расшифрованном тексте. Другой недостаток связан с тем, что разработка и реализация систем шифрования с обратной связью часто оказываются более трудными, чем систем поточного шифрования.

На практике для шифрования длинных сообщений применяют поточные шифры или шифры с обратной связью. Выбор конкретного типа шифра зависит от назначения системы и предъявляемых к ней требований.

Лабораторная работа №4

ИЗУЧЕНИЕ АЛГОРИТМА ДИФФИ-ХЕЛЛМАНА
ОТКРЫТОГО РАСПРЕДЕЛЕНИЯ КЛЮЧЕЙ

1. Цель работы

Изучить основные принципы и процедурные аспекты алгоритма Диффи-Хеллмана открытого распределения ключей.

2. Рекомендуемые источники

1.Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях /Под ред. В.Ф. Шаньгина.- 2-е изд.. перераб. и доп..-М.: Радио и связь, 2001, с. 182-199.

2.Соколов А.В., Шаньгин В.Ф. Защита информации в распределенных корпоративных сетях и системах.- М.: ДМК Пресс, 2002, с. 269-286.

3.Б.Шнайер. Прикладная криптография.-М.: Триумф, 2002, с. 573-576.

3. Подготовка к работе

1.Ознакомиться с алгоритмом открытого распределения ключей по одному из рекомендованных источников/1-3/.
2.Ознакомиться с содержанием данной методической разработки.
3.Подготовить бланк отчета, который должен содержать:
- цель работы;
- основные математические соотношения для обоснования алгоритма.
- заготовку таблицы, в которую будет заноситься ключевая информация в процессе изучения алгоритма.

4. Контрольные вопросы

1.Сущность процедуры управления криптографическими ключами.
2.Назначение и особенности генерации ключей.
3.Назначение и особенности хранения ключей.
4.Концепция иерархии ключей.
5.Назначение и особенности распределения ключей.
6.Централизованное распределение ключей.
7.Протокол аутентификации и распределения ключей для симметричных криптосистем.
8.Протокол для асимметричных криптосистем с использованием сертификатов открытых ключей.
9.Алгоритм открытого распределения ключей Диффи-Хеллмана.
10. Схема реализации алгоритма Диффи-Хеллмана.
11. Приведите математическое доказательство возможности формирования у абонентов двух одинаковых ключей без передачи секретной информации.

5. Содержание работы

1.Ознакомиться с постановкой задачи управления криптографическими ключами.
2.Ознакомиться с математическими соотношениями, лежащими в основе алгоритма Диффи-Хеллмана.
3.Изучить основные этапы алгоритма.
4.Произвести установку открытых и закрытых ключей двух абонентов.
5.Наблюдать поведение алгоритма при изменении соответствующих параметров.

6. Содержание отчета

1.Цель работы.
2.Структурная схема алгоритма распределения ключей Диффи-Хеллмана.

3. Основные математические соотношения, предписываемые алгоритмом Диффи-Хеллмана.
4. Привести значения предварительно устанавливаемых параметров и значения, устанавливаемые и возникающие в процессе каждого сеанса.

7. Методические указания к выполнению работы

Для реализации любой одноключевой криптосистемы необходим обмен секретными ключами шифрования. Передача таких ключей по открытым каналам не имеет смысла, так как может стать легкой добычей злоумышленника. Передавать ключи по закрытым каналам тоже опасно, так как в этом случае однократная компрометация канала рассекречивает весь дальнейший обмен. Поэтому единственным надежным средством доставки ключевой информации для особо важного обмена до недавнего времени считалась фельдъегерская почта. В современных условиях, например, для миллионов пользователей Internet такая система неприемлема.

В 1976 году Диффи и Хеллман /1,2,3/ опубликовали алгоритм открытого распределения ключей, который позволяет двум абонентам путем обмена открытыми сообщениями сформировать одинаковые секретные ключи для своей одноключевой (симметричной) системы шифрования.

В основе метода лежит свойство односторонней функции **y=f(x),** в которой, зная **x** сравнительно легко вычислить **y**, но вычисление **x** при известном **y** выливается в задачу, сопоставимую с полным перебором.

Суть алгоритма Диффи-Хеллмана состоит в следующем.

Предположим, что два пользователя **A** и **B** хотят организовать защищенный коммуникационный канал.

1. Обе стороны заранее уславливаются о модуле **n** (**n** должно быть простым числом) и примитивном элементе **c** **(1<c < n).** Эти два целых числа **n** и **c** могут не храниться в секрете. Как правило, эти значения являются общими для всех пользователей системы.

2. Затем пользователи **A** и **B** независимо друг от друга выбирают собственные секретные ключи X_a и X_b (X_a и X_b – случайные большие целые числа, которые хранятся пользователями **A** и **B** в секрете).

3. Далее пользователь **A** вычисляет открытый ключ $Y_a = (c^{Xa}) \bmod n$,
а пользователь **B** – открытый ключ $Y_b = (c^{Xb}) \bmod n$.

4. Затем стороны обмениваются вычисленными значениями открытых ключей Y_a и Y_b по любому, даже совсем незащищенному каналу, т.к. открытые ключи не являются секретом. (Мы считаем, что все данные, передаваемые по незащищенному каналу связи, могут быть перехвачены злоумышленником).

5. Далее пользователи **A** и **B** вычисляют общий секретный ключ, используя следующие соотношения:

пользователь **A**: $K_{ab} = (Y_b^{Xa}) \bmod n = (c^{XbXa}) \bmod n$,

пользователь **B**: $K_{ba} = (Y_a^{Xb}) \bmod n = (c^{XaXb}) \bmod n$.

Очевидно, что $K_{ab} = K_{ba}$.

Ключ **К** может использоваться в качестве общего секретного ключа (ключа шифрования), например, в таких симметричных криптосистемах, как DES, ГОСТ 28147-89, AES и др.

Возможные действия злоумышленника.

Перехватив значения **n, c**, Y_a и Y_b, криптоаналитик тоже хотел бы определить значения ключа **К**. Очевидный путь для решения этой задачи состоит в вычислении такого значения X_a по **n, c** и Y_a, что $(c^{Xa}) \bmod n = Y_a$ (поскольку в этом случае, вычислив X_a, можно найти $K = (Y_b^{Xa}) \bmod n$). Однако нахождение X_a по **n, c** и Y_a – это задача

нахождения дискретного логарифма в конечном поле, которая считается неразрешимой.

Выбор значений **n** и **c** может иметь существенное влияние на безопасность этой системы. Модуль **n** должен быть большим и простым числом. Число **(n-1)/2** также должно быть простым числом. Число **c** желательно выбирать таким, чтобы оно было примитивным элементом множества ненулевых элементов **Zn, т.е. $\{c,c^2,\ldots c^{n-1}\}$ = Zn – {0}.**

В принципе достаточно, чтобы **c** генерировало большую подгруппу мультипликативной группы по **mod n**.

Алгоритм открытого распределения ключей Диффи-Хеллмана позволяет обойтись без защищенного канала для передачи ключей. Однако, работая с этим алгоритмом, необходимо иметь гарантию того, что пользователь **A** получил открытый ключ именно от пользователя **B**, и наоборот. Эта проблема решается с помощью электронной подписи, которой подписываются сообщения об открытом ключе.

Метод Диффи-Хеллмана дает возможность шифровать данные при каждом сеансе связи на новых ключах. Это позволяет не хранить секреты на дискетах или других носителях. Не следует забывать, что любое хранение секретов повышает вероятность попадания их в руки конкурентов или противника.

Преимущество метода Диффи-Хеллмана по сравнению с методом RSA заключается в том, что формирование общего секретного ключа происходит в сотни раз быстрее. В системе RSA генерация новых секретных и открытых ключей основана на генерации новых простых чисел, что занимает много времени. Кроме того, сам процесс зашифровывания и расшифровывания информации в симметричных (одноключевых) системах протекает значительно быстрее, чем в асимметричных

(двухключевых), а система Диффи-Хеллмана как раз и ориентирована на то, что сам процесс информационного обмена будет производиться в симметричных системах.

8. Лабораторное задание

1.Произвести рапределение ключей при различных значениях аргументов **n**, **c**, $\mathbf{X_a}$ и $\mathbf{X_b}$.

2.Зафиксировать по секундомеру длительность выполнения основных процедур при различной длительности аргументов.

3.Выход на рабочее поле алгоритма производится из Главного меню вызовом пункта «Диффи-Хеллман».

4.Установку открытых ключей **n** и **c** произвести с учетом указанных ограничений. Вышеупомянутые требования к значениям **n** и **c** (**n** –простое, а **c** - примитивный элемент поля **GF(n)**) не являются строго обязательными.

5.Понаблюдайте за поведением системы при значениях **c=1** и **c=n-1**. Объясните результаты.

9. Общие сведения

9.1. Управление криптографическими ключами

Любая криптографическая система основана на использовании криптографических ключей. В симметричной криптосистеме отправитель и получатель сообщения используют один и тот же секретный ключ. Этот ключ должен быть неизвестен всем остальным и должен периодически обновляться одновременно у отправителя и получателя. Процесс распределения (рассылки) секретных ключей между участниками информационного обмена в симметричных криптосистемах имеет весьма сложный характер.

Асимметричная криптосистема предполагает использование двух ключей – открытого и личного

(секретного). Открытый ключ можно разглашать, а личный надо хранить в тайне. При обмене сообщениями необходимо пересылать только открытый ключ. Важным требованием является обеспечение подлинности отправителя сообщения. Это достигается путем взаимной аутентификации участников информационного обмена.

Под *ключевой информацией* понимают совокупность всех действующих в сети ключей. Если не обеспечено достаточно надежное управление ключевой информацией, то, завладев ею, злоумышленник получает неограниченный доступ ко всей информации.

Управление ключами – информационный процесс, включающий реализацию следующих основных функций:

- генерация ключей;
- хранение ключей;
- распределение ключей.

9.1.1. Генерация ключей

Безопасность любого криптографического алгоритма определяется используемым криптографическим ключом. Добротные криптографические ключи должны иметь достаточную длину и случайные значения битов.

Для получения ключей используются аппаратные и программные средства генерации случайных значений ключей. Как правило, применяют датчики псевдослучайных чисел (ПСЧ). Однако степень случайности генерации чисел должна быть достаточно высокой. Идеальными генераторами являются устройства на основе "натуральных" случайных процессов, например на основе *белого радиошума.*

В сети со средними требованиями защищенности вполне приемлемы программные генераторы ключей, которые вычисляют ПСЧ как сложную функцию от текущего времени и (или) числа, введенного пользователем.

Один из методов генерации сеансового ключа для симметричных криптосистем описан в стандарте ANSI X9.17. Он предполагает использование криптографического алгоритма DES (хотя можно применить и другие симметричные алгоритмы шифрования).

Обозначения:

E_K (X) – результат шифрования алгоритмом DES значения X;

K – ключ, зарезервированный для генерации секретных ключей;

V_0 – секретное 64-битовое начальное число;

T – временная отметка.

Схема генерации случайного сеансового ключа R_i в соответствии со стандартом ANSI X 9.17 показана на рис.4.1. Случайный ключ R_i генерируют, вычисляя значение

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

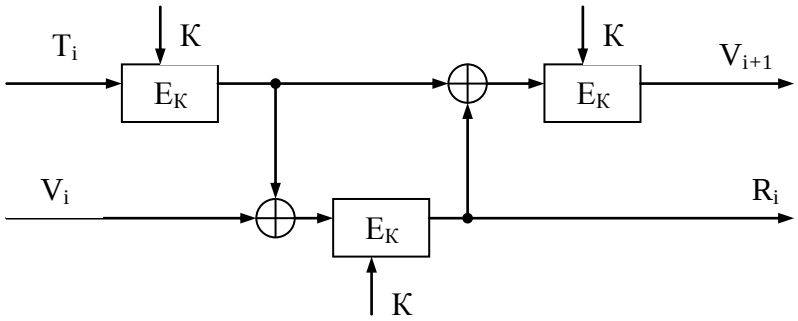

Рисунок4. 1- Схема генерации случайного ключа R_i в соответствии со стандартом ANSI X9.17

Следующее значение V_{i+1} вычисляют так:

$$V_{i+1} = E_K (E_K (T_i) \oplus R_i).$$

Если необходим 128-битовый случайный ключ, генерируют пару ключей R_i, R_{i+1} и объединяют их вместе.

Если ключ не меняется регулярно, это может привести к его раскрытию и утечке информации. Регулярную замену ключа можно осуществить, используя процедуру модификации ключа.

Модификация ключа – это генерирование нового ключа из предыдущего значения ключа с помощью односторонней (однонаправленной) функции. Участники информационного обмена разделяют один и тот же ключ и одновременно вводят его значение в качестве аргумента в одностороннюю функцию, получая один и тот же результат. Затем они берут определенные биты из этих результатов, чтобы создать новое значение ключа.

Процедура модификации ключа работоспособна, но надо помнить, что новый ключ безопасен в той же мере, в какой был безопасен прежний ключ. Если злоумышленник сможет добыть прежний ключ, то он сможет выполнить процедуру модификации ключа.

Генерация ключей для асимметричных криптосистем с открытыми ключами много сложнее, потому что эти ключи должны обладать определенными математическими свойствами (они должны быть очень большими и простыми и т.д.).

9.1.2. Хранение ключей

Под *функцией хранения ключей* понимают организацию их безопасного хранения, учета и удаления. Ключ является самым привлекательным для злоумышленника объектом, открывающим ему путь к конфиденциальной информации. Поэтому вопросам безопасного хранения ключей следует уделять особое внимание. Секретные ключи никогда не

должны записываться в явном виде на носителе, который может быть считан или скопирован.

Носители ключевой информации.

Ключевой носитель может быть технически реализован различным образом на разных носителях информации - магнитных дисках, устройствах хранения ключей типа TOUCH MEMORY, пластиковых картах и т.д..

Магнитные диски представляют собой распространенный тип носителя ключевой информации. Применение магнитного диска (МД) в качестве носителя ключа позволяет реализовать необходимое свойство отчуждаемости носителя ключа от защищенной компьютерной системы, т.е. осуществлять временное изъятие МД из состава технических средств компьютерной системы. Особенно целесообразно использование в качестве ключевых носителей съемных накопителей – гибких магнитных дисков, съемных магнитооптических носителей и т.д..

Основное преимущество МД по сравнению с другими носителями ключевой информации заключается в том, что оборудование для взаимодействия с МД (дисковод) входит в состав штатных средств компьютера.

Другая важная особенность, определяющая широкое распространение МД, - стандартный формат хранения информации на дисках и стандартные программные средства доступа к дискам. Кроме того, из всех средств хранения ключевой информации гибкие магнитные диски имеют самую низкую стоимость..

Для обеспечения надежного хранения ключевой информации на МД применяют как минимум двукратное резервирование объектов хранения. Это позволяет защитить ключевую информацию от ошибок при

считывании с МД и от сбоев программной и аппаратной части.

Для предотвращения возможности перехвата ключевой информации в процессе ее чтения с МД применяют хранение ключевой информации на МД в зашифрованном виде.

Устройство хранения ключей типа TOUCH MEMORY является относительно новым носителем ключевой информации, предложенным американской компанией Dallas Semiconductor. Носитель информации TOUCH MEMORY (TM) представляет собой энергонезависимую память, размещенную в металлическом корпусе, с одним сигнальным контактом и одним контактом земли. Корпус ТМ имеет диаметр 16,25 мм и толщину 3,1 или 5,89 мм (в зависимости от модификации прибора).

В структуру ТМ входят следующие основные блоки.

1. Постоянное запоминающее устройство (ПЗУ) хранит 64-разрядный код, состоящий из байтового кода типа прибора, 48-битового уникального серийного номера и 8-битовой контрольной суммы. Содержимое ПЗУ уникально и не может быть изменено в течение всего срока службы прибора.
2. Оперативное запоминающее устройство (ОЗУ) емкостью от 128 до 8192 байт содержат практически все модификации ТМ. В одной из модификаций оперативная память аппаратно защищена от несанкционированного доступа.
3. Встроенная миниатюрная литиевая батарейка со сроком службы не менее 10 лет обеспечивает питанием все блоки устройства.

Особенностью технологии хранения и обмена ключевой информации между носителем ТМ и внешними устройствами является сравнительно низкая скорость (обусловленная последовательной передачей данных) и

высокая вероятность сбоя в тракте чтения-записи, обусловленная тем, что контакт устройства ТМ с устройством чтения осуществляется пользователем вручную без дополнительной фиксации (простое касание, что и определило название прибора ТМ). В связи с этим особое значение приобретают вопросы надежного обмена между программами обработки ключевой информации пользователей и носителем ТМ.

В устройстве ТМ конструктивно отработаны вопросы надежности функционирования и вопросы интерфейса со считывающим устройством на основе одного сигнального контакта. Для обеспечения достоверного чтения применяются корректирующие коды, для обеспечения достоверной записи в приборе предусмотрена технология буферизации. При проведении операции записи первоначально вектор передаваемой в ТМ информации помещается в буфер, далее выполняется операция чтения из буфера, затем прочтенная из буфера информация сравнивается с записываемой и в случае совпадения подается сигнал на перенос информации из буфера в память долговременного хранения.

Таким образом, носитель ТМ является микроконтроллерным устройством без собственной вычислительной мощности и с ограниченным объемом хранения, но с достаточно высокими надежностными характеристиками. Поэтому применение ТМ вполне обосновано в случае повышенных требований к надежности носителя ключа и небольшого объема ключевой информации, хранимой в ТМ.

Электронные пластиковые карты становятся в настоящее время наиболее распространенным и универсальным носителем конфиденциальной информации, который позволяет идентифицировать и

аутентифицировать пользователей, хранить криптографические ключи, пароли и коды.

Интеллектуальные карты (смарт-карты), обладающие наибольшими возможностями, эффективно применяются не только для хранения ключевой информации, но и широко используются в электронных платежных системах, в комплексных решениях для медицины, транспорта, связи, образования и т.п. Более подробные сведения об электронных пластиковых картах приводятся в разделе 9.4.

9.1.3. Концепция иерархии ключей.

Любая информация об используемых ключах должна быть защищена, в частности храниться в зашифрованном виде.

Необходимость в хранении и передаче ключей, зашифрованных с помощью других ключей, приводит к концепции *иерархии ключей*. В стандарте ISO 8532 (Banking-Key Management) подробно изложен метод главных/сеансовых ключей (master/session keys). Суть метода состоит в том, что вводится иерархия ключей: главный ключ (ГК), ключ шифрования ключей (КК), ключ шифрования данных (КД).

Иерархия ключей может быть:

- двухуровневой (КК/КД),
- трехуровневой (ГК/КК/КД).

Самым нижним уровнем являются *рабочие или сеансовые КД*, которые используются для шифрования данных, персональных идентификационных номеров (PIN) и аутентификации сообщений. Когда эти ключи надо зашифровать с целью защиты при передаче или хранении, используют ключи следующего уровня – *ключи шифрования ключей*. Ключи шифрования ключей никогда не должны использоваться как сеансовые (рабочие) КД, и наоборот.

Такое разделение функций необходимо для обеспечения максимальной безопасности. Фактически стандарт устанавливает, что различные типы рабочих ключей (например, для шифрования данных, для аутентификации и т.д.) должны всегда шифроваться с помощью различных версий ключей шифрования ключей.

В частности, ключи шифрования ключей, используемые для пересылки ключей между двумя узлами сети, известны также как *ключи обмена между узлами сети* (cross domain keys). Обычно в канале используются два ключа для обмена между узлами сети, по одному в каждом направлении. Поэтому каждый узел сети будет иметь *ключ отправления* для обмена с узлами сети и *ключ получения* для каждого канала, поддерживаемого другим узлом сети.

На верхнем уровне иерархии ключей располагается *главный ключ, мастер-ключ*. Этот ключ применяют для шифрования КК, когда требуется сохранить их на диске. Обычно в каждом компьютере используется только один мастер-ключ.

Мастер-ключ распространяется между участниками обмена неэлектронным способом – при личном контакте, чтобы исключить его перехват и/или компрометацию. Раскрытие противником значения мастер-ключа полностью уничтожает защиту компьютера.

Значение мастер-ключа фиксируется на длительное время (до нескольких недель или месяцев). Поэтому генерация и хранение мастер-ключей являются критическими вопросами криптографической защиты. На практике мастер-ключ компьютера создается истинно случайным выбором из всех возможных значений ключей. Мастер-ключ помещают в защищенный по считыванию и записи и от механических воздействий блок криптографической системы таким образом, чтобы раскрыть значение этого ключа было невозможно. Однако

все же должен существовать способ проверки, является ли значение ключа правильным.

Проблема аутентификации мастер-ключа может быть решена различными путями. Один из способов аутентификации показан на рис. 4.2 .

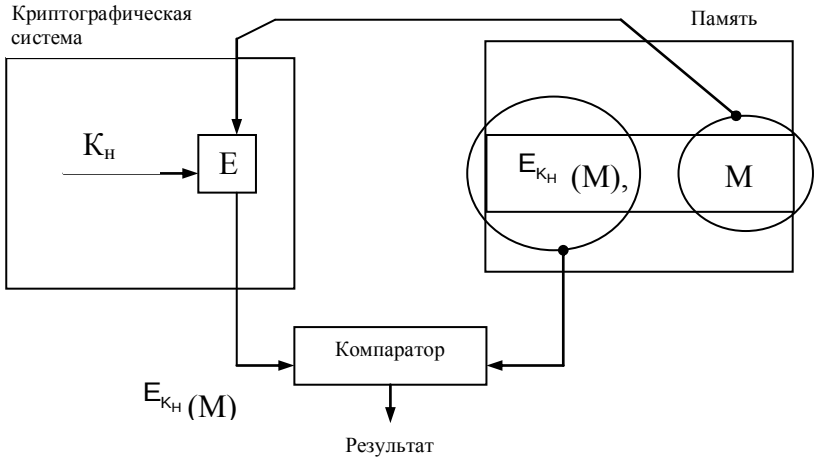

Рисунок 4.2- Схема аутентификации мастер-ключа хост-компьютера

Администратор, получив новое значение мастер-ключа K_H хост-компьютера, шифрует некоторое сообщение M ключом K_H. Пара (криптограмма E_{K_H} (M), сообщение M) помещается в память компьютера. Всякий раз, когда требуется аутентификация мастер-ключа хост-компьютера, берется сообщение M из памяти и подается в криптографическую систему. Получаемая криптограмма сравнивается с криптограммой, хранящейся в памяти. Если они совпадают, считается, что данный ключ является правильным.

Рабочие ключи (например, сеансовый) обычно создаются с помощью псевдослучайного генератора и

могут храниться в незащищенном месте. Это возможно, поскольку такие ключи генерируются в форме соответствующих криптограмм, т.е. генератор ПСЧ выдает вместо ключа K_S его криптограмму $E_{K_н}(K_S)$, получаемую с помощью мастер-ключа хост-компьютера. Расшифровывание такой криптограммы выполняется только перед использованием ключа K_S.

Схема защиты рабочего (сеансового) ключа показана на рис. 4.3. Чтобы зашифровать сообщение M ключом K_S, на соответствующие входы криптографической системы подается криптограмма $E_{K_н}(K_S)$ и сообщение M. Криптографическая система сначала восстанавливает ключ K_S, а затем шифрует сообщение M, используя открытую форму сеансового ключа K_S.

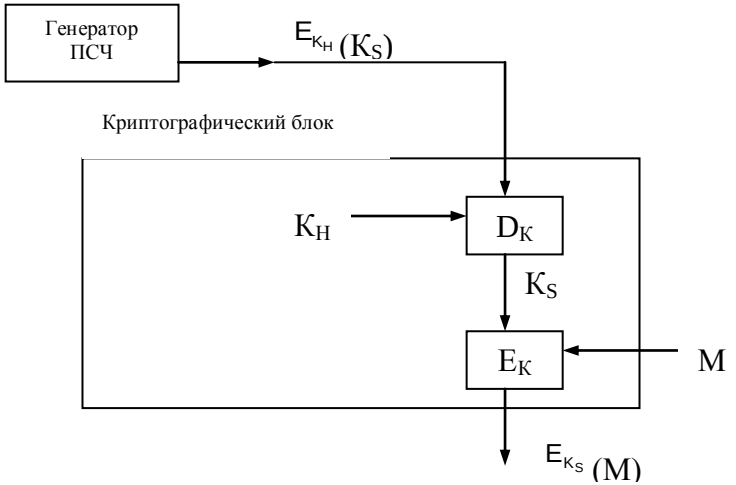

Рисунок4.3- Схема защиты сеансового ключа K_S

Таким образом, безопасность сеансовых ключей зависит от безопасности криптографической системы.

Криптографический блок может быть спроектирован как единая СБИС и помещен в физически защищенное место.

Очень важным условием безопасности информации является периодическое обновление ключевой информации. При этом должны переназначаться как рабочие ключи, так и мастер-ключи. В особо ответственных сетях обновление ключевой информации (сеансовых ключей) желательно делать ежедневно. Вопрос обновления ключевой информации тесно связан с третьим элементом управления ключами – распределением ключей.

9.1.4. Распределение ключей

Распределение ключей – самый ответственный процесс в управлении ключами. К нему предъявляются следующие требования:

- оперативность и точность распределения;
- скрытность распределяемых ключей.

Распределение ключей между пользователями компьютерной сети реализуется двумя способами:

1) использованием одного или нескольких центров распределения ключей;
2) прямым обменом сеансовыми ключами между пользователями сети.

Недостаток первого подхода состоит в том, что центру распределения ключей известно, кому и какие ключи распределены, и это позволяет читать все сообщения, передаваемые по сети. Возможные злоупотребления существенно влияют на защиту. При втором подходе проблема состоит в том, чтобы надежно удостоверить подлинность субъектов сети.

В обоих случаях должна быть обеспечена подлинность сеанса связи. Это можно осуществить, используя механизм запроса-ответа или механизм отметки времени.

Механизм запроса-ответа заключается в следующем. Пользователь А включает в посылаемое сообщение (запрос) для пользователя В непредсказуемый элемент (например, случайное число). При ответе пользователь В должен выполнить некоторую операцию с этим элементом (например, добавить единицу), что невозможно осуществить заранее, поскольку неизвестно, какое случайное число придет в запросе. После получения результата действий пользователя В (ответ) пользователь А может быть уверен, что сеанс является подлинным.

Механизм отметки времени предполагает фиксацию времени для каждого сообщения. Это позволяет каждому субъекту сети определить, насколько старо пришедшее сообщение, и отвергнуть его, если появится сомнение в его подлинности. При использовании отметок времени необходимо установить допустимый временной интервал задержки.

В обоих случаях для защиты элемента контроля используют шифрование, чтобы быть уверенным, что ответ отправлен не злоумышленником и не изменен штемпель отметки времени.

Задача распределения ключей сводится к построению протокола распределения ключей, обеспечивающего:

- взаимное подтверждение подлинности участников сеанса;
- подтверждение достоверности сеанса механизмом запроса-ответа или отметки времени;
- использование минимального числа сообщений при обмене ключами;
- возможность исключения злоупотреблений со стороны центра распределения ключей (вплоть до отказа от него).

В основу решения задачи распределения ключей целесообразно положить принцип отделения процедуры подтверждения подлинности партнеров от процедуры собственно распределения ключей. Цель такого подхода

состоит в создании метода, при котором после установления подлинности участники сами формируют сеансовый ключ без участия центра распределения ключей с тем, чтобы распределитель ключей не имел возможности выявить содержание сообщений.

9.1.5. Распределение ключей с участием центра распределения ключей

При распределении ключей между участниками предстоящего информационного обмена должна быть гарантирована подлинность сеанса связи. Для взаимной проверки подлинности партнеров приемлема *модель рукопожатия*. В этом случае ни один из участников не будет получать никакой секретной информации во время процедуры установления подлинности.

Взаимное установление подлинности гарантирует вызов нужного субъекта с высокой степенью уверенности, что связь установлена с требуемым адресатом и никаких попыток подмены не было. Реальная процедура организации соединения между участниками информационного обмена включает как этап распределения, так и этап подтверждения подлинности партнеров.

При включении в процесс распределения ключей центра распределения ключей (ЦРК) осуществляется его взаимодействие с одним или обоими участниками сеанса с целью распределения секретных или открытых ключей, предназначенных для использования в последующих сеансах связи.

Следующий этап – подтверждение подлинности участников – содержит обмен удостоверяющими сообщениями, чтобы иметь возможность выявить любую подмену или повтор одного из предыдущих вызовов.

Рассмотрим протоколы для симметричных криптосистем с секретными ключами и для асимметричных криптосистем с открытыми ключами. Вызывающий (исходный объект) обозначается через A, а вызываемый (объект назначения) – через B. Участники сеанса A и B имеют уникальные идентификаторы Id_A и Id_B соответственно.

9.1.6. Протокол аутентификации и распределения ключей для симметричных криптосистем

Рассмотрим в качестве примера протокол аутентификации и распределения ключей Kerberos (по-русски – Цербер). Первоначально протокол Kerberos был разработан в Массачусетском Технологическом Институте (США) для проекта Athena. Протокол Kerberos спроектирован для работы в сетях TCP/IP и предполагает участие в аутентификации и распределении ключей третьей доверенной стороны. Kerberos обеспечивает надежную аутентификацию в сети, разрешая законному пользователю доступ к различным машинам в сети. Протокол Kerberos основывается на симметричной криптографии (реализован алгоритм DES, хотя возможно применение и других симметричных криптоалгоритмов). Kerberos разделяет отдельный секретный ключ с каждым субъектом сети, и знание такого секретного ключа равносильно доказательству подлинности субъекта сети.

Основной протокол Kerberos является вариантом протокола аутентификации и распределения ключей Нидхема-Шредера. В основном протоколе Kerberos (версия 5) участвуют две взаимодействующие стороны A и B и доверенный сервер KS (Kerberos Server). Стороны A и B, каждая по отдельности, разделяют свой секретный ключ с сервером KS. Доверенный сервер KS выполняет роль центра распределения ключей ЦРК.

Пусть сторона А хочет получить сеансовый ключ для информационного обмена со стороной В.

Сторона А инициирует фазу распределения ключей, посылая по сети серверу KS идентификаторы Id_A и Id_B:

(1) $A \rightarrow KS$: Id_A, Id_B.

Сервер KS генерирует сообщение с временной отметкой Т, сроком действия L, случайным сеансовым ключом К и идентификатором Id_A. Он шифрует это сообщение секретным ключом, который разделяет со стороной В.

Затем сервер KS берет временную отметку Т, срок действия L, сеансовый ключ К, идентификатор Id_B стороны В и шифрует все это секретным ключом, который разделяет со стороной А. Оба эти зашифрованные сообщения он отправляет стороне А:

(2) $KS \rightarrow A$: $E_A(T, L, K, Id_B)$, $E_B(T, L, K, Id_A)$.

Сторона А расшифровывает первое сообщение своим секретным ключом, проверяет отметку времени Т, чтобы убедиться, что это сообщение не является повторением предыдущей процедуры распределения ключей.

Затем сторона А генерирует сообщение со своим идентификатором Id_A и отметкой времени Т, шифрует его сеансовым ключом К и отправляет стороне В. Кроме того, А отправляет для В сообщение от KS, зашифрованное ключом стороны В:

(3) $A \rightarrow B$: $E_K(Id_A, T)$, $E_B(T, L, K, Id_A)$.

Только сторона В может расшифровать сообщения (3). Сторона В получает отметку времени Т, срок действия L, сеансовый ключ К и идентификатор Id_A. Затем сторона В расшифровывает сеансовым ключом К вторую часть сообщения (3). Совпадение значений Т и Id_A в двух частях сообщения подтверждают подлинность А по отношению к В.

Для взаимного подтверждения подлинности сторона В создает сообщение, состоящее из отметки времени Т плюс 1, шифрует его ключом К и отправляет стороне А:

(4) $B \rightarrow A$: $E_K(T+1)$.

Если после расшифрования сообщения (4) сторона А получает ожидаемый результат, она знает, что на другом конце линии связи находится действительно В.

Этот протокол успешно работает при условии, что часы каждого участника синхронизированы с часами сервера KS. Следует отметить, что в этом протоколе необходим обмен с KS для получения сеансового ключа каждый раз, когда А желает установить связь с В. Протокол обеспечивает надежное соединение объектов А и В при условии, что ни один из ключей не скомпрометирован и сервер KS защищен.

Система Kerberos обеспечивает защиту сети от несанкционированного доступа, базируясь исключительно на программных решениях, и предполагает многократное шифрование передаваемой по сети управляющей информации.

Система Kerberos имеет структуру типа клиент-сервер и состоит из клиентских частей С, установленных на все машины сети (рабочие станции пользователей и серверы), и Kerberos-сервера KS, располагающегося на каком-либо (не обязательно выделенном) компьютере.

Kerberos-сервер, в свою очередь, можно разделить на две части: сервер идентификации AS (Authentication Server) и сервер выдачи разрешений TGS (Ticket Granting Server). Информационными ресурсами, необходимыми клиентам С, управляет сервер информационных ресурсов RS (рис.4.4).

Область действия системы Kerberos распространяется на тот участок сети, все пользователи которого зарегистрированы под своими именами и паролями в базе данных Kerberos-сервера.

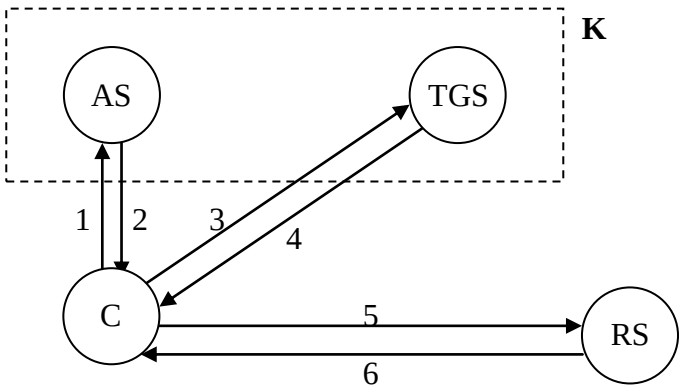

Рисунок 4.4- Схема и шаги протокола Kerberos.

Обозначения:

KS - сервер системы Kerberos;
AS - сервер идентификации;
TGS - сервер выдачи разрешений;
RS - сервер информационных ресурсов;
C - клиент системы Kerberos;
1 : C → AS : - запрос разрешить обратиться к TGS;
2 : AS → C : - разрешение обратиться к TGS;
3 : C → TGS : - запрос на допуск к RS;
4 : TGS → C : - разрешение на допуск к RS;
5 : C → RS : - запрос на получение информационного ресурса от RS;
6 : RS → C : - подтверждение подлинности сервера RS и предоставление информационного ресурса.

Укрупненно процесс идентификации и аутентификации пользователя в системе Kerberos можно списать следующим образом. Пользователь (клиент) С, желая получить доступ к ресурсу сети, направляет запрос серверу

идентификации AS. Последний идентифицирует пользователя с помощью его имени и пароля и выдает разрешение на доступ к серверу выдачи разрешений TGS, который в свою очередь, по запросу клиента С разрешает использование необходимых ресурсов сети с помощью целевого сервера информационных ресурсов RS.

Данная модель взаимодействия клиента с серверами может функционировать только при условии обеспечения конфиденциальности и целостности передаваемой управляющей информации. Без строгого обеспечения информационной безопасности клиент не может отправлять серверам AS, TGS и RS свои запросы и получать разрешения на доступ к обслуживанию в сети. Чтобы избежать возможности перехвата и несанкционированного использования информации, Kerberos применяет при передаче любой управляющей информации в сети сложную систему многократного шифрования с использованием комплекса секретных ключей (секретный ключ клиента, секретный ключ сервера, секретные сеансовые ключи, клиент-сервер).

9.1.7. Протокол для асимметричных криптосистем с использованием сертификатов открытых ключей

В этом протоколе используется идея сертификатов открытых ключей.

Сертификатом открытого ключа С называется сообщение ЦРК, удостоверяющее целостность некоторого открытого ключа объекта. Например, сертификат открытого ключа для пользователя А, обозначаемый C_A, содержит отметку времени Т, идентификатор Id_A и открытый ключ K_A, зашифрованные секретным ключом ЦРК $k_{ЦРК}$, т.е.

$$C_A = E_{k_{ЦРК}} (T, Id_A, K_A).$$

Отметка времени T используется для подтверждения актуальности сертификата и тем самым предотвращает повторы прежних сертификатов, которые содержат открытые ключи и для которых соответствующие секретные ключи несостоятельны.

Секретный ключ $k_{ЦРК}$ известен только менеджеру ЦРК. Открытый ключ $К_{ЦРК}$ известен участникам A и B. ЦРК поддерживает таблицу открытых ключей всех объектов сети, которые он обслуживает.

Вызывающий объект A инициирует стадию установления ключа, запрашивая у ЦРК сертификат своего открытого ключа и открытого ключа участника B:

(1) $A \rightarrow$ ЦРК : I_{d_A}, I_{d_B},

´Вышлите сертификаты ключей A и B´. Здесь I_{d_A} и I_{d_B} – уникальные идентификаторы соответственно участников A и B.

Менеджер ЦРК отвечает сообщением

(2) ЦРК $\rightarrow$ A : $E_{k_{ЦРК}}$ (T, I_{d_A}, $К_A$), $E_{k_{ЦРК}}$ (T, I_{d_B}, $К_B$).

Участник A, используя открытый ключ ЦРК $К_{ЦРК}$, расшифровывает ответ ЦРК, проверяет оба сертификата. Идентификатор I_{d_B} убеждает A, что личность вызываемого участника правильно зафиксирована в ЦРК и $К_B$ – действительно открытый ключ участника B, поскольку оба зашифрованы ключом $k_{ЦРК}$.

Хотя открытые ключи предполагаются известными всем, посредничество ЦРК позволяет подтвердить их целостность. Без такого посредничества злоумышленник может снабдить A своим открытым ключом, который A будет считать ключом участника B. Затем злоумышленник может подменить собой B и установить связь с A, и его никто не сможет выявить.

Следующий шаг протокола включает установление свя-зи A с B:

(3) $A \rightarrow B$: C_A, E_{k_A} (T), E_{K_B} (r_1).

Здесь

C_A – сертификат открытого ключа пользователя A;

E_{k_A} (T) – отметка времени, зашифрованная секретным ключом участника A и являющаяся подписью участника A, поскольку никто другой не может создать такую подпись;

r_1 – случайное число, генерируемое A и используемое для обмена с B в ходе процедуры подлинности.

Если сертификат C_A и подпись A верны, то участник B уверен, что сообщение пришло от A. Часть сообщения E_{K_B}(r_1) может расшифровать только B, поскольку никто другой не знает секретного ключа k_B, соответствующего открытому ключу $К_B$. Участник B расшифровывает значение числа r_1 и, чтобы подтвердить свою подлинность, посылает участнику A сообщение

(4) $B \rightarrow A$: E_{K_A} (r_1).

Участник A восстанавливает значение r_1, расшифровывая это сообщение с использованием своего секретного ключа k_A. Если это ожидаемое значение r_1, то A получает подтверждение, что вызываемый участник действительно B.

Протокол, основанный на симметричном шифровании, функционирует быстрее, чем протокол, основанный на криптосистемах с открытыми ключами. Однако способность систем с открытыми ключами генерировать цифровые подписи, обеспечивающие различные функции защиты, компенсирует избыточность требуемых вычислений.

9.1.8. Прямой обмен ключами между пользователями

При использовании для информационного обмена криптосистемы с симметричным секретным ключом два пользователя, желающие обменяться криптографически защищенной информацией, должны обладать общим

секретным ключом. Пользователи должны обменяться общим ключом по каналу связи безопасным образом. Если пользователи меняют ключ достаточно часто, то доставка ключа превращается в серьезную проблему.

Для решения этой проблемы можно применить два способа:

1) использование криптосистемы с открытым ключом для шифрования и передачи секретного ключа симметричной криптосистемы;
2) использование системы открытого распределения ключей Диффи–Хеллмана.

9.2. Алгоритм открытого распределения ключей Диффи–Хеллмана.

Алгоритм Диффи–Хеллмана был первым алгоритмом с открытыми ключами (предложен в 1976 г.). Его безопасность обусловлена трудностью вычисления дискретных логарифмов в конечном поле, в отличие от легкости дискретного возведения в степень в том же конечном поле.

Предположим, что два пользователя A и B хотят организовать защищенный коммуникационный канал.

1. Обе стороны заранее уславливаются о модуле N (N должен быть простым числом) и примитивном элементе $g \in Z_N$,
($1 \le g \le N-1$), который образует все ненулевые элементы множества Z_N, т.е.

$$\{g, g^2, ..., g^{N-1} = 1\} = Z_N - \{0\}.$$

Эти два целых числа N и g могут не храниться в секрете. Как правило, эти значения являются общими для всех пользователей системы.

2. Затем пользователи A и B независимо друг от друга выбирают собственные секретные ключи k_A и k_B (k_A и k_B – случайные большие целые числа, которые хранятся пользователями A и B в секрете).

3. Далее пользователь A вычисляет открытый ключ

$$y_A = g^{k_A} \pmod N,$$

а пользователь B – открытый ключ

$$y_B = g^{k_B} \pmod N.$$

4. Затем стороны A и B обмениваются вычисленными значениями открытых ключей y_A и y_B по незащищенному каналу. (Мы считаем, что все данные, передаваемые по незащищенному каналу связи, могут быть перехвачены злоумышленником.)

5. Далее пользователи A и B вычисляют общий секретный ключ, используя следующие сравнения:

пользователь A: $K = (y_B)^{k_A} = (g^{k_B})^{k_A} \pmod N$;

пользователь B: $K' = (y_A)^{k_B} = (g^{k_A})^{k_B} \pmod N$.

При этом $K = K'$, так как $(g^{k_B})^{k_A} = (g^{k_A})^{k_B} \pmod N$.

Схема реализации алгоритма Диффи–Хеллмана показана на рис. 4.5.

Ключ K может использоваться в качестве общего секретного ключа (ключа шифрования ключей) в симметричной криптосистеме.

Кроме того, обе стороны A и B могут шифровать сообщения, используя следующее преобразование шифрования (типа RSA): $C = E_K(M) = M^K \pmod N$.

Для выполнения расшифрования получатель сначала находит ключ расшифрования K* с помощью сравнения

$$K * K^* \equiv 1 \pmod{N-1},$$

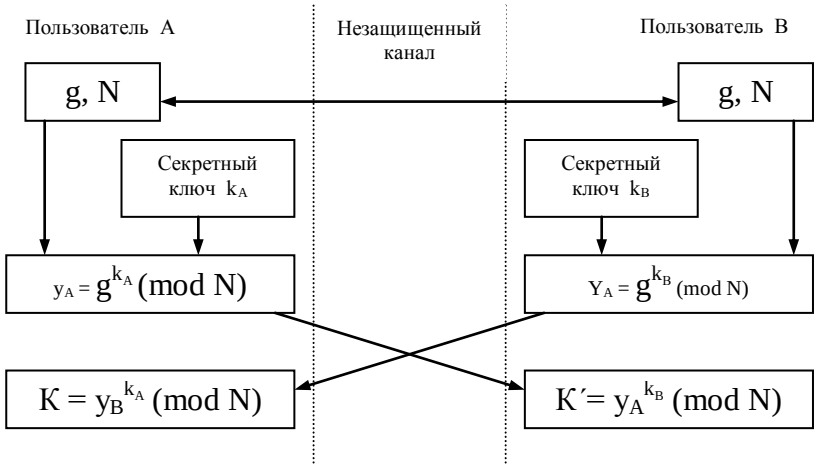

Рисунок 4.5- Схема реализации алгоритма Диффи–Хеллмана

а затем восстанавливает сообщение

$$M = D_K(C) = C^{K^*} \pmod N.$$

Злоумышленник, перехватив значения N, g, y_A и y_B, тоже хотел бы определить значение ключа K. Очевидный путь для решения этой задачи состоит в вычислении такого значения k_A по N, g, y_A, что $g^{k_A} \bmod N = y_A$ (поскольку в этом случае, вычислив k_A, можно найти $K = (y_B)^{k_A} \bmod N$). Однако нахождение k_A по N, g и y_A – задача нахождения дискретного логарифма в конечном поле, которая считается неразрешимой.

Выбор значений N и g может иметь существенное влияние на безопасность этой системы. Модуль N должен быть большим и простым числом. Число $(N-1)/2$ также должно быть простым числом. Число g желательно выбирать таким, чтобы оно было примитивным элементом множества Z_N. (В принципе достаточно, чтобы число g

генерировало большую подгруппу мультипликативной группы по mod N.)

Алгоритм открытого распределения ключей Диффи–Хеллмана позволяет обойтись без защищенного канала для передачи ключей. Однако, работая с этим алгоритмом, необходимо иметь гарантию того, что пользователь A получил открытый ключ именно от пользователя B, и наоборот. Эта проблема решается с помощью электронной подписи,которой подписываются сообщения об открытом ключе.

Метод Диффи–Хеллмана дает возможность шифровать данные при каждом сеансе связи на новых ключах. Это позволяет не хранить секреты на дискетах или других носителях. Не следует забывать, что любое хранение секретов повышает вероятность попадания их в руки конкурентов или противника.

Преимущество метода Диффи–Хеллмана по сравнению с методом RSA заключается в том, что формирование общего секретного ключа происходит в сотни раз быстрее. В системе RSA генерация новых секретных и открытых ключей основана на генерации новых простых чисел, что занимает много времени.

Лабораторная работа №5

ИЗУЧЕНИЕ АЛГОРИТМА ИДЕНТИФИКАЦИИ ГИЛЛОУ-КУИСКУОТЕРА

1. Цель работы

Изучить основные принципы и особенности алгоритма идентификации с нулевой передачей знаний.

2. Рекомендуемые источники

1. Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях /Под ред. В.Ф. Шаньгина.- 2-е изд.. перераб. и доп..-М.: Радио и связь, 2001, с. 143-160.
2. Соколов А.В., Шаньгин В.Ф. Защита информации в распределенных корпоративных сетях и системах.- М.: ДМК Пресс, 2002, с. 239-268.
3. Б. Шнайер. Прикладная криптография.-М.: Триумф, 2002, с. 563-572.

3. Подготовка к работе

1. Ознакомиться с алгоритмом идентификации по одному из рекомендованных источников/1-3/.
2. Ознакомиться с содержанием данной методической разработки.
3. Подготовить бланк отчета.

4. Контрольные вопросы

1. Сущность процедуры и основные алгоритмы идентификации.
2. Сущность процедуры и основные алгоритмы аутентификации.
3. Типовые схемы идентификации и аутентификации пользователя.
4. Особенности биометрической идентификации и аутентификации пользователя.
5. Сущность взаимной проверки подлинности на основе процедуры «рукопожатия».
6. Протоколы идентификации с нулевой передачей знаний.
7. Упрощенная схема идентификации с нулевой передачей знаний.
8. Параллельная схема идентификации с нулевой передачей знаний.
9. Математические соотношения, положенные в основу алгоритма Гиллоу-Куискуотера.
10. Сравнительный анализ процедуры «рукопожатия» и протоколов с нулевой передачей знаний: упрощенная схема, параллельная схема, схема Гиллоу-Куискуотера.

5. Содержание работы

1. Ознакомиться с постановкой задачи идентификации и аутентификации пользователей.
2. Изучить особенности алгоритма идентификации Гиллоу-Куискуотера.
3. Произвести несколько сеансов идентификации мобильных станций, изменяя произвольным образом идентификационную информацию владельца смарт-карты и общесистемные параметры.

6. Содержание отчета

1. Цель работы.
2. Основные математические соотношения алгоритма Гиллоу-Куискуотера.
3. Значения параметров, использованные в сеансах лабораторного задания.

7. Методические указания к выполнению работы

С каждым зарегистрированным в сети субъектом (пользователем или процессом) связана некоторая информация, однозначно идентифицирующая его. Это может быть число или строка символов, именующие данный субъект. Такую информацию называют *идентификатором* субъекта. Субъекты, имеющие зарегистрированные в сети идентификаторы, считаются легальными, остальные субъекты относятся к нелегальным.

Прежде чем получить доступ к ресурсам сети, субъект должен пройти процедуры первичного взаимодействия с сетью, которые включают две стадии - идентификацию и аутентификацию.

Идентификация - это процедура распознавания субъекта по его идентификатору (имени). Эта процедура выполняется в первую очередь, когда субъект делает попытку войти в сеть. Он сообщает по запросу сети свой идентификатор и система проверяет в базе данных его наличие.

Аутентификация - процедура проверки подлинности, позволяющая достоверно убедиться, что субъект является именно тем, кем он себя объявляет.

Идентификация и аутентификация - взаимосвязанные процессы распознавания и проверки подлинности субъектов.

После того как субъект идентифицирован и аутентифицирован, выполняется его *авторизация*, которая устанавливает сферу действия субъекта и доступные ему ресурсы.

Рассмотренные процедуры инициализации являются процедурами защиты и относятся к одному субъекту.

При защите каналов передачи данных выполняется *взаимная аутентификация субъектов*, то есть взаимное

подтверждение подлинности связывающихся между собой по линии связи субъектов.

Широкое распространение интеллектуальных карт (смарт-карт) для разнообразных применений (кредитные карты, мобильные телефоны, карты социального страхования, карты доступа в охраняемое помещение и т.п.) потребовало обеспечения безопасной идентификации таких карт и их владельцев. Главная проблема заключается в том, чтобы при предъявлении смарт-карты, оперативно обнаружить обман и отказать обманщику в допуске, ответе или обслуживании.

Секретный ключ владельца карты становится неотъемлемым признаком его личности. Однако, передавать этот ключ для удостоверения своей личности по открытому каналу недопустимо, так как он может быть перехвачен злоумышленником и использован в дальнейшем.

В связи с этим были разработаны различные протоколы идентификации, в том числе и протоколы с нулевой передачей знаний. Т.е. владелец смарт-карты с помощью определенной процедуры сообщает, что он обладает секретным ключом, избегая при этом необходимости передавать его по каналам в явном виде.

Одним из таких алгоритмов является алгоритм Гиллоу-Куискуотера.

Пусть сторона А - интеллектуальная карточка, которая должна доказать свою подлинность проверяющей стороне В. Идентификационная информация стороны А представляет собой битовую строку W, которая включает имя владельца карточки, срок действия, номер банковского счета и др. Иногда идентификационные данные могут занимать достаточно длинную строку, и тогда их хэшируют к значению W.

В результате длинное сообщение произвольной длины заменяется коротким сообщением фиксированной длины (дайджестом), которое сложным образом зависит от исходного сообщения.

Строка W является аналогом открытого ключа. Другой открытой информацией, которую используют все карты, участвующие в данном приложении, являются модуль n и показатель степени V. Модуль n является произведением двух секретных простых чисел.

Секретным ключом стороны А является величина G, выбираемая таким образом, чтобы выполнялось соотношение

$$W * G^V \equiv 1 \pmod{n}.$$

Сторона А отправляет стороне В свои идентификационные данные W. Далее ей нужно доказать стороне В, что эти идентификационные данные принадлежат именно ей. Чтобы добиться этого, сторона А должна убедить сторону В, что ей известно значение О.

Рассмотрим протокол доказательства подлинности А без передачи стороне В значения О:

1. Сторона А выбирает случайное целое x, такое, что $1 < x \le n - 1$. Она вычисляет

$$T = x^V \pmod{n}$$

и отправляет это значение стороне В.

2. Сторона В выбирает случайное целое d, такое, что $1 < d \le n - 1$, и отправляет это значение d стороне А.

3. Сторона А вычисляет

$$D = x * G^d \pmod{n}$$

и отправляет это значение стороне В.

4. Сторона В вычисляет значение

$$T' = D^V W^d \pmod{n}.$$

Если $T \equiv T' \pmod{n}$, то проверка подлинности успешно завершена.

Математические выкладки, использованные в этом протоколе, не очень сложны:

$$T' = D^V W^d = (xG^d)^V W^d = x^V G^{dV} W^d = \\ = x^V (WG^V)^d = x^V \equiv T (\mathrm{mod} n),$$

поскольку G вычислялось таким образом, чтобы выполнялось соотношение

$$WG^V \equiv 1 \pmod{n}.$$

8. Лабораторное задание

1. Вызов рабочего окна производится из Главного меню по пункту *Идентификация.*

2. Процесс идентификации представлен на схеме. Имитируется установление подлинности мобильной станции со стороны базовой станции.

3. Для удобства проведения работы размерности всех параметров взяты значительно меньшими реально допустимых значений с точки зрения защитных свойств. Например, на практике параметр **n** имеет длину 512 бит ($2^{512} = 4\ 10^{153}$).

4. В работе значения идентификационной информации W выбираются произвольно в указанных пределах. Произвольно выбирается и показатель степени V. Для определения величины модуля **n** необходимо выбрать два достаточно больших простых числа **p** и **q** и найти **n= pq**. Выбор простых чисел произвести с помощью опции «ПРОСТЫЕ ЧИСЛА» из Главного меню.

5. Произвести четыре сеанса идентификации мобильных станций, меняя произвольным образом идентификационную информацию владельца смарт-карты и общесистемные параметры.

9.Общие сведения

9.1. Основные понятия и определения

С каждым объектом компьютерной системы (КС) связана некоторая информация, однозначно идентифицирующая его. Это может быть *число, строка символов, алгоритм,* определяющий данный объект. Эту информацию называют *идентификатором объекта.* Если объект имеет некоторый идентификатор, зарегистрированный в сети, он называется законным (легальным) объектом; остальные объекты относятся к незаконным (нелегальным).

Идентификация объекта – одна из функций подсистемы защиты. Эта функция выполняется в первую очередь, когда объект делает попытку войти в сеть. Если процедура идентификации завершается успешно, данный объект считается законным для данной сети.

Следующий шаг – аутентификация объекта (проверка подлинности объекта). Эта процедура устанавливает, является ли данный объект именно таким, каким он себя объявляет.

После того как объект идентифицирован и подтверждена его подлинность, можно установить сферу его действия и доступные ему ресурсы КС. Такую процедуру называют *предоставлением полномочий (авторизацией)*.

Перечисленные три процедуры инициализации являются процедурами защиты и относятся к одному объекту КС.

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.

Термин "соединение" указывает на логическую связь (потенциально двустороннюю) между двумя объектами сети. Цель данной процедуры – 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:

(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.

Если эти четыре требования реализованы в КС, 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, ни его содержания, а получатель не может отрицать ни факта получения сообщения, ни подлинности его содержания.

9.2. Идентификация и аутентификация пользователя.

Прежде чем получить доступ к ресурсам компьютерной системы, пользователь должен пройти процесс представления компьютерной системе, который включает две стадии:

1) идентификацию - пользователь сообщает системе по ее запросу свое имя (идентификатор);
2) аутентификацию - пользователь подтверждает идентификацию вводя в систему уникальную, не известную другим пользователям информацию о себе (например, пароль).

Для проведения процедур идентификации и аутентификации пользователя необходимы:

- во-первых, наличие соответствующего *субъекта (модуля) аутентификации*;
- во-вторых, наличие *аутентифицирующего объекта*, хранящего уникальную информацию для аутентификации пользователя.

Различают две формы представления объектов, аутентифицирующих пользователя:

- внешний аутентифицирующий объект, не принадлежащий системе;
- внутренний объект, принадлежащий системе, в который переносится информация из внешнего объекта.

Внешние объекты могут быть технически реализованы на различных носителях информации - магнитных дисках, пластиковых картах и т.п. Естественно, что внешняя и внутренняя формы представления аутентифицирующего объекта должны быть семантически тождественны.

9.2.1. Типовые схемы идентификации и аутентификации пользователя

Рассмотрим структуры данных и протоколы идентификации и аутентификации пользователя.

Допустим, что в компьютерной системе зарегистрировано n пользователей. Пусть i-й аутентифицирующий объект i-го пользователя содержит два информационных поля:

ID_i - неизменный идентификатор i-го пользователя, который является аналогом имени и используется для идентификации пользователя;

K_i - аутентифицирующая информация пользователя, которая может изменяться и служит для аутентификации (например, пароль $P_i = K_i$).

Описанная структура соответствует практически любому ключевому носителю информации, используемому для опознания пользователя. Например, для носителей типа пластиковых карт выделяется неизменяемая информация ID_i первичной персонализации пользователя и объект в файловой структуре карты, содержащий K_i.

Совокупную информацию в ключевом носителе можно назвать первичной аутентифицирующей информацией i-го пользователя. Очевидно, что внутренний аутентифицирующий объект не должен существовать в системе длительное время (больше времени работы конкретного пользователя). Для длительного хранения следует использовать данные в защищенной форме.

Рассмотрим две типовые схемы идентификации и аутентификации .

Схема 1. В компьютерной системе выделяется объект-эталон для идентификации и аутентификации пользователей. Структура объекта-эталона для схемы 1 показана в таблице 5.1.

Таблица 5.1-Структура объекта-эталона для схемы 1

Номер пользователя	Информация для идентификации	Информация для аутентификации
1	ID_1	E_1
2	ID_2	E_2
...	...	...
N	ID_n	E_n

Здесь $E_i = F(ID_i, K_i)$,
где F - функция, которая обладает свойством "невосстановимости" значения K_i по E_i и ID_i. "Невосстановимость" K_i оценивается некоторой пороговой трудоемкостью T_o решения задачи восстановления аутентифицирующей информации K_i по E_i и ID_i. Кроме того, для пары K_i и K_j возможно совпадение соответствующих значений E. В связи с этим вероятность ложной аутентификации пользователя не должна быть больше некоторого порогового значения P_o. На практике задают $T_o = 10^{20} ... 10^{30}$, $P_o = 10^{-7} ... 10^{-9}$.

Протокол идентификации и аутентификации (для схемы 1).

1. Пользователь предъявляет свой идентификатор ID.
2. Если ID не совпадает ни с одним ID_i , зарегистрированным в компьютерной системе, то идентификация отвергается - пользователь не допускается к работе, иначе (существует $ID_i = ID$) устанавливается, что пользователь, назвавшийся пользователем i, прошел идентификацию.
3. Субъект аутентификации запрашивает у пользователя его аутентификатор K.
4. Субъект аутентификации вычисляет значение

$$Y = F(ID_i, K).$$

5. Субъект аутентификации производит сравнение значений Y и E_i. При совпадении этих значений устанавливается, что данный пользователь успешно аутентифицирован в системе. Информация об этом пользователе передается в программные модули, использующие ключи пользователей (т.е. в систему шифрования, разграничения доступа и т. д.). В противном случае аутентификация отвергается - пользователь не допускается к работе.

Данная схема идентификации и аутентификации пользователя может быть модифицирована. Модифицированная схема 2 обладает лучшими характеристиками по сравнению со схемой 1.

Схема 2. В компьютерной системе выделяется модифицированный объект-эталон, структура которого показана в таблице 5.2.

Таблица 5.2-Структура объекта-эталона для схемы 2

Номер пользователя	Информация для идентификации	Информация для аутентификации
1	ID_1, S_1	E_1
2	ID_2, S_2.	E_2
...	...	...
N	ID_n, S_n	E_n

В отличие от схемы 1, в схеме 2 значение
$E_i = F(S_i, K_i)$,
где S_i - случайный вектор, задаваемый при создании идентификатора пользователя, т.е. при создании строки, необходимой для идентификации и аутентификации пользователя;
F - функция, которая обладает свойством "невосстановимости" значения K_i по E_i и S_i.

Протокол идентификации и аутентификации (для схемы 2).

1. Пользователь предъявляет свой идентификатор ID.
2. Если ID не совпадает ни с одним ID_i, зарегистрированным в компьютерной системе, то идентификация отвергается - пользователь не допускается к работе, иначе (существует $ID_i = ID$) устанавливается, что пользователь, называвшийся пользователем i, прошел идентификацию.
3. По идентификатору ID_i выделяется вектор S_i.
4. Субъект аутентификации запрашивает у пользователя аутентификатор К.
5. Субъект аутентификации вычисляет значение

$$Y = F(S_i, K).$$

6. Субъект аутентификации производит сравнение значений Y и E_i. При совпадении этих значений устанавливается, что данный пользователь успешно аутентифицирован в системе. В противном случае аутентификация отвергается - пользователь не допускается к работе.

Вторая схема аутентификации применяется в ОС UNIX. В качестве идентификатора ID используется имя пользователя (запрошенное по Loqin), в качестве аутентификатора K_i - пароль пользователя (запрошенный по Password), функция F представляет собой алгоритм шифрования DES. Эталоны для идентификации и аутентификации содержатся в файле Etc/passwd.

Следует отметить, что необходимым требованием устойчивости схем аутентификации к восстановлению информации K_i является случайный равновероятный выбор K_i из множества возможных значений.

Системы парольной аутентификации имеют пониженную стойкость, поскольку в них выбор аутентифицирующей информации происходит из

относительно небольшого множества осмысленных слов. Мощность этого множества определяется энтропией соответствующего языка.

9.2.2. Особенности применения пароля для аутентификации пользователя.

Традиционно каждый законный пользователь компьютерной системы получает идентификатор и/или пароль. В начале сеанса работы пользователь предъявляет свой идентификатор системе, которая затем запрашивает у пользователя пароль.

Простейший метод подтверждения подлинности с использованием пароля основан на сравнении представляемого пользователем пароля P_A с исходным значением P_A' , хранящимся в компьютерном центре (рис. 5.1). Поскольку пароль должен храниться в тайне, он должен шифроваться перед пересылкой по незащищенному каналу. Если значения P_A и P_A' совпадают, то пароль P_A считается подлинным, а пользователь - законным .

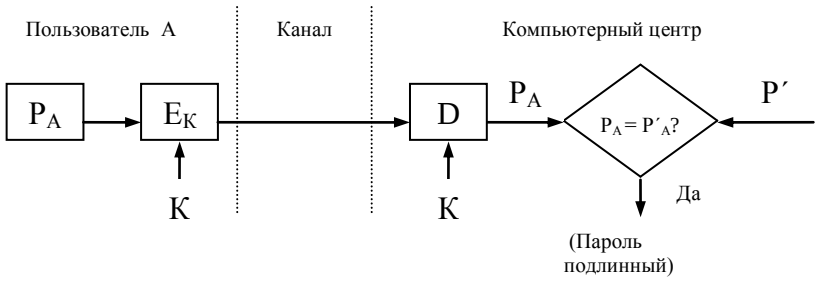

Рисунок 5.1- Схема простой аутентификации с помощью пароля

Если кто-нибудь, не имеющий полномочий для входа в систему, узнает каким-либо образом пароль и идентификационный номер законного пользователя, он получает доступ в систему.

Иногда 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:

$$\alpha(P) = E_{P \oplus K}(ID),$$

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 (рис. 5.2). Подтверждение подлинности состоит из сравнения двух отображений пароля $\alpha(P_A)$ и $\alpha'(P_A)$ и признания пароля P_A, если эти отображения равны. Конечно, любой, кто получит доступ к идентификационной таблице, может незаконно изменить ее содержимое, не опасаясь, что эти действия будут обнаружены.

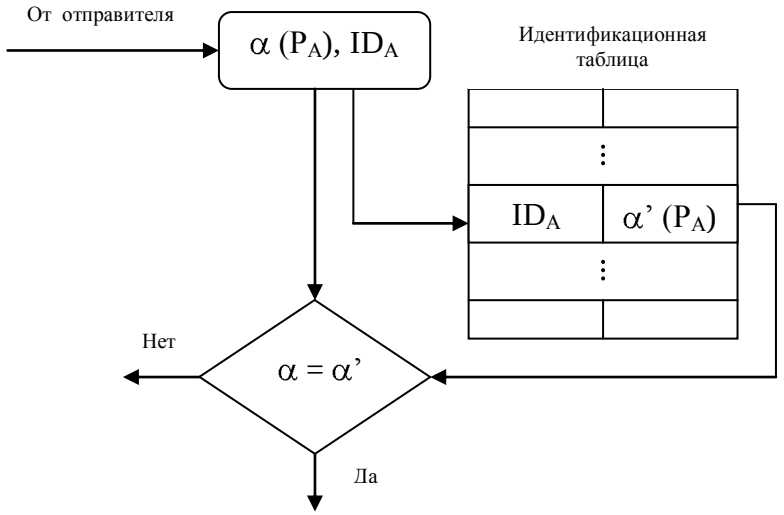

Рисунок 5.2- Аутентификация с помощью пароля с использованием идентификационной таблицы.

9.2.3. Биометрическая идентификация и аутентификация пользователя.

Процедуры идентификации и аутентификации пользователя могут базироваться не только на секретной информации которой обладает пользователь (пароль, секретный ключ, персональный идентификатор и т.п.).

В последнее время все большее распространение получает биометрическая идентификация и аутентификация пользователя, позволяющая уверенно идентифицировать потенциального пользователя путем измерения физиологических параметров и характеристик человека, особенностей его поведения.

Отметим основные достоинства биометрических методов идентификации и аутентификации пользователя по сравнению с традиционными:

- 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}, такие системы являются наиболее надежными среди всех биометрических систем. Такие средства идентификации применяются там, где требуется высокий уровень безопасности (например, в США в зонах военных и оборонных объектов).

Системы идентификации по отпечаткам пальцев являются самыми распространенными. Одной из основных причин широкого распространения таких систем является наличие 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 отличаются у различных 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 получило 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 .

9.3. 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 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 В, не являются ложными, он включает в посылаемое для В сообщение непредсказуемый элемент – запрос X (например, некоторое случайное число). При ответе пользователь В должен выполнить над этим элементом некоторую операцию (например, вычислить некоторую функцию f (X)). Это невозможно осуществить заранее, так как пользователю В неизвестно, какое случайное число X придет в запросе. Получив ответ с результатом действий В, пользователь А может быть уверен, что В – 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.

В обоих случаях для защиты механизма контроля следует применять шифрование, чтобы быть уверенным, что ответ послан не злоумышленником.

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?

Для взаимной проверки подлинности обычно используют *процедуру "рукопожатия"*. Эта процедура базируется на указанных выше механизмах контроля и заключается во взаимной проверке ключей, используемых

сторонами. 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 A и B. (Это допущение не влияет на общность рассмотрения. Такая же процедура используется, когда вступающие в связь стороны не являются пользователями). Пусть применяется симметричная криптосистема. Пользователи A и B разделяют один и тот же секретный ключ K_{AB}. Вся процедура показана на рис. 5.3.

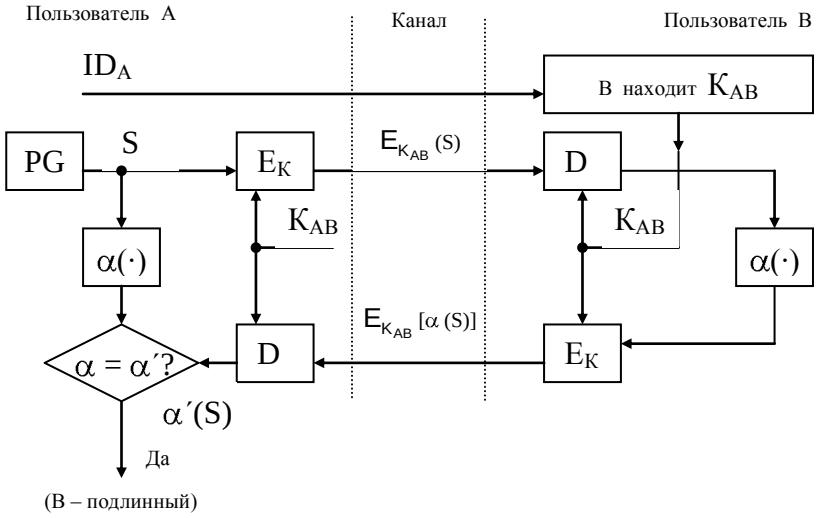

Рисунок 5.3.Схема процедуры рукопожатия (пользователь A проверяет подлинность пользователя B)

- Пусть пользователь A инициирует процедуру рукопожатия, отправляя пользователю B свой идентификатор ID_A в открытой форме.
- Пользователь B, получив идентификатор ID_A, находит в базе данных секретный *ключ* K_{AB} и вводит его в свою криптосистему.
- Тем временем *пользователь A* генерирует случайную *последовательность S* с помощью псевдослучайного генератора PG и отправляет ее *пользователю B* в виде криптограммы

$$E_{K_{AB}}(S).$$

- Пользователь B расшифровывает эту криптограмму и раскрывает исходный вид последовательности S.
- Затем оба пользователя A и B преобразуют последователь-ность S, используя открытую одностороннюю функцию $\alpha(\cdot)$.
- Пользователь B шифрует сообщение $\alpha(S)$ и отправляет эту криптограмму *пользователю A*.
- Наконец, пользователь A расшифровывает эту криптограмму и сравнивает полученное сообщение $\alpha'(S)$ с исходным $\alpha(S)$. Если эти сообщения равны, пользователь A признает подлинность пользователя B.

Очевидно, пользователь B проверяет подлинность пользователя A 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 рис. 5.4 . Передаваемая криптограмма имеет вид

$$E_K (ID_A, M),$$

где ID_A – идентификатор отправителя A; M – сообщение.

Получатель B, принявший эту криптограмму, расшифровывает ее и раскрывает пару (ID_A, M). Если принятый идентификатор ID_A совпадает с хранимым значением ID_A', получатель B признает эту криптограмму.

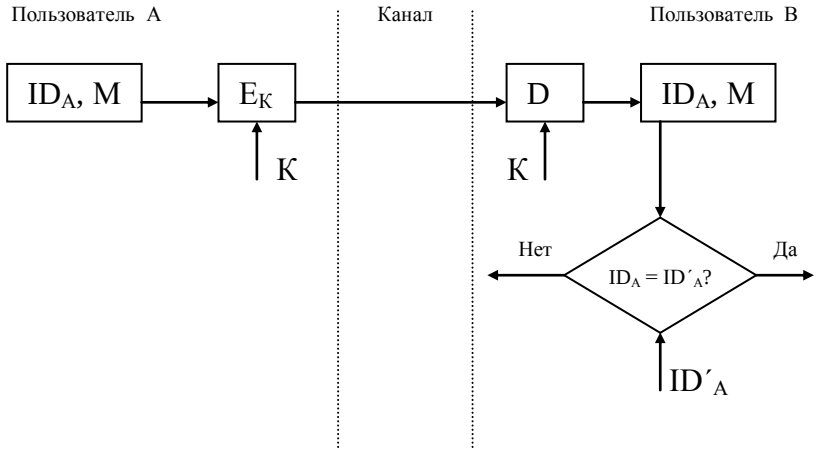

Рисунок 5.4- Схема непрерывной проверки подлинности отправителя

Другой вариант непрерывной проверки подлинности использует вместо идентификатора отправителя его секретный пароль. Заранее подготовленные пароли известны обеим сторонам. Пусть P_A и P_B – пароли пользователей A и B соответственно. Тогда пользователь A 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 A и B уже имеют общий *секретный сеансовый ключ*. Реальные процедуры предназначены для распределения ключей между подлинными партнерами и включает как этап распределения ключей, так и этап собственно подтверждения подлинности партнеров по информационному обмену.

9.4. Протоколы идентификации с нулевой передачей знаний

Широкое распространение интеллектуальных карт (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.

9.4.1. Упрощенная схема идентификации с нулевой передачей знаний

Схему идентификации с нулевой передачей знаний предложили в 1986 г. У.Фейге, А.Фиат и А.Шамир. Она является наиболее известным доказательством идентичности с нулевой передачей конфиденциальной информации.

Рассмотрим сначала упрощенный вариант схемы идентификации с нулевой передачей знаний для более четкого выявления ее основной концепции. 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 A, доказывающая свою подлинность,
- сторона B, проверяющая представляемое стороной A доказательство.

Для того чтобы сгенерировать открытый и секретный ключи для стороны A, доверенный арбитр (Центр) выбирает некоторое число V, которое является квадратичным вычетом по модулю n. Иначе говоря, выбирается такое число V, что сравнение

$$x^2 \equiv V \pmod{n}$$

имеет решение и существует целое число

$$V^{-1} \bmod n.$$

Выбранное значение V является *открытым ключом* для A. Затем вычисляют наименьшее значение S, для которого

$$S \equiv \text{sqrt}\,(V^{-1}) \pmod{n}.$$

Это значение S является *секретным ключом* для A.

Теперь можно приступить к выполнению протокола идентификации.

1. Сторона A выбирает некоторое случайное число r, $r < n$. Затем она вычисляет

$$x = r^2 \bmod n$$

и отправляет x стороне B.

2. Сторона B посылает A случайный бит b.

3. Если b=0, тогда A отправляет r стороне B. Если b=1, то A отправляет стороне B

$$y = r * S \bmod n.$$

4. Если b = 0, сторона B проверяет, что

$$x = r^2 \bmod n,$$

чтобы убедиться, что A знает sqrt (x). Если b=1, сторона B проверяет, что

$$x = y^2 * V \bmod n,$$

чтобы быть уверенной, что A знает sqrt (V^{-1}).

Эти шаги образуют один цикл протокола, называемый *аккредитацией*. Стороны A и B повторяют этот цикл t раз при разных случайных значениях r и b до тех пор, пока B не убедится, что A знает значение S.

Если сторона A не знает значения S, она может выбрать такое значение r, которое позволит ей обмануть сторону B, если B отправит ей b=0, либо A может выбрать такое r, которое позволит обмануть B, если B отправит ей b=1. Но этого невозможно сделать в обоих случаях. Вероятность того, что A обманет B в одном цикле, составляет 1/2. Вероятность обмануть B в t циклах равна $(1/2)^t$.

Для того чтобы этот протокол работал, сторона A никогда не должна повторно использовать значение r. Если A поступила бы таким образом, а сторона B отправила бы стороне A на шаге 2 другой случайный бит b, то B имела бы оба ответа A. После этого B может вычислить значение S, и для A все закончено.

9.4.2. Параллельная схема идентификации с нулевой передачей знаний

Параллельная схема идентификации позволяет увеличить число аккредитаций, выполняемых за один цикл, и тем самым уменьшить длительность процесса идентификации.

Как и в предыдущем случае, сначала генерируется число n как произведение двух больших чисел. Для того, чтобы сгенерировать открытый и секретный ключи для стороны A, сначала выбирают K различных чисел V_1, V_2, ..., V_K, где каждое V_i является квадратичным вычетом по модулю n. Иначе говоря, выбирают значение V_i таким, что сравнение

$$x^2 \equiv V_i \bmod n$$

имеет решение и существует $V_i^{-1} \bmod n$. Полученная строка $V_1, V_2, ..., V_K$ является *открытым ключом*.

Затем вычисляют такие наименьшие значения S_i, что

$$S_i = \text{sqrt}\,(V_i^{-1}) \bmod n.$$

Эта строка $S_1, S_2, ..., S_K$ является *секретным ключом* стороны A.

Протокол процесса идентификации имеет следующий вид:

1. Сторона A выбирает некоторое случайное число r, r<n. Затем она вычисляет $x=r^2 \bmod n$ и посылает x стороне B.

2. Сторона B отправляет стороне A некоторую случайную двоичную строку из K бит: $b_1, b_2, ..., b_K$.

3. Сторона A вычисляет

$$y = r * (S_1^{b1} * S_2^{b2} * ... * S_K^{bK}) \bmod n.$$

Перемножаются только те значения S_i, для которых $b_i=1$. Например, если $b_1=1$, то сомножитель S_1 входит в произведение, если же $b_1=0$, то S_1 не входит в произведение, и т.д. Вычисленное значение y отправляется стороне B.

4. Сторона B проверяет, что

$$x = y^2 * (V_1^{b1} * V_2^{b2} * ... * V_K^{bK}) \bmod n.$$

Фактически сторона B перемножает только те значения V_i, для которых $b_i=1$. Стороны A и B повторяют этот протокол t раз, пока B не убедится, что A знает $S_1, S_2, ..., S_K$.

Вероятность того, что A может обмануть B, равна $(1/2)^{Kt}$. Авторы рекомендуют в качестве контрольного значения брать вероятность обмана B равной $(1/2)^{20}$ при K=5 и t=4.

Стороны A и B повторяют этот протокол t раз, каждый раз с разным случайным числом r, пока сторона B не будет удовлетворена.

При малых значениях величин, как в данном примере, не достигается настоящей безопасности. Но если n представляет собой число длиной 512 бит и более, сторона B не сможет узнать ничего о секретном ключе стороны A, кроме того факта, что сторона A знает этот ключ.

В этот протокол можно включить идентификационную информацию .

Пусть I – некоторая двоичная строка, представляющая идентификационную информацию о владельце карты (имя, адрес, персональный идентификационный номер, физическое описание) и о карте (дата окончания действия и т.п.). Эту информацию I формируют в Центре выдачи интеллектуальных карт по заявке пользователя A.

Далее используют одностороннюю функцию f (·) для вычисления f (I, j), где j – некоторое двоичное число, сцепляемое со строкой I. Вычисляют значения

$$V_j = f\,(I, j)$$

для небольших значений j, отбирают K разных значений j, для которых V_j являются квадратичными вычетами по модулю n. Затем для отобранных квадратичных вычетов V_j вычисляют наименьшие квадратные корни из V_j^-

^{1}(mod n). Совокупность из K значений V_j образует открытый ключ, а совокупность из K значений S_j – секретный ключ пользователя A.

Лабораторная работа №6

ИЗУЧЕНИЕ АЛГОРИТМА ЭЛЕКТРОННОЙ ЦИФРОВОЙ ПОДПИСИ RSA

1. Цель работы

Изучить основные принципы и процедурные аспекты алгоритма электронной цифровой подписи (ЭЦП) варианта RSA.

2. Рекомендуемые источники

1. Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях /Под ред. В.Ф. Шаньгина.- 2-е изд.. перераб. и доп..-М.: Радио и связь, 2001, с. 161-169.
2. Соколов А.В., Шаньгин В.Ф. Защита информации в распределенных корпоративных сетях и системах.- М.: ДМК Пресс, 2002, с. 222-227.
3. Б. Шнайер. Прикладная криптография.-М.: Триумф, 2002, с. 541-548.

3. Подготовка к работе

1. Ознакомиться с алгоритмами постановки и проверки электронной цифровой подписи по одному из рекомендованных источников /1-3/.
2. Ознакомиться с содержанием данной методической разработки.
3. Подготовить бланк отчета, который должен содержать:
   - цель работы;
   - основные математические соотношения для обоснования алгоритма;
   - заготовку таблицы, в которую будут заноситься результаты выполнения работы.

4. Контрольные вопросы

1.Цель аутентификации электронных документов.
2.Назначение электронной цифровой подписи.
3.Назначение и особенности хэш-функции.
4.Процедуры постановки и проверки ЭЦП.
5.Алгоритм цифровой подписи RSA.
6.Отечественный стандарт хэш-функции.
7.Отечественный стандарт цифровой подписи.
8.Однонаправленные хэш-функции на основе симметричных блочных алгоритмов.
9.Требования, предъявляемые к хэш-функциям.
10. Виды атак на ЭЦП.
11. Недостатки ЭЦП RSA.

5. Содержание работы

1.Ознакомиться с сущностью ЭЦП.
2.Изучить математические соотношения, лежащие в основе постановки и проверки ЭЦП.
3.Изучить основные процедуры алгоритма ЭЦП.
4.Произвести передачу различных текстов с проверкой подлинности сообщений и их авторства.
5.Произвести поиск нескольких коллизий и зафиксировать их последствия.

6. Содержание отчета

1.Цель работы.
2.Структурную схему алгоритма ЭЦП RSA.
3.Основные математические соотношения, предписываемые алгоритмом ЭЦП.
4.Результаты постановки и проверки ЭЦП.

7. Методические указания к выполнению работы

При обмене электронными документами по сети связи существенно снижаются затраты на обработку и хранение документов, убыстряется их поиск. Но при этом возникает проблема аутентификации автора документа и самого документа, т.е. 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 C посылает документ абоненту B от имени абонента A;
- ренегатство – абонент A заявляет, что не посылал сообщения абоненту B, хотя на самом деле послал;
- подмена – абонент B изменяет или формирует новый документ и заявляет, что получил его от абонента A;
- повтор – абонент C повторяет ранее переданный документ, который абонент A посылал абоненту B.

Эти виды злоумышленных действий могут нанести существенный ущерб банковским и коммерческим структурам, государственным предприятиям и организациям, 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 (ЭЦП).

Электронная цифровая подпись используется для аутентификации текстов, передаваемых по телекоммуникационным каналам. 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.

Цифровая подпись представляет собой относительно небольшое количество дополнительной цифровой информации, передаваемой вместе с подписываемым текстом.

Система ЭЦП включает две процедуры: 1) процедуру постановки подписи; 2) процедуру проверки подписи. В процедуре постановки подписи используется секретный ключ отправителя сообщения, в процедуре проверки подписи – открытый ключ отправителя.

При формировании ЭЦП отправитель прежде всего вычисляет хэш-функцию h(M) подписываемого текста M. Вычисленное значение хэш-функции h(M) представляет собой один короткий блок информации m, характеризующий весь текст M в целом. Затем число m шифруется секретным ключом отправителя. Получаемая при этом пара чисел представляет собой ЭЦП для данного текста M.

При проверке ЭЦП получатель сообщения снова вычисляет хэш-функцию m = h(M) принятого по каналу текста M, после чего при помощи открытого ключа отправителя проверяет, соответствует ли полученная подпись вычисленному значению m хэш-функции.

Принципиальным моментом в системе ЭЦП является невозможность подделки ЭЦП пользователя без знания его секретного 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 содержит следующую информацию:

- дату подписи;
- 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.

Первой и наиболее известной во всем мире конкретной системой ЭЦП стала система RSA, математическая схема которой была разработана в 1977 г. в Массачуссетском технологическом институте США.

Сначала необходимо вычислить пару ключей (секретный ключ и открытый ключ). Для этого отправитель (автор) электронных документов вычисляет два больших простых числа P и Q, затем находит их произведение

$$N = P * Q$$

и значение функции

$$\varphi(N) = (P-1)(Q-1).$$

Далее отправитель вычисляет число E из условий:

$$E \le \varphi(N), \quad НОД(E, \varphi(N)) = 1$$

и число D из условий:

$$D < N, \quad E * D \equiv 1 \pmod{\varphi(N)}.$$

Пара чисел (E,N) является открытым ключом. Эту пару чисел автор передает партнерам по переписке для проверки его цифровых подписей. Число D сохраняется автором как секретный ключ для подписания.

Обобщенная схема формирования и проверки цифровой подписи RSA показана на рис.6.1.

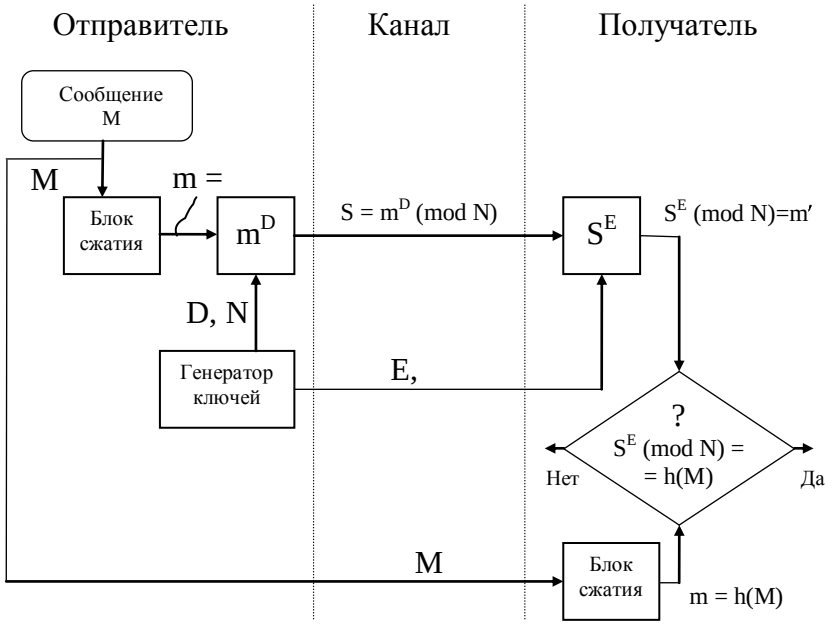

Рисунок 6.1- Обобщенная схема цифровой подписи RSA

Допустим, что отправитель хочет подписать сообщение M перед его отправкой. Сначала сообщение M (блок информации, файл, таблица) сжимают с помощью хэш-функции h(·) в целое число m:

$$m = h(M).$$

Затем вычисляют цифровую подпись S под электронным документом M, используя хэш-значение m и секретный ключ D:

$$S = m^D \pmod N.$$

Пара (M,S) передается партнеру-получателю как электронный документ M, подписанный цифровой подписью S, причем подпись S сформирована обладателем секретного ключа D.

После приема пары (M,S) получатель вычисляет хэш-значение сообщения M двумя разными способами. Прежде всего он восстанавливает хэш-значение m′, применяя криптографическое преобразование подписи S с использованием открытого ключа E:

$$m' = S^E \pmod N.$$

Кроме того, он находит результат хэширования принятого сообщения M с помощью такой же хэш-функции h(·):

$$m = h(M).$$

Если соблюдается равенство вычисленных значений, т.е.

$$S^E \pmod N = h(M),$$

то получатель признает пару (M,S) подлинной. Доказано, что только обладатель секретного ключа D может сформировать цифровую подпись S по документу M, а определить секретное число D по открытому числу E не легче, чем разложить модуль N на множители.

Кроме того, можно строго математически доказать, что результат проверки цифровой подписи S будет положительным только в том случае, если при вычислении S был использован секретный ключ D, соответствующий открытому ключу E. Поэтому открытый ключ E иногда называют "идентификатором" подписавшего.

Недостатки алгоритма цифровой подписи RSA.

1. При вычислении модуля N, ключей E и D для системы цифровой подписи RSA необходимо проверять большое количество дополнительных условий, что сделать практически трудно. Невыполнение любого из этих условий делает возможным фальсификацию цифровой подписи со стороны того, кто обнаружит такое невыполнение. При подписании важных документов нельзя допускать такую возможность даже теоретически.

2. Для обеспечения криптостойкости цифровой подписи RSA по отношению к попыткам фальсификации на уровне, например, национального стандарта США на шифрование информации (алгоритм DES), т.е. 10^{18}, необходимо использовать при вычислениях N, D и E целые числа не менее 2^{512} (или около 10^{154}) каждое, что требует больших вычислительных затрат, превышающих на 20…30% вычислительные затраты других алгоритмов цифровой подписи при сохранении того же уровня криптостойкости.

3. Цифровая подпись RSA уязвима к так называемой мультипликативной атаке. Иначе говоря, алгоритм цифровой подписи RSA позволяет злоумышленнику без знания секретного ключа D сформировать подписи под теми документами, у которых результат хэширования можно вычислить как произведение результатов хэширования уже подписанных документов

Организационное обеспечение ЭЦП включает регистрацию пользователей в некотором доверительном центре, оформление документов между пользователем и доверительным центром об ответственности за переданные друг другу открытые ключи и другие мероприятия.

Одним из методов защиты от атак на ЭЦП является уменьшение до минимума вероятности нахождения **коллизий**. В данном случае коллизия – это нужный злоумышленнику ложный текст, хэш-значение которого совпадает с хэш-значением истинного текста. Если такой текст будет найден, то отправителю будет трудно доказать, что он не подписывал ложный текст. Поэтому, исключительно важное значение имеет правильный выбор алгоритма хэширования.

8. Лабораторное задание

1.Вызов рабочего окна производится из главного меню по пункту EDS (ЭЦП). В окне представлены два поля MO

И MP для сообщений отправителя и получателя, соответственно. Отправляемое сообщение вводится в поле MO в виде последовательности десятичных цифр произвольной длины. В качестве алгоритма хэширования реализован недопустимо примитивный для реальных систем алгоритм: вычисление вычета по модулю N=8014003 построчной суммы всего текста. В качестве параметров ЭЦП RSA выбраны и вычислены следующие значения:

N= PQ= 4001 2003 = 8014003;

φ (N) = (P-1)(Q-1) = 4000 2002 = 8008000;

E = 986311, D= 7927

2.Произвести передачу трех различных текстов с проверкой подлинности сообщений и их авторства. Все данные зафиксировать в отчете.

3.Произвести имитацию подмены текста в сети связи, изменив одну из цифр в полученном сообщении. Прокомментируйте реакцию получателя.

4.С учетом особенностей реализованного в данной учебной программе алгоритма хэширования произведите поиск двух-трех коллизий. Учтите, что если в сообщении M последняя строка не полная, то ее младший разряд сдвигается влево.

9. Общие сведения

9.1. Однонаправленные хэш-функции

Хэш-функция предназначена для сжатия подписываемого документа M до нескольких десятков или сотен бит. Хэш-функция h(·) принимает в качестве аргумента сообщение (документ) M произвольной длины и возвращает хэш-значение h(M)=H фиксированной длины. Обычно хэшированная информация является сжатым двоичным представлением основного сообщения произвольной длины. Следует отметить, что значение хэш-

функции h(M) сложным образом зависит от документа M и не позволяет восстановить сам документ M.

Хэш-функция должна удовлетворять целому ряду условий:

- хэш-функция должна быть чувствительна к всевозможным изменениям в тексте M, таким как вставки, выбросы, перестановки и т.п.;
- хэш-функция должна обладать свойством необратимости, то есть задача подбора документа M', который обладал бы требуемым значением хэш-функции, должна быть вычислительно неразрешима;
- вероятность того, что значения хэш-функций двух различных документов (вне зависимости от их длин) совпадут, должна быть ничтожно мала.

Большинство хэш-функций строится на основе однонаправленной функции f(·), которая образует выходное значение длиной n при задании двух входных значений длиной n. Этими входами являются блок исходного текста M_i и хэш-значение H_{i-1} предыдущего блока текста (рис.6.1):

$H_i = f(M_i, H_{i-1})$.

Хэш-значение, вычисляемое при вводе последнего блока текста, становится хэш-значением всего сообщения M.

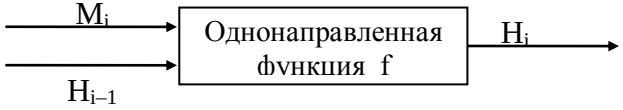

Рисунок 6.1- Построение однонаправленной хэш-функции

В результате однонаправленная хэш-функция всегда формирует выход фиксированной длины n (независимо от длины входного текста).

9.1.1. Однонаправленные хэш-функции на основе симметричных блочных алгоритмов

Однонаправленную хэш-функцию можно построить, используя симметричный блочный алгоритм. Наиболее очевидный подход состоит в том, чтобы шифровать сообщение M посредством блочного алгоритма в режиме CBC или CFB с помощью фиксированного ключа и некоторого вектора инициализации IV. Последний блок шифртекста можно рассматривать в качестве хэш-значения сообщения M. При таком подходе не всегда возможно построить безопасную однонаправленную хэш-функцию, но всегда можно получить код аутентификации сообщения MAC (Message Authentication Code).

Более безопасный вариант хэш-функции можно получить, используя блок сообщения в качестве ключа, предыдущее хэш-значение – в качестве входа, а текущее хэш-значение – в качестве выхода. Реальные хэш-функции проектируются еще более сложными. Длина блока обычно определяется длиной ключа, а длина хэш-значения совпадает с длиной блока.

Поскольку большинство блочных алгоритмов являются 64-битовыми, некоторые схемы хэширования проектируют так, чтобы хэш-значение имело длину, равную двойной длине блока.

Если принять, что получаемая хэш-функция корректна, безопасность схемы хэширования базируется на безопасности лежащего в ее основе блочного алгоритма. Схема хэширования, у которой длина хэш-значения равна длине блока, показана на рис. 6.2. Ее работа описывается выражениями:

$$H_0 = I_H,$$
$$H_i = E_A(B) \oplus C,$$

где I_H – некоторое случайное начальное значение; A, B и C могут принимать значения M_i, H_{i-1}, $(M_i \oplus H_{i-1})$ или быть константами.

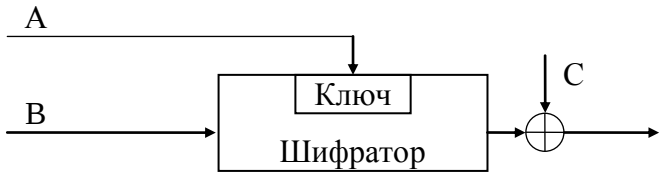

Рисунок 6.2- Обобщенная схема формирования хэш-функции

Сообщение M разбивается на блоки M_i принятой длины, которые обрабатываются поочередно.

Три различные переменные A, B и C могут принимать одно из четырех возможных значений, поэтому в принципе можно получить 64 варианта общей схемы этого типа. Из них 52 варианта являются либо тривиально слабыми, либо небезопасными. Остальные 12 безопасных схем хэширования перечислены в табл. 6.1.

Таблица 6.1- Схемы безопасного хэширования, у которых длина хэш-значения равна длине блока

Номер схемы	Функция хэширования
1	$H_i = E_{H_{i-1}}(M_i) \oplus M_i$
2	$H_i = E_{H_{i-1}}(M_i \oplus H_{i-1}) \oplus M_i \oplus H_{i-1}$
3	$H_i = E_{H_{i-1}}(M_i) \oplus H_{i-1} \oplus M_i$
4	$H_i = E_{H_{i-1}}(M_i \oplus H_{i-1}) \oplus M_i$
5	$H_i = E_{M_i}(H_{i-1}) \oplus H_{i-1}$
6	$H_i = E_{M_i}(M_i \oplus H_{i-1})\ M_i \oplus H_{i-1}$
7	$H_i = E_{M_i}(H_{i-1})\ M_i \oplus H_{i-1}$
8	$H_i = E_{M_i}(M_i \oplus H_{i-1}) \oplus H_{i-1}$
9	$H_i = E_{M_i \oplus H_{i-1}}(M_i) \oplus M_i$
10	$H_i = E_{M_i \oplus H_{i-1}}(H_{i-1}) \oplus H_{i-1}$
11	$H_i = E_{M_i \oplus H_{i-1}}(M_i) \oplus H_{i-1}$
12	$H_i = E_{M_i \oplus H_{i-1}}(H_{i-1}) \oplus M_i$

Первые четыре схемы хэширования, являющиеся безопасными при всех атаках, приведены на рис.6.3.

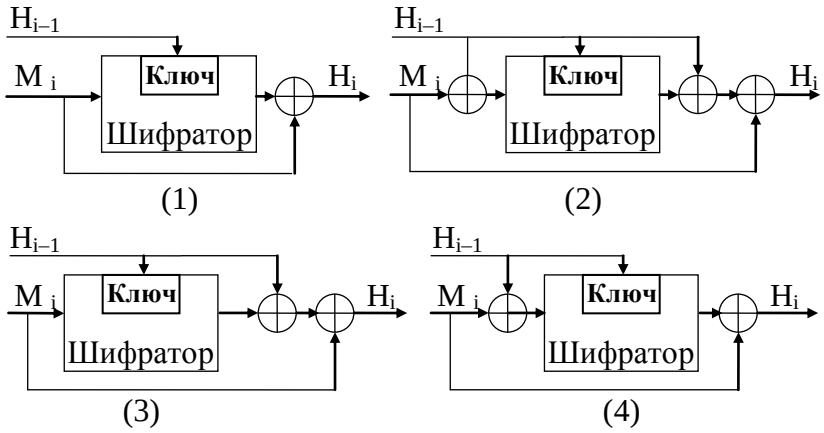

Рисунок 6.3- Четыре схемы безопасного хэширования

9.1.2. Отечественный стандарт хэш-функции

Российский стандарт ГОСТ Р 34.11-94 определяет алгоритм и процедуру вычисления хэш-функции для любых последовательностей двоичных символов, применяемых в криптографических методах обработки и защиты информации. Этот стандарт базируется на блочном алгоритме шифрования ГОСТ 28147-89, хотя в принципе можно было бы использовать и другой блочный алгоритм шифрования с 64-битовым блоком и 256-битовым ключом.

Данная хэш-функция формирует 256-битовое хэш-значение.

Функция сжатия $H_i = f(M_i, H_{i-1})$ (оба операнда M_i и H_{i-1} являются 256-битовыми величинами) определяется следующим образом:

1. Генерируются 4 ключа шифрования K_j, $j = 1…4$, путем линейного смешивания M_i, H_{i-1} и некоторых констант C_j.

2. Каждый ключ K_j, используют для шифрования 64-битовых подслов h_i слова H_{i-1} в режиме простой замены: $S_j = E_{K_j}(h_j)$. Результирующая последовательность S_4, S_3, S_2, S_1 длиной 256 бит запоминается во временной переменной S.

3. Значение H_i является сложной, хотя и линейной функцией смешивания S, M_i и H_{i-1}.

При вычислении окончательного хэш-значения сообщения M учитываются значения трех связанных между собой переменных:

H_n – хэш-значение последнего блока сообщения;

Z – значение контрольной суммы, получаемой при сложении по модулю 2 всех блоков сообщения;

L – длина сообщения.

Эти три переменные и дополненный последний блок M′ сообщения объединяются в окончательное хэш-значение следующим образом:

$$H = f(Z \oplus M', f(L, f(M', H_n))).$$

Данная хэш-функция определена стандартом ГОСТ Р 34.11-94 для использования совместно с российским стандартом электронной цифровой подписи.

9.2. Алгоритмы электронной цифровой подписи

Технология применения системы ЭЦП предполагает наличие сети абонентов, посылающих друг другу подписанные электронные документы. Для каждого абонента генерируется пара ключей: секретный и открытый. Секретный ключ хранится абонентом в тайне и используется им для формирования ЭЦП. Открытый ключ известен всем другим пользователям и предназначен для проверки ЭЦП получателем подписанного электронного документа. Иначе говоря, открытый ключ является необходимым инструментом, позволяющим проверить подлинность электронного документа и автора подписи. Открытый ключ не позволяет вычислить секретный ключ.

Для генерации пары ключей (секретного и открытого) в алгоритмах ЭЦП, как и в асимметричных системах шифрования, используются разные математические схемы, основанные на применении однонаправленных функций. Эти схемы разделяются на две группы. В основе такого разделения лежат известные сложные вычислительные задачи:

- задача факторизации (разложения на множители) больших целых чисел;
- задача дискретного логарифмирования.

9.2.1. Алгоритм цифровой подписи Эль Гамаля (EGSA)

Более надежный и удобный, чем RSA, для реализации на персональных компьютерах алгоритм цифровой подписи был разработан в 1984 г. американцем арабского происхождения Тахером Эль Гамалем. В 1991 г. НИСТ США обосновал перед комиссией Конгресса США выбор алгоритма цифровой подписи Эль Гамаля в качестве основы для национального стандарта.

Название EGSA происходит от слов El Gamal Signature Algorithm (алгоритм цифровой подписи Эль Гамаля). Идея EGSA основана на том, что для обоснования практической невозможности фальсификации цифровой подписи может быть использована более сложная вычислительная задача, чем разложение на множители большого целого числа,– задача дискретного логарифмирования. Кроме того, Эль Гамалю удалось избежать явной слабости алгоритма

цифровой подписи RSA, связанной с возможностью подделки цифровой подписи под некоторыми сообщениями без определения секретного ключа.

Рассмотрим подробнее алгоритм цифровой подписи Эль Гамаля. Для того чтобы генерировать пару ключей (открытый ключ – секретный ключ), сначала выбирают некоторое большое простое целое число P и большое целое число G, причем $G < P$. Отправитель и получатель подписанного документа используют при вычислениях одинаковые большие целые числа P ($\sim 10^{308}$ или $\sim 2^{1024}$) и G ($\sim 10^{154}$ или $\sim 2^{512}$), которые не являются секретными.

Отправитель выбирает случайное целое число X, $1 < X \le (P-1)$, и вычисляет

$$Y = G^X \bmod P.$$

Число Y является открытым ключом, используемым для проверки подписи отправителя. Число Y открыто передается всем потенциальным получателям документов.

Число X является секретным ключом отправителя для подписывания документов и должно храниться в секрете.

Для того чтобы подписать сообщение M, сначала отправитель хэширует его с помощью хэш-функции $h(\cdot)$ в целое число m:

$$m = h(M), \quad 1 < m < (P-1),$$

и генерирует случайное целое число K, $1 < K < (P-1)$, такое, что K и $(P-1)$ являются взаимно простыми. Затем отправитель вычисляет целое число a:

$$a = G^K \bmod P$$

и, применяя расширенный алгоритм Евклида, вычисляет с помощью секретного ключа X целое число b из уравнения

$$m = X * a + K * b \pmod{(P-1)}.$$

Пара чисел (a,b) образует цифровую подпись S:

$$S = (a,b),$$

проставляемую под документом M.

Тройка чисел (M,a,b) передается получателю, в то время как пара чисел (X,K) держится в секрете.

После приема подписанного сообщения (M,a,b) получатель должен проверить, соответствует ли подпись S = (a,b) сообщению M. Для этого получатель сначала вычисляет по принятому сообщению M число

$$m = h(M),$$

т.е. хэширует принятое сообщение M.

Затем получатель вычисляет значение

$$A = Y^a a^b \pmod P$$

и признает сообщение M подлинным, если, и только если

$$A = G^m \pmod P.$$

Иначе говоря, получатель проверяет справедливость соотношения

$$Y^a a^b \pmod P = G^m \pmod P.$$

Можно строго математически доказать, что последнее равенство будет выполняться тогда, и только тогда, когда подпись S=(a,b) под документом M получена с помощью именно того секретного ключа X, из которого был получен открытый ключ Y. Таким образом, можно надежно удостовериться, что отправителем сообщения M был обладатель именно данного секретного ключа X, не раскрывая при этом сам ключ, и что отправитель подписал именно этот конкретный документ M.

Следует отметить, что выполнение каждой подписи по методу Эль Гамаля требует нового значения K, причем это значение должно выбираться случайным образом. Если нарушитель раскроет когда-либо значение K, повторно используемое отправителем, то он сможет раскрыть секретный ключ X отправителя.

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 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 (т.е. всего два достаточно просто проверяемых условия).

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

Однако алгоритм цифровой подписи Эль Гамаля имеет и некоторые недостатки по сравнению со схемой подписи RSA. В частности, длина цифровой подписи получается в 1,5 раза больше, что, в свою очередь, увеличивает время ее вычисления.

9.2.2. Алгоритм цифровой подписи DSA

Алгоритм цифровой подписи DSA (Digital Signature Algorithm) предложен в 1991 г. 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, L бит каждое ($512 \le L \le 1024$); q – простое число длиной 160 бит (делитель числа (P –1)). Числа G, P, q являются открытыми и могут быть общими для всех пользователей сети.

Отправитель выбирает случайное целое число X, $1< X< q$. Число X является секретным 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. Число Y передается всем получателям документов.

Этот алгоритм также предусматривает использование односторонней функции хэширования h(·). В стандарте DSS определен алгоритм безопасного хэширования SHA (Secure Hash Algorithm).

Для того чтобы подписать документ M, отправитель хэширует его в целое хэш-значение m:

$$m = h(M), \quad 1< m < q,$$

затем генерирует случайное целое число K, $1< K< q$, и вычисляет число r:

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

Затем отправитель вычисляет с помощью секретного ключа X целое число s:

$$s = \frac{m + r * X}{K} \bmod q.$$

Пара чисел r и 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

$$w = \frac{1}{s} \bmod q,$$

хэш-значение

$$m = h(M)$$

и числа

$$u_1 = (m * w) \bmod q,$$
$$u_2 = (r * w) \bmod q.$$

Далее получатель с помощью открытого ключа Y вычисляет значение

$$v = ((G^{u_1} * Y^{u_2}) \bmod P) \bmod q$$

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 секретного ключа X, из которого был получен открытый ключ Y. Таким образом, можно надежно удостовериться, что отправитель сообщения владеет именно данным секретным 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, т.е. 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:

$$s = \frac{m + rX}{K} \pmod q, \quad w = \frac{1}{s} \pmod q,$$

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 хэш-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.

9.2.3. Отечественный стандарт цифровой подписи

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

$y = a^x \bmod p$.

Кроме того, этот алгоритм использует однонаправленную хэш-функцию H(x). Стандарт ГОСТ Р 34.11-94 определяет хэш-функцию, основанную на использовании стандартного симметричного алгоритма ГОСТ 28147-89.

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

Чтобы подписать некоторое сообщение m, а затем проверить подпись, выполняются следующие шаги.

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

2. Пользователь A вычисляет значения

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

$$s = (x * r + k\,(H(m))) \bmod q.$$

Если $H(m) \bmod q=0$, то значение $H(m) \bmod q$ принимают равным единице. Если r=0, то выбирают другое значение k и начина-ют снова.

Цифровая подпись представляет собой два числа:

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

Пользователь A отправляет эти числа пользователю B.

3. Пользователь B проверяет полученную подпись, вычисляя

$$v = H(m)^{q-2} \bmod q,$$

$$z_1 = (s * v) \bmod q,$$

$$z_2 = ((q - r) * v) \bmod q,$$

$$u = ((a^{z_1} * y^{z_2}) \bmod p) \bmod q.$$

Если u = r, то подпись считается верной.

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

$$s = (k^{-1}\,(x * r + (H(m)))) \bmod q\,,$$

что приводит к другому уравнению верификации.

Следует также отметить, что в отечественном стандарте ЭЦП параметр q имеет длину 256 бит. 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 г.

Приложение 1 Последовательность простых чисел

```
RPCh
5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,101,103,107,109,113,127,131,137,139,
149,151,157,163,167,173,179,181,191,193,197,199,211,223,227,229,233,239,241,251,257,263,269,271,277,
281,283,293,307,311,313,317,331,337,347,349,353,359,367,373,379,383,389,397,401,409,419,421,431,433,
439,443,449,457,461,463,467,479,487,491,499,503,509,521,523,541,547,557,563,569,571,577,587,593,599,
601,607,613,617,619,631,641,643,647,653,659,661,673,677,683,691,701,709,719,727,733,739,743,751,757,
761,769,773,787,797,809,811,821,823,827,829,839,853,857,859,863,877,881,883,887,907,911,919,929,937,
941,947,953,967,971,977,983,991,997,1009,1013,1019,1021,1031,1033,1039,1049,1051,1061,1063,1069,1087,
1091,1093,1097,1103,1109,1117,1123,1129,1151,1153,1163,1171,1181,1187,1193,1201,1213,1217,1223,1229,
1231,1237,1249,1259,1277,1279,1283,1289,1291,1297,1301,1303,1307,1319,1321,1327,1361,1367,1373,1381,
1399,1409,1423,1427,1429,1433,1439,1447,1451,1453,1459,1471,1481,1483,1487,1489,1493,1499,1511,1523,
1531,1543,1549,1553,1559,1567,1571,1579,1583,1597,1601,1607,1609,1613,1619,1621,1627,1637,1657,1663,
1667,1669,1693,1697,1699,1709,1721,1723,1733,1741,1747,1753,1759,1777,1783,1787,1789,1801,1811,1823,
1831,1847,1861,1867,1871,1873,1877,1879,1889,1901,1907,1913,1931,1933,1949,1951,1973,1979,1987,1993,
1997,1999,2003,2011,2017,2027,2029,2039,2053,2063,2069,2081,2083,2087,2089,2099,2111,2113,2129,2131,
2137,2141,2143,2153,2161,2179,2203,2207,2213,2221,2237,2239,2243,2251,2267,2269,2273,2281,2287,2293,
2297,2309,2311,2333,2339,2341,2347,2351,2357,2371,2377,2381,2383,2389,2393,2399,2411,2417,2423,2437,
2441,2447,2459,2467,2473,2477,2503,2521,2531,2539,2543,2549,2551,2557,2579,2591,2593,2609,2617,2621,
2633,2647,2657,2659,2663,2671,2677,2683,2687,2689,2693,2699,2707,2711,2713,2719,2729,2731,2741,2749,
2753,2767,2777,2789,2791,2797,2801,2803,2819,2833,2837,2843,2851,2857,2861,2879,2887,2897,2903,2909,
2917,2927,2939,2953,2957,2963,2969,2971,2999,3001,3011,3019,3023,3037,3041,3049,3061,3067,3079,3083,
3089,3109,3119,3121,3137,3163,3167,3169,3181,3187,3191,3203,3209,3217,3221,3229,3251,3253,3257,3259,
3271,3299,3301,3307,3313,3319,3323,3329,3331,3343,3347,3359,3361,3371,3373,3389,3391,3407,3413,3433,
3449,3457,3461,3463,3467,3469,3491,3499,3511,3517,3527,3529,3533,3539,3541,3547,3557,3559,3571,3581,
3583,3593,3607,3613,3617,3623,3631,3637,3643,3659,3671,3673,3677,3691,3697,3701,3709,3719,3727,3733,
```


Приложение 1 (продолжение)

```
3739,3761,3767,3769,3779,3793,3797,3803,3821,3823,3833,3847,3851,3853,3863,3877,3881,3889,3907,3911,
3917,3919,3923,3929,3931,3943,3947,3967,3989,4001,4003,4007,4013,4019,4021,4027,4049,4051,4057,4073,
4079,4091,4093,4099,4111,4127,4129,4133,4139,4153,4157,4159,4177,4201,4211,4217,4219,4229,4231,4241,
4243,4253,4259,4261,4271,4273,4283,4289,4297,4327,4337,4339,4349,4357,4363,4373,4391,4397,4409,4421,
4423,4441,4447,4451,4457,4463,4481,4483,4493,4507,4513,4517,4519,4523,4547,4549,4561,4567,4583,4591,
4597,4603,4621,4637,4639,4643,4649,4651,4657,4663,4673,4679,4691,4703,4721,4723,4729,4733,4751,4759,
4783,4787,4789,4793,4799,4801,4813,4817,4831,4861,4871,4877,4889,4903,4909,4919,4931,4933,4937,4943,
4951,4957,4967,4969,4973,4987,4993,4999,5003,5009,5011,5021,5023,5039,5051,5059,5077,5081,5087,5099,
5101,5107,5113,5119,5147,5153,5167,5171,5179,5189,5197,5209,5227,5231,5233,5237,5261,5273,5279,5281,
5297,5303,5309,5323,5333,5347,5351,5381,5387,5393,5399,5407,5413,5417,5419,5431,5437,5441,5443,5449,
5471,5477,5479,5483,5501,5503,5507,5519,5521,5527,5531,5557,5563,5569,5573,5581,5591,5623,5639,5641,
5647,5651,5653,5657,5659,5669,5683,5689,5693,5701,5711,5717,5737,5741,5743,5749,5779,5783,5791,5801,
5807,5813,5821,5827,5839,5843,5849,5851,5857,5861,5867,5869,5879,5881,5897,5903,5923,5927,5939,5953,
5981,5987,6007,6011,6029,6037,6043,6047,6053,6067,6073,6079,6089,6091,6101,6113,6121,6131,6133,6143,
6151,6163,6173,6197,6199,6203,6211,6217,6221,6229,6247,6257,6263,6269,6271,6277,6287,6299,6301,6311,
6317,6323,6329,6337,6343,6353,6359,6361,6367,6373,6379,6389,6397,6421,6427,6449,6451,6469,6473,6481,
6491,6521,6529,6547,6551,6553,6563,6569,6571,6577,6581,6599,6607,6619,6637,6653,6659,6661,6673,6679,
6689,6691,6701,6703,6709,6719,6733,6737,6761,6763,6779,6781,6791,6793,6803,6823,6827,6829,6833,6841,
6857,6863,6869,6871,6883,6899,6907,6911,6917,6947,6949,6959,6961,6967,6971,6977,6983,6991,6997,7001,
7013,7019,7027,7039,7043,7057,7069,7079,7103,7109,7121,7127,7129,7151,7159,7177,7187,7193,7207,7211,
7213,7219,7229,7237,7243,7247,7253,7283,7297,7307,7309,7321,7331,7333,7349,7351,7369,7393,7411,7417,
7433,7451,7457,7459,7477,7481,7487,7489,7499,7507,7517,7523,7529,7537,7541,7547,7549,7559,7561,7573,
7577,7583,7589,7591,7603,7607,7621,7639,7643,7649,7669,7673,7681,7687,7691,7699,7703,7717,7723,7727,
7741,7753,7757,7759,7789,7793,7817,7823,7829,7841,7853,7867,7873,7877,7879,7883,7901,7907,7919,7927
```

Содержание