Министерство высшего образования Российской Федерации

###### Пензенский государственный университет

Кафедра «Информационная безопасность систем и технологий»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

курсовой работе по теме

«Программирование алгоритма цифровой подписи ГОСТ Р 34.10-94»

ПГУ 3.090105.001 ПЗ

Дисциплина КМЗИ

Группа

Разработала студентка

Проект принят с оценкой

Руководитель проекта

ПЕНЗА, 2006

**Реферат**

Пояснительная записка 24 с., 3 источника, 3 прил.

ЦИФРОВАЯ ПОДПИСЬ, ПРОГРАММА, ИНТЕРФЕЙС, ФУНКЦИЯ, ФАЙЛЫ, ОТКРЫТЫЙ КЛЮЧ, ЗАКРЫТЫЙ КЛЮЧ.

Объект исследования – алгоритм цифровой подписи ГОСТ Р 34.10-94.

Целью работы является получение навыков в программировании криптоалгоритмов на примере алгоритма цифровой подписи ГОСТ Р 34.10-94.

В процессе работы была разработана программа, которая осуществляет процедуру подписи сообщения и процедуру проверки подписи.

В результате выполнения работы были получены навыки в программировании алгоритма цифровой подписи ГОСТ Р 34.10-94.

**Содержание**

Введение

1. Пользовательский интерфейс программы

2. Руководство пользователя

3. Описание алгоритмов

3.1. Алгоритм функции формирования подписи

3.2. Алгоритм функции проверки подписи

Заключение

Список использованных источников

Блок схема алгоритма функции формирования подписи

Блок схема алгоритма функции проверки подписи

Листинг программы

**Введение**

В настоящее время широкое распространение получили персональные компьютеры. Они, без преувеличения, вошли во все сферы человеческой деятельности. Это максимально облегчило человеческий труд. Сейчас трудно представить, чтобы делал человек без компьютера. Предприятия, фирмы, офисы, банки - это ещё далеко не все области применения персонального компьютера.

Без использования криптографии сегодня немыслимо решение задач по обеспечению безопасности информации, связанных с конфиденциальностью и целостностью. Если до 1990г. криптография обеспечивала закрытие государственных линий связи, то в наши дни использование криптографических методов получило широкое распространение благодаря развитию компьютерных сетей и электронного обмена данными в различных областях[3].

Владение языками программирования позволяет значительно расширить круг выполняемых задач специалистом по защите информации и более полно использовать возможности компьютера.

Данная курсовая работа написана на языке С.

С – это универсальный язык программирования с компактным способом записи выражений, современными механизмами управления структурами данных и богатым набором операторов[1].

Целью курсовой работы является разработка программы, которая реализует процедуру подписи сообщения и процедуру проверки подписи.

**1. Пользовательский интерфейс программы**

Интерфейс – аппаратная или программная система сопряжения объектов с различными характеристиками. Программный интерфейс позволяет согласовывать программы с различными параметрами, а также дают возможность пользователю работать в операционной среде с той или иной степенью комфорта.

Методы и средства взаимодействия с аппаратными и программными средствами называют пользовательским интерфейсом [2]. Любая программа должна обладать пользовательским интерфейсом, иначе, кроме своего разработчика, она будет недоступна в понимании лицу с ней не знакомому.

Интерфейс бывает различных типов. Разработанная программа обладает интерактивным интерфейсом. Основным устройством управления в данном случае является клавиатура.

**2. Руководство пользователя**

Программа предназначена для выполнения подписи сообщения и проверки подписи.

После запуска программа предлагает выбрать действие:

Последовательный вызов процедур подписи сообщения и проверки подписи.

Подписать сообщение.

Проверить подпись.

Входные данные хранятся в файле initial data.txt:

*р –* простое число, удовлетворяющее требованию *2509<р<2512*;

*q* – простое число, удовлетворяющее требованию *2254<q<2256*;

*a* – число, которое рассчитывается по формуле *aq(mod p)=1* и удовлетворяет требованию *1<а<р-1*;

*x* – закрытый ключ, удовлетворяющий требованию *1<x<q*;

*y* – открытый ключ, который рассчитывается по формуле *y=ax(mod p)*;

*m* – хэш-функция по алгоритму ГОСТ Р 34.11.

Выбрав действие, программа производит необходимые действия и записывает результаты в файлы: цифровая подпись записывается в signature.txt, а результат проверки подписи – в verification result.txt.

**3. Описание алгоритмов**

**3.1 Алгоритм функции формирования подписи**

Функция формирования подписи *signature* формирует подпись, используя начальные параметры, расположенные в файле *initial data.txt*. Начальные параметры должны удовлетворять следующим условиям:

простое число *р: 2509<р<2512*;

простое число *q: 2254<q<2256*;

*a: 1<а<р-1, aq(mod p)=1*;

закрытый ключ *x: 1<x<q*;

открытый ключ *y: y=ax(mod p)*;

хэш-функция *h(M)=m* по алгоритму ГОСТ Р 34.11.

Для формирования подписи необходимо вычислить:

*r=ak(mod p)*

*r’=r(mod q)*

*s=xr’+km(mod q)*

Таким образом, цифровая подпись для сообщения *М* есть: *<r’>||<s>*.

Алгоритм функции формирования подписи приведен в Приложении А.

**3.2 Алгоритм функции проверки подписи**

Функция проверки подписи *verify* проверяет подпись, использую параметры:

цифровую подпись сообщения *М*;

открытый ключ;

хэш-функцию;

простые числа *р* и *q*.

Для проверки подписи необходимо вычислить:

*v=mq-2(mod q)*

*z1=sv(mod q)*

*z2=(q-r’)v(mod q)*

*u=(az1yz2(mod p))(mod q)*

Если условие *r’=u* выполнено – подпись подлинная. Результат проверки подписи находится в файле *verification result.txt*.

Алгоритм функции проверки подписи приведен в Приложении Б.

**4. Отладка и проверка работоспособности программы**

Программа была откомпилирована и отлажена.

Для проверки работоспособности программы был запущен файл **sign.exe**, были выбраны действия, которые необходимо выполнить. Например, необходимо сформировать цифровую подпись и проверить ее подлинность.

Текст входного файла *initial data.txt*:

Текст выходного файла *signature.txt*:

Текст выходного файла *verification result.txt*:

Оценка результатов показала, что программа работает верно.

Листинг программы приведен в Приложении В.

**Заключение**

Разработанная программа написана на языке Си. Программа осуществляет формирование цифровой подписи по алгоритму ГОСТ Р 34.10-94 и проверку цифровой подписи.

Таким образом, задание на курсовую работу выполнено в полном объеме.

**Список использованных источников**

1. Керниган Б., Ритчи Д. Язык программирования Си\ Пер. с англ., 3-е изд., испр. – СПб.: «Невский диалект», 2001. – 352с.
2. Информатика: Базовый курс \ С.В. Симонович и др. – СПб,: Питер, 2001. – 640 с.
3. Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. Основы криптографии: Учебное пособие, 2-е изд., испр. И доп. – М.:Гелиос АРВ, 2002. – 480с., ил.

**ПРИЛОЖЕНИЕ А**

Блок схема алгоритма функции формирования подписи

**ПРИЛОЖЕНИЕ Б**

Блок схема алгоритма функции проверки подписи

**ПРИЛОЖЕНИЕ В**

Листинг программы

**Sign.cpp**

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include "Interfs.h"

#include "function.h"

#define LOWORD(T) ((unsigned short)(T))

#define HIWORD(T) ((unsigned short)(T>>16))

#define MAKELONG(a,b) ((((unsigned long)(a))<<16)+b)

#define S 32

/\*addition\*/

void add(unsigned short add\_a[], unsigned short add\_b[],unsigned short add\_c[], int add\_l\_a\_b)

{unsigned long T;

unsigned short d;

d=0;

for(int i=0; i<=(add\_l\_a\_b-1); i++)

{T=(unsigned long)add\_a[i]+(unsigned long)add\_b[i]+d;

add\_c[i]=LOWORD(T);

d=HIWORD(T);}

add\_c[add\_l\_a\_b]=d;}

void sub(unsigned short sub\_a[], unsigned short sub\_b[],unsigned short sub\_c[], int sub\_l\_a\_b)

{unsigned long T;

unsigned short d;

d=0;

for(int i=0; i<=(sub\_l\_a\_b-1); i++)

{T=(unsigned long)sub\_a[i]-(unsigned long)sub\_b[i]-d;

sub\_c[i]=LOWORD(T);

if(HIWORD(T)==0) d=0;

else d=1;}

sub\_c[sub\_l\_a\_b]=d;}

/\*multiplication c=a\*b\*/

void mul(unsigned short mul\_a[], unsigned short mul\_b[],unsigned short mul\_c[],int mul\_l\_a, int mul\_l\_b)

{unsigned long T;

unsigned short d;

for(int z=0; z<=(mul\_l\_a-1); z++)

{mul\_c[z]=0;}

for(int i=0; i<=(mul\_l\_a-1); i++)

{d=0;

for(int j=0; j<=(mul\_l\_b-1); j++)

{T=(unsigned long) mul\_c[i+j]+(unsigned long) mul\_a[i]\*(unsigned long) mul\_b[j]+d;

mul\_c[i+j]=LOWORD(T);

d=HIWORD(T);}

mul\_c[i+mul\_l\_b]=d;}}

/\*c=(a\*b) mod m\*/

void mul\_mod(unsigned short mm\_a[], unsigned short mm\_b[], unsigned short mm\_n[], unsigned short mm\_ost[], unsigned short w1[], int mm\_l\_a\_b\_n, int t)

{unsigned short \*mm\_d;

mm\_d=new(unsigned short[mm\_l\_a\_b\_n+t]);

mul(mm\_a, mm\_b, mm\_d, mm\_l\_a\_b\_n, t);

Div(mm\_d, mm\_n, w1, mm\_ost, mm\_l\_a\_b\_n\*2, mm\_l\_a\_b\_n);}

/\*c=a(exp b) mod n\*/

void exp\_mod(unsigned short exp\_mod\_a[], unsigned short exp\_mod\_b[], unsigned short exp\_mod\_n[], unsigned short exp\_mod\_c[], unsigned short w1[], int exp\_mod\_l\_a\_n, int exp\_mod\_l\_b)

{unsigned short \*exp\_mod\_c\_dop;

exp\_mod\_c\_dop=new(unsigned short[exp\_mod\_l\_a\_n]);

int e;

for(int l=1; l<=(exp\_mod\_l\_a\_n-1); l++)

{exp\_mod\_c[l]=0;

exp\_mod\_c\_dop[l]=0;}

exp\_mod\_c\_dop[0]=1;

exp\_mod\_c[0]=1;

for(e=(16\*exp\_mod\_l\_b-1); e>=0; e--)

{mul\_mod(exp\_mod\_c, exp\_mod\_c, exp\_mod\_n, exp\_mod\_c\_dop, w1, exp\_mod\_l\_a\_n, exp\_mod\_l\_a\_n);

if(exp\_mod\_b[e/16] & (1<<(e%16)))

{mul\_mod(exp\_mod\_a, exp\_mod\_c\_dop, exp\_mod\_n, exp\_mod\_c, w1, exp\_mod\_l\_a\_n, exp\_mod\_l\_a\_n);}

else

{for(int h=0; h<=exp\_mod\_l\_a\_n-1; h++)

{exp\_mod\_c[h]=exp\_mod\_c\_dop[h];}}}}

/\*сравнение\*/

int comp (unsigned short a[], unsigned short p[], int S1)

{int x=1;

for(int i=0; i!=S1; i++)

{if (a[i]!=p[i]) x=0;}

if(x==0) return 1;

else return 0;}

/\*чтение из файла\*/

int file\_read(unsigned short t[], long curpos, int sz)

{FILE \*fp;

unsigned short k;

fp=fopen("initial data.txt","r");

for(int i=0;i<sz;i++)

{fseek(fp,curpos,SEEK\_SET);

fscanf(fp,"%x",&k);

t[i]=k;

curpos=curpos+5;}

fclose(fp);

return curpos;}

/\*запись в файл\*/

int file\_write(unsigned short t[], long curpos, int sz)

{FILE \*fp;

fp=fopen("signature.txt","w");

for(int i=0;i<sz;i++) fprintf(fp,"%x ",t[i]);

fclose(fp);

return curpos;}

unsigned short sign[S];

int main()

{FILE \*fp1;

unsigned short p[S];

unsigned short a[S];

unsigned short q[S/2];

unsigned short x[S/2];

unsigned short m[S/2];

unsigned short k[S/2];

unsigned short y[S];

long curpos=3;

long cp=file\_read(p,curpos,S);

curpos=cp+7;

cp=file\_read(a,curpos,S);

curpos=cp+7;

cp=file\_read(q,curpos,S/2);

curpos=cp+7;

cp=file\_read(x,curpos,S/2);

curpos=cp+7;

cp=file\_read(m,curpos,S/2);

curpos=cp+7;

cp=file\_read(k,curpos,S/2);

curpos=cp+7;

cp=file\_read(y,curpos,S);

char symb;

int c;

printf("------------------------------------------------------------\n");

printf("|Select action: |\n");

printf("|1.Serial calling rountines \"Get signature\" and \"Verify signature\". |\n");

printf("|2.Get signature |\n");

printf("|3.Verify signature |\n");

printf("|Press 0 to exit |\n");

printf("----------------------------------------------------------\n\n");

do

{symb=getch();

c=atoi(&symb);

int rs;

switch(c)

{case 0:

return 0;

case 1:

signature(p,a,q,x,m,k,sign);

file\_write(sign, 0, S);

rs=verify(p,a,q,y,m,sign);

fp1=fopen("verification result.txt","w+");

if(fp1==NULL) printf("Error: couldn't open file");

if(rs==0) fprintf(fp1, "Verification successful: u=r', signature is authentic\n");

else fprintf(fp1, "WARNING: verification unsuccessful: u!=r', signature is non-authentic\n");

printf("Operations complete. Result of operation writed to \"verification result.txt\"\n");

fclose(fp1);

break;

case 2:

signature(p,a,q,x,m,k,sign);

file\_write(sign, 0, S);

printf("Operation complete. Result of operation writed to \"signature.txt\"\n");

break;

case 3:

verify(p,a,q,y,m,sign);

rs=verify(p,a,q,y,m,sign);

if(rs==0) fprintf(fp1, "Verification successful: u=r', signature is authentic\n");

else fprintf(fp1, "WARNING: verification unsuccessful: u!=r', signature is non-authentic\n");

printf("Operation complete. Result of operation writed to \"verification result.txt\"\n");

break;

default: printf("Wrong number. Try again\n"); break;}}

while(1);}

**Function.cpp**

#include "function.h"

#include "INTERFS.h"

void signature(unsigned short sign\_p[], unsigned short sign\_a[], unsigned short sign\_q[], unsigned short sign\_x[], unsigned short sign\_m[], unsigned short sign\_k[], unsigned short sign[])

{unsigned short w[S\*2];

unsigned short res[S];

unsigned short result[S];

unsigned short tmp[S/2];

unsigned short tmp1[S/2];

unsigned short sum[S+1];

unsigned short s[S+1];

exp\_mod(sign\_a,sign\_k,sign\_p,res,w,S,S/2);

Div(res,sign\_q,result,tmp,S,S/2);

mul(sign\_x,tmp,res,S/2,S/2);

mul(sign\_k,sign\_m,result,S/2,S/2);

add(res,result,sum,S);

Div(sum,sign\_q,s,tmp1,S,S/2);

for(int i=S/2-1, j=S-1; i>=0; i--, j--)

{sign[j]=tmp[i];

sign[j-16]=tmp1[i];}}

int verify(unsigned short p[],unsigned short a[],unsigned short q[],unsigned short y[], unsigned short m[],unsigned short sign[])

{unsigned short v[16];

unsigned short q\_2[16];

unsigned short w[S];

unsigned short s[S/2];

unsigned short tmp[S];

unsigned short tmp\_[S];

unsigned short tmp\_1[S\*2];

unsigned short tmp\_2[S\*2];

unsigned short tmp1[S/2];

unsigned short z1[S/2];

unsigned short z2[S/2];

unsigned short r\_[S/2];

for(int i=S/2-1; i>=0; i--) q\_2[i]=q[i];

q\_2[0]=q\_2[0]-0x0002;

exp\_mod(m,q\_2,q,v,w,S/2,S/2);

for(int b=S/2-1, j=S-1; b>=0; b--, j--)

{s[b]=sign[b];

r\_[b]=sign[j];}

mul(s,v,w,S/2,S/2);

Div(w,q,tmp,z1,S,S/2);

sub(q,r\_,tmp1,S/2);

mul(tmp1,v,w,S/2,S/2);

Div(w,q,tmp,z2,S,S/2);

exp\_mod(a,z1,p,tmp,w,S,S/2);

exp\_mod(y,z2,p,tmp\_,w,S,S/2);

mul(tmp,tmp\_,tmp\_1,S,S);

Div(tmp\_1,p,tmp\_2,tmp,S\*2,S);

Div(tmp,q,tmp\_,s,S,S/2);

return comp(s,r\_,S/2);}