МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»**

Кафедра теоретических основ компьютерной безопасности и криптографии

**Скрытый канал связи**

ОТЧЁТ

ПО ДИСЦИПЛИНЕ

«КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ»

студента 5 курса 531 группы

специальности 10.05.01 Компьютерная безопасность

факультета компьютерных наук и информационных технологий

Арбузова Матвея Александровича

|  |  |  |
| --- | --- | --- |
| Преподаватель  аспирант | \_\_\_\_\_\_\_\_\_\_ | Р. А. Фарахутдинов |
|  | подпись, дата |  |

Саратов 2023

**1 Постановка задачи**

Необходимо реализовать скрытый канал связи на основе DSA.

**2 Теоретические сведения**

Предположим, Алису и Боба арестовали и посадили в разные тюрьмы. Надзиратель Уолтер позволяет Алисе и Бобу обмениваться сообщениями, но запрещает их шифровать. Уолтер подозревает, что они замышляют побег, поэтому хочет читать всё, что они пишут.

Алисе и Бобу необходимо согласовать свои планы, чтобы сделать это, они должны обмануть надзирателя и найти метод передачи секретной информации. С этой целью они могут под носом Уолтера создать скрытый канал – тайный канал связи, по которому передаются сообщения, сами по себе не содержащие секретную информацию.

Густавус Симмонс выдвинул концепцию создания скрытого канала с помощью обычного алгоритма цифровой подписи: скрытые сообщения спрятаны в том, что выглядит как обычная цифровая подпись. Уолтер видит, как туда и обратно передаются подписанные безобидные сообщения, однако действительно стоящая информация передаётся незаметно для него по скрытому каналу.

По существу, алгоритм подписи, используемый в крытом канале, ничем не отличается от обычного алгоритма создания подписи, по крайней мере, для Уолтера. Последний не только не может читать сообщения, передаваемые по скрытому каналу, но даже не подозревает о его существовании.

В схеме DSA существует несколько скрытых каналов, одни из них позволяет Алисе и Бобу обмениваться в каждой подписи одним битом скрытой информации.

**Скрытый канал на основе DSA**

*Вход*: – битовая длина числа такая, что и , – битовая длина секретного сообщения .

*Выход*: Скрытое сообщение, которое получил Боб.

**1) Генерация основных параметров схемы:**

Шаг 1. Выбирается простое число , битовая длина которого равна 160, при этом удовлетворяет условию: , где – безобидное сообщение, – -ти битовая строка, полученная от с помощью алгоритма SHA;

Шаг 2. Вычисляется простое число по формуле: ;

Шаг 3. Вычисляется по формуле: (mod p), где – произвольное число от до такое, что (в большинстве случаев удовлетворяет данному условию);

Шаг 4. Вычисляется большое простое число , отличающееся от – секретный ключ Алисы и Боба для скрытого канала;

Шаг 5. Выбирается от до – закрытый ключ Алисы;

Шаг 6. Вычисляется – открытый ключ Алисы.

Элементы являются открытыми.

**2) Для каждого бита секретного сообщения выполняются пункты 2.1)-2.4).**

**2.1) Генерация подписи для сообщения m:**

Шаг 1. Алиса выбирает от до ; n = p ⋅ q жЖ

Шаг 2. Алиса вычисляет по формуле , если текущий бит секретного сообщения равен , то должно быть квадратичным невычетом по модулю , если текущий бит секретного сообщения равен , то должно быть квадратичным вычетом по модулю . Если данные условия не выполняются или , то перейти на шаг 1 и сгенерировать новое ;

Шаг 3. Алиса вычисляет по формуле . Если , то перейти на шаг 1 и сгенерировать новое .

**2.2) Алиса посылает Бобу сообщение и подпись .**

**2.3) Проверка подписи:**

Шаг 1. Боб вычисляет ;

Шаг 2. Боб вычисляет ;

Шаг 3. Боб вычисляет ;

Шаг 4. Боб вычисляет . Если , то подпись прошла проверку, иначе нет.

**2.4)** **Восстановление скрытого бита:** если квадратичный невычет по модулю , то Боб восстанавливает 0, если квадратичный вычет по модулю , то Боб восстанавливает 1.

**3 Практическая реализация**

**3.1 Описание программы**

Программа была написана на языке C++, и имеет множество функций.

Функция является точкой старта программы и отвечает за проверку корректности введённых, при запуске программы, битовой длины числа , битовой длины секретного сообщения и количества подробно описанных итераций.

Функция содержит все шаги описанной выше схемы, при этом для генерации простых чисел , и получения используются функции из библиотеки .

В программе используются большие числа, работать с которыми позволяет подключённая библиотека , кроме того, силами данной библиотеки осуществляется генерация случайных чисел из заданного диапазона в функции .

Для поиска обратного элемента и двух чисел в поле используется расширенный алгоритм Евклида – функция . Для проверки числа на простоту используется алгоритм Миллера-Рабина, который реализован в функции . Генерация безобидного и секретного сообщений происходит в функции . Проверка на квадратичный вычет и невычет выполняется с помощью символа Якоби – функция . Кроме того, в программе реализована функция быстрого возведения в степень по модулю, которая используется в большом количестве шагов – функция .

**3.2 Результаты тестирования программы**

Отсутствие одного или нескольких параметров, при запуске программы, ведёт к соответствующей ошибке, данный запуск представлен на рисунке 1.

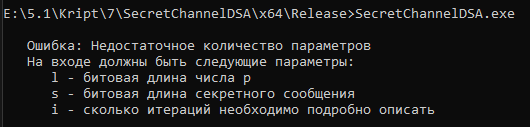


Рисунок 1 – Запуск программы без параметров

Число должно делиться на 64 и быть в диапазоне от 512 до 1024 – рисунок 2.

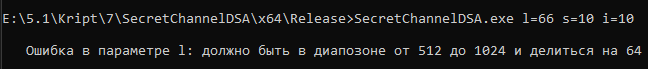


Рисунок 2 – Запуск программы с параметром , не удовлетворяющим условиям

Число должно быть больше 0 – рисунок 3.

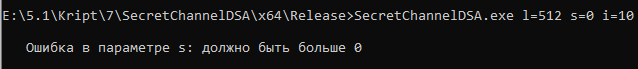


Рисунок 3 – Запуск программы с параметром , меньшим 1

Число должно быть в промежутке от 0 до – рисунок 4.



Рисунок 4 – Запуск программы с параметром , больше

На рисунках 5-6 представлен успешный запуск программы, в котором подробно выводятся две итерации отправки бит секретного сообщения из десяти.

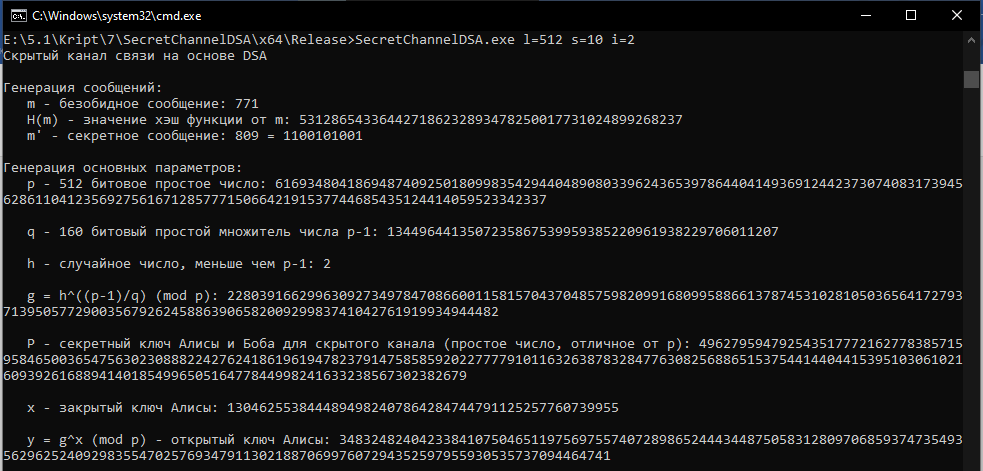


Рисунок 5 – Запуск программы с подробным выводом 2-х итераций из 10-ти (часть1)

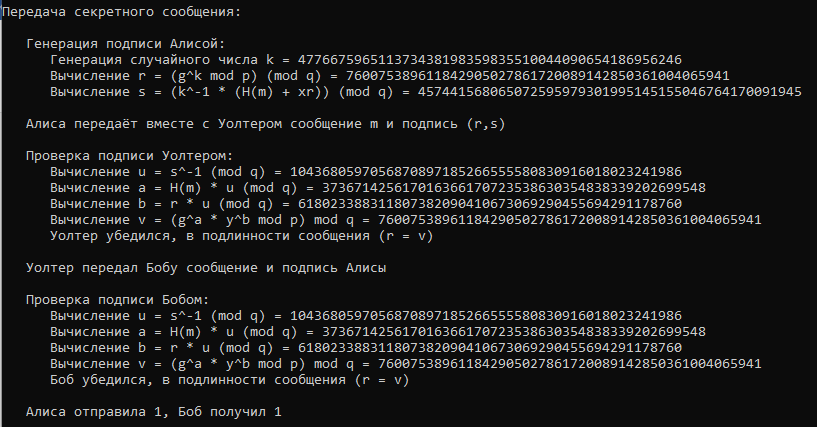


Рисунок 6 – Запуск программы с подробным выводом 2-х итераций из 10-ти (часть 2)

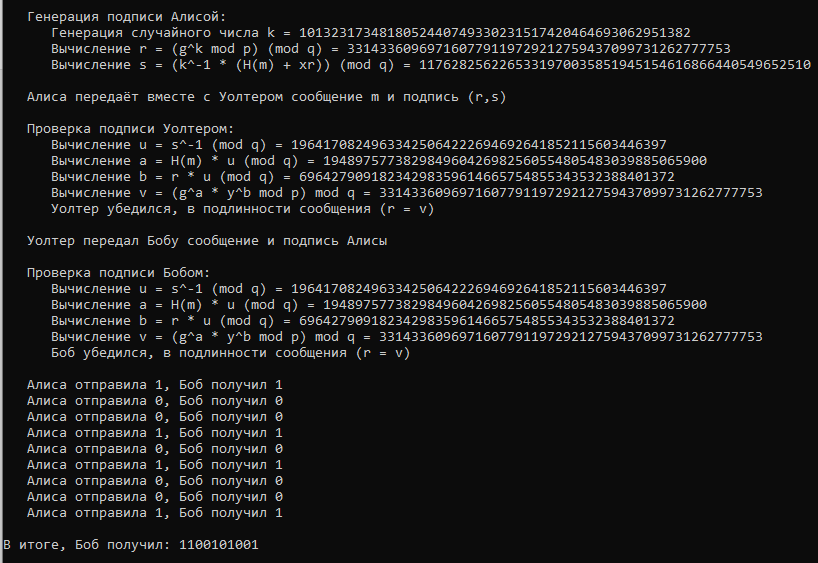


Рисунок 7 – Запуск программы с подробным выводом 2-х итераций из 10-ти (часть 3)

**Листинг кода**

#include <iostream>

#include <string>

#include "osrng.h"

#include "dsa.h"

#include "dh.h"

#include <random>

#include "files.h"

#include <set>

#include <hex.h>

#include <boost/multiprecision/cpp\_int.hpp>

#include <boost/random/uniform\_int.hpp>

#include <boost/random/variate\_generator.hpp>

#include <boost/multiprecision/cpp\_dec\_float.hpp>

using namespace CryptoPP;

using namespace std;

using namespace boost::multiprecision;

using namespace boost::random;

AutoSeededRandomPool rng;

cpp\_int ModNegative(cpp\_int a, cpp\_int p) {

if (a < 0)

a = a + p \* (((-1 \* a) / p) + 1);

return a % p;

}

vector <cpp\_int> ExtendedEuclid(cpp\_int a, cpp\_int b) {

vector <cpp\_int> res(3);

if (a == 0) {

res = { b, 0, 1 };

return res;

}

vector <cpp\_int> c = ExtendedEuclid(b % a, a);

res = { c[0], c[2] - (b / a) \* c[1], c[1] };

return res;

}

cpp\_int Exponentiation(cpp\_int x, cpp\_int n, cpp\_int m, int k) {

cpp\_int N = n, Y = 1, Z = x;

if (k == 1)

Z = Z % m;

else

N--;

while (N != 0) {

cpp\_int lastN = N % 2;

N = N / 2;

if (lastN == 0) {

Z = (Z \* Z);

if (k == 1)

Z = Z % m;

continue;

}

Y = (Y \* Z);

if (k == 1)

Y = Y % m;

if (N == 0)

break;

Z = (Z \* Z);

if (k == 1)

Z = Z % m;

}

if (k == 1)

Y = Y % m;

return Y;

}

cpp\_int Jac(cpp\_int a, cpp\_int b) {

if (ExtendedEuclid(a, b)[0] != 1)

return 0;

else {

int r = 1;

while (a != 0) {

cpp\_int t = 0;

while (a % 2 == 0) {

t = t + 1;

a = a / 2;

}

if (t % 2 != 0)

if (Exponentiation(b, 1, 8, 1) == 3 || Exponentiation(b, 1, 8, 1) == 5)

r = r \* (-1);

if (Exponentiation(a, 1, 4, 1) == 3 && Exponentiation(b, 1, 4, 1) == 3)

r = r \* (-1);

cpp\_int c = a;

if (c != 0)

a = Exponentiation(b, 1, c, 1);

b = c;

}

return r;

}

}

bool MilRab(cpp\_int p, cpp\_int k) {

if (p == 1 || p == 2 || p == 3)

return true;

if (p % 2 == 0)

return false;

cpp\_int t = p - 1;

cpp\_int s = 0;

while (t % 2 == 0) {

t = t / 2;

s++;

}

for (cpp\_int i = 0; i < k; i++) {

cpp\_int a = rand() % (p - 3) + 2;

if (ExtendedEuclid(p, a)[0] > 1)

return false;

cpp\_int x = Exponentiation(a, t, p, 1);

if (x == 1 || x == p - 1)

continue;

for (cpp\_int g = 1; g < s; g++) {

x = x \* x % p;

if (x == 1)

return false;

if (x == p - 1)

break;

}

if (x != p - 1)

return false;

}

return true;

}

cpp\_int Random(cpp\_int minim, cpp\_int maxim) {

random\_device gen;

boost::random::uniform\_int\_distribution<cpp\_int> ui(minim, maxim);

return ui(gen);

}

cpp\_int IntegerToCppint(const Integer number) {

ostringstream oss;

oss << number;

string str(oss.str());

str.erase(str.size() - 1, 1);

cpp\_int res(str);

return res;

}

cpp\_int GenMess(int l) {

cpp\_int m = 1;

cpp\_int deg = 2;

for (int i = 1; i < l - 1; i++) {

m = m + (deg \* (rand() % 2));

deg = deg \* 2;

}

m = m + deg;

return m;

}

cpp\_int toSHA1(string mess) {

string digest, res;

SHA1 hash;

hash.Update((const byte\*)mess.data(), mess.size());

digest.resize(hash.DigestSize());

hash.Final((byte\*)&digest[0]);

StringSource(digest, true, new HexEncoder(new StringSink(res)));

istringstream stream(res);

cpp\_int dec;

stream >> hex >> dec;

return dec;

}

vector <cpp\_int> toBits(cpp\_int n) {

vector <cpp\_int> res1, res2;

while (n > 1) {

res1.push\_back(n % 2);

n = n / 2;

}

res1.push\_back(n);

for (int i = res1.size() - 1; i > -1; i--)

res2.push\_back(res1[i]);

return res2;

}

bool checkSignature(cpp\_int s, cpp\_int q, cpp\_int H, cpp\_int r, cpp\_int g, cpp\_int p, cpp\_int y, bool check, string str) {

cpp\_int u = ModNegative(ExtendedEuclid(s, q)[1], q);

cpp\_int a = (H \* u) % q;

cpp\_int b = (r \* u) % q;

cpp\_int prom1 = Exponentiation(g, a, p, 1);

cpp\_int prom2 = Exponentiation(y, b, p, 1);

cpp\_int v = ((prom1 \* prom2) % p) % q;

if (check) {

cout << "\n Проверка подписи " << str << "ом:\n";

cout << " Вычисление u = s^-1 (mod q) = " << u << "\n";

cout << " Вычисление a = H(m) \* u (mod q) = " << a << "\n";

cout << " Вычисление b = r \* u (mod q) = " << b << "\n";

cout << " Вычисление v = (g^a \* y^b mod p) mod q = " << v << "\n";

if (r == v)

cout << " " << str << " убедился, в подлинности сообщения (r = v)\n";

}

if (r == v)

return true;

return false;

}

void SecretChannelDSA(int l, int sizem, int iter) {

//Сообщения

cpp\_int m, msh, H;

m = GenMess(sizem);

H = toSHA1(to\_string(m));

msh = GenMess(sizem);

vector <cpp\_int> bitsMsh = toBits(msh);

cout << "\nГенерация сообщений:\n";

cout << " m - безобидное сообщение: " << m;

cout << "\n H(m) - значение хэш функции от m: " << H;

cout << "\n m' - секретное сообщение: " << msh << " = ";

for (int i = 0; i < bitsMsh.size(); i++)

cout << bitsMsh[i];

//p и q

DH dh1; cpp\_int p , q;

cpp\_int deg = Exponentiation(2, l - 160, 1, 2);

for (;;) {

do {

dh1.AccessGroupParameters().GenerateRandomWithKeySize(rng, 160);

q = IntegerToCppint(dh1.GetGroupParameters().GetModulus());

} while (H >= q);

p = deg \* q + 1;

if (MilRab(p, 10))

break;

}

cout << "\n\nГенерация основных параметров:\n";

cout << " p - " << l << " битовое простое число: " << p;

cout << "\n\n q - " << 160 << " битовый простой множитель числа p-1: " << q;

//генерация h и g

cpp\_int h = 2, g;

do {

g = Exponentiation(h, (p - 1) / q, p, 1);

if (Exponentiation(g, q, p, 1) != 1) {

g = 0;

h = Random(2, p - 2);

continue;

}

} while (g < 2);

cout << "\n\n h - случайное число, меньше чем p-1: " << h;

cout << "\n\n g = h^((p-1)/q) (mod p): " << g << "\n";

//генерация P

cpp\_int P = p;

DH dh;

while (p == P) {

dh.AccessGroupParameters().GenerateRandomWithKeySize(rng, rand() % 1020 + 4);

P = IntegerToCppint(dh.GetGroupParameters().GetModulus());

}

cout << "\n P - секретный ключ Алисы и Боба для скрытого канала (простое число, отличное от p): " << P;

//открытый и закрытый ключ Алисы

DH dh2;

cpp\_int x, y;

x = Random(1, q - 1);

y = Exponentiation(g, x, p, 1);

cout << "\n\n x - закрытый ключ Алисы: " << x;

cout << "\n\n y = g^x (mod p) - открытый ключ Алисы: " << y << "\n";

//Протокол

cout << "\nПередача секретного сообщения:";

iter--;

vector <cpp\_int> bitsBob;

for (int i = 0; i < bitsMsh.size();) {

//Алиса

cpp\_int k = Random(1, q - 1);

if (ExtendedEuclid(k, q)[0] != 1)

continue;

cpp\_int r = Exponentiation(g, k, p, 1) % q;

if (bitsMsh[i] == 1) {

if (Jac(r, P) != 1)

continue;

}

else{

if (Jac(r, P) != -1)

continue;

}

if (r == 0)

continue;

cpp\_int s = (ModNegative(ExtendedEuclid(k, q)[1], q) \* (H + x \* r)) % q;

if (s == 0 || ExtendedEuclid(s, q)[0] != 1)

continue;

bool checkIter = iter >= i;

if (checkIter) {

cout << "\n\n Генерация подписи Алисой:";

cout << "\n Генерация случайного числа k = " << k;

cout << "\n Вычисление r = (g^k mod p) (mod q) = " << r;

cout << "\n Вычисление s = (k^-1 \* (H(m) + xr)) (mod q) = " << s;

cout << "\n\n Алиса передаёт вместе с Уолтером сообщение m и подпись (r,s)\n";

}

//Уолтер

bool checkSignW = checkSignature(s, q, H, r, g, p, y, iter >= i, "Уолтер");

if (checkSignW) {

if (checkIter)

cout << "\n Уолтер передал Бобу сообщение и подпись Алисы\n";

//Боб

bool checkSignB = checkSignature(s, q, H, r, g, p, y, iter >= i, "Боб");

if (checkSignB) {

cpp\_int quad = Jac(r, P);

cpp\_int bitBob;

if (quad == 1)

bitBob = 1;

if (quad == -1)

bitBob = 0;

cout << "\n Алиса отправила " << bitsMsh[i] << ", Боб получил " << bitBob;

bitsBob.push\_back(bitBob);

}

}

i++;

}

cout << "\n\nВ итоге, Боб получил: ";

for (int i = 0; i < bitsBob.size(); i++)

cout << bitsBob[i];

cout << "\n";

}

void ErrMess() {

cerr << " На входе должны быть следующие параметры:\n"

<< " l - битовая длина числа p\n"

<< " s - битовая длина секретного сообщения\n"

<< " i - сколько итераций необходимо подробно описать\n";

}

int main(int argc, char\*\* argv) {

setlocale(LC\_ALL, "Russian");

srand(time(NULL));

if (argc < 2) {

cerr << "\n Ошибка: Недостаточное количество параметров\n";

ErrMess();

return 0;

}

int s = -1, l = -1, iter = -1;

string prom1, prom2;

for (int i = 1; i < argc; i++) {

prom1 = argv[i];

try {

prom1.erase(2, prom1.size() - 2);

if (prom1 == "l=") {

prom2 = argv[i];

prom2.erase(0, 2);

try {

l = stoi(prom2);

}

catch (exception) {

cerr << "\n Ошибка в параметре l: передано некорректное число\n";

ErrMess();

return 0;

}

continue;

}

if (prom1 == "s=") {

prom2 = argv[i];

prom2.erase(0, 2);

try {

s = stoi(prom2);

}

catch (exception) {

cerr << "\n Ошибка в параметре s: передано некорректное число\n";

ErrMess();

return 0;

}

continue;

}

if (prom1 == "i=") {

prom2 = argv[i];

prom2.erase(0, 2);

try {

iter = stoi(prom2);

}

catch (exception) {

cerr << "\n Ошибка в параметре i: передано некорректное число\n";

ErrMess();

return 0;

}

continue;

}

}

catch (exception) {

continue;

}

}

if (l < 0 || s < 0 || iter < 0) {

cerr << "\n Ошибка: некоторые параметры отсутствуют или пустые\n";

ErrMess();

return 0;

}

if (l < 512 || l > 1024 || l % 64 != 0) {

cerr << "\n Ошибка в параметре l: должно быть в диапозоне от 512 до 1024 и делиться на 64\n";

return 0;

}

if (s < 1) {

cerr << "\n Ошибка в параметре s: должно быть больше 0\n";

return 0;

}

if (iter < 0 || iter > s) {

cerr << "\n Ошибка в параметре i: должно быть в диапозоне от 0 до " << s << "\n";

return 0;

}

cout << "Скрытый канал связи на основе DSA\n";

SecretChannelDSA(l, s, iter);

return 0;

}