Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

по лабораторной работе № 4

по курсу «Защита информации»

на тему: «Реализация программы для создания и проверки электронной подписи с использованием алгоритма шифрования RSA и алгоритма хеширования SHA-1»

Вариант № 2

Студент ИУ7-73Б (Группа) _____ (Подпись, дата) Марченко В. (И. О. Фамилия)

Преподаватель _____ (Подпись, дата) Чиж И. С. (И. О. Фамилия)

2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

RSA (аббревиатура от фамилий Rivest, Shamir и Adleman) — криптографический алгоритм с открытым ключом, основывающийся на вычислительной сложности задачи факторизации больших полупростых чисел. Криптосистема RSA стала первой системой, пригодной и для шифрования, и для цифровой подписи. Алгоритм используется в большом числе криптографических приложений, включая PGP, S/MIME, TLS/SSL, IPSEC/IKE и других [1].

Secure Hash Algorithm 1 — алгоритм криптографического хеширования. Описан в RFC 3174. Для входного сообщения произвольной длины (максимум $2^{64} - 1$ бит, что примерно равно 2 эксабайта) алгоритм генерирует 160-битное (20 байт) хеш-значение, называемое также дайджестом сообщения, которое обычно отображается как шестнадцатеричное число длиной в 40 цифр. Используется во многих криптографических приложениях и протоколах. Принципы, положенные в основу SHA-1, аналогичны тем, которые использовались Рональдом Ривестом при проектировании MD4 [2].

Целью данной лабораторной работы является программная реализация алгоритма шифрования RSA и алгоритма хеширования SHA-1.

Задачи лабораторной работы:

1) изучить принцип работы алгоритма шифрования RSA;

2) изучить принцип работы алгоритма хеширования SHA-1;

3) разработать программное обеспечение для создания и проверки электронной подписи;

4) протестировать разработанное программное обеспечение.

1 Алгоритм шифрования RSA

Генерация ключей. Для использования RSA нужно сгенерировать два ключа — открытый (public) и закрытый (private). Сначала выбираются два простых числа p и q. Например, $p = 13$ и $q = 11$. Затем вычисляется $n = p \times q = 143$. n является второй частью обоих ключей. Далее вычисляется $\phi(n) = (p-1) \times (q-1) = 120$. После этого нужно найти число e, которое является взаимно простым с $\phi(n)$. Число e должно быть таким, чтобы выполнялось неравенство $1 < e < \phi(n)$. Пусть $e = 13$. Это первая часть открытого ключа. Число d можно найти из формулы $d \times e \ mod \ \phi(n) = 1$. Подойдет значение $d = 37$. Таким образом, открытый ключ — пара $(13,\ 143)$, а закрытый — $(37,\ 143)$.

Замечание: шифровать с помощью ключа можно только такие значения, которые меньше, чем n.

Шифрование. Допустим, нужно зашифровать число $m = 13$. Берем открытый ключ получателя — $(13,\ 143)$ — и вычисляем. $c = m^e \ mod \ n = 13^{13} \ mod \ 143 = 52$.

Расшифрование. Берем закрытый ключ получателя — $(37,\ 143)$ — и вычисляем. $m = c^d \ mod \ n = 52^{37} \ mod \ 143 = 13$.

2 Алгоритм хеширования SHA-1

SHA-1 реализует хеш-функцию, построенную на идее функции сжатия. Входами функции сжатия являются блок сообщения длиной 512 бит и выход предыдущего блока сообщения. Выход представляет собой значение всех хеш-блоков до этого момента. Хеш-значением всего сообщения является выход последнего блока.

Длина входного блока — 512 бит. Последний блок сообщения всегда модифицируется. Если его длина меньше 56 байт, в конец сообщения добавляется единичный бит, а далее все заполняется нулями, кроме последних 64-х бит. Они являются зарезервированными. В них записывается длина исходного сообщения в битах. Если последний блок имеют длину 56 или более байт, то добавляется новый блок и заполняется аналогично.

Хеширование осуществляется на протяжении 80-и раундов. Для каждых двадцати раундов есть своя функция $f(t;\ B,\ C,\ D)$ и константа $K(t)$ (описаны в RFC 3174).

Сначала блок длиной 512 бит попадает на вход. Он делится на 16 слов длиной 32 бита каждое. Обозначим эти слова как $W(0),\ W(1),\ ...,\ W(15)$. Далее с помощью этих 16-и слов вычислим еще 64 слова по правилу: $W(t) = (W(t-3) \oplus W(t-8) \oplus W(t-14) \oplus W(t-16)) << 1,\ t \in [16,\ 80)$, где $<<$ — циклический побитовый сдвиг влево, $\oplus$ — операция XOR. Каждый раунд обрабатывает одно слово.

Пусть $A = H_0,\ B = H_1,\ C = H_2,\ D = H_3,\ E = H_4$ — текущие значения хеш-слов. Значения $H_0,\ ...,\ H_4$ изначально известны (описаны в RFC 3174). Затем каждый раунд происходят следующие преобразования: $T = A << 5 + f(t;\ B,\ C,\ D) + E + W(t) + K(t)$, где $z = x + y$ вычисляется как $z = (x + y)\ mod\ 2^{32}$. $E = D,\ D = C,\ C = B << 30,\ B = A,\ A = T$.

После 80-го раунда вычисляются новые значения хеш-слов: $H_0 = H_0 + A,\ H_1 = H_1 + B,\ ...,\ H_4 = H_4 + E$.

Данные шаги выполняются для всех 512-битных блоков $M(i)$ исходного сообщения M.

Результат работы алгоритма SHA-1 — 160-битное значение, которое вычисляется как $H = H_0H_1H_2H_3H_4$.

3 Электронная подпись

Электронная подпись (ЭП), электронная цифровая подпись (ЭЦП), цифровая подпись (ЦП) позволяет подтвердить авторство электронного документа (будь то реальное лицо или, например, аккаунт в криптовалютной системе). Подпись связана как с автором, так и с самим документом с помощью криптографических методов и не может быть подделана с помощью обычного копирования [3].

ЭЦП — это реквизит электронного документа, полученный в результате криптографического преобразования информации с использованием закрытого ключа подписи и позволяющий проверить отсутствие искажения информации в электронном документе с момента формирования подписи (целостность), принадлежность подписи владельцу сертификата ключа подписи (авторство), а в случае успешной проверки подтвердить факт подписания электронного документа (неотказуемость) [3].

Как было сказано в разделе про RSA, с помощью этого алгоритма можно не только шифровать данные, но и создавать электронную подпись. Разница в том, что при создании ЭП шифрование происходит с помощью закрытого ключа отправителя, а расшифрование — с помощью открытого ключа отправителя.

На рисунке 3.1 показана схема вычисления электронной подписи, а на рисунке 3.2 — схема проверки электронной подписи.

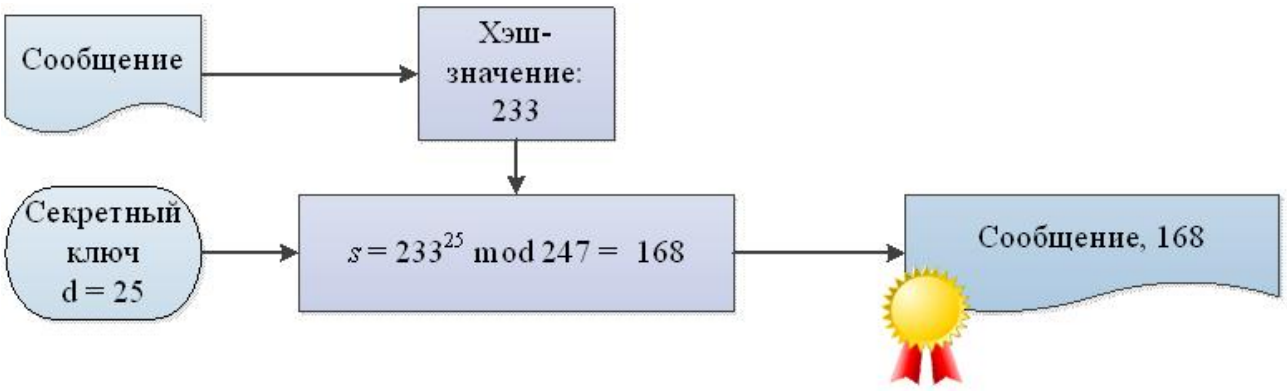

Рисунок 3.1 – Схема вычисления электронной подписи [4]

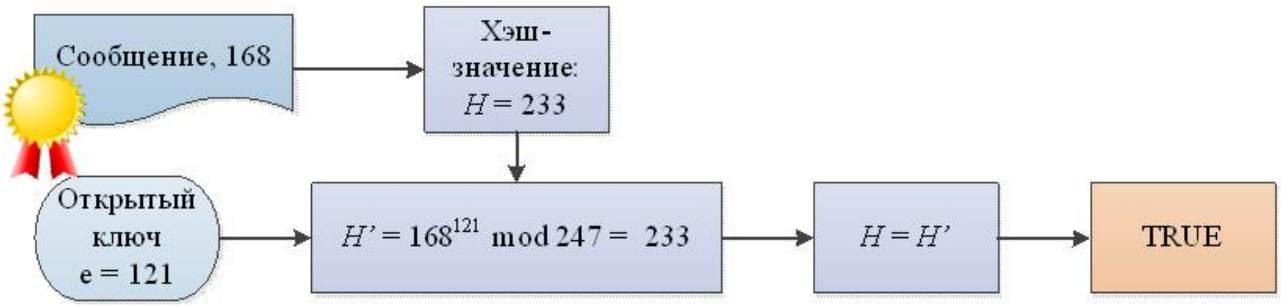

Рисунок 3.2 – Схема проверки электронной подписи [4]

4 Требования к входным данным

Программа работает в нескольких режимах и принимает разное количество аргументов командной строки. Чтобы узнать информацию о режимах и аргументах, можно запустить программу с ключом –**help** или **-h**.

Режим генерации RSA ключей: –**rsa-keys**. Простые числа p и q читаются из файлов cfg/p.txt и cfg/q.txt соответственно.

Режим шифрования/расшифрования с помощью алгоритма RSA: –**rsa exp_file mod_file in_file out_file**, где exp_file — путь к файлу, в котором записана закрытая/открытая экспонента, mod_file — путь к файлу, в котором записан модуль, in_file — входной файл, out_file — выходной файл.

Режим хеширования: –**sha in_file out_file**, где in_file — входной файл, out_file — выходной файл.

При наличии ошибок в аргументах командной строки программа выдаст сообщение об ошибке и завершится.

Программное обеспечение для создания и проверки электронной подписи было написано на языке программирования C.

Программа работает с файлами любых типов.

5 Тестирование программного обеспечения

В таблице 5.1 приведены тесты для проверки корректности работы реализованного программного обеспечения.

Таблица 5.1 – Тесты

Описание	**Открытый текст**	**Результат хеширования**
Неправильное колво аргументов командной строки		Error: no options were given. Use -h or –help for description. parameters.
Пустой входной файл		da39a3ee 5e6b4b0d 3255bfef 95601890 afd80709
Текст длиной менее 64 байт	Sha	ba79baeb 9f10896a 46ae7471 5271b7f5 86e74640
Текст длиной более 64 байт	В чащах юга жил бы цитрус? Да, но фальшивый экземпляр!	9e32295f 8225803b b6d5fdfc c0674616 a4413c1b

Все тесты пройдены успешно.

ЗАКЛЮЧЕНИЕ

В результате выполнения данной лабораторной работы были реализованы алгоритм шифрования RSA и алгоритм хеширования SHA-1.

Были выполнены следующие задачи:

1) изучен принцип работы алгоритма шифрования RSA;
2) изучен принцип работы алгоритма хеширования SHA-1;
3) разработано программное обеспечение для создания и проверки электронной подписи;
4) протестировано разработанное программное обеспечение.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Википедия*. RSA. — 2023. — (Дата обращения: 10.11.2023). `https://ru.wikipedia.org/wiki/RSA`.
2. *Википедия*. SHA-1. — 2023. — (Дата обращения: 10.11.2023). `https://ru.wikipedia.org/wiki/SHA-1`.
3. *Википедия*. Digital signature. — 2023. — (Дата обращения: 10.11.2023). `https://en.wikipedia.org/wiki/Digital_signature`.
4. *Береснева А.* Гайд по криптографии: что такое электронная цифровая подпись и как она работает. — 2023. — (Дата обращения: 10.11.2023). `https://xakep.ru/2016/12/15/crypto-part5/`.