Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Исследование алгоритмов генерации

и верификации ЭЦП

Студент: Раченок И.А.

ФИТ 3 курс 4 группа

Преподаватель:

Сазонова Дарья Владимировна

Минск 2023

## 1. Описание приложения

Приложение позволяет выполнить следующие задачи:

* генерация ЭЦП на основе алгоритмов RSA, Эль-Гамаля и Шнора;
* оценка времени выполнения указанных процедур при реальных параметрах.

## 2. Методика выполнения поставленных задач

## 2.1. Алгоритм RSA

Создание ЭЦП (Электронной Цифровой Подписи) на основе RSA включает следующие шаги:

1. Генерация ключей:

* Сгенерируйте два случайных простых числа *p* и *q*.
* Вычислите модуль n как произведение *p* и *q*: *n* = *p* × *q*.
* Вычислите значение функции Эйлера φ(*n*) = (*p* - 1) × (*q* - 1).
* Выберите открытую экспоненту e, которая является взаимно простой с φ(*n*) и меньше φ(*n*).
* Вычислите закрытую экспоненту *d*, которая является мультипликативно обратной к e по модулю φ(*n*): *d* ≡ (mod φ(*n*)).
* Открытый ключ: (*n*, e)
* Закрытый ключ: (*n*, *d*)

1. Подписывание сообщения:

* Преобразуйте сообщение в хэш-значение с использованием хэш-функции, например, SHA-256.
* Зашифруйте хэш-значение с помощью закрытого ключа RSA: signature = (mod *n*).
* Полученное значение signature представляет собой ЭЦП для сообщения.

1. Проверка подписи:

* Получите открытый ключ RSA (*n*, *e*) отправителя.
* Расшифруйте ЭЦП с использованием открытого ключа RSA: decrypted\_hash = ×(mod *n*).
* Вычислите хэш-значение исходного сообщения с использованием той же хэш-функции.
* Сравните полученное хэш-значение с расшифрованным хэш-значением. Если они совпадают, подпись считается верной.

Ограничения:

При генерации ключей необходимо выбирать достаточно большие простые числа p и q для обеспечения безопасности RSA.

Размер модуля n должен быть достаточно большим, например, 2048 бит или более, для обеспечения защиты от атак по факторизации.

Выбор открытой экспоненты e должен быть осуществлен с учетом безопасности и стандартных рекомендаций, например, использование простых чисел или значения 65537.

Хэш-функция, используемая для преобразования сообщения в хэш-значение, должна быть стойкой к коллизиям и рекомендованной к использованию, например, SHA-256.

Программная реализация генерации ЭЦП представлена на рисунке 2.1.

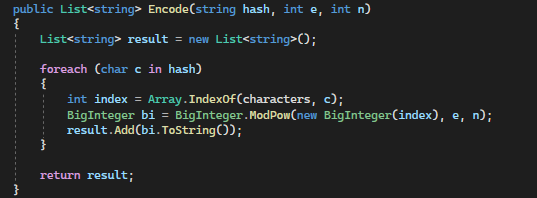


Рисунок 2.1 – Генерация ЭЦП RSA

Реализация генерации хеша на стороне получателя для дальнейшего сравнения с полученным хешем представлена на рисунке 2.2.

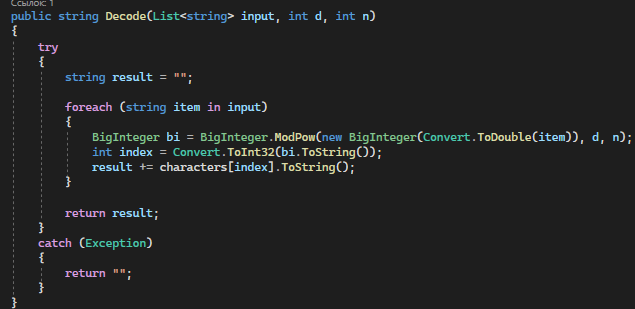


Рисунок 2.2 – Генерация хэша для проверки ЭЦП

Результат работы приложения представлен на рисунке 2.3.

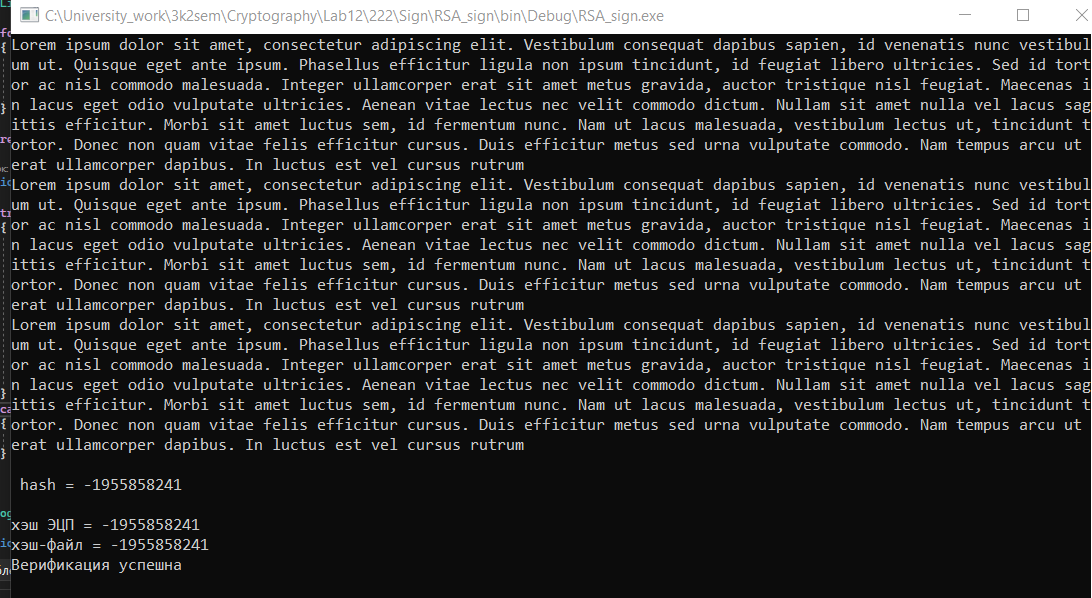


Рисунок 2.3 – Результат работы алгоритма RSA

## 2.2. Алгоритм Эль-Гамаля

Алгоритм ЭЦП (Электронной Цифровой Подписи) на основе схемы Эль-Гамаля включает следующие шаги:

1. Генерация ключей:

* Выберите простое число *p*.
* Выберите первообразный корень g по модулю *p*.
* Выберите случайное число x, такое что 1 <= *x* <*p* -1.
* Вычислите *y* = mod *p*.
* Открытый ключ: (*p*, g, *y*)
* Закрытый ключ: *x*

1. Подписывание сообщения:

* Преобразуйте сообщение в хэш-значение с использованием хэш-функции, например, SHA-256.
* Выберите случайное *k*, такое что 1 <= *k* < *p* -1 и НОД(*k*, *p* -1) = 1.
* Вычислите *r* = mod *p*.
* Вычислите *s* = (hash + *x* ×*r*) × mod (*p* -1).
* Полученная подпись: (*r*, *s*)

1. Проверка подписи:

* Получите открытый ключ Эль-Гамаля (*p*, *g*, *y*) отправителя.
* Вычислите значение хэш-значения сообщения.
* Вычислите *w* = mod (*p* -1).
* Вычислите *u*1 = (hash × *w*) mod (*p* -1) и *u*2 = (*r* × *w*) mod (*p* -1).
* Вычислите v = ( × mod *p*) mod *p*.
* Если *v* равно *r*, то подпись считается верной.

Ограничения:

Выбор простого числа *p* должен быть осуществлен с учетом безопасности и рекомендаций, например, использование достаточно больших простых чисел.

Размер числа *p* должен быть достаточно большим, чтобы обеспечить безопасность схемы.

Выбор случайных чисел *x* и *k* должен осуществляться с помощью надежного генератора случайных чисел.

Хэш-функция, используемая для преобразования сообщения в хэш-значение, должна быть стойкой к коллизиям и рекомендованной к использованию, например, SHA-256.

Важно сохранять закрытый ключ в тайне, так как он используется для создания подписи и его компрометация может привести к подделке подписей.

Функции представлены на рисунке 2.4.

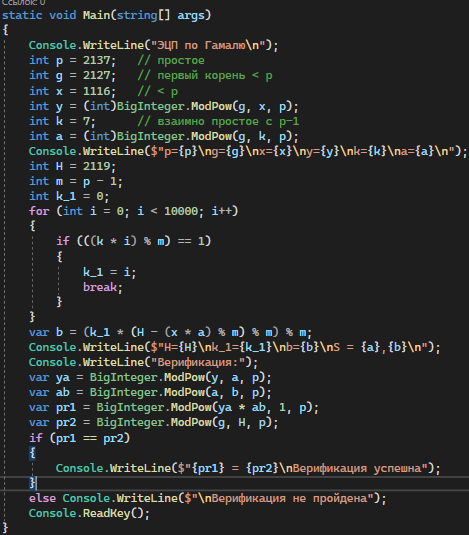


Рисунок 2.4 – Функции для работы алгоритма Эль-Гамаля

Результат ЭЦП Эль-Гамаля представлен на рисунке 2.5.

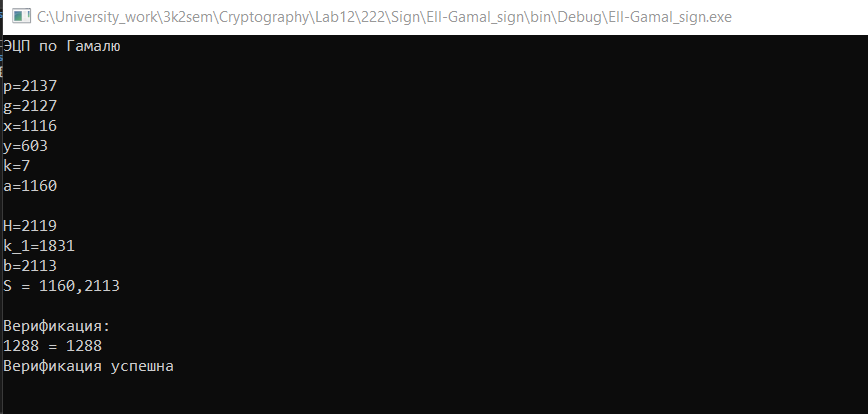


Рисунок 2.5 – Результат работы ЭЦП Эль-Гамаля

## 2.3. Алгоритм Шнорра

Алгоритм Шнорра для ЭЦП (Электронной Цифровой Подписи) является одним из популярных алгоритмов подписывания сообщений. Он включает следующие шаги:

1. Генерация ключей:

* Выберите простое число p большое и безопасное.
* Выберите целое число *q*, которое является простым делителем числа *p*-1.
* Выберите случайное целое число *g*, такое что mod *p* = 1.
* Выберите случайное целое число *x*, такое что 1 <= *x* < *q*.
* Вычислите *y* = mod *p*.
* Открытый ключ: (*p*, *q*, *g*, *y*)
* Закрытый ключ: *x*

1. Подписывание сообщения:

* Преобразуйте сообщение в хэш-значение с использованием хэш-функции, например, SHA-256.
* Выберите случайное целое число *k*, такое что 1 <= *k* < *q*.
* Вычислите *r* = ( mod *p*) mod *q*.
* Вычислите *e* = хэш-значение сообщения.
* Вычислите *s* = (*k* – *x* ×*r*) × mod *q*.
* Полученная подпись: (*r*, *s*)

1. Проверка подписи:

* Получите открытый ключ Шнорра (*p*, *q*, *g*, *y*) отправителя.
* Вычислите значение хэш-значения сообщения.
* Вычислите *v* = × mod *p* mod *q*.
* Если *v* равно *r*, то подпись считается верной.

Ограничения:

Выбор простых чисел *p* и *q* должен быть осуществлен с учетом безопасности и рекомендаций, например, использование достаточно больших простых чисел.

Размер чисел *p* и *q* должен быть достаточно большим, чтобы обеспечить безопасность схемы.

Выбор случайных чисел *x* и *k* должен осуществляться с помощью надежного генератора случайных чисел.

Хэш-функция, используемая для преобразования сообщения в хэш-значение, должна быть стойкой к коллизиям и рекомендованной к использованию, например, SHA-256.

Важно сохранять закрытый ключ в тайне, так как он используется для создания подписи и его компрометация может привести к подделке подписей.

Протестируем приложение, запустив его. Результат в консоли представлен на рисунке 2.6.

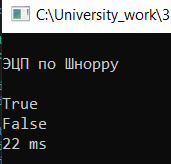


Рисунок 2.6 – Результат работы алгоритма Шнорра

Также на рисунке 2.7 представлен график оценки времени представленных процедур создания ЭЦП.

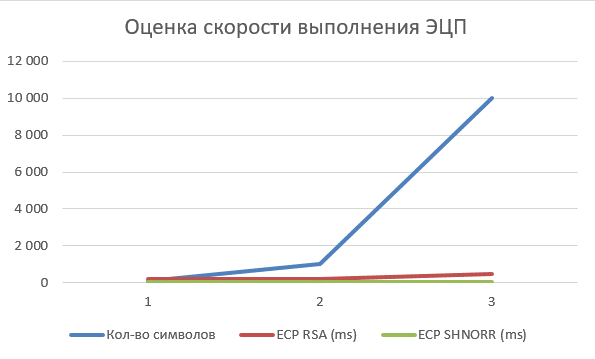


Рисунок 2.7 – Оценочный график выполнения всех процедур создания ЭЦП

## Вывод

В ходе лабораторной работы были изучены алгоритмы генерации и верификации электронной цифровой подписи, а также приобретены практические навыки их реализации.

Также была оценена скорость генерации и верификации ЭЦП.