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

Библиотека для постквантовой криптографии

АВТОРЕФЕРАТ
дипломной работы

студента 6 курса 631 группы
специальности 10.05.01 Компьютерная безопасность
факультета компьютерных наук и информационных технологий
Тарана Александра Владимировича

Научный руководитель
старший преподаватель _____ А. А. Лобов
22.01.2024 г.

Заведующий кафедрой
д. ф.-м. н., доцент _____ М. Б. Абросимов
22.01.2024 г.

Саратов 2024

ВВЕДЕНИЕ

Актуальность постквантовой криптографии в настоящее время стала неоспоримой среди специалистов в области информационной безопасности. Традиционные криптографические алгоритмы и протоколы, ранее считавшиеся надежными, теряют свою актуальность в связи с развитием квантовых вычислений. Появление квантовых компьютеров, способных решать сложные математические задачи значительно быстрее, ставит под угрозу существующие методы шифрования, основанные на сложности факторизации или поиске дискретного логарифма.

В свете таких изменений появляется неотложная задача поиска эффективных и безопасных способов обеспечения конфиденциальности и целостности информации в условиях, когда классическая криптография окажется уязвимой перед атаками, совершаемыми с помощью квантовых компьютеров. Именно поэтому активно начинает развиваться и привлекать внимание исследователей постквантовая криптография.

Целью данной работы является изучение разработок в области постквантовой криптографии, а задачей – реализация программной библиотеки для написания программного обеспечения в этой области.

Дипломная работа состоит из введения, 5 разделов, заключения, списка использованных источников и 4 приложений. Общий объем работы – 71 страница, из них 43 страницы – основное содержание, в том числе 23 рисунка, список использованных источников из 12 наименований.

КРАТКОЕ СОДЕРЖАНИЕ

1 Общие сведения

Постквантовая криптография – раздел криптографии, остающийся актуальным после появления квантовых компьютеров и, как следствие, атак, проводимых с их помощью. Квантовый компьютер – вычислительное устройство, которое использует явления квантовой механики для передачи и обработки данных.

Большинство традиционных криптосистем опирается на проблемы факторизации целых чисел или задачи дискретного логарифмирования, которые могут быть решены за приемлемое время на достаточно больших квантовых компьютерах, использующих алгоритм Шора. Существуют криптосистемы, которые опираются на вычислительно сложные задачи и имеют ряд существенных отличий от указанных выше систем, из-за чего их гораздо сложнее решить. Эти системы (по крайней мере, на данный момент) не имеют алгоритмов решения на квантовых компьютерах, и, следовательно, их считают квантово-устойчивыми или «постквантовыми» криптосистемами.

2 Основные подходы на данный момент

На данный момент существует 5 основных принципов, на которых основано построение постквантовых криптосистем:

1. Криптография, основанная на хэш-функциях;
2. Криптография на основе кодов исправления ошибок;
3. Криптография на основе изогений;
4. Криптография, основанная на решетках;
5. Криптография в многомерных квадратичных системах.

Далее каждая из них будет рассмотрена подробнее.

2.1 Криптография, основанная на хэш-функциях

Безопасность криптосистем, основанных на хэш-функциях опирается на свойством необратимости: для хэш-функции H и её заданного значения m

должно быть вычислительно неосуществимо найти блок данных X, для которого $H(X) = m$.

2.2 Криптография на основе кодов исправления ошибок

В криптографии, основанной на кодах исправления ошибок, отправитель сообщения целенаправленно вводит дефекты в кодовое слово, чтобы затруднить декодирование, а затем и расшифровку. Получатель сообщения может расшифровать его, используя некоторые секретные знания (часто касающиеся структуры кода), но злоумышленник, не имеющий доступа к секретным знаниям, не может этого сделать.

2.3 Криптография на основе изогений

Эти криптографические системы полагаются на свойства графов изогении эллиптических кривых (и абелевых многообразий более высокой размерности) над конечными полями, в частности, суперсингулярных графов изогении, для создания криптографических систем.

2.4 Криптография, основанная на решетках

Криптография на основе решеток – это общий термин для конструкций криптографических примитивов, в которых используются решетки либо в самой конструкции, либо в доказательстве безопасности.

2.5 Криптография в многомерных квадратичных системах

Сюда входят криптографические системы, которые основана на сложности решения систем уравнений, основанных на многомерных полиномах над конечным полем. Безопасность многомерной криптографии основывается на предположении, что решения системы квадратичных многочленов над конечным полем, в общем случае, является NP-полной задачей в сильном смысле или просто NP-полной.

3 Победители конкурса NIST

В 2022 году NIST (National Institute of Standards and Technology - национальный институт стандартов и технологий) опубликовал алгоритмы победителей проводимого в несколько этапов конкурса по постквантовой криптографии.

Группа ученых разработала на основе решеток проект CRYSTALS, включающий в себя алгоритм распределения ключей и ассиметричного шифрования CRYSTALS-KYBER и алгоритм ЭЦП CRYSTALS-DILITHIUM.

Помимо этого, были представлены еще два алгоритма создания ЭЦП:

- FALCON – так же основан на решетках NTRU;
- SPHINCS+ – основанный на хэш-кодах.

4 Алгебра кватернионов и ее приложения

Отечественными учеными была предложена новая форма задания скрытой задачи дискретного логарифмирования и новая постквантовая схема ЭЦП. В качестве перспективного алгебраического носителя для предложенной вычислительно трудной задачи и схемы ЭЦП рассматривается конечная алгебра кватернионов, представленная в модифицированном виде, и исследуются ее свойства, применяемые для вычисления параметров схемы ЭЦП. Далее следуют необходимые определения.

4.1 Структура алгебры

Алгоритм работает с четырехмерными векторами над полем $GF(p)$, где p – простое число. Обозначим базисные вектора за $\bar{e}, \bar{\imath}, \bar{\jmath}, \bar{k}$, тогда любой вектор можно записать в виде $\bar{v} = a\bar{e} + b\bar{\imath} + c\bar{\jmath} + d\bar{k}$. Операция сложения, вычитания и умножения векторов на скаляры выполняются покоординатно:

$$\alpha\bar{v} \pm \beta\bar{w} = (\alpha a_1 \pm \beta a_2)\bar{e} + (\alpha b_1 \pm \beta b_2)\bar{\imath} += (\alpha c_1 \pm \beta c_2)\bar{\jmath} + (\alpha d_1 \pm \beta d_2)\bar{k} \quad (1)$$

Операция умножения вектора на вектор выполняется более сложным образом:

1. Каждый вектор представляется в виде линейной комбинации базисных векторов, т.е.

$$\bar{v} \circ \bar{w} = \left(a_1\bar{e} + b_1\bar{\imath} + c_1\bar{\jmath} + d_1\bar{k}\right) \circ \left(a_2\bar{e} + b_2\bar{\imath} + c_2\bar{\jmath} + d_2\bar{k}\right) \quad (2)$$

2. Перемножить полученные суммы и привести подобные слагаемые:

$$\bar{v} \circ \bar{w} = a_1a_2(\bar{e} \circ \bar{e}) + a_1b_2(\bar{e} \circ \bar{\imath}) + \cdots + d_1d_2\left(\bar{k} \circ \bar{k}\right); \quad (3)$$

3. Базисные вектора перемножаются согласно следующей таблице:

Таблица 1 – Таблица умножения базисных векторов (ТУББВ)

	$\bar{e}$	$\bar{\imath}$	$\bar{\jmath}$	$\bar{k}$
$\bar{e}$	$\bar{e}$	$\bar{\imath}$	$\bar{\jmath}$	$\bar{k}$
$\bar{\imath}$	$\bar{\imath}$	$-\varepsilon\bar{e}$	$\varepsilon\bar{k}$	$-\bar{\jmath}$
$\bar{\jmath}$	$\bar{\jmath}$	$-\varepsilon\bar{k}$	$-\varepsilon\bar{e}$	$\bar{\imath}$
$\bar{k}$	$\bar{k}$	$\bar{\jmath}$	$-\bar{\imath}$	$-\bar{e}$

Результатом умножения базисных векторов будет ячейка таблицы в соответствии строка-столбец, где ε – структурная константа алгорима.

4.2 Алгоритм подписи

Для построения схемы ЭЦП в скрытой конечной циклической группе примем в качестве аналога алгоритм ЭЦП Шнорра.

Процедура генерации ключей состоит из следующих шагов:

1. Сгенерировать простое p достаточно размера (от 512 бит);

2. Сгенерировать случайный необратимый элемент $\bar{g}$ большого (не менее 256 бит) простого порядка $\bar{q}$;

3. Сгенерировать случайные обратимые элементы $\bar{d}, \bar{u}$: $\bar{g} \circ \bar{d} \neq \bar{d} \circ \bar{g}, \bar{g} \circ \bar{u} \neq \bar{u} \circ \bar{g}, \bar{u} \circ \bar{d} \neq \bar{d} \circ \bar{u}$;

4. Сгенерировать три случайных числа w, t, x;

5. Выбрать случайную правую локальную единицу $\overline{e_g}$ для вектора $\bar{g}$;

6. Вычислить так называемый элемент согласования $\bar{l} = \bar{d}^{-w} \circ \overline{e_g} \circ \bar{u}^t$;

7. Вычислить открытый ключ $\bar{y} = \bar{d}^{-w} \circ \bar{g}^x \circ \bar{d}^w, \bar{z} = u^{-t} \circ \bar{g} \circ \bar{u}^t$.

Таким образом, перед злоумышленником для вскрытия системы стоит задача найти x при заданных $\bar{y}, \bar{z}, \bar{l}$ и неизвестных $\bar{d}, \bar{u}, \bar{g}, \overline{e_g}, w, t$.

Далее процедура подписания сообщения $\bar{m}$ (в виде вектора) состоит из:

1. Сгенерировать случайное число k;

2. Вычислить разовый открытый ключ $\bar{r} = \bar{d}^{-w} \circ \bar{g}^{k} \circ \bar{u}^{t}$;

3. Вычислить первую часть подписи $h = F_H(\bar{m}||\bar{r})$;

4. Вычислить вторую часть подписи $s = k + ex(mod\ q)$.

Итоговая подпись состоит из двух чисел – (h, s). Алгоритм проверки состоит всего из двух шагов:

1. Вычислить $\bar{r}' = \bar{y}^{q-e} \circ \bar{l} \circ \bar{z}^{s}$;

2. Вычислить $h' = F_H(\bar{m}||\bar{r}')$.

Результатом проверки будет сравнение $h = h'$.

4.3 Алгоритм распределения ключей

Схема открытого распределения ключей на основе алгебры кватернионов строится следующим образом. Пусть Алиса и Боб хотят провести протокол обмена ключами. Задается конечная некоммутативная группа четырехмерных векторов над простым полем $GF(p)$. Генерируются два некоммутирующих вектора Q и G, каждый из которых имеет порядок, равный q (т. е. $Q \circ G \neq G \circ Q$). Открытый ключ в общем виде задается по формуле:

$$Y = G^{w} \circ Q^{x} \circ G^{-w}; \quad (4)$$

Далее каждый из двух участников обмена ключами генерирует свой закрытый ключ – (x_a, w_a) для Алисы и (x_b, w_b) для Боба. Каждый из них генерирует и передает оппоненту свой открытый ключ, соответственно: $Y_a = G^{w_a} \circ Q^{x_a} \circ G^{-w_a}$ и $Y_b = G^{w_b} \circ Q^{x_b} \circ G^{-w_b}$. Согласование общего ключа происходит следующим образом.

1. Алиса вычисляет $K_a = G^{w_a} \circ Y_b^{x_a} \circ G^{-w_a} = G^{w_a} \circ (G^{w_b} \circ Q^{x_b} \circ G^{-w_b})^{x_a} \circ G^{-w_a} = G^{w_a + w_b} \circ Q^{x_a x_b} \circ G^{-w_a - w_b}$;

2. Боб вычисляет $K_b = G^{w_b} \circ Y_a^{x_b} \circ G^{-w_b} = G^{w_b} \circ (G^{w_a} \circ Q^{x_a} \circ G^{-w_a})^{x_b} \circ G^{-w_b} = G^{w_a+w_b} \circ Q^{x_a x_b} \circ G^{-w_a-w_b}$.

5 Реализация библиотеки для работы с кватернионами

5.1 Описание используемых компонентов

В практической части данной работы представлена библиотека на языке программирования Java 19-ой версии. Использована система сборки Maven.

Далее представлены конфигурации внешних библиотек из файла описания сборки проекта – pom.xml. Для сериализации и десереализации данных использована библиотека Jaskson, для сокращения исходного кода использована библиотека Lombok, для логирования использована библиотека API Slf4J и конкретная реализация – Log4J.

5.2 Обзор кода проекта

Так как для сборки используется Maven, то реализуемая библиотека выполнена по классической иерархии для проектов с такой системой. Скомпилированные в байт-код классы, как и финальный jar-архив лежат в каталоге target. с данными объектами и доступный для расширения при необходимости.

Для более удобного восприятия форматом выходных файлов выбран JSON.

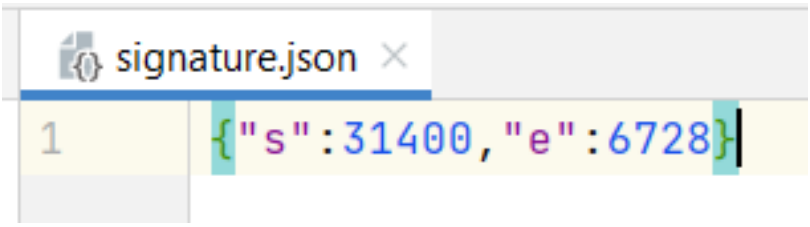

Рисунок 1 - Пример файла подписи

Далее будет рассмотрена часть, реализующая алгоритм распределения ключей, описанный в пункте 4.3 данной работы.

Так как шаги симметричны для обеих сторон, то методы создания закрытого и открытого ключей можно вызывать, передав нужные значения, полученные с другой стороны.

Помимо отдельных методов-шагов была реализована утилита для консоли, позволяющая провести все шаги обмена ключами.

5.3 Примеры работы программы подписи

Зададим размер p 16 бит, структурную константу как 1. Для начала необходимо сгенерировать подписи, для этого нужно подать на вход пути к выходным файлам открытого и закрытого ключей и указать нужный режим работы.

```
C:\Users\taran\IdeaProjects\quaternions>C:\Users\taran\.jdks\corretto-19.0.2\bin\java.exe
 -jar target/quaternions-1.0-SNAPSHOT.jar -m create-keys --pSize 10 --epsilon 1 --publicK
eyPath C:\Users\taran\IdeaProjects\quaternions\public.json --privateKeyPath C:\Users\tara
n\IdeaProjects\quaternions\private.json
2023-12-11 23:43:49 INFO  SignatureImpl:191 - Данные были записаны в файл C:\Users\taran\
IdeaProjects\quaternions\public.json
2023-12-11 23:43:49 INFO  SignatureImpl:191 - Данные были записаны в файл C:\Users\taran\
IdeaProjects\quaternions\private.json
2023-12-11 23:43:49 INFO  SignatureImpl:83 - Ключи сохранены в соответствующие файлы
2023-12-11 23:43:49 INFO  SignatureImpl:87 - Время выполнения 100 миллисекунд

C:\Users\taran\IdeaProjects\quaternions>
```

Рисунок 2 - Создание ключей

```
C:\Users\taran\IdeaProjects\quaternions>C:\Users\taran\.jdks\corretto-19.0.2\bin\java.exe
 -jar target/quaternions-1.0-SNAPSHOT.jar -m make-signature --filePath C:\Users\taran\Ide
aProjects\quaternions\file.txt --privateKeyPath C:\Users\taran\IdeaProjects\quaternions\p
rivate.json --publicKeyPath C:\Users\taran\IdeaProjects\quaternions\public.json --signatu
rePath C:\Users\taran\IdeaProjects\quaternions\signature.json
2023-12-11 23:45:18 INFO  SignatureImpl:191 - Данные были записаны в файл C:\Users\taran\
IdeaProjects\quaternions\signature.json
2023-12-11 23:45:18 INFO  SignatureImpl:141 - Время выполнения 127 миллисекунд

C:\Users\taran\IdeaProjects\quaternions>
```

Рисунок 3 - Подписание файла

На последнем этапе, проверке подписи, на вход подаются пути к файлу, подпись которого мы проверяем, к открытому ключу, и, соответственно, самому файлу подписи.

```
C:\Users\taran\IdeaProjects\quaternions>C:\Users\taran\.jdks\corretto-19.0.2\bin\java.exe
 -jar target/quaternions-1.0-SNAPSHOT.jar -m check-signature --filePath C:\Users\taran\Id
eaProjects\quaternions\file.txt --publicKeyPath C:\Users\taran\IdeaProjects\quaternions\p
ublic.json --signaturePath C:\Users\taran\IdeaProjects\quaternions\signature.json
2023-12-11 23:46:43 INFO  SignatureImpl:172 - Подпись верна
2023-12-11 23:46:43 INFO  SignatureImpl:180 - Время выполнения 92 миллисекунд

C:\Users\taran\IdeaProjects\quaternions>
```

Рисунок 4 - Проверка подписи

Если отредактировать файл после подписания, то результат будет противоположный:

```
C:\Users\taran\IdeaProjects\quaternions>C:\Users\taran\.jdks\corretto-19.0.2\bin\java.exe
 -jar target/quaternions-1.0-SNAPSHOT.jar -m check-signature --filePath C:\Users\taran\Id
eaProjects\quaternions\file.txt --publicKeyPath C:\Users\taran\IdeaProjects\quaternions\p
ublic.json --signaturePath C:\Users\taran\IdeaProjects\quaternions\signature.json
2023-12-11 23:46:43 INFO  SignatureImpl:172 - Подпись неверна
2023-12-11 23:46:43 INFO  SignatureImpl:180 - Время выполнения 97 миллисекунд

C:\Users\taran\IdeaProjects\quaternions>
```

Рисунок 5 - Проверка подписи после редактирования

5.4 Пример работы программы обмена ключами

Согласно алгоритму, необходимо сначала сгенерировать параметры рабочей группы:

```
C:\Users\taran\IdeaProjects\quaternions>C:\Users\taran\.jdks\corre
tto-19.0.2\bin\java.exe -jar target/quaternions-1.0-SNAPSHOT.jar -
m generate-private --paramsFile C:\Users\taran\IdeaProjects\quater
nions\parameters.json --privateKeyPath C:\Users\taran\IdeaProjects
\quaternions\private.json
2024-01-12 22:01:24 INFO  SignatureMain:178 - Данные были записаны
 в файл C:\Users\taran\IdeaProjects\quaternions\private.json

C:\Users\taran\IdeaProjects\quaternions>
```

parameters.json

```
{
  "q":{"p":839,"structureConstant":1,"e":697,"i":476,"j":331,"k":124},
  "g":{"p":839,"structureConstant":1,"e":619,"i":124,"j":705,"k":625}
}
```

Рисунок 6 - Генерация параметров группы и результат

Далее каждый из участников (как и в теоретическом описании алгоритма, обозначим их за Боба и Алису) генерирует свои закрытые ключи:

```
C:\Users\taran\IdeaProjects\quaternions>C:\Users\taran\.jdks\corre
tto-19.0.2\bin\java.exe -jar target/quaternions-1.0-SNAPSHOT.jar -
m generate-private --paramsFile C:\Users\taran\IdeaProjects\quater
nions\parameters.json --privateKeyPath C:\Users\taran\IdeaProjects
\quaternions\private_alice.json
2024-01-12 22:01:07 INFO  SignatureMain:178 - Данные были записаны
 в файл C:\Users\taran\IdeaProjects\quaternions\private_alice.json


C:\Users\taran\IdeaProjects\quaternions>
```

private_alice.json

```
{"w":26490,"x":18632}
```

private_bob.json

```
{"w":43437,"x":10163}
```

Рисунок 7 - Генерация закрытых ключей и результаты

Следующим шагом следует генерация открытых ключей, Алисой и Бобом соответственно:

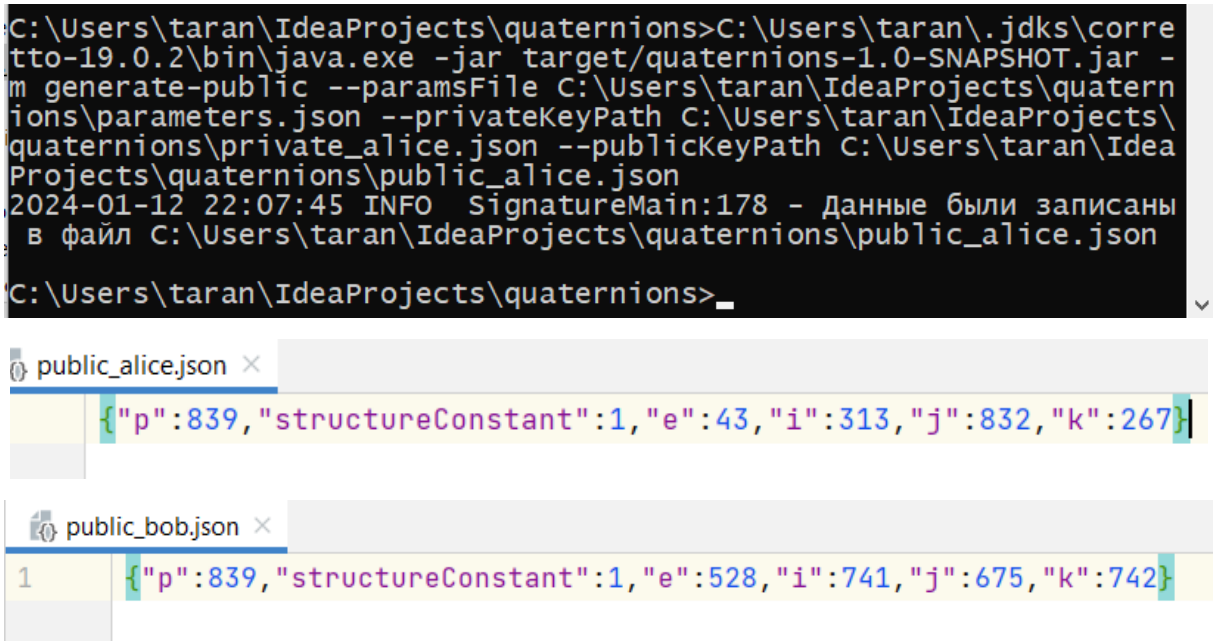

Рисунок 8 - Создание открытых ключей и результаты

Последний шаг алгоритма – создание общих ключей обоими оппонентами и проверка, что они получились одинаковые. Соответственно входными параметрами будут собственный закрытый ключ и чужой открытый.

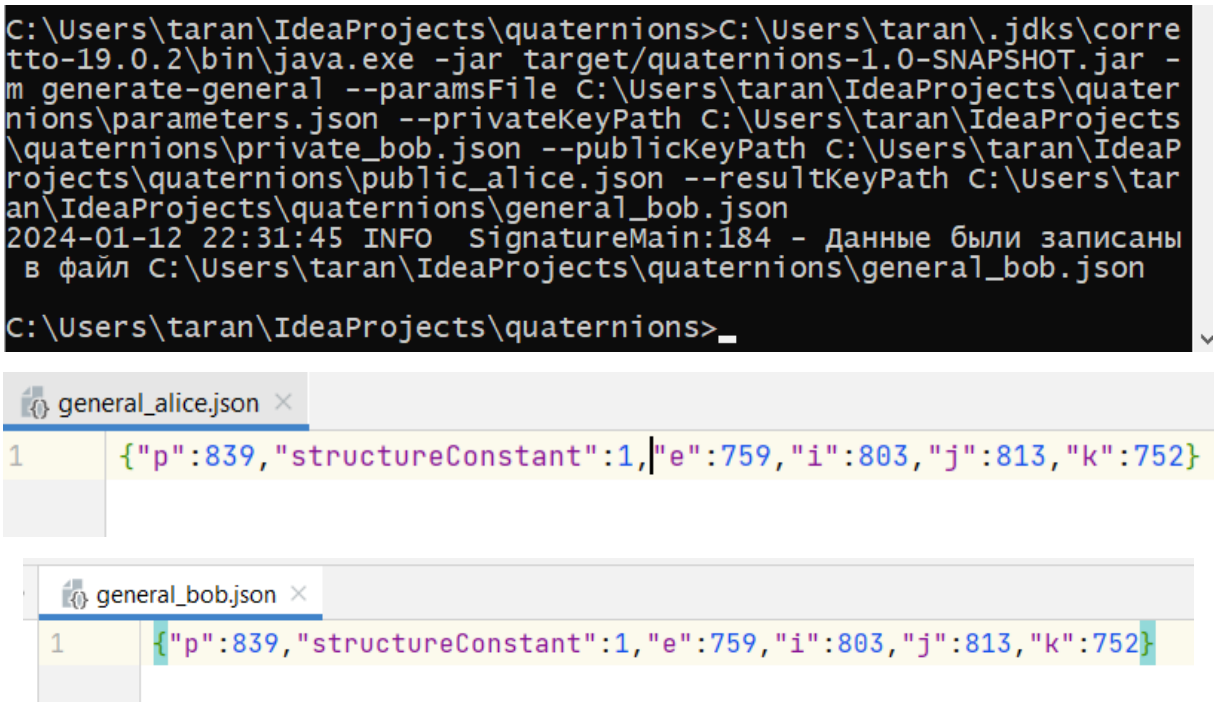

Рисунок 9 - Создание общего ключа и результат

ЗАКЛЮЧЕНИЕ

В данной работе были рассмотрены теоретические и практические вопросы, связанные с методами и алгоритмами постквантовой защиты информации, а именно алгебра кватернионов и программная реализация библиотеки, описывающей эту алгебру.

Также была рассмотрена схема цифровой подписи на кватернионах, доказана ее теоретическая работоспособность и продемонстрировано использование на практике: реализована библиотека для работы с кватернионами, созданием подписи и проведение алгоритма распределения ключей на их основе и утилита для использования этой библиотеки. Таким образом, цели и задачи данной работы можно считать выполненными.