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

**Протоколы передачи секретного ключа по открытому каналу**

ОТЧЁТ

ПО ДИСЦИПЛИНЕ

«КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ»

студента 5 курса 531 группы

специальности 10.05.01 Компьютерная безопасность

факультета компьютерных наук и информационных технологий

Гельфанова Даниила Руслановича

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

Саратов 2023

**1 Постановка задачи**

Необходимо реализовать протоколы передачи секретного ключа по открытому каналу Encrypted Key Exchange (EKE) на базе алгоритма Эль-Гамаля.

**2 Теоретические сведения**

Безопасность схемы Эль-Гамаля основана на трудоемкости вычисления дискретных логарифмов в конечном поле. Для генерации пары ключе сначала выбирается простое число и 2 случайных числа, и , где – первообразный корень по модулю , а . Затем вычисляется . Открытым ключом являются . И , и можно сделать общими для группы пользователей. Закрытым ключом является .

Протокол обмена зашифрованными ключами (Encrypted Key Exchange, EKE) был разработан Стивом Белловином и Майклом Мерритом. Он обеспечивает безопасность и проверку подлинности в компьютерных сетях, новым образом используя и симметричную криптографию, и криптографию с открытым ключом: общий секретный ключ используется для шифрования генерированного случайным образом открытого ключа. В протоколе Алиса и Боб имеют общий пароль .

Алгоритм работы протокола EKE на базе Эль-Гамаля.

Вход: простое число и первообразный корень .

Выход: установление сеансового ключа .

Шаг 1. Алиса выбирает свой закрытый ключ и считает открытый ключ . После чего посылает Бобу незашифрованное сообщение:

Шаг 2. Боб генерирует случайный сеансовый ключ .Затем он выбирает случайное число (закрытый ключ) (для алгоритма Эль-Гамаля, независимо от других случайных чисел, выбираемых для ), и сообщение, которое он посылает Алисе, выглядит так:

Шаг 3. Алиса расшифровывает сообщение и получает ключ как . Она генерирует случайную строку , шифрует с помощью ключа и посылает результат:

Шаг 4. Боб расшифровывает сообщение, получая . Он генерирует другую случайную строку, , шифрует обе строки ключом и посылает Алисе результат:

Шаг 5. Алиса расшифровывает сообщение, получая . Если строка , полученная от Боба, это та самая строка, которую она послала Бобу на шаге 3, она, используя , шифрует строку и посылает Бобу:

Шаг 6. Боб расшифровывает сообщение, получая . Если строка , полученная от Алисы, это та самая строка, которую он послал ей на шаге 4, то протокол завершен. Теперь обе стороны могут обмениваться информацией, используя в качестве сеансового ключа.

Ева, сидя между Алисой и Бобом, знает только и несколько сообщений, зашифрованных ключом . В других протоколах Ева могла бы попробовать угадать и затем проверить свои предположения. В рассматриваемом протоколе Ева не может проверять свои предположения, не вскрыв при этого и алгоритм с открытым ключом ( и ). Тогда, если и выбираются случайным образом, то эта проблема будет непреодолимой.

Ответная часть протокола, соответствующая шагам 3-6, обеспечивает подтверждение. Шаги 3-5 доказывают Алисе, что Боб знает ключ, этапы 4-6 доказывают Бобу, что Алиса знает ключ. В протоколе для решения этой же задачи используется обмен метками времени.

**3 Результаты работы**

**3.1 Сведения о программе**

Программа была реализована на языке программирования Java. В ней есть 4 класса: , , и .

В классе происходит инициализация входного параметра – длины простого числа , который будет сгенерирован для схемы Эль-Гамаля. Также этот параметр будет являться ограничением для длины сеансового ключа.

Класс – класс участника протокола. Для инициализации объекта указывается имя участника, простое число , первообразный корень и объект класcа . В классе реализованы следующие методы:

- public void generateSessionKey(BigInteger p) – генерация сеансового ключа;

- public void setSessionKey(BigInteger sessionKeyInt) – установка сеансового ключа.

- private void setRandomPrivateKey() – установка закрытого ключа для схемы Эль-Гамаля.

- public String getEncrypted(String info, SecretKey key) – шифрование данных по ключу.

- public String getDecrypted(String encInfo, SecretKey key) – расшифрование данных по ключу.

- public void generateString() – генерация случайной строки.

Класс – класс с описанием методом для шифрования и расшифрования. При инициализации объекта данного класса генерируется общий ключ и инициализационный вектор. В классе описаны следующие методы:

* public static SecretKey generateKey(int keyLength) – генерация ключа длины в шифровании .
* public static IvParameterSpec generateIv() – генерация инициализационного вектора.
* public static String encrypt(String algorithm, String input, SecretKey key, IvParameterSpec iv) – шифрование строки шифром .
* public static String decrypt(String algorithm, String cipherText, SecretKey key, IvParameterSpec iv) – расшифрование строки шифром .
* public static String convertSecretKeyToString(SecretKey secretKey) – конвертации ключа в строку.
* public static SecretKey convertStringToSecretKeyto(String encodedKey) – конвертация строки в ключ.

Класс – реализация работы протокола. Для инициализации объекта передается простое число . При инициализации случайным образом вычисляется первообразный корень и создаются объект класса и 2 участника протокола – Алиса и Боб. После чего происходит запуск работы протокола. Каждому шагу алгоритма выше соответствует собственный метод. Помимо этих методов в классе описаны следующие методы:

* public static BigInteger generatePrimitiveRoot(BigInteger n) – генерация примитива по модулю . В данном методы выбирается случайное число до . И начиная от него в цикле происходит поиск примитивного корня.
* public static boolean isPrimitive(BigInteger prime, BigInteger n, ArrayList<BigInteger> orders) – проверка, является число примитивом по утверждению, описанному в пункте 2.
* public static ArrayList<BigInteger> getDivisors(BigInteger num) – метод для получения списка делителей числа.

**3.2 Тестирование программы**

Для реализации модульного тестирования были написаны тестовые классы и .

В классе содержится 2 метода для тестирования шифрования, расшифрования данных, а также конвертации ключа в строку и обратно. Результат отработки теста представлен на рисунке 1.

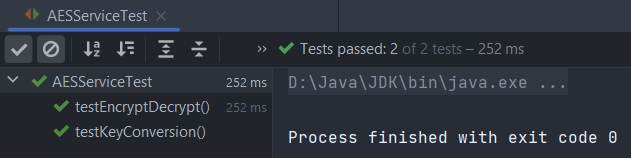


Рисунок 1 – Запуск

В классе содержится 2 тестовых метода. Первый – для проверки вызова исключения , если для создания объекта передано составное число. Второй – проверка работы протокола , где смотрится, что после работы протокола у двоих пользователей совпадает сеансовый ключ. Результат отработки тестов представлен на рисунке 2.

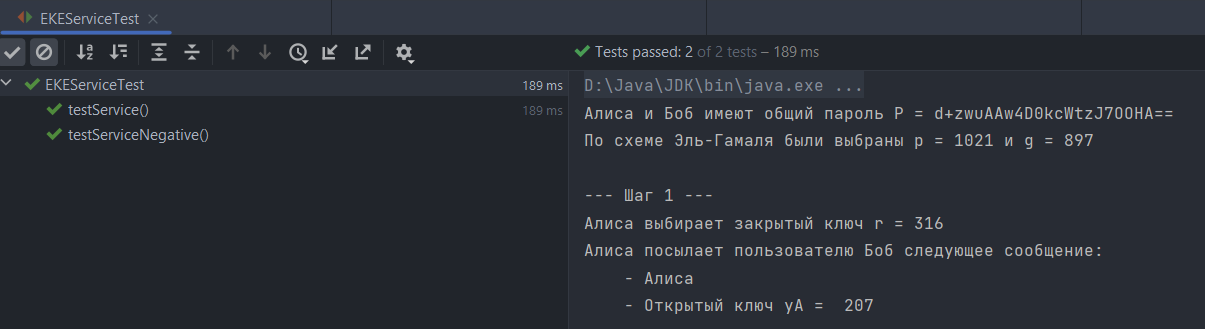


Рисунок 2 – Запуск

На рисунках 3-7 представлено тестирование программы.



Рисунок 3 – Негативное тестирование (входной параметр не указан)

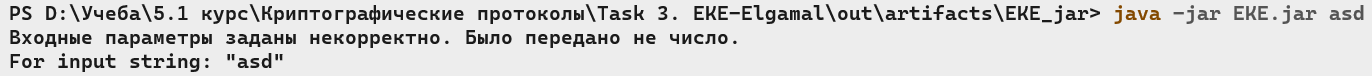


Рисунок 4 – Негативное тестирование (входной параметр не является числом)



Рисунок 5 – Положительное тестирование (входной параметр 25), шаги 1-3

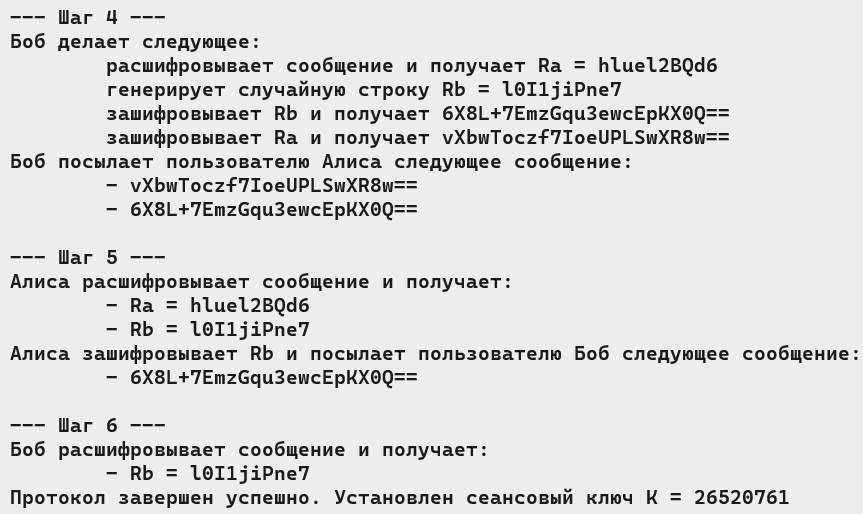


Рисунок 6 – – Положительное тестирование (входной параметр 25), шаги 4-6

**ПРИЛОЖЕНИЕ А**

**Листинг программы**