УО «Белорусский государственный университет информатики и радиоэлектроники»
Кафедра ПОИТ

Отчет по лабораторной работе №3
по предмету
Теория Информации

Вариант “Эллиптические кривые”

Выполнил:
Матюшенко А.Д.

Проверил:
Болтак С.В.

Группа 351001

Минск 2025

Задание:

1. Для заданного M (ввести с кл) определить значения a и b, которые позволяют построить эллиптическую группу EM(a, b).
2. Для найденных в задании 1 параметров сгенерировать все элементы эллиптической группы EM(a, b).
3. Реализовать алгоритм обмена ключами для эллиптической группы EM(a,b).
4. Разработать алгоритм цифровой подписи на основе эллиптической груп пы EM(a, b).

   Алгоритм пунктов 1-3 реализован отдельно в виде консольного приложения для более удобной демонстрации.
   Алгоритм цифровой подписи с использование стандартной кривой с известными параметрами реализован в виде учебной веб-страницы.

1. Теоретическая справка

Алгоритмы, основанные на эллиптических кривых, используются в криптографии для обеспечения безопасной передачи данных. Основной объект — эллиптическая кривая над конечным полем, которая задаётся уравнением: $y^2 = x^3 + ax + b \pmod p$. Условием её корректности является невырожденность: $4a^3 + 27b^2 \neq 0 \pmod p$. Элементы группы — это пары (x, y), удовлетворяющие уравнению, а также специальная точка на бесконечности.

Обмен ключами осуществляется с помощью алгоритма Диффи-Хеллмана в группе точек эллиптической кривой, называемый ECDH. Безопасность алгоритма обеспечивается сложностью задачи дискретного логарифмирования.

2. Поиск параметров эллиптической кривой

Для заданного простого числа M подбираются коэффициенты a и b, такие что $4a^3 + 27b^2 \neq 0 \pmod M$.

3. Построение группы точек EM(a, b)

После выбора коэффициентов a и b, вычисляются все точки (x, y), удовлетворяющие уравнению кривой по модулю M. Также включается точка на бесконечности.

4. Обмен ключами на эллиптических кривых (ECDH)

Алгоритм Диффи-Хеллмана с использованием эллиптической группы.
Используются:
1. Общая для всех эллиптическая кривая.
2. Общая базовая точка G (генератор) на этой кривой.

Процесс:
1. Alice:
   - Выбирает случайный секретный (приватный) ключ 'a' (целое число).
   - Вычисляет свой публичный ключ A = a*G (умножение точки G на скаляр 'a').
   - Отправляет A Бобу.
2. Bob:
   - Выбирает случайный секретный (приватный) ключ 'b' (целое число).
   - Вычисляет свой публичный ключ B = b*G.
   - Отправляет B Алисе.
3. Вычисление общего секрета:
   - Alice вычисляет: S_A = a*B = a*(b*G) = (a*b)*G.
   - Bob вычисляет: S_B = b*A = b*(a*G) = (b*a)*G.

Результат:
S_A и S_B равны (S_A = S_B = S). Это и есть общий секретный ключ, который могут использовать Alice и Bob.
Значение S (или его производная) используется для симметричного шифрования.

Безопасность основана на сложности вычисления дискретного логарифма в группе точек эллиптической кривой:
зная G и A=a*G (или G и B=b*G), вычислить 'a' (или 'b') практически невозможно для правильно выбранных параметров кривой

5. Алгоритм цифровой подписи

Реализация алгоритма цифровой подписи производилась в виде отдельного приложения. Использовалась кривая secp256k1.

Для эцп используются:
1. Общая для всех эллиптическая кривая:

```
export const curveConfig :{...} = {  Show usages
    a: new BN(0),
    b: new BN(7),
    p: new BN(
        "115792089237316195423570985008687907853269984665640564039457584007908834671663"
    )
};
```

Параметры кривой secp256k1

2. Общая базовая точка G (генератор) на этой кривой:

```
export const generatorPoint : {orderN: any, x: any, y: any} = {  Show usages
    x: new BN(
        "55066263022277343669578718895168534326250603453777594175500187360389116729240"
    ),
    y: new BN(
        "326705100207588169780830851305070431844712733806592432759389043357573374824 24"
    ),
    orderN: new BN(
        "1157920892373161954235709850086879078528375642790749043826051631415181614943 37"
    )
};
```

Параметры базовой точки G

Генерация подписи (signMessage)
2.1. Входные данные:

- сообщение, представляемое как целое e = SHA-256(message)
- приватный ключ d ∈ [1, n–1], где n — порядок генератора G

2.2. Шаги алгоритма:

1. Выбор случайного k ∈ [1, n–1].
2. Вычисление точки R = k·G методом «двойного и сложения».
3. Вычисление r = R.x mod n.
   Если r = 0, вернуть к шагу 1 и выбрать новое k.
4. Нахождение обратного элемента k^{-1} mod n (расширенный алгоритм Евклида).
5. Вычисление $s = k^{-1}\cdot(e + r\cdot d) \bmod n$.
6. Подпись представляется парами (r, s).

Проверка подписи (verifySignature)
3.1. Входные данные:

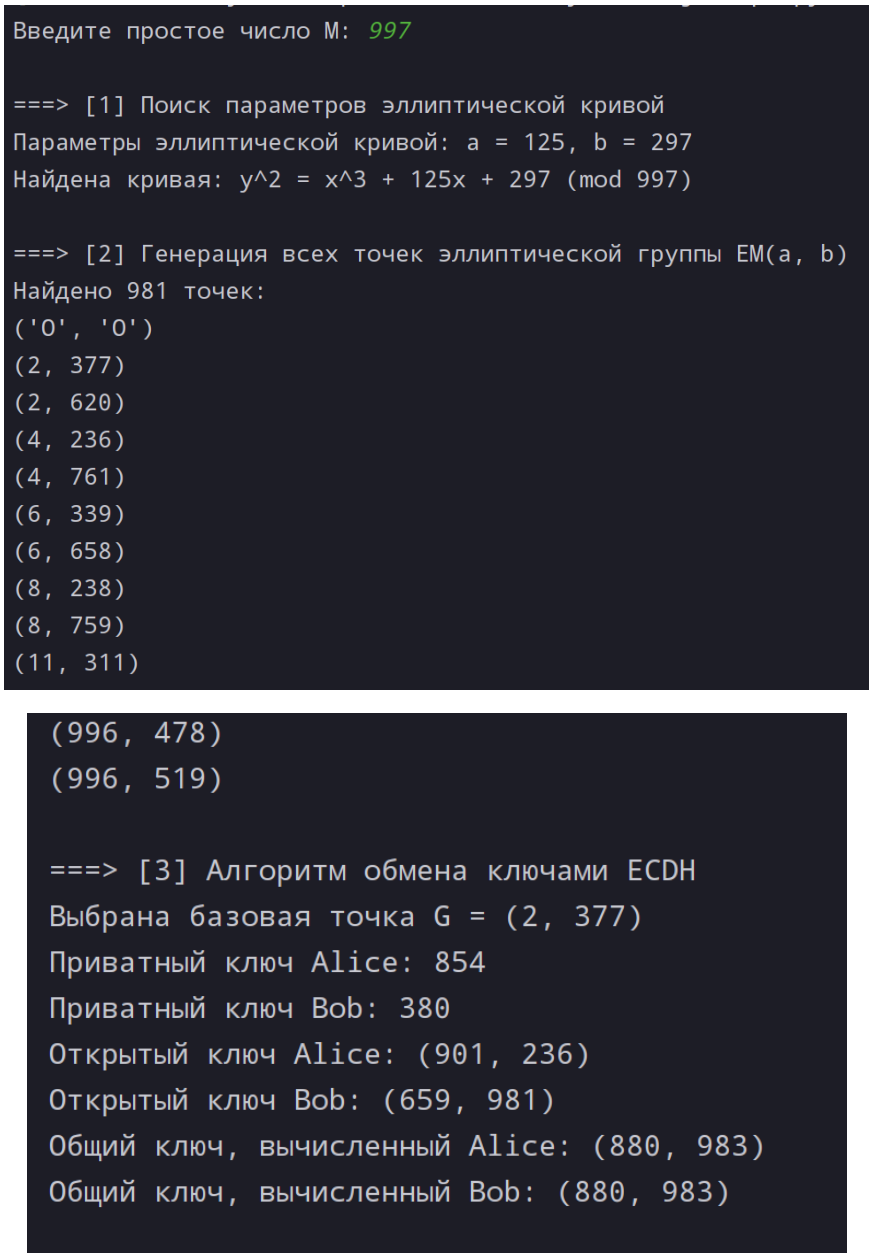
- сообщение → e = SHA-256(message)
- подпись (r, s)
- публичный ключ Q = d·G

3.2. Шаги алгоритма:

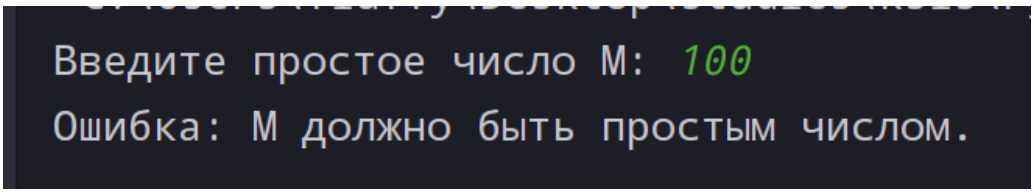
1. Проверить, что $1 \le r < n$ и $1 \le s < n$ (иначе — неверная подпись).
2. Вычислить s^{-1} mod n.
3. Вычислить два скаляра:
   $u_1 = e\cdot s^{-1} \bmod n$,
   $u_2 = r\cdot s^{-1} \bmod n$.

4. Вычислить точку $C = u_1 \cdot G + u_2 \cdot Q$.
5. Извлечь C.x и проверить условие
   $r \equiv C.x \bmod n$.
   Если оно выполняется, подпись считается корректной.

Демонстраций работы программы пункты 1-3:

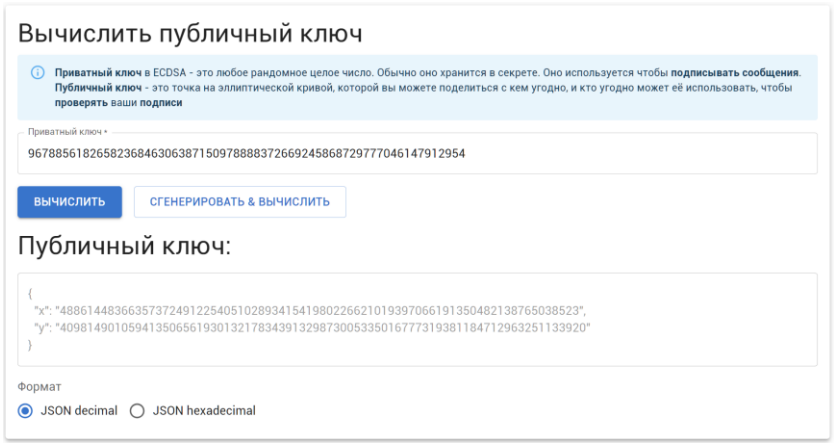


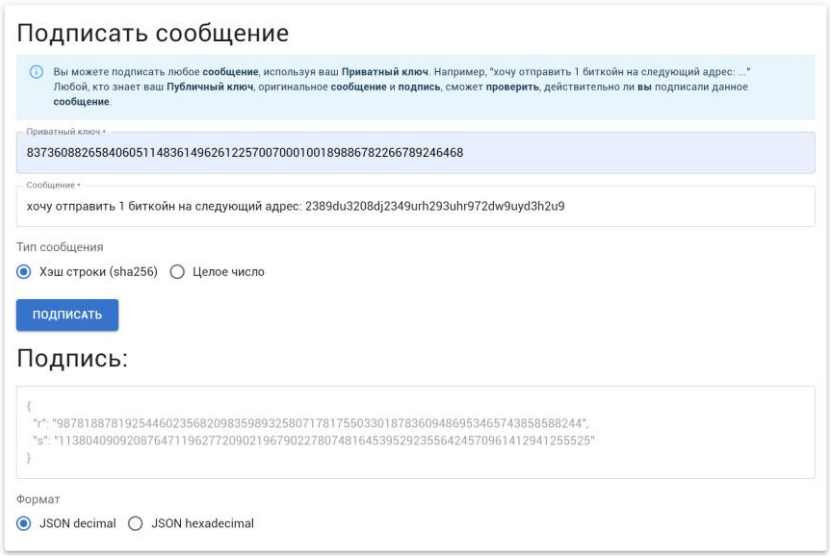
Ввод простого числа, корректное выполнение операций

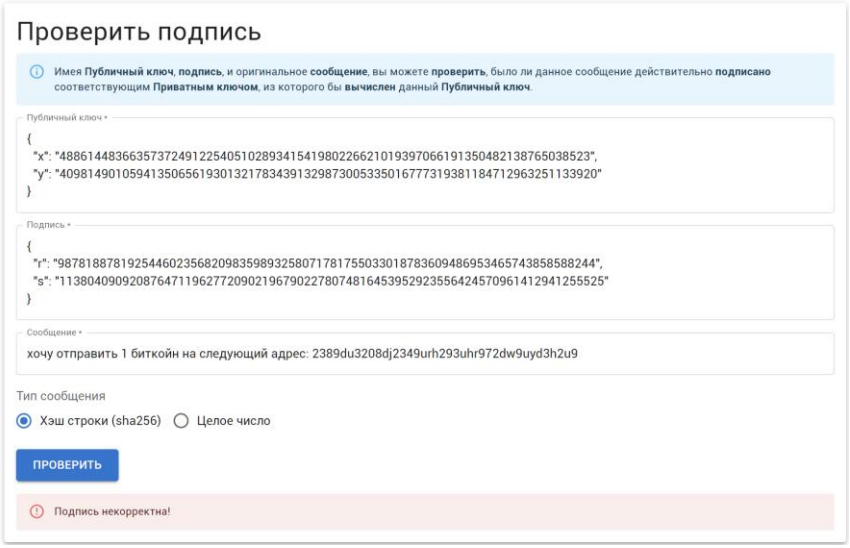


Ввод не простого числа, выдает ошибку

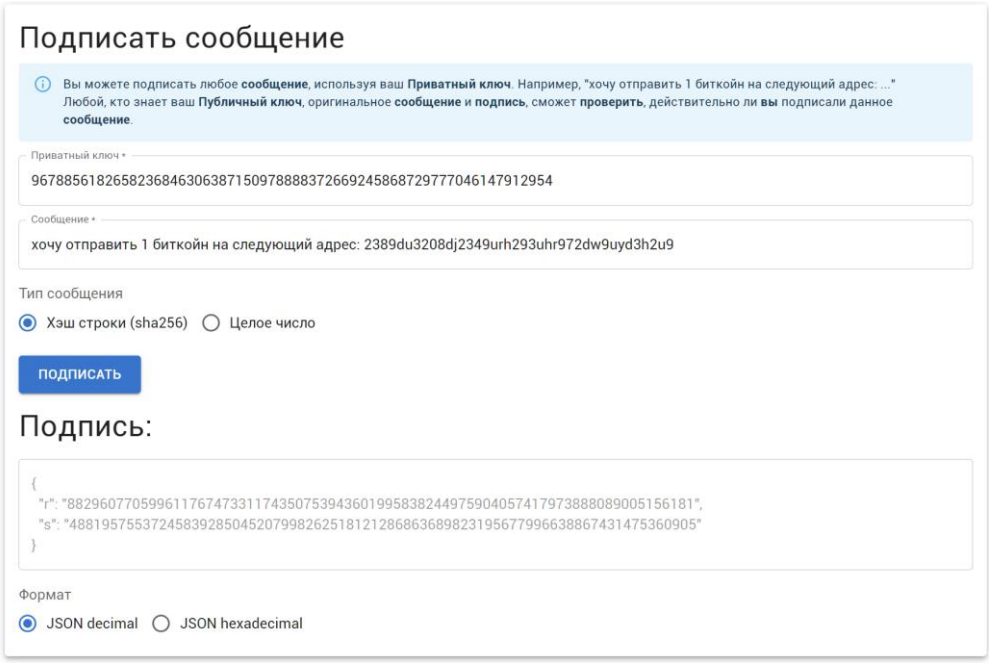
Демонстрация ЭЦП

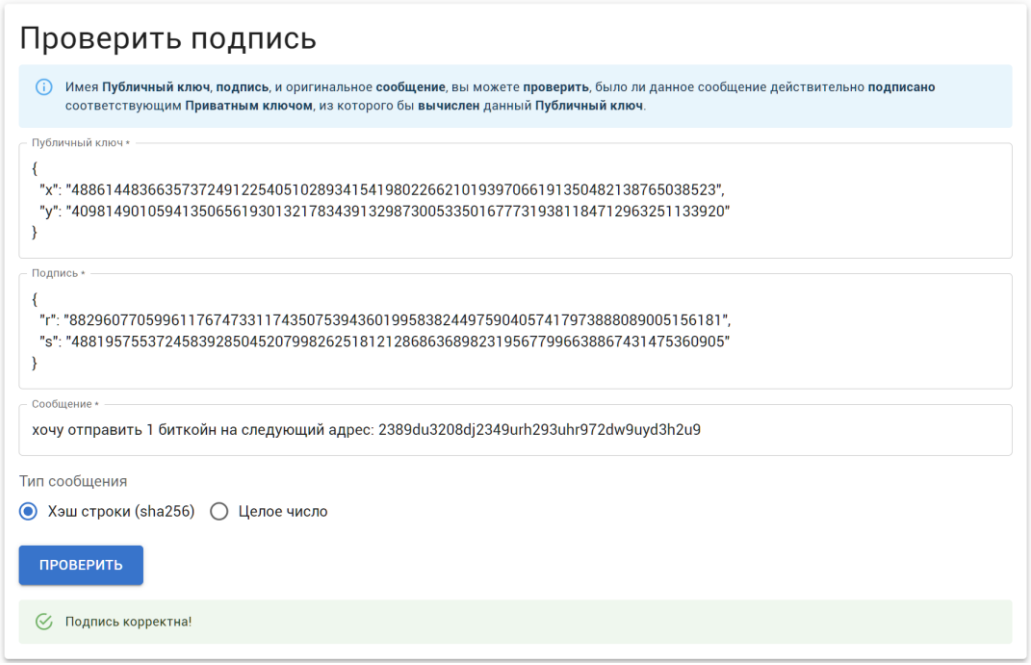




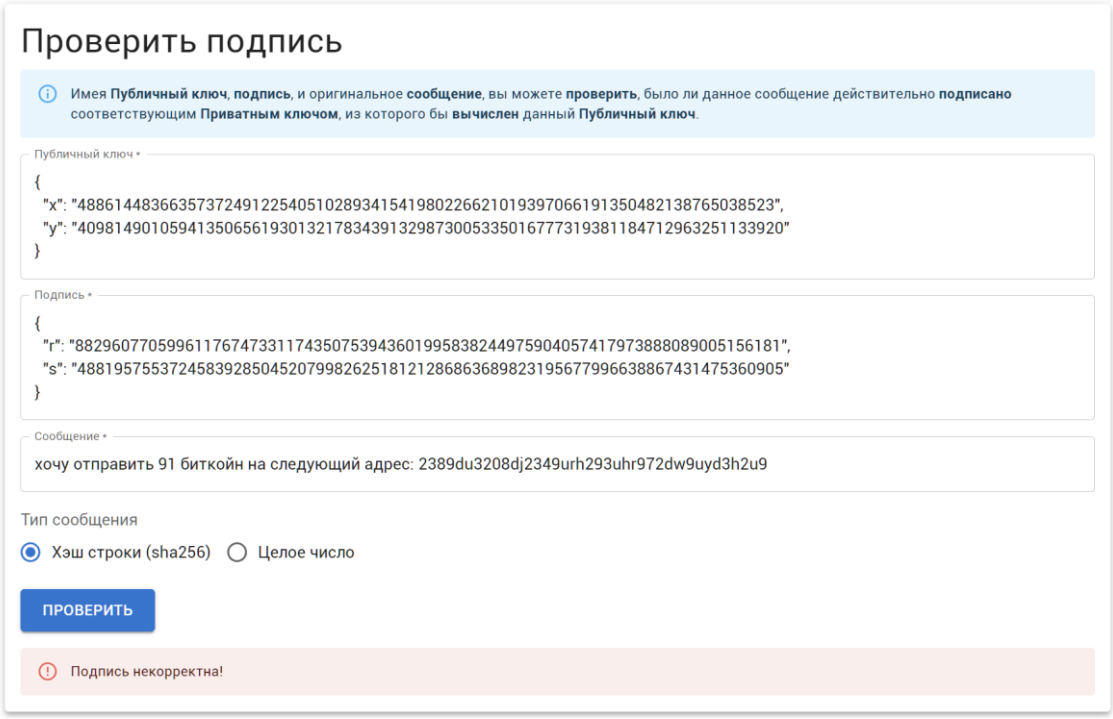


Использовались разные приватные ключи для подписи и генерации публичного ключа. Некорректно.

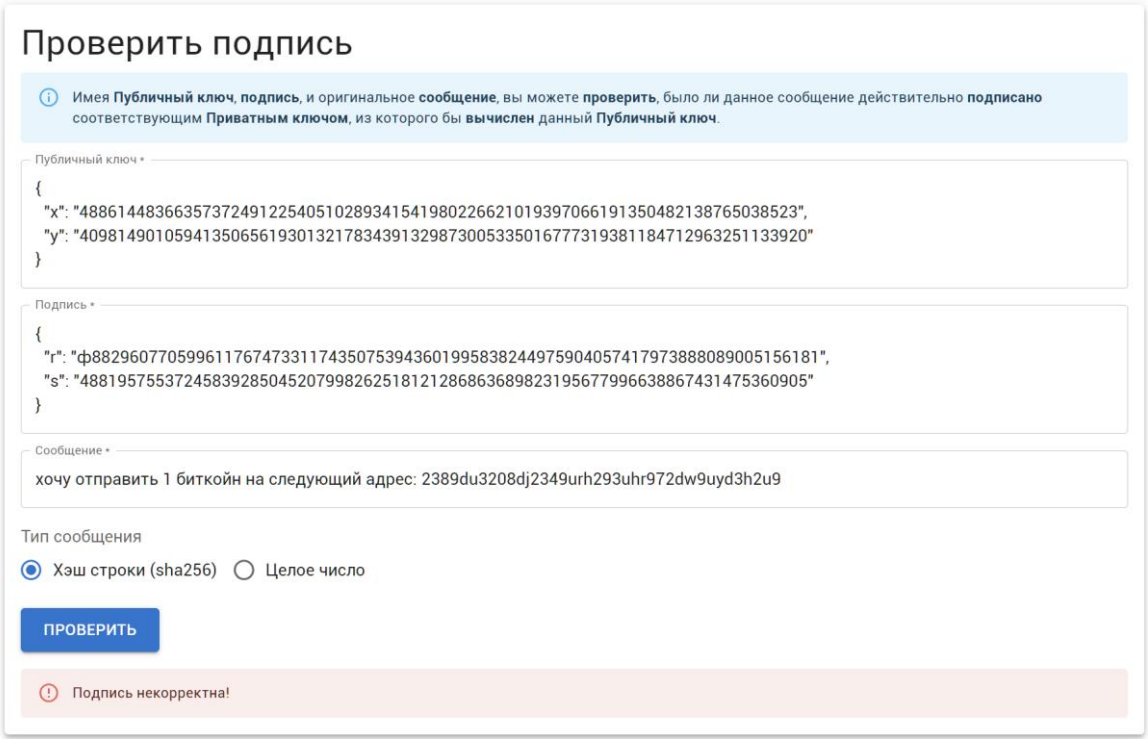




Одинаковые ключи. Корректно.



Изменилось сообщение. Нарушена целостность.



Изменилась подпись. Некорректно

Иные комбинации тест-кейсов, демонстрируют, что алгоритм находит нарушения целостности и аутентичности при передаче цифровой информации.