Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Методы защиты информации

ОТЧЁТ
к лабораторной работе №6
на тему

ЦИФРОВАЯ ПОДПИСЬ

Выполнил: студент гр.253505
Снежко М.А.

Проверил: ассистент кафедры информатики
Герчик А.В.

Минск 2025

СОДЕРЖАНИЕ

1 ЦЕЛЬ РАБОТЫ

Современный этап развития информационных технологий характеризуется повсеместным обменом цифровыми документами и острой необходимостью обеспечения их подлинности и юридической значимости. Под аутентичностью понимается гарантия того, что электронный документ был создан конкретным отправителем и не подвергался изменениям после подписания. Для решения этой фундаментальной задачи информационной безопасности используются механизмы электронной цифровой подписи (ЭЦП).

В отличие от алгоритмов симметричного шифрования, ЭЦП основана на принципах асимметричной криптографии и представляет собой математическую схему, обеспечивающую проверку подлинности электронных документов. Она преобразует произвольный массив данных в уникальную цифровую подпись фиксированной длины с использованием закрытого ключа подписанта. Ключевыми свойствами ЭЦП являются невозможность подделки (стойкость к созданию подписи без знания закрытого ключа), неотрекаемость (невозможность отказа от факта подписания) и гарантия целостности (любые изменения документа делают подпись недействительной).

Одним из таких стандартов является российский алгоритм ГОСТ Р 34.10-2012. Его основное преимущество заключается в соответствии требованиям безопасности Российской Федерации и использовании современных криптографических преобразований. Алгоритм работает в связке с хеш-функцией ГОСТ Р 34.11-2012 ("Стрибог"), что обеспечивает высокий уровень криптостойкости.

Целью данной лабораторной работы является теоретическое и практическое изучение принципов работы электронной цифровой подписи и реализация программного средства формирования и проверки ЭЦП. В рамках работы будет проведена реализация и отечественного стандарта ГОСТ 34.10-2012. Это позволит на практике оценить математические основы алгоритма, этапы формирования и проверки подписи, а также особенности криптографических преобразований.

В практической части работы выполняется программная реализация обоих алгоритмов на языке *Javascript*.

2 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Основное назначение цифровой подписи заключается в обеспечении целостности сообщений, аутентификации отправителя и невозможности последующего отказа от авторства (неотказуемости). Криптографически стойкая схема цифровой подписи должна удовлетворять ряду требований: уникальность подписи для каждого сообщения, невозможность восстановления закрытого ключа по публичному ключу или по подписи, а также защиту от подделки подписи даже при наличии множества подписанных сообщений.

Цифровая подпись на основе эллиптических кривых (алгоритм ГОСТ 34.10) использует математические свойства эллиптических кривых над конечными полями. Основная идея заключается в том, что точку на кривой можно умножать на целое число, получая новую точку. При этом обратная операция – нахождение числа по известной точке – является вычислительно сложной (проблема дискретного логарифмирования), что обеспечивает криптографическую стойкость схемы.

Процесс работы алгоритма включает три ключевых этапа:

1 Генерация ключей. Выбирается случайный закрытый ключ sk, и вычисляется соответствующий публичный ключ $pk=sk \cdot G$, где G – базовая точка на кривой. Публичный ключ может использоваться всеми участниками для проверки подписи, при этом восстановление закрытого ключа невозможно.

2 Создание подписи. Для подписи сообщения сначала вычисляется хеш сообщенияс помощью криптографической хеш-функции (в моем случае, ГОСТ 34.11). Затем выбирается случайное число k, и на его основе формируется пара чисел (r,s), которая и является подписью. Наличие случайного числа k обеспечивает уникальность подписи даже для одного и того же сообщения.

3 Проверка подписи. Проверка подписи выполняется с использованием публичного ключа pk, хеша сообщения и подписи (r,s). Алгоритм вычисляет точки две точки на кривой и проверяет их особое сочетание. Если подпись корректна, результаты проверок совпадают с компонентой r, что подтверждает подлинность подписи и целостность сообщения.

Таким образом, схема цифровой подписи на основе эллиптических кривых сочетает в себе высокую криптографическую стойкость и эффективность вычислений благодаря использованию коротких ключей и сложной структуры эллиптической кривой. В сравнении с другими алгоритмами цифровой подписи, такими как *RSA*, эллиптические кривые позволяют достичь одинакового уровня безопасности при существенно меньших размерах ключей и меньшей вычислительной нагрузке, что делает их удобными для современных приложений, включая защищённую электронную почту, блокчейн и интернет-банкинг.

3 ХОД РАБОТЫ

Программное средство реализовано при помощи языка программирования *Javascript*.

На первом этапе проведено изучение математических основ и структуры алгоритма цифровой подписи. Особое внимание уделено изучению теории эллиптических кривых, генерации ключевых пар и механизмов формирования и проверки подписи. Для ГОСТ 34.10-2012 изучены принципы работы с закрытым и открытым ключом, процесс хеширования по ГОСТ 34.11-2012 и математические преобразования при создании подписи.

После освоения теоретического материала приступили к практической реализации. Были написаны функции для создания ключевой пары и обработки входных сообщений. Для ГОСТ 34.10-2012 реализованы алгоритмы формирования цифровой подписи с использованием закрытого ключа и проверки подписи с использованием открытого ключа.

На рисунке 3.1 изображен результат работы программы.

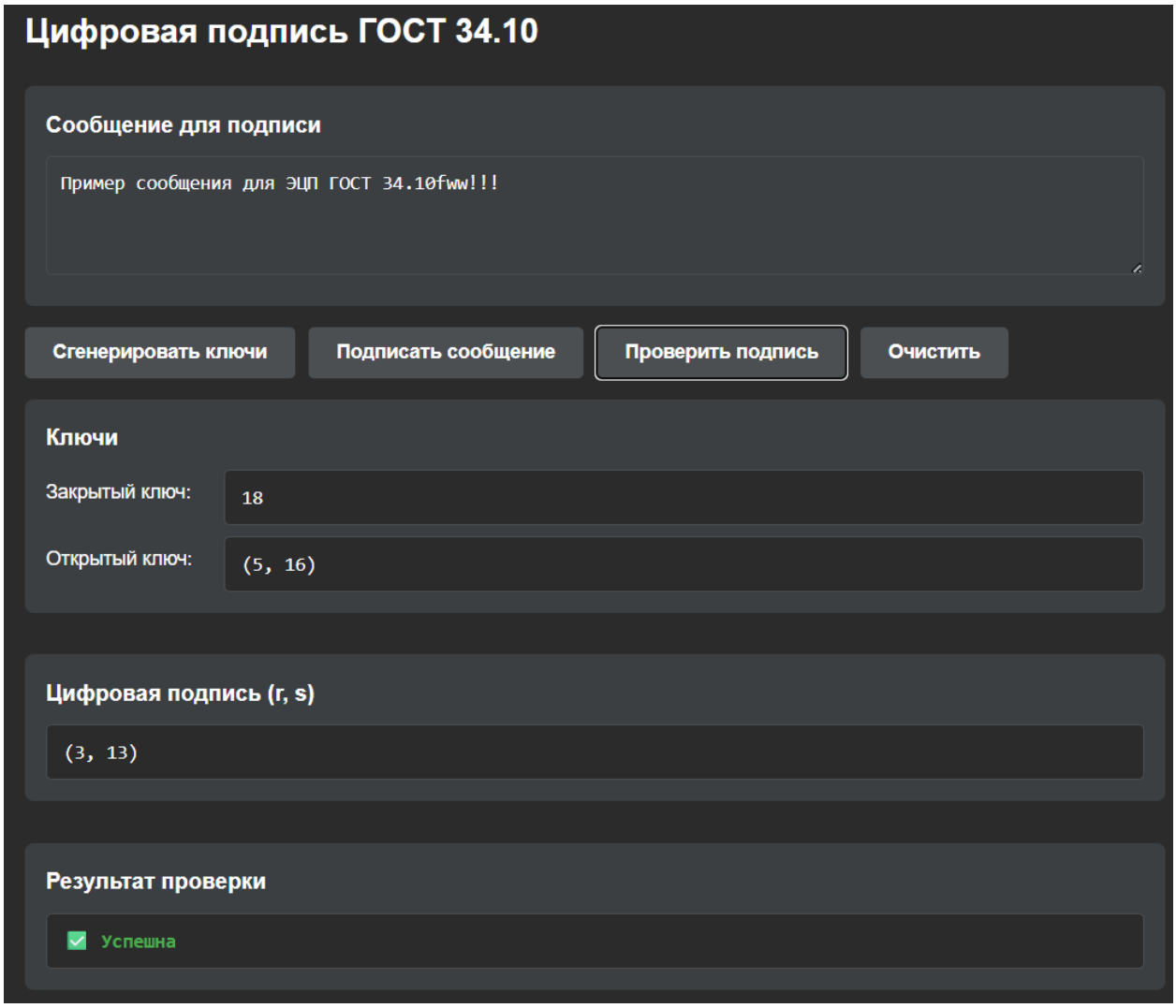

Рисунок 3.1 – Результат работы программы

ЗАКЛЮЧЕНИЕ

В результате выполнения лабораторной работы была успешно реализована система электронной цифровой подписи на базе алгоритма ГОСТ Р 34.10-2012. Практическая реализация охватила все ключевые этапы работы с ЭЦП: создание ключевой пары, формирование цифровой подписи с использованием закрытого ключа и верификацию подписи с помощью открытого ключа. Основным достижением работы стало создание работоспособного программного средства формирования и проверки ЭЦП, наглядно демонстрирующего принципы работы современной криптографической системы.

Реализация подтвердила теоретические положения алгоритма, в частности, важность корректного выполнения математических операций на эллиптической кривой и строгого соблюдения процедуры формирования подписи. Особое значение имела точная реализация взаимодействия с хеш функцией ГОСТ Р 34.11-2012 и преобразования хеш-значения в число для вычислений. Разработанная система продемонстрировала способность надежно подписывать электронные документы и детектировать любые попытки их изменения после подписания.

Таким образом, реализованная система электронной цифровой подписи представляет собой законченное решение, которое демонстрирует принципы построения криптографических средств аутентификации и служит основой для дальнейшего изучения современных методов обеспечения информационной безопасности.

ПРИЛОЖЕНИЕ А
(обязательное)
Листинг программного кода

```
const p = 17
const a = 2;
const b = 2;
const G = [5, 1];
const n = 19;

const curve = new EllipticCurve(p, a, b, G, n);

function hashMessage(message) {
    // Используем GOST3411 из отдельного файла
    if (typeof GOST3411 === 'undefined') {
        console.error('GOST3411 не найден. Убедитесь, что файл gost_34_11.js
подключен.');
        // Fallback: простая хеш-функция для демонстрации
        let hash = 0;
        for (let i = 0; i < message.length; i++) {
            hash = (hash * 31 + message.charCodeAt(i)) % n;
        }
        return hash === 0 ? 1 : hash;
    }

    const gost = new GOST3411(256);
    const hashHex = gost.hashHex(message);


    const hashBigInt = BigInt('0x' + hashHex);
    const hashInt = Number(hashBigInt % BigInt(n));

    return hashInt === 0 ? 1 : hashInt; // Избегаем нулевого хеша
}

function generateKeys() {
    const privateKey = Math.floor(Math.random() * (n - 1)) + 1;
    const publicKey = curve.multiplyPoint(privateKey, G);
    return { privateKey, publicKey };
}

function signMessage(privateKey, message) {
    const e = hashMessage(message);

    while (true) {
        const k = Math.floor(Math.random() * (n - 1)) + 1;
        const R = curve.multiplyPoint(k, G);

        if (R === null) continue;

        const r = R[0] % n;
        if (r === 0) continue;

        const kInv = modinv(k, n);
        const s = (kInv * (e + privateKey * r)) % n;

        if (s !== 0) {
            return { r, s };
        }
    }
}
```