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

УДК 004.056.55

Высоцкий
Георгий Валентинович

Криптография на эллиптических кривых для встраиваемых систем

АВТОРЕФЕРАТ
на соискателя степени магистра технических наук
по специальности 1-40 80 01 «Компьютерная инженерия»

Научный руководитель
Станкевич Андрей Владимирович
доцент, кандидат технических наук

Минск 2021

ВВЕДЕНИЕ

В настоящее время ни для кого не секрет, что компьютерные технологии уже прочно закрепились в нашей повседневной жизни. Уже сложно себе представить компанию или предприятие, которые бы могли обойтись без компьютеров или серверов. Хранить ценную информацию в голове или на бумаге уже не представляется возможным из-за большого количества данных, потому компьютерные технологии столь ценны для нашего общества. Несмотря на то, что компьютерные технологии призваны помочь человеку, но наряду с безграничными возможностями эти технологии приносят новые проблемы.

Эллиптическая криптография изучает ассиметричные криптосистемы, в основе которых лежат эллиптические кривые над конченными полями. Одно из основных преимуществ эллиптической криптографии является то, что на данный момент неизвестно наличие субэкспоненциальных алгоритмов для решения задачи дискретного логарифмирования. Впервые использование эллиптических кривых было независимо предложено Нилом Коблицем и Виктором Миллером в 1985 году.

Особый интерес к эллиптической криптографии вызван такими свойствами, которые применяются в беспроводных коммуникациях – высокое быстродействие и небольшая длина ключа. Использование алгоритмов на базе эллиптических кривых предполагает, что не существует субэкспоненциальных алгоритмов для решения задач дискретного логарифмирования в группах точек кривых, при этом порядок группы точек кривой определяет сложность задачи.

Основными задачами работы являются анализ быстродействия и затрат на оперативную память алгоритмов цифровой подписи, а также программная реализация оптимизированных алгоритмов.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Цель данной работы: анализ алгоритмов ECC для выдачи рекомендаций по их практическому использованию для встраиваемых систем.

Задачи исследования: проанализировать алгоритмы выработки и проверки цифровой подписи в соответствии с СТБ 34.101.45 и ECDSA; выбрать варианты реализации быстрых алгоритмов выработки и проверки цифровой подписи применительно к их использованию; программно реализовать указанные алгоритмы для встраиваемых систем и оценить их вычислительную сложность.

Объект исследования: алгоритмы выработки и проверки цифровой подписи в соответствии с СТБ 34.101.45 и ECDSA.

Предмет исследования: быстрые модификации алгоритмов выработки и проверки цифровой подписи в соответствии с СТБ 34.101.45 и ECDSA.

Личный вклад автора выражен в самостоятельном исследовании:

– анализ стандартов и алгоритмов цифровой подписи;

– сравнительный анализ существующих программных реализаций электронных цифровых подписей;

– анализ алгоритмов цифровой подписи с целью максимизации производительности;

– разработка модуля профилирования и реализация стандарта электронной цифровой подписи СТБ 34.101.45 на языке Python.

Результатом произведенного анализа, расчетов и оптимизации явилась разработка программной реализации СТБ 34.101.45 и модуля для анализа потребляемых ресурсов на языке программирования Python.

Практическая значимость результатов диссертации состоит в выборе алгоритмов для максимизации производительности программной реализации.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

В введении показано, какими основными преимуществами обладают эллиптические кривые в приложениях криптографии.

Использование только лишь алгоритмов шифрования и хеширования не позволило обеспечить надлежащий уровень защиты, а с развитием вычислительной мощности компьютеров задача взлома стала достижимой за приемлемое время.

Использование алгоритмов электронной цифровой подписи позволило решить проблему авторства данных и их уникальности.

Были выделены основные задачи работы:

- анализ стандартов и алгоритмов цифровой подписи;
- сравнительный анализ существующих программных реализаций электронных цифровых подписей;
- анализ алгоритмов цифровой подписи с целью максимизации производительности и минимизации ресурсов системы.

В главе 1 описываются основными свойства и операции над эллиптическими кривыми, выбор наборам параметров для задания эллиптической кривой, описаны основные шаги при шифровании, а также приведено описание, основные параметры и шаги выполнения алгоритма электронной цифровой подписи стандарта СТБ 34.101.45.

Во второй главе были описаны и проанализированы оптимизированные алгоритмы на базе эллиптических кривых, а также рассмотрены такие операции как сложение, умножение, возведение в квадрат, уменьшение размерности поля, поиск обратного элемента и общие модульные алгоритмы для полей $\mathbb{F}p$ и $\mathbb{F}_{2^m}$. Также разобраны и описаны алгоритмы умножения Карацубы, умножение и поиск обратного элемента по методу Монтгомери и по методу Баррета.

В третьей главе были рассмотрены оптимизированные алгоритмы хэширования на примере алгоритмов MD5 и семейства алгоритмов SHA-2, включающих в себя SHA-224, SHA-256, SHA-384 и SHA-512. Описаны оптимизации и модификации для little-endian и big-endian вариантов машин при реализации MD5. Были проанализированы и описаны шаги работы 4ых алгоритмов SHA, описана работы хэш-функций при распараллеливании, а также многорежимная архитектура.

В четвёртой главе были проведены эксперименты по анализу затраченного времени и оперативной памяти при разных наборах входных параметров для реализации стандарта электронной цифровой подписи. Структура проекта выглядела следующим образом:

- ec.py (реализация логики точки эллиптической кривой и операциями над ней)

– verify.py (выработка и проверка цифровых подписей с различным параметром $l \in \{128,192,256\}$)
– belt.py (реализация криптографического стандарта BELT)
– stb.py (реализация стандарта электронной цифровой подписи СТБ-34.101.45-2013)
– profile.py (модуль для профилирования методов по временным затратам и затраченной оперативной памяти)
– montgomery.py (алгоритмы поиска обратного элемента и умножение методом Монтгомери)
– barret.py (алгоритмы поиска обратного элемента и умножение методом Баррета)

Для проведения серии экспериментов будем менять длину параметра эллиптической кривой $l \in \{128,192,256\}$, а также параметры модуль p, коэффициенты a, b, и порядок q. Дополнительно будем изменять методы нахождения обратных элементов со стандартной реализации на методы Монтгомери и Баррета соответственно.

Тестирование будет проводиться на физической машине с операционной системой macOS Big Sur (version 11.0.1), процессором 2.6 GHz 6-core Intel Core i7 и оперативной памятью 16 Gb 2400 MHz DDR4. Результаты всех экспериментов будут представлены ниже. Для начала проверим корректность работы полученной реализации электронной цифровой подписи СТБ-34.101.45-2013 на следующих проверочных параметрах:

Таблица 1 – Стандартный набор параметров СТБ-34.101.45-2013

p $\langle p \rangle_{256}$	$2^{256} - 189$ 43FFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
a $\langle a \rangle_{256}$	$2^{256} - 192$ 43FFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFC$_{16}$
$\langle b \rangle_{256}$	F1039CD6 6B7D2EB2 53928B97 6950F54C BEFBD8E4 AB3AC1D2 EDA8F315 156CCE77$_{16}$
$seed$	5E380100 00000000$_{16}$
q $\langle q \rangle_{256}$	2^{256} – 51 359303463 308904523 350978545 619999225 07663D26 99BF5A7E FC4DFB0D D68E5CD9 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$

Продолжение таблицы 1

$\langle y_G \rangle_{256}$	936A5104 18CF291E 52F608C4 66399178 5D83D651 A3C9E45C 9FD616FB 3CFCF76B$_{16}$

Таблица 2 – Проверочные параметры для выработки электронной цифровой подписи

X	B194BAC8 0A08F53B 366D008E 58$_{16}$
H	ABEF9725 D4C5A835 97A367D1 4494CC25 42F20F65 9DDFECC9 61A3EC55 0CBA8C75$_{16}$
$\langle k \rangle_{256}$	4C0E74B2 CD5811AD 21F23DE7 E0FA742C 3ED6EC48 3C461CE1 5C33A77A A308B7D2$_{16}$
$\langle x_R \rangle_{256}$	CCEEF1A3 13A40664 9D15DA0A 851D486A 695B641B 20611776 252FFDCE 39C71060$_{16}$
$\langle y_R \rangle_{256}$	7C9EA1F3 3C23D20D FCB8485A 88BE6523 A28ECC32 15B47FA2 89D6C9BE 1CE837C0$_{16}$
S_0	E36B7F03 77AE4C52 4027C387 FADF1B20$_{16}$
S_1	CE72F153 0B71F2B5 FD3A8C58 4FE2E1AE D20082E3 0C8AF650 11F4FB54 649DFD3D$_{16}$
S	E36B7F03 77AE4C52 4027C387 FADF1B20 CE72F153 0B71F2B5 FD3A8C58 4FE2E1AE D20082E3 0C8AF650 11F4FB54 649DFD3D$_{16}$

Запустим предварительно написанный на языке Python тест для выработки электронной цифровой подписи, на выходе которого мы должны получить сформированную подпись, которая должна быть равна значениям $(S_0,\ S_1)$:

```
(venv) → ECC(python) python verify.py
Signature created:
0xe36b7f0377ae4c524027c387fadf1b20
0xce72f1530b71f2b5fd3a8c584fe2e1aed20082e30c8af65011f4fb54649dfd3d
```

Рисунок 1 – Результат теста выработки электронной цифровой подписи

Как видно из рисунка 4.1 проверочные значения совпали с результатами нашего теста, а значит логика выработки электронной цифровой подписи работает корректно.

Таблица 3 – Входные параметры для первого эксперимента

Модуль p	$2^{256} - 189$
Коэффициент a	$2^{256} - 192$
Коэффициент b	F1039CD6 6B7D2EB2 53928B97 6950F54C BEFBD8E4 AB3AC1D2 EDA8F315 156CCE77$_{16}$
Порядок q	07663D26 99BF5A7E FC4DFB0D D68E5CD9 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$

Продолжение таблицы 3

y_G	936A5104 18CF291E 52F608C4 66399178 5D83D651 A3C9E45C 9FD616FB 3CFCF76B$_{16}$
Метод	Стандартная реализация

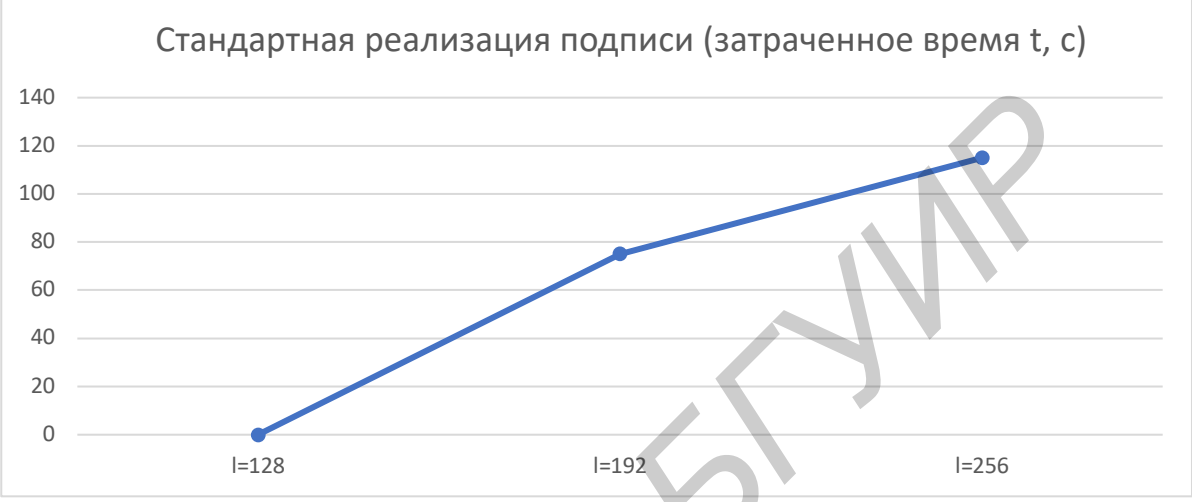

Рисунок 2 – Время, затраченное на выработку 1000 подписей

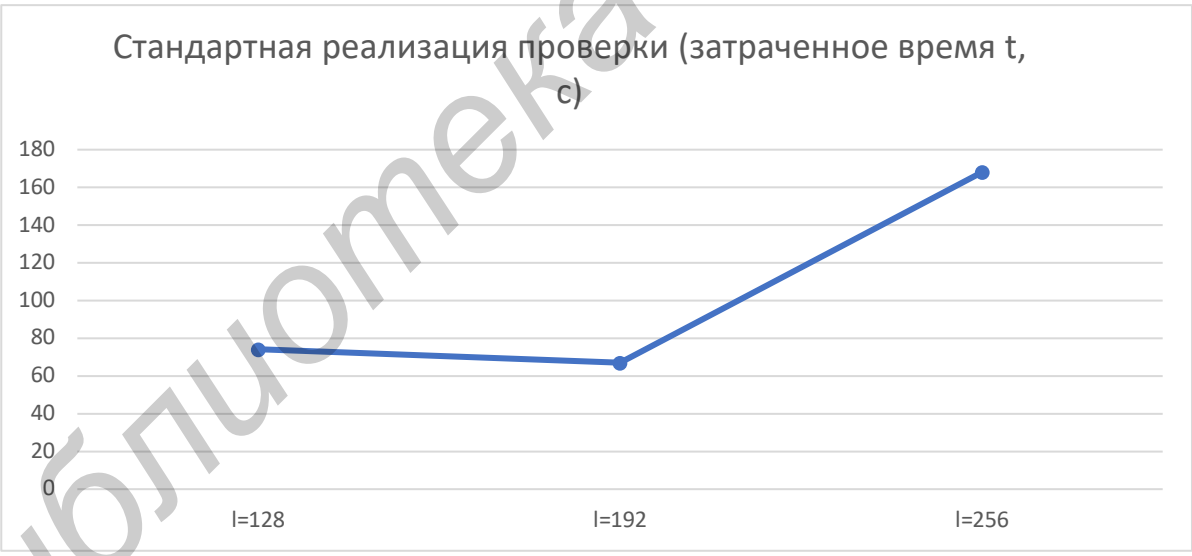

Рисунок 3 – Время, затраченное на проверку 1000 подписей

Таблица 4 – Входные параметры для второго эксперимента

Модуль p	$2^{384} - 317$
Коэффициент a	$2^{384} - 320$
Коэффициент b	64BF7368 23FCA7BC 7CBDCEF3 F0E2BD14 3A2E71E9 F96A21A6 96B1FB0F BB482771 D2345D65 AB5A0733 20EF9C95 E1DF753C$_{16}$
Порядок q	B7A70CF3 3FDCB73D 0AFFA4A6 E7DA4680 BB7BAF73 03C4CC6C FEFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$

Продолжение таблицы 4

y_G	51C433F7 31CB5EEA F9422A6B 273E4084 55D3B166 9EE74905 A0FF86DC 119A723A 89BF2D43 7E113063 9E9E2EA8 2482435D$_{16}$
Метод	Монтгомери

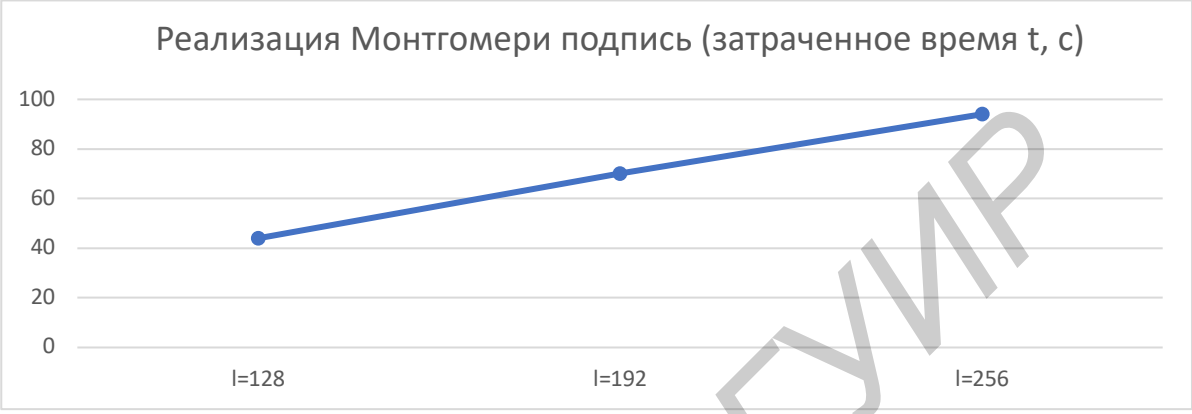

Рисунок 4 – Время, затраченное на выработку 1000 подписей

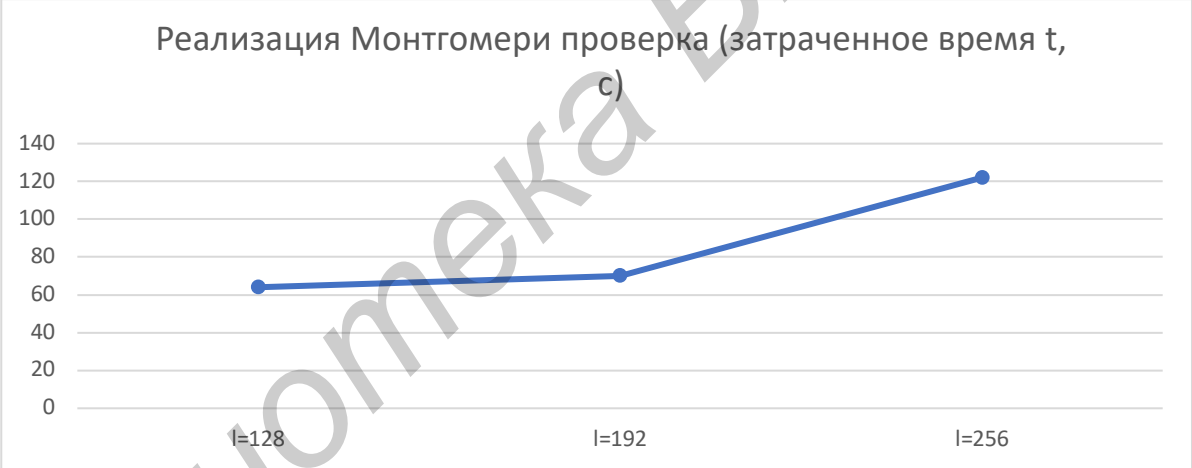

Рисунок 5 – Время, затраченное на проверку 1000 подписей

Таблица 5 – Входные параметры для третьего эксперимента

Модуль p	$2^{512} - 569$
Коэффициент a	$2^{512} - 572$
Коэффициент b	909C13D6 98693409 7AA2493A 272286EA 43A2AC87 8C003329 955E24C4 B5DC1127 88B0ADDA E313CE17 51255DDD EEA9C65B 8958FD60 6A5D8CD8 438C3B93 4459B46C$_{16}$
Порядок q	F18E060D 49ADFFDC 32DF5695 E5CA1B36 F413212E B0EB6BF2 4E009801 2C09C0B2 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
y_G	BDEDEFCE 6FAE92B7 040D4CC9 B983AA67 6122E8EE 957377FF D26FFA0E E2DD7369 DACACC00 1BF8EDD2 E2BC61B3 B341ABB0 AB8FD1A0 F7E682B1 817603E4 7AFF26A8$_{16}$
Метод	Баррет

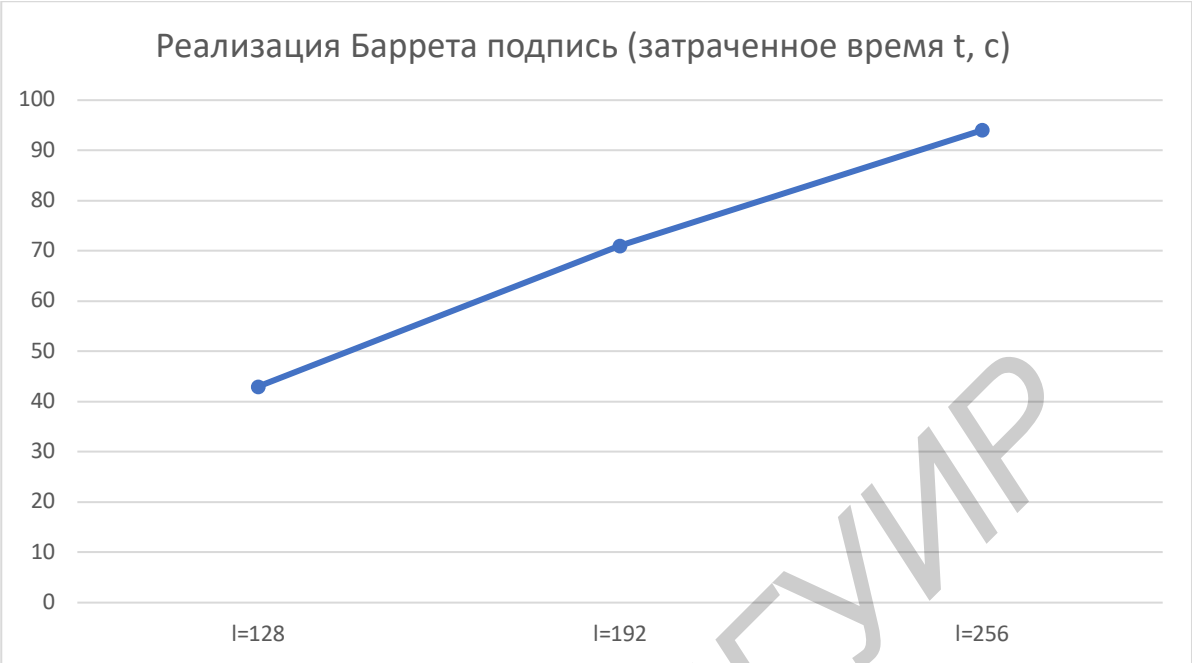

Рисунок 6 – Время, затраченное на выработку 1000 подписей

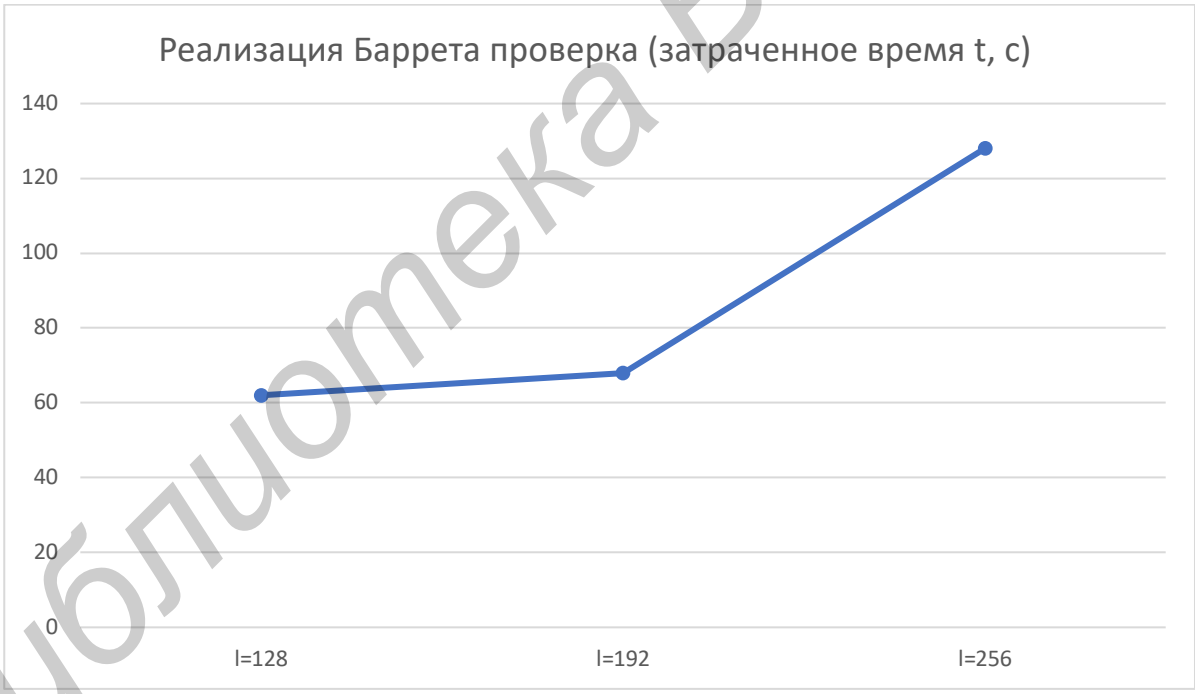

Рисунок 7 – Время, затраченное на проверку 1000 подписей

На представленных выше графиках указываются только временные затраты для трёх методов (стандартного, Баррета и Монтгомери), поскольку затраты по оперативной памяти являются приблизительно одинаковыми и составляют 65Мб для формирования подписей и порядка 1-2Мб для их проверки. Составим итоговую таблицу, в которой будут отражены результаты всех тестов.

Таблица 6 – Обобщённые результаты всех экспериментов

Алгоритм	Значение l, бит	Время выполнения, с (для 1000 подписей)	Тип операции
Стандартная реализация	128	55	Выработка подписи
Стандартная реализация	192	75	Выработка подписи
Стандартная реализация	256	115	Выработка подписи
Стандартная реализация	128	74	Проверка подписи
Стандартная реализация	192	87	Проверка подписи
Стандартная реализация	256	168	Проверка Подписи
Монтгомери	128	45	Выработка подписи
Монтгомери	192	65	Выработка подписи
Монтгомери	256	95	Выработка подписи
Монтгомери	128	64	Проверка подписи
Монтгомери	192	78	Проверка подписи
Монтгомери	256	128	Проверка подписи
Баррет	128	50	Выработка подписи
Баррет	192	70	Выработка подписи
Баррет	256	105	Выработка подписи
Баррет	128	70	Проверка подписи
Баррет	192	83	Проверка подписи
Баррет	256	110	Проверка подписи

Как видно из таблицы 6 в среднем метод Баррета работает как для выработки, так и для проверки подписи на 10-15% лучше стандартной реализации, тогда как метод Монтгомери показывает себя лучше метода Баррета в среднем на 15-20%. Если посчитать количество умножений для нахождения обратного элемента числа длины $2k$, то получим следующие результаты: стандартный метод – $k(k + 2.5)$, метод Баррета – $k(k + 4)$, метод Монтгомери – $k(k + 1)$.

ЗАКЛЮЧЕНИЕ

В результате написания магистерской диссертации были изучены основные математические методы для работы с эллиптическими кривыми, применительно к задачам криптографии, проанализированы современные криптографические стандарты электронной цифровой подписи и алгоритмы для работы с эллиптическими кривыми, проведены работа по изучению реализаций основных хэш-функций и их оптимизациям, изучены модификации алгоритмов для эллиптических кривых и их оптимизации на программном уровне.

Также был реализован стандарт электронной цифровой подписи СТБ 34.101.45-2013 на языке программирования Python, разработан модуль для профилирования кода и алгоритмов как по временным затратам, так и по расходам оперативной памяти.

Проведено сравнение разных типов алгоритмов, а также их влияние на производительность базовой реализации стандарта цифровой подписи. Была произведена оценка производительности по времени выполнения и расходу оперативной памяти двух методов (умножения и нахождения обратного элемента методом Монтгомери, а также умножение и нахождение обратного элемента методом Баррета) по сравнению со стандартной реализацией. Исходя из полученных данных, можно сделать вывод о том, что в среднем метод Баррета обеспечивает производительность как для выработки, так и для проверки подписи на 10-15% выше стандартной реализации, тогда как метод Монтгомери показывает себя лучше метода Баррета в среднем на 15-20%.

Улучшение работы разработанной системы может быть проведено по следующим направлениям:

- уменьшение количества операций копирования объектов и использования кэша объектов;
- разработка системы на языке программирования более низкого уровня, что позволит уменьшить время выполнения кода за счёт меньшего числа абстракций над обработкой кода;
- использование оптимизированных решений более высокой точности, использующих более оптимальные математические методы и модели;
- выбор отличного от стандартного алгоритма хэширования и анализ его производительности.

Таким образом, реализованный стандарт и модуль для профилирования кода выполняют все функции, которые были заложены в целях выполнения данной работы, однако как было показано выше, при более высоких требованиях к производительности системы она может быть доработана с учётом указанных замечаний.

СПИСОК ОПУБЛИКОВАННЫХ РАБОТ

[1-А.] Высоцкий, Г. В. Криптография на эллиптических кривых для встраиваемых систем / Г. В. Высоцкий // Вычислительные методы, модели и образовательные технологии 2020. – Брест: БрГУ. — 2020.

[2-А.] Высоцкий, Г. В. Elliptic-curve cryptography / Г. В. Высоцкий, И. Ю. Супринович // Проблемы экономики и информационных технологий: сборник тезисов докладов 56ой научной конференции аспирантов, магистрантов и студентов БГУИР, Минск, 18 – 20 мая 2020 г. — 2020.