Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Інститут інформатики та радіоелектроніки, факультет радіоелектроніки та телекомунікацій
(повне найменування інституту, назва факультету)

Кафедра захисту інформації
(повна назва кафедри)

Пояснювальна записка

до дипломного проекту (роботи)
магістра
(ступінь вищої освіти (освітній ступінь))

на тему Дослідження криптосистем на базі NTRU

Виконав: студент 6 курсу, групи РТ-813м
спеціальності (напряму підготовки)
125 Кібербезпека
(код і назва напряму підготовки, спеціальності)

Яцький О. В.
(прізвище та ініціали)

Керівник Козіна Г. Л.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

м.Запоріжжя
2018 рік

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Запорізький національний технічний університет

(повне найменування вищого навчального закладу)

Інститут, факультет **ІПРЕ, ФРЕТ**

Кафедра захисту інформації

Ступінь вищої освіти (освітній ступінь) магістр

Спеціальність **125 Кібербезпека**

(код і назва)

Напрям підготовки Безпека інформаційних і комунікаційних систем

(код і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри захисту інформації,

проф., д.т.н., Карпуков Л.М.

"____" _____2018 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЕКТ СТУДЕНТУ

Яцькому Олегу Віталійовичу

(прізвище, ім'я, по батькові)

1. Тема проекту Дослідження криптосистем на базі NTRU

керівник проекту Козіна Галина Леонідівна, канд. фіз.-мат. наук, доцент кафедри захисту інформації ,

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від " 7 " листопада 2018 року №328

2. Строк подання студентом проекту 4 грудня 2018

3. Вихідні дані до проекту Література з досліджень криптосистем на базі перетворень у кільцях зрізаних поліномів, пакети прикладних програм для математичних обчислень.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Визначення базових операцій у кільцях зрізаних поліномів та задач на яких базується складність криптосистеми NTRU, побудова процедур генерації ключової пари, отримання хеш-повідомлення електронного документа, формування і перевірки цифрового підпису, здійснення підписання документа і перевірки підпису за допомогою програмно реалізованих процедур за рекомендованими значеннями параметрів алгоритму NTRU.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Консультанти розділів проекту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
Основний	Козіна Г.Л., доцент кафедри захисту інформації		
Охорона праці	Коробко О.В., старший викладач кафедри ОПНС		
Економічна частина	Лівошко Т.В., доцент кафедри обліку та оподаткування		
Нормо-контроль	Нікуліщев Г.І., старший викладач кафедри захисту інформації		

7. Дата видачі завдання 1 жовтня 2018

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту	Строк виконання етапів проекту (роботи)	Примітка
1	Підбір та ознайомлення з літературою	1.10 – 7.10	
2	Ознайомлення з математичним обґрунтуванням криптосистем NTRU	8.10 – 14.10	
3	Ознайомлення з процедурами алгоритму цифрового підпису на базі NTRU	15.10 – 21.10	
4	Програмна реалізація процедур генерації ключів, хешування, формування і перевірки підпису NTRU	22.10 – 28.10	
5	Моделювання алгоритму цифрового підпису NTRU за рекомендованих значень параметрів	29.10 – 4.11	
6	Аналіз і обґрунтування застосування криптосистем на базі NTRU	5.11 – 11.11	
7	Охорона праці та безпека у надзвичайних ситуаціях	12.11 – 18.11	
8	Техніко-економічне обґрунтування проекту	19.11 – 25.11	
9	Оформлення пояснювальної записки	26.11– 4.12	

Студент _____ Яцький О. В.
(підпис) (прізвище та ініціали)

Керівник проекту _____ Козіна Г. Л.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

ПЗ: 98 страниц, 37 рисунков, 15 таблиц, 2 приложения,18 источников.

Объект исследования – криптографические алгоритмы основывающийся на преобразованиях в усёченных кольцах многочленов NTRU.

Цель работы – создание программного обеспечения реализующего алгоритм электронной цифровой подписи на базе NTRU для подписания электронных документов.

Метод исследования – выбор алгоритмов и генерация параметров для реализации программного обеспечения получения и проверки цифровой подписи электронных документов.

Эффективные квантовые компьютеры, которые могут появиться в будущем, позволяют взламывать существующие криптографические алгоритмы за приемлемое время. Однако уже сейчас существуют альтернативные алгоритмы, устойчивые к квантовому криптоанализу. В этой работе проводится анализ работы алгоритма цифровой подписи, основывающихся на преобразованиях в кольцах полиномов. Рассматривается базовые операции и вычислительно-сложные задачи, на которых основываются такие алгоритмы. Приводится сравнительная характеристика алгоритмов на устойчивость и быстродействие, обосновывается область их применения. Описываются процедуры генерации ключевой пары, выбор основных и дополнительных параметров, формирования и проверки подписи. На основе алгоритмического описания алгоритма выполнена программная реализация его процедур на языке Maple.

ЭЛЕКТРОННАЯ ЦИФРОВАЯ ПОДПИСЬ, УСЕЧЁННЫЙ ПОЛИНОМ, РЕШЁТКА, КОЛЬЦО ПОЛИНОМОВ, БАЗИС РЕШЁТКИ, КВАНТОВЫЕ ВЫЧИСЛЕНИЯ

СОДЕРЖАНИЕ

СПИСОК СОКРАЩЕНИЙ

ЭЦП, ЦП – электронная цифровая подпись

ФКП – фактор кольцо полиномов

ПО – программное обеспечение

SVP – задача нахождения кратчайшего вектора

CVP - задача нахождения ближайшего вектора

ПК – персональный компьютер

ГО – гражданская оборона

ЧС – чрезвычайные ситуация

$\mathbb{Z}$ - множество целых чисел

R – кольцо многочленов степени, не превосходящей N

N – целое число, определяющее максимальную степень полиномов, размерность NTRU-решётки

q – большой модуль преобразования в кольце, имеет степень 2.

i, j – порядковый номер итерации или массива

R^N – векторное пространство степени N

L^N – решётка, размерностью N

Z^N – конечное число вычетов по модулю N в пространстве R^N

U, B – базис решётки

$\lambda(L)$ – длина вектора решётки L

I – идеал

$f(x)$ – унитарный приведённый многочлен степени N

p – малый модуль преобразования в кольце

L_M - пространство сообщений

f, g или f, f' – полиномы, представляющие собой секретный ключ

f^{-1} – полином, обратный f в кольце R

F, G – короткие полиномы, являющиеся элементами секретного ключа

D_{FG} – пространство из векторов, центральная норма которых меньше параметра KeyNormBound

D_h – пространство открытого ключа

h – полином, представляющий собой открытый ключ

τ – параметр, обозначающий тип ключа

(s, t) – вектор из полиномов, такой что $(s, t) \in L$, компоненты подписи

r – целое число, компонент подписи

m – полином сообщения, представляется в виде вектора (m_1, m_2)

H – хеш-функция на цифровом пространстве документа $\mathcal{D}$

$\mathcal{N}$ - нормирующая связь, параметр “normBound”

R_f, R_g – результанты f и $X^N - 1$, g и $X^N - 1$

b – Эвклидова норма полиномов, вычисленная при проверке подписи

$\mathbb{R}$ - множество вещественных чисел

D – цифровой документ

e – единичная матрица

k – уровень безопасности в битах

$R_в$ – сопротивление растеканию тока одиночного вертикального заземлителя

$R_и$ – сопротивление искусственного заземлителя

$R_{гр}$ – сопротивление растеканию тока группового заземлителя

$В_а$ – амортизационные отчисления

$Н_а$ – норма амортизации

$К_{пі}$ - затраты на разработку продукта

$К_к$ - капитальные вложения в производственные фонды

K - капитальные затраты

$В_{экс}$ - эксплуатационные расходы

$Т_{ок}$ - срок окупаемости капитальных затрат

E - коэффициент экономической эффективности вложений

P - годовая экономия от внедрения

$Е_р$ - годовой экономический эффект

ВВЕДЕНИЕ

Для ускорения и упрощения процесса документооборота в нынешнее время как правило всё чаще используют электронную форму. Для того, чтобы юридическая сила электронных документов приравнивалась к бумажным, на них следует оставить электронную цифровую подпись.

Электронная цифровая подпись или ЭЦП представляет собой реквизит электронного документа, полученный в результате криптографического преобразования информации с использованием секретного ключа подписи. В отличии от рукописного аналога, электронная цифровая подпись не только предоставляет информацию о лице, которое подписало документ, но также и позволяет убедиться, что документ не был изменён или подделан после подписания.

На данный момент в Украине электронные цифровые подписи нашли своё место среди сотрудников государственной и местной власти при подаче деклараций, ответственных лиц различных организаций и предприятий, а также физических лиц для подачи налоговой отчётности через интернет. Кроме того, каждый год в стране расширяется перечень услуг, предоставляемых через интернет. И, если речь идет о тех, где используются персональные данные или требуется установить личность пользователя, такие сервисы требуют регистрации и/или входа в систему с помощью электронной цифровой подписи.

Цифровые подписи используются для получения информации и оформления документов в Пенсионном фонде, для получения справки о судимости онлайн и для многих других онлайн-сервисов (минюста, фискальной службы) которые требуют надёжной идентификации пользователя. В частности с её помощью теперь можно заказать загранпаспорт, заказать и получить электронную выписку из многих госреестров, другие документы. Только в Минюсте насчитывается 33

информационных системы, для пользования которыми необходимо иметь ключ ЭЦП.

С другой стороны, имеет место процесс непрерывного совершенствования информационных технологий и вычислительной техники. Вследствие этого, существующие на сегодняшний день криптографические протоколы и алгоритмы, в том числе и электронные цифровые подписи, удовлетворяющие по производительности и успешно выполняющие свои задачи по защите информации сегодня, уже вызывают сомнения в своей эффективности на ближайшее будущее.

В последнее время активно разрабатываются алгоритмы и стандарты криптосистем, основанных на математических задачах теории решеток, которые отличаются от предыдущих стандартов тем, что обеспечивают более высокий уровень стойкости при меньшей вычислительной сложности криптографических преобразований. Важность этого направления исследований обусловлена также наличием у задач теории решеток свойств криптостойкости к алгоритмам, выполняемым на квантовых компьютерах. Роль последнего обстоятельства, в свете недавнего открытия коллективом ученных под руководством Джереми О’Брайена механизма создания фотонных квантовых компьютеров посредством традиционной литографии СБИС, резко возрастает.

При реализации криптографических систем одним из наиболее критических моментов является увеличение уровня стойкости. Этот вопрос напрямую связан со свойствами методов и алгоритмов, которые используются для формирования криптографических параметров и ключевых данных.

1 МАТЕМАТИЧЕСКОЕ ОБОСНОВАНИЕ КРИПТОСИСТЕМ NTRU

В 1990-х годах математики Джефри Хостейн, Джил Пайфер, Джозеф Сильверман разработали метод ассиметричного криптографического преобразования в фактор кольцах полиномов (ФКП) – NTRU, что означает N-Th Deegre Truncated Polynomial Ring – в переводе это кольцо усечённых полиномов N-й степени [1, с. 1].

В этом разделе рассматриваются понятие кольца многочленов как такового, понятие решётки, её базиса, кратчайшего и ближайшего вектора, а также вычислительно-сложные задачи, которые лежат в основе NTRU криптосистем.

1.1 Определение кольца полиномов и операции в нём

Кольцо – это алгебраическая структура, в которой заданы две бинарные операции: обратимого сложения и операция умножения, по свойствам похожие на соответствующие операции над числами. Для всех элементов кольца выполняются следующие условия:

- коммутативности сложения, а в некоторых случаях и умножения;
- ассоциативности сложения;
- существование нейтрального элемента относительно сложения;
- существование противоположного элемента относительно сложения;
- дистрибутивности.

Примером кольца может служить множество целых чисел $\mathbb{Z}$. Также в качестве элементов кольца могут выступать и полиномы.

В криптографических системах на базе NTRU преобразования осуществляются в кольце полиномов степени не превосходящей $N - 1$:

$$R = Z[X]/\ (X^N - 1), \tag{1.1}$$

где N – это простое целое число.

В некоторых случаях может использоваться фактор кольцо $R_q = Z_q[X]/(X^N - 1)$, где коэффициенты взяты по модулю q в диапазоне (-q/2, q/2), где q как правило имеет степень 2[2, с. 3].

Полином в кольце $a(x) \subset R$, может быть представлен вектором его коэффициентов следующим образом:

$$a = \sum_{i=1}^{N-1} a_i x_i = (a_0, a_1, \dots, a_{N-1}). \quad (1.2)$$

Также для любого полинома $a \in R$ удобно определить матрицу свёртки следующим образом: пусть M_a – это циркулянтная матрица размерностью $N \times N$ с индексами {0, …, N - 1}, где элемент с позицией (i, j) равен $a_{(j - i) \bmod N}$ [3, с. 4]:

$$M_a = \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-2} & a_{n-1} \\ a_{n-1} & a_0 & \cdots & a_{n-3} & a_{n-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_1 & a_2 & \cdots & a_{n-1} & a_0 \end{pmatrix}. \quad (1.3)$$

Умножение в кольце R имеет сходство с обычным умножением полиномов, но с учетом соотношения $X^{N+k} = X^k$ для любого k $\geq$ 0. Произведение двух полиномов a * b, где $a = a_0 + a_1X + \dots + a_{N-1}X^{N-1}$, $b = b_0 + b_1X + \dots + b_{N-1}X^{N-1}$ представляется в виде:

$$(a * b)_k = \sum_{i+j=k \bmod N} a_i b_i. \quad (1.4)$$

Умножение двух полиномов в кольце R также называется циклическим произведением или циклической свёрткой двух полиномов.

Произведение полиномов a и b может быть также представлено как произведение вектора $(a_0, \ldots, a_{N-1})$ на матрицу M_b [2, с. 4].

1.2 Понятие решётки, её базис и кратчайший вектор

Решётка представляет собой дискретную аддитивную подгруппу, заданную в пространстве R^N, в этом случае решётку L можно представить как множество целочисленных линейных комбинаций:

$$L(b_1, \ldots, b_N) = \{\sum_{i=1}^{N} x_i\, b_i : x_1, \ldots, x_N \in Z\}, \qquad (1.5)$$

где N - количество линейно независимых базисных векторов $\{\bar{b}_1, \ldots, \bar{b}_N\} \subset R^N$ в N-мерном Эвклидовом пространстве, также N соответствует размерности самой решётки [4].

Простейшим примером решётки в теории групп может служить Z^N - конечное кольцо вычетов (подгруппа) по модулю натурального числа N в пространстве R^N. Графически решётку можно представить как разбиение пространства примитивной ячейкой. На рисунке 1.1 представлены решётки в Эвклидовой плоскости с различными базисами [4, с. 2].

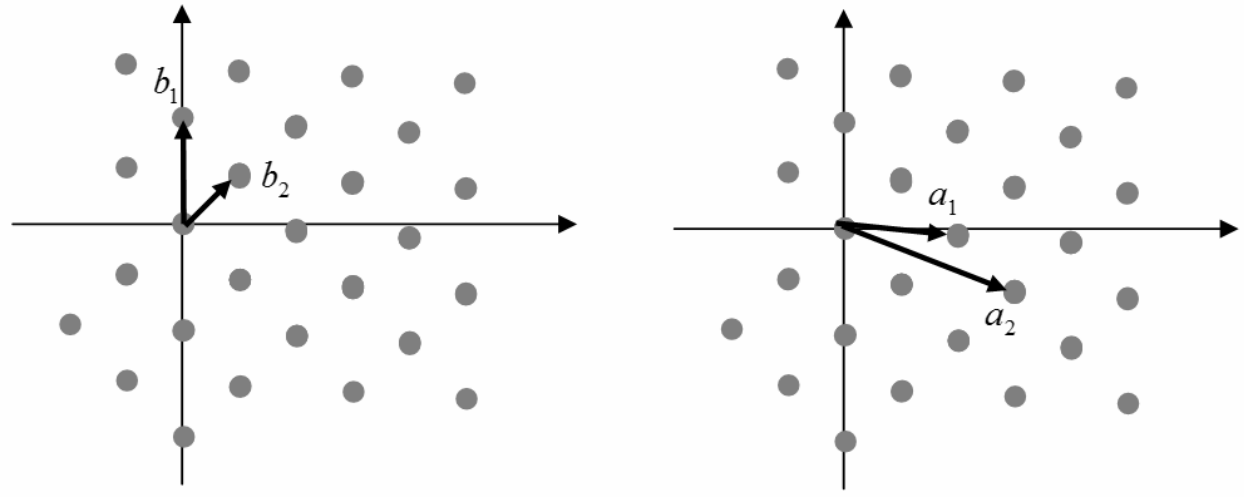

Рисунок 1.1 – Решётки: а) с базисом $\{\bar{b}_1, \bar{b}_2\} \in B$;
б) с базисом $\{\bar{a}_1, \bar{a}_2\} \in B$.

В тоже время, аналогично множеству натуральных чисел, идею решёток можно рассматривать для любого конечномерного векторного пространства над каким угодно полем или кольцом.

Под базисом решётки понимается множество *N* линейно независимых векторов, при помощи которых путём линейной комбинации можно получить любой вектор векторного пространства R^N. Таким образом, базис решётки – это множество линейно независимых векторов её порождающих, в свою очередь у решётки может быть множество базисов $L = \sum_{i=1}^{n} \overline{a}_{\text{i}} \cdot Z$.

Ненулевой вектор решётки минимальной длины называется её кратчайшим вектором. Под кратчайшим вектором решётки понимается вектор, длина которого $\lambda(L) = \min\limits_{x,y \in L,\ x \neq y} \|x - y\| = \min\limits_{x \in L,\ x \neq 0} \|x\|$ (рис. 1.2) [5, с. 3].

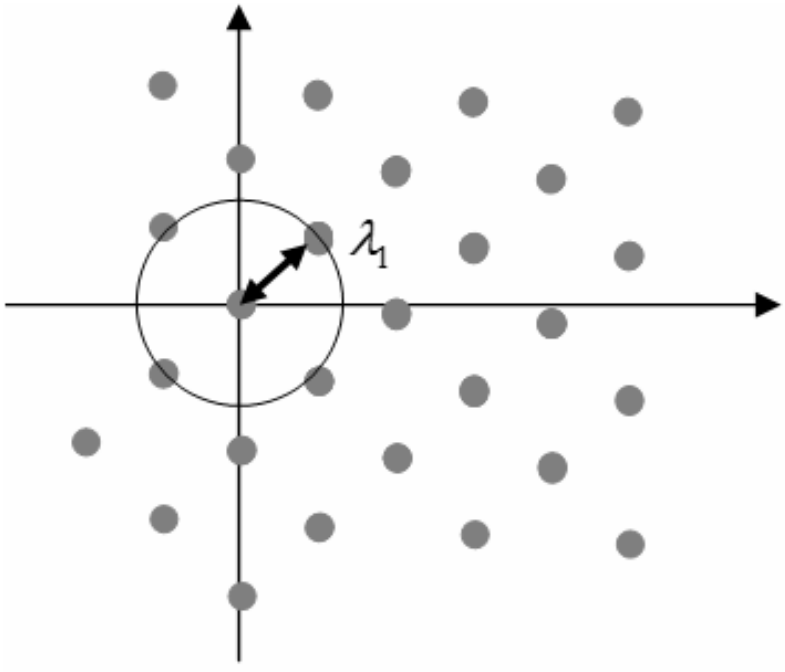

Рисунок 1.2 – Кратчайший вектор

Тогда многомерным обобщением этого понятия для решётки размерностью *N*, L^N будет i – й последовательный минимум $\lambda_i(L)$, под которым понимают наименьший радиус окружности (шара в случае

трёхмерного пространства и т.д.) содержащий i – линейно независимых векторов:

$$\lambda_i^p(L) = r, r \in R : \exists v_i \in L, \max_i \|v_i\| \leq r, \qquad (1.6)$$

где v_i - это линейно независимые вектора (рис. 1.3). Первый последовательный минимум в решётке $\lambda_i^p(L)$ соответствует длине кратчайшего вектора в решётке [5, с. 3].

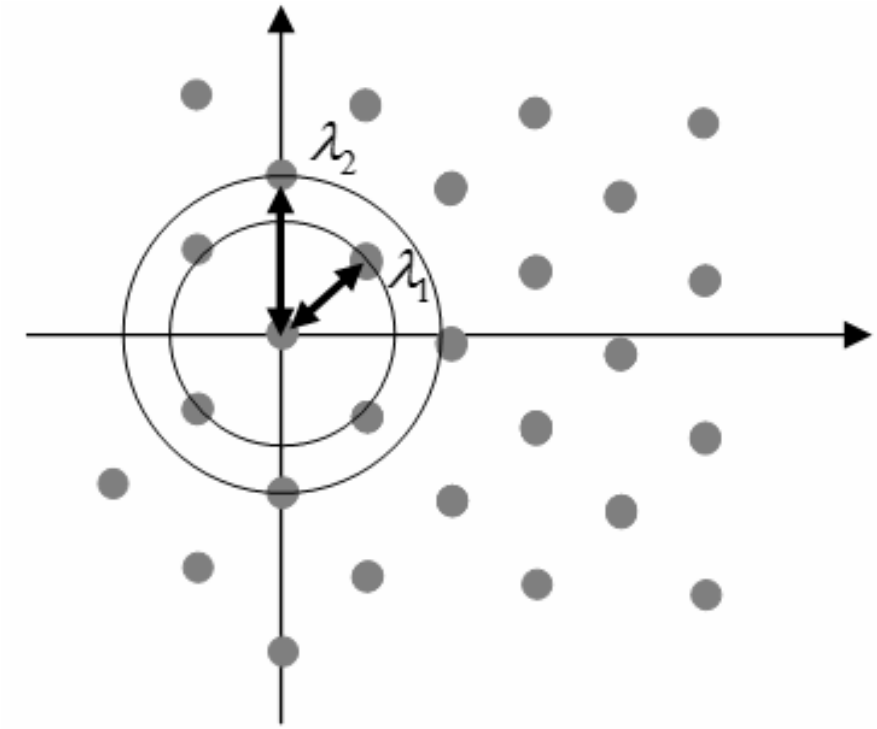

Рисунок 1.3 – Кратчайший базис решётки L в R^2

Идеальная решётка – это решетка со свойствами идеала на некотором кольце чисел. Под идеалом I в теории групп понимается такое подмножество кольца R, для которого выполняется условие: $R \cdot I \subset I$. Таким образом результат сложения и умножения векторов в идеальной решетке так же принадлежит ей самой.

Пусть задано кольцо многочленов с целочисленным коэффициентами $R = Z[x]/f(x)$ взятых по модулю $f(x)$, где $f(x)$ – унитарный приведённый

многочлен степени N. Так как заданное отношение изоморфно над Z^N и аддитивные группы и идеалы кольца являются подгруппами в R, полученная конструкция представляет собой решётку. Такой класс решёток принято называть идеальными решётками [5, с. 3].

Таким образом, когда определены основные понятия и определения теории решёток можно перейти к разбору основных задач, применяемых при построении и оценке сложности в криптографических системах рассматриваемого класса NTRU.

1.3 Вычислительно сложные задачи NTRU

Криптосистемы на базе NTRU основываются преимущественно на двух типах задач:

- нахождения по базису решётки B кратчайшего ненулевого вектора SVP (Shortest vector problem)
- нахождения ближайшего вектора $\bar{b} \in L(B)$ по базису решётки B и заданному вектору $\bar{j} \notin L(B)$, CVP (Closest vector problem).

Пусть U – это базис решётки L. Задача нахождения кратчайшего вектора (задача SVP) состоит в том, чтобы найти такой вектор $\lambda \in L$, $\lambda \neq L$, что $\forall \nu \in L$, $\|\lambda\| \leq \|\nu\|$. То, насколько короткой может быть длина ненулевого вектора в произвольной решётке зависит от таких свойств, как размерность решётки и её детерминант. Так N-размерная решётка L имеет экспоненциально много векторов с нормой $d = \sqrt{N}\det(L)^{1/\mathrm{N}}$.

Пример задачи нахождения кратчайшего вектора SVP для пространства R^2 приведён на рисунке 1.4 [5, с. 4].

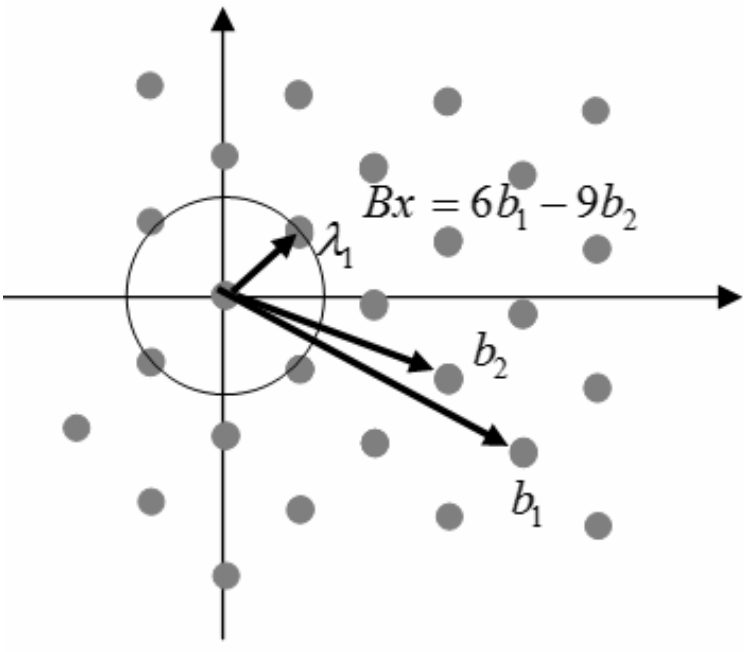

Рисунок 1.4 – Пример SVP-задачи в R^2

Задача CVP (нахождения ближайшего вектора) тесно связана и является неоднородным (гетерогенным) вариантом задачи нахождения кратчайшего вектора. Она состоит в нахождении вектора $\bar{b} \in L$, который является ближайшим к вектору $\bar{j}$, где $\bar{j} \in R^N$ и $\bar{j}$ не находится в L [5, с. 4].

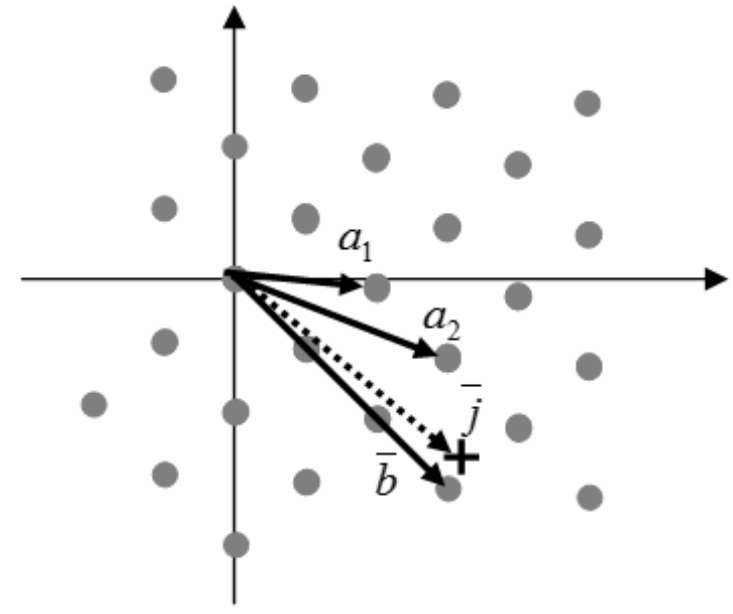

Рисунок 1.5 – Пример CVP-задачи в R^2

Таким образом, необходимо найти такой вектор $\bar{b} \in L$, который минимизировал бы Эвклидову норму $\left\|\bar{\jmath} - \bar{b}\right\|$. Здесь выражение $\left\|\bar{\jmath} - \bar{b}\right\|$ обозначает наименьшее расстояние между векторами $\bar{\jmath}$ и $\bar{b}$, которое вычисляется как Эвклидова норма вектора $\|\cdot\|$. В то же время, Эвклидова норма вектора $a = (a_0, a_1, \dots, a_{N-1})$ определяет его длину и вычисляется по формуле:

$$\|a\| = \sqrt{(a_0)^2 + (a_1)^2 + \ \dots + (a_{N-1})^2}. \tag{1.7}$$

Далее будет применятся понятие базиса минимальной длины, под которым понимается такой базис U решётки L, который состоит из кратчайших векторов $\lambda_i \in L$, таким образом $U = (\lambda_0, \lambda_1, \dots, \lambda_{N-1})$ и $\forall \nu \in L$, $\forall \lambda_i \in U : \|\lambda_i\| \leq \|\nu\|$ [1, с. 3].

Для удобства оценки длины векторов предлагается различать большие вектора $a = (a_0, a_1, \dots, a_{N-1})$, когда их длина намного больше чем длина кратчайшего вектора решётки $\forall \lambda_i \in U : \|\lambda_i\| = \|a\|$.

Аналогично будет применятся понятие коротких векторов $a = (a_0, a_1, \dots, a_{N-1})$, в этом случае их норма приблизительно равна [5]: $\|a\| \approx \sqrt{(N-1)/12}$.

Стоит отметить, что под длиной полинома $a = \sum_{i=1}^{N-1} a_i x_i$ будет пониматься длина соответствующего вектора $a = (a_0, a_1, \dots, a_{N-1})$, таким образом под коротким или большим полиномом будет пониматься короткий или большой вектор соответственно, в свою очередь базис, составленный из больших векторов, будет называться большим базисом [1, с. 3].

2. ОПИСАНИЕ АЛГОРИТМА ЦИФРОВОЙ ПОДПИСИ NTRU

Метод ассиметричного криптографического преобразования в фактор кольце полиномов послужил основой для разработки алгоритмов шифрования таких как NTRUEncrypt и алгоритмов цифровой подписи с открытым ключом.

В данном разделе рассматривается алгоритм цифровой подписи NTRUSign, приведено описание основных и дополнительных параметров алгоритма, алгоритмы генерации ключей в NTRU, описание алгоритмов формирования и проверки подписи, а также рассматривается сам процесс получения и проверки подписи на математическом примере.

2.1 Основные и дополнительные параметры NTRUSign

К основным параметрам в первую очередь относятся параметры N, q и p, а к дополнительным относятся такие параметры, как KeyNormBound, MaxAdjustment, perturbationBases.

Степень N определяет размерность усечённого кольца многочленов R. Поскольку параметр N определён как степень NTRUSign, элементы кольца представлены как полиномы степени $N - 1$.

Параметр q – больший модуль преобразования. Также в паре с ним может использоваться и p – малый модуль преобразования, который должен являться взаимно простым с q (например: $p = 3$, $q = 2048$). Кроме того, параметр q определяет интервал, которому должны принадлежать все коэффициенты многочленов, использующихся в криптосистеме. Так, пространство сообщений L_M определяется как $L_M = \{M(x) \in R\}$, при этом коэффициенты полиномов сообщений лежат в диапазоне $[-(q - 1) / 2, (q - 1) / 2]$ [6, с. 195].

Секретный ключ NTRUSign включает в себя полиномы (f, g), его матричное представление называется секретным базисом решётки

L, который является базисом минимальной длины. Решётка L генерируется всеми линейными комбинациями строк матрицы [3, с. 6]:

$$\begin{pmatrix} f & F \\ g & G \end{pmatrix}. \tag{2.1}$$

В общем случае, f и g могут быть выбраны как любые небольшие полиномы, но f обязательно должен иметь обратный f^{-1} в $(Z / qZ)[X](X^N - 1)$. Однако для увеличения производительности, f может выбираться из D_f – пространства всех полиномов степени $N - 1$, которые имеют d_f коэффициентов, равных 1, и остальные коэффициенты равны 0. Аналогично g может выбираться из пространства D_g степени $N - 1$, которые имеют d_g коэффициентов, равных 1, и остальные равны 0 [7, с. 101].

После выбора f и g, формирующая базис пара (F, G) вычисляется так, что (f, g) и (F, G) формируют небольшой базис для NTRUSign модуля. Необходимо чтобы выполнялось условие $f \cdot G - g \cdot F = q$, также (F, G) короткие полиномы, их норма векторного представления примерно соответствует $\|F\| = \sqrt{N/12}$. Пар (F, G) для заданных (f, g) может быть множество, но необходимо выбирать наименьшие из этого множества.

Для того чтобы максимизировать вероятность генерации «хорошей» подписи, (F, G) должны быть ограничены пространством D_{FG}, которое состоит из всех векторов, у которых центрированная норма меньше чем параметр безопасности KeyNormBound. Если в процессе генерации пара (F, G) не может быть найдена с центрированной нормой меньше чем KeyNormBound, секретный ключ должен быть отброшен и выбран заново. Для минимизации коэффициентов полиномов может использоваться итерационный алгоритм, в котором количество итераций для выбора ограничивается параметром MaxAdjustment [7, с. 102].

Для формирования ЦП следует использовать полиномы пар (f, g) и пар (F, G). Выбор элементов пары определяется типом личного ключа. Тип

ключа принимает 2 значения: “standard” (стандартный) или “transpose” (транспонированный). Использование транспонированного базиса предпочтительнее, поскольку позволяет значительно повысить производительность в сравнении со стандартным базисом. Однако в этом случае, один компонент подписи будет значительно меньше чем другой, что приведёт к раскрытию информации о секретном ключе быстрее, чем в случае стандартного базиса. Поэтому использование транспонированного базиса рекомендуется только тогда, когда был использован, по крайней мере, хотя бы один цикл искажения (пертурбации) секретного ключа.

Каждое использованное искажение в цикле значительно увеличивает количество подписей, которые необходимо использовать злоумышленнику для атаки на секретный ключ, поэтому создание ЦП выполняется в несколько этапов. Для каждого этапа используются внутренний секретный и открытый ключи. Для каждого очередного шага вычисляется новое значение подписываемых данных. Для формирования ЦП на всех этапах необходимо иметь соответствующее количество личных ключей. Поэтому вместо обычной функции генерации личного ключа используется цикл генерации. Для проверки ЦП на стороне получателя используется последний открытый ключ. Количество базисов искажений задаётся параметром perturbationBases. Стоит отметить, что многоуровневое формирование ЦП приводит к увеличению значения нормы на величину $\sqrt{perturbationBases + 1}$ [6, с. 196].

В стандартной решётке пространство открытого ключа D_h состоит из всех полиномов h степени $N - 1$ таких, что $h = f^{-1} \cdot g$ с коэффициентами по модулю q, где f и g выбраны из D_f и D_g соответственно, и f^{-1} – полином с коэффициентами по модулю q такой, что $f^{-1} \cdot f = f \cdot f^{-1} = 1 \, mod \, q$ в $(Z / qZ)[X](X^N - 1)$.

В транспонированном базисе, пространство открытого ключа состоит из всех полиномов h степени $N - 1$ таких, что $h \equiv f^{-1} \cdot F \equiv g^{-1} \cdot G \; (mod \; q)$.

Открытый ключ h формирует так называемый открытый базис решётки, который является большим базисом, где e – единичная матрица [7, с. 102]:

$$\begin{pmatrix} e & h \\ 0 & q \end{pmatrix}. \quad (2.2)$$

В отличие от алгоритмов ЦП, которые нашли широкое распространение (RSA, DSA, ECDSA, ДСТУ 4145-2004), в NTRU открытый ключ используется не только для проверки, но и для ее создания. Поэтому открытый ключ либо вычисляется в процессе выработки ЦП, либо входит в структуру с секретным ключом. Последнее более эффективно с точки зрения вычислительной сложности операции создания ЦП. Необходимость использования открытого ключа при создании ЦП связана с тем, что не все ЦП, которые получаются в соответствии с алгоритмом создания, могут быть проверены. ЦП считается «хорошей» не только в случае, если норма вектора соответствующего полиному, полученному в качестве ЦП, удовлетворяет определенным соотношениям. Предельное значение нормы определяется параметром $\mathcal{N}$ или NormBound. Поэтому после выработки ЦП она проверяется, и, только в случае успешной проверки, возвращается в качестве окончательного значения ЦП. Параметр выбирается достаточно малым для предотвращения атак подделки подписи и достаточно большим для того, чтобы подпись лежала ниже границы NormBound. Значение Эвклидовой нормы полиномов вычисляется по формуле 2.3:

$$\|s\|^2 = \sum_{i=0}^{N-1} s_i^2 - \frac{1}{N}\left(\sum_{i=0}^{N-1} s_i\right)^2. \quad (2.3)$$

При этом, ограничивается количество попыток, которые используются для создания ЦП. Параметр, который определяет число попыток, обозначается SignFailTolerance [6, с. 196].

2.2 Описание процедур генерации ключей, формирования и проверки подписи

Генерация ключей в NTRUSign состоит из следующих этапов:

1. Входные данные: целые *N, q*, и строка τ = “standard” или “transpose”.
2. Произвольно выбрать $f, g \in R$.
3. Найти малые $F, G \in R$, такие что $f \cdot G - F \cdot g = q$.
4. Если τ = “standard”, задать $f = f$ и $f' = F$. Если τ = “ transpose”, то задать $f = f$ и $f' = g$.
5. Вычислить $h = f^{-1} \cdot f' \; mod \; q$,.
6. Открытый ключ: $h \equiv f^{-1} \cdot f' \; mod \; q$.
7. Секретный ключ: множество $\{f, g, F, G\}$.

Подпись – это вектор $(s, t) \in L$, который находится очень близко от сообщения m = (m_1, m_2), где m_1 и m_2 это две равные части полинома m, которые при конкатенации образуют целостное сообщение $m_1 \parallel m_2$. Сообщение вначале хэшируется, так что m – это, собственно, хэш-функция $H : \mathcal{D} \rightarrow R$ на цифровом пространстве документа $\mathcal{D}$. В процессе подписания документа также вычисляется норма полинома $\|\cdot\| : R^2 \rightarrow \mathbb{R}$ и “normBound” (нормирующая связь) $\mathcal{N} \in \mathbb{R}$ [3, с. 3].

Алгоритм формирования подписи состоит из следующих этапов:

1. Входные данные: цифровой документ $D \in \mathcal{D}$ и секретный ключ $\{f, g, F, G\}$.
2. Задать $r = 0$.
3. Сформировать секретный базис решётки M (матрицу секретного ключа).
4. Выполнить инверсию M^{-1}.
5. Если $r > 0$ то представить r в виде битовой строки и задать $m = H(D \parallel r)$, иначе $m = H(D)$.
6. Вычислить $M_m^{-1} = m \cdot M^{-1}$.

7. Округлить коэффициенты M_m^{-1} до ближайшего целого.
8. Вычислить подпись $(s, t) = M_m^{-1} \cdot M$.
9. Проверка подписи на соответствие $\mathcal{N}$:
   9.1. Задать $b = \|(s - m_1, t - m_2)\|$.
   9.2. Если $b \geq \mathcal{N}$, то задать $r = r + 1$ и вернуться на шаг 5.
10. На выходе подписанное документ $\{ D, (s, t), r\}$.

По сути, полученный вектор (s, t) – это выражение вектора (m_1, m_2) в секретном базисе решётки с округлением. Правильная подпись демонстрирует, что подписывающий знает точку решётки (r, s) в пределах нормальной границы (normBound) от вектора сообщения m [8]. При проверке подписи вычисляется расстояние от (s, t) до (m_1, m_2), как норма разницы между этими векторами, это расстояние должно быть не больше чем заранее вычисленное проверочное расстояние $\mathcal{N}$. Оно должно быть небольшим, так как при реализации подписи используются короткие полиномы, если же расстояние от (s, t) до (m_1, m_2) больше чем $\mathcal{N}$, то подпись недействительна. То есть при формировании подписи использовались полиномы с коэффициентами большими чем у секретного ключа. Таким образом, действительная подпись демонстрирует решение задачи нахождения ближайшего вектора $(s, t) \in L$ к заданному вектору $(m_1, m_2) \in R$. Согласно расчётам в работе [8]:

$$\mathcal{N} = \frac{c^2 N^2}{6} + \frac{c^2 N^3}{72}, \tag{2.4}$$

где $c = \sqrt{(2\pi e/q\lambda)(\lambda^2\|f\|^2 + \|2g\|^2)}$, при $\lambda = 1$.

Проверка подписи на стороне проверяющего требует такую же хеш-функцию H, норму полинома $\|\cdot\|$ и normBound $\mathcal{N} \in \mathbb{R}$, и состоит из таких этапов [3, с. 4]:

1. Входные данные: подписанный документ $\{ D , (s, t), r\}$ и открытый ключ h.

2. Если $r > 0$ то представить r в виде битовой строки и задать $m = H(D \parallel r)$, иначе $m = H(D)$.

3. Вычислить $b = \|(s - m_1, t - m_2)\|$.

4. Подпись верна если $b < \mathcal{N}$, в противном случае подпись не проходит проверку.

В качестве рекомендуемых разработчиками параметров могут приниматься следующие:

$$(N, q, \tau, \mathcal{N}) = (251, 128, \text{“transpose”}, 310). \quad (2.5)$$

2.3 Пример формирования подписи NTRUSign

Для наглядности был рассмотрен пример подписи без искажений [1, с. 5], где определено то, какой же базис больше подходит для формирования подписи: открытый или секретный. Пусть, для примера задаются такие параметры:

$$q = 32, N = 2, h = (0, -20).$$

Сначала строится решётка при помощи базиса минимальной длины:

$$(0, 3) = f, (0, 4) = g, (0, -5) = F, (0, 4) = \text{G}.$$

Затем находится точка (s, t), достаточно близкая к сообщению m = (0, 0, 0, 17), то есть m_1 = (0, 0), m_2 = (0, 17). Также открытый базис будет сформирован как циклический сдвиг q и полиномов e, h в матрице с размерностью 2N:

$$\begin{pmatrix} 0 & 1 & 0 & -20 \\ 1 & 0 & -20 & 0 \\ 0 & 0 & 0 & 32 \\ 0 & 0 & 32 & 0 \end{pmatrix}.$$

Для данного секретного ключа нормальную границу $\mathcal{N}$ можно рассчитать по следующему соотношению [9]:

$$\|s - m_1, t - m_2\|^2 = \|\varsigma_1 f + \varsigma_2 F, \varsigma_1 g + \varsigma_2 G\|^2. \quad (2.6)$$

Это более точная граница для маленьких чисел примера, в данном случае:

$$\|\varsigma_1 f + \varsigma_2 F, \varsigma_1 g + \varsigma_2 G\| =$$
$$= \sqrt{(3 \cdot 3 + (4 \cdot 4) + \big((-5) + (-5)\big) + 2 \cdot 2} = 8{,}12.$$

Тут $\varsigma_{1,2}$ – полиномы, которые имеют коэффициенты в диапазоне (-1/2, 1/2) [8], а значит ими можно пренебречь для примера с маленькими числами, то есть $\varsigma_{1,2} = 1$.

Выполнено подписание на секретном ключе. Матрица секретного ключа формируется циклическим сдвигом полиномов данного ключа:

$$\begin{pmatrix} f & g \\ F & G \end{pmatrix}^{-1} = \begin{pmatrix} 0 & 3 & 0 & 4 \\ 3 & 0 & 4 & 0 \\ 0 & -5 & 0 & 4 \\ -5 & 0 & 4 & 0 \end{pmatrix}^{-1} =$$
$$= \begin{pmatrix} 0 & 4/32 & 0 & 4/32 \\ 4/32 & 0 & -4/32 & 0 \\ 0 & 5/32 & 0 & 3/32 \\ 5/32 & 0 & 3/32 & 0 \end{pmatrix},$$

откуда:

$$(m_1, m_2)\begin{pmatrix} f & g \\ F & G \end{pmatrix}^{-1} =$$

$$= (0,0,0,17)\begin{pmatrix} 0 & 4/32 & 0 & 4/32 \\ 4/32 & 0 & -4/32 & 0 \\ 0 & 5/32 & 0 & 3/32 \\ 5/32 & 0 & 3/32 & 0 \end{pmatrix} =$$

$$= \left(\frac{85}{32}, 0, \frac{51}{32}, 0\right) \approx (3,0,2,0).$$

Наконец имеем подпись:

$$(s,t) = (3,0,2,0)\begin{pmatrix} 0 & 3 & 0 & 4 \\ 3 & 0 & 4 & 0 \\ 0 & -5 & 0 & 4 \\ -5 & 0 & 4 & 0 \end{pmatrix} = (0,-1,0,20).$$

Чтобы удостоверится что (*s*, *t*) принадлежит решётке, вычислим: $s \cdot h = (0, -1) \cdot (0, -20) \equiv (0, 20)\ mod\ 32 = t'$. Здесь и далее операции умножения векторов означают умножение полиномов [1, с. 5].

Теперь, для проверки подписи, находится расстояние между подписью и сообщением: $\|(0 - 0) + (0 - (-1)) + (0 - 0) + (17 - 20)\| = \sqrt{10} \approx 3$. Таким образом полученное расстояние меньше нормированной границы, $3 < \mathcal{N} = 8{,}12$.

Большой базис не подходит для нахождении ближайшего вектора. Для того чтобы показать это, подписано сообщение и осуществлена проверка при помощи открытого базиса. Имеем:

$$\begin{pmatrix} e & h \\ 0 & q \end{pmatrix}^{-1} = \begin{pmatrix} q/q & -h/q \\ 0 & e/q \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 20/32 \\ 1 & 0 & 20/32 & 0 \\ 0 & 0 & 0 & 1/32 \\ 0 & 0 & 1/32 & 0 \end{pmatrix},$$

откуда:

$$(m_1, m_2)\begin{pmatrix} e & h \\ 0 & q \end{pmatrix}^{-1} =$$

$$= (0,0,0,17)\begin{pmatrix} 0 & 1 & 0 & 20/32 \\ 1 & 0 & 20/32 & 0 \\ 0 & 0 & 0 & 1/32 \\ 0 & 0 & 1/32 & 0 \end{pmatrix} =$$

$$= (0,0,0,17) \approx (0,0,1,0),$$

в результате получается:

$$(s', t') = (0,0,1,0)\begin{pmatrix} 0 & 1 & 0 & -20 \\ 1 & 0 & -20 & 0 \\ 0 & 0 & 0 & 32 \\ 0 & 0 & 32 & 0 \end{pmatrix} = (0,0,0,32).$$

Полученная подпись имеет вид (s′, t') = (0, 0, 0, 32) и расстояние между подписью и сообщением больше, чем в предыдущем случае:

$$\|s' - m_1, t' - m_2\| =$$

$$\sqrt{(0-0)^2 + (0-0)^2 + (0-0)^2 + (17-32)^2} = 15,$$

поэтому теперь подпись не прошла проверку: 15 > $\mathcal{N}$ = 8,12. В первом примере расстояние между подписью и сообщением будет наименьшим, это значит что подпись в первом примере подписана на секретном ключе, и первая подпись прошла проверку [1, с. 6].

3 РЕАЛИЗАЦИЯ АЛГОРИТМА ЦИФРОВОЙ ПОДПИСИ НА БАЗЕ NTRU

В качестве среды разработки выбран программный пакет математического моделирования Maple 13, который в текущей версии поддерживает множество библиотек, в том числе и для работы с полиномами (polynomialTools), векторами и матрицами (VectorCalculus и LinearAlgebra), файлами (FileTools) и строковыми переменными (stringTools).

Для проверки корректности реализации принципов работы алгоритмов генерации параметров ЭЦП, её создания и верификации использован пример, рассмотренный в подразделе 2.3, в котором значения основных параметров выбраны малыми для понимания процесса.

3.1 Определение общесистемных параметров и генерация ключевой пары

Сначала определяются целые числа N и q. Целое число N в общем случае может быть любым и задаёт модуль усечённого кольца полиномов $X^N - 1$, а также определяет размерность самих полиномов таким образом, что их степень не превосходит N. Число q – ограничивает все коэффициенты полиномов по модулю q, что собственно и делает их «усечёнными» (рис. 3.1).

```
> q := 32; N := 2;
                    q := 32
                    N := 2
```

Рисунок 3.1 – Определение параметров q и N

При генерации секретного ключа сначала случайным образом вычисляются полиномы $f(x)$, $g(x)$ принадлежащие кольцу $R = Z[X]/(X^N - 1)$, которые могут иметь любые коэффициенты по модулю q. Для примера с

малыми значениями выбраны полиномы с нулевыми коэффициентами при старшей степени $N - 1 = 1$ (рис. 3.2).

```
> f := 0·x + 3 mod q; g := 0·x + 4 mod q;
                        f := 3
                        g := 4
```

Рисунок 3.2 – Определение полиномов *f(x)* и *g(x)*

Важным условием при выборе полинома f является наличие в кольце R обратного ему f^{-1}, без наличия которого невозможна генерация открытого ключа. Поэтому если в кольце не существует обратного для полинома f, то следует его отбросить и выбрать другое значение f. Для нахождения обратного полинома используется расширенный алгоритм Эвклида, которому соответствует функция для полиномов gcdex (рис. 3.3) [10].

```
> gcdex(f, x^N - 1, x, 'ff', 'm1');
                        1
> ff; ## обратный полиному f
                        1
                        3
```

Рисунок 3.3 – Нахождение f^{-1} с помощью функции gcdex

Для нахождения обратного f^{-1} на вход функции следует передать f, модуль кольца $X^N - 1$, главную переменную x и переменные с найденными коэффициентами при многочленах f, $X^N - 1$, одна из которых как раз и будет искомым значением f^{-1}. Функция по умолчанию возвращает НОД, который должен быть равен 1, либо выдаёт сообщение об отсутствии обратного для f.

Другой частью секретного ключа являются полиномы F и G которые являются отображением полиномов f и g на множество целых чисел. Это достигается при помощи нахождения результанта f и $X^N - 1$, и g и $X^N - 1$ соответственно. Результант для f и $X^N - 1$ определяется по формуле [3, с. 5]:

$$R_f = \prod_{i=1}^{N-1} f\left(x^i\right) mod\ \Phi\ \in \mathbb{Z}, \qquad (3.1)$$

где $\Phi(X) = \sum_{i=0}^{N-1} X^i \in R$. Аналогично вычисляется значение результанта для g и $X^N - 1$. Для вычисления результантов R_f и R_g используется соответствующая функция в Maple 13 (рис. 3.4) [10].

```
> Rf := resultant(f, x^N - 1, x); Rg := resultant(g, x^N - 1, x);
                              Rf := 9
                              Rg := 16
```

Рисунок 3.4 – Нахождение результантов R_f и R_g

Следует убедится что R_f и R_g взаимно простые, чтобы найти $\alpha, \beta \in \mathbb{Z}$, такие что $\alpha R_f + \beta R_g = 1$. Достигается это при помощи расширенного алгоритма Эвклида, реализованного функцией *igcdex*, которая возвращает наибольший общий делитель равный 1 в случае взаимной простоты R_f и R_g, а также α и β (рис. 3.5).

```
> igcdex(Rf, Rg,'alf','bet'); alf; bet;
                               1
                              -7
                               4
```

Рисунок 3.5 – Вычисление α и β с помощью функции *igcdex*

Затем схожим образом при помощи расширенного алгоритма Эвклида находятся значения ρ_f и ρ_g (рис. 3.6).

```
> igcdex(alf·f, bet·g,'rof','rog'); rof; rog;
                               1
                               3
                               4
```

Рисунок 3.6 - Вычисление ρ_f и ρ_g с помощью функции *igcdex*

Тогда искомые значения F и G находятся как $F = -q\beta\rho_g$ и $G = q\alpha\rho_f$ (рис 3.7).

```
> F2 := -q·bet·rog; G2 := q·alf·rof;
                    F2 := -512
                    G2 := -672
```

Рисунок 3.7 – Вычисление значений F и G

Значения полиномов F и G должны удовлетворять соотношению (3.2):

$$f * G - g * F = q. \tag{3.2}$$

Проверка соотношения (3.2) приведена на рисунке 3.8.

```
> qq2 := f·G - F·g;
                    qq2 := 32
```

Рисунок 3.8 – Получение значения параметра q

Таким образом полиномы секретного ключа F и G уже определены верно, тем не менее они всё ещё имеют слишком большие значения коэффициентов. Для того чтобы получить полиномы с минимизированными коэффициентами $F' = F + c * f$ и $G' = G + c * g$ вычисляется число c, округлённое до ближайшего целого:

$$c = \left| \frac{\bar{f} * F + \bar{g} * G}{f * \bar{f} + g * \bar{g}} \right|, \tag{3.3}$$

где полином $\bar{f}(x) = f(1/x) \bmod X^N - 1$, по аналогии также и с полиномом $\bar{g}$ [3, с. 15].

Для округления и взятия абсолютного значения используются встроенные функции round и abs соответственно [10, с. 27] (рис. 3.9).

$$> \quad c := \mathrm{abs}\left(\mathrm{round}\left(\frac{(f \cdot F + g \cdot G)}{f \cdot f + g \cdot g}\right)\right);$$

$$Fs1 := F + c \cdot f;\ Gs1 := G + c \cdot g;$$

$$c := 169$$

$$Fs1 := -5$$

$$Gs1 := 4$$

Рисунок 3.9 – Вычисление минимизированных F' и G'

Теперь можно перейти к генерации открытого ключа (рис. 3.10), который находится из соотношения:

$$h = f^{-1} * g \ mod \ q. \tag{3.4}$$

$$> \quad h := ff \cdot g \ \mathbf{mod}\ q; \#\ открытый\ ключ$$

$$h := 12$$

Рисунок 3.10 – Получение открытого ключа h

На данном этапе получены значения основных параметров и ключей, далее рассмотрим процедуру получения хеша из документа.

3.2 Получение хеш-сообщения

В алгоритме NTRU для получения ЦП подписываемого документа необходимо хеш-сообщение в виде вектора размерностью $2N$. Алгоритм, используемый для получения хеш-сообщения может быть любым, отвечающий современным стандартам безопасности и быстродействия. В данной работе используется алгоритм MD5 128 бит. Документ используемый для подписания приведён в приложении А.

Для того чтобы получить доступ к документу(файлу) в Maple нужно его сначала открыть, так программа сохраняет за собой право на чтение и запись файла, в то время как для других пользователей доступ к этому документу будет закрыт. Для открытия файла применяется функция *fopen*

(рис. 3.11), принимающая на вход путь к файлу, метод доступа на чтение *READ*, тип представления документа в виде потока байт *BINARY* [10, с. 103].

```
> doc
      := fopen (
      "D:\\Documents\\Универ\\Магистр\\NtruSign\\программы\\D.
      docx", READ, BINARY);

                        doc := 0
```

Рисунок 3.11 - Открытие файла функцией *fopen*

Функция возвращает дескриптор файла равный 0, означает что файл успешно открыт, в противном случае выдаст ошибку, например если файл уже был открыт ранее.

Чтобы получить хеш документа, нужно сохранить его в переменную в виде массива байт при помощи функции *readbytes*. Функция принимает на вход дескриптор файла (путь к файлу) и размер читаемых байт *n*. Размер файла в байтах получается при помощи функции *Size*. После получения массива байт документа его желательно закрыть, используя функцию *fclose* (рис. 3.12).

```
> n := FileTools:-Size(doc);# Size in Bytes
                        n := 13289
> DB := readbytes(doc, n):
> fclose(doc);
```

Рисунок 3.12 – Получение массива байт файла документа

На чтение файла и запись его в массив понадобится некоторое время, поэтому предпочтительно работать с файлами не более 20 кБ [10, с. 104].

После того как файл сохранён в массив, то на этом этапе следует осуществить конкатенацию исходного файла и параметра r если $r \neq 0$, который представляется в виде битовой строки. Таким образом удастся получить подходящее значение хеша файла. Поскольку в 1 цикле формирования подписи $r = 0$, то этот этап пропущен.

Для получения непосредственно хеша используется встроенная хеш-функция MD5 *Hash* (рис. 3.13), которая принимает строку на вход и на выходе 128- битная последовательность (16-ричное число) в виде строки [10].

```
> str := convert(DB, string):
> Hsh128 := Hash(str);
Hsh128 := "6cf25f395a8b3c15bc1f285347f7ff3
```

Рисунок 3.13 – Получение хеша файла документа

Теперь необходимо преобразовать 128-битную хеш-строку в вектор сообщения с размерностью 2*N*.

Преобразуем хеш из строки в десятичное число функцией *convert*, где первым аргументом задаётся хеш-строка, вторым десятичная система исчисления *decimal*, третьим исходная шестнадцатиричная система *hex*. Конвертируем хеш в десятичном виде обратно в строку, после используя функции *seq* и *parse* получим список из всех цифр десятичного числа, предварительно вычислив длину списка функцией *length* (рис. 3.14) [10, с. 91].

```
> H2Dec := convert(Hsh128, decimal, hex):
  lenH := length(H2Dec);
                    lenH := 39
> HDec2S := convert(H2Dec, string);
  HDec2S := "144815090757475928097246745551516991
> H2Ar := [seq(parse(HDec2S[i]), i = 1..lenH)];
H2Ar := [1, 4, 4, 8, 1, 5, 0, 9, 0, 7, 5, 7, 4, 7, 5, 9, 2, 8, 0, 9, 7, 2, 4, 6, 7, 4,
  5, 5, 5, 1, 5, 1, 6, 9, 9, 1, 2, 9, 1]
```

Рисунок 3.14 – Преобразование хеша в массив десятичных цифр

Далее выполняется преобразование списка в строчный вектор размерности 2*N* или более функцией *Vector*, при необходимости вектор обрезается с помощью функции взятия подвектора *SubVector* пределах от 1

до 2*N*. Таким образом получается вектор хеш-сообщения, необходимый для формирования и проверки подписи (рис. 3.15).

```
> imax := ceil(2·N / lenH):
> mV := Vector[row]([seq(H2Ar, i = 1..imax)]):
> m := SubVector(mV, 1..2·N);
                    m := [1 4 4 8]
```

Рисунок 3.15 – Получение вектора хеш-сообщения размерностью 2*N*

3.3 Процедура формирования подписи

Чтобы получить подпись сначала генерируется матрица секретного ключа (1.8). Ниже на рисунке 3.16 приведён код программы, который циклически для каждого полинома *f*, *g*, *F*, *G* строит матрицы циклическим сдвигом этих полиномов размерностью *N*x*N*.

```
P[1] := f: P[2] := g: P[3] := F: P[4] := G:

M := Array(1..4);

for i from 1 to 4 do

coef[i] := CoefficientVector(P[i], x);
V[i] := Vector[row](N);
dim[i] := Dimension(coef[i]);
V[i][N − dim[i] + 1..−1] := coef[i]:
#M_i := Matrix(N);
M0 := Matrix(N);

for j from 1 to N do

M0[j, 1..−1] := V[i];
V[i] := SubVector(V[i], 2..−1);
end do:
M_i := M0;
end do:
```

Рисунок 3.16 – Код программы для построения матриц из полиномов

Для построения матриц в Maple 13 используется функция-конструктор *Matrix* [10, с. 91]. Матрица секретного ключа формируется из

матриц, полученных циклическим сдвигом влево полиномов f, g, F, G (рис. 3.17).

```
> MSK2 := Matrix([[M1, M2], [M3, M4]]);
    # Матрица секретного ключа
```

$$MSK2 := \begin{bmatrix} 0 & 3 & 0 & 4 \\ 3 & 0 & 4 & 0 \\ 0 & -5 & 0 & 4 \\ -5 & 0 & 4 & 0 \end{bmatrix}$$

Рисунок 3.17 – Построение матрицы секретного ключа

Далее согласно подразделу 2.3 выполняется инверсия матрицы секретного ключа (рис. 3.18).

```
> MSKj2 := MatrixInverse(MSK2);
    # Инверсия матрицы секретного ключа
```

$$MSKj2 := \begin{bmatrix} 0 & \frac{1}{8} & 0 & -\frac{1}{8} \\ \frac{1}{8} & 0 & -\frac{1}{8} & 0 \\ 0 & \frac{5}{32} & 0 & \frac{3}{32} \\ \frac{5}{32} & 0 & \frac{3}{32} & 0 \end{bmatrix}$$

Рисунок 3.18 – Получение инверсии матрицы секретного ключа

Производится умножение вектора хеш-сообщения на обратную матрицу функцией *VectorMatrixMultiply*. На выходе имеем вектор, коэффициенты которого округляются до ближайшего целого функцией *round*.

Затем наконец получим подпись в виде вектора коэффициентов полиномов (r, s) путём умножения полученного вектора на матрицу секретного ключа (рис. 3.19).

```
> mMSKj2 := VectorMatrixMultiply(m, MSKj2);
```

$$mMSKj2 := \begin{bmatrix} \frac{85}{32} & 0 & \frac{51}{32} & 0 \end{bmatrix}$$

```
> mMskj := round(mMSKj2);# округление до ближ целого
```

$$mMskj := \begin{bmatrix} 3 & 0 & 2 & 0 \end{bmatrix}$$

```
> ST := mMskj . MSK2;#Подпись
```

$$ST := \begin{bmatrix} 0 & -1 & 0 & 20 \end{bmatrix}$$

Рисунок 3.19 – Получение вектора подписи документа

Можно разбить вектор подписи на подвектора s, t и преобразовать в полиномы функцией *FromCoefficientList* (рис. 3.20) [10].

```
> S := SubVector(ST, 1..N);
```

$$S := \begin{bmatrix} 0 & -1 \end{bmatrix}$$

```
> T := SubVector(ST, N + 1..N·2);
```

$$T := \begin{bmatrix} 0 & 20 \end{bmatrix}$$

```
> s := FromCoefficientVector(S, x, termorder = reverse);
  # Из векторра в полином
```

$$s := -1$$

Рисунок 3.20 – Преобразование вектора подписи в полиномы s и t

Проверим что найденная подпись принадлежит решётке, в случае принадлежности $s = s' \bmod q$ согласно формулы (3.5) [1, с. 6]:

$$t' = s * h \; mod \; q. \tag{3.5}$$

Результат проверки подписи представлен на рисунке 3.21.

```
> # проверка что подпись принадл решётке
> tt := s·h mod q;##Проверка успешная tt = t
```

$$tt := 20$$

Рисунок 3.21 – Проверка подписи на принадлежность решётке

Поскольку $t = t'$, то подпись найдена верно и принадлежит решётке.

3.4 Процедура проверки подписи

В процессе формирования подписи в NTRUSign следует выполнить проверку подписи на стороне подписывающего, поскольку не все сгенерированные подписи могут быть проверены в виду специфики работы алгоритма.

Для проверки подписи сначала следует задаться значением допустимой нормальной границы (normBound), в данном случае взятой из примера в пункте 1.7 $\mathcal{N} = 8{,}12$. Этот общесистемный параметр определяет максимальное расстояние между точками – норму полиномов подписи и хеш-сообщения в пределах которой она должна находится.

Подписывающий уже знает хеш-сообщение, в то время как проверяющему необходимо будет его получить как описано в 3.2.

Проверяющий также находит t' используя соотношение (3.5), затем находится норма полиномов b. Для вычисления нормы (2.6) использована функция нахождения нормы для векторов *Norm*, соответственно на вход подаётся вектор ($s - m_1$, $t' - m_2$), а в качестве второго аргумента указывается ‘2’ или *Euclidean*, что подразумевает вычисление центрированной Эвклидовой нормы, вычисленное значение Эвклидовой нормы представим в виде числа с плавающей точкой *float* [10, с. 73].

Параметр нормированной границы $\mathcal{N}$ выбирается достаточно малым, чтобы существовала только одна такая пара точек, которые находятся очень близко в пределах нормы. Однако величина $\mathcal{N}$ в свою очередь, зависит от размерности кольца N, доверительной вероятности попадания нормы в пределы границы и прочих параметров, поэтому процесс получения $\mathcal{N}$ в данной работе не рассматривается.

Процесс проверки подписи представлен на рисунке 3.22.

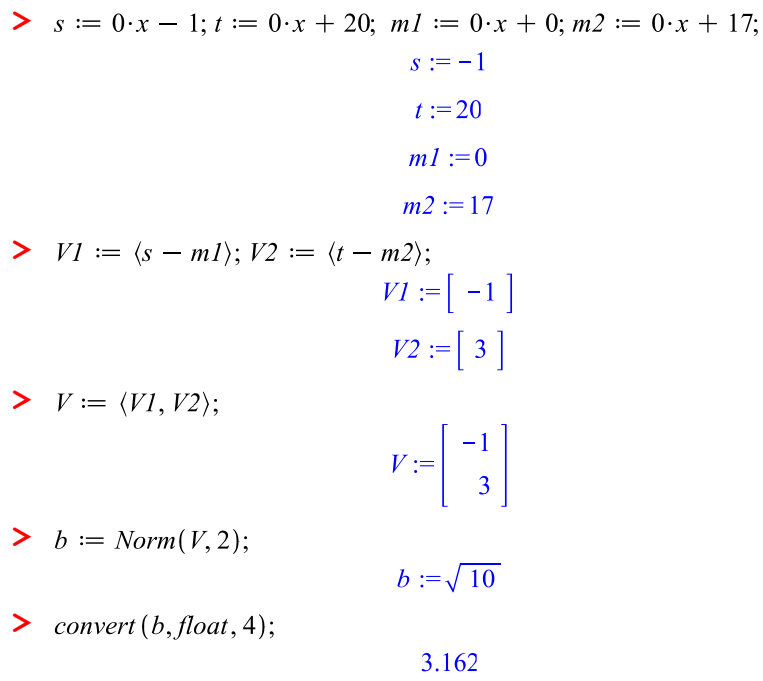

Рисунок 3.22 – Процедура проверки подписи

Как видно, значение b = 3,162 < 8,12, значит подпись действительна и прошла проверку. Если значение b было бы большим чем $\mathcal{N}$, то подпись бы соответственно проверку не прошла. Следует заметить, что в том случае, если такую проверку подпись не проходит на стороне у подписывающего, то такая подпись отбрасывается, и к старому значению r добавляется 1, после чего подпись проверяется заново.

3.5 Моделирование алгоритма ЭЦП NTRU для рекомендуемых значений параметров

В этом разделе рассматриваются этапы выбора общих параметров, генерации ключей, формирования и проверки подписи для больших значений, соответствующих необходимому уровню безопасности подписи. В процессе данной работы были реализованы процедуры алгоритма ЭЦП, описанные в предыдущих подразделах, которые выполнены в виде отдельных Maple-функций и объединены в один пакет в виде файла

«NTRU.m». Программный код процедур также предоставлен в приложении Б, документ использованный для подписания представлен в приложении А.

Для моделирования работы алгоритма были выбраны рекомендуемые общесистемные параметры: модуль усечённого кольца полиномов $N = 127$; модуль коэффициентов полиномов кольца $q = 256$; допустимая нормальная граница $\mathcal{N} = 122.94$. Все указанные значения взяты из [11, с. 18] и рекомендованы для обеспечения уровня безопасности $k = 80$ бит.

Основные процедуры пакета «NTRU.m» в качестве параметра “Gparams” принимают путь к файлу с указанными значениями общесистемных параметров q, N, $\mathcal{N}$. Поэтому сначала нужно сохранить параметры при помощи команды «save» в такой файл, например с именем «GParams.m» (рис. 3.22). Далее необходимо подключить файл с процедурами алгоритма «NTRU.m» при помощи команды «read», а также для удобства указать дескрипторы к файлам документа «D.docx» и «Keys.txt», которые понадобятся для получения хеш-сообщения (рис. 3.23) [10, с. 103].

```
> N := 127; q := 256; Nor := 122.94;
                        N := 127
                        q := 256
                        Nor := 122.94
> save N, q, Nor, "GParams.m";
```

Рисунок 3.22 – Запись параметров в файл «GParams.m»

```
> read "NTRU.m";
> doc
      := "D:\\Documents\\Универ\\Магистр\\NtruSign\\программы\\D
      docx"
   doc := "D:\Documents\Универ\Магистр\NtruSign\программы\D.doc
> doc2
      := "D:\\Documents\\Универ\\Магистр\\NtruSign\\программы\\K
      ys.txt"
   doc2 :=
      "D:\Documents\Универ\Магистр\NtruSign\программы\Keys.txt"
```

Рисунок 3.23 – Подключение файла с процедурами «NTRU.m» и файлов электронных документов «doc», «doc2»

Теперь сгенерируем две пары секретных и открытых ключей с помощью процедуры *KeyGen*(*i*), которая принимает на вход количество пар ключей и возвращает последнюю сгенерированную пару.

Полиномы секретного ключа *f*, *g* и *F* представлены на рисунке 3.24, полином секретного ключа *G* представлен на рисунке 3.25, полином открытого ключа *h* представлен на рисунке 3.26.

```
> ## Генерация ключа
> KeyGen(2);
```

$$f =, 4 - 184x^{103} - 45x^{82} + 214x^{63} + 248x^{35} - 32x^{17}$$

$$g =, 233x^{106} - 60x^{80} - 99x^{71} - 57x^{68} - 40x^{49} - 71x^{11}$$

$$\begin{aligned} F =, {} & 0.00722x - 0.429x^{94} - 0.494x^{82} + 0.280x^{63} - 0.890x^{35} \\ & + 0.0935x^{17} + 0.278x^{80} + 40.8x^{71} - 0.150x^{68} - 0.859x^{49} \\ & + 0.0572x^{11} + 0.0107x^{102} + 0.507x^{87} + 0.0441x^{84} \\ & - 0.848x^{76} - 0.451x^{73} - 0.554x^{88} + 0.126x^{34} - 0.567x^{31} \\ & + 0.0146x^{16} - 0.0129x^{8} + 0.00280x^{5} - 0.437x^{58} \\ & + 0.00399x^{26} + 0.136x^{23} + 0.141x^{29} - 0.0172x^{3} + 0.650x^{75} \\ & - 0.0227x^{10} - 2.04x^{55} - 0.615x^{41} - 0.00277x^{9} - 0.442x^{28} \\ & - 0.0335x^{62} - 0.836x^{59} - 0.226x^{44} + 0.364x^{36} - 0.124x^{33} \\ & - 0.0550x^{18} - 194.x^{52} + 0.105x^{20} + 0.766x^{51} + 0.706x^{46} \\ & - 0.642x^{43} + 0.118x^{64} - 0.629x^{56} - 0.0462x^{53} + 0.716x^{69} \\ & + 0.811x^{54} + 1.17x^{48} - 0.108x^{40} - 0.911x^{61} - 0.482x^{50} \\ & - 1.08x^{47} + 0.996x^{42} + 0.752x^{39} + 0.833x^{60} - 0.229x^{65} \\ & + 0.0700x^{14} - 1.51x^{45} - 0.120x^{37} + 0.0173x^{13} - 0.799x^{57} \\ & - 0.0185x^{12} - 0.693x^{70} - 0.938x^{38} - 0.0131x^{67} + 0.0656x^{32} \\ & + 0.0189x^{7} - 0.150x^{30} + 28.8x^{6} + 0.0115x^{4} - 1.85x^{27} \\ & - 0.00612x^{2} + 0.119x^{25} - 224.x^{24} - 0.0320x^{22} - 0.107x^{21} \\ & + 0.0621x^{19} - 0.00915x^{15} + 0.244x^{72} + 0.0171x^{93} \\ & + 0.479x^{101} + 0.406x^{77} - 0.413x^{91} + 0.386x^{83} + 0.177x^{81} \\ & + 0.536x^{98} + 166.x^{92} + 0.561x^{96} + 0.253x^{74} - 1.05x^{97} \\ & - 0.422x^{100} + 0.464x^{99} + 1.41x^{95} - 0.456x^{86} + 0.451x^{66} \\ & + 1.58x^{85} + 0.201x^{90} + 0.610x^{78} + 0.432x^{89} - 0.698x^{79} \\ & - 0.0135 \end{aligned}$$

Рисунок 3.24 – Сгенерированные полиномы секретного ключа *f*, *g* и *F*

$$
\begin{aligned}
G=, & 0.523x^{94} + 0.535x^{103} + 0.753x^{82} + 0.400x^{63} + 0.368x^{35} \\
& + 0.216x^{17} - 0.348x^{80} + 0.184x^{71} - 0.213x^{68} + 0.000172x^{49} \\
& + 0.239x^{11} - 0.587x^{102} - 0.0558x^{87} - 0.221x^{84} + 0.589x^{76} \\
& + 0.932x^{73} - 2.01x^{88} - 0.686x^{34} + 0.564x^{31} - 0.0497x^{16} \\
& + 0.484x^{58} + 0.162x^{26} + 0.329x^{23} - 0.0486x^{29} - 0.129x^{75} \\
& + 0.641x^{55} + 0.203x^{41} + 0.251x^{28} + 0.386x^{62} - 0.462x^{59} \\
& + 0.140x^{44} - 0.267x^{36} + 0.170x^{33} - 0.335x^{18} + 0.617x^{52} \\
& + 0.0491x^{20} - 0.426x^{51} + 0.607x^{46} + 0.135x^{43} + 0.906x^{64} \\
& - 0.615x^{56} + 0.832x^{53} + 54.1x^{69} - 0.309x^{54} - 0.0853x^{48} \\
& + 0.132x^{40} - 0.222x^{61} + 0.657x^{50} - 0.378x^{47} + 0.136x^{42} \\
& - 0.266x^{39} + 92.7x^{60} - 0.731x^{65} + 0.305x^{14} - 0.234x^{45} \\
& + 0.322x^{37} + 0.109x^{13} + 51.6x^{57} - 0.128x^{12} - 0.144x^{70} \\
& + 36.0x^{38} + 0.660x^{67} + 0.263x^{32} - 0.234x^{30} - 0.259x^{27} \\
& - 1.24x^{25} - 0.307x^{24} - 1.01x^{22} + 0.403x^{21} + 0.230x^{19} \\
& - 0.205x^{15} - 0.383x^{72} - 0.255x^{93} - 0.678x^{101} - 0.0205x^{77} \\
& + 0.701x^{91} - 0.204x^{83} - 0.629x^{81} - 1.79x^{98} - 0.547x^{92} \\
& - 0.0217x^{96} - 0.464x^{74} + 0.544x^{97} + 1.33x^{100} - 0.710x^{99} \\
& - 211.x^{95} - 0.489x^{86} + 0.236x^{66} + 0.626x^{85} - 0.642x^{90} \\
& - 0.493x^{78} + 0.577x^{89} + 0.751x^{79} - 0.0135x^{105} - 0.607x^{104} \\
& + 64.
\end{aligned}
$$

Рисунок 3.25 - Полином секретного ключа G

$$
\begin{aligned}
h=, & 224 + 128x + 166x^{94} + 128x^{103} + 208x^{82} + 208x^{63} + 128x^{35} \\
& + 232x^{106} + 128x^{80} + 128x^{71} + 160x^{68} + 192x^{49} + 48x^{87} \\
& + 208x^{76} + 96x^{88} + 172x^{34} + 180x^{31} + 128x^{8} + 106x^{5} \\
& + 160x^{58} + 32x^{26} + 84x^{75} + 152x^{10} + 192x^{55} + 128x^{41} \\
& + 8x^{9} + 96x^{28} + 200x^{59} + 128x^{44} + 128x^{36} + 128x^{33} \\
& + 44x^{18} + 64x^{52} + 160x^{51} + 112x^{43} + 192x^{64} + 219x^{56} \\
& + 228x^{69} + 64x^{54} + 96x^{48} + 240x^{40} + 128x^{61} + 112x^{50} \\
& + 96x^{47} + 160x^{39} + 96x^{60} + 184x^{45} + 154x^{37} + 128x^{13} \\
& + 96x^{57} + 16x^{12} + 128x^{70} + 192x^{32} + 104x^{7} + 192x^{2} \\
& + 192x^{25} + 19x^{24} + 32x^{21} + 208x^{19} + 208x^{15} + 188x^{101} \\
& + 56x^{77} + 32x^{98} + 128x^{92} + 224x^{96} + 242x^{97} + 128x^{100} \\
& + 32x^{99} + 96x^{95} + 32x^{86} + 152x^{66} + 160x^{85} + 64x^{90} \\
& + 28x^{78} + 192x^{89} + 240x^{79} + 136x^{104} + 32x^{115} + 232x^{126} \\
& + 64x^{121} + 192x^{122} + 128x^{110} + 96x^{117} + 192x^{111} \\
& + 192x^{124} + 240x^{120} + 192x^{118} + 176x^{107} + 207x^{116} \\
& + 201x^{113} + 16x^{123} + 44x^{125} + 240x^{114}
\end{aligned}
$$

Риунок 3.26 – Полином открытого ключа h

В основе процедуры генерации ключей лежит функция *randpoly*, позволяющая генерировать случайные полиномы f и g, полиномы F и G вычисляются аналогично алгоритму, описанному в подразделе 2.1. Стоит заметить, что процедура генерации ключей потребляет значительное количество оперативной памяти и занимает продолжительное время, поэтому в данной реализации максимальное число сгенерированных пар ограничено до 10.

Полученные значения ключей удобно хранить в файлах, поэтому процедура записывает все сгенерированные секретные ключи в файлы «KeysBig_1.m» и «KeysBig_2.m», и соответственно открытые ключи в файлы «PublicBig1.m» и «PublicBig2.m».

Теперь можно выполнить подписание документа «D.docx» вызовом процедуры *SignDoc(doc, secret, public)*. В качестве аргументов функция принимает путь к файлам документа для подписи, а также пути к файлам открытого и секретного ключа (рис. 3.27) и (рис. 3.28).

```
> SignDoc(doc, "KeysBig_1.m", "PublicBig1.m");
```

$$
\begin{aligned}
s=,\ & 204 + 156x + 91x^{103} + 169x^{82} + 26x^{63} + 160x^{35} + 104x^{17} \\
& + 198x^{106} + 228x^{80} + 86x^{71} + 40x^{68} + 172x^{49} + 8x^{11} \\
& + 92x^{47} + 216x^{102} + 89x^{83} + 244x^{75} + 60x^{16} + 64x^{13} \\
& + 164x^{7} + 226x^{95} + 4x^{32} + 188x^{30} + 149x^{84} + 12x^{76} \\
& + 90x^{73} + 235x^{88} + 216x^{31} + 220x^{8} + 236x^{58} + 100x^{26} \\
& + 112x^{23} + 76x^{29} + 180x^{10} + 80x^{55} + 24x^{41} + 76x^{9} \\
& + 128x^{28} + 40x^{62} + 24x^{59} + 252x^{44} + 248x^{36} + 164x^{33} \\
& + 72x^{18} + 184x^{52} + 176x^{20} + 12x^{51} + 52x^{46} + 52x^{43} \\
& + 54x^{64} + 24x^{56} + 76x^{53} + 182x^{69} + 84x^{54} + 192x^{48} \\
& + 124x^{40} + 56x^{61} + 8x^{42} + 176x^{39} + 16x^{60} + 154x^{65} \\
& + 240x^{14} + 160x^{45} + 100x^{37} + 56x^{57} + 248x^{12} + 234x^{70} \\
& + 28x^{38} + 70x^{67} + 16x^{27} + 68x^{25} + 128x^{24} + 108x^{22} \\
& + 60x^{21} + 200x^{19} + 148x^{15} + 62x^{72} + 232x^{93} + 180x^{101} \\
& + 104x^{77} + 13x^{91} + 192x^{81} + 249x^{98} + 161x^{92} + 230x^{96} \\
& + 96x^{74} + 23x^{97} + 100x^{100} + 38x^{99} + 17x^{86} + 56x^{66} \\
& + 154x^{85} + 177x^{90} + 106x^{78} + 193x^{89} + 14x^{79} + 230x^{94} \\
& + 204x^{6} + 204x^{4} + 52x^{2} + 60x^{105} + 141x^{104} + 88x^{50} \\
& + 137x^{107} + 95x^{108} + 162x^{109} + 229x^{114} + 195x^{115} \\
& + 106x^{116} + 128x^{117} + 26x^{110} + 81x^{111} + 75x^{112} + 74x^{113} \\
& + 153x^{122} + 48x^{123} + 148x^{124} + 201x^{125} + 84x^{118} \\
& + 205x^{119} + 209x^{120} + 204x^{121} + 131x^{126}
\end{aligned}
$$

Рисунок 3.27 – Вызов процедуры *SignDoc* и вывод *s*

$$t=, 207 + 123x + 27x^{103} + 142x^{82} + 12x^{63} + 60x^{35} + 98x^{17} + 200x^{106} + 29x^{80} + 103x^{71} + 141x^{68} + 187x^{49} + 110x^{11} + 152x^{47} + 20x^{102} + 181x^{83} + 114x^{75} + 103x^{16} + 119x^{13} + 58x^{7} + 150x^{95} + 71x^{32} + 190x^{30} + 109x^{87} + 201x^{84} + 153x^{76} + 201x^{73} + 242x^{88} + 252x^{34} + 128x^{31} + 111x^{8} + 24x^{58} + 129x^{26} + 15x^{23} + 6x^{29} + 186x^{10} + 229x^{55} + 177x^{41} + 233x^{9} + 234x^{28} + 176x^{62} + 72x^{59} + 229x^{44} + 253x^{36} + 3x^{33} + 141x^{18} + 248x^{52} + 27x^{20} + 239x^{51} + 8x^{46} + 67x^{43} + 116x^{64} + 44x^{53} + 229x^{69} + 71x^{54} + 255x^{48} + 35x^{40} + 236x^{61} + 82x^{42} + 226x^{39} + 80x^{60} + 120x^{65} + 16x^{14} + 94x^{45} + 121x^{37} + 140x^{57} + 116x^{12} + 239x^{70} + 138x^{38} + 128x^{67} + 123x^{27} + 248x^{25} + 248x^{24} + 169x^{22} + 49x^{21} + 114x^{19} + 191x^{15} + 28x^{72} + 22x^{93} + 248x^{101} + 44x^{77} + 88x^{91} + 69x^{81} + 185x^{98} + 243x^{92} + 23x^{96} + 65x^{74} + 170x^{97} + 252x^{100} + 232x^{99} + 115x^{86} + 192x^{66} + 198x^{85} + 235x^{90} + 92x^{78} + 251x^{89} + 77x^{79} + 227x^{94} + x^{5} + 248x^{3} + 30x^{6} + 249x^{4} + 16x^{2} + 51x^{105} + 25x^{104} + 120x^{50} + 55x^{107} + 193x^{108} + 23x^{109} + 234x^{114} + 160x^{115} + 109x^{116} + 250x^{117} + 163x^{110} + 44x^{111} + 194x^{112} + 153x^{113} + 163x^{122} + 122x^{123} + 10x^{124} + 50x^{125} + 34x^{118} + 16x^{119} + 48x^{120} + 61x^{121} + 244x^{126}$$

$$r=, 0$$

Рисунок 3.28 – Вывод компонентов подписи *t*, *r* функцией *SignDoc*

Процедура возвращает значения полученной подписи *s*, *t*, *r* для документа «D.docx», и записывает их в файл «Signature.m».

Выполним проверку полученной подписи с помощью процедуры со стороны проверяющего *VerifyDoc(doc, sig, public)*, принимающая на вход путь к подписанному документу, путь к файлу подписи и открытого ключа (рис. 3.29).

```
> VerifyDoc (doc, "Signature1.m","PublicBig1.m");
        "Signature Valid","b=", 121.5252913,"<", "Norm=", 122.94
```

Рисунок 3.29 – Проверка подписи документа функцией *VerifyDoc*

В данном случае процедура возвращает сообщение об успешной проверке подписи «Signature Valid» и значение вычисленной нормы векторов подписи и сообщения b, которое меньше допустимого значения нормы $\mathcal{N}$. Таким образом подписание документа выполнено успешно.

Рассмотрим случаи, при котором подпись проверку не пройдёт (рис. 3.30), например если в качестве параметров процедуры проверки подписи *VerifyDoc* передать другое значение открытого ключа или другой документ для верификации.

```
> VerifyDoc(doc, "Signature1.m", "PublicBig2.m");
        "Signature Invalid;", "b=", 123.4063014, ">", "Norm=", 122.94
> VerifyDoc(doc2, "Signature1.m", "PublicBig1.m");
        "Signature Invalid;", "b=", 127.3262015, ">", "Norm=", 122.94
```

Рисунок 3.30 – Проверка подписи документа не пройдена

В этом случае процедура *VerifyDoc* возвращает сообщение о том что подпись для данного документа проверку не прошла, а значение вычисленной нормы векторов b выше допустимого значения $\mathcal{N}$.

Процедуры создания и проверки подписи *SignDoc* и *VerifyDoc* для получения хеш-сообщения из файла подписываемого документа используют процедуру *Hmesg*, которая принимает на вход дескриптор подписываемого документа и размерность кольца N, и может быть вызвана самостоятельно (рис. 3.31).

```
> Hmesg(doc, 127);
        [ 1 .. 254  Vector_row
          Data Type: anything
          Storage: rectangular
          Order: Fortran_order ]
```

Рисунок 3.31 – Вызов функции получения хеш-сообщения документа

Процедура возвращает вектор хеш-сообщения m документа, имеющий размерность $2N$ согласно алгоритму рассмотренному в подразделе 3.2.

4. АНАЛИЗ И ОБОСНОВАНИЕ ПРИМЕНЕНИЯ КРИПТОСИСТЕМ НА БАЗЕ NTRU

Электронные цифровые подписи (ЭЦП) на сегодняшний день играют довольно важную роль среди современного информационного потока, поскольку являются неотъемлемыми частями электронных документов. Благодаря возложенным на них функциям проверки отсутствия искажений в электронном документе с момента формирования подписи, принадлежности подписи владельцу, а также неотказуемость как подтверждение факта подписания электронного документа в случае его успешной проверки. Квалифицированная электронная цифровая подпись нашла свое применение в случаях необходимости осуществления гражданско-правовых сделок, оказании государственных и муниципальных услуг и прочих юридически значимых действий.

На данный момент уже созданы и широко применяются цифровые подписи, которые основываются на преобразованиях в кольцах (RSA системы), конечных полях (DSA системы) и эллиптических кривых. Кроме этого практически разработаны и проходят проверку цифровые подписи на основе гиперэллиптических кривых и спаривания точек эллиптических кривых. Но в связи с ростом производительности крипто-аналитических систем, а также возникновением новых тенденций в области дальнейшего развития вычислительной техники криптографическая стойкость сегодняшних алгоритмов вызывает сомнение. Для повышения стойкости разработчики систем постоянно увеличивают размеры общесистемных параметров для этих алгоритмов. Особенно эта проблема остро возникает в связи с разработками нейрокомпьютеров, в которых параллельно будут выполняться не только потоки, но и отдельные фрагменты программы (мелкозернистый параллелизм). Еще большую опасность несёт возможное создание квантовых систем криптоанализа на основе квантового компьютера, поскольку благодаря ему становится возможной реализация алгоритма Шора,

который позволяет решать задачи факторизации и дискретного логарифмирования за приемлемое время.

В то же время существует необходимость реализации электронных цифровых подписей в реальном времени со скоростью десятков и сотен мб/с. Поэтому важной задачей является поиск криптопреобразований, которые смогли бы прийти на смену существующим, иметь улучшенную криптостойкость и быстродействие. Решением данной проблемы может стать создание алгоритмов цифровой подписи, основанных на преобразованиях в кольцах усечённых полиномов (NTRU) [12].

4.1 Криптосистема NTRU, современное состояние, стандарты и их применение

Со времени изобретения NTRUEncrypt в 1996 году было предложено несколько схем подписи. Первую версию NTRU подписи – NSS (NTRU signature scheme) представили на сессии Crypto 2000 года. Но спустя некоторое время в ней была найдена уязвимость. В своей презентации на Eurocrypt авторы NSS пересмотрели эту схему подписи и новая схема получила название R-NSS (Revised NTRU signature scheme). Все предыдущие версии подписи NTRU имеют такую же самую защищённость при тех же самых параметрах относительно: атак грубой силы, простых возведений в решётках и комбинаторной атаки [1, с. 1].

NTRUSign была разработана между 2001 и 2003 годами создателями NTRUEncrypt вместе с Н. Хоугревом-Грехемом и В. Уайтом. Исследования криптосистемы NTRU является достаточно актуальным для криптографии из за того, что она, как уверяют её авторы [14] остаётся устойчивой к атакам при помощи квантовых компьютеров. Подпись была стандартизирована EESS1 version 2, аспекты защищённости NTRUSign рассматриваются более детально в IEEE P1363.1 IEEE P1363.1 [8] “Проекте Стандарта

Спецификации для криптографии с открытым ключом, техник, которые основываются на трудности задач алгебры решёток.”

NTRU принадлежит к новому классу альтернативных схем открытого ключа – криптографии на основе решёток, которая сама по себе представляет собой активную область для исследований. Существует несколько сложных математических задач, которые могут быть использованы для построения криптосистемы на базе решёток, самой популярной из них является проблема нахождения кратчайшего вектора. Это направление включает в себя такие криптографические системы, как GGH и NTRU. Согласно работам Микианцио и Регева [15], криптографию на основе решёток можно разделить на две категории: практические криптосистемы, такие как NTRU, и теоретические, такие как матрицы на основе обучения с ошибками (Learning with Errors, LWE), которые имеют достаточно сильные доказательства безопасности, но используют ключи, которые являются слишком большими для общего использования. Также в последние годы ведутся исследования в области объединения этих двух категорий и создания нового класса криптографических систем на основе решёток [1, с. 1].

Таким образом, криптографические конструкции на основе решёток имеют большие перспективы для своего использования в сфере информационной безопасности, так криптосистема NTRU шифрования уже стандартизирована (стандарты ANSI X9.98, IEEE 1363.1) и может использоваться для предоставления услуги конфиденциальности. Для Украины также является важным развитие этого направления, учитывая то, что такая криптосистема устойчива к квантовому криптоанализу.

4.2 Анализ стойкости криптопреобразований в кольцах усечённых полиномов

В данном подразделе рассматривается вычислительная сложность функций генерации и проверки цифровых подписей, а также зависимость

параметров алгоритмов от необходимого уровня безопасности. Полученные результаты сравниваются с аналогичными результатами для формирования и проверки ЦП для RSA и ECDSA алгоритмов [16, 14].

Как уже было отмечено, криптостойкость подписей в кольцах усечённых полиномов, в частности NTRUSign, основывается на сложности решения математической задачи нахождения кратчайшего вектора решётки CVP. Ниже в таблице 4.1 приведены сложности возможных различных атак на NTRUSign:

Таблица 4.1 – Сложность для различных методов атак на NTRUSign

№	Метод	Сложность метода
1.	Нахождение кратчайшего вектора при помощи алгоритма LLL.	$O(N^3\beta^{\beta/3})$ или $O(N^3(k/6)^{k/4})$
2.	Атака грубая сила.	$O\left(\frac{N!}{df!\,(N-df)!}\cdot\frac{(N-df)!}{(df-1)!\cdot(N-2df+1)!}\right)$
3.	Комбинаторная атака на ЕЦП. Количество шагов алгоритма (подбор полиномов f).	$O\left(\frac{C_{N/2}^{df/2}}{\sqrt{N}}\right)$
4.	Атака подделки подписи. Вероятность успешного осуществления	P (combinational forgery) = $\sqrt{\frac{\pi^{\frac{N-1}{2}}}{q^{N-1}\left(\frac{N-1}{2}\right)!}\left(\frac{N}{\beta}\right)^{N-1}} < 2^{-k}$

При выборе параметров авторы исходили из криптостойкости, которая обеспечивается при применении соответствующих алгоритмов. В [11] определены зависимости уровня безопасности (эквивалентной длины ключа *k*, который необходимо найти путём полного перебора) от параметров цифровой подписи для алгоритмов RSA, ECC с простым и двоичными

полями и NTRU с количеством пертурбаций 0 (преобразований направленных на улучшение защиты от подделки подписи при большом количестве перехваченных ЦП), в случае одноразового подписания.

Данные о зависимости параметров алгоритмов от необходимого уровня безопасности представлены в таблице 4.2.

Таблица 4.2 – Зависимость параметров алгоритмов ЭЦП от необходимого уровня безопасности

Уровень безопасности, k	80	112	128	192	256
Длина полиномов секретного ключа NTRU	314	394	446	653	743
Двоичная длина модуля n преобразований RSA	1024	2048	3072	7680	15360
Двоичная длина простого числа p для базового поля GF(p) при реализации ECC	192	224	256	384	521
Степень расширения m для базового поля GF (2^m) при реализации ECC	163	233	283	409	571

Основным методом криптоанализа задач, построенных на нахождении наименьшего вектора, является LLL подобные алгоритмы [17]. Время работы таких алгоритмов принято оценивать в зависимости от длины полинома N.

Результаты времени выполнения атаки LLL выполнялись по формуле [7]:

$$\log_{10} T = 0{,}1095 \cdot N - 12{,}6402, \qquad (4.1)$$

где N обозначает максимальную длину полинома секретного ключа.

Таким образом, размерность решётки будет $2N$. В таблице 4.3 приведено время T в MIPS-годах, которое потребуется для взлома подписи для различных параметров [1, с. 9].

Таблица 4.3 – Результаты выполнения LLL атаки для NRTU и RSA

Уровень безопасности, k	80	112	128	192	256	360
Длина полиномов секретного ключа NTRU	314	394	446	653	743	1024
Двоичная длина модуля n преобразований RSA	1024	2048	3072	7680	15360	-
Время T в MIPS-годах криптоанализа LLL NTRU	$10^{21.7}$	$10^{30.5}$	$10^{36.1}$	10^{45}	$1{,}01 \cdot 10^{68{,}7}$	$10^{99{,}48}$
Время T в MIPS-годах криптоанализа RSA	$1{,}07 \cdot 10^{10}$	$1{,}25 \cdot 10^{19}$	$4{,}74 \cdot 10^{25}$	$1{,}06 \cdot 10^{45}$	$1{,}01 \cdot 10^{65}$	-

Как можно заметить по данным табл. 4.3, при относительно небольшом увеличении размеров ключа NTRU значительно увеличивается уровень безопасности алгоритма и криптостойкость. Для достижения таких же результатов уровня безопасности RSA необходимо увеличивать длину ключей более чем в 2 раза.

Далее приведены результаты сравнения процедур формирования и проверки подписей у алгоритмов NTRUSign и RSA по двум критериям: длина полинома секретного ключа N и время выполнения этих операций. Данные таблицы 4.4 и таблицы 4.5 соответствуют результатам, полученным для подписи на языке программирования Java [1, с. 9].

Таблица 4.4 – Время выполнения операций в RSA

Длина ключа	k	RSA		
		Генерация ключей, мс	Подписание, мс	Проверка подписи, мс
512	k<60	200	5	1
1024	80	603	10	3
2048	112	1753	68	3
3072	128	20876	211	6
7680	192	278876	2981	21
15360	256	13170341	42570	753

Таблица 4.5 – Время выполнения операций в NTRUSign

Длина ключа	k	NTRUSign		
		Генерация ключей, мс	Подписание, мс	Проверка подписи, мс
157	80	491	300	290
394	112	520	701	503
446	128	7751	931	1030
653	192	19170	5011	2014
743	256	40167	40167	4088

Принимая во внимание сравнительную характеристику, из приведённых результатов можно сделать вывод, что системы на базе NTRU обладают такими преимуществами как быстродействие и значительно более существенная криптографическая стойкость по сравнению с существующими криптосистемами с открытым ключом. Главным же отличием является отсутствие успешных попыток криптоанализа с помощью квантовых алгоритмов.

5 ОХРАНА ТРУДА И БЕЗОПАСНОСТЬ ПРИ ЧРЕЗВЫЧАЙНЫХ СИТУАЦИЯХ

5.1 Анализ потенциальных опасностей

На сегодняшний день невозможно представить какую-либо профессиональную деятельность, где бы не рассматривались вопросы относительно условий рабочего или производственного процесса и обеспечению защиты рабочего персонала и окружающей среды от различного рода вредных факторов сопровождающих производственный процесс, в то же время являющихся неотъемлемой частью такового. Поэтому основной задачей охраны труда является минимизация воздействия всех возможных вредных факторов до приемлемых для безопасной работы значений.

В данной работе рассматривается процесс проектирования и разработки программного обеспечения, предназначенного для осуществления цифровой подписи электронных документов. В качестве рассматриваемого объекта относительно вопросов по охране труда выступает рабочее офисное помещение. Офисное помещение находится в доме с кирпичной кладкой на втором этаже здания, перекрытия железобетонные, имеет площадь 75 м^2. Имеются деревянные столы, кресла, оргтехника, в том числе 4 ПК, помещение предназначено для работы 5-х человек.

В первую очередь охрана труда основывается и регулируется такими нормативно-правовыми актами как Конституция Украины, Законодательство Украины о здравоохранении, Основы земельного законодательства Украины, Закон о трудовых коллективах, Закон Украины об Охране труда. Едиными правилами и нормами, распространяющимися на все отрасли народного хозяйства, являются требования государственных стандартов ССБТ «Строительные нормы и правила» (СНиП), «Санитарные нормы проектирования промышленных предприятий» (СН 245–71).

На рассматриваемом объекте среди прочих учитываются следующие нормативно-правовые акты: ДБН В.2.2-28:2010 «Будинки і споруди. Будинки адміністративного та побутового призначення»; НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин»; ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин»; ГОСТ 12.2.007.0-75* (2001) «ССБТ. Изделия электротехнические. Общие требования безопасности»; ДБН В.2.5-28-2006 «Природне і штучне освітлення»; ДСН 3.3.6-042-99 «Санітарні норми мікроклімату виробничих приміщень»; ДБН В.2.5-67:2013«Опалення, вентиляція та кондиціонування»; НАПБ А.01.001-14 «Правил пожежної безпеки в Україні»; ДБН В.2.5-56:2014 «Системи протипожежного захисту»;

К опасным производственным факторам, характерным для данной деятельности относятся:

- поражение электрическим током;
- повышенный уровень электромагнитных излучений;
- отсутствие или недостаток естественного света, либо наоборот повышенная яркость;
- физические и нервно-психические перегрузки;
- неудовлетворительный микроклимат;
- опасность возгорания.

5.2 Мероприятия по обеспечению безопасности

Основным и главным опасным фактором для разработчиков программного обеспечения работающих с электроникой, в частности за персональными рабочими станциями является поражение электрическим током.

Поражение электрическим током происходит при замыкании электрической цепи через тело человека. Наиболее частым случаем

поражения током в тех случаях, когда человек касается двух или одного проводов, имея при этом контакт с землёй, либо это может быть прикосновение к металлическим поверхностям приборов после перехода на них тока с токоведущих частей и проводов.

Для обеспечения защиты персонала от случайного прикосновения к токоведущим частям применяются такие мероприятия как:

- проверка изоляции токоведущих частей не реже 1 раза в 3 года и своевременная их замена;
- проверка изоляции токоведущих частей после ремонта оргтехники
- применены тугоплавкие предохранители на распределительном щитке для защиты от перегрузки сети;
- оборудование распределительного щитка офисного помещения аварийным выключателем.
- Над розетками вывешены надписи с указанием напряжения в сети (~220 В)

Важной защитой сотрудников от поражения электрическим током является защитное заземление к которому подключаются все электроприборы, находящиеся в рабочем помещении. Монтаж защитного заземления предполагает расчёт параметров со следующими исходными данными: длина L и ширина B помещения соответственно 10 м и 7,5 м; напряжение сети питания U = 220 В; сопротивление растеканию тока природных заземлителей $R_е$ = 12 Ом; длина вертикального стержня $l_в$ = 3 м и диаметр d = 0,012 м; ширина полосы b_c = 0,032; удельное сопротивление земли в ходе измерения $\rho_{вим}$ = 100 Ом · м.

Согласно требованиям главы 1.7 «ПУЕ» и ГОСТ 12.1.030-81 (2001) «ССБТ. Электробезопасность. Защитное заземление, зануление» расчётное нормированное сопротивление заземляющего устройства $R_з$ не более 4 Ом. Вычисление сопротивления искусственного заземлителя выполнено по формуле (5.1):

$$R_u = \frac{R_e R_з}{R_e - R_з}. \quad (5.1)$$

Рассчитаем необходимое сопротивление искусственного заземлителя R_u при $R_з$ = 4 Ом:

$$R_u = \frac{12 \cdot 4}{12-4} = 6.$$

Определение удельного сопротивления грунта ρ Ом·м при коэффициенте сезонности при нормальной влажности грунта Ψ = 1,5 выполнено по формуле (5.2):

$$\rho = \rho_{\text{вим}} \cdot \Psi, \quad (5.2)$$

$$\rho = 100 \cdot 1{,}5 = 150.$$

Расстояние от поверхности грунта до середины длины вертикального стержня $t_в$ м рассчитывается по формуле (5.3):

$$t_{\text{в}} = 0{,}8 + \frac{1}{2} l_{\text{в}}, \quad (5.3)$$

$$t_{\text{в}} = 0{,}8 + 1{,}5 = 2{,}3.$$

Расчёт сопротивления растеканию тока одиночного вертикального заземлителя $R_в$ Ом рассчитывается по формуле (5.4):

$$R_{\text{в}} = \frac{\rho}{2 \cdot \pi \cdot l_{\text{в}}} \left(\ln \frac{2 \cdot l_{\text{в}}}{d} + \frac{1}{2} \cdot \ln \frac{4 \cdot t_{\text{в}} + l_{\text{в}}}{5 \cdot t_{\text{в}} - l_{\text{в}}} \right), \quad (5.4)$$

$$R_{\text{в}} = \frac{150}{6 \cdot \pi} \left(\ln \frac{6}{0{,}012} + \frac{1}{2} \cdot \ln \frac{12{,}2}{8{,}5} \right) = 33{,}8.$$

Минимальное количество вертикальных стержней определяется по формуле (5.5):

$$n' = \frac{R_{\text{в}}}{R_u}. \quad (5.5)$$

Рассчитаем минимальное количество вертикальных стержней заземления, *n*' шт :

$$n' = \frac{33{,}8}{6} = 6.$$

Исходя из размеров контура $P = 2 \cdot (L + B)$ и расстоянием между стержнями $a = 3$ м, количество стержней n определяется по формуле (5.5):

$$n = \frac{P}{a}, \quad (5.5)$$

$$n = \frac{35}{3} = 12.$$

Далее определяем длину горизонтальной полосы $l_г$ м по формуле (5.6):

$$l_{\text{г}} = 1{,}05 \cdot a \cdot n, \quad (5.6)$$

$$l_{\text{г}} = 1{,}05 \cdot 3 \cdot 12 = 37{,}8.$$

Расстояние от поверхности грунта до середины ширины вертикального стержня $t_г$ м рассчитывается по формуле (5.7):

$$t_{\text{г}} = 0{,}8 + \frac{1}{2} b_c, \quad (5.7)$$

таким образом оно составляет:

$$t_{\text{г}} = 0{,}8 + 0{,}016 = 0{,}82.$$

Сопротивление растеканию тока определяется по формуле (5.8):

$$R_{\text{г}} = \frac{\rho}{2 \cdot \pi \cdot l_{\text{г}}} \cdot \ln \frac{2 \cdot {l_{\text{г}}}^2}{b_c \cdot t_{\text{г}}}. \quad (5.8)$$

Вычислим сопротивление растеканию тока горизонтальной соединяющей полосы $R_г$, Ом:

$$R_{\text{г}} = \frac{150}{6 \cdot \pi} \cdot \ln \frac{2857{,}7}{0{,}0262} = 61{,}5.$$

Рассчитаем эквивалентное сопротивление растеканию тока группового заземлителя $R_{гр}$ по (5.9) при коэффициентах использования $\eta_в = 0{,}55$, $\eta_г = 0{,}326$:

$$R_{\text{гр}} = \frac{R_{\text{в}} R_{\text{г}}}{R_{\text{в}} \cdot \eta_{\text{г}} + R_{\text{г}} \cdot \eta_{\text{в}} \cdot n}, \quad (5.9)$$

$$R_{\text{гр}} = \frac{33{,}8 \cdot 61{,}5}{33{,}8 \cdot 0{,}312 + 61{,}5 \cdot 0{,}53 \cdot 12} = 5{,}2.$$

Поскольку выполняется условие $R_{гр} < R_u$, расчёт параметров защитного заземления выполнен корректно.

Для защиты сотрудников от длительного воздействия ЭМИ корпуса рабочих станций выполнены из экранирующих материалов, таким образом за 8 часовой рабочий день предельная плотность потока мощности ЭМП не превышает 0,1 Вт/м^2 согласно ДСН 239-96 «Державні санітарні норми і

правила захисту населення від впливу електромагнітних випромінювань». Также используются безвредные жидкокристаллические дисплеи.

Для решения проблемы негативного воздействия длительной работы за компьютером рекомендуется каждый час отводить 15 минут на отдых согласно ДК 003-95 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин».

5.3 Мероприятия по обеспечению производственной санитарии и гигиены труда

Мероприятия по производственной санитарии и гигиене труда направлены на предотвращение или уменьшение воздействия вредных факторов окружающей среды на работающих, что позволяет обеспечивать на рабочих местах комфортные условия микроклимата, необходимую освещенность рабочих мест, сводят воздействие шума излучений и других вредных производственных факторов приемлемым.

Освещение рабочего помещения устанавливается согласно нормативного документа ДБН В.2.5-28-2006 «Інженерне обладнання будинків і споруд. Природне і штучне освітлення», выполнено смешанным (естественным и искусственным).

Естественное освещение в помещении осуществляется в виде бокового освещения через оконные проёмы стен зданий в количестве трёх единиц. Величина коэффициента естественной освещенности (КЕО) соответствует нормативным уровням по ДБН В.2.5-28-2006.

Искусственное освещение выполнено общим в виде расположенных в два ряда люминесцентных ламп типа ЛДЦ (дневного света с исправленной передачей цветов) с индексом светопередачи не менее 70 (R>70).

Величина освещенности при искусственном освещении люминесцентными лампами соответствует в горизонтальной плоскости не ниже 300 лк.

Для того чтобы исключить засветку экранов дисплеев прямым световым потоком светильников общего освещения, рабочие места располагаются сбоку от светильников параллельно линии зрения рабочего за компьютером и стене с окнами. Дополнительно для помещения, оборудованного рабочими станциями с визуальными дисплеями определяются эргономические характеристики мониторов в соответствии с требованиями директивы ЕС 90/270 ЕЕС «Мінімальні вимоги з охорони праці» учтены основные требования при выборе мониторов, регламентирующие безопасные условия работы с ними:

- символы на экране чёткие и хорошо различимы;
- изображение лишено мерцания;
- яркость и/или контрастность легко регулируются;
- экраны слабо проявляют отблескивание и отражения света.

Для уменьшения проявления стресса вследствие нервно-психического и умственного перенапряжения предусмотрено наличие места для отдыха на рабочем месте, и выход на улицу. На рабочем месте поддерживается дружественная коллективная обстановка, проводятся праздники и корпоративные мероприятия.

На объекте предполагается пятидневная рабочая неделя, продолжительность непрерывного отдыха составляет не менее 42-х часов в неделю. Продолжительность сверхурочных работ не превышает 4-х часов на протяжении двух дней подряд и 120 часов в год.

Согласно требованиям санитарных норм ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку» поддерживается допустимый уровень звука и шумов в помещении до 50 дБА. Снижение уровня внешних шумов достигается шумоизолирующими

металлопластиковыми окнами, системы охлаждения компьютеров выполнены малошумными.

Для обеспечения оптимального уровня параметров воздушной производственной среды использованы нормы ДСН 3.3.6-042-99 «Санітарні норми мікроклімату виробничих приміщень» и ГОСТ 12.1.005-88 (1991) «ССБТ. Общие санитарно-гигиенические требования к воздуху рабочей зоны». При выборе оптимальних параметров микроклимата учитывается характер выполняемых работ. Рассматриваемый объект относится к I категории лёгких работ, поскольку основная работа выполняется сидя и не требует систематического физического напряжения.

К основным параметрам микроклимата относятся:

- температура;
- относительная влажность;
- скорость перемещения воздуха;
- атмосферное давление.

Для I категории работ согласно нормам в холодный и переходный периоды со среднесуточной температурой наружного воздуха ниже +10ºС устанавливается температура рабочего помещения + 20…+23ºС. В тёплый период года со среднесуточной температурой наружного воздуха выше +10ºС устанавливается температура +22…+25 ºС. Для отопления помещения в холодный период предусмотрена система водяного отопления согласно ДБН В.2.5-67:2013 «Опалення, вентиляція та кондиціювання».

Относительная влажность воздуха согласно нормам поддерживается в пределах 40 – 60%.

Для поддержания необходимой температуры и влажности воздуха в тёплый и переходный периоды года применяется система кондиционирования воздуха Neoclima NS09AHB обеспечивающая 2-3-кратный воздухообмен в час, а также естественное кондиционирование (аэрация и проветривание помещения). Скорость перемещения воздуха внутри помещения не превышает 0,1 м/с.

При небольшой загрязненности наружного воздуха кондиционирование помещений осуществляется с переменными расходами наружного воздуха и рециркуляционного. Системы охлаждения и кондиционирования устройств ПК спроектированы, исходя от 90 % -ной рециркуляции.

Атмосферное давление не подлежит регулированию и соответствует естественному атмосферному давлению на уровне моря в пределах 550-950 мм.рт.ст.

5.4 Мероприятия по пожарной безопасности

Комплекс противопожарных мероприятий на рассматриваемом объекте разработки программного обеспечения оборудованного ПК разработан согласно требованиям НАПБ А.01.001-2014 «Правила пожежної безпеки в Україні».

Причиной возникновения пожара на объекте может служить неосторожное обращение с огнём, халатность, возгорание вследствие неправильной эксплуатации приборов и аппаратов, а также короткое замыкания в цепи электропитания приборов и аппаратов. Также возгорание приборов и аппаратуры может происходить вследствие перегрева эллементов эллектронных схем, блоков питания.

В помещении находятся персональные рабочие станции ПК, вспомогательная электроника (принтер, кондиционер), мебель из ДСП и пластмасс (столы и кресла), пол выполнен из лакированного деревянного паркета. Исходя из имеющихся в помещении материалов возможен пожар класса А (горение твёрдых веществ) и класс Е (горение электроустановок находящихся под напряжением) согласно ГОСТ 27331-87 (СТ СЭВ 5637-86) «Пожарная техника. Классификация пожаров».

Таким образом согласно ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та

пожежною небезпекою» рассматриваемое помещение относится к категории «Д», потому как материалы хранящиеся в помещении находятся в холодном состоянии (при комнатной температуре), удельная пожарная нагрузка для твёрдых легковоспламенимых, горючих и негорючих веществ и материалов на площади 10 $м^2$ не превышает 180 $МДж \cdot м^{-2}$.

В соответствии с требованиями норм ДБН В.1.1-7:2016 «Пожежна безпека об'єктів будівництва. Загальні вимоги» предусмотрен аварийный выход из рабочего помещения через дверь в коридор и на лестницу вниз на первый этаж непосредственно к выходу. Коридоры и проходы, предназначенные для эвакуации, имеют приемлемую длину и минимальное количество поворотов, ширина проходов на всём протяжении эвакуационного пути не менее 1 м. На всем протяжении прохода нет порогов или промежуточных ступеней.

Все двери на пути эвакуационного выхода открываются в направлении выхода людей из здания, не имеют запоров препятствующих их свободному открытию изнутри без ключа в случае пожара. На всём следовании пути эвакуации размещены условные знаки и обозначения указывающие направление движения в сторону ближайшего аварийного выхода, двери по ходу эвакуационного пути обозначены опознавательным знаком и надписями «Аварійний вихід».

Поскольку помещение относится к категории пожарной опасности «Д», то максимальное отдаление от наиболее отдалённого рабочего места согласно п. 2.29 (табл. 2) СНиП 2.09.02-85 «Производственные здания» не зависимо от объёма внутренних помещений не ограничивается для категорий огнестойкости здания I, II, III и III а, тем не мене не превышает 17 м.

Оборудование, силовые и осветительные сети рабочего помещения с рабочими станциями ПК выполнены согласно нормам НПАОП 40.1-1.32-01 «Правила будови електроустановок. Електрообладнання спеціальних установок», всё оборудование и аппараты, а также силовые линии и кабели

находящееся в помещении имеют минимальную степень защиты изоляции IP-44.

Исходя из технических и организационных требований предотвращения пожаров в рабочем помещении с ПК на силовой и осветительной цепях электропитания согласно требованиям пункта 3.1 «ПУЕ», установлены защитные устройства, отключающие источники питания от участка электрической цепи, в которой возникло короткое замыкание.

В рабочем помещении имеются средства выявления возгораний и пожаров согласно требованиям ДБН В.2.5-56:2014 «Системи протипожежного захисту». Для этих целей применяется система автоматической пожарной и охранной сигнализации «Лунь-7Т», в составе которой применены дымовые пожарные извещатели «Altronics SM01», теловыми датчиками «Артрон ТПТ-3». Система обеспечивает выявление дымовых и тепловых признаков возгорания с точностью размещения датчика, оповещает сотрудников о пожаре звуковым сигналом и вызывает противопожарную службу через связь GSM. При выборе пожарных извещателей учитывались конкретные условия их эксплуатации: особенности помещения и воздушной среды, наличие пожарных материалов, характер возможного горения и т.п.

Поскольку в рассматриваемом помещении используется электроника включённая в электросеть, то возникший пожар нельзя тушить водой, чтобы не повредить цепи электропитания и не вывести из строя работоспособную электротехнику, а также избежать поражения электрическим током. В первую очередь, при возникшем пожаре следует отключить сеть электропитания в пределах помещения, и вывести людей и только затем приступить к тушению. При невозможности потушить локальный пожар самостоятельно, следует немедленно покинуть место возникновения пожара и вызвать спасательную противопожарную службу.

Так как помещение площадью 75 м^2 оборудовано рабочими станциями ПК, то согласно п. 3.8 раздела «Типові норми належності вогнегасників» ДСТУ 4297:2004 «Пожежна техніка. Технічне обслуговування вогнегасників. Загальні технічні вимоги» для тушения электроустановок находящихся под напряжением предусмотрены углекислотные огнетушители типа ОУ-2 в количестве 4 шт. с расчётом по 1 огнетушителю на 20 м^2 площади помещения с величиной огнетушащего вещества не менее 3 кг. Огнетушители легкодоступны и размещены в местах наиболее вероятного возникновения пожара, таким образом расстояние от предполагаемых очагов пожара до места размещения огнетушителей не превышает 70 м в соответствии с «Д» категорией пожаробезопасности помещения. По крайней мере 1 огнетушитель размещается возле выхода из помещения.

5.5 Мероприятия по обеспечению безопасности при чрезвычайных ситуациях

Защита населения в чрезвычайных ситуациях — одна из главных задач гражданской обороны. Под режимом защиты понимается применение средств и способов, максимально снижающих вероятность заражения, отравления либо облучения людей в зоне поражения.

Способами защиты населения являются:

1. Своевременное оповещение населения.
2. Мероприятия по противорадиационной и противохимической защите (ПРиПХЗ).
3. Укрытие людей в защитных сооружениях.
4. Использование средств индивидуальной защиты.
5. Проведение эвакомероприятий (рассредоточение и эвакуация населения из городов в загородную зону).

Своевременное оповещение осуществляется органами ГО. Оно организуется по радио- и телевещанию. Чтобы привлечь внимание населения, используют сигналы транспортных средств, а также включают гудки на предприятиях. Услышав сигналы «Внимание всем!», надо немедленно включить теле- и радиоприемники и ждать сообщения от местных органов власти или штаба ГО. Все дальнейшие действия определяются их указаниями.

Рассредоточение и эвакуация населения — один из способов защиты от оружия массового поражения. Под рассредоточением понимают организованный вывоз из городов и других населенных пунктов и размещение в загородной зоне свободной от работы смены рабочих и служащих, продолжающих работу в военное время. Рабочие и служащие после расселения в загородной зоне посменно выезжают в город для работы на своих предприятиях, а по окончании работы возвращаются. Эвакуация представляет собой организованный вывоз или вывод из городов и других населенных пунктов и размещение в загородной зоне остального населения, а также вывоз или вывод населения из зон возможного затопления. Районы расселения рабочих и служащих в загородной зоне должны находиться на таком удалении от города, которое обеспечило бы их безопасность, а на переезд людей для работы в город и их возвращение в загородную зону для отдыха затрачивалось бы минимальное время поэтому они располагаются вблизи железнодорожных станций и автомагистралей.

Для защиты населения от ядерного, химического и бактериологического (биологического) воздействия существуют специальные сооружения. Они в зависимости от защитных свойств подразделяются на убежища и противорадиационные укрытия (ПРУ). За счет прочности ограждающих конструкций и перекрытий убежища обеспечивают наиболее надежную защиту людей от всех поражающих факторов — отравляющих веществ и бактериальных средств, от высоких температур и вредных газов в зонах пожаров, от обвалов, происшедших в результате

взрыва. В убежищах предусмотрены надлежащие санитарно-гигиенические условия, обеспечивающие нормальную жизнедеятельность людей. Наиболее распространены встроенные убежища, оборудованные в подвальных или полуподвальных помещениях производственных, общественных и жилых зданий.

При объявлении о ЧС все население должно быть обеспечено средствами индивидуальной защиты. Личный состав формирований, рабочие и служащие получают средства индивидуальной защиты на своих объектах, население — в ЖЭС, ДЭЗ, а также самостоятельно изготавливает тканевые маски, ватно-марлевые повязки и другие простейшие средства защиты органов дыхания, а для защиты кожных покровов подготавливают различные накидки, плащи, резиновую обувь, резиновые или кожаные перчатки. Средства индивидуальной защиты следует хранить на рабочих местах.

Наиболее надежным средством защиты органов дыхания людей являются противогазы. Они предназначаются для защиты органов дыхания, лица и глаз человека от вредных примесей, находящихся в воздухе. По принципу действия все противогазы подразделяются на фильтрующие и изолирующие. В настоящее время в системе гражданской обороны для взрослого населения используются фильтрующие противогазы ГП-5.

Изолирующие противогазы (ИП-4, ИП-5, ИП-46, ИП-46М) являются специальными средствами защиты органов дыхания, глаз, кожных покровов от всех вредных примесей, содержащихся в воздухе. Их используют в том случае, когда фильтрующие противогазы не обеспечивают такую защиту, а также в условиях недостатка кислорода в воздухе.

Таким образом, все мероприятия, описанные в этом разделе, направлены на обеспечение безопасности персонала ведущего разработку программного обеспечения при чрезвычайных ситуациях.

6 ТЕХНИКО ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ПРОЕКТА

6.1 Планирование работ по проектированию ПО

Рациональное планирование работ по проектированию и разработке программного обеспечения позволяет добиться минимальных затрат рабочего времени, материальных и денежных ресурсов.

Следует выполнить такие задачи по планированию работ, как:

- определение трудоёмкости и длительности работ;
- составить календарный график работ;
- вычислить затраты в процессе выполнения работ.

Для начала, весь объём работ по проектированию, разработке и обоснованию программного обеспечения подразделяется на этапы, на каждом из которых указываются ожидаемая продолжительность работ, ответственные исполнительные лица, их квалификация и количество, а также трудоёмкость работ по каждому этапу.

Поскольку работы по проектированию и созданию ПО в сфере криптографии имеют исследовательский характер и затруднительно определить общую длительность работ, то для определения общей трудоёмкости использован вероятностный подход. Поэтому в таком случае используются две или три вероятные оценки ожидаемого времени выполнения работы, которые даются исполнителями на каждом этапе работы. Оценки ожидаемого времени работы выполняются по формуле (6.1):

$$t_{\text{ож}} = \frac{3t_{min} + 2t_{max}}{5}, \tag{6.1}$$

где $t_{\text{ож}}$ – ожидаемая оптимальная оценка времени выполнения работы в днях, $t_{\min}$ – минимальное количество дней на выполнение работы при благоприятных условиях, $t_{\max}$ –максимальное количество дней на выполнение работ при неблагоприятных условиях.

Доверительная вероятность P к $t_{ож}$ определяется расчётом ожидаемой величины между минимальной и максимальной оценками времени по формуле (6.2):

$$P = \frac{t_{max} - t_{min}}{5}. \quad (6.2)$$

Данные оценки ожидаемого времени выполнения работ по проектированию ПО приведены в табл. 6.1.

Таблица 6.1 – Оценка ожидаемого времени выполнения работ

Наименование работы, этапы	Временные оценки			P	Исполнители	
	t_{min}	t_{max}	$t_{ож}$		Квалификация (профессиональная)	Количество чел.
1. Анализ, утверждение технического задания	3	5	4	0,4	Специалист по криптографии	1
2. Планирование работ по проектированию	8	11	9	0,6	Специалист по криптографии	2
3. Проведение расчётных исследований	8	10	9	0,4	Специалист по криптографии	1
4. Разработка и отладка ПО	16	21	18	1	Инженер-программист	3
5. Тестирование и доработка ПО	12	14	13	0,4	Инженер-программист	3
6. Обобщение полученных результатов	3	4	3	0,2	Специалист по криптографии	1
7. Экономическое обоснование проекта	3	5	4	0,4	Экономист	1
8. Оформление технической документации	7	9	8	0,4	Специалист по криптографии	2
Всего	60	79	68	-	-	-

Трудоёмкость работ определяется как время, необходимое для выполнения определённой работы одним специалистом и соответственно измеряется в человеко-днях. Трудоёмкость также зависит от таких характеристик, как сложность разработки, научная новизна и неопределённость, наличие опыта по решению каких-либо проблематик и вопросов, доступности и надёжности оборудования и программного обеспечения на котором осуществляется работа, условий работы и требований нормативных документов и т.п.

Продолжительность этапов работ определяется как результат деления трудоёмкости конкретного этапа работы на произведение количества исполнителей и планового коэффициента выполнения нормы. Продолжительность этапов работ рассчитывается по формуле (6.3):

$$T_{\text{ц}} = \frac{Q}{R \cdot k_{\text{в.н.}}}, \tag{6.3}$$

где $T_ц$ – длительность цикла в днях; Q – трудоёмкость работ в человеко-днях; R – число исполнителей работы, чел; $k_{в.н.}$ – плановый коэффициент выполнения нормы, принимаемый за $k_{в.н.}$ = 1,1. Рассчитаем продолжительность этапов работ:

$$T_{\text{ц}_1} = \frac{4}{1 \cdot 1{,}1} = 4\ (дн), \qquad T_{\text{ц}_5} = \frac{4}{1 \cdot 1{,}1} = 4\ (дн);$$

$$T_{\text{ц}_2} = \frac{9}{2 \cdot 1{,}1} = 4\ (дн), \qquad T_{\text{ц}_6} = \frac{4}{1 \cdot 1{,}1} = 3\ (дн);$$

$$T_{\text{ц}_3} = \frac{9}{1 \cdot 1{,}1} = 8\ (дн), \qquad T_{\text{ц}_7} = \frac{4}{1 \cdot 1{,}1} = 4\ (дн);$$

$$T_{\text{ц}_4} = \frac{4}{1 \cdot 1{,}1} = 6\ (дн), \qquad T_{\text{ц}_8} = \frac{4}{1 \cdot 1{,}1} = 4\ (дн);$$

Таким образом, общая длительность всего объёма работ составляет $T = \sum T_{\text{ц}_i} = 37$ дней. Данные трудоёмкости и длительности работ по разработке ПО приведена в табл. 6.2.

Таблица 6.2 – Длительность и трудоёмкость этапов работы

Наименование работы, этапы	Трудоёмкость Q		Исполнители		Длительность, дней
	Чел.-дней	% от итога	Квалификация (профессиональная)	Количество чел.	
Анализ, утверждение технического задания	4	5,88	Специалист по криптографии	1	4
Планирование работ по проектированию	9	13,24	Специалист по криптографии	2	4
Проведение расчётных исследований	9	13,24	Специалист по криптографии	1	8
Разработка и отладка ПО	18	26,47	Инженер-программист	3	6
Тестирование и доработка ПО	13	19,12	Инженер-программист	3	4
Обобщение полученных результатов	3	4,41	Специалист по криптографии	1	3
Экономическое обоснование, проекта	4	5,88	Экономист	1	4
Оформление технической документации	8	11,77	Специалист по криптографии	2	4
Всего	68	100	-	-	37

Основываясь на результатах, полученных по табл. 6.1 и табл. 6.2 можно выполнить построение календарного графика выполнения работ по разработке программного обеспечения. Поскольку этапы выполнения работ должны последовательно следовать друг за другом, то построим линейный календарный график работ с учётом максимального согласования во времени

работ на смежных этапах. Линейный график работ представлен на рисунке 6.1.

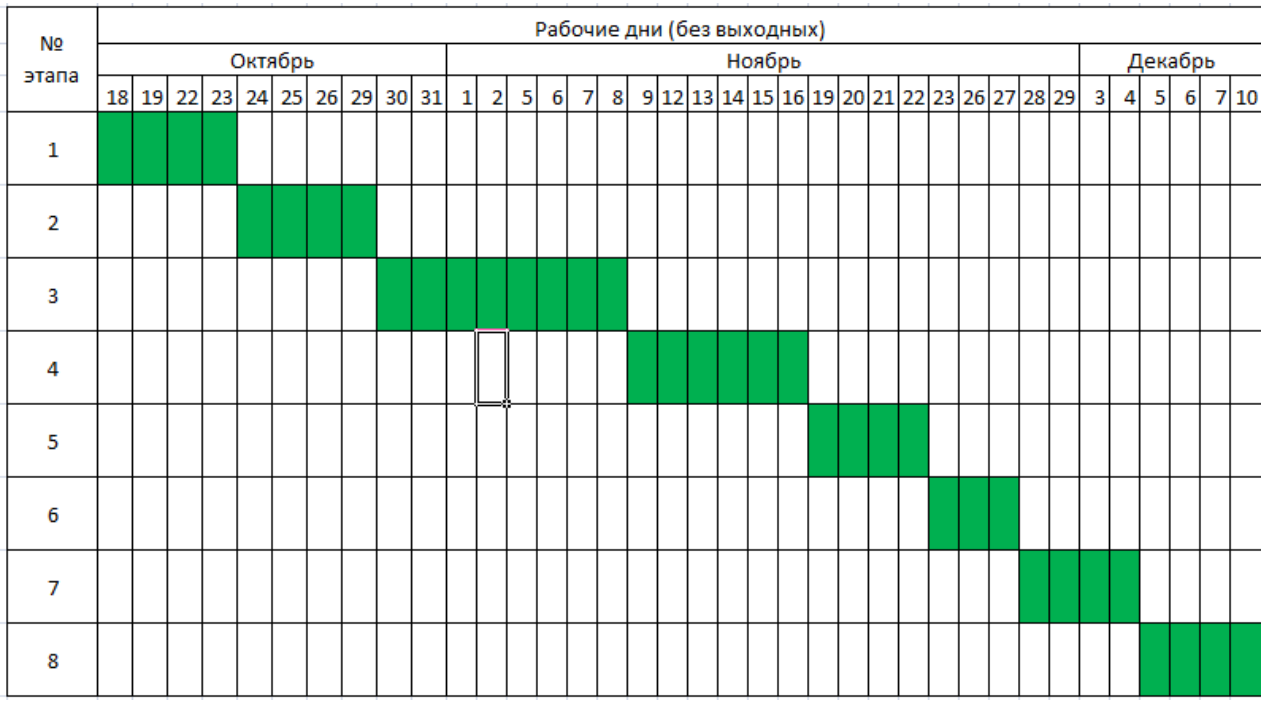

Рисунок 6.1 – График выполнения работ по разработке ПО.

6.2 Определение затрат на разработку ПО

Для того чтобы определить денежные затраты на осуществление разработки программного обеспечения составляется смета по затратам:

- материалы;
- основная заработная плата;
- дополнительная заработная плата;
- единый социальный взнос;
- затраты на специальное оборудование;
- амортизационные отчисления;
- накладные затраты;
- прочие затраты.

Вначале определяются затраты на материалы. Цена по каждому из материалов определяется из прейскурантов или других справочных данных.

Транспортно-закупочные расходы принимаются равными 3-10% от стоимости материалов. В эту статью включается стоимость основных и вспомогательных материалов (бумага, модули памяти USB, картридж, тонер, канцелярские принадлежности и т.п.), необходимые в ходе разработки ПО, расчёт которых приведён в таблице 6.3.

Таблица 6.3 – Вычисление затрат на материалы

Наименование, тип, марка	Единица измерения	Цена за единицу измерения, грн	Коли-чество единиц измерения	Сумма затрат, грн
Бумага для принтера, А4	пачка	95	3	285
Флеш-накопитель, Transend	шт	200	3	600
Набор для скрепления бумаги	шт	73,65	2	147,30
Картридж, Cannon, чёрный	шт	404	1	404
Степлер	шт	54	4	216
Блокнот	шт	11,40	5	57
Батарейки, Express Power	пачка	12,35	3	37,05
Кабель, сетевой	м	66	40	2640
Вместе	-	-	-	4386,35
Транспортно-заготовительные работы	-	-	-	220
Всего	-	-	-	4606,35

Далее можно перейти к расчёту основной, дополнительной заработной платы и социальных взносов. Затраты по этой статье расходов складываются из планового фонда заработной платы категорий работников, которые заняты разработкой проекта. Размер дополнительной заработной платы определяется в виде премий в размере 15% от основной заработной платы сотрудников штата. Единый социальный взнос определяется в процентном соотношении к сумме основной и дополнительной заработной платы в соответствии с установленной общей ставкой единого социального взноса 22%.

Данные по расчёту основной и дополнительной заработной платы, а также единого социального взноса представлены в таблице 6.4.

Таблица 6.4 – Вычисление основной, дополнительной заработной платы и социального взноса

Должность исполнителя	Количество, чел	Месячный оклад, грн	Средне-дневная зарплата, грн	Количество дней работы	Сумма зарплаты, грн		Сума единого социального взноса, грн
					основная	премия	
Специалист по криптографии	1	7825	257,26	23	5917	887	1497
Инженер-программист	3	6440	212,72	10	6351,8	2858	2026,2
Экономист	1	7450	245	4	980	146	247,9
Всего	5	-	-	37	13249	3893	3771

Теперь перейдём к расчёту затрат на специальное оснащение офиса. К этой статье затрат относятся затраты на покупку, транспортировку, монтаж, настройку и эксплуатацию специального оснащения, которое используется при разработке ПО, проведения вычислений, отладки и тестов. К этому оснащению относятся компьютеры (рабочие станции), принтеры, сканеры, мониторы и прочее оборудование долгосрочного использования. Всё это является основными фондами производства.

Определим затраты на приобретение, доставку и настройку специального оборудования и ПО к нему. Данные сведены в таблицу 6.5.

Таблица 6.5 – Расчёт затрат на основные фонды

Наименование оборудования/ ПО	Количество единиц, шт	Стоимость ОС, грн	Стоимость доставки, грн
Системный блок, Acer	5	16300	70
Монитор, Samsung	5	12300	65
Принтер, Cannon	1	4210	35
Клавиатура, Logitech	5	500	10
Мышь, Logitech	5	1000	10
ОС Windows 7	1	1250	-
Программный пакет разработки Maple 13	1	1730	-
Всего		37290	190
Общие затраты			37480

Далее перейдём к расчёту амортизационных отчислений. Начисление амортизации осуществляется в течении всего срока эксплуатации объекта, который является активом, и приостанавливается на период вывода объекта

из использования. Минимально допустимый срок эксплуатации для электронно-вычислительных машин, предназначенных для автоматической обработки, хранения и передачи информации составляет не менее 2 лет.

Амортизационные отчисления определяются по формуле (6.4):

$$В_а = Ф_б \cdot \frac{Н_а}{100}, \quad (6.4)$$

где $Ф_б$ – балансовая стоимость оборудования, грн.; $Н_а$ – норма амортизационных отчислений на полное восстановление оборудования данного вида.

Норма амортизации определяется по формуле 6.5:

$$Н_а = \frac{1}{Т_{службы}}. \quad (6.5)$$

Срок службы оборудования принимается равным двум годам $Т_{службы} = 2$. Результаты вычисленных норм амортизационных отчислений сведены в таблицу 6.6.

Таблица 6.6 – Расчёт суммы амортизационных отчислений

Тип, наименование оборудования	Балансовая стоимость, грн	Норма амортизации	Годовые амортизационные отчисления, грн
Системный блок, Acer	16300	½	8150
Монитор, Samsung	12300	½	6150
Принтер, Cannon	4210	½	2105
ОС Windows 7	1250	½	625
Всего	34060	-	17030

Для ПО, используемом на данном оборудовании амортизационные отчисления не рассчитываются, поскольку не являются материальными активами.

Амортизационные отчисления А за весь период разработки программного обеспечения составляет:

$$А = \frac{17030}{365} \cdot 37 = 1726 \text{ (грн).}$$

Далее вычислим расходы на эксплуатацию оборудования, которые включают затраты на потребляемую оборудованием электроэнергию, вспомогательные материалы и годовой фонд оплаты специалиста сторонней организации, который обслуживает данное оборудование с учётом отчислений единого социального взноса. Затраты на потребляемую оборудованием электроэнергию и эксплуатацию электронно-вычислительных машин определяются по формуле (6.6):

$$В_{ел} = П \cdot Ф_{еф} \cdot k_{вм} \cdot Ц_{ел}, \quad (6.6)$$

где П – потребляемая мощность оборудования, кВт; $Ф_{еф}$ – эффективный фонд времени работы оборудования, ч.; $k_{вм}$ – коэффициент использования мощности оборудования; $Ц_{ел}$ – цена 1 кВт-часа электроэнергии, грн.

Эффективный фонд времени работы оборудования принимается равным $Ф_{еф} = 8 \cdot 37 = 296$ часов, расход электроэнергии составляет 1900 кВт. Цена 1 кВт-часа – 1,50 грн. Общая сумма затрат электроэнергии составляет $Ц_{ел} = 2850$ грн.

Данные по расчёту электроэнергии приведены в таблице 6.7.

Таблица 6.7 – Затраты на потребляемую оборудованием электроэнергию

Тип, наименование оборудования	Мощность оборудования, кВт	Коэффициент использования мощности	Затраты на электроэнергию, грн
Системный блок, Acer	0,3	0,55	1200
Монитор, Samsung	0,07	1	1200
Принтер, Cannon	0,1	0,1	450
Всего	0,47	-	2850

Затраты на оплату труда специалиста по ремонту и эксплуатации электронно-вычислительных машин определяются по формуле (6.7):

$$В_{зп} = O \cdot K_1 \cdot 12 \cdot K_2 \cdot K_3, \quad (6.7)$$

где O – среднемесячный оклад специалиста по ремонту и эксплуатации электронно-вычислительных машин, грн; K_1 – коэффициент, учитывающий премии и надбавки: K_1 = 1+ 0,15 = 1,15; K_2 – коэффициент, учитывающий процент единого социального взноса: K_2 = 1+ 0,22 = 1,22; K_3 - коэффициент, учитывающий, занятость специалиста в данном проекте: K_3 = 0,1.

Поскольку специалист по ремонту и эксплуатации электронно-вычислительных машин является подрядчиком, тогда затраты по услугам специалиста рассчитываются следующим образом:

$$B_{зп} = 6200 \cdot 1{,}15 \cdot 12 \cdot 1{,}22 \cdot 0{,}02 = 2088 \text{ (грн)}.$$

Расходы на эксплуатацию оборудования рассчитываются по формуле (6.8):

$$В_{експ.оборуд} = В_{ел} + В_{зп}, \quad (6.8)$$

и таким образом составляют:

$$В_{експ.оборуд} = 2850 + 2088 = 4938 \text{ (грн)}.$$

Затраты на текущий ремонт оборудования принимаются равными в размере 5% от его балансовой стоимости: 34060 · 0,05 = 1703 грн.

Прочие затраты составляют 2% от суммы всех предыдущих статей расходов на содержание и эксплуатацию оборудования: (4938 + 1703 + 1726 + 37480 + 13249 + 3893 + 3771 + 4606) · 0,02 = 1427 грн. Данные сметы по статьям затрат на содержание и эксплуатацию оборудования приведены в таблице 6.8.

Таблица 6.8 – Смета расходов на содержание и эксплуатацию оборудования

Статья расходов	Сума, грн
Амортизация оборудования	1726
Эксплуатация оборудования	4938
Текущий ремонт оборудования	1703
Прочие расходы	1427
Эксплуатационные расходы, всего	9794

Теперь можно вычислить расходы на оплату машинного времени используемых электронно-вычислительных машин, которые могут быть определены по формуле (6.9):

$$С_{мч} = Р_{експ} \cdot Т_{м}, \quad (6.9)$$

где $С_{мч}$ – затраты на оплату машинного времени, грн.; $Р_{експ}$ – эксплуатационные расходы на один час машинного времени, грн./ч.; $Т_{м}$ – время использования электронно-вычислительных машин при проектировании, часов.

Эксплуатационные расходы за один час машинного времени рассчитывается по (6.10), путём распределения затрат на разработку проекта по смете расходов на содержание и эксплуатацию оборудования $В_{сод.експ}$ из таблицы 6.8. При этом, эффективный фонд времени на разработку проекта принимается равным $Ф_{еф}$ = 296 часам в случае односменной работы.

$$Р_{експ} = \frac{В_{сод.експ}}{Ф_{еф}}, \quad (6.10)$$

$$Р_{експ} = \frac{9794}{296} = 33{,}09 \text{ (грн./час).}$$

Тогда расходы на оплату машинного времени составляют:

$$С_{мч} = 33{,}09 \cdot 37 = 1224 \text{ (грн).}$$

К накладным расходам относятся затраты на общее управление и общехозяйственные расходы. Сюда входят расходы на содержание и эксплуатацию здания в котором находиться офисное помещение, оплата электроэнергии потребляемой источниками света и системой кондиционирования воздуха, оплата отопления офисного помещения в холодный период года. В сумме эти затраты условно составляют 9783 грн. за весь период разработки программного обеспечения

Также к накладным расходам относятся затраты на услуги сторонних организаций (контрагентные расходы), к которым относятся расходы на

услуги и оплату работ специалистам, предоставленных другими организациями для данной деятельности в соответствии с составленными договорами. В частности, такими расходами являются расходы на предоставление услуги доступа к Интернету, на оплату труда охранного холдинга по охране помещения и его уборку клининг-холдингом.

В качестве провайдера для предоставления услуги доступа к сети Интернет был выбран «Укртелеком». Согласно договору, подключение является бесплатным, а абонентская плата за пользование услугами доступа к Интернету составляет 45 грн. ежемесячно. Общая сума расходов на услуги доступа к сети Интернет за весь срок разработки программного обеспечения составляет:

$$\mathrm{P}_{\text{ин}} = 45 \cdot 2 = 90 \text{ (грн)}.$$

Оплата труда сотрудникам охранного холдинга «Сесориту» за охрану помещения составляет 120 грн./сут. Расходы на охрану помещения в течение всего цикла разработки ПО составляют:

$$\mathrm{P}_{\text{ох}} = 120 \cdot 37 = 4440 \text{ (грн)}.$$

Плата за уборку в помещении клининг-холдингу «Клинмастер» составляет 40 грн./день. Соответственно, расходы на уборку офисного помещения в течение всего цикла разработки ПО составляют:

$$\mathrm{P}_{\text{уб}} = 40 \cdot 37 = 1480 \text{ (грн)}.$$

Таким образом накладные расходы в течение всего цикла разработки программного обеспечения составляют:

$$\mathrm{P}_{\text{кон}} = 4440 + 1480 + 9783 + 90 = 15793 \text{ (грн)}.$$

Все остальные затраты принимаются равными 1% от сумы предыдущих затрат.

На основе всех предшествующих расчётов затрат составляется смета стоимости всех работ, представленная в таблице 6.7.

Таблица 6.7 – Смета стоимости работ по разработке ПО

Статья расходов	Сумма, грн	Структура, %
Материалы	4606,35	5,1
Основная заработная плата	13249	14,8
Дополнительная заработная плата	3893	4,4
Единый социальный взнос	3771	4,2
Расходы на специальное оборудование	37480	41,9
Эксплуатационные расходы оборудования	9794	10,9
Накладные затраты	15793	17,7
Прочие расходы	886	0,99
Итого	89472	100

6.3 Определение экономической эффективности программного обеспечения

Экономическая эффективность програмного обеспечения принимается равной эффективности автоматизированной системы, в которой данная программа эксплуатируется или определяется, исходя из частичной доли, приходящейся на программу в экономическом эффекте автоматизированной системы.

Объём капитальных вложений K потребителя програмного обеспечения, которые связаны с её приобретением и внедрением определяются по формуле (6.11):

$$К = К_{пі} + К_к, \quad (6.11)$$

где $К_{пз}$ – предпроизводственные затраты при разработке і-й программы, грн.; $К_к$ – капитальные вложения в производственные фонды, которые необходимы для внедрения программы, грн.

Предпроизводственные затраты включают расходы, связанные с разработкой и настройкой программного, математического и информационного обеспечения, проектированием автоматизированной системы и определяются по формуле (6.12):

$$K_{пі} = \frac{K_n}{n} + K_{др.п.з.}, \quad (6.12)$$

где K_n – цена программного обеспечения в случае если программа предназначена для реализации или суммарные расходы (себестоимость) на разработку программы, в том случае если она для внутреннего использования, грн.; n – количество потребителей программного продукта; $K_{др.п.з.}$ – другие предпроизводственные затраты, грн.

Таким образом объём предпроизводственных затрат составляет:

$$K_{пі} = \frac{94277}{10} + 345 = 9772{,}7 \text{ (грн).}$$

Капитальные вложения в производственные фонды включают расходы на приобретение, установку, монтаж, и настройку комплекса технических средств, с учётом возможного его использования для нескольких различных программ, определяются по формуле (6.13):

$$K_к = \frac{К_{КТЗ} \cdot Т_{КТЗ}}{Ф^{еф}_{КТЗ}}, \quad (6.13)$$

где $К_{КТЗ}$ – затраты на приобретение, установку, монтаж, и наладку комплекса

технических средств, грн.; $Т_{КТЗ}$ – машинное время комплекса технических средств, которое необходимо для задач, решаемых при помощи предоставленной программы, машино-час/год; $Ф^{еф}_{КТЗ}$ - годовой эффективный фонд времени работы комплекса технических средств, машино-час/год.

Капитальные вложения в производственные фонды составляют:

$$K_к = \frac{8027 \cdot 180,8}{89,5} = 16215,4 \text{ (грн)}.$$

Теперь можно рассчитать объём капитальных вложений потребителя программного обеспечения, который составляет:

$$K = 9772,7 + 16215,4 = 25988 \text{ (грн)}.$$

Затраты, связанные с эксплуатацией программного обеспечения $В_{эксп}$ определяются в грн./год на потребителя программы по формуле 6.14:

$$В_{експ} = Р_{експ} \cdot Т_{КТЗ} + \frac{K_n}{n \cdot Т_с}, \quad (6.14)$$

где $Р_{эксп}$ – эксплуатационные расходы, которые приходятся на 1 час машинного времени комплекса технических средств, грн./машино-час.; $Т_{КТЗ}$ – машинное время комплекса технических средств, нужное данному потребителю программного обеспечения для задач, решаемых с помощью данной программы, машино-час.; K_n – суммарные расходы на разработку программы, грн.; n – количество потребителей программного продукта; $Т_с$ – предполагаемый срок службы данного программного обеспечения до его морального устаревания, лет.

Рассчитаем затраты потребителя на эксплуатацию данного программного обеспечения:

$$В_{эксп} = 15 \cdot 180{,}8 + \frac{94277}{10 \cdot 15} = 3340{,}5 \text{ (грн./год).}$$

Годовая экономия от внедрения программного обеспечения, позволяющая снизить расходы машинного времени электронно-вычислительных комплексов для решения определённой задачи по сравнению с программным обеспечением, использовавшимся ранее, определяется по формуле (6.15):

$$P = T_{m1} - T_{m2} \cdot Р_{эксп} + E_н \cdot \frac{К_{КТЗ} \cdot T_{m1} - T_{m2}}{Ф^{еф}_{КТЗ}} - \frac{K_n}{Т_с}, \qquad (6.15)$$

где T_{m1}, T_{m2} – машинное время, необходимое для решения поставленных задач, в соответственно в новом и предыдущем вариантах, машино-час./год; $Р_{эксп}$ – эксплуатационные расходы, которые приходятся на 1 час машинного времени комплекса технических средств, грн./машино-час; $E_н$ – нормативный коэффициент эффективности капитальных вложений (0,33); $К_{КТЗ}$ – балансовая стоимость комплекса технических средств электронно-вычислительных машин, грн.; $Ф^{еф}_{КТЗ}$ - годовой эффективный фонд времени работы комплекса технических средств, машино-час/год.; K_n – суммарные расходы на разработку программного обеспечения, грн.; $Т_с$ – предполагаемый срок службы данного программного обеспечения до его морального устаревания, лет.

Годовая экономия от внедрения нового программного обеспечения для решения определённой задачи по сравнению с предшествующим программным обеспечением составляет:

$$P = 6858 - 6403 \cdot 15 + 0{,}33 \cdot \frac{8027 \cdot 6858 - 6403}{89{,}5} - \frac{94277}{15} = 14047 \text{ (грн).}$$

Таким образом, годовой экономический эффект от внедрения нового программного обеспечения $E_р$ можно определить по формуле (6.16):

$$E_{\text{р}} = P - E_{\text{н}} \cdot K. \quad (6.16)$$

Рассчитаем годовой экономический эффект от внедрения нового программного обеспечения, который составляет:

$$E_{\text{р}} = 14047 - 0{,}33 \cdot 25988 = 5471 \text{ (грн).}$$

Коэффициент экономической эффективности вложений средств в программное обеспечение E вычисляется по формуле (6.17), а срок их окупаемости $T_{\text{ок}}$ вычисляется по формуле (6.18):

$$E = \frac{P}{K}, \quad (6.17)$$

$$\text{Т}_{\text{ок}} = \frac{K}{P}. \quad (6.18)$$

Определим коэффициент экономической эффективности вложений средств в программное обеспечение и срок их окупаемости:

$$E = \frac{14047}{25988} = 0{,}54,$$

$$\text{Т}_{\text{ок}} = \frac{25988}{14047} = 1{,}9.$$

Все результаты производимых вычислений сведены в итоговую таблицу 6.8 экономических показателей и предоставляются окончательные выводы по экономической эффективности разработки программного обеспечения.

Таблица 6.8 – Экономические показатели

Показатель	Условное обозначение	Значение показателя
Затраты на разработку программного продукта, грн.	$К_{пі}$	9772,7
Капитальные вложения в производственные фонды, грн.	$К_к$	16215,4
Капитальные затраты, грн.	$К$	25988
Эксплуатационные расходы, грн	$В_{эксп}$	3340,5
Срок окупаемости капитальных затрат, лет	$Т_{ок}$	1,9
Коэффициент экономической эффективности вложений	$Е$	0,54
Годовая экономия от внедрения, грн.	$Р$	14047
Годовой экономический эффект, грн.	$Е_р$	5471

Таким образом, если сопоставить полученный коэффициент экономической эффективности ($Е$ = 0,54) с нормативным ($Е_н$ = 0,33) и полученный срок окупаемости капиталовложений ($Т_{ок}$ = 1,9 года) с нормативным сроком окупаемости ($Т_{норм}$ = 3 года), то можно сделать вывод о том что данная разработка экономически эффективна и целесообразна, так как выполняются условия экономической эффективности $Е > Е_н$ и $Т_{ок} < Т_{норм}$.

ВЫВОДЫ

В данной работе рассматриваются основные принципы и особенности построения криптографических алгоритмов с использованием теории решёток в кольцах усечённых многочленов, а также реализация электронной цифровой подписи на базе NTRU.

Определены текущие стандарты на базе NTRU, их современное состояние и применение. Рассмотрена актуальность и обоснована возможность разработки электронной цифровой подписи, основывающаяся на криптографических преобразованиях в кольцах полиномов и построения NTRU-решёток. Проведено математическое обоснование такой криптосистемы, рассмотрены понятия кольца и особенностей проведения операций в нём, а также понятия решётки, её базиса и кратчайшего вектора решётки. Рассмотрены основные возможные вычислительно сложные задачи, на которых основывается криптографическая стойкость NTRU-алгоритмов, в частности задача нахождения кратчайшего вектора решётки.

После осуществления математического обоснования NTRU криптосистемы, написана программа на языке Maple, реализующая алгоритм работы NTRUSign, которая позволила осуществить этапы создания ключевой пары, формирования и проверки подписи для рекомендуемых значений параметров, а также исследовать особенности такой реализации.

Наконец, приведен перечень и рассмотрены основные особенности возможных атак на рассматриваемый класс криптосистем, в частности приведена сравнительная характеристика криптографической стойкости и производительности совместно с другими классами электронных цифровых подписей. Выделены преимущества криптосистем NTRU на фоне других.

Можно утверждать, что криптосистемы NTRU являются весьма перспективными для предоставления услуг информационной безопасности.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Шевцов О. Сутність та оцінка стійкості криптографічних перетворень в NTRUSign / О. Шевцов // Радиотехника. – 2015. - № 181. – С. 5-11

2. Gentry C. Cryptanalysis of the Revised NTRU Signature Scheme / C. Gentry, M. Szydlo / DoCoMo USA Lab. – San Jose, 2002. – 32 p.

3. NTRUSign: Digital Signatures Using the NTRU Lattice / J. Hoffsthein, N. Howgrave-Graham, J. Pipher, J. H. Silverman, W. Whyte / NTRU Cryptosystems. – Burlington, 2001. – 18 p.

4. Атья М. Введение в коммутативную алгебру / М. Атья, И. Макдональд. – М.: Мир, 1972. – 160 с.

5. Усатюк В. Задачи теории решёток и их взаимные редукции: Автореф. …аспирант / В. Усатюк / Братский государственный университет. – Братск, 2011. – 12 с.

6. Качко Е. Обоснование и исследование практической реализации улучшенного алгоритма цифровой подписи NTRUSign / Е. Качко, Д. Балагура, Ю. Горбенко // Прикладная радиоэлектроника. – 2012. - № 2 – С. 195 – 199.

7. Стафийчук П. Анализ и сравнение свойств ключевых данных системы NTRUSign / П. Стафийчук, О. Товма // Системи обробки інформації. – 2012. - № 5 – 103. – С. 101 – 104.

8. Meskanen Tommi. On the NTRU Cryptosystem [Электронный ресурс] / Tommi Meskanen. – Режим доступа: http://www/tucs./publications/-attachment.php?fname=DISS63.pdf, свободный.

9. Hoffstein J. An Introduction to Mathematical Cryptography [Електронний ресурс] / J. Hoffstein, J. Pipher, J. H. Silverman. – Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.11182.9999, свободный.

10. Сараев П. В. Основы использования математического пакета MAPLE в моделировании: Учебное пособие / П. В. Сараев / Международный институт компьютерных технологий. – Липецк, 2006. – 119 с.

11. Performance Improvements and a Baseline Parameter Generation Algorithm for NTRUSign / J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. H. Silverman, W. Whyte. / NTRU Cryptosystems. – Burlington, 2005. – 18 p.

12. Горбенко Ю. Інфраструктури відкритих ключів. Системи ЕЦП. Теорія та практика / Ю. Горбенко, І. Горбенко. – Харків: Форт, 2010, - 593 с.

13. Таранников Ю. В. Комбинаторные свойства дискретных структур и приложения к криптологии / Ю. В. Таранников. – К.:Литрес, 2012, - 153 с.

14. Hoffstein Jeffrey. NSS: The NTRU Signature SchemeNTRU Cryptosystems [Электронный ресурс] / Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman. – Режим доступа: http://www/citeseerx.ist.psu.edu, свободный.

15. Nguyen P. Q. Learning a Zonotope and More: Cryptanalysis of NTRUSign countermeasures [Электронный ресурс] / L. Ducas, P. Q. Nguyen. – Режим доступа: http://www.di.ens.fr/ducas/-NTRUSignCryptanalysis/Ducas-Nguyen/Learning.pdf, свободный.

16. Min Sung Jun. Weak property of malleability in NTRUSign [Электронный ресурс] /Sung Jun Min, Go Yamamoto, and Kwangjo Kim. – Режим доступа: http://www.academy.ualiberty.com/ru/goodsquality/details/174, свободный.

17. Napias, Huguette. A generalizaion of the LLL algorithm over euclidean rings or orders [Электронный ресурс] / Napias, Huguette – Режим доступа: http://www.numdam.org/item?id=JTNB_1996__8_2_387_0, свободный

18. Правила улаштування електроустановок: Нормативне виробничо-практичне видання : вид. 5-те, перероблене й доповнене : затв. М-вом енергетики та вугільної промисловості України 20.06.14 : введення в дію з 20.11.14. – Х. : Міненерговугілля України, 2014. – 793 с. ; 24 см. – 5000 прим.

Приложение А

ДОКУМЕНТ ДЛЯ ПОДПИСИ

Гендиректору А. В. Серебрякову

ООО «Нафтогаз»

Бухгалтера Филимонова Ф.Ю

ЗАЯВЛЕНИЕ

Прошу разрешения на взятие кредита у банка «Восточный» на сумму не менее 20.000.000 у. е. для покрытия необходимых расходов в этом месяце до 30.04.2014. Просьба уведомить меня по электронной почте :Filimonof@mail.com или по телефону 096-964-45-55.

Дата: 12.04.2014 Подпись

Приложение Б

ЛИСТИНГ ПРОГРАММЫ

```
with(PolynomialTools);
with(LinearAlgebra);
with(MTM);
with(StringTools);
with(ListTools);

Hmesg := proc (path, N, r) local doc, n, DB, str, Hsh128, H2Dec, lenH, HDec2S,
        H2Ar, mV, m, imax, r256;
doc := fopen(path, READ, BINARY);
n := FileTools:-Size(doc);
DB := readbytes(doc, n);
fclose(doc);

if 0 < r then
r256 := Reverse(convert(r, base, 256));
DB := [op(DB), op(r256)] end if;
str := convert(DB, string);
Hsh128 := Hash(str);
H2Dec := convert(Hsh128, decimal, hex);
lenH := length(H2Dec);
HDec2S := convert(H2Dec, string);
H2Ar := [seq(parse(HDec2S[i]), i = 1 .. lenH)];
H2Ar := ceil((1/5)*H2Ar*q);
H2Ar := [seq(H2Ar[z]-q, z = 1 .. lenH)];
imax := ceil(2*N/lenH);
mV := Vector[row]([seq(H2Ar, i = 1 .. imax)]);
return SubVector(mV, 1 .. 2*N)
```

```
end proc;

SignDoc := proc (doc, Secret, Public) local mm, m, P, M, coef, V, dim, M0,
        MSK2, MSKj2, mMSKj2, mMskj, ST, S, T, s, t, b, r;
read "GParams.m";
read Secret;
read Public;
mm := x^N-1;
r := -1; b := Nor+1;
P := [f, g, F, G];
M := Array(1 .. 4);

for i to 4 do

coef[i] := CoefficientVector(P[i], x, termorder = reverse);
V[i] := Vector[row](N);
dim[i] := Dimension(coef[i]);
V[i][N-dim[i]+1 .. -1] := coef[i];
M0 := Matrix(N);

for j to N do

M0[j, 1 .. -1] := V[i];
V[i] := SubVector(V[i], 2 .. -1)

end do;

M[i] := M0

 end do;
```

```
MSK2 := Matrix([[M[1], M[2]], [M[3], M[4]]]);
MSKj2 := MatrixInverse(MSK2);

while Nor < b do

r := r+1;
m := Hmesg(doc, N, r);
mMSKj2 := VectorMatrixMultiply(m, MSKj2);
mMskj := round(mMSKj2);
ST := mMskj.MSK2;
S := SubVector(ST, 1 .. N);
T := SubVector(ST, N+1 .. 2*N);
s := FromCoefficientVector(S, x, termorder = reverse);
t := FromCoefficientVector(T, x, termorder = reverse);
s := `mod`(s, q);
t := `mod`(rem(s*h, mm, x), q);
b := Verify(s, t, m)

end do;

save s, t, "Signature.m";
print(`s=`, s);
print(`t=`, t);
print(`r=`, r);

end proc;

Verify := proc (s, t, m) local m1d, m2d, m1, m2, S0, T0, Vs, Vt, V;
m1d := SubVector(m, 1 .. N);
m2d := SubVector(m, N+1 .. 2*N);
```

```
m1 := Vector[row](Reverse(convert(m1d, list)));
m2 := Vector[row](Reverse(convert(m2d, list)));
S0 := CoefficientVector(s, x);
T0 := CoefficientVector(t, x);
Vs := Vector[row](N, S0);
Vt := Vector[row](N, T0);
V := Vector[row]([Vs-m1, Vt-m2]);
return evalf((1/27)*Norm(V, 2))

end proc;

VerifyDoc := proc (doc, Sig, pub) local mm, m, b, t;
read Sig;
read pub;
read "GParams.m";
mm := x^N-1;
m := Hmesg(doc, N, r);
t := `mod`(rem(s*h, mm, x), q);
b := Verify(s, t, m);
if b <= Nor then print("Signature Valid", "b=", b, "<", "Norm=", Nor)
        else print("Signature Invalid:", "b=", b, ">", "Norm=", Nor)
end if;
end proc;

KeyGen := proc (j) local i, mm, RfRgcd, qq, f, g, F, G, ff, fmod, h, Rf, Rg, alf, bet,
          rof, rog, fcoef, gcoef, fV, gV, fx, gx, dd, cn, c, key, pub;
read "GParams.m";
if j <= 0 then return `Enter i!`
end if;
if 10 < j then j := 10
```

```
end if;

for i to j do

mm := x^N-1;
RfRgcd := -9;
qq := 0;

while RfRgcd <> 1 or qq <> q do

f := randpoly(x, degree = N-1);
g := randpoly(x, degree = N-1);
f := mods(f, q);
g := mods(g, q);
gcdex(f, mm, x, 'ff');

try
fmod := `mod`(ff, q);
catch: next;
error
end try;

Rf := resultant(f, mm, x);
Rg := resultant(g, mm, x);
RfRgcd := igcdex(Rf, Rg, 'alf', 'bet');
gcdex(alf*f, bet*g, x, 'rof', 'rog');
F := -q*bet*rog;
G := q*alf*rof;
qq := expand(f*G-F*g)
```

```
end do;

h := `mod`(rem(fmod*g, mm, x), q);
fcoef := CoefficientVector(f, x);
gcoef := CoefficientVector(g, x);
fV := Vector[row](N, fcoef);
gV := Vector[row](N, gcoef);
fx := FromCoefficientVector(fV, x, termorder = reverse);
gx := FromCoefficientVector(gV, x, termorder = reverse);
dd := rem(fx*F+gx*G, mm, x);
cn := rem(f*fx+g*gx, mm, x);
c := round(quo(dd, cn, x));
F := F+c*f;
G := G+c*g;
key := cat("SecretKey", i, ".m");
pub := cat("PublicKey", i, ".m");
save f, g, F, G, h, key;
save h, pub;
return f, g, F, G, h

end do

end proc;

save Hmesg, Verify, SignDoc, VerifyDoc, KeyGen, "NTRU.m";
```