Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: Розроблення алгоритмів несиметричного шифрування: схема Мак-Еліса на еліптичних кодах для мобільних засобів зв'язку

Виконав: студент ІІ курсу, групи СБм-61
спеціальності 125 Кібербезпека
(шифр і назва спеціальності)

(підпис) Сава Л.М. (прізвище та ініціали)

Керівник (підпис) Александер М. Б. (прізвище та ініціали)

Нормоконтроль (підпис) Лечаченко Т.А. (прізвище та ініціали)

Завідувач кафедри (підпис) Загородна Н.В. (прізвище та ініціали)

Рецензент (підпис) (прізвище та ініціали)

Тернопіль
2023

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«____» _____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 125 Кібербезпека
(шифр і назва спеціальності)

Студенту Саві Луці Михайловичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення алгоритмів несиметричного шифрування: схема Мак-Еліса на еліптичних кодах для мобільних засобів зв'язку

Керівник роботи Александер Марек Богуслав, д.т.н., проф. кафедри КБ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «16» листопада 2023 року № 4/7-1061
2. Термін подання студентом завершеної роботи 14 грудня 2023р.
3. Вихідні дані до роботи

4. Зміст роботи (перелік питань, які потрібно розробити):
1. Дослідження засобів і механізмів забезпечення безпеки інформації в КМ
2. Основні принципи побудови теоретико-кодових схем
3. Проєктування та тестування програмного пакету "Забезпечення конфіденційності та вірогідності даних, що обробляються в засобах мобільного зв'язку"
4. Охорона праці та безпека в надзвичайних ситуаціях

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1 Титульна сторінка 2 Мета. завдання дослідження. наукова новизна
3 Аналіз сучасного стану 4 Протокол SSL 5 Протокол TLS
6 Аналіз протоколів забезпечення інформаційної прихованості та достовірності: протокол IPSEC, 7 Класифікація загроз безпеці в КМ, 8 Крипто-кодові конструкції
9 Дослідження властивостей крипто-кодових конструкцій 10 – Модель бізнес-процесів
11 Результати експериментальних досліджень статистичної безпеки
12 Діаграма вартіантів використання 13 Логічна модель БД 14 Діаграма станів 15 Діаграма Класів 16 Висновки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Осухівська Г.М., к.т.н., доцент		
Безпека в надзвичайних ситуаціях	Клепчик В.М., проректор з адміністративно-господарської роботи та будівництва		

7. Дата видачі завдання 16 листопада 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	16.11.2023-17.11.2023	*Виконано*
2.	Підбір наукових джерел про загрози для засобів		
	мобільного зв'язку	18.11.2023-20.11.2023	*Виконано*
3.	Переклад та опрацювання наукових джерел про		
	теоретико-кодові схеми	21.11.2023-23.11.2023	*Виконано*
4.	Розробка програмного пакету для забезпечення		
	конфіденційності та вірогідності даних		
	на основі схеми Мак-Еліса	24.11.2023-27.11.2023	*Виконано*
5.	Оформлення розділу «Дослідження засобів та		
	механізмів забезпечення властивостей безпеки		
	Інформації в комп'ютерних мережах»	28.11.2023-30.11.2023	*Виконано*
6.	Оформлення розділу «Основні принципи побудови		
	теоретико кодових схем»	01.12.2023-04.12.2023	*Виконано*
7.	Оформлення розділу «Проєктування програмного		
	пакету «Забезпечення конфіденційності та		
	вірогідності даних, що обробляються в засобах		
	мобільного зв'язку»	05.12.2023-09.12.2023	*Виконано*
8.	Виконання завдання до підрозділу «Охорона праці»	10.12.2023-11.12.2023	*Виконано*
9.	Виконання завдання до підрозділу «Безпека в надзви-		
	чайних ситуаціях»	12.12.2023-13.12.2023	*Виконано*
10.	Оформлення кваліфікаційної роботи	14.12.2023-15.12.2023	*Виконано*
11.	Нормоконтроль	16.12.2023	*Виконано*
12.	Перевірка на плагіат	18.12.2023	*Виконано*
13.	Попередній захист кваліфікаційної роботи	20.12.2023	*Виконано*
14.	Захист кваліфікаційної роботи	27.12.2023	

Студент _____ Сава Л.М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Александер Марек Богуслав
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розроблення алгоритмів несиметричного шифрування: схема Мак-Еліса на еліптичних кодах для мобільних засобів зв'язку // Кваліфікаційна роботи рівня «Магістр» // Сава Лука Михайлович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем та програмної інженерії, кафедра кібербезпеки, група СБм–61 // Тернопіль, 2023 // с. - , рис. - 64 , табл. – 12, бібліогр. – .

Ключові слова: МАК-ЕЛІС, ТЕОРЕТИКО-КОДОВІ СХЕМИ, АЛГЕБРО-ГЕОМЕТРИЧНІ КОДИ, ДОСТОВІРНІСТЬ, СКРИТНІСТЬ, ЕЛІПТИЧНІ КРИВІ.

Об'єкт дослідження – процес підвищення вірогідності та конфіденційності переданих даних за допомогою несиметричної крипто-кодової системи Мак-Еліса.

Предмет дослідження – алгоритми підвищення вірогідності та конфіденційності переданих даних за допомогою несиметричної крипто-кодової системи Мак-Еліса та їх дослідження.

Мета дослідження – підвищення вірогідності та конфіденційності переданих даних за допомогою несиметричної крипто-кодової системи Мак-Еліса на еліптичних кодах у мобільних каналах зв'язку.

Розроблено програмний продукт який дозволяє інтегровано забезпечити вимоги щодо достовірності та інформаційної скритності в протоколах з автоперезапитом або з прямим виправленням помилок.

Результати можуть бути впроваджені в комп'ютерних мережах в протоколи з автоперезапитом на основі використання теоретико кодової схеми Мак-Еліса.

ANNOTATION

Development of Asymmetric Encryption Algorithms: Elliptic Curve MacEliece Scheme for Mobile Communication Devices // Qualification paper of the educational level “Master” // Luka Sava // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cyber Security, SBm-61 group // Ternopil, 2023 // P. , fig. - 64 , tables - 12 , annexes - , references - .

Key words: MC-ELICE, THEORETICAL CODE SCHEMES, ALGEBRA-GEOMETRIC CODES, RELIABILITY, SKILLS, ELLIPTIC CURVES.

The object of the study is the process of increasing the reliability and confidentiality of transmitted data using the asymmetric crypto-code system of McElis.

The subject of the study is algorithms for increasing the reliability and confidentiality of transmitted data using the asymmetric crypto-code system of McElice and their research.

The purpose of the study is to increase the reliability and confidentiality of transmitted data using the asymmetric McElice crypto-code system on elliptic codes in mobile communication channels.

A software product has been developed that allows for the integrated provision of requirements for reliability and information secrecy in protocols with autorequest or with direct error correction.

The results can be implemented in computer networks in protocols with auto-request based on the use of the theoretical code scheme of McElice.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ТКС – теоретико-кодові схеми;

UML – Unified Modeling Language;

БД – база даних;

БСШ – блочно-симетричні шифри;

IPSec – Internet Protocol Security;

TLS – Transport Layer Security;

ПЗ – програмне забезпечення.

ЗМІСТ

ВСТУП.....

1 ДОСЛІДЖЕННЯ ЗАСОБІВ І МЕХАНІЗМІВ ЗАБЕЗПЕЧЕННЯ ВЛАСТИВОСТЕЙ БЕЗПЕКИ ІНФОРМАЦІЇ В КОМП'ЮТЕРНИХ МЕРЕЖАХ .....

1.1 Анал із протоколів забезпечення конфіденційності та цілісності даних..... .....

1.1.1 Протокол SSL .....

1.1.2 Протокол TLS.....

1.1.3 Протокол IPSec .....

1.2 Аналіз загроз в комп'ютерних мережах..... .....

1.3 Аналіз механізми забезпечення достовірності даних..... .....

1.4 Механізми забезпечення інформаційної скритності даних..... .....

1.5 Постановка задач дослідження…….

2 ОСНОВНІ ПРИНЦИПИ ПОБУДОВИ ТЕОРЕТИКО-КОДОВИХ СХЕМ ....

2.1 Загальна характеристика теоретико-кодових схем ……..……

2.2 Система Рао-Нама…….…….

2.3 Система Мак-Еліса ……..……

2.4 Система Нідеррайтера……

3 ПРОЕКТУВАННЯ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ПАКЕТУ "ЗАБЕЗПЕЧЕННЯ КОНФІДЕНЦІЙНОСТІ ТА ВІРОГІДНОСТІ ДАНИХ, ЩО ОБРОБЛЯЮТЬСЯ В ЗАСОБАХ МОБІЛЬНОГО ЗВ'ЯЗКУ"…….……..…….....……..……..

3. 1 Опис предметної області……

3.2. Розроблення специфікацій бізнес-вимог до модуля "Забезпечення конфіденційності та вірогідності даних"……..

3.2.1 Опис функціональних вимог…….

3.2.2 Опис постановки комплексу задач модуля "Забезпечення конфіденційності та вірогідності даних"……

3.2.2.1 Характеристика комплексу задач програмного продукту "Забезпечення конфіденційності та вірогідності даних"……..

3.2.2.2 Вихідна інформація…….

3.2.2.3. Вхідна інформація……..

3.3 Математична постановка комплексу задач модуля……

3.4 Розроблення не функціональних вимог для модуля “Забезпечення конфіденційності та вірогідності даних, що оброблюються в інформаційних системах”…….

3.4.1 Надійність, безпека та захист інформації……

3.4.2 Опис архітектури системии……..

3.5 Опис інформаційних потоків……

3.6 Проектування структури бази даних…….

3.7 Розроблення програмного продукту “Криптосистема МакЕліса на еліптичних кривих”…….

3.7.1 Розроблення інтерфейсу програмного продукту……

3.7.2 Склад та взаємодія програмних модулів…….

3.7.3 Заходи щодо забезпечення захисту інформації, реалізовані у програмному продукту……..

3.8 Результати тестування програмного продукту…….

3.9 Контрольний приклад…….

3.10 Впровадження та супровід програмного продукту …….

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ……...

4.1 Охорона праці ……

4.2 Безпека життєдіяльності……

ВИСНОВКИ .....

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ……..…

ДОДАТКИ .....

ВСТУП

Швидкий приріст обсягів оброблюваних даних та прогрес у сфері обчислювальної техніки ставлять нові вимоги до надійності та забезпечення безпеки даних у комп'ютерних мережах та засобах мобільного зв'язку. Основні механізми забезпечення інформаційної скритності й достовірності інформації у комп’ютерних мережах на різних рівнях засновані на використанні протоколів автоперезапиту та блочно-симетричних механізмах шифрування (методах симетричної та несиметричної криптографії) [1, 3].

Методи симетричної криптографії засновані на простих та легко реалізуємих блоках підстановок і перестановок. Однак вони вимагають дуже складної та коштовної системи розподілу таємних ключових даних. Методи криптографії з відкритим ключем позбавлені цього недоліку, вони засновані на використанні відповідної теоретико-складної задачі (факторизації, дискретного логарифмування і т.п.), однак дуже складні в реалізації та не забезпечують вимоги щодо оперативності обробки даних [1, 3].

Серед відомих прикладів несиметричних криптосистем особливе місце займають несиметричні секретні системи доказової стійкості (теоретико-кодові схеми) на алгебраїчних блокових кодах, що мають істотну перевагу – високу швидкість криптографічного перетворення інформації і можливість реалізувати криптографічне перетворення із використанням відкритих ключів.

Одним з найбільш ефективних засобів захисту переданих даних від виникаючих помилок є використання спеціальних процедур, заснованих на застосуванні завадостійких (коригувальних) кодів, суть яких складается у внесенні по певному алгоритму в дані, що передаються певної збитковості (перевірочної частини). Завадостійке кодування дозволяє виправляти помилки і при його використанні потужність сигналу можно знизити, лишаючи швидкість передачі інформації незмінною.

Проведений аналіз показав, що застосування криптографічних засобів захисту інформації доказової стійкості (деяких джерелах вони отримали назву

теоретико-кодових схем (ТКС)) дозволяє сполучити завадостійке кодування з маскуванням даних під випадкову послідовність, і, таким чином, інтегровано (одним прийомом) забезпечити інформаційну скритність та достовірність даних які використовуються у сучасних комп'терних мережах [1, 3].

Таким чином, перспективним напрямком у розв'язанні протиріч між збільшенням обсягів оброблюваних даних, підвищенням ймовірносно-часових вимог до безпеки та достовірності інформації, надійності її одержання та існуючих підходів теорії захисту інформації щодо побудови підсистем обміну конфіденційними повідомленнями є розробка протоколів, інтегровано забезпечуючих інформаціїну скритність і достовірність інформації, яка передається у комп'ютерних мережах побудованих з використанням теоретико-кодових схем.

Проектування та розроблення модуля буде здійснюватися з використанням сучасних спеціалізованих програмних засобів та систем: MSVS2013 для розробки та NIST SP 800-22 для аналізу та тестування випадковасті шифртексту (кодового слову).

1 ДОСЛІДЖЕННЯ ЗАСОБІВ І МЕХАНІЗМІВ ЗАБЕЗПЕЧЕННЯ ВЛАСТИВОСТЕЙ БЕЗПЕКИ ІНФОРМАЦІЇ В КОМП'ЮТЕРНИХ МЕРЕЖАХ

1.1 Аналіз протоколів забезпечення конфіденційності та цілісності даних

За безпеку інформації в комп'ютерних мережах, на сьогоднішній день відповідають протоколи SSL, TLS та IPSec.

1.1.1 Протокол SSL

З кожним роком число різних сервісів, що надаються в мережі Internet, зростає. Серед них активно розвивається і електронна комерція та інші сервіси, які використовують в свій роботі конфіденційну інформацію користувача, наприклад номер кредитної карти і т.д., в результаті виникає необхідність захисту цієї інформації від несанкціонованого доступу з боку сторонніх людей.

Протокол SSL (secure socket layer) забезпечує захист даних між сервісними і транспортними протоколами. Часто для нього використовується абревіатура HTTPS. Саме ця латинська літера «s» вказує, що використовується не звичайний, а захищений канал передачі даних по протоколу HTTP [1, 3].

Рівень захищених сокетів (SSL) — це протокол, який забезпечує безпечний зв'язок через Інтернет. Він використовує як симетричну, так і асиметричну криптографію. Протокол SSL забезпечує автентифікацію сервера та автентифікацію клієнта:

– Аутентифікація сервера виконується, коли клієнт підключається до сервера. Після початкового рукостискання сервер надсилає свій цифровий сертифікат клієнту. Клієнт перевіряє сертифікат сервера або ланцюжок сертифікатів.

– Автентифікація клієнта виконується, коли сервер надсилає запит на сертифікат клієнту під час рукостискання. Якщо сертифікат або ланцюжок клієнта

перевірено та повідомлення про перевірку сертифіката перевірено, рукостискання триває далі.

– Додаткова додаткова автентифікація виконується шляхом порівняння загального імені в сертифікаті з повним іменем домену сервера за допомогою зворотного пошуку на сервері доменних імен (DNS), де можна отримати повне ім’я домену сервера.).

Ось основні принципи,за якими працює SSL:

– Процес рукостискання (Handshake Process) Коли користувач отримує доступ до сайту з підтримкою SSL, його браузер запитує сертифікат SSL сайту. Цей сертифікат, виданий центром сертифікації (CA), підтверджує автентичність веб-сайту.

– Обмін ключами (Key Exchange) Після перевірки сертифіката браузер і сервер узгоджують метод шифрування та обмінюються унікальними ключами шифрування.

– Шифрування та передача (Encryption and Transmission): усі дані, якими обмінюється користувач та сервер - зашифровані, що робить їх нечитабельними для будь-яких потенційних перехоплювачів.

– Розшифровка та відображення (Decryption and Display): одержувач розшифровує дані, що надійшли, за допомогою попередньо узгоджених ключів шифрування та відображаються в оригінальному форматі.

Дослідження протоколу SSL показали, що він починає використовуватись в той момент, коли користувач починає вводити адресу URL, що починається з абревіатури HTTPS у відповідному рядку свого браузера. За замовчуванням, HTTPS з’єднання використовує TCP порт 443. HTTP, який ввадається несек’юрним протоколом, використовує порт 80. Клієнтом будемо вважати браузер користувача. Він посилає серверу умовну вітальну (hello message). У свою чергу сервер, також посилає клієнту назад своє зворотнє повідомлення. Ці початкові повідомлення, є первинними та містять інформацію, використовувану при подальшій настройці відкривається секретного каналу.

Попередньо генеруються два випадкових числа, які є параметрами протоколу. І сервер, і клієнт, генерують такі числа незалежно один від одного, а потім, на основі цих чисел рахують деякі відкриті параметри і обмінюються ними один з одним. Після того, як сервер отримав повідомлення запит на спілкування від клієнта, то він відсилає свій сертифікат, якщо такий у нього є.

На практиці, процес обміну ключами та сертифікатами, іноді може бути досить тривалим. З цією метою, часто передбачається можливість повторного використання одних і тих же ідентифікаційних даних.

Аналіз протоколу показав, що SSL можна умовно розділити на декілька модулів (підпротоколів) по меті їх використання (рис. 1.1):

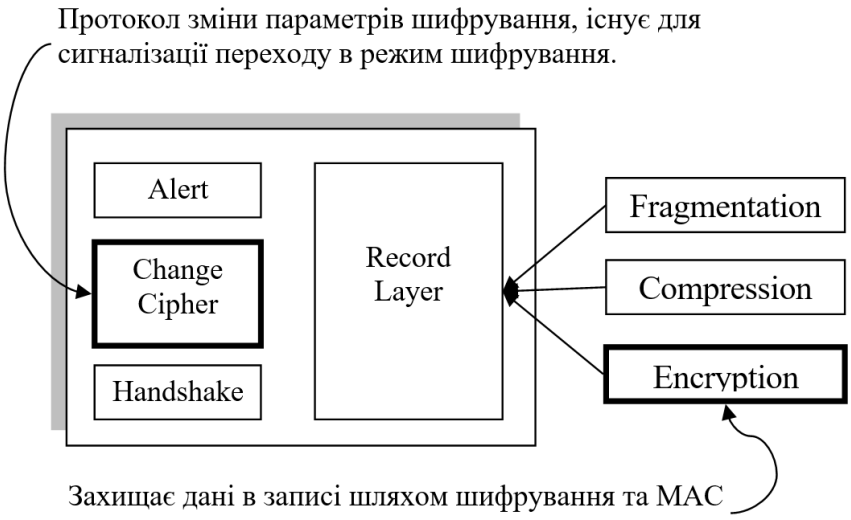


Рисунок 1.1 – Структура протоколу SSL

Роботу протоколу SSL можна розділити на два рівні. Перший рівень – шар протоколу формування спільних сеансових ключів (Handshake Protocol Layer). Він складається із трьох підпротоколів: протокол підтвердження підключення (Handshake Protocol), протокол зміни параметрів шифру (Change Cipher Spec Protocol) та попереджувальний протокол (Alert protocol). Другий рівень – це шар протоколу запису.

1.1.2 Протокол TLS

Первинною метою протоколу TLS (Transport Layer Security), як і його попередника SSL, є забезпечення конфіденційності та цілісності даних при комунікації двох додатків. Протокол має два рівні: протокол записів TLS і протокол діалогу TLS. На нижньому рівні, працюючому поверх надійного транспортного протоколу (наприклад, TCP), розміщується протокол записів TLS. Існують певні відмінності між TLS та SSL, але, загалом це один і той самий протокол.

Однією з переваг TLS є те, що він не залежить від протоколу додатку. Протоколи верхнього рівня можуть розміщуватися поверх протоколу TLS прозорим чином. Стандарт TLS, однак, не визначає те, як протоколи збільшують безпеку за допомогою TLS. рішення про те, як ініціалізувати TLS-діалог і як інтерпретувати сертифікати аутентифікації, залишається на розгляд розробників протоколів і програм, які працюють поверх TLS.

Структурна схема роботи протоколу TLS зображена на рис. 1.2. Спочатку клієнт ініціалізує обмін повідомленнями. Сервер відсилає свій сертифікат

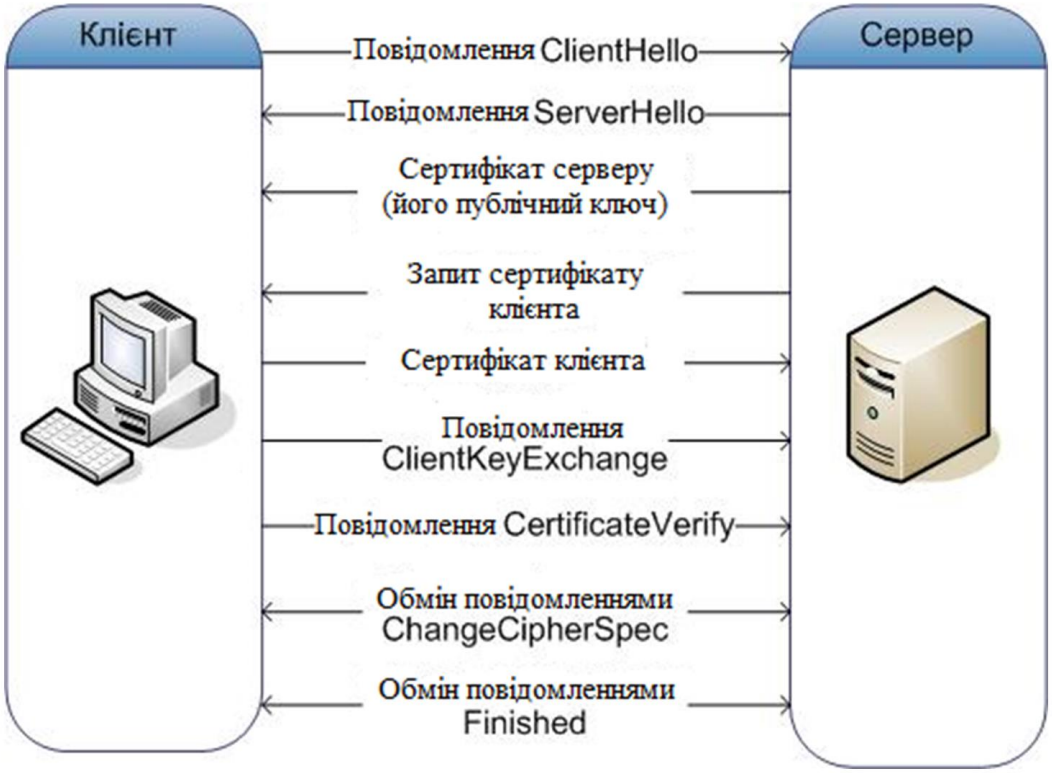


Рисунок 1.2– Робота протоколу TLS в режимі логічного з’єднання

1.1.3 Протокол IPSec

Internet Protocol Security (IPSec) – це набір протоколів для захисту даних, що передаються через Інтернет або будь-яку публічну мережу. IPsec спочатку включав 3 базових протоколи, опубліковані як RFC-документи для захисту IP-пакетів: Security Architecture for the Internet Protocol, Authentication Header (AH) і Encapsulating Security Payload (ESP) (RFC1825, 1826 і 1827). Перший описує механізми безпеки для IPv4 та IPv6, другий – призначений для забзпеченя цілісності даних, а третій – для шифрування та перевірки автентичності даних.

Ядро IPSec складається з трьох протоколів (рис. 1.3):

Рисунок 1.3 – Структура протоколу IPSec

Існує два режими функціонування IPSec: транспортний та тунельний. У транспортному режимі зашифровується лише вміст IP-пакета, вихідний заголовок при цьому залишається без змін. Місця розміщення додаткової інформації, що вставляється протоколами в пакет, представлені на рис. 1.4.

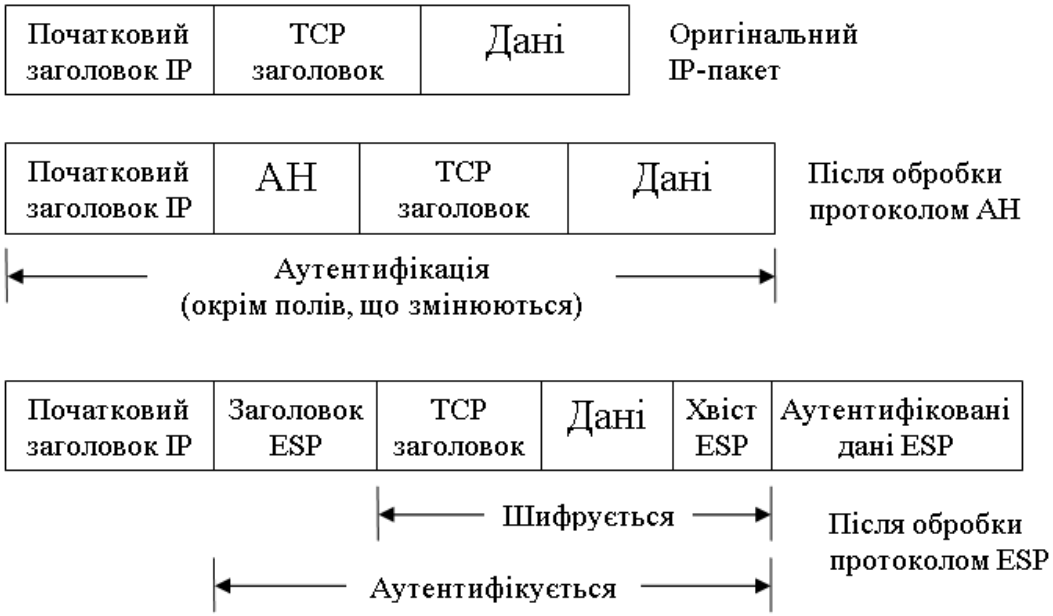


Рисунок 1.4 – Транспортний режим протоколу IPSec

У тунельному режимі весь вихідний IP-пакет, а саме: дані, заголовок, маршрутна інформація підлягає шифруванню (рис. 1.5)

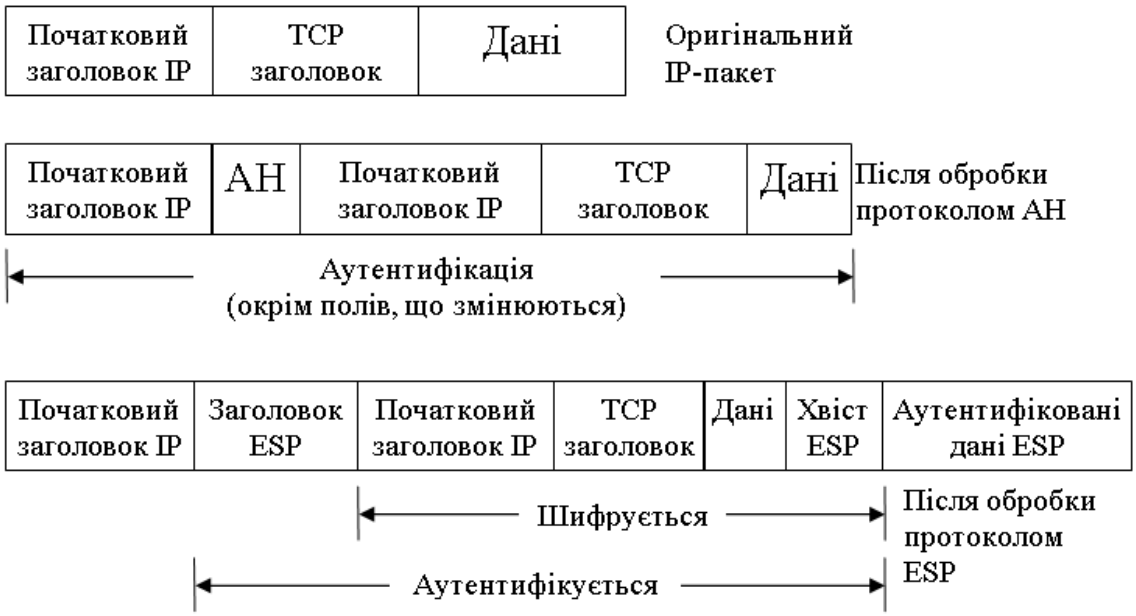


Рисунок 1.5 – Тунельний режим протоколу IPSec

Кожен з протоколів має свої переваги та недоліки . ESP забезпечує конфіденційність даних, але не повну аутентифікацію всього пакета. AH повністю аутентифікує, але дані при цьому залишаються відкритими. Для забезпечення належного рівня безпекирекомендують поєднувати протоколи.

Режими IPsec не є взаємовиключними. На одному і тому ж вузлі деякі SA можуть використовувати транспортний режим, а інші – тунельний.

1.2 Аналіз загроз в комп'ютерних мережах

Кількість користувачів сучасних комп'ютерних мереж зростає з колосальною швидкістю, постійно розширюється спектр та сфера послуг, які надаються, люди купують, працюють, спілкуються, навчаються, розважаються в Інтернет [1, 13, 24]. Ці тенденції призводять до різкого підвищення навантаження на мережі. Тому поруч з переваги зростають і вимоги до безпеки, під якими розуміють здатність забезпечити наступні властивості інформації: цілісність, автентичність і конфіденційність даних. Рівень захищеності від сучасних загроз мережної безпеки без сумнівів впливає на якість послуг, які надаються.

Класифікація загроз в комп'ютерних мережах представлена на рис.1.6.

Одними з найважливіших показників сучасних телекомунікаційних мереж звісно є надійність, відмовостійкість і вартість. Однак, останнім часом все більш актуальною є безпека телекомунікаційної мережі, яка впливає на безпеку конфіденційних даних, які можуть оброблюватись та зберігатись в апаратно-програмному середовище ТКМ. Сьогодні безпека інформації в мережах є проблемою, яка постійно буде вирішуватись але ніколи не буде вирішеною на сто відсотків.

Рисунок 1.6 – Класифікація загроз в комп'ютерних мережах

За даними компанії «Arbor Networks», що займається питаннями безпеки комп'ютерних мереж в США, цілі нападів в більш загальному виді зображено на рис. 1.7. Клієнти залишаються метою номер один. Сервісна інфраструктура та атаки на мережеву інфраструктуру ділять друге та третє місця [47].

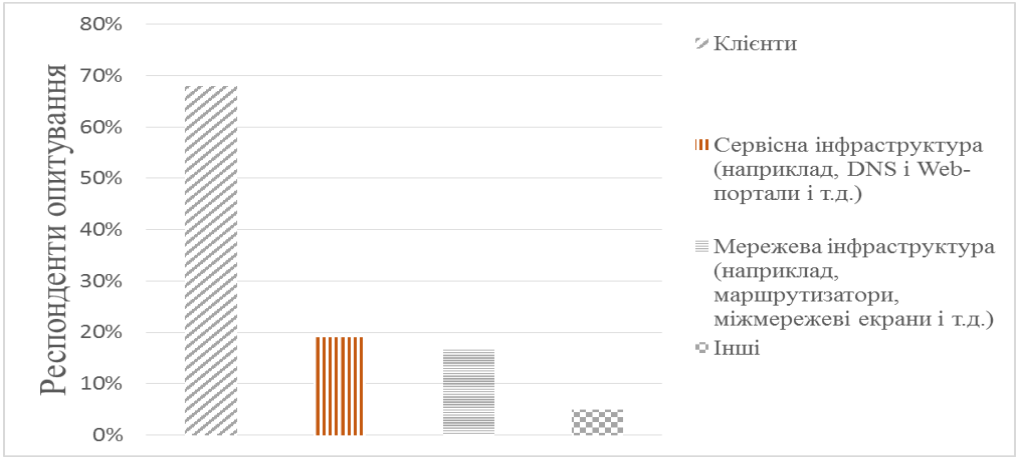


Рисунок 1.7 – Розподіл цілей кібер-злочинів

Дивлячись на послуги рівня додатків мішенню більш таємних атак є дві послуги обміну: HTTP і DNS. Три чверті респондентів зазнали атак на обидві ці послуги (рис. 1.8).

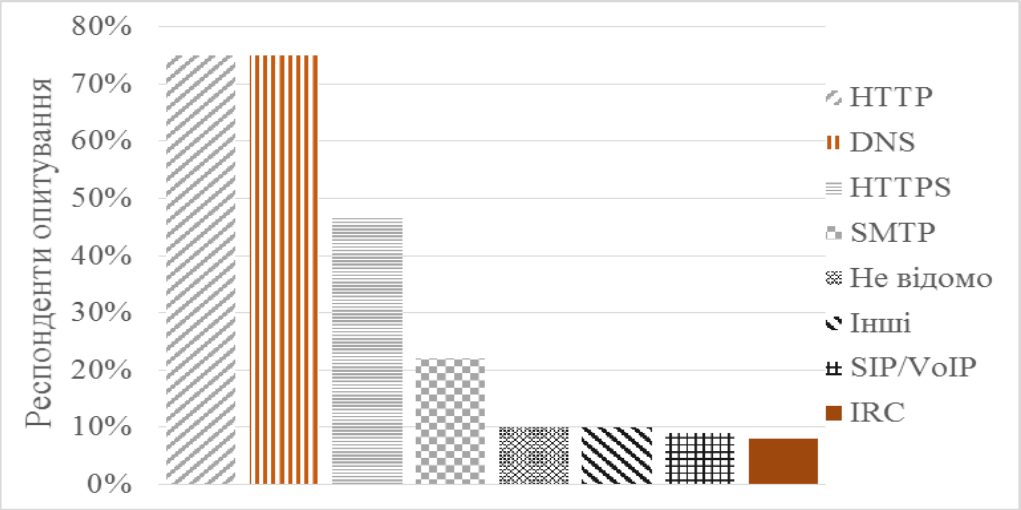


Рисунок 1.8 – Цілі атак на прикладному рівні

Що особливо цікаво в цих результатах, і, можливо, всупереч загальному сприйняттю – це зменшення частки респондентів, що зазнали атак на зашифровані веб-служби (HTTPS). Зниження в цьому році може, в якійсь мірі, бути пов'язано зі збільшенням використання/посилення DDoS атак. Проведений аналіз нападів на послуги конфіденційності та цілісності показав, що атаки підрозділяються на чотири категорії [47]:

– атаки, орієнтовані на обмін даними за протоколами SSL/TLS;

– атаки націлені на стан з’єднання (декількох з’єднань);

– об’ємні атаки, які просто перевантажують трафік в портах послуг;

– атаки прикладного рівня, які націлені на основну послугу беспосередньо за повністю узгодженим SSL/TLS з’єднанням.

За даними минулорічного опитування, приблизно одна п'ята частина респондентів відчувають напади, принаймні однієї категорії, але майже дві третини не знають, які напади відбуваються. Остання статистика непокоїть: це вказує на обмежену видимість і виявлення атак для зашифрованого трафіку. Враховуючи, що

ці послуги часто використовуються у фінансових додатках та для електронної комерції, успішна атака може мати значні фінансові та репутаційні наслідки.

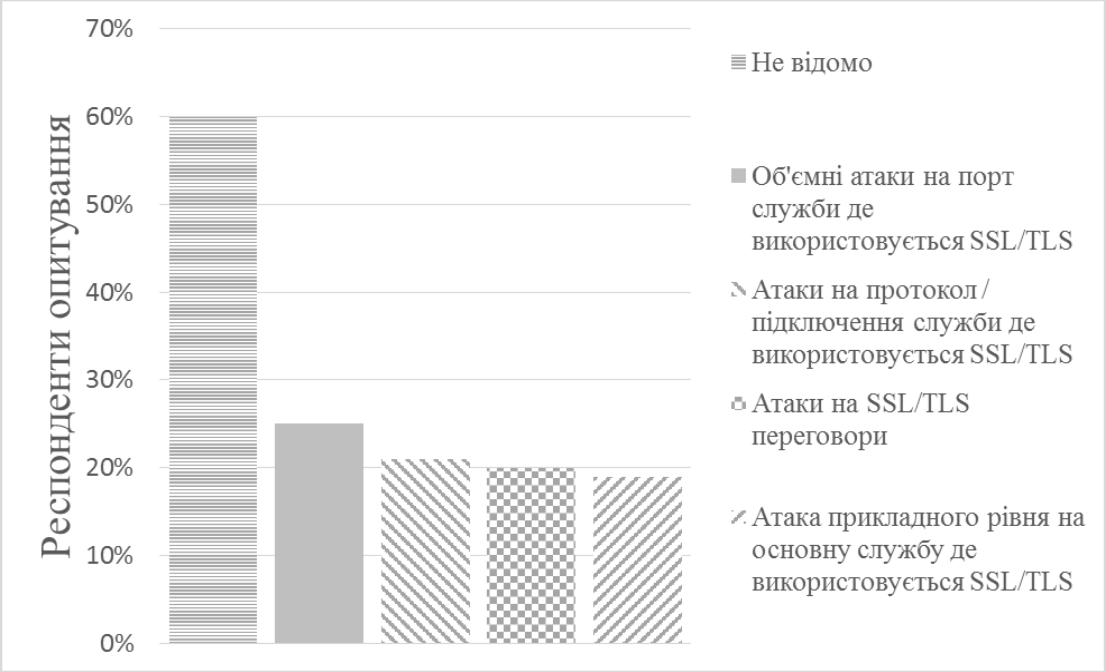


Рисунок 1.9 – Типи атак на послуги конфіденційності та цілісності

Проведений аналіз показав, що лідируючу позицію з реалізації загроз мережної безпеки займають порушення роботи SSL/TLS та IPSec протоколів. Зі 100 випадків реалізації загроз безпеки 87 стосуються порушення конфіденційності, цілісності та автентичності даних. Потрібно дослідити новітні механізми забезпечення достовірності та скритності даних, що дозволять покращити безпеку вищеназваних протоколів.

1.3 Аналіз механізми забезпечення достовірності даних

Проведений аналіз механізмів забезпечення достовірності показав, що дана задача вирішується шляхом використання у протоколах методів завадостійкого кодування.

За своєю структурою всі завадостійкого коди діляться на лінійні та нелінійні. На рис. 1.10 схематично зображена класифікація найбільш відомих

методів завадостійкого кодування. Їх розвиток йшов за двома основними напрямками. Перший напрямок призвів до появи неблокових кодів нескінченної довжини, які можна описати деревом і декодувати за допомогою алгоритмів пошуку по дереву.

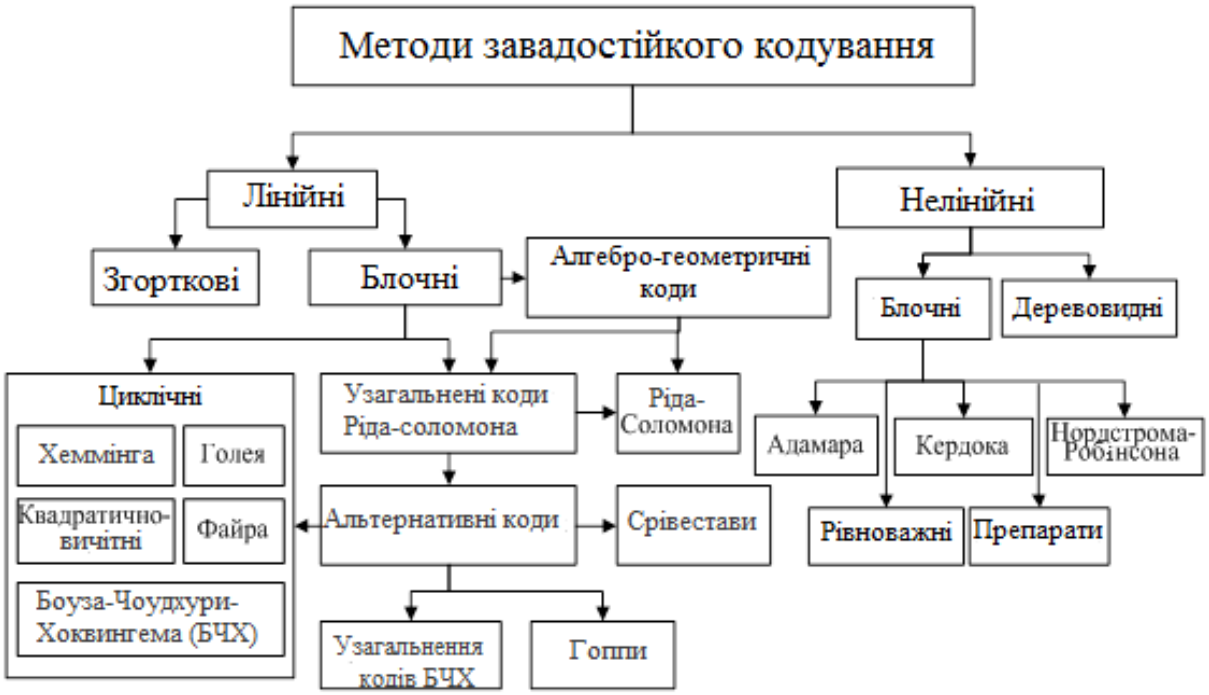


Рисунок 1.10 – Класифікація методів завадостійкого кодування

Щоб спроектований код, що виправляє помилки, був ефективним, необхідно враховувати надмірність, що додається у формі додаткових символів. Зокрема, надмірність повинна працювати таким чином, щоб робити схожі кодові слова більш різними і давати можливість відрізняти помилкову комбінацію від правильної.

Розбіжності між кодовими словами можна вимірювати з використанням метрики-відстані. Для того, щоб виправити помилку, слід порівняти вістані від отриманого кодового слова до всіх можливих кодових слів, і вибрати найближче кодове слово.

При передачі інформації каналом зв’язку ймовірність помилки залежить від співвідношення сигнал/шум на вході демодулятора, таким чином, при постійному рівні шуму остаточний вплив має потужність передавача. Крім того, в певних

системах зв’язку безмірно підвищувати потужність сигналу не дозволяють технічні межі.

Для з'ясування ідеї завадостійкого кодування розглянемо двійковий код, що знайшов на практиці найбільш широке застосування. Довжина коду – це загальна кількість бітів n у кодовій комбінації. Вага кодової комбінації (позначається w) визначається кількістю одиниць у кодовій комбінації.

Відстань між кодами d визначається ступінню відмінності будь-яких двох кодових комбінацій даного коду - числом відмінних позицій або символів, як вага суми за модулем два цих кодових комбінацій.

Як виявилося, завадостійкість кодування забезпечується за рахунок внесення надмірності в кодові комбінації.

Для оцінки потенційної коригувальної здатності (n, k, d)-кодів проводять дослідження їх кодових характеристик порівняно з кодовими межами. Коди, що лежать вище нижнйої межі Варшамова – Гілберта до межі Сінглтона мають добрий енергетичний виграш від кодування. На рис. 1.12 наведені кодові межі Варшамова-Гілберта та Сінглтона.

Під терміном «енергетичний виграш від кодування» розуміють кількісний показник підвищення достовірності передачі інформації в цифровій системі зв'язку при використанні кодування з випадком його відсутності (тобто беззбитковість кодування). Точне його значення показує, наскільки може бути зменшена енергія сигналу при кодуванні в порівнянні з випадком передача не кодованого потоку даних. При цьому повинна забезпечуватися однакова достовірність (оцінюється по ймовірності помилкового рішення) і швидкість передачі в біт/сек. Найпростіший варіант оцінки виграшу від кодування базується на порівнянні мінімуму евклидової відстані між блоками некодованих даних і аналогічної характеристикою для найближчих кодових слів розглянутого блокового коду. Потрібно зазначити, що результат порівняння відстаней залежить від фізичного сигналу, використовуваного для відображення довічного інформаційного символу.

У загальному випадку для виявлення t помилок мінімальна кодова відстань дорівнює:

$$d = t + 1, \tag{1.1}$$

Для кодів, що тільки виправляють помилки, мінімальна кодова відстань дорівнює:

$$d = 2t + 1, \tag{1.2}$$

де t – кількість помилок, що виправляються.

Завадостійкість визначається мінімальним співвідношенням сигнал/шум, необхідним для забезпечення необхідної достовірності. Зафіксуємо співвідношення сигнал шум і вид модуляції. Припустимо, що передача цифрових повідомлень здійснюється по дискретному каналу без пам'яті, тобто помилки в послідовно переданих кодових символах відбуваються незалежно з імовірністю. Тоді ймовірність помилки кратності на довжині блоку буде дорівнювати [1; 3]:

$$P_i = C_n^i P_o^i (1 - P_o)^{n-i}, \tag{1.3}$$

Якщо процедура декодування дозволяє виправити $t = \lfloor (d-1)/2 \rfloor$ помилок, то ймовірність помилкового декодування дорівнює:

$$P_{пом} = \sum_{i=t+1}^{n} P_i = \sum_{i=t+1}^{n} C_n^i P_o^i (1 - P_o)^{n-i}, \tag{1.4}$$

При інтегрованому рішенні задач достовірності та інформаційної скритності передачі даних в КМ теоретико-кодова схема буде виправляти $(1 - \rho) \times t$ виниклих помилок, отже:

$$P_{пом} = \sum_{i=(1-\rho)t+1}^{n} P_i = \sum_{i=(1-\rho)t+1}^{n} C_n^i P_o^i (1-P_o)^{n-i}, \quad (1.5)$$

При кодуванні з надмірністю мінімальне евклідова відстань визначається кодовою відстанню, в результаті чого найближчі кодові вектора збігаються на n – d і відрізняються на d позиціях. Тоді при використанні бінарної фазової модуляції

$$d_{min} E = 2\sqrt{dE_c}, \quad (1.6)$$

У підсумку виграш від кодування визначиться як

$$G = \frac{d_{min}^2 E_2}{d_{min}^2 E_1} = dR, \quad (1.7)$$

Далі оцінимо граничну величину енергетичного виграшу від кодування, скориставшись межами для кодової відстані.

Межа Сінлгтона вказує на максимально досяжний кодова відстань при заданих параметрах коду і записується у вигляді:

$$d \le n - k + 1. \quad (1.8)$$

Коди, що лежать на межі Сінлгтона, називають кодами з максимально досяжним кодовою відстанню (МДР коди).

Межа Варшамова-Гілберта є нижньою кодової межі, тобто вона гарантує існування кодів з параметрами, що лежать на цій межі. Узагальнення межі Варшамова-Гілберта на недвійкові коди має вигляд:

$$q^{n-k} \ge \sum_{i=0}^{d-2} C_{n-1}^i (q-1)^i \text{, або } n - k \ge log_q \left(\sum_{i=0}^{d-2} C_{n-1}^i (q-1)^i \right). \quad (1.9)$$

Результати дослідження енергетичного виграшу від кодування з різними формами модуляції та використанням завадостійкого кодування представленій на рис.1.11. Аналіз показав, що ближче всього до межі Шенона лежать згорткові коди. Зі зростанням довжини недвійкового коду (при ефективних (n, k, d)-параметрах) енергетичний виграш від кодування зростає.

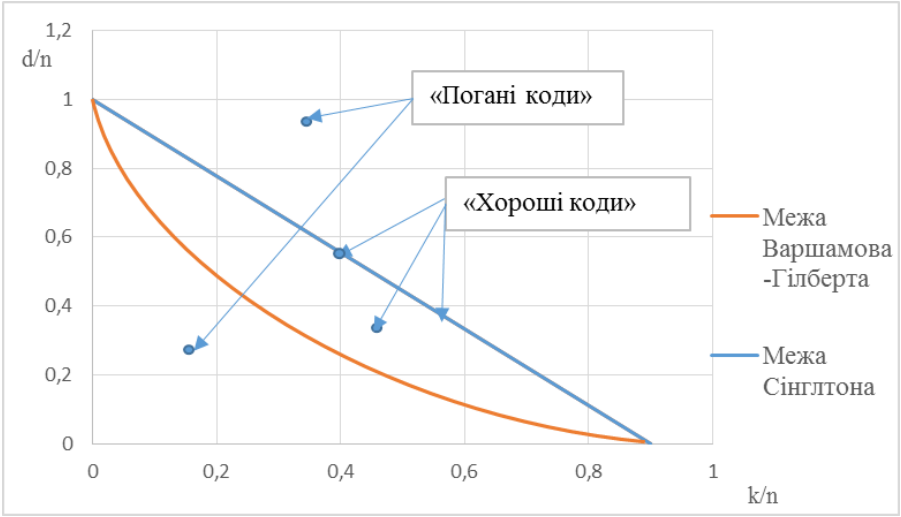


Рисунок 1.11 – Кодові межі Варшамова-Гілберта та Сінглтона

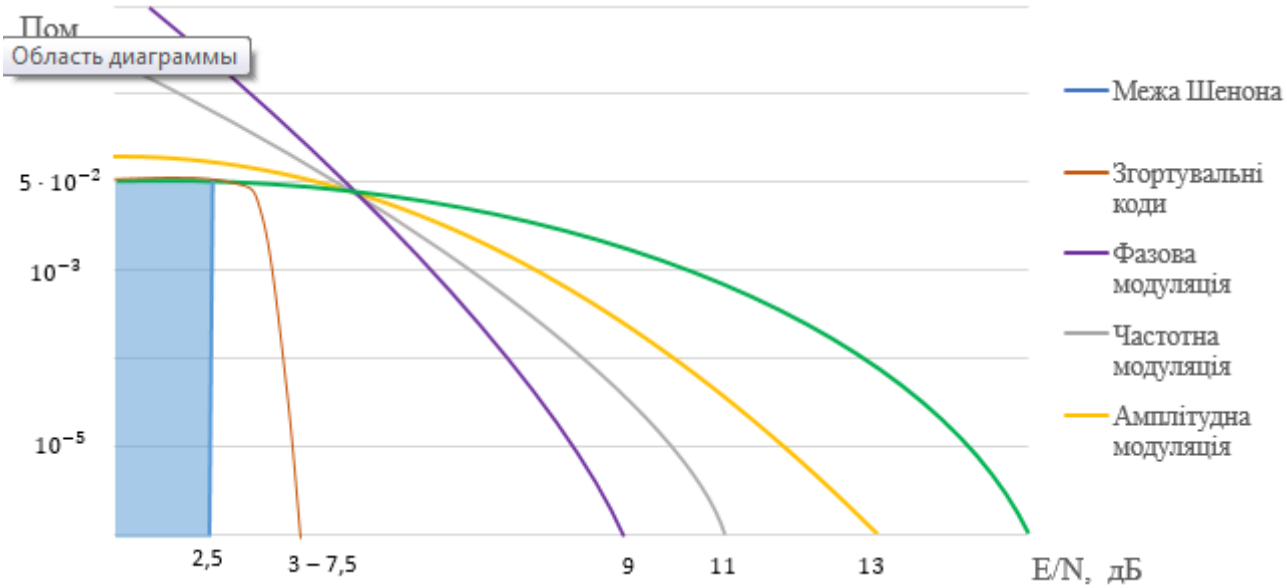


Рисунок 1.12 – Відношення амплітуди до відношення енергії сигналу та шуму

Аналіз показав, що при формуванні ймовірності помилки до 5×10^{-2} використання завадостійкого кодування не дає енергетичний виграш від кодування. Зі збільшенням (n, k, d)-параметрів ЕВК двійкових кодів зменьшується.

1.4 Механізми забезпечення інформаційної скритності даних

Механізми безпеки інформації в комп'ютерних системах і мережах у більшості засновані на криптографічних методах. Загальна класифікація методів криптографічної обробки інформації в телекомунікаційних системах представлена на рис. 1.13. Це методи симетричної та несиметричної криптографії, розвитку яких присвячені роботи [11, 12, 19, 29]. Методи симетричної криптографії засновані на простих та блоках підстановок і перестановок. Методи криптографії з відкритим ключем засновані на використанні відповідної теоретико-числової задачі (факторизації, дискретного логарифмування і т.п.).

Рисунок 1.13 – Методи симетричної і асиметричної криптографії

Існуючі на сьогоднішній день криптографічні засоби захисту інформації доказової стійкості (криптосистеми з відкритим ключем) не забезпечують часових вимог: складність реалізації криптографічних перетворень на 3 – 5 порядків вище, ніж в аналогічних систем тимчасової стійкості (блочно-симетричних шифрів (БСШ)), що в умовах стрімкого збільшення обсягів оброблюваних і переданих даних неприпустимо.

Перспективним напрямком у розвитку криптографічних засобів захисту інформації доказової стійкості є крипто-кодові механізми, стійкість який зводиться до вирішення задачі декодування випадкового коду [7 – 11]. У деяких джерелах вони відомі під назвою теоретико-кодових схем (ТКС).

З літературних джерел відомо, що швидкість крипто-кодового перетворення інформації можна порівняти зі швидкістю шифрування (розшифрування) блоковими симетричними шифрами. Отже, з одному боку – достатньо висока швидкість симетричних шифрів, а з іншого усі переваги асиметричної криптографії. В роботах [10 – 12] показано, що практичне використання крипто-кодових засобів захисту інформації дозволяє на основі інтеграції механізмів канального кодування та шифрування комплексно забезпечити безпеку і достовірність переданих даних.

З аналізу літератури випливає, що відомі несиметричні крипто-кодові засоби захисту інформації будуються за двома схемами: з маскуванням породжуючої матриці коду (схема Мак-Еліса) і з маскуванням перевірочної матриці коду (схема Нідеррайтера) [7; 8]. Симетрична схема Рао-Нама по суті є деяким спрощенням несиметричною конструкції Мак-Еліса.

1.5 Постановка задач дослідження

Перспективнім напрямком в розвитку теорії і методів забезпечення інформаційної таємності є розробка та дослідження кодових схем захисту даних. Кодові схеми будуються по принципу маскування алгебраїчного блокового коду

під випадковий код, їх стійкість, відповідно, базується на складності декодування випадкового коду.

Об'єктом дослідження є процес забезпечення скритності та достовірності даних на основі використання крипто-кодовій системи Мак-Еліса.

Предметом дослідження є розроблення алгоритмів несиметричного шифрування схема Мак-Еліса на еліптичних кодах мобільних засобів зв’язку.

Метою дипломної роботи є автоматизація процесів вирішення комплексу задач програмного пакету “Забезпечення інформаційної скритності, та вірогідності даних, що обробляються в засобах мобільного зв’язку” з використанням сучасних інформаційних технологій.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

– виконати огляд проблеми забезпечення інформаційної скритності та вірогідності даних на основі крипто-кодових систем;

– розробити математичне та алгоритмічне забезпечення алгоритмів шифрування/розшифрування в несиметричної крипто-кодовій системі для використання в засобах мобільного звязку;

– провести дослідження складності алгоритмів, криптостійкості, запропанових механізмів, складу та змісту бізнес-процесів забезпечення конфіденційності та вірогідності даних;

– розробити опис програмного забезпечення для вирішення задач забезпечення інформаційної скритності і вірогідності;

– застосувати розроблене програмне забезпечення для вирішення поставленої задачі.

2 ОСНОВНІ ПРИНЦИПИ ПОБУДОВИ ТЕОРЕТИКО-КОДОВИХ СХЕМ

2.1 Загальна характеристика теоретико-кодових схем

Проведений аналіз теоретико-кодових схем показав доцільність їх застосування зважаючи на швидкість виконання криптопротоколів та доказову стійкість.

Теоретико-кодові схеми базуються на алгебраїчних блокових кодах і дозволяють поєднати завадостійке кодування й захист інформації. Насправді, декодування довільного лінійного коду є досить складним обчислювальним завданням, складність її рішення росте екпоненціально [13, 21, 22].

Так для кореляційного декодування довільного (n, k, d) коду над GF(q) необхідно, у загальному випадку, зрівняти прийняту послідовність із усіма qk кодовими словами й вибрати найближче (у метриці Хемінга). Навіть для невеликих n, k, d і q задача кореляційного декодування обчислювально складна і вимагає багато ресурсів. Це положення лежить в основі всіх криптосистем на теоретико-кодових схемах. Загальна класифікація відомих методів побудови теоретико-кодових схем наведена на рис. 2.1. У ряді робіт показано, що застосування теретико-кодових схем дозволяє будувати механізми захисту, які інтегровано, вирішують завдання безпеки й достовірності інформації [21, 22]. Реалізуються вони у вигляді одного пристрою або програмного продукту, складність реалізації якого порівнянна із БСШ.

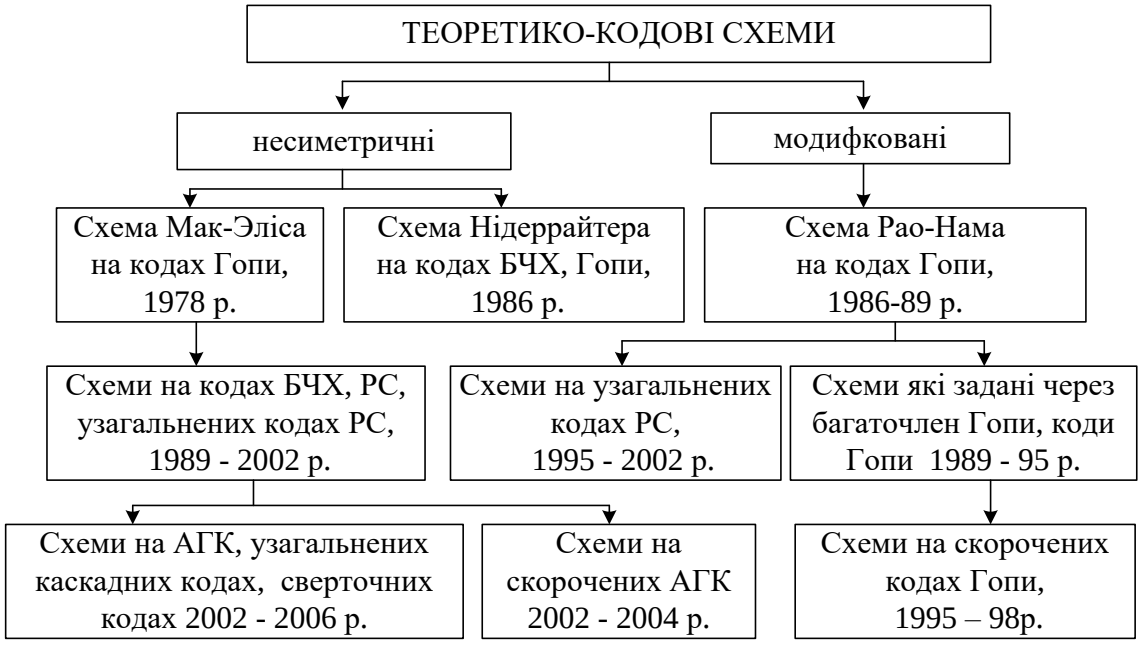


Рисунок. 2.1 – Загальна класифікація теоретико-кодових схем

Застосування теоретико-кодових схем є ефективним економічнм рішенням, оскільки дозволяє істотно знизити сумарні обчислювальні витрати на одиницю інформації, тобто підвищується своєчасність.

Проте існує низка робіт, зокрема авторів [21, 22], в яких запропоновано запропонований ефективний спосіб злому несиметричних схем Мак-Еліса і Нідеррайтера, побудованих на узагальнених кодах Ріда-Соломона. Хорошою перспективою розвитку потенційно стійких теоретико-кодових схем є використання алгеброгеометричних кодів для їх побудови. Автори [5, 6] вперше розглянули алгеброгеометричні коди як лінійні системи, побудовані по точках алгебраїчних кривих. Основними переваги методів алгеброгеометричного кодування є довгі недвійкові блокові коди, що володіють гарними асимптотичними властивостями [11, 26]. Перспективним напрямком у дозволі протиріч між різко зрослими обсягами оброблюваних даних, підвищенням ймовірносно-тимчасових вимог до безпеки й достовірності інформації, надійності її одержання й існуючих підходів теорії захисту інформації до побудови підсистем обміну конфіденційними повідомленнями є розробка протоколів, інтегровано забезпечуючих захист і достовірність переданої інформації, побудованих з використанням ТКС на основі алгебраїчних блокових кодів.

2.2 Система Рао-Нама

Симетрична схема Рао-Нама є спрощеною схемою, побудованою на основі несиметричної конструкції Мак-Еліса. На виході алгоритму можуть бути алгебраїчні блокові коди зі швидким (поліноміальної складності) алгоритмом декодування, такі, наприклад, як коди Гопи, Ріда-Соломона (РС), Боуза-Чоудхурі-Хоквігнема (БЧХ). Основна ідея схеми полягає в тому, що породжувальна матриця G лінійного блокового коду є єдиними секретним ключем.

Секретна інформація (кодограма) у теоретико-кодовій схемі Рао-Нама визначається у вигляді вектора довжини і обчислюється як

$$c^* = I \times G + e, \tag{2.1}$$

Формування кодограми здійснюється шляхом кодування інформаційної послідовності I довжиною k інформаційних символів у кодове слово довжиною n кодових символів і додаванні до нього випадкового вектора помилки e. Вага вектора помилок $w(e) \le t$, де t – кількість помилок, яке може виправити (n,k,d) блоковий код, $d = 2 \times t + 1$.

Для того, щоб декодувати отриману кодограму користувач декодує кодове слово з помилками (n,k,d) алгебраїчного блокового коду. Завдання декодування алгебраїчного блокового коду (наприклад, коду Гопи) має поліноміальну складність. Декодування довільного лінійного коду (коду загального положення) є неполіноміальним обчислювальним завданням. Складність його розв’язання зростає експоненціально [8, 21]. Це властивість покладена в основу теоретико-кодових схем Рао-Нама. Схему передачі секретного повідомлення від абонента А абонентові Б у симетричній схемі Рао-Нама представлено на рис. 2.2.

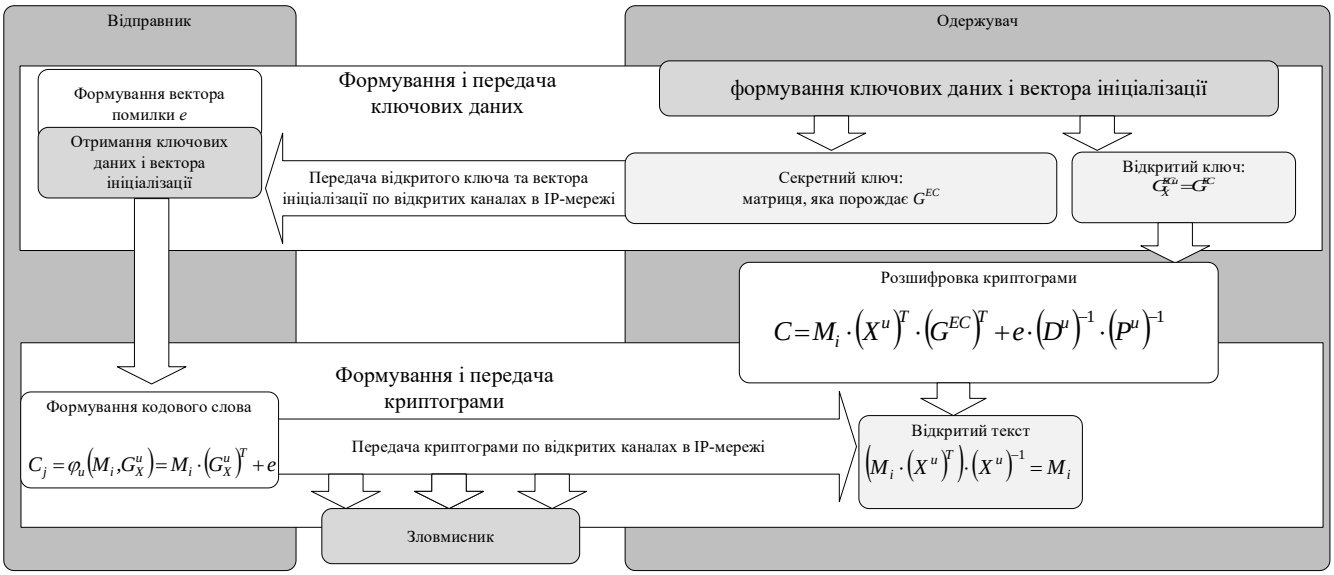


Рисунок 2. 2 – Схема передачі кодограми в симетричній ТКС Рао-Нама

2.3 Система Мак-Еліса

Розглянемо теоретико-кодову схему Мак-Елліса, уперше запропоновану в [11, 12, 29]. Позначимо G – породжувальну матриця лінійного (n, k, d)-коду над GF(q) з поліноміальною складністю декодування. Нехай X – невироджена k × k-матриця над GF(q), нехай D – діагональна матриця з ненульовими на діагоналі елементами, P – переставна матриця розміру $n \cdot n$. Переставна матриця фактично представляє перестановку координат вектора у вигляді матричного множення, а саме, елемент p_{ij} матриці P дорівнює 1 тоді й тільки тоді, коли координата з номером i переходить за допомогою перестановки в координату з номером j. В інших випадках $p_{ij} = 0$. Отже, кожний стовпець і кожний рядок матриці P містить тільки одну одиницю. Добуток матриць $\Lambda = P \times D$ задає переставну матрицю Λ з ненульовими елементами поля GF(q). Переставна матриця Λ (уніпотентна матриця) за умови перестановки координат вектора зберігає відстань по Хеммінгу, тобто $d(a, b) = d(a \times \Lambda, b \times \Lambda)$, де d(x, y) – відстань по Хеммінгу між векторами x і y.

Відкритим ключем у схемі Мак-Елліса є матриця $GX = X \times G \times P \times D$, секретним (закритим) ключем є матриці X, P, D. Закрита інформація (кодограма) являє собою вектор довжини n і обчислюється за правилом

$$c_X^* = i \cdot G_X + e, \tag{2.2}$$

де вектор $c_X = i \times G_X$ належить (n, k, d)-коду з породжувальною матрицею GX;

i – k-розрядний інформаційний вектор;

вектор e – секретний вектор помилок ваги (t).

Нападнику необхідно декодувати кодограму c_X^* з відомою породжувальною матрицею G_X. Якщо зловмисник не знає матриць X, P і D, то він не може відновити G за поліноміальний час. Не існує ефективних обчислювальних алгоритмів, щоб декодувати випадковий код великої довжини (експонентна складність при кореляційному декодуванні). Для авторизованого законного користувача декодування кодограми проводиться за ефективний поліноміальний час. Законний користувач, одержавши вектор c_X^*, будує вектор $\bar{c}^* = c_X^* \cdot D^{-1} \cdot P^{-1}$. Уніпотентна матриця $\Lambda = P \times D$ містить дані про вектор e (вагу по Хеммінгу). Практично, це означає, що вектор $\bar{c}^*$ є кодовим словом коду з матрицею, що породжує, G, перекручений не більш ніж в t розрядах. Далі законний користувач має можливість декодути вектор $\bar{c}^* = i' \cdot G + e'$, тобто зайти i' за поліноміальний час. Потім обчислює k-розрядний інформаційний вектор $i = i' X^{-1}$.

Таким чином, у теоретико-кодовій схемі Мак-Елліса основним засобом маскування лінійного (n, k, d)-коду з поліноміальною розв'язною задачею декодування є матриці X, P, D. Схему передачі секретного повідомлення від абонента А абонентові Б у несиметричній схемі Мак-Елліса представлено на рис. 2.3.

Передача кодограми випереджається наступними операціями. Абонент Б випадково, рівноймовірно, незалежно від інших абонентів формує матриці X, P, D і зберігає їх у секреті (закритий ключ). Обчислює матрицю $G_X = X \times G \cdot \times P \times D$ і публікує її як відкритий (загальнодоступний) ключ.

Абонент А для відправлення секретного повідомлення i формує кодограму $c_X^* = i \cdot G_X + e$. Її може сформувати будь-який користувач, що знає публічний (загальнодоступний) ключ.

Абонент А для відправлення секретного повідомлення i формує кодограму $c_X^* = i \cdot G_X + e$. Її може сформувати будь-який користувач, що знає публічний (загальнодоступний) ключ.

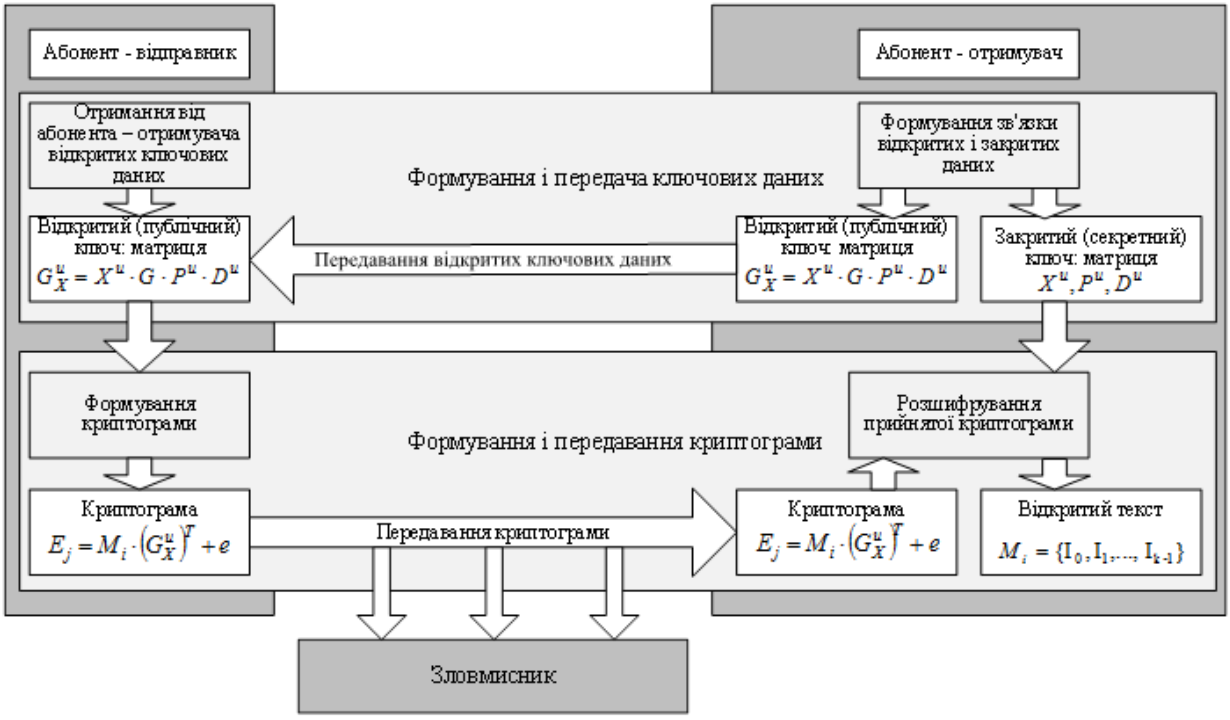


Рисунок 2. 3 – Схема передачі кодограми в несиметричній ТКС Мак-Елліса в режимі прямого виправлення помилок

Противник, не знаючи секретного ключа абонента Б, не зможе розкрити вміст кодограми (прочитати інформаційне повідомлення), для нього декодування – важкорозв'язна задача (експонентної складності). Навпроти, абонент Б декодує кодограму з алгоритмами поліноміальної складності.

Відомі приклади несиметричних теоретико-кодових схем Мак-Елліса розглянуті на випадок кодів БЧХ і та їх підкласу (кодів Ріда-Соломона). Автори [22, 23] запропоновали достатньо ефективний метод злому схеми Мак-Елліса на основі кодів Ріда-Соломона.

2.4 Система Нідеррайтера

Розгляньмо на прикладі несиметричну кодову криптосистему Нідерайтера: нехай H – перевірочна матриця лінійного (n, k, d) коду над GF(q) з поліноміальною складністю декодування. Нехай X – невироджена r × r матриця над GF(q), D – діагональна матриця з ненулевими елементами на діагоналі, P – перестановочна матриця розміру n × n. Відкритим ключем в криптосистемі Нідерайтера є матриця HX = X × H × P × D, секретним (закритим) ключем є матриці X, P, D. Закрита інформація (кодограмм) SX являє собою вектор величини r = n – k і обчислюється за правилом

$$S_x = e \cdot H_X^T, \tag{2.3}$$

де e – вектор довжиною n і вагою ≤ t, який містить секретну інформацію (інформаційне повідомлення, яке підлягає закриттю).

Авторизований користувач знаходить одне з можливих рішень рівняння $S_X = c_X \cdot (H_X^{EC})^T$. Знайдений розв'язок рівняння – це кодове слово з помилками. Далі користувач створює вектор $\bar{c}$, проводить декодування отриманого слова і обчислення кодового слова c та вектору помилок e'. На завершення виробляється обчислення вектора e = e ' × P × D, який містить секретну інформацію. Ілюстрацію передачі конфіденційного повідомлення від абонента А к абонентові Б у несиметричній схемі Нідерайтера представлено рис. 2.4.

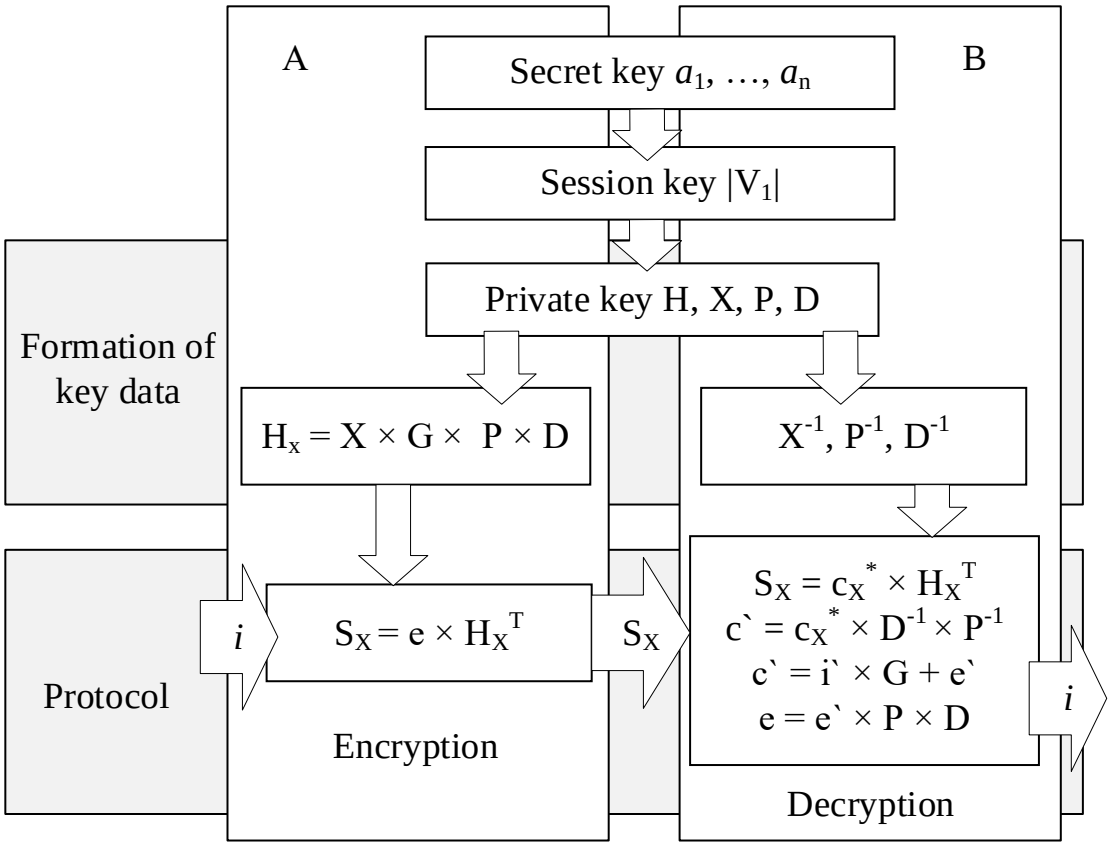


Рисунок 2. 4 – Схема передачі кодограми в несиметричній ТКС Нідерайтера в режимі автоперезапиту

Таким чином, в кодової криптосистемі Нідеррайтера основним засобом маскування лінійного (n, k, d) коду є матриці X, P, D. У роботах [2] показано, що перспективним напрямком вважається використання криптосистем на алгеброгеометричних кодах (АГК) [2, 5].

Передача криптограми в даній криптосистемі передує слідуючим операціям. Абонент Б випадково, равноймовірно, незалежно від інших абонентів формує матриці X, P, D і зберігає їх в секреті (таємний ключ). Обчислює матрицю HX і публікує її як відкритий (загальнодоступний, відкритий) ключ.

Таким чином, у формуванні повідомлення M і беруть участь алгоритми рівноважного кодування, які, в свою чергу є алгоритмами надлишкового (завадостійкого) кодування.

3 ПРОЕКТУВАННЯ ПРОГРАМНОГО ПАКЕТУ “ЗАБЕЗПЕЧЕННЯ КОНФІДЕНЦІЙНОСТІ ТА ВІРОГІДНОСТІ ДАНИХ, ЩО ОБРОБЛЯЮТЬСЯ В ЗАСОБАХ МОБІЛЬНОГО ЗВ’ЯЗКУ”

3.1 Опис предметної області

Першим етапом аналізу та моделювання бізнес-процесів системи є створення контекстної діаграми модуля “Забезпечення конфіденційності та вірогідності даних”. На вході використовуються наступні документи: вектор помилок, Інформаційна посилка та матриці маскування. Вихідними документами є кодове слово, Інформаційна посилка, відкритий ключ. В якості управління – користувач системи, в якости регулювання – Закон України “Про захист інформації в інформаційно-телекомунікаційних системах”, Закон України “Про захист інформацію”.

Декомпозиція складається з наступних робіт:

- формування відкритого ключа;
- формування кодової послідовності;
- розкодування кодової послідовності.

Контекстна діаграма “Забезпечення конфіденційності та вірогідності даних” наведена на рис. 3.1.

Декомпозиція контекстної діаграми наведена на рис. 3.2.

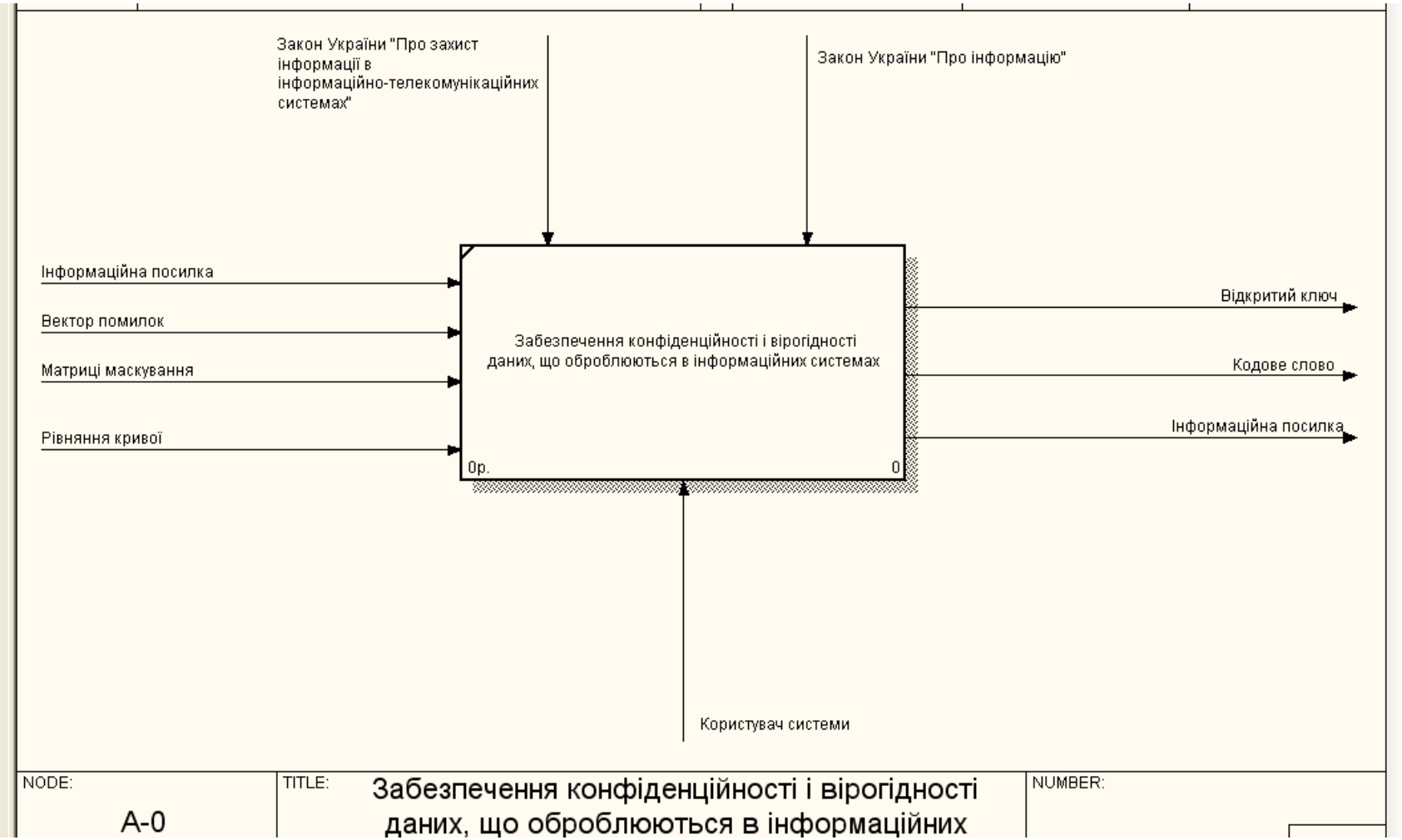


Рисунок 3.1 – Контекстна діаграма “Забезпечення конфіденційності та вірогідності даних” в IDEF0

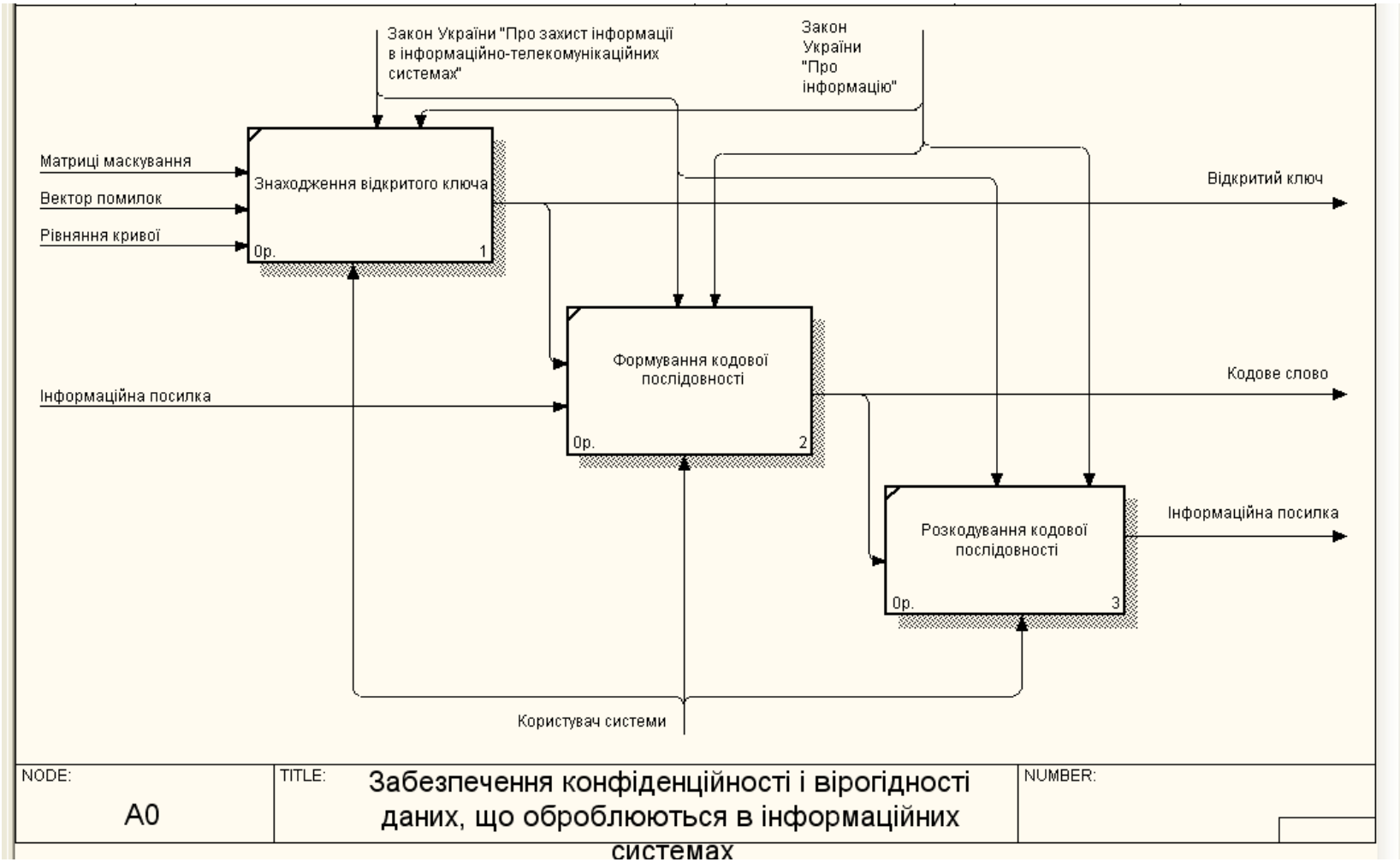


Рисунок 3.2 – . Декомпозиція контекстної діаграми “Забезпечення конфіденційності та вірогідності даних” в IDEF0

Аналіз протоколів і механізмів захисту показав, що для забезпечення автентифікації, цілісності і конфіденційності передачі даних у внутріплатіжних системах використаються криптографічні методи, засновані на використанні симетричних і несиметричних алгоритмів перетворення інформації. Разом з тим, подальше посилення погроз, указує на необхідність інтегрованого підходу для забезпечення захисту переданої інформації в засобах мобільного зв'язку.

Застосування теоретичних кодових схем для каналів з автоперезапитом дозволяє забезпечити несиметричну обробку інформації з ефективним захистом переданих даних від випадкового або навмисного впливу та забезпечити необхідні показники вірогідності і конфіденційності даних при передачі відкритими каналами.

Проведемо декомпозицію блоків отриманої діаграми. Подальша декомпозиція роботи "Формування відкритого ключа" також здійснюється за допомогою методології IDEF0 та складається з таких робіт:

- формування рівняння еліптичної кривої;
- знаходження породжувальної матриці;
- знаходження відкритого ключа;

Декомпозиція роботи "Формування відкритого ключа" наведена на рис. 3.3.

Аналогічно проводимо декомпозицію роботи "Розкодування кодової послідовності" за допомогою методології IDEF0, яка складається з таких робіт:

- знаходження зворотних матриць;
- перемноження кодового слова з зворотними матрицями;
- процедура Ченя; локалізація місця помилок; знаходження інформаційної посилки.

Декомпозиція роботи "Розкодування кодової послідовності" наведена на рис. 3.4.

Аналогічно проводимо декомпозицію роботи "Формування кодової послідовності" за допомогою методології IDEF0, яка складається з таких робіт:

- перемноження закритого ключа та інформаційної посилки;
- додавання до матриці C_x вектора помилок; передача кодового слова каналом зв'язку. Декомпозиція роботи "Формування кодової послідовності" наведена на рис. 3.5.

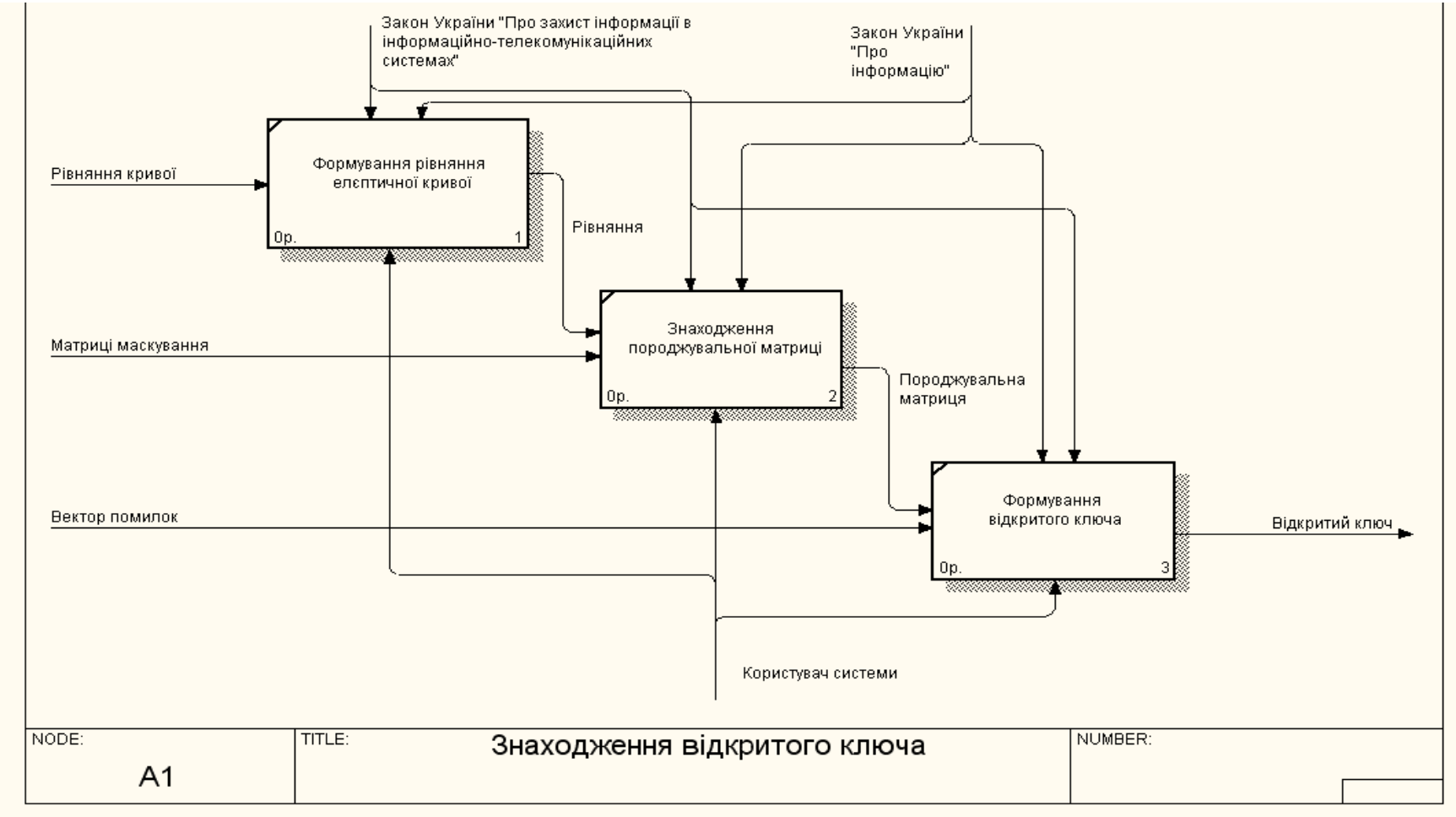


Рисунок 3.3 – Декомпозиція роботи “Знаходження відкритого ключа” в IDEF0

15 47

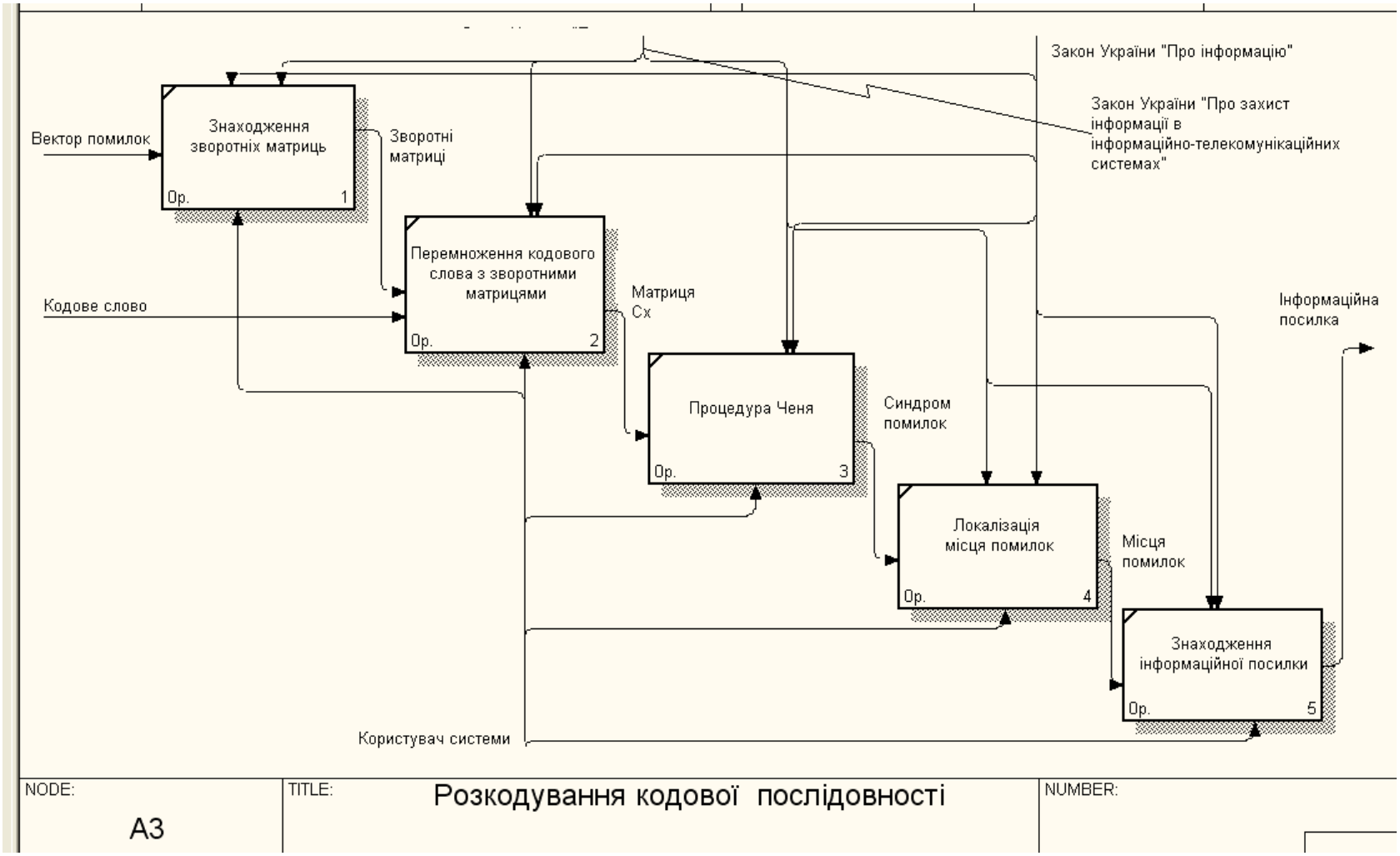


Рисунок 3.4 – Декомпозиція роботи “Розкодування кодової послідовності” в IDEF0

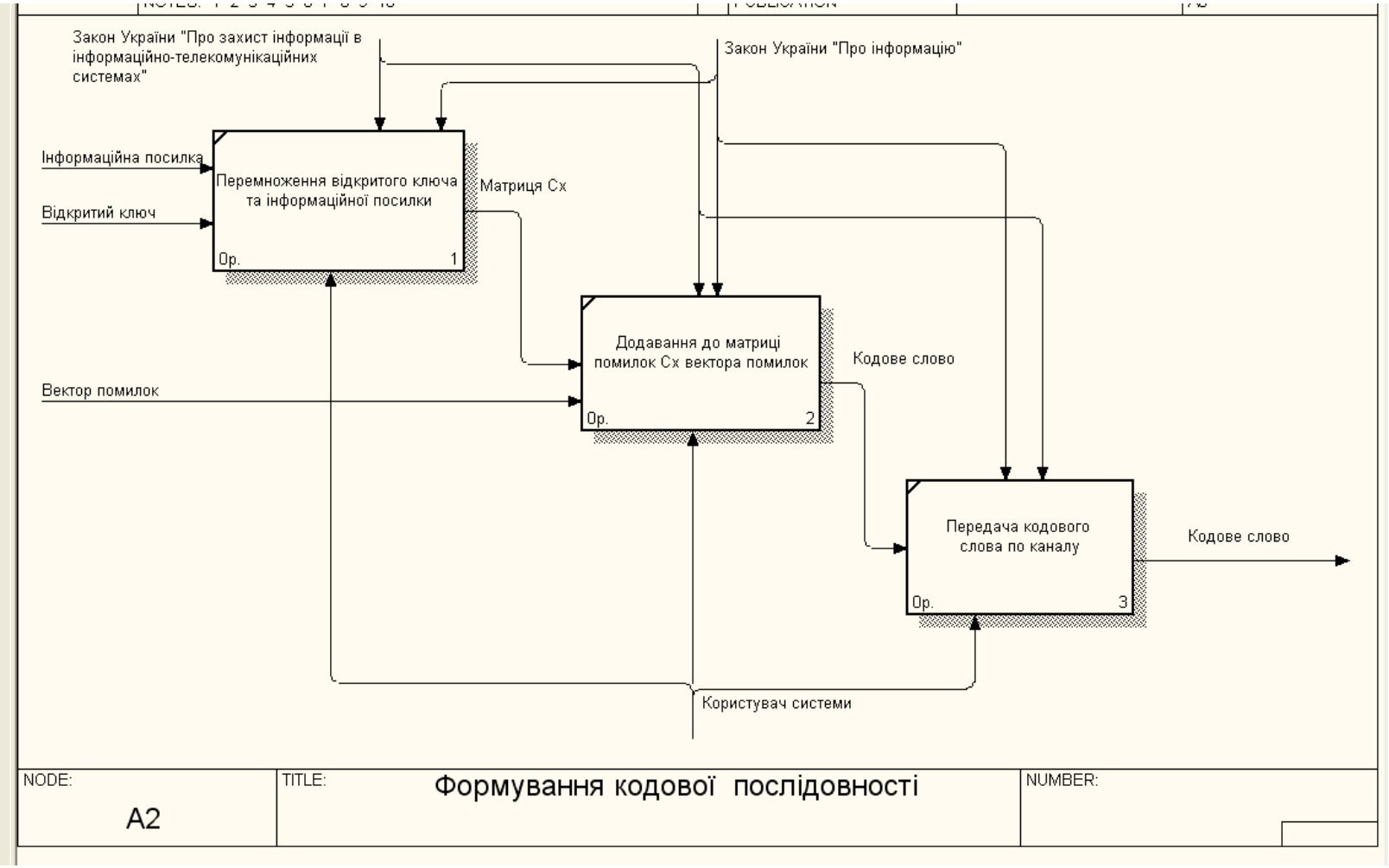


Рисунок 3. 5 – Декомпозиція роботи “Формування кодової послідовності” в IDEF0

3.2 Розроблення специфікацій бізнес-вимог до модуля “Забезпечення конфіденційності та вірогідності даних”

1) Вимоги з точки зору бізнесу

Програмне забезпечення містить в собі товарну цінність тільки у тому випадку, якщо задовольняє вимогам попиту або замовника. Бізнес аналіз та виконання його вимог є запорукою успішного попиту на програмне забезпечення та вкрай важливими для постачальників. Основні бізнес-вимоги до модуля “Забезпечення конфіденційності та вірогідності даних, що обробляються в інформаційних системах та базах даних” можна охарактеризувати наступним чином:

- забезпечення конфіденційності даних;
- забезпечення вірогідності даних;
- забезпечення автентифікації кінцевого користувача.

Модуль “Забезпечення конфіденційності та вірогідності даних” дозволяє автоматизувати функції формування та розкодування кодової послідовності, а саме формування та розкодування завадостійких кодів на еліптичних кривих та забезпечення конфіденційності та вірогідності даних.

З точки зору сучасного бізнесу розроблювана система повинна атоматизовувати забезпечення конфіденційності та вірогідності даних що обробляються у інформаційних системах та базах даних. Використовуючи розроблений автоматизований інформаційний модуль, користувач системи має можливість найбільш ефективно захистити конфіденційні данні від можливості розкодування або внесення у них якої-небудь інформації зловмисником тим самим забезпечивши цілісність та конфіденційність цих данних.

Вирішення задач модуля автоматизованим способом дає можливість користувачу виконувати автоматизовано наступні дії: автоматизоване формування закритого ключа, автоматизоване формування відкритого ключа, автоматизоване формування кодової послідовності, автоматизоване розкодування кодової послідовності, автоматизоване генерування необхідних математичних одиниць,

автоматизований розрахунок зворотніх матриць та знаходження помилок внесених користувачем. Взагалі модуль має вигляд набір компонентів та функцій для рішення поставленої задачі, але до захисту дипломного проекту був створений зручний та наглядний інтерфейс для імітації роботи модуля поза системою з демонстрацією всіх ітерацій роботи модуля. Користувач повинен відчувати швидкість досягнення результату при використанні системи та захист інформації від несанкціонованого доступу.

Модуль має найпростіший інтерфейс для демонстрації функцій формування та розкодування кодової послідовності, а саме формування та розкодування завадостійких кодів на еліптичних кривих

2) Вимоги користувача системи

Інтерфейси користувача системи наведено в табл. 3.1.

Таблиця 3.1 – Інтерфейси користувача

Ідентифікатор	Інтерфейси користувача
IU-01	Робота з БД ведеться шляхом додавання, редагування, видалення, збереження записів таблиць БД.
IU-02	Діалогові вікна програми призначені для здійснення поетапного вирішення процесу персоніфікованого обліку співробітників на підприємстві

Інтерфейси програмного забезпечення наведені в табл. 3.2.

Таблиця 3.2 – Інтерфейси програмного забезпечення

Ідентифікатор	Інтерфейси програмного забезпечення
IS-01	Підтримка мови програмування C#.
IS-02	Підтримка технології SqlServer.
IS-03	Платформа .Net Framework 4.0

Вимоги до продуктивності наведені в табл. 3.3.

Атрибути якості наведені в табл. 3.4.

Таблиця 3.3 – Вимоги до продуктивності

Ідентифікатор	Вимога до продуктивності
PR-01	Час запуску системи – 2 с.
PR-02	Час формування матриць – не більше 5 с.
PR-03	Час формування кодової послідовності – не більше 2 с.
PR-04	Час розкодування кодової послідовності – не більше 1 с.

Таблиця 3.4 – Атрибути якості

Ідентифікатор	Атрибут якості
QA-01	Зручний і функціональний інтерфейс, що не вимагає великої кількості часу на освоєння й роботу
QA-02	Лекгість обслуговування системи (повинен бути передбаченний відповідний функціонал для обслуговування й системи)

На рис. 3.6. наведена зображена діаграма бізнес-варіантів для модулю забезпечення конфіденційності та вірогідності даних.

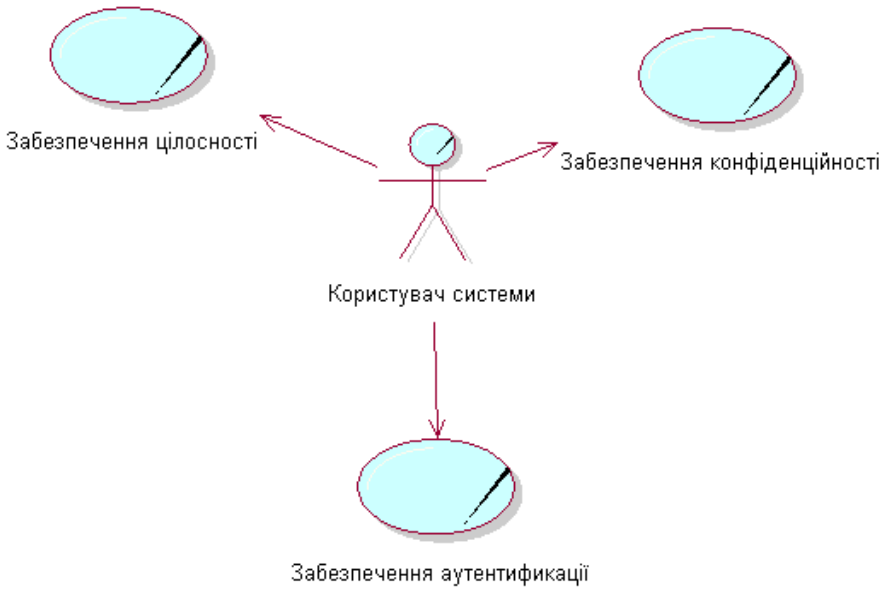


Рисунок 3. 6 – Діаграма бізнес-варіантів використання модулю “Забезпечення конфіденційності та вірогідності даних”

3.2.1 Опис функціональних вимог

Програмний продукт “Забезпечення конфіденційності та вірогідності даних” складається з трьох бізнес-процесів: “Формування відкритого та закритого ключів”, “Формування кодової послідовності” та “Розкодування кодової послідовності”. В якості діючої особи виступає користувач системи

У рамках об’єктного підходу опис вимог до розроблюваної ІС здійснюється на підставі стану об’єктів предметної області, що реалізується в таких моделях, як діаграми варіантів використання та діаграми послідовності взаємодії.

Отже побудуємо діаграму варіантів використання, що відображає взаємодію між варіантами використання та дійовими особами. Тобто між функціями, які виконує система, та зацікавленими особами стосовно створюваної системи.

Діаграми використання є широко вживаними в різних технологіях проектування. Їх головне завдання – специфікація вимог до системи на початкових етапах проектування, коли вирішуються найбільш загальні завдання призначення, що розробляється.

Діаграми використання є досить популярними і включаються в різні методики проектування завдяки своїм наступним достоїнствам: формулювання вимог, орієнтована на користувача, тобто система свідомо проектується так як її буде бачити користувач, простота моделі, її орієнтованість на непідготовленого користувача, можливість опису великих проектів за рахунок декомпозиції на великі блоки використання, яких у будь-якій системі не так багато, як структурних одиниць самої системи.

Діаграма варіантів використання наведена зображена на рис. 3.7.

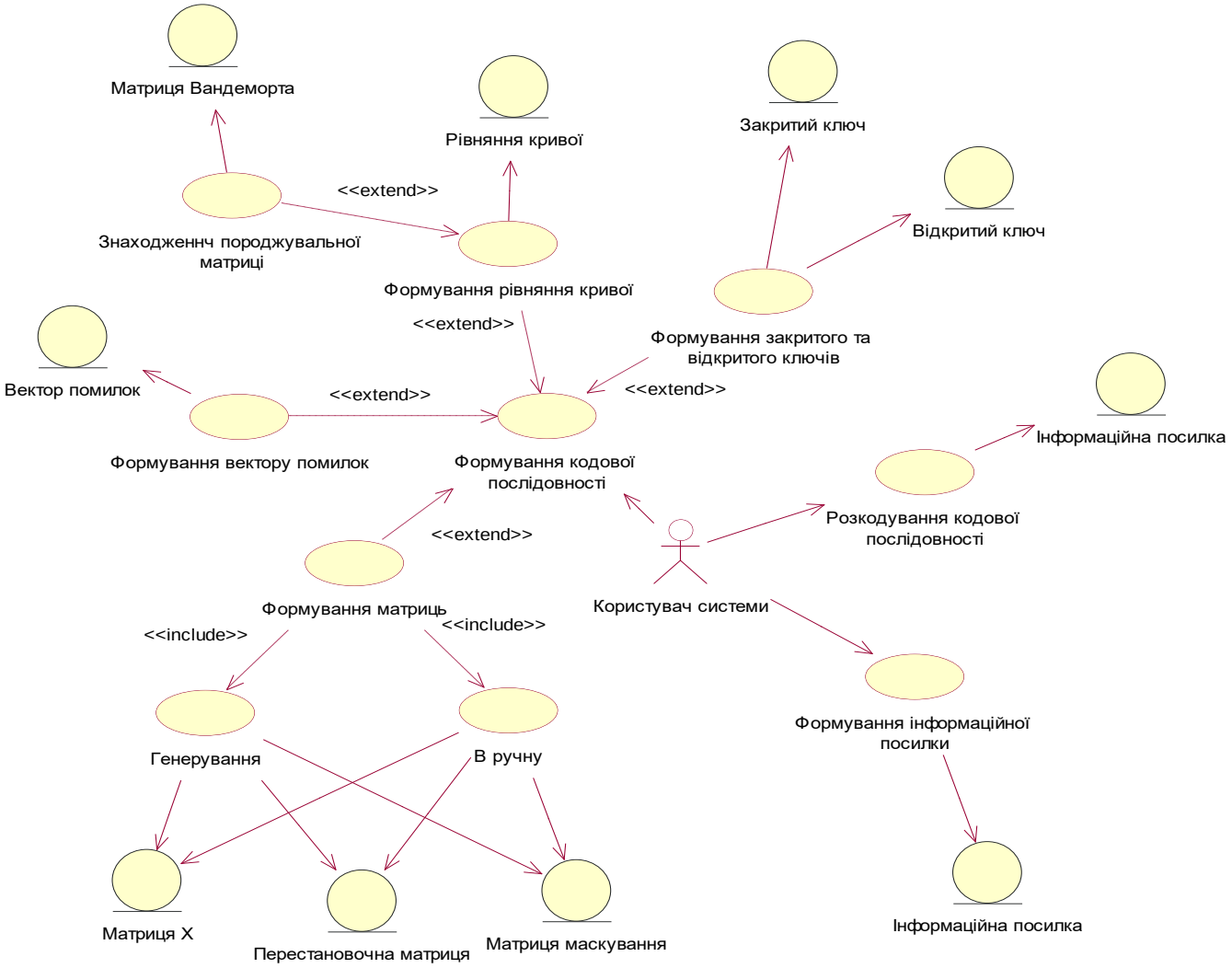


Рисунок 3.7 – Діаграма варіантів використання для модулю “Забезпечення конфіденційності та вірогідності даних”

У мові UML існує діаграма взаємодії, яка має назву діаграма послідовності. До речі, діаграми послідовності впорядковані за часом та відображають кожен потік дій, що відбувається в рамках варіанта використання. Варіант використання “Формування матриць вручну” передбачає послідовний ввід користувачем цифрових даних до таблиць матриць та запис їх до бази даних. На рис. 3.8 наведена діаграма послідовності основного потоку дій варіанту використання “Формування матриць вручну”.

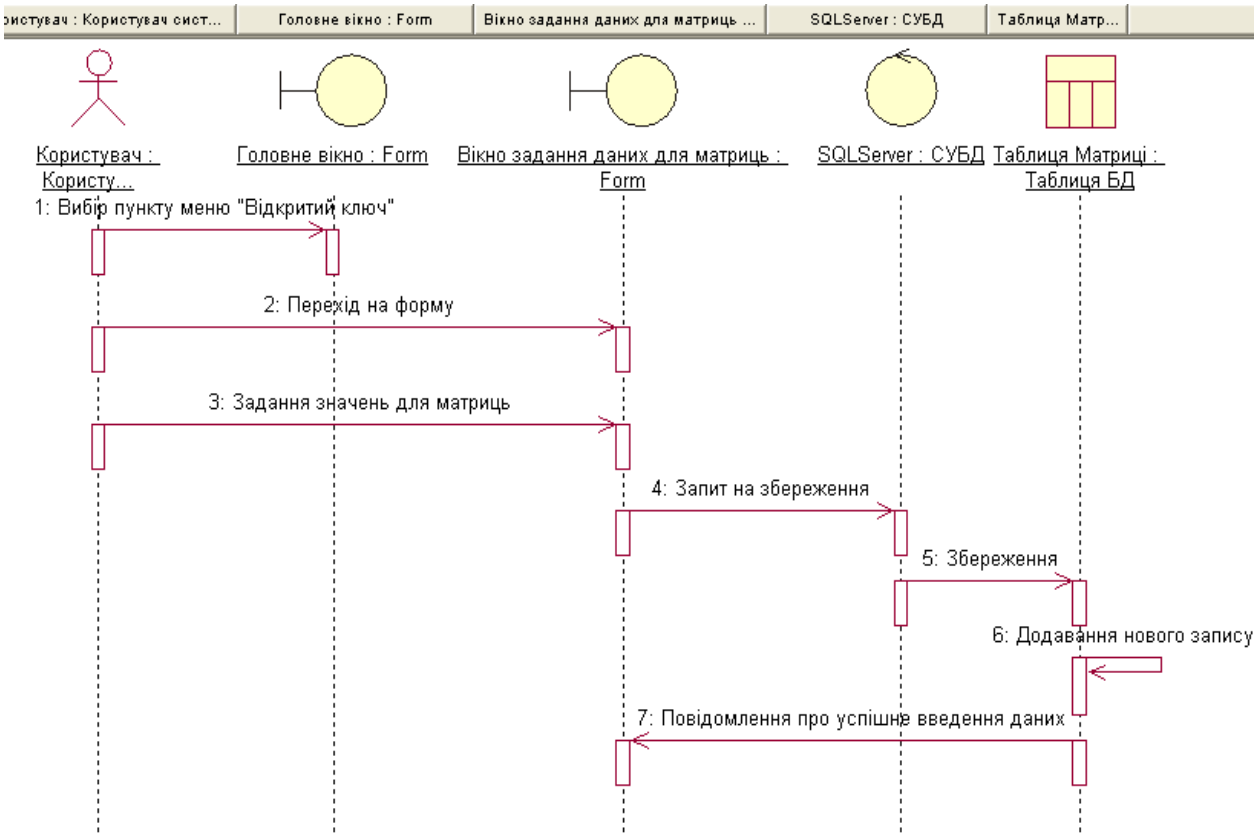
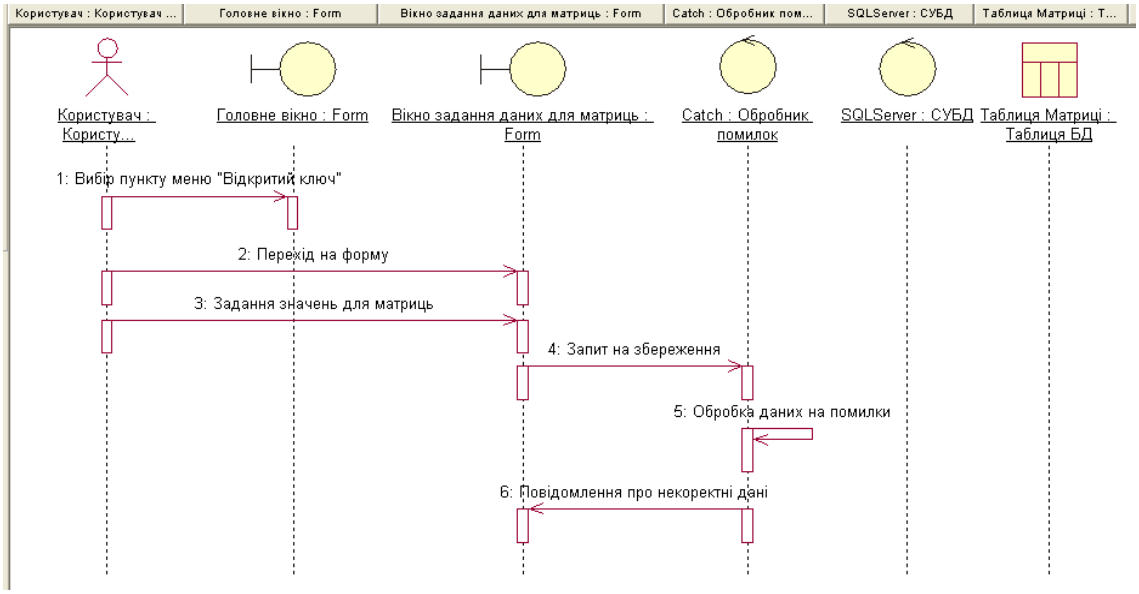


Рисунок 3.8 – Діаграма послідовності основного потоку варіанту використання “Формування матриць вручну”

Далі наведена діаграма послідовності альтернативного потоку варіанту використання “Формування матриць вручну” (рис. 3.9).

Стислий опис: дозволяє користувачеві ввести матриці маскування вручну або згенерувати їх автоматично, на основі котрих потім буде формуватися закритий ключ.

Основний потік подій: даний варіант використання починає виконуватися, коли користувач системи вибирає пункт меню “Ввести данні”



Риунок 3. 9 – Діаграма послідовності альтернативного потоку варіанту використання “Формування матриць вручну”

Варіант використання “Формування вектору помилок” передбачає послідовний ввід користувачем цифрових даних до таблиць матриць та запис їх до бази даних. На рис. 3.10 наведена діаграма послідовності основного потоку дій варіанту використання “Формування вектору помилок”.

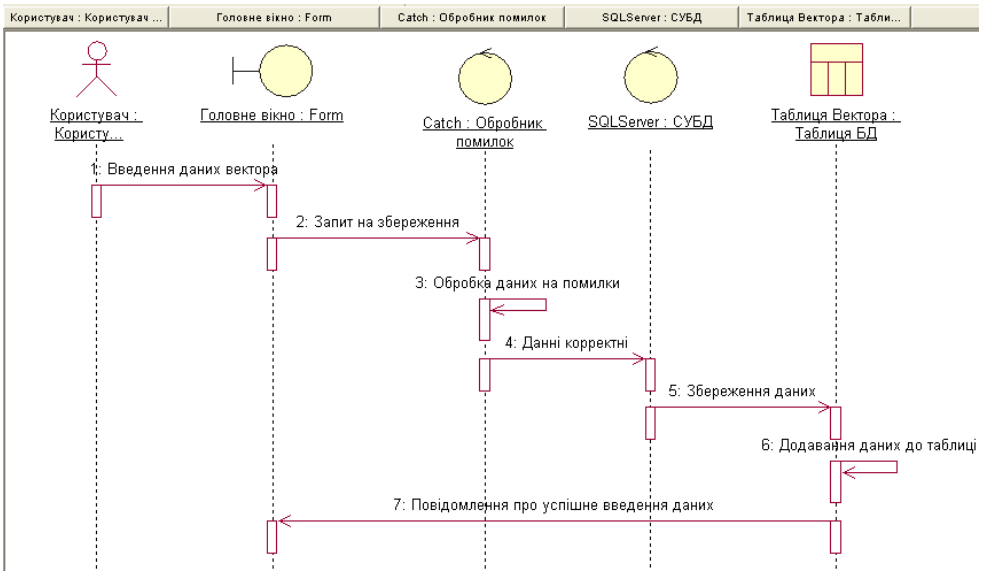


Рисунок 3. 10 – Діаграма послідовності основного потоку варіанту використання “Формування вектору помилок”

Також на рис. 3.11 наведена діаграма послідовності альтернативного потоку формування вектору помилок

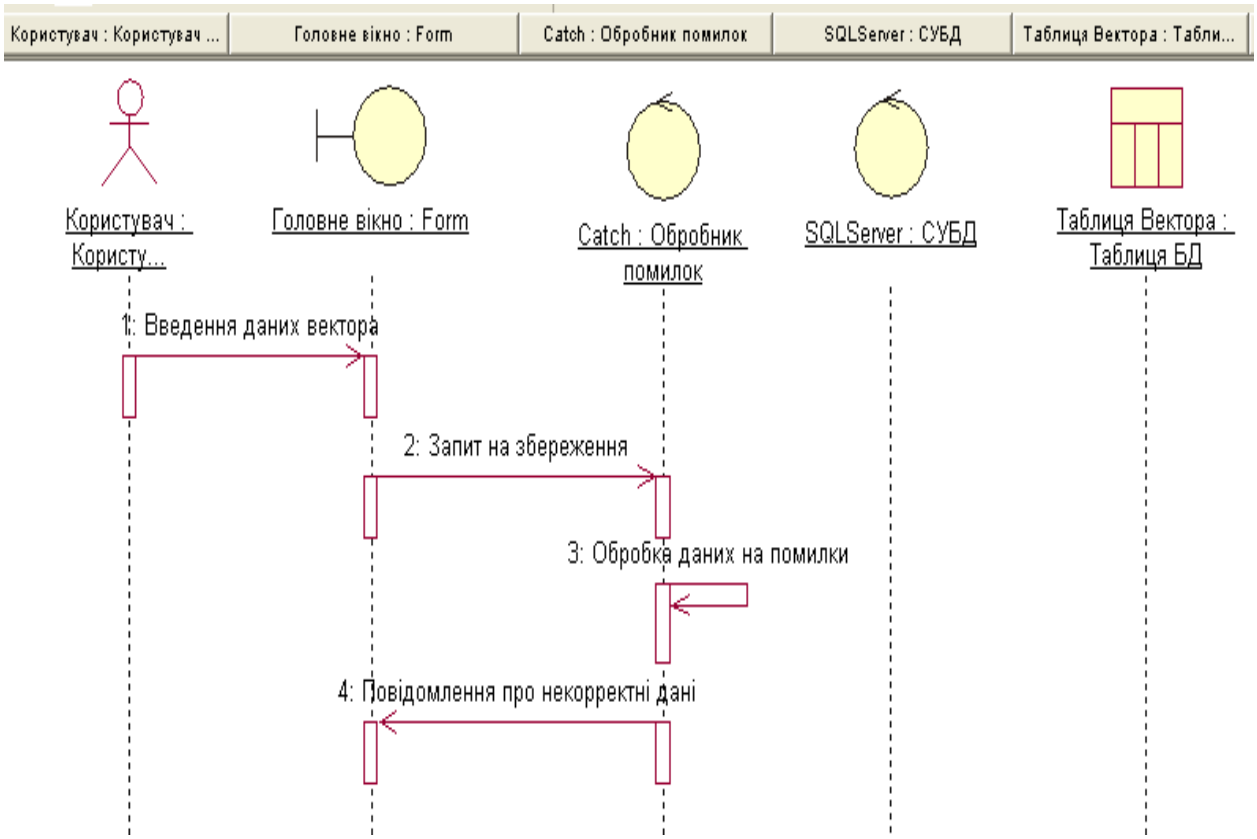


Рисунок 3. 11 – Діаграма послідовності альтернативного потоку варіанту використання “Формування вектору помилок”

Варіант використання «Розкодування кодової послідовності» передбачає послідовність повідомлень, що дозволяють видаляти дані з таблиці бази даних. На рис. 3.12 зображена діаграма послідовності основного потоку дій варіанту використання “Розкодування кодової послідовності”.

Стислий опис: дозволяє користувачеві ввести задати вектор помилок, котрий потім буде використовуватись в кодуванні інформацій посилки

Основний потік подій: даний варіант використання починає виконуватися, коли користувач системи вводить матриці маскування та вибирає рівняння кривої.

На рис. 3.13 наведена діаграма послідовності альтернативного потоку варіанту використання “Розкодування кодової послідовності”.

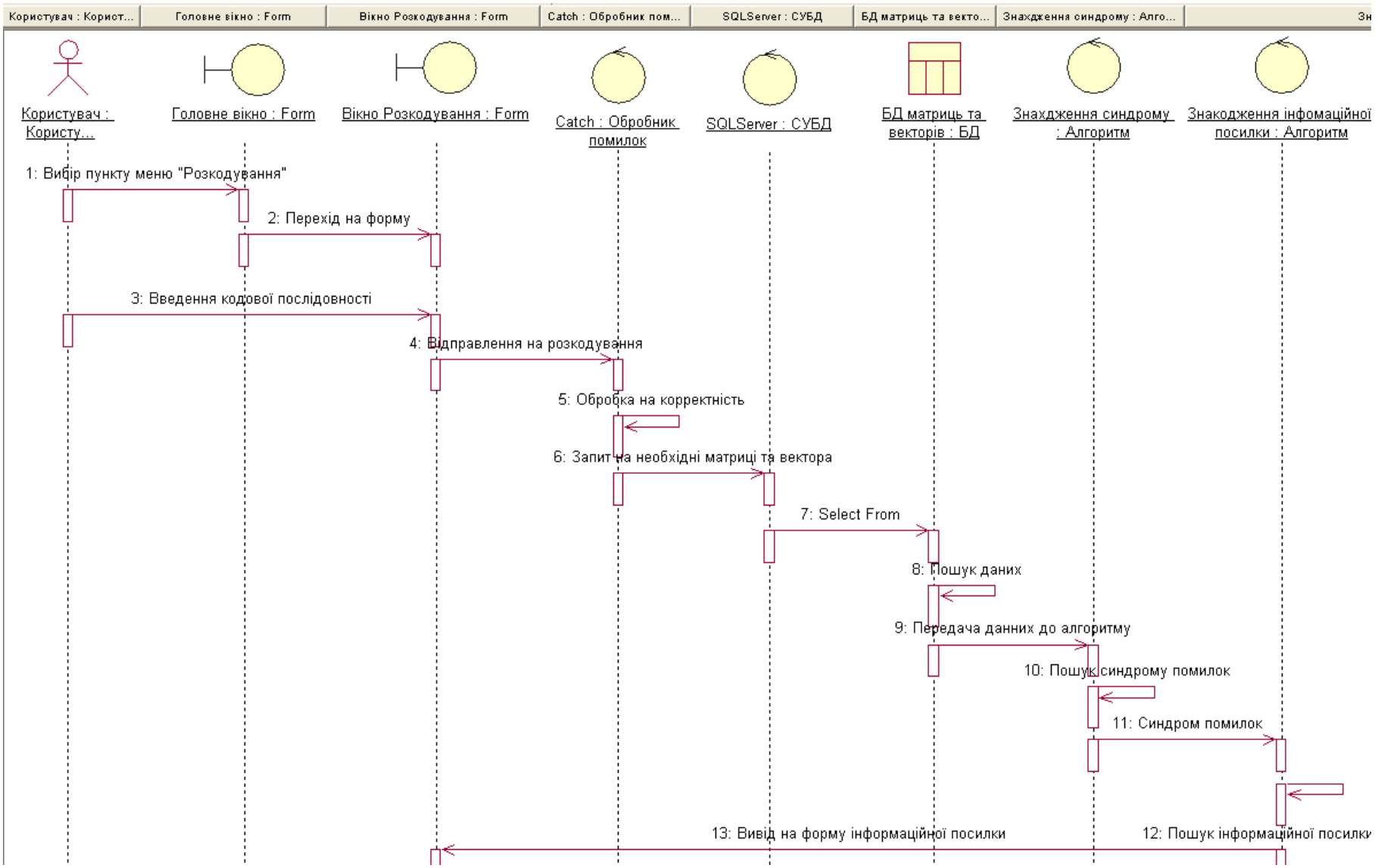


Рисунок 3. 12 – Діаграма послідовності основного потоку варіанту використання “Розкодування кодової послідовності”

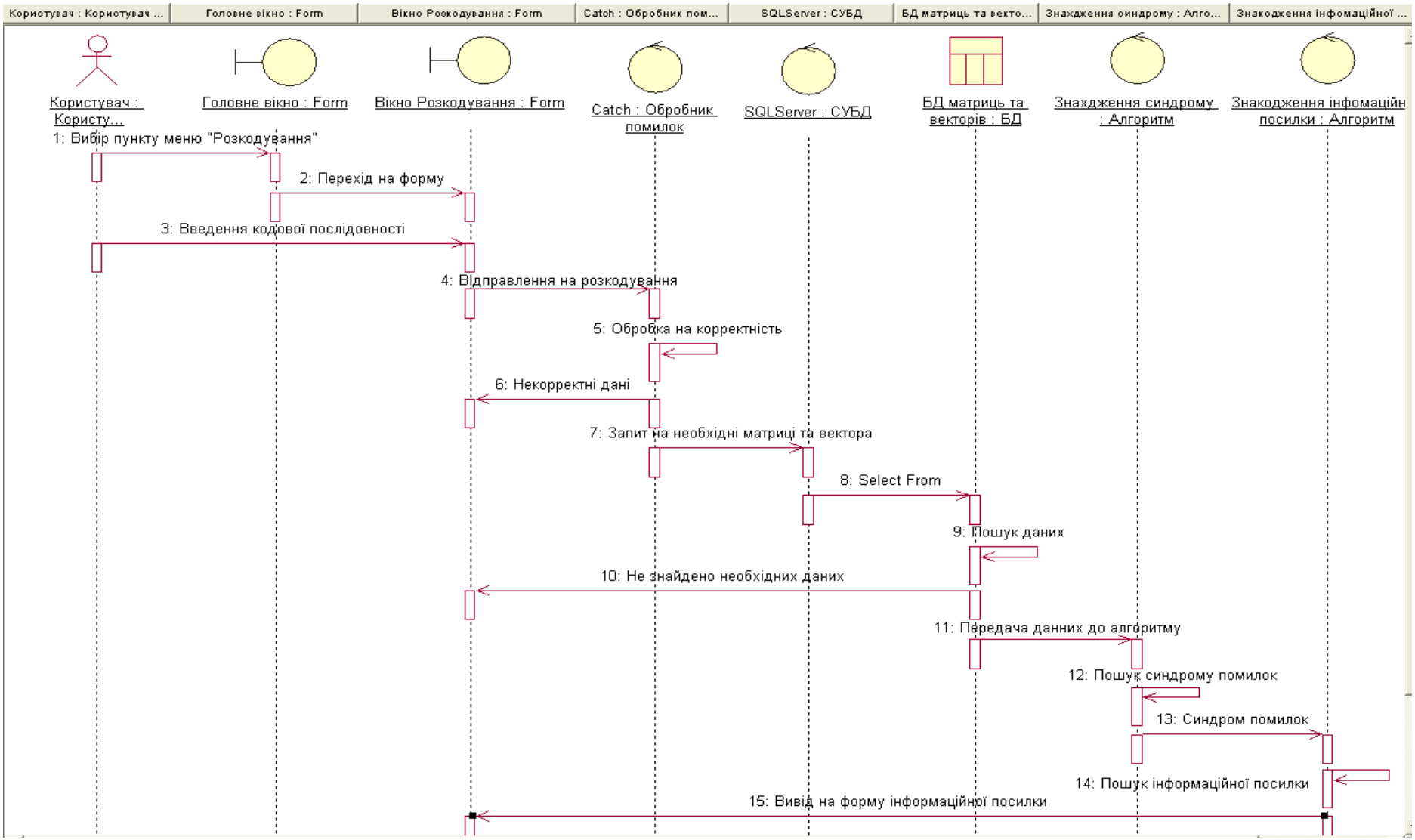


Рисунок 3. 13 – Діаграма послідовності альтернативного потоку варіанту використання “Розкодування кодової послідовності”

3.2.2 Опис постановки комплексу задач модуля “Забезпечення конфіденційності та вірогідності даних”

3.2.2.1 Характеристика комплексу задач програмного продукту “Забезпечення конфіденційності та вірогідності даних”

Призначення комплексу задач модуля

Функціональний модуль “Забезпечення конфіденційності та вірогідності даних” входить до підсистеми “Забезпечення конфіденційності та вірогідності даних в інформаційних системах та базах даних”.

Код підсистеми: 01.

До складу модуля входять наступні задачі:

- 0101 – задача “Формування відкритого ключа”;
- 0102 – задача “Формування кодової послідовності”;
- 0103 – задача “Розкодування кодової послідовності”.

Мета вирішення комплексу задач модуля – автоматизація бізнес-процесів забезпечення конфіденційності та вірогідності даних

Призначення комплексу задач модуля – формування і розрахунок наступних математичних показників:

- рівняння кривої;
- кінцеве поле;
- вектор помилок;
- синдром помилок;
- корні рівнянь;

Це завдання порівняно з ручною технологією її рішення забезпечує виконання автоматизованим способом наступних операцій:

1) автоматизоване створення відкритого та закритого ключів

2) автоматизоване формування матриць перестановки, маскування та породжувальної матриці.

3) автоматизоване формування кодового слова;

4) автоматизоване кодування кодової послідовності;

5) автоматизоване знаходження зворотних матриць;

6) автоматизоване формування кінцевого поля;

7) автоматизоване знаходження синдрому помилок;

8) автоматизоване розкодування кодової послідовності;

9) автоматизоване знаходження інформаційної посилки;

10) автоматизоване розв’язання рівнянь еліптичної кривої.

Обмін інформацією між робочими місцями здійснюється за допомогою ліній зв’язку.

Програмно-технічна платформа рішення завдання відповідає сучасному розвитку ІТ.

Доцільність автоматизованого рішення комплексу задач модуля обґрунтовується:

– відсутністю на ринку програмних продуктів по забезпеченню конфіденційності та вірогідності даних, продуктів які б задовольняли вимогам на 100%;

– необхідністю використання при кодуванні конфіденційної інформації завадостійких алгоритмів.

Перелік об’єктів, при управлінні якими розв’язується комплекс задач модуля

В результаті рішення комплексу задач модуля автоматизуються бізнес-процеси автоматизація бізнес-процесів забезпечення конфіденційності та вірогідності даних, які вирішують користувачі системи.

Періодичність і тривалість рішення комплексу задач модуля

автоматизація бізнес-процесів забезпечення конфіденційності та вірогідності даних виконується в залежності від потреб користувачів. Тривалість рішення задачі не перевищує 2 хв.

Умови, при яких припиняється вирішення комплексу задач модуля автоматизованим способом.

Виконання комплексу задач модуля автоматизованим способом неможливо при руйнуванні БД, виході з ладу ПК чи відсутності деяких даний у вхідних документах, що використовуються при розрахунку показників.

Зв’язки комплексу задач модуля з іншими задачами АІС.

Зв’язок комплексу задач модуля наведен на рис. 3.14.

Рисунок 3. 14 – Зв’язок комплексу задач модуля з іншими задачами

1 – матриці X, P, D, вектор помилок;

2 – масиви НДІ;

3 – кодова послідовність;

4 – закритий та відкритий ключ.

6) Посади осіб і назви підрозділів, які визначають умови комплексу задач модуля.

Терміни рішення комплексу задач модуля визначає користувач системи

7) Відповідальність персоналу і технічних засобів при різних ситуаціях вирішення комплексу задач модуля.

З комплексом задач модуля можна ознайомитись у діалоговому режимі під управлінням користувачів. Користувачі відповідають за підтримку БД в актуальному стані, за своєчасне введення первинної інформації за призначенням і

виконання розрахунків, передачу вихідної інформації за призначенням і використання результатів розрахунків з ціллю прийняття управлінського рішення.

Усі розрахунки виконуються цілком автоматизованим способом на ПК.

3.2.2.2 Вихідна інформація

Вихідною інформацією є кодова послідовність, інформаційна посилка,закритий ключ.

Кодова послідовність повинна містити закодовану інформаційну посилку.

Інформаційна посилка повинна містити розкодовану кодову послідовність.

Закритий ключ містить інформацію за допомогою якої відбувається аутентифікації кінцевого користувача

Перелік та опис вихідних повідомлень подано у табл. 3.5

Таблиця 3.5 – Перелік та опис вихідних повідомлень

Іденти-фікатор (код повідом-лення)	Найменування вихідного повідомлення	Форма представ лення (МГ, ВК, масив)	Періодич-ність видачі	Термін видачі і припустимий час затримки вирішення	Одержувачі	Призначення вихідної інформації
0101301	Кодова послідовність	ВК	За запитом	Визначається часом надходження запиту	Кінцевий користувач	Призначена для формування розкодованої інформаційної посилки
0101302	Інформаційна посилка	ВК	За запитом	Визначається часом надходження запиту	Кінцевий користувач	Призначена для формування кодової послідовності
0101303	Відкритий ключ	масив	За запитом	Визначається часом надходження запиту	Задачі 0102 та 0103	Забезпечення автентифікації користувачів

Опис структурних одиниць та загальний перелік вихідних машинограм і відеокадрів наведений у табл. 3.6.

Таблиця 3. 6 –Опис структурних одиниць та перелік вихідних повідомлень

Найменування структурної одиниці інформації	Ідентифікатор	Вимоги до точності
ВК "Кодова послідовність"		
Закодована інформація	CX	N(20)
Розмір	NAIM_SKLAD	N(5)
ВК "Інформаційна посилка"		
Текстова інформація	I	X (200)
Розмір	I_SIZE	N(5)

3.2.2.3 Вхідна інформація

Перелік і опис вхідних документів наведені у табл. 3.7.

Таблиця 3.7 – Перелік і опис вхідних документів

Ідентифікатор документа	Найменування документа	Термін надходження	Частота надходження	Постачальник документа
0101304	Інформаційна посилка	За запитом	За запитом	Користувач системи
0101305	Вектор помилок	За запитом	За запитом	Користувач системи
0101306	Матриця маскування	За запитом	За запитом	Користувач системи
0101307	Матриця X	За запитом	За запитом	Користувач системи
0101308	Перестановочна матриця	За запитом	За запитом	Користувач системи

Характеристика масивів вхідної інформації наведена у табл. 3.8.

Таблиця 3. 8 – Характеристика масивів вхідної інформації

Найменування масиву	Ідентифікатор (Ім'я) масиву	Тип масиву	ологія формування
Довідник поля	POLE	НДІ	Формується в єдиній базі даних на підставі документу "Довідник поля"
Довідник альфа	ALPHA	НДІ	Формується в єдиній базі даних на підставі документу "Довідник альфа"
Довідник коефіцієнтів кривої	KOEF	НДІ	Формується в єдиній базі даних на підставі документу «Довідник коефіцієнтів кривої»
Відкритий ключ	PUBLIC KEY	Оперативна інформація	Формується в процесі обробки вхідних даних
Матриці	MATRIX	Оперативна інформація	Формується із вхідних даних
Вектори	VECTORS	Оперативна інформація	Формується із вхідних даних

Форми вихідних повідомлень наведені на рис. 3.15 − 3.17.

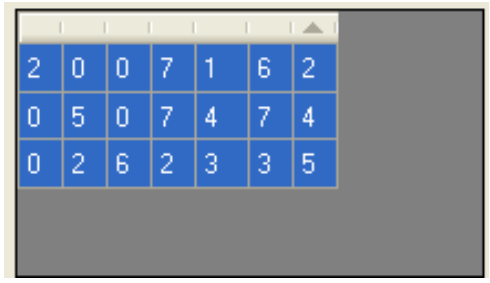


Рисунок 3. 15 – Форма відкритого ключа

Рисунок 3. 16 – Форма вектора помилок

Рисунок 3. 17 – Форма кодового слова

Форми вхідних повідомлень наведені на рис. 3. 18 – 3. 22.

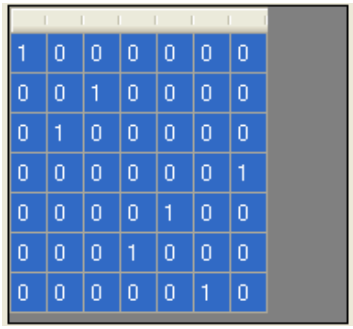


Рисунок 3. 18 – Форма матриці маскування

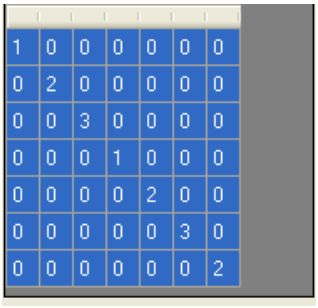


Рисунок 3. 19 – Форма матриці перестановки

Рисунок 3. 20 – Форма матриці X

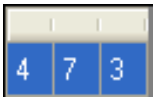


Рисунок 3. 21 – Форма інформаційної посилка

Рисунок 3. 22 – Форма вектора помилок

3.3 Математична постановка комплексу задач модуля

Для формування АГК внесемо ЕК, зафіксуємо кінцеве поле $GF(q)$. Нехай X – гладка проективна алгебраїчна крива в проективному просторі P^n над $GF(q)$, $g = g(X)$ – рід кривій, $X(GF(q))$ – множина її точок над кінцевим полем, $N = X(GF(q))$ – їх загальний обсяг. Нехай C – клас дивізорів на X ступеня $\alpha > g - 1$. Тоді відображення $\varphi: X \to P^{k-1}$, де $k \geq \alpha - g + 1$ визначає клас С. Набір $y_i = \varphi(x_i)$ задає код. Кількість точок у перетинанні $\varphi(X)$ з гіперплощиною дорівнює α, тобто $n - d \leq \alpha$.

Ця математична модель дає можливість будувати коди з параметрами $k + d \geq n - g + 1$, довжина n яким менше або дорівнює кількості точок на кривій X. При $2g < \alpha \leq n$ алгеброгеометричний код має параметри $(n, \alpha - g + 1, d)$, $d \geq n - \alpha$. Двоїстий до нього код також є алгеброгеометричним і має параметри $(n, n - \alpha + g - 1, d^{\perp})$, $d^{\perp} \geq \alpha - 2g + 2$. Розглянемо більш детально алгебро геометричний код. На рис. 3.23 наведена схема алгоритму формування кодограми [12 – 16].

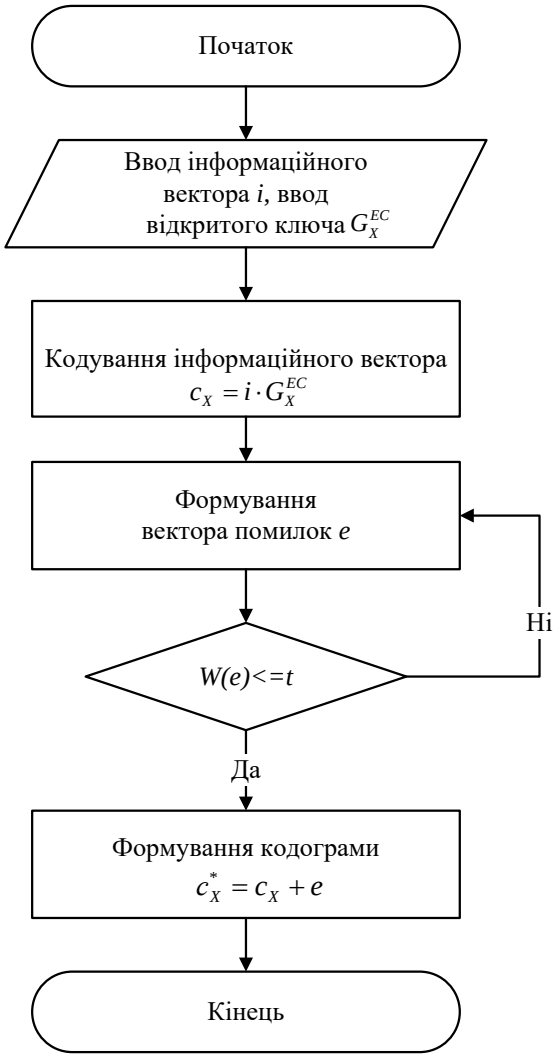


Рисунок 3. 23 – Схема алгоритму формування кодограми

Зафіксуємо кінцеве поле *GF(q)*. Нехай *X* – гладка проективна алгебраїчна крива в проективному просторі P^n над *GF(q)*, $g = g(X)$ – рід кривій, *X(GF(q))* – множина її точок над кінцевим полем, $N = X(GF(q))$ – їх кількість. Нехай *C* – клас дивізорів на *X* ступеня $\alpha > g - 1$. Тоді *C* визначає відображення $\varphi: X \to P^{k-1}$, де $k \geq \alpha - g + 1$. Набір $y_i = \varphi(x_i)$ задає код. Кількість точок у перетинанні $\varphi(X)$ з гіперплощиною дорівнює α, тобто $n - d \leq \alpha$. Ця конструкція дозволяє будувати коди з параметрами $k + d \geq n - g + 1$, довжина n яким менше або дорівнює кількості точок на кривій *X*. При $2g < \alpha \leq n$ алгебро геометричний код має параметри $(n, \alpha - g + 1, d), d \geq n - \alpha$. Двоїстий до нього код також є алгебро геометричним і має параметри $(n, n - \alpha + g - 1, d^{\perp}), d^{\perp} \geq \alpha - 2g + 2$. Дамо наступне визначення алгебро геометричному коду [12-16].

Визначення *1*. Нехай X – гладка проективна алгебраїчна крива в проективному просторі P^n, тобто сукупність рішень однорідного алгебраїчного рівняння ступеня deg X з коефіцієнтами *з GF(q*). Розглянемо різноманіття, що відповідають проективним гіперповерхням, заданим в P^n рівняннями $F = 0$, де F – однорідні одночлени ступеня *degF*. Нехай $I(i_1, i_2, ..., i_n)$ – інформаційна послідовність. Алгеброгеометрічний код за кривою X над $GF(q)$ – це лінійний код довжини $n \le N$, кодові слова $C(c_1, c_2, ..., c_n)$ якого задаються рівністю:

$$\sum_{i=0}^{k-1} i_j F_j(P_i) = c_i ,$$

де $P_i(X_i, Y_i, Z_i)$ – проективні точки кривій X, тобто (X_i, Y_i, Z_i) – рішення однорідного алгебраїчного рівняння, що задають криву X, $i = \overline{1,n}$; $F_j(P_i)$ – значення генераторних функцій у точках кривої.

Це визначення рівносильне матричному поданню алгеброгеометричного коду:

$$G(i_0, i_1, ..., i_{k-1})^T = (c_0, c_1, ..., c_{n-1}),$$

де G – породжувальна матриця k×n, k = α – g+1, α = degX·degF виду:

$$G = \begin{pmatrix} F_0(P_0) & F_0(P_1) & ... & F_0(P_{n-1}) \\ F_1(P_0) & F_1(P_1) & ... & F_1(P_{n-1}) \\ ... & ... & ... & ... \\ F_{k-1}(P_0) & F_{k-1}(P_1) & ... & F_{k-1}(P_{n-1}) \end{pmatrix} = \left\| F_j(P_i) \right\|_{n,k} .$$

Нехай X – невирождена $k \times k$-*матриця* над $GF(q)$, D – діагональна матриця з ненульовими на діагоналі елементами, P – перестановочна матриця розміру $n \times n$. Визначимо несиметричну схему Мак-Еліса з еліптичним кодом:

– відкритий ключ – матриця $G_X^{EC} = X \cdot G^{EC} \cdot P \cdot D$,

– секретний (закритий) ключ – матриці X, P, D.

Кодограма – це вектор довжини n та обчислюється за правилом:

$$c_X^* = i \cdot G_X^{EC} + e ,$$

де вектор $c_X = i \cdot G_X^{EC}$ належить еліптичному *(n, k, d)* коду з погоджувальною матрицею G_X^{EC} , *i* – k-розрядний інформаційний вектор, вектор *e* – секретний вектор помилок ваги ≤ *t*.

На множині проективних точок кривої *X*, зображених в однорідних координатах у вигляді *P(X, Y, 1)*

Перевірочна матриця *H* запишеться у вигляді:

$$H = \begin{pmatrix} 1 & 1 & \dots & 1 \\ X_0 & X_1 & \dots & X_{n-1} \\ \dots & \dots & \dots & \dots \\ Y_0^{\deg F} & Y_1^{\deg F} & \dots & Y_{n-1}^{\deg F} \end{pmatrix}$$

Елементи синдромної послідовності, як елементи вектора $\|S_{lm}\|_r$, обчислимо за правилом

$$S_{lm} = \sum_{i=0}^{n-1} c_i^* X_i^l Y_i^m = \sum_{i=0}^{n-1} e_i X_i^l Y_i^m ,$$

де $l + m \le degF$, або, у матричній формі,

$$\|S_{lm}\|_r = H \left\|c_n^*\right\|_n^T = \left\|X_i^l Y_i^m\right\|_{n,r} \|e_i\|_n^T .$$

Таким чином, задача розкодування алгебро геометричного коду, побудованого через відображення проективних точок *P(X, Y, 1)* кривої однорідними одночленами ступеня *degF,* еквівалентна задачі рішення системи з $r = d + g - 1$ нелінійних рівнянь від $3t$ змінних.

Для рішення цієї задачі скористаємося штучним прийомом, що полягає у введенні в розгляд многочлена локаторів помилок, рішення якого однозначно локалізують (указують місце розташування) виниклих помилок.

Визначимо многочлен локаторів помилок алгеброгеометричного коду як многочлен від двох змінних, ступеня $\le (t - 1)$:

$a_{00} + a_{10}x + \dots + y^{t-1} = 0,$

де *t* – кількість помилок, що може виправити алгеброгеометричний код.

Помноживши обидві частини многочлена (2.9) на e_i і просумувавши по всім i = 0 .. n – 1, значенням у точці (x = X_i, y = Y_i), одержимо рекурентний вираз:

a00S00 + a10 S10 + … + S0 t-1 = 0,

Даний вираз є основою для формування системи лінійних рівнянь для знаходження невідомих коефіцієнтів многочлена локаторів помилок. У матричному способі представлення система лінійних рівнянь має вигляд:

$$\begin{pmatrix} S_{00} & S_{10} & \dots & S_{1\,t-2} \\ S_{10} & S_{20} & \dots & S_{2\,t-2} \\ \dots & \dots & \dots & \dots \\ S_{1\,t-2} & S_{0\,t-2} & \dots & S_{2\,2t-4} \end{pmatrix} \cdot \begin{pmatrix} a_{00} \\ a_{10} \\ \dots \\ a_{1\,t-2} \end{pmatrix} = \begin{pmatrix} -S_{0\,t-1} \\ -S_{1\,t-1} \\ \dots \\ -S_{1\,2t-3} \end{pmatrix}.$$

Після того, як знайшли коефіцієнти многочлена локаторів помилок, потрібно провести процедуру локалізації помилок. Для цього необхідно підставити значення всіх можливих локаторів і вибрати лише ті, які перетворюють у нуль многочлен локаторів помилок. Ця процедура відома в літературі як процедура Ченя [1; 3]. При декодуванні кодів БЧХ у многочлен локаторів помилок підставляються всі елементи поля $X \in GF(q^m)$. При декодуванні алгеброгеометричних кодів у многочлен локаторів підставляються всі пари (X, Y) проективних точок, що ототожнюють всі точки кривої, задані в однорідних координатах P(X, Y, 1).

Після етапу, на якому знайшли локатори помилок, потрібно знайти значення кратності помилки (значення всіх ei ≠ 0). Це можна зробитишляхом підстановки локаторів у систему, що транформується потім в систему лінійних рівнянь.

3.4 Розроблення не функціональних вимог для модуля “Забезпечення конфіденційності та вірогідності даних, що оброблюються в інформаційних системах”

3.4.1 Надійність, безпека та захист інформації

Для забезпечення надійності безпеки та захисту інформації використовуються алгебро геометричні коди основані на еліптичних кривих. Так, використовуючи еліптичні коди достатньо стійку несиметричну кодову систему. Для взлома такої

криптосистеми зловмиснику необхідно декодувати випадковий код – виконати > 10^{100} операцій.

На рис. 3.24 наведені залежності важкості шифрування та дешифрування та важкості зламу.

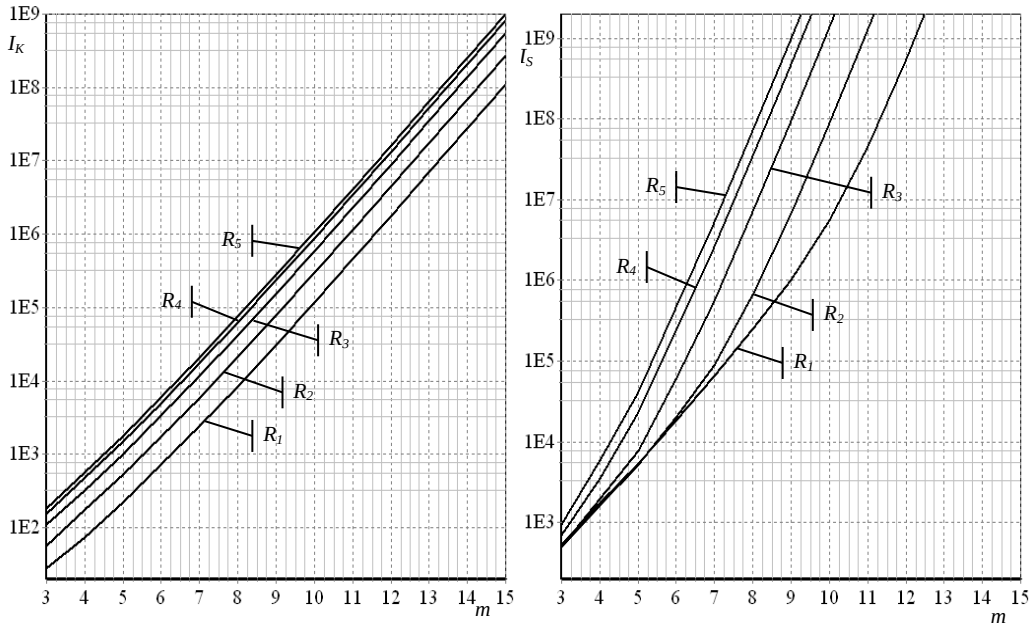


Рисунок 3. 24 – Важкість шифрування та дешифрування над полем GF(2^m)

На рис. 3.25 наведено обсяг відкритих ключових даних теоретико-кодових схем на еліптичних кодах у відповідності до кількох критеріїв стійкості.

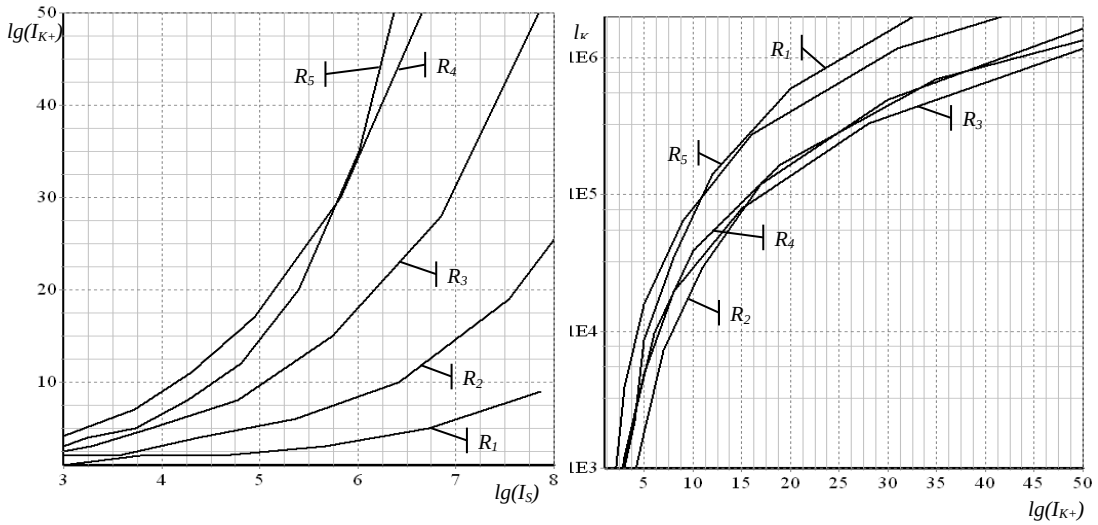


Рисунок 3. 25 – Залежності обсягу відкритих ключових даних теоретико-кодових схем на еліптичних кодах для різних показників стійкості

В результаті проведенних досліджень вірогідності помилок декодування передаваемих кодограмм і стійкості к зламу методом перестановочного декодування встановлено, що достовірність і інформаційна скритність передачи даних з використанням теоретико-кодових схем на елліптичних кодах перевищує аналогічні показники для теоретико-кодових схем на кодах Ріда-Соломона.

Формування кодових слів еліптичних кодів здійснюється алгоритмами з поліноміальною складністю від довжини коду і його виправляючої здібності. Це дозволяє побудувати несиметричні теоретико-кодові схеми з високою швидкістю обробки переданих даних. Декодування еліптичних кодів здійснюється алгоритмами з поліноміальним показником складності від довжини коду і його виправляючої здібності

На рис. 3.26 – 3.27 наведений ЕВК над GF(2^3).

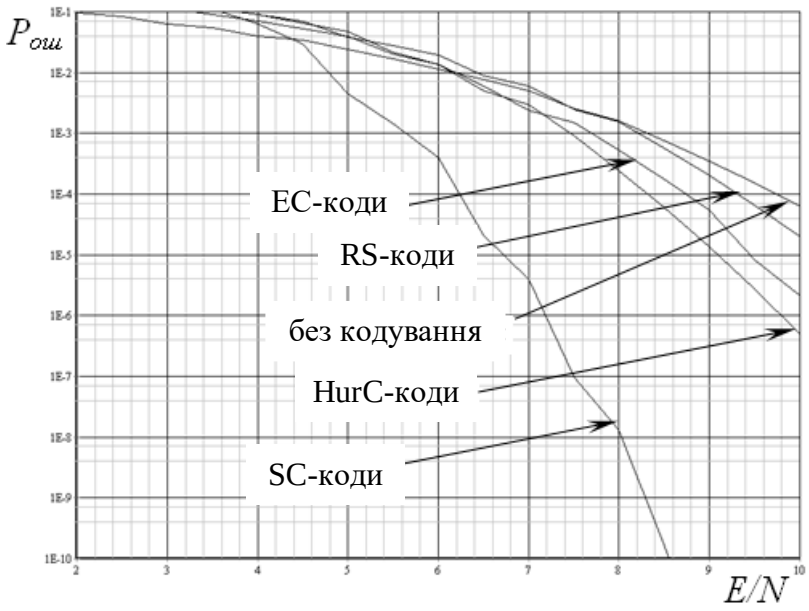


Рисунок 3. 26 – Порівняльний аналіз ЕВК над GF(2^3) R=1/2

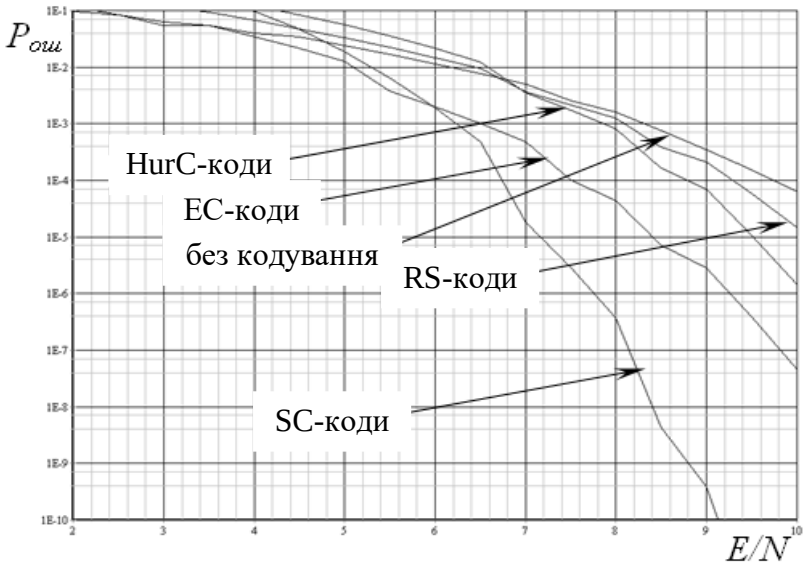


Рисунок 3. 27 – Порівняльний аналіз ЕВК над GF(2^3) R=1/3

3.4.2. Опис архітектури системи

На рис. 3.28 наведена діаграма розміщення системи, що розробляється.

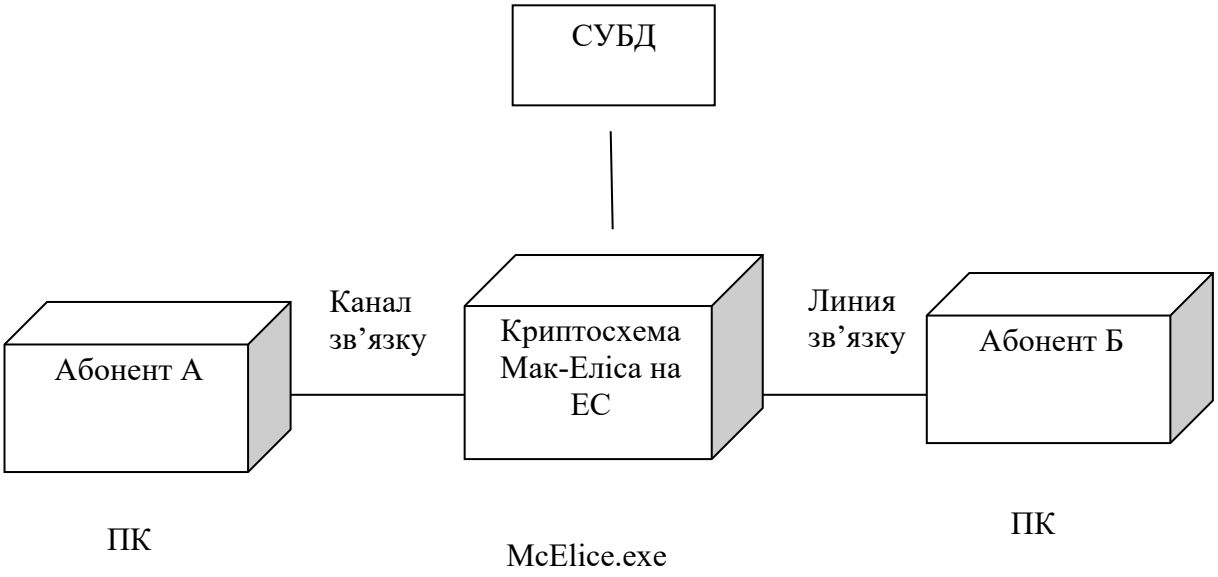


Рисунок 3. 28 – Діаграма розміщення для задачі "Забезпечення конфіденційності та вірогідності даних"

В якості ліній зв'язку можуть виступати різні телекоммуникаційні системи, спеціалізованого або комерційного характеру.

В якості СУБД виступає Microsoft Access 2007.

3.5 Опис інформаційних потоків

Для опису інформаційних потоків скористаємось методологією DFD. Бо такий тип діаграм зручно використовувати для опису документообігу та обробки інформації. На вході використовується вектор помилок, інформаційна посилка, перестановочна матриця, матриця маскування, та матриця X. Контекстна діаграма “Забезпечення конфіденційності та вірогідності даних” та декомпозиція контекстної діаграми, яка складається з наступних робіт, які наведені на рис. 3.29 – 3.30:

- формування відкритого та закритого ключів;
- формування кодової послідовності;
- розкодування кодової послідовності.

В якості довідників використовуються наступні таблиці: таблиця векторів; таблиця матриць; таблиця коефіцієнтів; таблиця розміру поля.

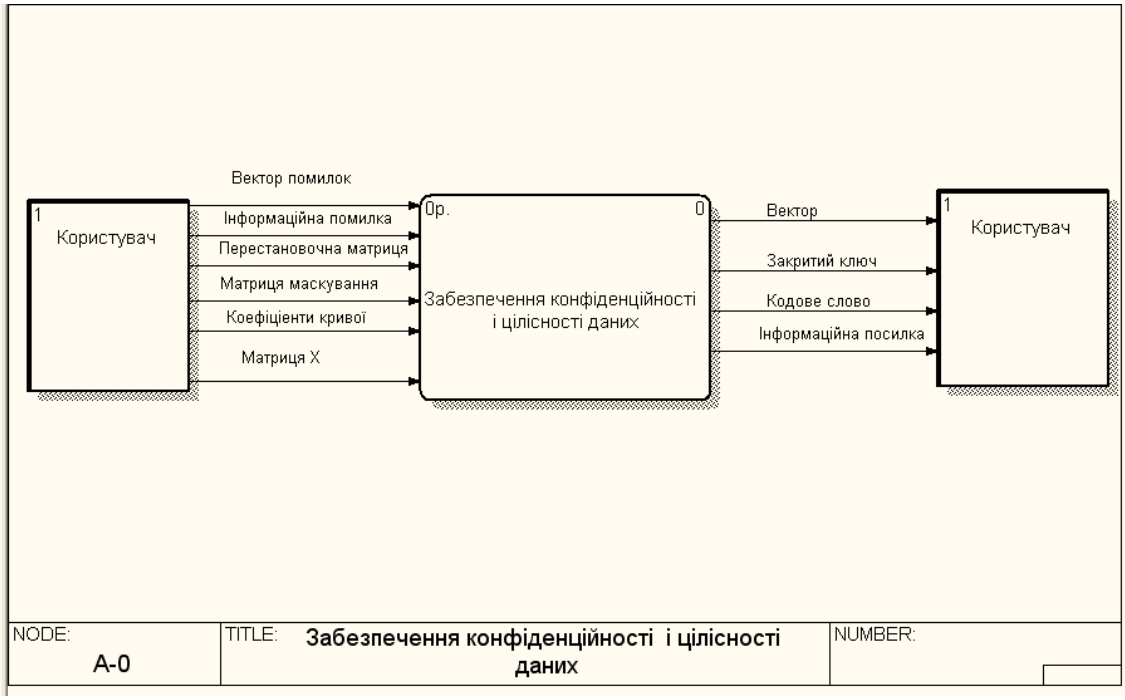


Рисунок 3. 29 – . Контекстна діаграма “Забезпечення конфіденційності та вірогідності даних” в DFD

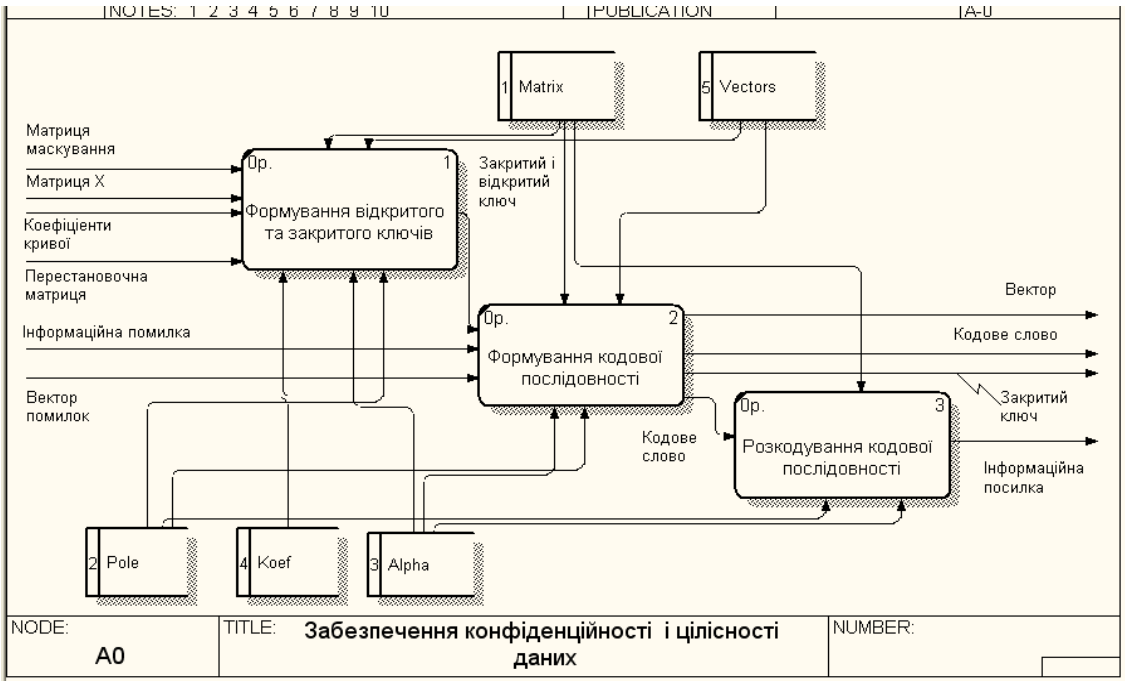


Рисунок 3. 30 – Декомпозиція контекстної діаграми “Забезпечення конфіденційності та вірогідності даних” в DFD

3.6 Проектування структури бази даних

Проектування бази даних є складовою частиною при розробці інформаційної системи. Схема БД, виконана в середовищі Erwin. На рис. 3.31 наведена логічна модель БД, а на рис. 3.32 наведена фізична модель БД.

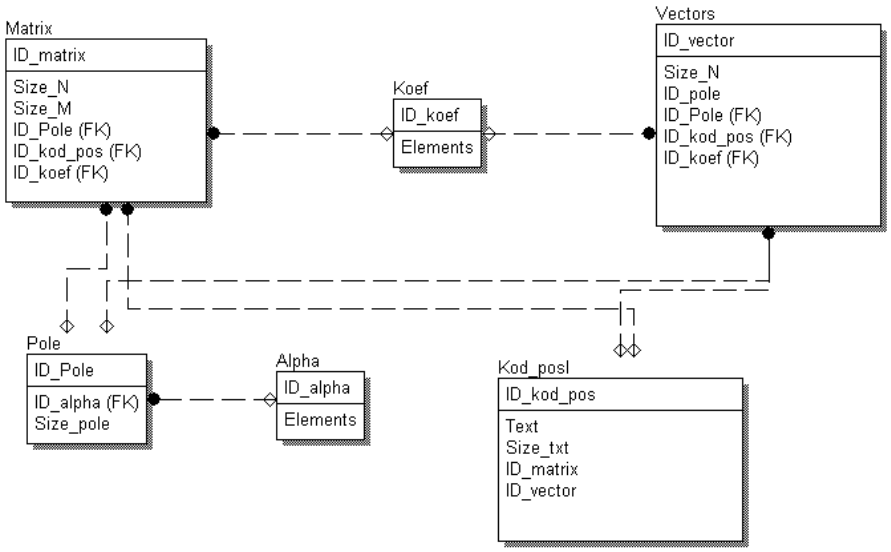


Рисунок 3. 31 – Логічна схема бази даних

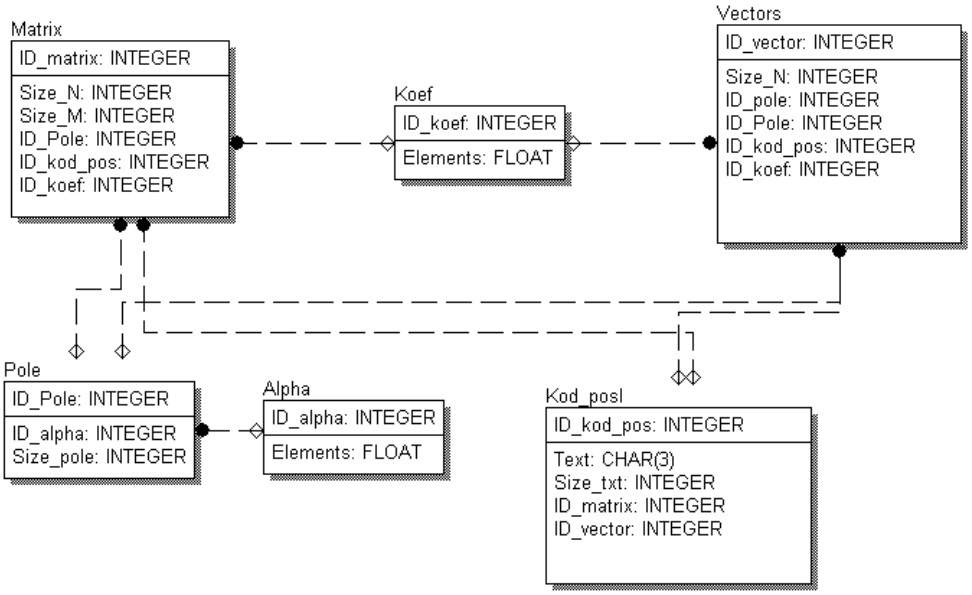


Рисунок 3. 32 – Фізична схема бази даних

В табл. 3.9 наведений інформаційний список документів.

Таблиця 3.9 – Інформаційний список документів

Код документа	Назва	Вхідний/вихідний
0101301	Кодова послідовність	Вихідний
0101302	Інформаційна посилка	Вихідний
0101303	Відкритий ключ	Вихідний
0101304	Інформаційна посилка	Вхідний
0101306	Матриця маскування	Вхідний
0101307	Матриця X	Вхідний
0101308	Перестановочна матриця	Вхідний

В табл. 3.10 наведений список реквізитів вхідних вихідних документів

Таблиця 3. 10 – Список реквізитів вхідних вихідних документів

№ п.п.	Назва реквізиту	Фактичний розраховуємий	Призначення реквізити
1	ID_matrix	фактичний	Первиний ключ таблиці Matrix
2	Size_N	розраховуємий	Кількість рядків матриці
3	Size_M	розраховуємий	Кількість столбців матриці
4	ID_pole	фактичний	Первиний ключ таблиці Pole
5	ID_kod_pos	фактичний	Первиний ключ таблиці Kod_posl
6	ID_koef	фактичний	Первиний ключ таблиці Koef
7	ID_alpha	фактичний	Первиний ключ таблиці Alpha
8	Size_pole	фактичний	Розмір поля Галуа
9	Elements	розраховуємий	Елементи поля
10	Text	розраховуємий	Текст кодової посилки
11	Size_Text	розраховуємий	Розмір кодової посилки

Обмеження атрибутів сутностей наведені в табл. 3.11.

Таблиця 3. 11 – Обмеження атрибутів

№ п/п	Назва атрибуту	Тип	Розмір	Границі	Умова	Значення за замовчуванням
1	2	3	4	5	6	7
1	ID_matrix	INTEGER	-	-	-	-
2	Size_N	INTEGER	-		>8	3
3	Size_M	INTEGER	-		<8	3
4	ID_pole	INTEGER	-	-	-	-
5	ID_kod_pos	INTEGER	-	-	-	-
6	ID_koef	INTEGER	-	-	-	-
7	ID_alpha	INTEGER	-	-	-	-
8	Size_pole	INTEGER	-	-	-	3
9	Elements	FLOAT	7	0 – 7	-	0
10	Text	CHAR	1024	-	< 1024	-
11	Size_Text	INTEGER	1024	-	< 1024	1

3.7 Розроблення програмного продукту “Криптосистема МакЕліса на еліптичних кривих”

3.7.1 Розроблення інтерфейсу програмного продукту.

Інтерфейс розроблено за допомогою діаграми станів (рис. 3.33). Це дало можливість унаочнити та візуалізувати архітектуру розробленого ПЗ.

Діаграму станів можна представити у вигляді спеціального графу, яким представляють деякий автомат. У вершинах графу розташовані можливі стани автомата, а переходи зі стану в стан відповідають дугам цього графу. Діаграми станів можуть бути представлені у більш складному вигляді, вкладені одна в одну для більш детального представлення окремих елементів моделі.

Різні структури взаємодії забезпечують різні ступені гнучкості для користувачів. Зазвичай, чим гнучкіше структура, тим більше вона вимагає від користувача навчання, розуміння, і часу на роботу з вікнами (відкрити, закрити, розмістити і т.д.)

На діаграмі станів наведені усі компоненти інтерфейсу розробленого модуля такі як: форми, кнопки, таблиці даних та інші, що дає повне представлення про інтерфейс.

Розроблений інтерфейс модуля відповідає усім вимогам щодо ергономічності та інтуїтивності.

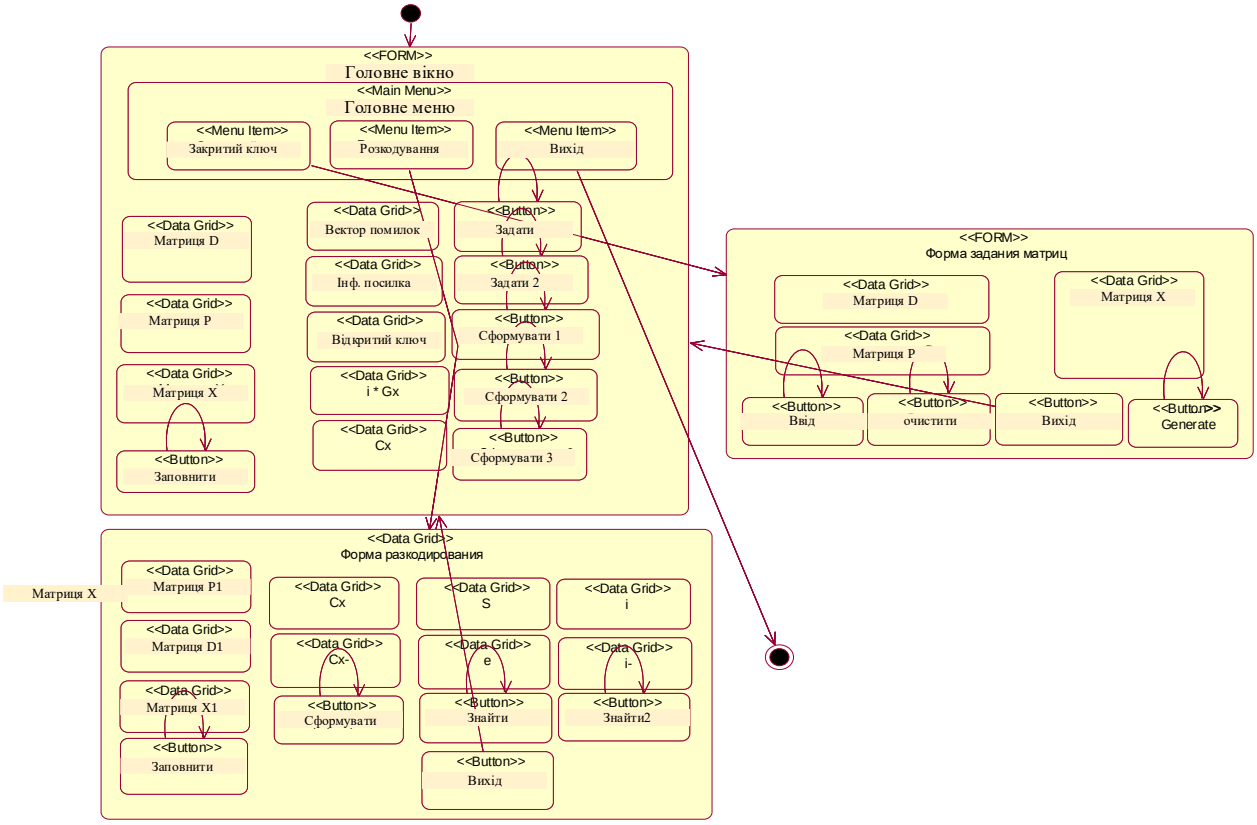


Рисунок 3. 33 – Діаграма станів системи “Забезпечення конфіденційності та вірогідності даних”

3.7.2 Склад та взаємодія програмних модулів

Діаграми діяльності, які зображують схему технологічного процесу зображені нижче на рис. 3.34 – 3.37.

Діаграма класів формування вихідних та вхідних документів модуля “Забезпечення конфіденційності та вірогідності даних” наведена на рис. 3.37.

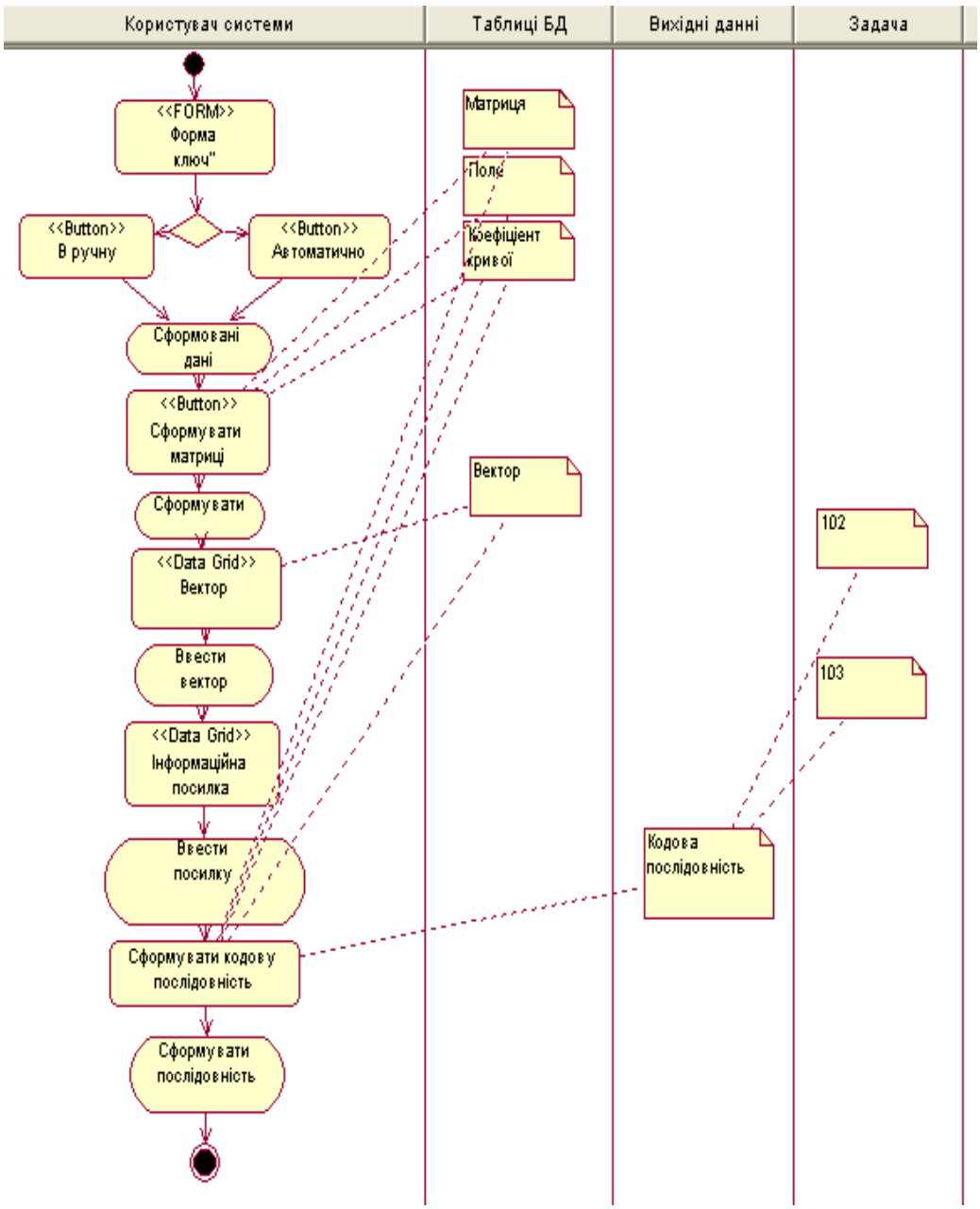


Рисунок 3. 34 – Діаграма діяльності початку роботи з системою

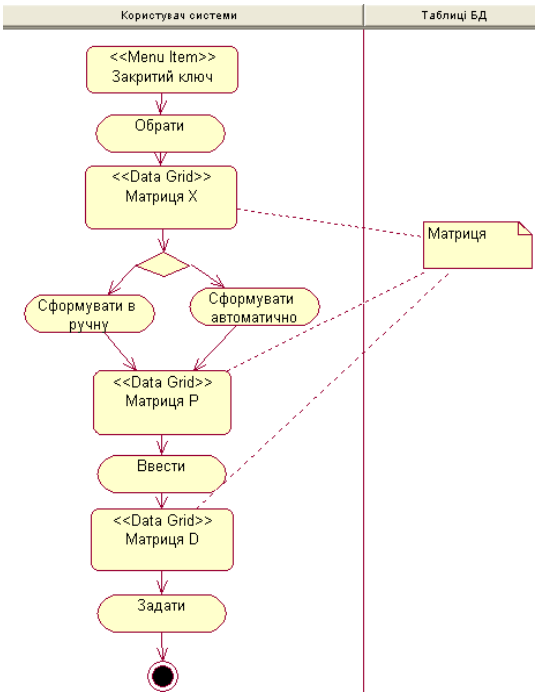


Рисунок 3. 35 – Діаграма діяльності початку роботи з пунктом меню “Закритий ключ”

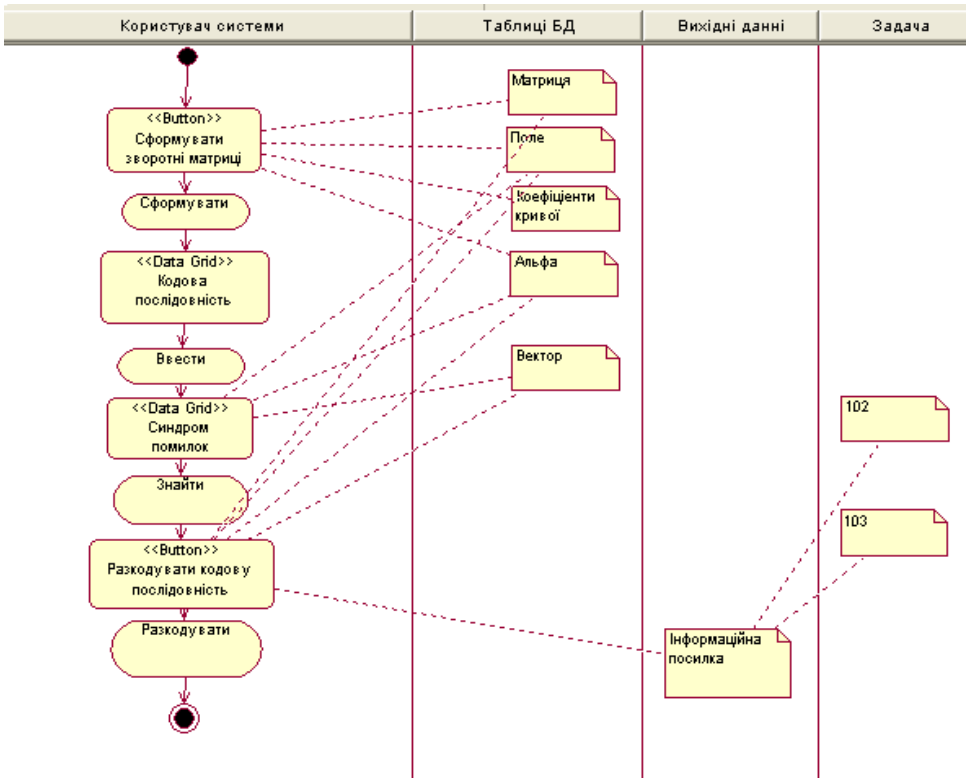


Рисунок 3.36 – Діаграма діяльності початку роботи з пунктом меню “Розкодування інформації”

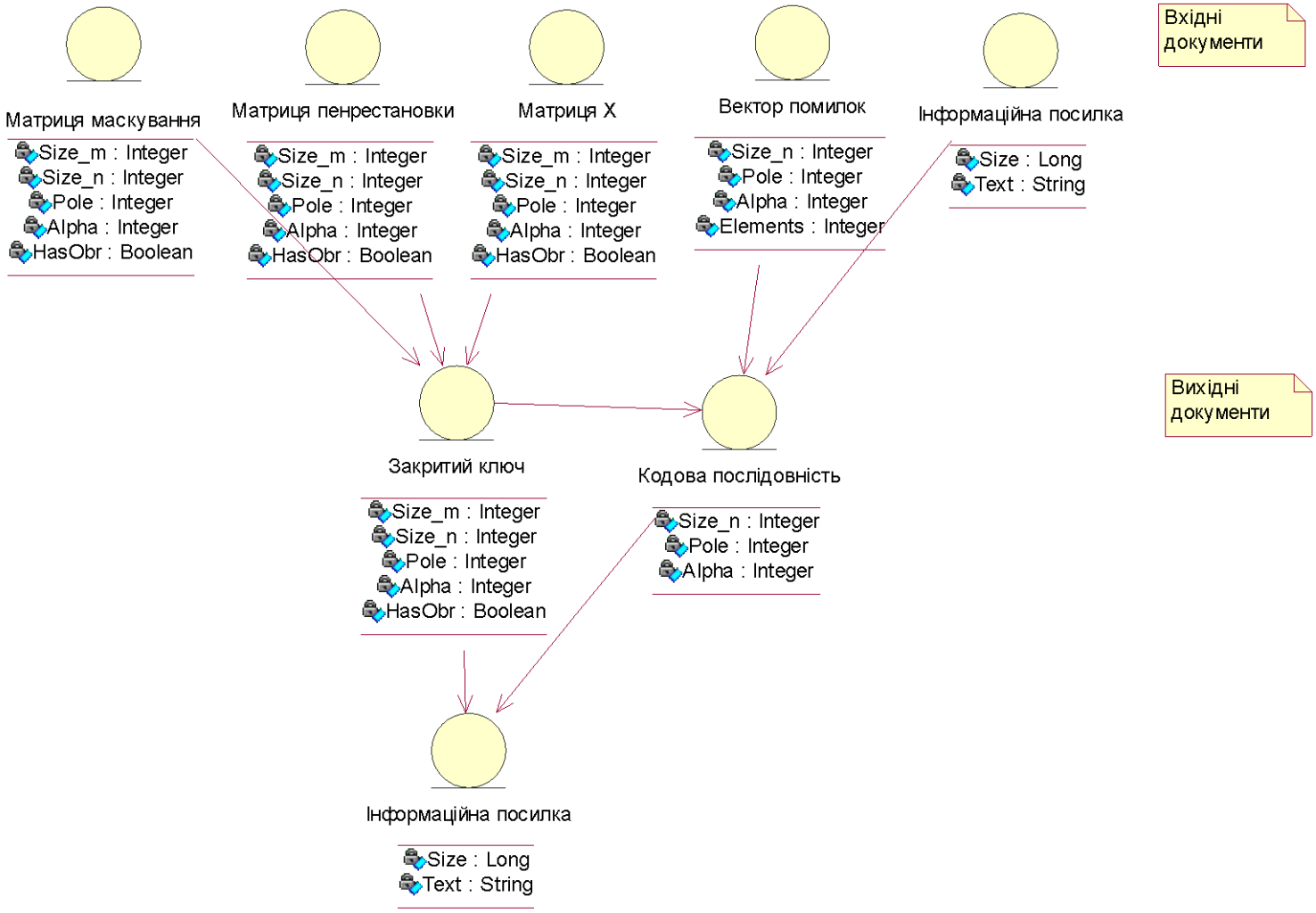


Рисунок 3. 37 – Діаграма класів модуля “Забезпечення конфіденційності та вірогідності даних”

3.7.3 Заходи щодо забезпечення захисту інформації, реалізовані у програмному продукту

Для забезпечення захисту інформації в програмному продукті реалізовані нессеметричні методи шифрування зокрема, схема МакЕліса, на еліптичних кодах. Запропонований метод побудови несиметричних теоретико-кодових схем на еліптичних кодах, грунтується на використанні теоретико-важкої проблеми декодування випадкового коду. Випадковий код одержують маскуванням еліптичного коду так, щоб для не уповноваженого користувача криптосистеми завдання декодування представлялася важкою з експоненціальним показником складності, а для уповноваженої користувача існувала лазівка – алгоритм декодування з поліноміальним показником складності.

Несиметричні методи шифрування з використанням еліптичних кодів володіють високими показниками швидкодії. Алгоритми кодування і декодування еліптичних кодів, що лежать в основі процедур шифрування і дешифрування криптограм, мають поліноміальний показник складності

Як показали проведені дослідження, теоретико-кодові схеми мають високі показники швидкодії і потенційної стійкості. Так, використовуючи еліптичні коди з R = 0,5 над GF (2^{10}), можна побудувати несиметричну теоретико-кодову схему зі складністю шифрування $<10^{15}$ групових операцій. Для зламу такий криптосистеми зловмисникові необхідно декодувати випадковий код – виконати $> 10^{100}$ операцій (при переставному декодуванні). Платою за такі параметри несиметричною теоретико-кодової схеми є великі обсяги ключових даних (у розглянутому прикладі десятки мегабіт). Таким чином, основним недоліком несиметричних криптосистем, побудованих з використанням теоретико-кодових схем, є великий обсяг ключових даних.

3.8 Результати тестування програмного продукту

Основна перевага ТКС на ЕК полягає у високій швидкості криптографічного перетворення інформації та інтеграції завадостійкого кодування з шифруванням.

Побудовані теоретико-кодові схеми на еліптичних кодах володіють високими показниками ефективності: їх застосування дозволяє забезпечити несиметричну обробку інформації для ефективного захисту переданих даних від випадкового або навмисного впливу.

Важливим питанням практичного використання теоретико-кодових схем на еліптичних кодах для забезпечення захисту переданої інформації є дослідження їх безпеки.

Дослідження безпеки теоретико-кодових схем проводилися у відповідності з методикою тестування NIST SP 800-22, рекомендованої Національним інститутом з стандартизації і технологій США.

Для проведення тестування були взяті наступні параметри:

- довжина тестової послідовності n = 106 біт;
- кількість тестових послідовностей m = 100;

Таким чином, обсяг тестової вибірки становить:

- N = 106 x 100 = 108 біт;
- кількість (q) для різних довжин q = 189, таким чином, статистичний портрет генератора становить 18900 значень імовірності P. Результати тестування теоретико-кодових схем на еліптичних кодах наведені зведені в табл. 3.12.

Таблиця 3. 12 – Результати тестування

	Теоретико-кодові схеми на еліптичних кодах	Теоретико-кодові схеми на кодах Ріда-Соломона	BBS	FIPS 197
Кількість тестів, у яких тестування пройшло 99% послідовностей	141	132	134	126
Кількість тестів, у яких тестування пройшло 96% послідовностей	189	188	189	189
Кількість тестів, в яких значення ймовірності P <= 0,01	2	2		0
Кількість тестів, в яких значення ймовірності P <= 0,001	1	0		0
Кількість тестів, в яких значення ймовірності P <= 0,05	13	8		8
Допустиме значення частки проходження тесту для вибірки розміром 100 двоечних послідовностей дорівнює	0,96015			
Допустиме значення частки проходження тесту для вибірки розміром 71 двоечних послідовностей для тесту Random-Excursion дорівнює	0,954575			

У табл. 3.11 наведені так само результати статистичного дослідження теоретико-кодових схем на кодах Ріда - Соломона, докладно розглянуті і результати дослідження алгоритму блокового симетричного шифрування FIPS 197. У графі BBS наведені дані, які відповідають тестової послідовності генератора псевдовипадкових чисел BBS, рекомендованої Національним інститутом з стандартизації і технологій США і поставляється разом з пакетом тестів NIST SP 800-22.

На рис. 3.38 наведений статистичний портрет програмної реалізації теоретико-кодових схем, побудованих по кривій $y^2 + xy = x^3 + x^2 + 1$ над $GF(2^3)$.

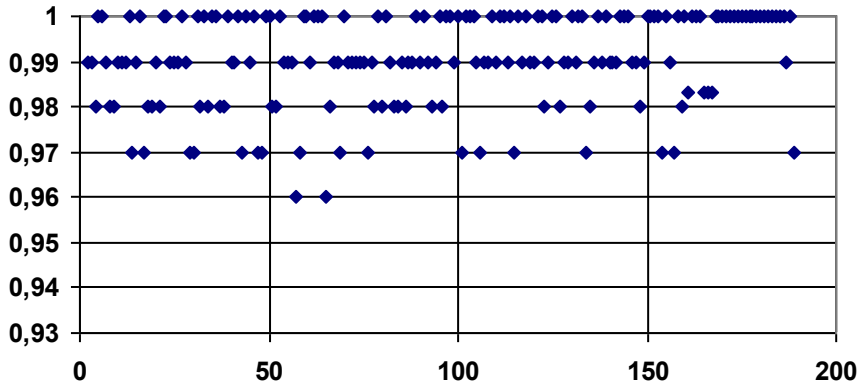


Рисунок 3.38 – Статистичний портрет програмної реалізації ТКС, побудованих по еліптичній кривій $y^2 + xy = x^3 + x^2 + 1$ над $GF(2^3)$

Статистичний портрет описується діаграмою ймовірностей проходження відповідних статистичних тестів. Результати статистичного портрету програмної реалізації теоретико-кодових схем, побудованих по еліптичній кривій, свідчать про те, що більше 96% послідовностей відповідає поставленим вимогам.

У таблиці 3.12 наведено результати тестування програмної реалізації несиметричних теоретико-кодових схем, побудованих по еліптичних кодами, кодами Ріда – Соломона, тестової послідовності генератора псевдовипадкових чисел BBS і генетарора випадкових чисел з використанням алгоритму блокового симетричного шифрування FIPS 197 в режимі лічильника. Наведені в в табл. 3.12 дані свідчать про те, що генератори на теоретико-кодових схемах володіють хорошими статистичними властивостями. Дійсно, за результатами дослідження статистичної безпеки видно, що кодові схеми захисту інформації забезпечують проходження тестів з більшою ймовірністю, ніж тестовий генератор псевдовипадкових чисел BBS і алгоритм блокового симетричного шифрування FIPS 197.

Таблиця 3.12 – Результати тестування

Генератор	Кількість тестів, у яких тестування пройшло більше 99% послідовностей	Кількість тестів, у яких тестування пройшло більше 96% послідовностей
BBS	134 (71%)	189 (100%)
FIPS 197	126 (67%)	189 (100%)
Теоретико-кодові схеми на еліптичних кодах	141 (74%)	189 (100%)
Теоретико-кодові схеми на на кодах Ріда-Соломона	132 (69%)	188 (99%)

Проведено експериментальні дослідження статистичної безпеки кодових схем, побудованих з використанням теоретико-кодових схем на еліптичних кодах. Оцінка проводилась за стандартом NIST SP 800-22. Несиметричні теоретико-кодові схеми, побудовані по еліптичних кривих успішно пройшли всі тести з довірчим рівнем Pi > 0,96015. Майже 75% тестів успішно пройшли тести довірчим рівнем Pi > 0,99, що є кращим результатом, в порівнянні з генератором BBS і алгоритмом блокового шифрування FIPS 197.

3.9 Контрольний приклад

Дано: Алгебраїчна крива $x^3+y^2z+yz^2=0$ над полем $GF(2^2)$, точки кривої:

	P1	P2	P3	P4	P5	P6	P7	P8	P9
X	0	0	0	1	2	3	1	2	3
Y	1	0	1	2	2	2	3	3	3
Z	0	1	1	1	1	1	1	1	1

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 2 & 3 & 1 & 2 & 3 \\ 0 & 1 & 2 & 2 & 2 & 3 & 3 & 3 \\ 0 & 0 & 1 & 3 & 2 & 1 & 3 & 2 \\ 0 & 0 & 2 & 3 & 1 & 3 & 1 & 2 \\ 0 & 1 & 3 & 3 & 3 & 2 & 2 & 2 \end{bmatrix} \quad G = \begin{bmatrix} 2 & 2 & 3 & 0 & 1 & 3 & 0 & 1 \\ 3 & 3 & 2 & 1 & 0 & 2 & 1 & 0 \end{bmatrix} \quad X = \begin{bmatrix} 1 & 2 \\ 3 & 0 \end{bmatrix} \quad X^{-1} = \begin{bmatrix} 0 & 2 \\ 3 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad D^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad P^{-1} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \quad i = 11, \ e = 00100200$$

Знаходим відкритий ключ $G_x = X \times G \times P \times D$

$$G_x = \begin{bmatrix} 1 & 2 \\ 3 & 0 \end{bmatrix} \times \begin{bmatrix} 2 & 2 & 3 & 0 & 1 & 3 & 0 & 1 \\ 3 & 3 & 2 & 1 & 0 & 2 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} 2 & 1 & 3 & 0 & 1 & 1 & 1 & 0 \\ 0 & 2 & 2 & 2 & 2 & 0 & 3 & 2 \end{bmatrix}$$

Знаходим $C_x = i \times G_x + e$ $\quad C_x = 11 \times \begin{bmatrix} 2 & 1 & 3 & 0 & 1 & 1 & 1 & 0 \\ 0 & 2 & 2 & 2 & 2 & 0 & 3 & 2 \end{bmatrix} \oplus 00100200 = 23023322;$

C_x = 23023322– в канал;

Знаходим $C_x^* = C_x \times D^{-1} \times P^{-1}$

$$C_x^* = 23023322 \times \begin{bmatrix} 1&0&0&0&0&0&0&0 \\ 0&3&0&0&0&0&0&0 \\ 0&0&2&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&3&0&0&0 \\ 0&0&0&0&0&2&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \end{bmatrix} \times \begin{bmatrix} 0&0&0&1&0&0&0&0 \\ 1&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&1&0&0 \end{bmatrix} = 22202221$$

$C_x^* = 22202221$

Знаходим $S = C_x \times H^T$,

$$S = 22202221 \times \begin{bmatrix} 1&1&1&1&1&1&1&1 \\ 0&0&1&2&3&1&2&3 \\ 0&1&2&2&2&3&3&3 \\ 0&0&1&3&2&1&3&2 \\ 0&0&2&3&1&3&1&2 \\ 0&1&3&3&3&2&2&2 \end{bmatrix}$$

S_{00}= 1

S_{10}= 2+1+2+3+3=1

S_{01}= 2+3+3+1+1+3=1

S_{20}= 2+3+2+1+2=0

S_{11}= 3+2+1+2+2=0

S_{02}=2+1+1+3+3+2=0

S = (1,1,1,0,0,0);

Знаходимо многочлен локаторів помилок $\Lambda(x) = a_{00}+a_{10}x+y = 0$

$$\begin{bmatrix} S_{00} & S_{10} \\ S_{10} & S_{20} \end{bmatrix} \times \begin{bmatrix} a_{00} \\ a_{01} \end{bmatrix} = \begin{bmatrix} S_{01} \\ S_{11} \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \qquad a_{00} = 0; \quad a_{10} = 1;$$

$\Lambda(xy) = x+y = 0$ – многочлен локаторів помилок

Находим локатори помилок по процедуре Ченя.

$P_1(0,0,1)\ \Lambda(x,y) = 0 + 0 = 0$ – помилка

$P_2(0,1,1)\ \Lambda(x,y) = 0 + 1 = 1$

$P_3(1,2,1)\ \Lambda(x,y) = 1 + 2 = 3$

$P_4(2,2,1)\ \Lambda(x,y) = 2 + 2 = 0$ – помилка

$P_5(3,2,1)\ \Lambda(x,y) = 3 + 2 = 1$

$P_6(1,3,1)\ \Lambda(x,y) = 1 + 3 = 2$

$P_7(2,3,1)\ \Lambda(x,y) = 2 + 3 = 1$

$P_8(3,3,1)\ \Lambda(x,y) = 3 + 3 = 0$ – помилка

$e^* = e_1 00 e_4 000 e_8$

Знаходим : $e^* \times H = S$, розв’язавши систему рівнянь получимо

$e_1 = 0,\ e_4 = 2,\ e_8 = 3$

$e^* = 00020003$

Знаходим $i^* = e^* + C_x^*$

$$i^* = 00020003 \oplus 22202221 = 22;$$

Знаходим $i = i^* \times X^{-1}$

$$i = 22 \times \begin{bmatrix} 0 & 2 \\ 3 & 1 \end{bmatrix} = 11 \qquad i = i$$

3.10 Впровадження та супровід програмного продукту

Для установки програмного продукту достатньо скопіювати папку Криптосистема Макеліс на ЕК у вказану дерікторію і далі запустити файл MacEliss.exe, який знаходиться всередині папки, після чого запуститися головне вікно програми.

Для коректної роботи програмного продукту на комп’ютері повинен бути встановлений. NET Framework версії не нижче 4.0, а також SQLServer 2005 – 2008.

Даний програмний продукт може працювати під керівництвом операційних систем: Windows XP, Windows Vista і Windows Seven.

Системні вимоги:

Процесор не нижче Pentium 4 3.0 Gh;

512 і вище ОЗУ.

Для зручності користувачів створена інтерактивна довідкова система по роботі з програмним продуктом.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Охорона праці

При дослідженні асиметричного шифрування на еліптичних кодах важливим є дотримання вимог охорони праці, техніки безпеки та протипожежної безпеки при роботі з комп’ютерною технікою. Основними регламентуючими нормативними документами охорони праці користувачів ПК є НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями», а також Примірна інструкція з охорони праці під час експлуатації електронно-обчислювальних машин, затверджена наказом Міністерства доходів і зборів України від 5 вересня 2013 р. № 443, ДБН В.2.5-56:2014 «Системи протипожежного захисту».

Організація робочого місця користувача відеотерміналу та ЕОМ повинна забезпечувати відповідність усіх елементів робочого місця та їх розташування ергономічним вимогам ДСТУ 8604:2015 «Дизайн і ергономіка. Робоче місце для виконання робіт у положенні сидячи. Загальні ергономічні вимоги»

Заходи з охорони праці користувачів персонального комп’ютера розглядаються в трьох основних аспектах: соціальному, психологічному та медичному.

У соціальному плані розв'язання цих проблем пов'язане з оптимізацією умов життя, праці, відпочинку, харчування, побуту, розвитком культури, транспорту.

Значне місце у профілактиці розладів здоров'я належить психології праці. Тому заходи, пов'язані з формуванням раціональних виробничих колективів, у яких відсутня психологічна несумісність, сприяють зменшенню нервово-психічного перенапруження, підвищенню працездатності та ефективності праці.

Оскільки відразу цю проблему усунути неможливо на рівні підприємств, установ та організацій послідовно усували такі виробничі умови, які є сприятливими для розвитку стресу.

Важливу роль у профілактиці захворювань відводиться медицині. Комплекс даних заходів спрямований на відновлення функціонального стану зорового та опорно-рухового апарату, також направлений на зниження ймовірності розвитку перевтоми та перенапруження.

Для забезпечення безпеки на робочому місці проводиться розробка національних нормативних документів, спрямованих на охорону праці користувачів ПК. Найбільш повним нормативним документам щодо забезпечення охорони праці користувачів ПК є «Державні санітарні правила й норми роботи з візуальними дисплейними терміналами (ВДТ) електронно-обчислювальних машин» ДСанПіН 3.3.2.007–98.

До організації робочого місця користувача ПК висуваються ряд вимог серед яких: вимоги до виробничих приміщень, гігієнічні вимоги до організації та обладнання робочих місць, вимоги до режимів праці та відпочинку та вимоги до профілактичних медичних оглядів.

Вимоги до виробничих приміщень полягають у наступному:

– площа на одне робоче місце становить не менше 6 м2, а об’єм не менше ніж 20 м3;

– приміщення для роботи повинно мати штучне та природне освітлення;

– необхідно щоденно робити вологе прибирання.

Природне освітлення має здійснюватись через світлові прорізи, орієнтовані переважно на північ чи північний схід, і забезпечувати коефіцієнт природної освітленості (КПО) не нижче, ніж 1,5%.

Гігієнічні вимоги до параметрів виробничого середовища включають вимоги до параметрів мікроклімату, освітлення, шуму й вібрації, рівнів електромагнітного та іонізуючого випромінювання. У виробничих приміщеннях на робочих місця із ПК мають забезпечуватись оптимальні значення параметрів мікроклімату: температури, відносної вологості й рухливості повітря.

Норми мікроклімату для приміщень з ПК:

– у холодну пору року температура повітря при категорії робіт легка-1а має бути 22-24 ºС, вологість 40-60%;

– у холодну пору року температура повітря при категорії робіт легка-1б температура має бути 21-23 ºС, вологість також 40-60 %;

– у теплу пору року при категорії робіт легка-1а температура в приміщеннях має бути 23-25 ºС з вологістю 40-60%;

– у теплу пору року при категорії робіт легка-1б температура має бути 22-24 ºС з вологістю 40-60%.

Штучне освітлення в приміщеннях із робочими місцями, обладнаними ПК має здійснюватись системою загального рівномірного освітлення.

Значення освітленості на поверхні робочого столу в зоні розміщення документів має становити 300–500 лк, згідно з ДБН В.2.5–28 – 2006 «Природне та штучне освітлення». Якщо це неможливо забезпечити системою загального освітлення, допускається використовувати місцеве освітлення.

Конструкція робочого місця користувача ПК має забезпечити підтримання оптимальної робочої пози.

Робочі місця із ПК розташовуються відносно світлових прорізів, щоб природне світло падало збоку, переважно зліва.

При розміщенні робочих столів із персональним комп’ютером дотримуються таких відстаней: між бічними поверхнями БДТ – 1,2 м; від тильної поверхні одного ПК до екрана іншого – 2,5 м.

Екран ПК розташовують на оптимальній відстані від очей користувача, що становить від 60 до 70 см, але не ближче, ніж за 60 см з урахуванням розміру літерно-цифрових знаків і символів.

Клавіатуру розташовують на поверхні столу на відстані від 10 до 30 см від краю, звернутого до працівника.

Для забезпечення захисту і досягнення нормованих рівнів комп'ютерних випромінювань застосовують екранні фільтри, локальні світлофільтри (засоби індивідуального захисту очей) та інші засоби захисту, що пройшли випробування в акредитованих лабораторіях і мають щорічний гігієнічний сертифікат.

При організації праці, пов'язаної з використанням ПК, для збереження здоров'я працівників, запобігання професійним захворюванням і підтримки працездатності передбачаються внутрішньо змінні регламентовані перерви для відпочинку.

Працівники з ПК підлягають обов'язковим медичним оглядам: попереднім – при влаштуванні на роботу і періодичним – протягом трудової діяльності, відповідно до наказу МОЗ України N45 від 31.03.94 р.

Періодичні методичні огляди мають проводитися раз на два роки комісією в складі терапевта, невропатолога та офтальмолога.

Охорона праці робочого місця користувача персонального комп’ютера є дуже важливою, оскільки не дотримання вимог може призвести до професійних хвороб.

4.2 Безпека життєдіяльності

Трудова діяльність користувачів комп'ютерів (ВДТ) відбувається у певному виробничому середовищі, яке впливає на їх функціональний стан. Найбільш значимі — фізичні фактори виробничого середовища, до яких належать електромагнітні хвилі різних частотних діапазонів, електростатичні поля, шум, параметри мікроклімату та ціла низка світлотехнічних показників. Вплив хімічних та, особливо, біологічних факторів виробничого середовища на користувачів комп'ютерів — значно менший.

Трудовий процес суттєво впливає на психофізіологічні можливості користувачів комп'ютерів, оскільки їх діяльність характеризується значними статичними фізичними навантаженнями; недостатньою руховою активністю; напруженнями сенсорного апарату, вищих нервових центрів, які забезпечують функції уваги, мислення, регуляції рухів. Окрім того, трудовий процес користувачів комп'ютерів відзначається значними інформаційними навантаженнями.

Професійні якості та виробничий досвід, які визначають внутрішні засоби діяльності, обумовлюють надійну та безпомилкову діяльність користувачів комп'ютерів, дозволяють знаходити безпечні методи розв'язання виробничих завдань навіть у нестандартних ситуаціях

Зовнішні засоби діяльності, які в основному визначаються ергономічними показниками щодо організації робочого місця, форми та параметрів його елементів, просторового розташування основного і допоміжного устаткування, можуть суттєво знизити фізичні та психофізіологічні навантаження, що діють на користувачів комп'ютерів.

У професійних операторів частіше зустрічаються порушення органів зору, опорно-рухового апарату, центральної нервової, серцево-судинної, імунної та статевої систем, захворювання шкіри. Зафіксована значна кількість скарг операторського персоналу на загальне недомагання, передчасне стомлювання, головний біль, порушення функцій органів зору, які здійснювали несприятливий психофізіологічний вплив на самопочуття та працездатність операторів.

Сучасна професія користувача ВДТ належить до розумової праці, яка характеризується: високою напруженістю зорових функцій; одноманітною позою; великою кількістю стереотипних висококоординованих рухів, що виконуються лише м'язами кистей рук на фоні малої загальної рухової активності; значним нервовоемоційним компонентом, особливо в умовах дефіциту часу; роботою з великими масивами інформації, що викликає активізацію уваги та інших вищих психічних функцій. Крім того, при роботі з дисплеями на електронно-променевих трубках виникає вплив на користувача цілої низки факторів фізичної природи — електростатичні поля, радіочастотне та рентгенівське випромінювання тощо.

Діяльність професіоналів можна поділити на три групи:

1) Діяльність, яка пов'язана з виконанням нескладних багаторазово повторюваних операцій, що не вимагають великого розумового напруження. Наприклад, робота операторів комп'ютерного набору, працівників довідкових служб.

2) Діяльність, яка пов'язана із здійсненням логічних операцій, що постійно повторюються. Це робота інженера-економіста, інженера-проектувальника, оператора автоматизованого виробництва.

3) Діяльність, коли в процесі роботи необхідно приймати рішення за відсутності заздалегідь відомого алгоритму. Наприклад, робота інженера-програміста, диспетчерів руху залізничного транспорту, аеропортів тощо.

У користувачів, які інтенсивно використовують комп'ютер в умовах значних розумових напружень досить часто (40—70%) виникають психологічні та поведінкові порушення (нервозність, роздратування, тривога, нерішучість, замкнутість тощо). Серед користувачів ВДТ в США і Європі значного поширення набуло специфічне захворювання, яке отримало назву синдром комп'ютерного стресу (СКС). СКС супроводжується головним болем, запаленням очей, алергією, роздратованістю, млявістю і депресією. Інформаційне перевантаження користувачів ВДТ супроводжується низкою специфічних захворювань, які називають інформаційними. Першим симптомом їх є головний біль. Дослідження, проведені в США, Німеччині, Швейцарії та інших країнах, показали, що робота з обслуговування ВДТ супроводжується підвищеним напруженням зору, інтенсивністю і монотонністю праці, збільшенням статичних навантажень, нервово-психічним напруженням, впливом різного виду випромінювань та ін. Внаслідок цього серед операторів ВДТ, як зазначають фахівці Всесвітньої організації охорони здоров'я, частіше, ніж в інших групах працюючих, трапляються такі професійні захворювання, як передчасна стомлюваність, погіршення зору, м'язові і головні болі, психічні й нервові розлади, хвороби серцево-судинної системи, онкологічні захворювання та ін. Вважається, що стан організму операторів ВДТ визначається комплексним впливом факторів трудового процесу і середовища, значення яких є неоднаковим. На операторів з малим стажем роботи на ВДТ домінуючий вплив чинять фактори середовища, а на операторів зі стажем понад 5 років - фактори трудового процесу.

ВИСНОВКИ

1. Для забезпечення вірогідності та інформаційних скритності в інформаційних системах і базах даних використовуються спеціальні механізми, засновані на методах завадостійкого кодування та криптографічних методах. На сьогоднішній день у зв'язку з розвитком обчислювальної техніки висуваються нові вимоги щодо вірогідності нові вимоги щодо вірогідності ($P_o = 10^{-9} - 10^{-12}$), швидкодії (швидкість обробки даних 10 – 100 Мб/с), інформаційної скритності (кількість групових операцій $10^{-25} - 10^{-35}$) з цього існуючі механізми (симе-тричні і несиметричні криптосистеми) не повною мірою задовольняють висунутим вимогам.

2. Розроблюваний програмний продукт на основі ТКС Мак Еліса на АГК дозволяє забезпечити необхідні показники вірогідності, конфіденційності та інформаційної скритності. Проведені дослідження крипто стійкості та завадостійкості підтверджують теоретично розроблену криптосистему.

3. Практичне використання отриманих результатів дозволяє сполучити завадостійке кодування і маскування інформації під випадкову послідовність і забезпечити необхідні показники вірогідності та конфіденційності. Стійкість до негативного впливу противника (кількість операцій, необхідних для зламу системи) запропонованих теоретико − кодових систем на еліптичних кодах становить 10^{25} – 10^{35} групових операцій. Втрата вірогідності інформації (імовірність помилки) не перевершує 10^{-9} – 10^{-12}.

Програмний продукт наведений у вигляді додатку, в якому реалізується взаємодія з базою даних, розробленою в СУБД Microsoft Access 2007, через зручний інтерфейсу користувача. Додаток розроблений за допомогою мови програмування C#, що дозволило реалізувати функції для можливості роботи з даними в повному обсязі згідно з проектною документацією.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Edited by Serhii Yevseiev, Volodymir Ponomarenko, Oleksandr Laptiev, Oleksandr Milov. Synergy of building cybersecurity systems: monograph / S. Yevseiev, V. Ponomarenko, O. Laptiev, O. Milov and others. – Kharkiv: PC TECHNOLOGY CENTER, 2021. – 188 p.

2.Використання крипто-кодових конструкцій для забезпечення захисту даних в Ethernet мережах / О. С. Циганенко // Проблеми інформатизації : тези доп. 5-ї міжнар. наук.-техн. конф., 13–15 листопада 2017 р., м. Черкаси, м. Харків, м. Баку, м. Бельсько-Бяла / Черк. держ. технолог. ун-т [та ін.]. – Харків : Циганенко О.С., 2017. – С. 11.

3. Development of digital signature algorithm based on the Niederreiter crypto-code system / O. Tsyhanenko // Information Processing Systems, 2020. – Issue 3 (162) – С. 86 –94.

4. Practical implementation of the Niederreiter modified crypto-code system on truncated elliptic codes / S. Yevseiev, O. Tsyhanenko, S. Ivanchenko, V. Aleksiyev, D. Verheles, S. Volkov, R. Korolev, H. Kots, O. Milov, O. Shmatko // Eastern-European Journal of Enterprise Technologies, 2018. – № 6/4 (96). – С. 24–31.

5. Використання паралельних обчислень в реалізації алгоритму формування кодограми в НККС Нідеррайтера / О. С. Циганенко, С. П. Євсеєв // Проблеми і перспективи розвитку ІТ-індустрії : матеріали міжнарод. науково-практ. конф., 19–20 квітня 2018 р. : тези доповід. – Х.: ХНЕУ ім. Семена Кузнеця, 2018..

6. Р. В. Грищук, та Ю. Г. Даник, “Синергія інформаційних та кібернетичних дій”, Труди університету. НУОУ, № 6 (127), с. 132–143. 2014.

7. В. Л. Бурячок, Р. В. Грищук, та В. О. Хорошко, під заг. ред. проф. В. О. Хорошка, “Політика інформаційної безпеки”, ПВП «Задруга»,. 2014.

8. Ю. Г. Даник та ін., “Основи захисту інформації” навч. пос., Житомир : ЖВІ ДУТ, 2015.

9. S. Evseev, and B. Tomashevsky, “Two-factor authentication methods threats analysis”, Радіоелектроніка, інформатика, управління, Вип. 1(32), с. 52 – 60, 2015.

10. Р. В. Грищук, “Синтез систем інформаційної безпеки за заданими властивостями”, Вісник національного університету “Львівська політехніка”. Серія : Автоматика, вимірювання та керування : зб. наук. пр., ЛП, № 74, с. 271 – 276, 2012.

11. Р. В. Грищук, “Атаки на інформацію в інформаційно-комунікаційних системах”, Сучасна спеціальна техніка, №1(24), с.61 – 66. 2011.

12. Р. В. Грищук, і В. В. Охрімчук, “Постановка наукового завдання з розроблення шаблонів потенційно небезпечних кібератак”, Безпека інформації, Том 21, № 3, с. 276 – 282, 2015.

13. Ю. Г. Даник, Р. В. Грищук, “Синергетичні ефекти в площині інформаційного та кібернетичного протиборства”, Наук.-практ. конф. “Актуальні проблеми управління інформаційною безпекою держави”, Київ, 19 берез, 2015, с. 235 – 237.

14. Р. В. Грищук, В. В. Охрімчук, “Напрямки підвищення захищеності комп’ютерних систем та мереж від кібератак”, ІІ Міжнар. наук.-практ. конф. “Актуальні питання забезпечення кібербезпеки та захисту інформації” (Закарпатська область, Міжгірський район, село Верхнє Студене, 24-27 лют. 2016 р.). – К. : Видавництво Європейського університету, 2016 с. 60 – 61.

15. Захист інформації в комп’ютерних системах від несанкціонованого доступу. / За ред. С.Г. Лаптєва. – К., 2001. – 321 с.

16. Конституція України : Прийнята на п'ятій сесії Верховної Ради України 28 черв. 1996 р. Станом на 1 жовтня. 2011р. – Х. : Фактор, 2011. – 128 с. – (Бібліотека законодавства). – К-2 [34С(С2) / К65]

17. Циганенко О.С. Еліптична криптографія / Цыганенко А. С. // VIII всеукраїнська студентська науково – технічна конференція "Природничі та гуманітарні науки. Актуальні питання" 23-24 квітня 2015 р. – ТНТУ ім. Івана Пулюя, 2015 – 332 с.

18. B.-Z. Shen Which linear codes are algebraic-geometric? / B.-Z. Shen, G.J.M. van Wee, Ruud Pellikaan IEEE Trans. Inform. Theory, 1991.– vol. IT-37. – P. 583–602.

H. Niederreiter. Knapsack-Type Cryptosystems and Algebraic Coding Theory //

Probl. Control and Inform. Theoty. – 1986. – V.15. – P. 19–34.

19. Ian Blake Algebraic-Geometry Codes. / Ian Blake, Chris Heegard, Tom Hoholdt, Victor K. W. Wei. IEEE Trans. Info. Theory, 1998. – vol. IT–44. P. 2596 – 2618.

20. Johan P. Hansen. Codes on the Klein quartic, ideals, and decoding (Corresp.) // IEEE Trans. Info. Theory, 1987. – vol. IT–33. – P.923–925.

21. R.J. McEliece. A Public-Key Criptosystem Based on Algebraic Theory // DGN Progres Report 42–44, Jet Propulsi on Lab. Pasadena, CA. January – February, 1978. – P. 114–116.

22. Ruud Pellikaan. Asymptotically good sequences of curves and codes. // Proc. 34th Allerton Conf. on Communication, Control, and Computing, Urbana-Champaign, October 2–4, 1996. – 1996. – 276–285.

23. T. Høholdt Algebraic geometry codes / T. Høholdt, J.H. van Lint, Ruud Pellikaan. Handbook of Coding Theory. (V.S. Pless, W.C. Huffman and R.A. Brualdi, Eds.) Elsevier, Amsterdam, 1998. – vol 1. – P. 871–961.

24. T. R. N. Rao and K. H. Nam. Private-key algebraic-coded cryptosystem. Advances in Cryptology – CRYPTO 86, New York. – NY: Springer. – pp. 35-48.

25. Tsfasman M.A. Modular Gurves, Shimura Gurves and Goppa Codes, Bitter than Varshamov-Gilbert Bo-und / M.A. Tsfasman, S.G. Vladut, Zink Th. Math. Nachrichten, 1982. – vol. 109. – P. 21–28.

26. Worldwide Infrastructure Security Report. 2014. Arbor Networks, Inc Електронний ресурс: https://www.google.com.ua/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CB8QFjAA&url=http%3A%2F%2Fpages.arbornetworks.com%2Frs%2Farbor%2Fimages%2FWISR2014_EN2014.pdf&ei=DyR2VfznJOPgyQOghoN4&usg=AFQjCNGP0_ZTliItqCtofJ-cXfZT9QHRiQ&sig2=4hgA_vIye1idQyQgsTIZXg&bvm=bv.95039771,d.bGQ

27. S. Yevseiev, H. Kots, and Y. Liekariev, “Developing of multi-factor authentication method based on Niederreiter-McEliece modified crypto-code system”, Eastern-European Journal of Enterprise Technologies, 6/4(84), p. 11 – 23, 2016 *(Scopus)*

28. С. Євсеєв, С. Остапов, Х. Рзаєв, та В. Ніколаєнко, “Оцінка обміну даними в глобальних обчислювальних мережах на основі комплексного показника якості обслуговування мережі”, *Науковий журнал Радіоелектроніка, інформатика, управління*, № 1(40), с. 115 – 128, 2017. *(Web of Science)*

29. S. Yevseiev, O. Korol, and H. Kots, “Construction of hybrid security systems based on the crypto-code structures and flawed codes”, Eastern-European Journal of Enterprise Technologies, 4/9(88), p. 4 – 20, 2017. *(Scopus)*

30. S. Yevseiev, H. Kots, S. Minukhin, O. Korol, and A. Kholodkova, “The development of the method of multifactor authentication based on hybrid crypto-code constructions on defective codes”, Eastern-European Journal of Enterprise Technologies, 5/9(89), p. 19 – 35, 2017.

31. Models of socio-cyber-physical systems security: monograph / S. Yevseiev, Yu. Khokhlachova, S. Ostapov, O. Laptiev and others. – Kharkiv: PC TECHNOLOGY CENTER, 2023. – 168 p.

32. Євсеєв С.П. Кібербезпека: сучасні технології захисту. / Євсеєв С.П, Остапов С.Е., Король О.Г. // Навчальний посібник для студентів вищих навчальних закладів. Львів: “Новий Світ- 2000”, 2019. – 678. – Режим доступу: http://ns2000.com.ua/wp-content/uploads/2019/11/Kiberbezpeka-suchasni-tekhnolohii-zakhystu.pdf

33. Євсеєв С.П. Технології захисту інформації. Мультимедійне інтерактивне електронне видання комбінованого використання / уклад. Євсеєв С. П., Король О. Г., Остапов С. Е., Коц Г. П. – Х.: ХНЕУ ім. С. Кузнеця, 2016. – 1013 Мб. ISBN 978-966-676-624-6

34. Bonaventure O. Computer Networking: Principls, Protocols and Practice. – Louvain-la-Neuve: Universite catholique de Louvain (Belgium), 2019. – 272 p.

35. О. О. Кузнецов, С. П. Євсеєв, та С. В. Кавун Захист інформації та економічна безпека підприємства. Монографія. Харків, Україна: Вид. ХНЕУ, 2009

36. Edited by Serhii Yevseiev, Volodymir Ponomarenko, Oleksandr Laptiev, Oleksandr Milov. Synergy of building cybersecurity systems: monograph / S. Yevseiev,

V. Ponomarenko, O. Laptiev, O. Milov and others. – Kharkiv: PC TECHNOLOGY CENTER, 2021. – 188 p.

37. Cohen, G. Frey, R. Avanzi, C. Doche, T. Lange, K. Nguyen, F. Vercauteren. Handbook of elliptic and hyperelliptic curve cryptography// Kenneth H. Rosen Ed. 2006. 843 p.

38. Міжнародний стандарт ДСТУ ISO/IEC 27002:2015 «Інформаційні технології. Методи захисту. Звід практик щодо заходів інформаційної безпеки» [Електронний ресурс]. – Режим доступу: http://online.budstandart.com/ua/catalog/doc-page?id_doc=66911.

Міжнародний стандарт ДСТУ ISO/IEC 27003:2018 «Інформаційні технології. Методи захисту. Системи керування інформаційною безпекою. Настанова» [Електронний ресурс]. – Режим доступу: http://online.budstandart.com/ua/catalog/doc-page.html?id_doc=78517

40. Cyber Security Standards [Електронний ресурс]. – Режим доступу: https://www.itgovernance.co.uk/cybersecurity-standards.

41. A. Rukhin, J. Soto. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. NIST Special Publication 800-22, 2000.

42. Modeling of security systems for critical infrastructure facilities: monograph / S. Yevseiev, R. Hryshchuk, K. Molodetska, M. Nazarkevych and others. – Kharkiv: PC TECHNOLOGY CENTER, 2022. – 196 p.

ДОДАТКИ

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

XI НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

13-14 грудня 2023 року

ТЕРНОПІЛЬ
2023

УДК 004.056

Л. Сава
Тернопільський національний технічний університет імені Івана Пулюя

РОЗРОБЛЕННЯ АЛГОРИТМІВ НЕСИМЕТРИЧНОГО ШИФРУВАННЯ ДЛЯ МОБІЛЬНИХ ЗАСОБІВ ЗВ'ЯЗКУ

L. Sava

DEVELOPMENT OF ASYMMETRIC ENCRYPTION ALGORITHMS FOR MOBILE COMMUNICATIONS

Розвиток інтернет-технологій разом із мобільними, комп'ютерними технологіями сформували смарт-технології, які дозволяють формувати як кіберфізичні, а й соціокіберфізичні системи. Основою смарттехнологій є комплексування стандартів бездротових каналів із мобільними та комп'ютерними протоколами. Крім цього, технології 4G/5G комплексують із різними веб-платформами з урахуванням цифровізації послуг у кіберпросторі. Для забезпечення послуг безпеки автентичності та цілісності, як правило (шифруються понад 80% трафіку Хром), використовується мережевий протокол SSL/TLS, заснований на гібридизації симетричних алгоритмів шифрування з алгоритмами гешування (режим AEAD), що забезпечує досконалу пряму секретність. Однак, цей протокол піддається не тільки атакам "Зустріч на середині", POODLE, BEAST, CRIME, BREACH, а з появою повномасштабного квантового комп'ютера може бути зламаний.

Створення сучасних синтезованих мереж ґрунтується на гібридизації технологій бездротових мобільних та соціо-кіберфізичних систем на основі Інтернету речей. Класичні комп'ютерні системи та технології інтегрують елементи Інтернет речей та формують принципово нові напрямки розвитку IT-індустрії –smart-технології, які поєднують досягнення мобільних, бездротових та соціо-кіберфізичних систем.

Однак стрімке розширення mesh-, сенсорних мереж з використанням стандартів бездротових каналів: мобільних технологій LTE, IEEE802.16, IEEE802.16e, IEEE802.15.4, IEEE802.11, Bluetooth не забезпечує безпеку інформаційних потоків. У гонитві за надшвидкістю в таких каналах не надаються послуги конфіденційності та цілісності. Протокол Diameter забезпечує взаємодію між клієнтами з метою аутентифікації, авторизації та обліку різних сервісів, проте має суттєві недоліки до сучасних кібератаків.

Для забезпечення безпеки в кіберфізичних системах на основі інтернет-речей використовується стандарт KNX (ISO/IEC 14543) на основі використання VPN-каналів (шифрування AES-128, -256). Проте, всі наявні механізми безпеки не забезпечать необхідний рівень безпеки в постквантовий період (поява повномасштабного квантового комп'ютера). Фахівці NIST США висувають сумніви у стійкості сучасних симетричних та несиметричних криптосистем (включаючи алгоритми та на еліптичних кривих). У таких умовах постквантові криптоалгоритми на основі синтезу теорій завадостійкого кодування та захисту інформації – крипто-кодові конструкції можуть розглядатися як альтернативний механізм забезпечення безпеки.

Крім цього, виникає необхідність розгляду концепції двох контурів безпеки (внутрішнього – безпосередньо інфраструктура мережі та зовнішнього – хмарні платформи – сервера управління кіберфізичними системами) в умовах інтеграції мереж та хмарних технологій.

ДОДАТОК Б

Лістинг програми

Класс MacEllis.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace MacEllisLibrary
{
   public class MacElis
   {

      public static int[,] TransponMatrixEllept(int[,] A)
      {
         int[,] B = new int[7, 4];

         for (int j = 0; j < 4; j++)
         {
            for (int i = 0; i < 7; i++)
            {
               B[i, j] = A[j, i];

            }
         }

         return B;
      }

      public string text;

      public int[,] d = { { 0, 0, 0, 0, 0, 0, 0 } };
      public int[,] p = { { 0, 0, 0, 0, 0, 0, 0 } };
      public int[,] x = { { 0, 0, 0 }, { 0, 0, 0 }, { 0, 0, 0 } };


      public int[,] i = { { 0, 0, 0 } };
      public int[,] e = { { 0, 0, 0, 0, 0, 0, 0 } };
```

```
public int[,] d1 = { { 0, 0, 0, 0, 0, 0, 0 } };
public int[,] p1 = { { 0, 0, 0, 0, 0, 0, 0 } };
public static int[,] x1 = { { 0, 0, 0 }, { 0, 0, 0 }, { 0, 0, 0 } };

public int[,] D = { { 0, 0, 0, 0, 0, 0, 0 } };
public int[,] P = { { 0, 0, 0, 0, 0, 0, 0 } };
public int[,] X = { { 0, 0, 0 }, { 0, 0, 0 }, { 0, 0, 0 } };

public int[,] D1 = { { 0, 0, 0, 0, 0, 0, 0 } };
public int[,] P1 = { { 0, 0, 0, 0, 0, 0, 0 } };
public int[,] X1 = { { 0, 0, 0 }, { 0, 0, 0 }, { 0, 0, 0 } };
public int[,] e1 = { { 0, 0, 0, 0, 0, 0, 0 } };

public int[,] gxpd = { { 0, 0, 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0, 0, 0 } };
public int[,] cdp = { { 0, 0, 0, 0, 0, 0, 0 } };
public int[,] iGx = { { 0, 0, 0, 0, 0, 0, 0 } };
public static int[,] Cx = { { 0, 0, 0, 0, 0, 0, 0 } };
public static int[,] g = { { 1, 0, 0, 7, 6, 6, 3 }, { 0, 1, 0, 2, 3, 5, 1 }, { 0, 0, 1, 7, 7, 4, 2 } };


// public static int[,] g = { { 1, 0, 0, 7, 6, 6, 3 }, { 0, 1, 0, 2, 3, 5, 1 }, { 0, 0, 1, 7, 7, 4, 2 }};

 // public static int[,] h={{1,1,1,1,1,1,1},{1,2,3,4,5,6,7},{1,3,5,7,2,4,6},{1,4,7,3,6,2,5}};
   public static int[,] ht = { {1,1,1,1 }, {1,2,3,4 }, {1,3,5,7 }, {1,4,7,3 }, {1,5,2,6 }, {1,6,4,2 }, {1,7,6,5 } };



public static int[,] h = { { 1, 1, 1, 1, 1, 1, 1 }, { 1, 2, 3, 4, 5, 6, 7 }, { 1, 3, 5, 7, 2, 4, 6 }, { 1, 4, 7, 3, 6, 2, 5 } };
//public static int[,] h = { { 0, 0, 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0, 0, 0 } };


// public static int[,] ht = TransponMatrixEllept(h);
//   public static int[,] g = FindCx(h);

public static int[,] MatrixMult(int[,] a, int[,] b, int n, int m, int k)
{
    int alphaA, alphaB, alphaSum = 0;
    string rez = "";
    int res = 0;

    int[,] c = new int[n, k];
```

```
    string el;

    for (int z = 0; z < k; z++)
    {
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < m; j++)
            {
                alphaA = GetAlpha(a[i, j]);
                alphaB = GetAlpha(b[j, z]);
                alphaSum = GetAlphaSum(alphaA, alphaB);

                if (j == 0)
                {

                    rez = GetBinary(alphaSum);



                }
                else
                {
                    el = GetBinary(alphaSum);
                    res = GetResult(rez, el);
                    rez = GetBinary(GetAlpha(res));

                }

            }
            c[i, z] = res;

        }
    }
    return c;
}
public static int[,] SummVec(int[,] vec1, int[,] vec2)
{
    int summ = 0;
    int[,] vec_summ = { { 0, 0, 0, 0, 0, 0, 0 } };
    for (int i = 0; i < 7; i++)
    {
        summ = GetResult(GetBinary(GetAlpha(vec1[0, i])), GetBinary(GetAlpha(vec2[0, i])));
```

```
 vec_summ[0, i] = summ;
            }

            return vec_summ;
        }
        public static int[] GaussDesMatrix(int k1, int k2, int k3, int s, int kp1, int kp2, int kp3, int sp2, int kpp1, int kpp2, int kpp3, int spp2)
        {
            int[] stroka = { 0, 0, 0 };
            int koef1 = GetAlpha(k1);
            int koef2 = GetAlpha(k2);
            int koef3 = GetAlpha(k3);

            int koefp1 = GetAlpha(kp1);
            int koefp2 = GetAlpha(kp2);
            int koefp3 = GetAlpha(kp3);

            int koefp12 = GetAlpha(kpp1);
            int koefp22 = GetAlpha(kpp2);
            int koefp32 = GetAlpha(kpp3);

            int a1 = 0;
            int a0 = 0;
            int a2 = 0;

            int a1A = 0;
            int a2A = 0;
            int a0A = 0;

            int a1AA = 9;
            int a2AA = 9;
            int a0AA = 9;

            int summ = 0;
            int summ_prom = 0;

            int summp1 = 0;
            int summp1_prom = 0;



            int summp2 = 0;
            int summp1_prom2 = 0;
```

```
int operand1, operand2, operand3 = 0;
string stroper1;
string stroper2;
string stroper3;

int[] A1 = { -1, 0, 1, 2, 3, 4, 5, 6 };
int[] A0 = { -1, 0, 1, 2, 3, 4, 5, 6 };
int[] A2 = { -1, 0, 1, 2, 3, 4, 5, 6 };

for (int i = 0; i < A0.Length; i++)
{
   for (int j = 0; j < A1.Length; j++)
   {
      for (int k = 0; k < A2.Length; k++)
      {


         operand1 = GetAlphaSum(koef1, A0[i]);
         operand2 = GetAlphaSum(koef2, A1[j]);
         operand3 = GetAlphaSum(koef3, A2[k]);

         stroper1 = GetBinaryGauss(operand1);
         stroper2 = GetBinaryGauss(operand2);
         stroper3 = GetBinaryGauss(operand3);

         summ_prom = GetResultGauss(stroper1, stroper2);
         summ = GetResultGauss(stroper3, GetBinaryGauss(GetAlpha(summ_prom)));

         if (summ == s)
         {

            a0 = A0[i];
            a1 = A1[j];
            a2 = A2[k];




            operand1 = GetAlphaSum(koefp1, a0);
            operand2 = GetAlphaSum(koefp2, a1);
            operand3 = GetAlphaSum(koefp3, a2);
```

```
                stroper1 = GetBinaryGauss(operand1);
                stroper2 = GetBinaryGauss(operand2);
                stroper3 = GetBinaryGauss(operand3);

                summp1_prom = GetResultGauss(stroper1, stroper2);
                summp1 = GetResultGauss(stroper3, GetBinaryGauss(GetAlpha(summp1_prom)));
                if (summp1 == sp2)
                {

                   a0A = A0[i];
                   a1A = A1[j];
                   a2A = A2[k];



                   operand1 = GetAlphaSum(koefp12, a0A);
                   operand2 = GetAlphaSum(koefp22, a1A);
                   operand3 = GetAlphaSum(koefp32, a2A);



                   stroper1 = GetBinaryGauss(operand1);
                   stroper2 = GetBinaryGauss(operand2);
                   stroper3 = GetBinaryGauss(operand3);

                   summp1_prom2 = GetResultGauss(stroper1, stroper2);
                   summp2 = GetResultGauss(stroper3, GetBinaryGauss(GetAlpha(summp1_prom2)));



                   if (summp2 == spp2)
                   {
                      a0AA = a0A + 1;
                      a1AA = a1A + 1;
                      a2AA = a2A + 1;
                   }
                }

             }
          }
       }

}
```

```
        stroka[0] = a0AA;
        stroka[1] = a1AA;
        stroka[2] = a2AA;

        return stroka;
    }

    public static int[,] MatrixCx()
    {
        int[,] Cx = { { 1, 0, 0, 0, 0, 0, 0 }, { 0, 1, 0, 0, 0, 0, 0 }, { 0, 0, 1, 0, 0, 0, 0 } };

        Random r = new Random();
        for (int i = 0; i < 3; i++)
        {
            for (int j = 4; j < 7; j++)
            {
                Cx[i, j] = r.Next(0, 7);
            }
        }

        return Cx;
    }
    public static int[,] FindCx(int[,] H)
    {

        int[,] Cx = { { 1, 0, 0, 0, 0, 0, 0 }, { 0, 1, 0, 0, 0, 0, 0 }, { 0, 0, 1, 0, 0, 0, 0 } };
        int[] cx = new int[4];
        int m = 0;

        cx = GaussDesMatrix5(H[0, 3], H[0, 4], H[0, 5], H[0, 6], H[0, 0], H[1, 3], H[1, 4], H[1, 5], H[1, 6], H[0, 1], H[2, 3],
H[2, 4], H[2, 5], H[2, 6], H[0, 2], H[3, 3], H[3, 4], H[3, 5], H[3, 6], H[0, 3]);

        for (int i = 3; i < 7; i++)

        {
            Cx[0, i] = cx[m];
            m++;
        }

        m = 0;
```

```
        cx = GaussDesMatrix5(H[0, 3], H[0, 4], H[0, 5], H[0, 6], H[0, 1], H[1, 3], H[1, 4], H[1, 5], H[1, 6], H[1, 1], H[2, 3], H[2, 4], H[2, 5], H[2, 6], H[2, 1], H[3, 3], H[3, 4], H[3, 5], H[3, 6], H[3, 1]);

        for (int i = 3; i < 7; i++)
        {
           Cx[1, i] = cx[m];
           m++;
        }
        m = 0;

        cx = GaussDesMatrix5(H[0, 3], H[0, 4], H[0, 5], H[0, 6], H[0, 2], H[1, 3], H[1, 4], H[1, 5], H[1, 6], H[1, 2], H[2, 3], H[2, 4], H[2, 5], H[2, 6], H[2, 2], H[3, 3], H[3, 4], H[3, 5], H[3, 6], H[3, 2]);

        for (int i = 3; i < 7; i++)
        {
           Cx[2, i] = cx[m];
           m++;
        }
     return Cx;
    }
    public static int[] GaussDesMatrix5(int k11, int k21, int k31, int k41, int s1, int k12, int k22, int k32, int k42, int s2, int k13, int k23, int k33, int k43, int s3, int k14, int k24, int k34, int k44, int s4)
    {
       int[] Cx = new int[4];
       int koef11 = GetAlpha(k11);
       int koef21 = GetAlpha(k21);
       int koef31 = GetAlpha(k31);
       int koef41 = GetAlpha(k41);
       int koef12 = GetAlpha(k12);
       int koef22 = GetAlpha(k22);
       int koef32 = GetAlpha(k32);
       int koef42 = GetAlpha(k42);

       int koef13 = GetAlpha(k13);
       int koef23 = GetAlpha(k23);
       int koef33 = GetAlpha(k33);
      int koef43 = GetAlpha(k43);
       int koef14 = GetAlpha(k14);
       int koef24 = GetAlpha(k24);
       int koef34 = GetAlpha(k34);
       int koef44 = GetAlpha(k44);
       int a11 = 0;
```

```
int a01 = 0;
int a21 = 0;
int a31 = 0;
int a12 = 0;
int a02 = 0;
int a22 = 0;
int a32 = 0;
int a13 = 0;
int a03 = 0;
int a23 = 0;
int a33 = 0;
int a14 = 9;
int a04 = 9;
int a24 = 9;
int a34 = 9;
int summ = 0;
int summ_prom1 = 0;
int summ_prom2 = 0;
int summ_prom3 = 0;
int summ_prom4 = 0;
int summ_prom5 = 0;
int summ_prom6 = 0;
int operand1, operand2, operand3, operand4, operand5, operand6, operand7 = 0;
string stroper1;
string stroper2;
string stroper3;
string stroper4;
string stroper5;
string stroper6;
string stroper7;

int[] A0 = { -1, 0, 1, 2, 3, 4, 5, 6 };
int[] A1 = { -1, 0, 1, 2, 3, 4, 5, 6 };
int[] A2 = { -1, 0, 1, 2, 3, 4, 5, 6 };
int[] A3 = { -1, 0, 1, 2, 3, 4, 5, 6 };



for (int i1 = 0; i1 < A0.Length; i1++)
{
    for (int i2 = 0; i2 < A1.Length; i2++)
    {
```

```
for (int i3 = 0; i3 < A2.Length; i3++)
{
   for (int i4 = 0; i4 < A3.Length; i4++)
   {



      operand1 = GetAlphaSum(koef11, A0[i1]);
      operand2 = GetAlphaSum(koef21, A1[i2]);
      operand3 = GetAlphaSum(koef31, A2[i3]);
      operand4 = GetAlphaSum(koef41, A3[i4]);



      stroper1 = GetBinaryGauss(operand1);
      stroper2 = GetBinaryGauss(operand2);
      stroper3 = GetBinaryGauss(operand3);
      stroper4 = GetBinaryGauss(operand4);



      summ_prom1 = GetResultGauss(stroper1, stroper2);
      summ_prom2 = GetResultGauss(stroper3, GetBinaryGauss(GetAlpha(summ_prom1)));
      summ_prom3 = GetResultGauss(stroper4, GetBinaryGauss(GetAlpha(summ_prom2)));

      if (summ_prom3 == s1)
      {
         a01 = A0[i1];
         a11 = A1[i2];
         a21 = A2[i3];
         a31 = A3[i4];



         operand1 = GetAlphaSum(koef12, a01);
         operand2 = GetAlphaSum(koef22, a11);
         operand3 = GetAlphaSum(koef32, a21);
         operand4 = GetAlphaSum(koef42, a31);

         stroper1 = GetBinaryGauss(operand1);
         stroper2 = GetBinaryGauss(operand2);
         stroper3 = GetBinaryGauss(operand3);
```

```
stroper4 = GetBinaryGauss(operand4);



summ_prom1 = GetResultGauss(stroper1, stroper2);
summ_prom2 = GetResultGauss(stroper3, GetBinaryGauss(GetAlpha(summ_prom1)));
summ_prom3 = GetResultGauss(stroper4, GetBinaryGauss(GetAlpha(summ_prom2)));



if (summ_prom3 == s2)
{

   a02 = A0[i1];
   a12 = A1[i2];
   a22 = A2[i3];
   a32 = A3[i4];



   operand1 = GetAlphaSum(koef13, a02);
   operand2 = GetAlphaSum(koef23, a12);


  operand3 = GetAlphaSum(koef33, a22);
   operand4 = GetAlphaSum(koef43, a32);



   stroper1 = GetBinaryGauss(operand1);
   stroper2 = GetBinaryGauss(operand2);
   stroper3 = GetBinaryGauss(operand3);
   stroper4 = GetBinaryGauss(operand4);


   summ_prom1 = GetResultGauss(stroper1, stroper2);
   summ_prom2 = GetResultGauss(stroper3, GetBinaryGauss(GetAlpha(summ_prom1)));
   summ_prom3 = GetResultGauss(stroper4, GetBinaryGauss(GetAlpha(summ_prom2)));



   if (summ_prom3 == s3)
```

```
            {

                a03 = A0[i1];

                a13 = A1[i2];
                a23 = A2[i3];
                a33 = A3[i4];



                operand1 = GetAlphaSum(koef14, a03);
                operand2 = GetAlphaSum(koef24, a13);
                operand3 = GetAlphaSum(koef34, a23);
                operand4 = GetAlphaSum(koef44, a33);


                stroper1 = GetBinaryGauss(operand1);
                stroper2 = GetBinaryGauss(operand2);
                stroper3 = GetBinaryGauss(operand3);
                stroper4 = GetBinaryGauss(operand4);


                summ_prom1 = GetResultGauss(stroper1, stroper2);
                summ_prom2 = GetResultGauss(stroper3, GetBinaryGauss(GetAlpha(summ_prom1)));
                summ_prom3 = GetResultGauss(stroper4, GetBinaryGauss(GetAlpha(summ_prom2)));



                if (summ_prom3 == s4)
                {

                    a04 = a03 + 1;
                    a14 = a13 + 1;
                    a24 = a23 + 1;
                    a34 = a33 + 1;

                }
            }
        }
    }
}
```

```
        }



    }


    Cx[0] = a04;
    Cx[1] = a14;
    Cx[2] = a24;
    Cx[3] = a34;

    return Cx;

}
public static bool SravMatrix(int[,] a)
{
    bool result = true;


    int[,] eden = { { 1, 0, 0 }, { 0, 1, 0 }, { 0, 0, 1 } };

    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
       {
          if (eden[i, j] != a[i, j])
          {
             result = false;
          }
       }


    }
    return result;
}
public static int[] ChensProcedure(int a1, int a0)
{

    int summ1;
    int summStep;
    int summOper1;
```

```
    int summAll = 0;
    string chl = GetBinary(GetAlpha(a0));
    int[] vektor = { 0, 0, 0, 0, 0, 0, 0 };

    for (int i = 1; i < 8; i++)
    {
        summStep = GetAlphaSum(GetAlpha(i), GetAlpha(i));
        summOper1 = GetAlphaSum(GetAlpha(a1), GetAlpha(i));

        summ1 = GetResult(GetBinary(summStep), GetBinary(summOper1));
        summAll = GetResult(GetBinary(GetAlpha(summ1)), chl);

        if (summAll == 0)
        {

            vektor[i - 1] = 1;



        }
        else
        {

        }


    }

    return vektor;
}

public static int[,] GaussDes(int k1, int k2, int s, int kp1, int kp2, int sp2)
{
    int[,] reshenie = { { 0, 0 } };
    int koef1 = GetAlpha(k1);
    int koef2 = GetAlpha(k2);

    int koefp1 = GetAlpha(kp1);
    int koefp2 = GetAlpha(kp2);

    int a1 = 0;
    int a0 = 0;
    int a1A = 0;
```

```
int a0A = 0;
int summ = 0;
int summp1 = 0;
int operand1, operand2 = 0;
string stroper1;
string stroper2;

int[] A1 = { 0, 1, 2, 3, 4, 5, 6 };
int[] A0 = { 0, 1, 2, 3, 4, 5, 6 };

for (int i = 0; i < A0.Length; i++)
{
   for (int j = 0; j < A1.Length; j++)
   {

      operand1 = GetAlphaSum(koef1, A0[i]);
      operand2 = GetAlphaSum(koef2, A1[j]);

      stroper1 = GetBinaryGauss(operand1);
      stroper2 = GetBinaryGauss(operand2);

      summ = GetResultGauss(stroper1, stroper2);

      if (summ == s)
      {

         a0 = A0[i] + 1;
         a1 = A1[j] + 1;

         int alphaA0 = GetAlpha(a0);
         int alphaA1 = GetAlpha(a1);

         operand1 = GetAlphaSum(koefp1, alphaA0);
         operand2 = GetAlphaSum(koefp2, alphaA1);

         stroper1 = GetBinaryGauss(operand1);
         stroper2 = GetBinaryGauss(operand2);

         summp1 = GetResultGauss(stroper1, stroper2);
         if (summp1 == sp2)
         {
            a0A = a0;
```

```
                    a1A = a1;
                }
            }
        }

        reshenie[0, 0] = a0A;
        reshenie[0, 1] = a1A;



    }



    return reshenie;
}
public static int[,] CreateMatP(int[,] a)
{
    int[,] P = new int[a.Length, a.Length];

    for (int j = 0; j < a.Length; j++)
    {
        for (int i = 0; i < a.Length; i++)
        {
            P[i, j] = 0;
}
        }

    for (int i = 0; i < a.Length; i++)
    {
        P[i, a[0, i] - 1] = 1;
    }

    return P;
}
public static int[,] CreateMatD(int[,] a)
{

    int[,] D = new int[a.Length, a.Length];
    int[,] d = new int[a.Length, a.Length];
    for (int j = 0; j < a.Length; j++)
    {
        for (int i = 0; i < a.Length; i++)
```

```
        {
            D[i, j] = 0;
        }
    }

    for (int i = 0; i < a.Length; i++)
    {
        D[i, i] = a[0, i];
    }


    return D;
}
public static int[,] CreateMatP1(int[,] a)
{
    int[,] P = new int[a.Length, a.Length];

    for (int j = 0; j < a.Length; j++)
    {
        for (int i = 0; i < a.Length; i++)
        {
            P[i, j] = 0;
        }
    }

    for (int i = 0; i < a.Length; i++)
    {
        P[a[0, i] - 1, i] = 1;
    }

    return P;
}

public static int[,] CreateMatD1(int[,] a)
{



    int[,] D = new int[a.Length, a.Length];


    for (int j = 0; j < a.Length; j++)
```

```
{
   for (int i = 0; i < a.Length; i++)

  {
      D[i, j] = 0;
   }
}


for (int i = 0; i < a.Length; i++)
{
   if (a[0, i] == 1)
   {
      D[i, i] = 1;
   }
   if (a[0, i] == 2)
   {
      D[i, i] = 7;
   }
   if (a[0, i] == 3)
   {
      D[i, i] = 6;
   }
   if (a[0, i] == 4)
   {
      D[i, i] = 5;
   }
   if (a[0, i] == 5)
   {
      D[i, i] = 4;
   }
   if (a[0, i] == 6)
   {
      D[i, i] = 3;
   }
   if (a[0, i] == 7)
   {
      D[i, i] = 2;
   }

}
```

```
        return D;
    }
    public static int GetAlpha(int a)
    {
        int alpha = a - 1;
        return alpha;

    }
    public static int GetChislo(string text)
    {
        int ch = 0;
        string[] binary = { "000", "001", "010", "100", "011", "110", "111", "101" };

        for (int i = 0; i < binary.Length; i++)
        {
            if (text == binary[i])
            {
                ch = i;
            }
        }

        return ch;
    }

public static string GetBinary3(int n)
    {
        string res = "";
        string[] binary = { "000", "001", "010", "100", "011", "110", "111", "101" };

        for (int i = 0; i < binary.Length; i++)
        {
            if (n == i)
            {
                res = binary[i];
            }

        }

        return res;
    }
```

```
public static int GetAlphaSum(int a, int b)
{
    int result;
    if (a < 0 && b > 0)

    {
        result = -1;
    }
    if (a > 0 && b < 0)
    {
        result = -1;
    }
    if (a < 0 || b < 0)
    {
        result = -1;
    }
    else
    {
        result = a + b;
    }
    if (result > 6)
    {
        result = result - 7;
    }
    return result;
}
public static string GetBinary(int a)
{
    string binary = "";
    string[] poleByte = { "0001", "0010", "0100", "0011", "0110", "0111", "0101" };
    if (a < 0)
    {
        binary = "0000";
    }
    else
    {
        for (int i = 0; i < poleByte.Length; i++)
        {
            if (a == i)
            {
                binary = poleByte[i];
            }
```

```
}
        }
        return binary;
    }
    public static string GetBinaryGauss(int a)
    {
        string binary = "";
        string[] poleByte = { "0001", "0010", "0100", "0011", "0110", "0111", "0101" };
        if (a < 0)
        {
            binary = "0000";
        }
        else
        {
            for (int i = 0; i < poleByte.Length; i++)
            {
                if (a == i)
                {
                    binary = poleByte[i];
                }

            }
        }
        return binary;
    }

    public static int GetResult(string operand1, string operand2)
    {
        int result = 0;
        string[] poleByte = { "0001", "0010", "0100", "0011", "0110", "0111", "0101" };
        string rez = "";

        for (int i = 0; i < 4; i++)
        {
            if (operand1[i] == '0' && operand2[i] == '0')
                rez += "0";
            if (operand1[i] == '0' && operand2[i] == '1')
                rez += "1";
            if (operand1[i] == '1' && operand2[i] == '0')
```

```
            rez += "1";
        if (operand1[i] == '1' && operand2[i] == '1')
            rez += "0";
    }

    for (int i = 0; i < poleByte.Length; i++)
    {
        if (rez == poleByte[i])
        {
            result = i + 1;
        }
    }
    return result;
}

public static int GetResultGauss(string operand1, string operand2)
{
    int result = 0;
    string[] poleByte = { "0001", "0010", "0100", "0011", "0110", "0111", "0101" };
    string rez = "";


for (int i = 0; i < 4; i++)
    {
        if (operand1[i] == '0' && operand2[i] == '0')
            rez += "0";
        if (operand1[i] == '0' && operand2[i] == '1')
            rez += "1";
        if (operand1[i] == '1' && operand2[i] == '0')
            rez += "1";
        if (operand1[i] == '1' && operand2[i] == '1')
            rez += "0";
    }

    for (int i = 0; i < poleByte.Length; i++)
    {
        if (rez == poleByte[i])
        {
            result = i + 1;

        }
```

```
    }
    if (rez == "0000")
    {
        return 0;
    }
    else
    {
        return result;
    }
}

public struct PrivateKey
{
    public int[,] d;
    public int[,] p;
    public int[,] D;
    public int[,] P;
    public int[,] D1;
    public int[,] P1;
    public int[,] gxpd;
    public string text;
}

public static string CodeText(string text, ref PrivateKey mac_comp)
{
    string codetext = null;

    mac_comp.d = new int[1, 7];
    mac_comp.p = new int[1, 7];
    mac_comp.D = new int[7, 7];
    mac_comp.P = new int[7, 7];

    mac_comp.gxpd = new int[3, 7];


    Random rand = new Random();

   int[,] p = new int[1, 7];

   int[,] d = new int[1, 7];
    //int[,] p = {{1,3,2,7,5,4,6}};
    //int[,] d = { { 1, 2, 3, 1, 2, 3, 2 } };
```

```
        int[,] e = { { 0, 0, 0, 0, 0, 0, 0 } };

int pos2 = 0;
        int pos = rand.Next(0, 6);

        do
        {
            pos2 = rand.Next(0, 6);

        } while (pos== pos2);

        e[0, pos] = rand.Next(1, 7);
        e[0, pos2] = rand.Next(1, 7);

        int[,] h = { { 1, 1, 1, 1, 1, 1, 1 }, { 1, 2, 3, 4, 5, 6, 7 }, { 1, 3, 5, 7, 2, 4, 6 }, { 1, 4, 7, 3, 6, 2, 5 } };
        int[,] x = { { 2, 0, 1 }, { 0, 5, 3 }, { 4, 0, 1 } };

        for (int i = 0; i < d.Length; i++)
        {
            d[0, i] = rand.Next(1, 7);
        }

        List<int> p_vec = new List<int>();

        while (p_vec.Count < 7)
        {
            int elem = 0;

            elem = rand.Next(1, 8);

            if (!p_vec.Contains(elem))
            {
                p_vec.Add(elem);
            }
        }


        for (int i = 0; i < 7; i++)
        {
            p[0, i] = p_vec[i];
        }
```

```
for (int z = 0; z < text.Length; z++)
{
   string ch = text[z].ToString();

   byte[] _byte = new byte[9];

   ASCIIEncoding encode = new ASCIIEncoding();

   _byte = Encoding.Default.GetBytes(ch);


   string str = Convert.ToString(_byte[0], 2);

   if (str.Length != 9)
   {
      int num = 9 - str.Length;
      int max = str.Length;
      string str_null = "0";

      for (int i = max; i < 9; i++)
      {

         str_null += str;
         str = str_null;
         str_null = "0";
      }
   }
   mac_comp.text = str;
   int[,] ix = new int[1, 3];

   string i1;
   string i2;
   string i3;


   i1 = mac_comp.text[0].ToString();
   i1 += mac_comp.text[1];
   i1 += mac_comp.text[2];

   i2 = mac_comp.text[3].ToString();
   i2 += mac_comp.text[4];
   i2 += mac_comp.text[5];
```

```
    i3 = mac_comp.text[6].ToString();
    i3 += mac_comp.text[7];
    i3 += mac_comp.text[8];


    ix[0, 0] = MacElis.GetChislo(i1);
    ix[0, 1] = MacElis.GetChislo(i2);
    ix[0, 2] = MacElis.GetChislo(i3);

    mac_comp.d = d;
    mac_comp.p = p;

    mac_comp.D = MacElis.CreateMatD(mac_comp.d);

    mac_comp.P = MacElis.CreateMatP(mac_comp.p);



    int[,] gx = { { 0, 0, 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0, 0, 0 } };
    int[,] gxp = { { 0, 0, 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0, 0, 0 } };
    int[,] gxpd = { { 0, 0, 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0, 0, 0 } };

    gx = MacElis.MatrixMult(x, MacElis.g, 3, 3, 7);
    gxp = MacElis.MatrixMult(gx, mac_comp.P, 3, 7, 7);
    gxpd = MacElis.MatrixMult(gxp, mac_comp.D, 3, 7, 7);

    mac_comp.gxpd = gxpd;

    int[,] iGx = MacElis.MatrixMult(ix, mac_comp.gxpd, 1, 3, 7);



    int[,] Cx = MacElis.SummVec(iGx, e);



    for (int i = 0; i < 7; i++)
    {
        codetext += Cx[0, i].ToString();
    }
}
```

```
    mac_comp.text = "";
    return codetext;
}

public static string DecodeText(string text, PrivateKey mac_comp)
{
    string decodetext = null;


    mac_comp.D1 = new int[7, 7];
    mac_comp.P1 = new int[7, 7];

    int[,] x1 = { { 1, 0, 1 }, { 2, 4, 7 }, { 4, 0, 2 } };

    string substr = "";

    for (int l = 0; l < text.Length; l++)
    {
        substr += text[l];

        if (substr.Length == 7)
        {


            int[,] Cx = new int[1, 7];

            for (int j = 0; j < substr.Length; j++)
            {
                Cx[0, j] = int.Parse(substr[j].ToString());
            }

            mac_comp.D1 = MacElis.CreateMatD1(mac_comp.d);

            mac_comp.P1 = MacElis.CreateMatP1(mac_comp.p);




            int[,] cd = { { 0, 0, 0, 0, 0, 0, 0 } };
            int[,] cdp = { { 0, 0, 0, 0, 0, 0, 0 } };
```

```
cd = MacElis.MatrixMult(Cx, mac_comp.D1, 1, 7, 7);
cdp = MacElis.MatrixMult(cd, mac_comp.P1, 1, 7, 7);


int[,] s = { { 0, 0, 0, 0 } };
int[] pos = { 0, 0 };
int[] vec_e = { 0, 0, 0, 0, 0, 0, 0 };


s = MacElis.MatrixMult(cdp, MacElis.ht, 1, 7, 4);

int[,] a0a1 = MacElis.GaussDes(s[0, 0], s[0, 1], s[0, 2], s[0, 1], s[0, 2], s[0, 3]);

vec_e = MacElis.ChensProcedure(a0a1[0, 1], a0a1[0, 0]);


int[] position = { 0, 0 };
int k = 0;

for (int i = 0; i < vec_e.Length; i++)
{
   if (vec_e[i] == 1)
   {
      position[k] = i;
      k++;
   }
}


int[,] e0e1 = MacElis.GaussDes(MacElis.h[0, position[0]], MacElis.h[0, position[1]], s[0, 0], MacElis.h[1, position[0]], MacElis.h[1, position[1]], s[0, 1]);
k = 0;
for (int i = 0; i < vec_e.Length; i++)
{
   if (vec_e[i] == 1)
   {
      vec_e[i] = e0e1[0, k];
      k++;
   }
}

int[,] e1 = { { 0, 0, 0, 0, 0, 0, 0 } };
```

```
for (int i = 0; i < 1; i++)
{
    for (int j = 0; j < 7; j++)
    {
        e1[i, j] = vec_e[j];
    }

}
int[,] i1 = { { 0, 0, 0, 0, 0, 0, 0 } };
int[,] i1_3 = { { 0, 0, 0 } };
int[,] ik = { { 0, 0, 0 } };

i1 = MacElis.SummVec(e1, cdp);

for (int i = 0; i < 1; i++)
{
    for (int j = 0; j < 3; j++)
    {
        i1_3[i, j] = i1[i, j];
    }
}
ik = MacElis.MatrixMult(i1_3, x1, 1, 3, 3);
string ikon = "";
for (int i = 0; i < 3; i++)
{
    ikon += MacElis.GetBinary3(ik[0, i]);
}
int strDec = Convert.ToInt32(ikon, 2);
```