Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2025 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти
на тему
“Дослідження та програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях”

Виконав здобувач вищої освіти
ІІ курсу, групи КІ-24М
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Андрусик Б.М.
«____» _____ 2025 р.

Керівник проекту
доктор технічних наук, професор
_____ Коваленко О.В.
«____» _____ 2025 р.
Рецензент _____
_____

м. Кропивницький

АНОТАЦІЯ

Андрусик Б.М. Дослідження та програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2025.

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи скремблювання цифрового сигналу на мобільних пристроях.

Метою розробки є дослідження та програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях.

Об'єктом дослідження є процес скремблювання цифрового сигналу на мобільних пристроях.

Предметом дослідження є методи скремблювання цифрового сигналу на мобільних пристроях.

Методи дослідження базуються на методах теорії сигналів та теорії захисту інформації в мережі, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на мобільному пристрої з ОС Android.

Програму розроблено в середовищі Python.

Ключові слова: комп'ютерна інженерія, скремблювання, цифровий сигнал, мобільні пристрої

ABSTRACT

Andrusyk B.M. Research and software implementation of a digital signal scrambling system on mobile devices. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2025.

In this final qualification work for the second (master's) level of higher education, software has been developed, which is intended for a digital signal scrambling system on mobile devices.

The purpose of the development is the research and software implementation of a digital signal scrambling system on mobile devices.

The object of the research is the process of digital signal scrambling on mobile devices.

The subject of the research is the methods of digital signal scrambling on mobile devices.

The research methods are based on the methods of signal theory and the theory of information protection in the network, methods of mathematical statistics, and methods of software development.

The result of the work is the software implementation of a digital signal scrambling system on mobile devices.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A user-friendly user interface has been developed. Instructions for working with the software are provided.

The program can be used on a mobile device with OS Android.

The program was developed in the Python environment.

Keywords: computer engineering, scrambling, digital signal, mobile devices

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

БД	–	база даних
ПЗ	–	програмне забезпечення
AES	–	Advanced Encryption Standard
USB	–	universal serial bus

ВСТУП

Актуальність теми. Сучасному суспільству часом доводилося зіштовхуватися із проблемою прослуховування телефонних розмов. І цим нікого не здивуєш. В еру новітніх технологій ніколи не можна бути впевненим у повній конфіденційності чого-небудь. Зловмисники, намагаючись заволодіти анонімною інформацією, найчастіше підключаються до телефонних ліній як мобільних, так і стаціонарних телефонів, приносячись абонентам як стаціонарних, так і стільникових телефонів безліч проблем.

Скремблером називають пристрій, призначений для шифрування мови, що проходить через телефон, тим самим воно забезпечує захист і анонімність переговорів.

Шифрування відбувається за допомогою розбивки звукових сигналів на, так звані, під діапазони. Далі кожна частина піддається частотної інверсії, тобто перетворенню високих частот у низькі й навпаки. Поділ спектра на під діапазони відбувається на певній частоті, що називають крапкою розбивки. Крапка розбивки може бути фіксованою (під час розмови не виконується перемикання між режимами) або приймає одне з декількох можливих значень, коли під час телефонного переговору користувачі самі перемикаються між режимами скремблювання.

Процес розшифровки відбувається в тому випадку, якщо алгоритми скремблювання стоять на обох телефонах. Програми, працюючи одночасно, синхронно шифрують розмову. Таким чином, абоненти відмінно розуміють один одного, а зловмисники чують мову, що спотворюється повністю.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем скремблювання цифрового сигналу на мобільних пристроях.

– Дослідження системи скремблювання цифрового сигналу на мобільних пристроях.

– Програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях.

Об'єктом дослідження є процес скремблювання цифрового сигналу на мобільних пристроях.

Предметом дослідження є методи скремблювання цифрового сигналу на мобільних пристроях.

Методи дослідження базуються на методах теорії сигналів та теорії захисту інформації в мережі, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод скремблювання цифрового сигналу на мобільних пристроях.

– Розроблено вітчизняний продукт скремблювання цифрового сигналу на мобільних пристроях, який має більш широкі можливості, на відміну від існуючих аналогів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі скремблювання цифрового сигналу на мобільних пристроях.

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведеним у науковій літературі.

Робота апробована на LVII Науково-технічній конференції здобувачів вищої освіти LV науково-технічної конференції «Наука в ЦНТУ: основні досягнення та перспективи розвитку» (2025 р.), основні положення випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти надруковані у статті збірника праць молодих науковців ЦНТУ, випуск №15.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Пристрій, що користується величезним успіхом у багатьох організаціях – змінювач голосу. Це порівняно невеликий пристрій, що працює одночасно зі стільниковим телефоном. Під час дзвінка абонент на іншому кінці проведення буде чути не ваш голос, тому, ніколи не догадається, з ким має честь говорити.

Пристрої змінювач мови й скремблер багато в чому схожі, по суті, але й існують відмінності: при прослуховуванні телефону, коли працює скремблер, мова зашифрована й чутні шуми, а змінювач міняє лише тембр голосу, робить його більше грубим, високим або низьким.

Засобами захисту інформації користуються найчастіше секретні служби, великі організації, які найчастіше мають комерційні таємниці. Звичайно ж, і звичайному абонентові може придатися захист. Адже нерідко ми стаємо жертвами зловмисників, які приводять до жахаючих наслідків, аж до шантажу.

Процвітаючий бізнесмен обов'язково повинен подбати про конфіденційність особистої інформації. Тому захист телефонних ліній йому просто необхідна, особливо якщо в абонента є підстави підозрювати кого-небудь в інтересі до його дзвінків.

Звичайно ж, скремблювання далеко не єдиний спосіб захистити свої переговори, але воно є таки надійним і ефективним.

1.2 Область застосування

Можливо, що крім скремблера, вам знадобиться й додаткові послуги, приміром, вам потрібно передати секретну інформацію з мобільного, але ви не хочете бути впізнаними. У такому випадку рекомендовано користуватися

пристроєм зміни голосу. Мова буде відтворюватися настільки перекрученим образом, що неможливо буде відрізнити навіть – жіночий він або чоловічий.

Пропоновані змінювачі, працюють за принципом зміни тональності мови від більше низької до високої. Змінювач голосу для стільникових телефонів має 4 варіанти голосу (4 режими). Вибирати їх можна вручну на ваш розсуд.

Ще один плюс, що ми не можемо залишити без уваги: підключення змінювача голосу. Незважаючи на складність і багатозадачність пропонованого пристрою, підключити його зможе кожної. Для цього не потрібно ніяких спеціальних знань і навичок у подібній сфері. Скористатися пристроєм зможе навіть дитина.

Існує кілька варіантів приєднання до телефону: в одному випадку змінювачі підключаються до мобільного телефону за допомогою гарнітури hands free. Більше зручним же способом підключення є пристрій Bluetooth. Але в другому варіанті мобільний телефон повинен підтримувати bluetooth voice changer.

Скремблери, як і змінювачі голосу, мають компактний розмір. Кожна модель відрізняється своєю унікальністю й акцентуванням на будь-якого покупця. Представлені пристрої по максимуму гарантують вам анонімність.

Змінювачі голосу для телефонів і скремблери є абсолютно узаконеними в будь-якій державі. Пристрої носять персональний характер, тому купувати їх чи ні – справа на ваш розсуд. Ми лише зобов'язані сповістити вас про те, що ніяких злобливих махінацій подібні пристрої не несуть, вони вільні в продажі й легалізовані.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

1. Signal: Найкращий варіант для зашифрованих шафок та безпечного обміну повідомленнями

За моїм досвідом, Signal був ідеальною системою миттєвого обміну повідомленнями для надсилання та отримання зашифрованих файлів, а також редагування або зміни розшифрованих файлів. Оскільки я помітив, що надсилати зашифровані файли одержувачам за допомогою інших інструментів було складно, я надав сервіс миттєвого обміну повідомленнями для додавання власних документів із зашифрованими вкладеннями та їх обміну на будь-якій хмарній платформі, такій як Dropbox, Google Drive тощо.

Signal, визнаний лідером у категорії шифрування, має найбільшу силу – це наскрізне шифрування. Він блокує ваші дані повідомлень, записи дзвінків та збережені файли у своєму безпечному сховищі. Мені не довелося турбуватися про те, що хтось шпигуватиме за моїми розмовами, що заспокоює, оскільки сьогодні особисті дані вразливі для всіх глобальних серверів.

Я також дізнався, що він використовує протокол Signal для захисту файлів під час завантаження або вивантаження. Одна з перших речей, яку я помітив, це те, наскільки простий і зрозумілий інтерфейс. Він не перевантажений зайвими функціями, тому ви можете шифрувати та надсилати свої документи чи файли без жодних проблем.

Хоча Signal не перевантажує вас опціями, він все ж пропонує все, що мені потрібно від платформи обміну миттєвими повідомленнями. Він може знищувати незашифровані повідомлення або файли, магнітні дані файлів, входити в систему

з мобільного телефону чи будь-якого іншого гаджета та створювати групові чати для надсилання та отримання зашифрованих файлів.

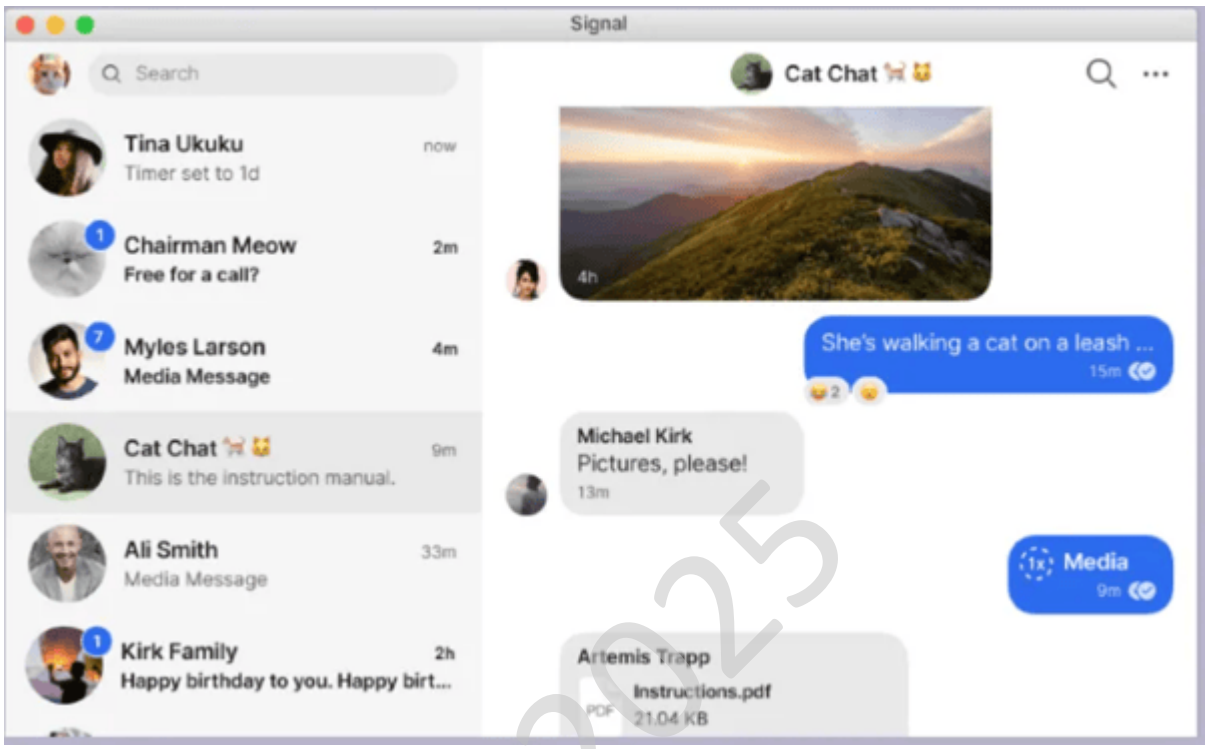

Рисунок 2.1 – Інтерфейс користувача Signal

Signal надійний для безпечного обміну повідомленнями, але є обмеження. Він не підтримує вбудоване дискове або хмарне сховище, а локальне шифрування файлів застосовується лише після ручного конвертування та завантаження файлів на сервер. Кілька користувачів відзначають це як прогалину в обробці файлів та гнучкості безпеки.

Деякі функції, такі як зникаючі повідомлення та керування резервними копіями, не одразу інтуїтивно зрозумілі. Знадобилося трохи часу, щоб їх вивчити, і користувачі згадують подібну криву навчання щодо налаштувань конфіденційності.

Шифрування обмежене контактами, які також користуються Signal. Якщо одержувач не перебуває на платформі, повідомлення не шифруються від початку до кінця, що знижує його ефективність у змішаних розмовах, що часто

підкреслюється навіть в оглядах.

Йому також бракує таких функцій, як менеджер паролів або прозорі параметри шифрування. Хоча Signal корисний для приватного спілкування, він міг би зробити більше для підтримки ширших потреб безпеки даних.

Переваги Signal:

– Мені сподобалося, що Signal перезаписує повідомлення та знищує їх через певний проміжок часу, щоб забезпечити безпеку відправника та одержувача.

– Я міг додати кількох людей до групового чату, і для налаштування чатів було доступно безліч аватарів, емодзі та тем.

Недоліки Signal:

– Хоча Signal забезпечував наскрізне шифрування, я не міг отримувати повідомлення, доки не відкрив додаток Signal. Навіть після налаштування необмеженого використання даних батареї та вимкнення призупинення роботи додатка проблема все ще залишається.

– Я помітив, що версія мелодії дзвінка для дзвінка на робочому столі трохи ріже мені вуха. Вона здавалася дуже старою і її можна було б оновити за допомогою іншої мелодії дзвінка.

2. Microsoft Bitlocker: найкращий для шифрування всього пристрою

Якщо й існує програма, яка виділяється шифруванням файлів та пропонує високотехнологічний шифр для захисту даних компанії, то для мене це не хто інший, як Microsoft Bitlocker.

Microsoft Bitlocker – це вбудований інструмент шифрування від Microsoft для Windows. Він може шифрувати файли або навіть просто блокувати незашифровані файли без їх шифрування.

Якщо шафка відкрита, я можу переглянути всі файли та папки. Але якщо шафка закрита, дані залишаються невловимими та невідстежуваними. Він використовує алгоритм розширеного стандарту шифрування (AES) зі 128-бітними або 256-бітними ключами для захисту ваших даних від сторонніх очей.

Він також пропонує додаткові онлайн- сервіси безпечного видалення та резервного копіювання.

Я також розгорнув Microsoft Bitlocker на різних операційних системах Windows, але виявив, що він ідеально інтегрувався з Windows 10 або Windows 11. Вам не потрібна жодна додаткова інтеграція для налаштування та запуску програми.

Ще однією ключовою особливістю було те, що я міг зберігати всі свої паролі у вбудованому менеджері паролів, додавати шари хешування до шафок і навіть входити в хмарну консоль, щоб отримати доступ до всіх своїх збережених шафок. Microsoft Bitlocker забезпечував безперебійну сумісність з хмарою та надійні служби шифрування для моїх файлів.

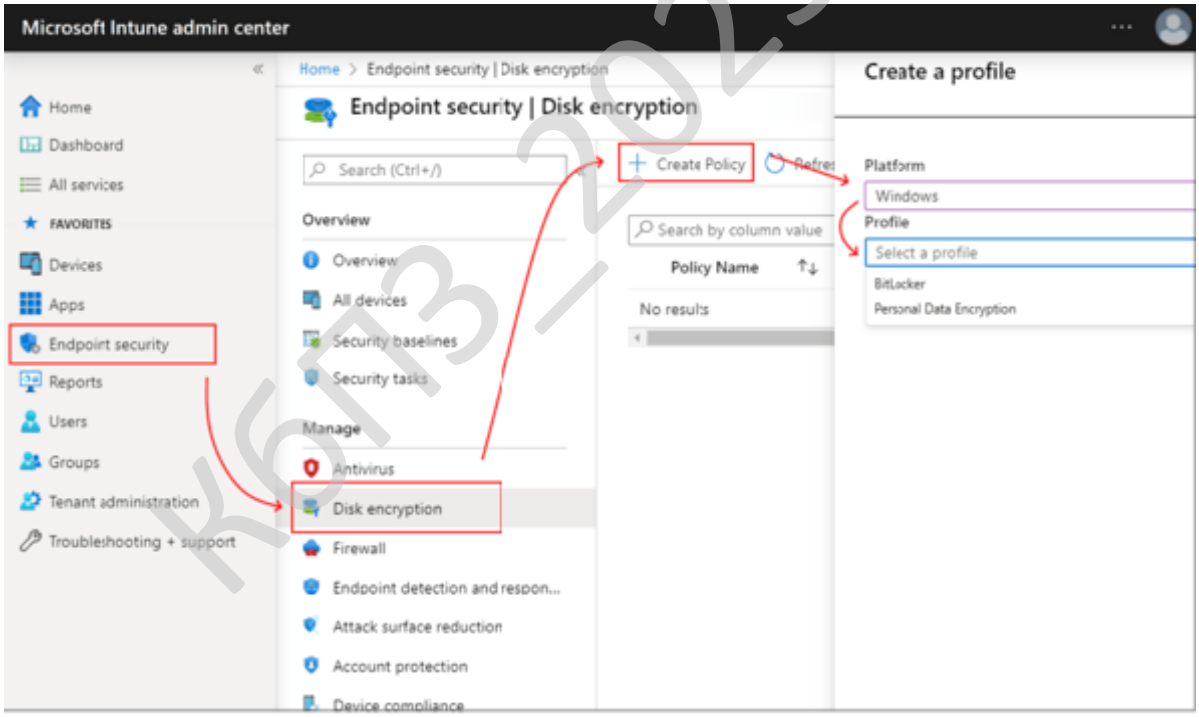

Рисунок 2.2 – Інтерфейс користувача Microsoft Bitlocker

З огляду на це, я зіткнувся з кількома недоліками у зручності використання. Налаштування ключів відновлення було неінтуїтивно зрозумілим, і якщо ви забудете ключ відновлення, доступ до ваших даних буде назавжди

втрачено. Хоча це підвищує безпеку, у мене не було резервного варіанту. Зашифрований файл залишався в моїй системі, повністю недоступним. Рецензенти відзначили це як поширену проблему, особливо коли можливості відновлення обмежені.

Сумісність платформ – ще одна проблема. Згідно з постійними відгуками, BitLocker працює лише у Windows, тому я не міг використовувати його на macOS або Android, а також ділитися зашифрованими файлами з користувачами цих систем. Ця відсутність кросплатформної підтримки часто очевидна в середовищах зі змішаними пристроями.

Деякі функції також здавалися надмірно технічними. Під час перевірки TPM процес оновлення нагадував інструмент командного рядка, яким було нелегко користуватися. Я не зміг завершити процес відновлення пароля без сторонньої допомоги. Відгуки відображають аналогічні занепокоєння щодо налаштування та усунення несправностей, особливо для користувачів без технічних знань.

Переваги Microsoft Bitlocker:

– Завантаження файлів, їх шифрування та налаштування механізму інтерфейсу з відкритим ключем для безпечної передачі даних було дуже простим та безпроблемним.

– Він бездоганно інтегрувався з іншими інструментами для співпраці Microsoft, такими як Microsoft Outlook або Microsoft OneNote, що дозволило мені надсилати файли, не боячись їх підробки.

Недоліки Microsoft Bitlocker:

– Хоча платформа пропонує загалом інтуїтивно зрозумілий інтерфейс, я був розчарований тим, що не зміг налаштувати автоматизацію для розгортання PIN-коду "перед завантаженням" без допомоги служби підтримки клієнтів моєї операційної системи.

– Я також помітив, що процес відновлення пароля дуже складний для зашифрованих файлів, що запобігає доступу навіть авторизованих користувачів

до критично важливих даних.

Регулярно перевіряйте стан вашої системи та діагностуйте вразливості в режимі реального часу за допомогою безкоштовного антивірусного програмного забезпечення, щоб запобігти будь-якій ймовірності зараження троянами чи шкідливими програмами.

3. Progress MOVEit: Найкращий для безпечної передачі файлів

Щойно я протестував Progress MOVEit, я одразу припустив, що це буде ідеальний вибір для команд кібербезпеки середнього та великого рівня. Він керує, контролює та автоматизує всі процеси обміну та передачі файлів на одній централізованій платформі.

Чи то послуги шифрування, автоматизація файлів, відстеження рівня активності чи підтримка угод про рівень обслуговування, Progress MOVEit не має нестачі функцій. Він використовує передові протоколи передачі файлів та асиметричну криптографію для доставки файлів з однієї системи в іншу.

Крім того, це також дозволило мені залучити нових партнерів з обробки даних, налаштувати вхід до хмарного сховища та запускати резервну синхронізацію незашифрованих документів, щоб переконатися, що дані не втрачені. Від простого інтерфейсу до інтуїтивно зрозумілих робочих процесів, вам не потрібно бути IT-генієм, щоб працювати з цією платформою. Я міг швидко ділитися зашифрованими файлами та навіть налаштувати автоматизацію робочих процесів, щоб зменшити ручний моніторинг та діагностику даних.

Щоразу, коли я стикався з труднощами, їхня команда підтримки клієнтів завжди була на місці. Крім того, база знань, яку вони надають, є скарбницею корисної інформації, ідеальної для самостійного вирішення незначних проблем. Незалежно від того, чи ви займаєтесь рутинними передачами даних, чи міграцією великих обсягів даних, Progress MOVEit – це надійний та надійний інструмент для роботи.

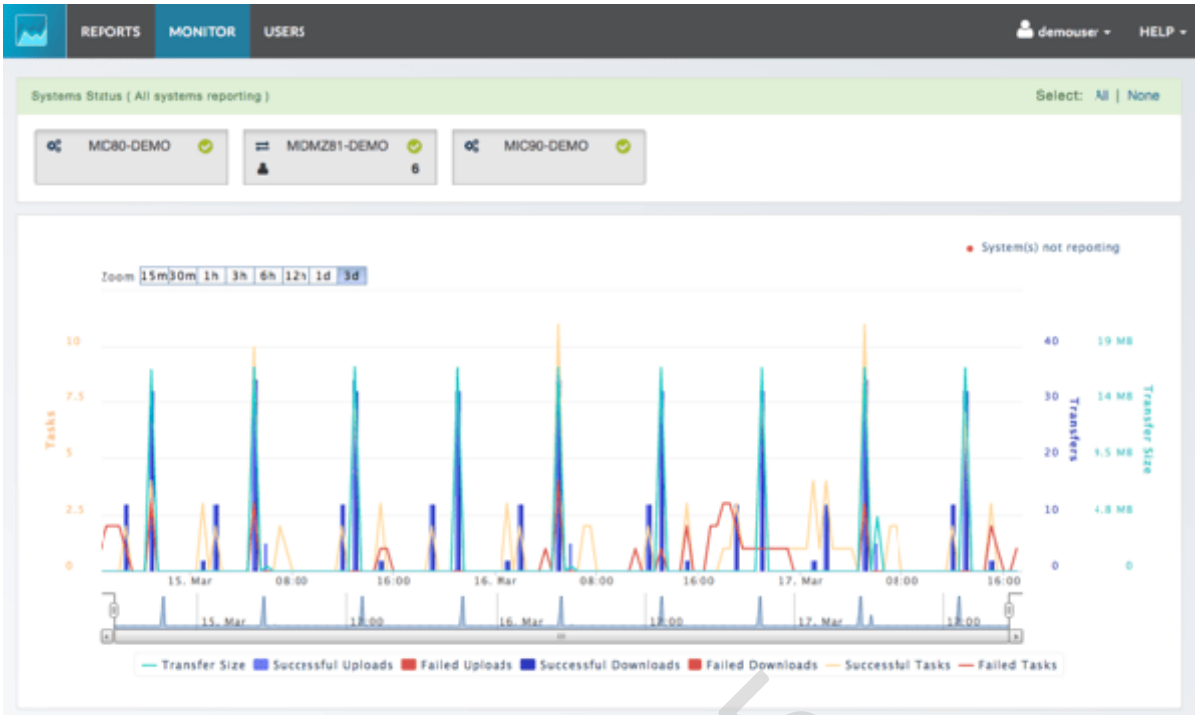

Рисунок 2.3 – Інтерфейс користувача Progress MOVEit

Progress MOVEit надійний у плані шифрування, але під час використання я зіткнувся з кількома проблемами. Моя система зависала під час тривалих оновлень, що ускладнювало шифрування файлів за потреби. Користувачі також згадували про таке уповільнення продуктивності, особливо коли програмне забезпечення працює одночасно з іншими системними процесами.

Я також вважаю, що звітність та аналітика обмежені. Було важко отримати чіткий огляд ключових деталей, таких як кількість сканованих дисків, інформація про останню резервну копію або користувацькі звіти з позначками часу. Як людина, яка цінує прозорість та детальні журнали, я вважаю це неприємним. Оглядачі, які шукають розширену аналітику, повторили схожі відгуки.

Хоча я довіряю можливостям шифрування MOVEit, минулі інциденти безпеки залишили мене дещо настороженими. Я хотів би бачити сильніший контроль паролів та механізми хешування, особливо для стиснення та захисту кількох файлів. Кілька користувачів зазначили, що вдосконалені заходи

зміцнення довіри можуть підвищити впевненість у його довгостроковому використанні.

Переваги Progress MOVEit:

– Мені сподобався інтерфейс відстеження активності, який відображає облікові дані користувачів, які востаннє входили в систему та отримували доступ до файлів через програму.

– Інтеграції та оновлення для нових партнерів з обробки даних були зрозумілими та дозволили мені налаштувати надійні інтеграції безпеки.

Недоліки Progress MOVEit:

– Інструменти налаштування документів загалом досить зручні та прості у використанні. Однак я зіткнувся з обмеженням, коли міг посилатися лише на одне зашифроване вкладення в одному електронному листі, що обмежувало мій робочий процес.

– Я також помітив, що не можна вимкнути алгоритми шифрування на кількох хостах одним клацанням миші, що вимикало обмін файлами, навіть коли одержувач мав пароль.

Розгляньте різницю між токенізацією та шифруванням, щоб з'ясувати, який спосіб безпеки даних краще підходить для захисту цифрових активів вашого бізнесу.

4. FileVault: найкращий для резервного копіювання та відновлення

FileVault не лише запропонував послуги криптографії для моїх окремих файлів, але й розширив послугу шифрування на весь диск. FileVault пропонує функції шифрування, такі як контейнери для зберігання, сховища та шафки, які можуть зберігати будь-що, від документів, що захищають особу, до інформації про кредитні картки.

FileVault – це вбудована функція шифрування macOS. Вона надає такі послуги, як повне шифрування диска, онлайн-подрібнення файлів, протокол безпеки файлів, а також онлайн-резервне копіювання та відновлення. Вона безкоштовна, входить до кожного Mac і не потребує додаткового налаштування.

Що мені найбільше подобається у FileVault, так це його надійний захист. Він використовує шифрування XTS-AES-128 з 256-бітним ключем, що звучить надто технічно – і це так – але, якщо говорити простою мовою, це означає, що мої дані заблоковані, як Форт-Нокс, за допомогою головного пароля.

Чи то особисті документи, робочі файли чи щось інше, FileVault гарантує, що файли будуть зашифровані та перетворені на шифр, щоб захистити їх від неетичних хижаків. Усі дані пов'язані з моїми обліковими даними Mac, тобто доступ до них матиму лише я або хтось із моїм закритим ключем.

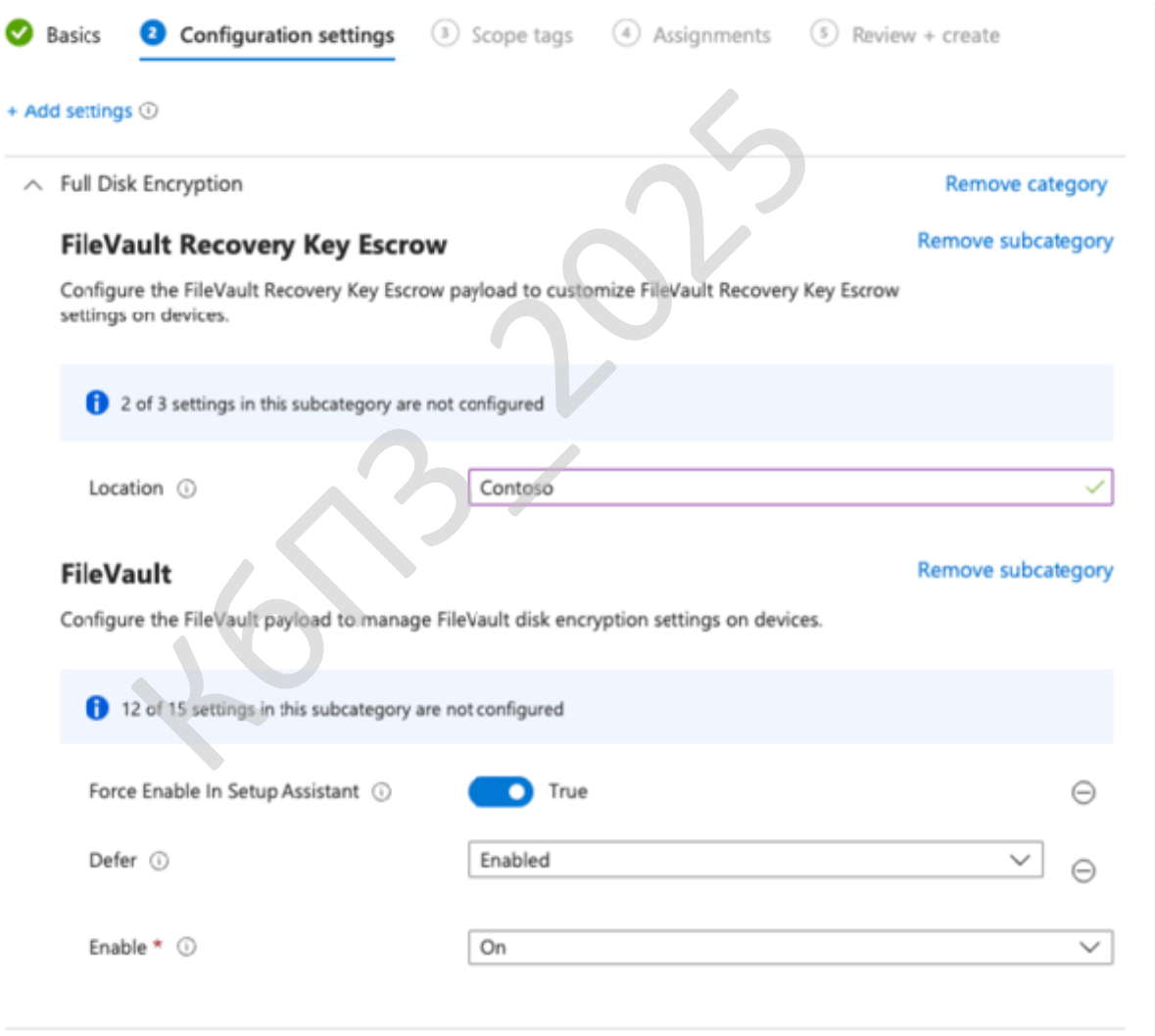

Рисунок 2.4 – Інтерфейс користувача FileVault

Тим не менш, FileVault має свої Недоліки. Коли його ввімкнено, я помітив невелике падіння продуктивності. Це можна контролювати, але якщо ви користуєтеся старішою версією macOS або запускаєте програми, що витрачають багато ресурсів, затримки можуть бути більш помітними. Кілька користувачів також повідомляли про уповільнення продуктивності в подібних налаштуваннях.

Доступність – ще один недолік, який я помітив. Немає мобільного додатку чи доступу з різних пристроїв, тому я не можу керувати зашифрованими файлами або переглядати їх віддалено. Також бракує елементів керування на рівні користувача; якщо один Mac використовують кілька людей, важко налаштувати маскування даних або обмежити доступ між обліковими записами. Це питання було піднято на користувачами, яким потрібен більш детальний захист у спільних середовищах.

Незважаючи на ці проблеми, FileVault все ще забезпечує надійне шифрування кінцевих точок. Однак його обмеження варто враховувати для більш гнучких сценаріїв або сценаріїв зі спільним використанням пристроїв.

Переваги FileVault:

– FileVault забезпечив повне шифрування диска та заблокував увесь диск моїх файлів, щоб вони залишалися прихованими від інших програм.

– Я міг легко перейти до FileVault через Mac та зашифрувати свої файли набагато зручнішим та доступнішим способом.

Недоліки FileVault:

– Функції шифрування добре працюють для захисту окремих файлів, що я дуже ціную. Але мені було важко зберегти цілісність даних, коли кілька користувачів отримували доступ до мого Mac і намагалися відкрити шафки, що створювало деякі неочікувані проблеми.

– Я також виявив, що FileVault втручався в деякі збережені файли в моїй системі та зрештою змінював їх або робив нечитабельними.

5. Virtru: Найкращий для захисту файлів на основі PKI

З мого досвіду, Virtru має бути найуніверсальнішим інструментом

шифрування електронної пошти для передачі файлів електронною поштою з одного комп'ютера на інший.

Virtru захищає вміст надісланих електронних листів, а також надає мені журнал аудиту вкладень, безпечно надісланих поштовим сервером. Мені особливо подобається, як легко він інтегрується з такими платформами, як Gmail або іншими поштовими клієнтами. Вам просто потрібно ввімкнути його, і він забезпечить оптимальну безпеку та підтримку алгоритмів шифрування для ваших вкладень електронної пошти.

Virtru також використовує наскрізне шифрування файлів, щоб гарантувати вам та вашому цільовому одержувачу доступ до пошти. Мені просто потрібно було встановити відкритий ключ для шифрування електронного листа та поділитися ним через пристрій обміну миттєвими повідомленнями з користувачем, який може надати пароль та відкрити вкладення. Це здавалося простим, зручним та водночас безпечним.

Мене також вразив набір інтеграцій Virtru в режимі реального часу, особливо ті, що стосуються Microsoft Outlook та Google Workspace. Йдеться не лише про блокування власної поштової скриньки, а й про захист конфіденційних даних, таких як бізнес-інформація, дані про особисті транзакції тощо.

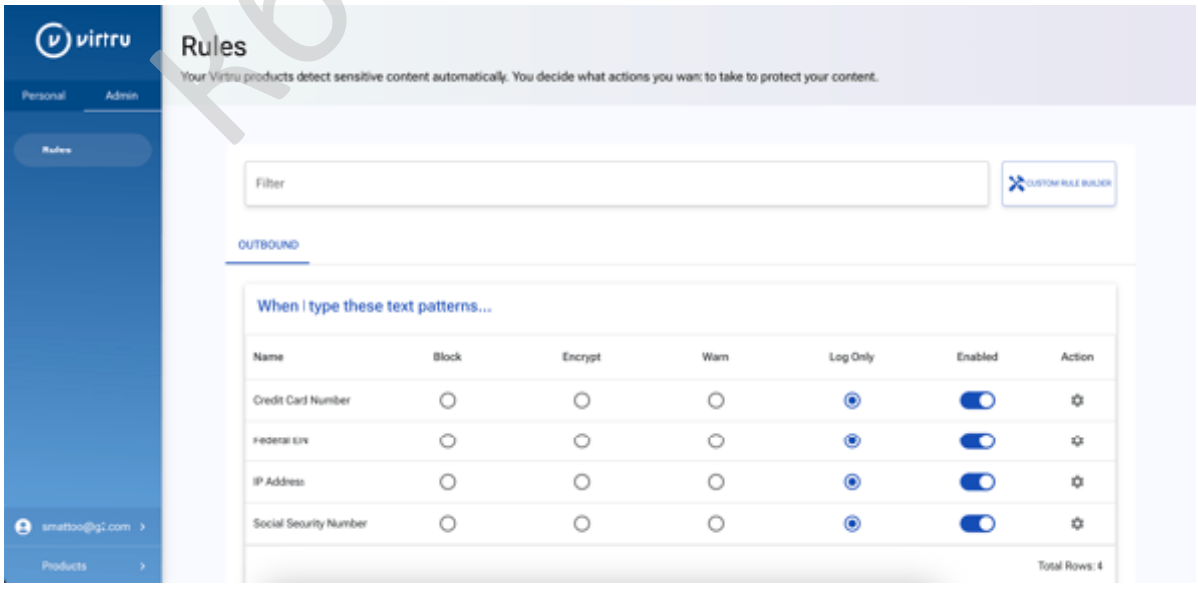

Рисунок 2.5 – Інтерфейс користувача Virtru

Однак, я зіткнувся з кількома проблемами під час використання мобільного пристрою. На моєму телефоні електронні листи не завжди відкривалися плавно, що порушувало мій робочий процес у дорозі. Кілька користувачів вказували на подібні проблеми сумісності з мобільним доступом, особливо в різних поштових клієнтах.

Я також зіткнувся з труднощами під час обміну файлами. Хоча шифрування електронної пошти та контроль доступу працюють добре, я не міг надсилати документи у форматі RTF з кількома зашифрованими вкладеннями, коли розмір файлу перевищував ліміт Virtru. Платформа повідомляла мене, що не може надсилати або отримувати файли, що перевищують певний розмір, що ускладнювало безпечний обмін більшими документами, що також обговорювалося в оглядах для користувачів, які працюють з вкладеннями великого обсягу.

Тим не менш, для тих, хто, як я, надає пріоритет безпеці даних електронної пошти, Virtru все ще пропонує міцний душевний спокій. Він чудово підходить для галузей, що обробляють конфіденційну інформацію, таких як охорона здоров'я, фінанси та електронна комерція. Але краща підтримка мобільних пристроїв та менше обмежень щодо файлів зробили б його ще ефективнішим.

Переваги Virtru:

– Завдяки розширеній функції аудиту електронної пошти я міг точно бачити, хто переглядав мої листи, а також дату й час.

– Я зміг налаштувати власні елементи керування тим, хто може переглядати вкладення моїх електронних листів, і навіть скасувати доступ через певний проміжок часу.

Недоліки Virtru:

– Хоча інструмент пропонував надійні можливості шифрування, я мав проблеми з надзвичайно складним інтерфейсом користувача та періодичними затримками, що впливало на зручність використання як для відправників, так і

для одержувачів.

– Я вважаю, що це трохи завищено порівняно з іншими платформами, які я тестував, особливо враховуючи, що подібні інструменти пропонують конкурентні функції за нижчими цінами.

6. Tresorit: Найкращий для аудиту відповідності вимогам

Якщо ви, як і я, фанат безпеки, Tresorit стане для вас універсальним сервісом для редагування, керування та розшифрування файлів із покращеним захистом.

Tresorit – це інструмент для спільної роботи над контентом, який дозволяє вам взаємодіяти, вести розмови, ділитися вкладеннями та ресурсами, а також керувати журналами аудиту даних для процесів вашої компанії. Я можу блокувати отримані дані в папках і приховувати їх у головному меню, щоб вони залишалися недоступними для інших учасників контенту.

Для бізнесу та всіх, хто хоче дотримуватися GDPR або HIPAA, Tresorit пропонує варіанти зберігання даних для токенізації файлів даних та їх доставки через захищені мережі.

Ще однією функцією, яку я вважаю надзвичайно зручною, було налаштування посилань для безпечного депозиту. Це дозволяє людям безпечно надсилати мені файли. І їм неймовірно легко цим користуватися. Крім того, у мене було кілька варіантів, таких як шифрування файлів і папок, знищення файлів, перезапис файлів і доступ до сховищ через командний рядок для керування та захисту цифрових файлів, а також відстеження активності користувачів у режимі реального часу по всіх офісах компанії.

Tresorit пропонує надійний захист для керування контентом, але є кілька недоліків. Ціна може бути високою для фрілансерів або невеликих команд; вона дещо вища, особливо порівняно з іншими інструментами безпечного зберігання. Оглядачі на аналогічних посадах часто називають вартість ключовим фактором для користувачів, які стежать за бюджетом.

Доступ офлайн не був таким гладким, як я очікував. Завантаження файлів

для використання офлайн здавалося незграбним, а керування файлами без підключення до Інтернету було неінтуїтивно зрозумілим. Інтеграція Tresorit з VPN також вимагає налаштування IP-адрес та безпечних URL-адрес, що додає накладних витрат на налаштування. Відгуки відображають аналогічні відгуки щодо зручності використання офлайн та етапів технічної інтеграції.

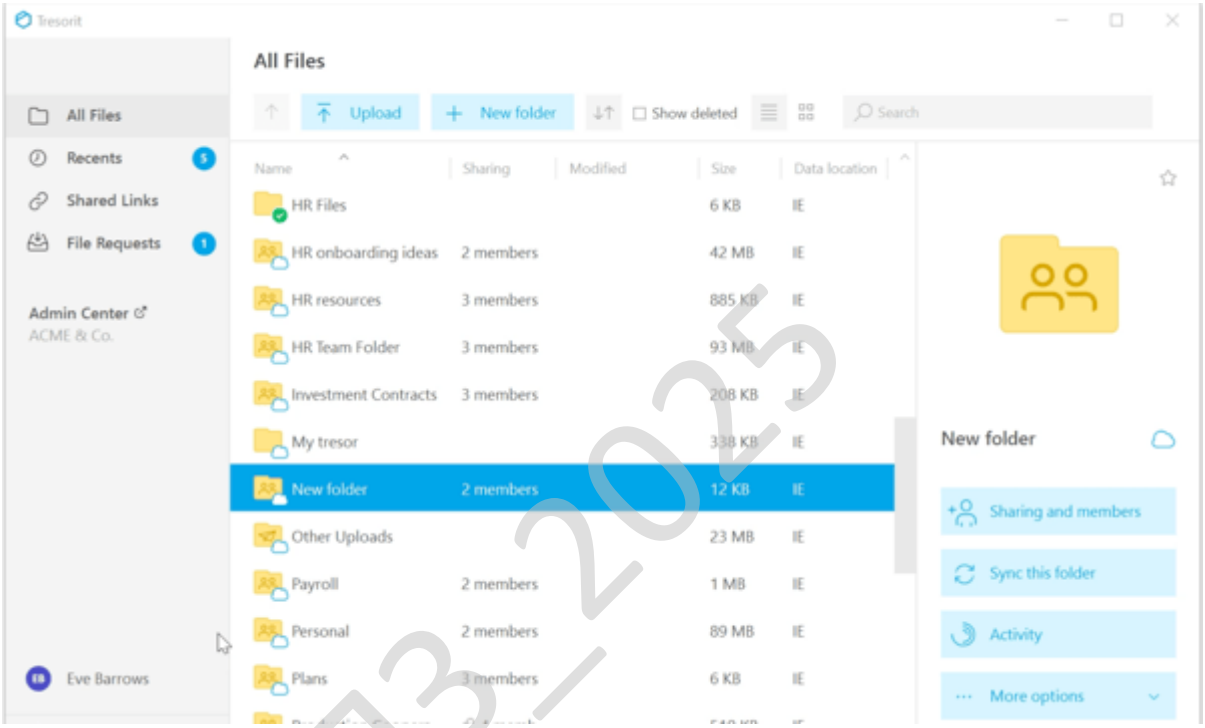

Рисунок 2.6 – Інтерфейс користувача Tresorit

Я також помітив затримки під час синхронізації та шифрування. Хоча це й не було критичною проблемою, це уповільнювало мене під час роботи в умовах обмеженого часу або спроб швидко оновити бази даних. Кілька користувачів зазначили, що продуктивність у режимі реального часу можна покращити, особливо в динамічних середовищах.

Переваги Тресоріті:

– Мені вдалося автоматизувати резервне копіювання та синхронізацію на кожному пристрої без ручної інтеграції з будь-яким іншим стороннім інструментом резервного копіювання.

– Мені сподобалася сумісність обладнання, яка дозволила мені використовувати мобільний додаток для зв'язку з клієнтами та забезпечення максимальної безпеки даних.

Недоліки Tresorit:

– Хоча я міг блокувати та шифрувати контент, я не міг сканувати та ділитися своїми PDF-файлами безпосередньо з командою, оскільки сканер документів не мав опції «поділитися».

– Я дійшов висновку, що щоразу, коли я видаляв резервний файл, я випадково видаляв і оригінальний файл із хмари. Незашифрована версія файлу не зберігалася.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Python – це потужна мова програмування, яка проста у вивченні. Він має ефективні структури даних високого рівня та простий, але ефективний підхід до об’єктно-орієнтованого програмування. Елегантний синтаксис і динамічна типізація Python разом з його інтерпретованим характером роблять його ідеальною мовою для створення сценаріїв і швидкої розробки додатків у багатьох сферах на більшості платформ.

Інтерпретатор Python і обширна стандартна бібліотека доступні у вихідному або двійковому вигляді для всіх основних платформ на веб-сайті Python https://www.python.org/ і можуть вільно поширюватися. Цей же сайт також містить дистрибутиви та вказівники на багато безкоштовних сторонніх модулів Python, програм і інструментів, а також додаткову документацію.

Інтерпретатор Python легко розширюється за допомогою нових функцій і типів даних, реалізованих у C або C++ (або інших мовах, які можна викликати з C). Python також підходить як мова розширення для налаштовуваних програм.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи скремблювання цифрового сигналу на мобільних пристроях.

В процесі розробки випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Програмне забезпечення для шифрування пропонує симетричні та асиметричні криптографічні функції для шифрування та захисту файлів, а також розшифрування зашифрованих файлів за допомогою відкритого або закритого ключа. Воно захищає ваші локальні файли та веб-матеріали від несанкціонованого шпигунства або зловмисників. Під час тестування та експериментів я дізнався, як змінилися шаблони шифрування та перейшли від традиційних алгоритмів шифрування до сучасних функцій, таких як хешування, парольні фрази та блокування.

Цікаво спостерігати, як ми пройшли довгий шлях від шифрування та захисту файлів за допомогою шифротексту до їх зберігання в хмарних контейнерах або шафках. Незалежно від того, чи зберігаються мої дані локально, чи в хмарі, ці найкращі інструменти шифрування розширюють свої послуги завдяки простій та безпечній автентифікації на публічні та приватні хмари. Крім того, функція інтерфейсу відкритого ключа (PKI) допомогла мені обмінюватися захищеними даними, надаючи спільний відкритий ключ одержувачу, зберігаючи при цьому його конфіденційність від інших користувачів.

Одне можна сказати точно: це програмне забезпечення створює цифрову фортецю навколо ваших конфіденційних даних, недоступну для шпигунів чи спецслужб, і забезпечує повний захист від шкідливих програм та атак методом грубої сили. Але хоча я ділюся власним досвідом того, наскільки я задоволений послугами шифрування, я б сказав, що реєстрація на пробну версію або реєстрація на користувацьку демонстрацію дасть вам безпосередній досвід роботи з інструментами для формування короткого списку ваших процесів шифрування даних.

Розробка програмного забезпечення для шифрування був для мене двояким процесом: я хотів розробити інструменти, які пропонують наскрізне шифрування файлів і які легко налаштувати, впровадити та запустити. Я

експериментував з різними локальними та хмарними пристроями з різною ємністю сховища, щоб оцінити, який інструмент шифрування пропонує максимальну сумісність та можливості обміну даними для передачі даних з точки А в точку Б без втрати пакетів.

Окрім складання контрольного списку алгоритмів шифрування, присутніх у програмному забезпеченні для шифрування, я пошукав додаткові інструменти plug-and-play та онлайн-сервіси резервного копіювання, щоб подвійно перевірити, чи не залишилося в системі незашифрованих файлів. Таким чином, я окреслив відповідні параметри для інструменту шифрування, які слід розробити, щоб отримати віддачу від своїх інвестицій та створити сильну систему кібербезпеки для вашої компанії.

– Онлайн-знищення файлів: Якщо інструмент шифрування залишає зашифрований файл після блокування зашифрованої версії, це може збільшити ризик шкідливого програмного забезпечення або атак методом грубої сили. Наявність опції знищення або перезапису файлів гарантує, що на диску не залишиться слідів оригінального файлу або двійкових кодів, які можуть бути зламані або розшифровані зовнішніми хакерами. Інструменти шифрування, які пропонували додаткову інтеграцію з інструментами безпечного видалення або знищувачами файлів, також потрапили до цього списку, оскільки після шифрування файлу я міг використовувати ці інструменти для автоматичного видалення оригінального файлу.

– Алгоритми хешування: Встановлення головного пароля для шафки або сховища дуже мене бентежило, якщо інструмент не мав менеджера паролів, і я в результаті встановлював передбачувані паролі. Ось чому алгоритми хешування є обов'язковою добавкою в програмному забезпеченні для шифрування, яке встановлює гарячі клавіші для спільних даних з одержувачем. Розширені алгоритми хешування, такі як SHA-3 або Argon2, додають додатковий рівень безпеки до зашифрованих файлів, щоб вони не були розшифровані, навіть якщо хтось знає пароль.

– Алгоритми шифрування: Алгоритми шифрування, такі як AES, RSA або Blowfish, використовуються для зберігання конфіденційної інформації про співробітників або захисту даних за допомогою токенізації. Я використовував віртуальну приватну мережу, яка робила мої файли даних досить вразливими та

схильними до вірусів, бо я використовував алгоритми шифрування для захисту своїх файлів від веб-трафіку. Програмне забезпечення для шифрування повинно пропонувати алгоритми шифрування, щоб запобігти зовнішньому перериванню або порушенням під час процесу передачі даних.

– Підтримка апаратного шифрування: Ще однією важливою функцією, яку я оцінив, була можливість шифрування та токенізації файлів на всіх апаратних обчислювальних системах. Будь то Windows, віртуальна машина macOS чи USB-накопичувач, я додав інструменти, які інтегруються з апаратними модулями, такими як модулі довіреної платформи (TPM) АБО апаратні модулі безпеки (HSM) для захисту цифрових файлів на різних пристроях та створення швидкого та ефективного робочого процесу шифрування для компаній.

– Інтеграція з інфраструктурою відкритих ключів (PKI): Завдяки функції інфраструктури відкритих ключів (PKI) я можу зашифрувати файл за допомогою відкритого ключа та поділитися ним з одержувачем, який розшифрує його за допомогою закритого ключа. Це означає, що якщо файл відкритий, будь-хто може отримати до нього доступ за допомогою відкритого ключа, але якщо він заблокований, він повністю недоступний для всіх, і тільки я можу відкрити його за допомогою закритого ключа. Ця система ідеально підходить для безпечного зв'язку, де я можу вільно ділитися відкритими ключами, не турбуючись про те, що кожен прочитає вміст файлу, призначеного для мене.

– Повне шифрування диска та розділу: Повне шифрування диска виконує всі функції: шифрує файли на диску, знищує дублікати файлів та перезаписує дисковий простір новими папками для забезпечення більшої цілісності даних. Шифрування розділів також корисне, оскільки я можу вибирати, стискати та шифрувати лише критично важливі документи на своєму диску та захищати їх у сховищі, і ця функція є найбільш вигідною, якщо інші також мають доступ до вашого диска. Повне шифрування диска є важливою функцією, оскільки вона блокує та захищає всі файли та папки, збережені на диску, і навіть забезпечує хмарне шифрування для хмарних дисків.

– Ескроу-ключ безпеки: Ескроу -ключ безпеки – це система безпеки. Це схоже на надання довіреної авторизації ваших даних відомим зацікавленим сторонам та обмін ключем шифрування через іншу систему обміну повідомленнями, щоб мати резервну копію на випадок, якщо я забуду ключ. Для

тих, хто, як я, має проблеми з керуванням кількома паролями, це позбавляє мене клопоту з налаштуванням відновлення пароля, оскільки я можу поділитися ключем з уповноваженими учасниками для доступу до зашифрованих файлів і папок.

– Безпечне завантаження та інтеграція прошивки: Безпечне завантаження – це питання довіри. Коли я вмикаю свій пристрій, безпечне завантаження гарантує, що завантажуються лише довірені операційні системи та ядра. Це запобігає будь-якому зовнішньому втручанню або стеженню під час завантаження операційної системи для захисту апаратних та програмних файлів. Програмне забезпечення для шифрування, яке пропонує захист прошивки, є надзвичайно важливим, якщо ви хочете додати кілька рівнів захисту до своїх локальних файлів та веб-файлів.

– Шардінг ключів: Деякі інструменти шифрування повністю нівелювали будь-які масштаби крадіжки даних, оскільки вони не лише керували моїми паролями для сховищ, що зберігали мої документи, але й розділяли ключ на частини та зберігали кожну частину ключа доступу в окремому місці. Тільки я зміг зібрати свій ключ назад та захистити свої файли та папки від доступу під час моєї відсутності.

Ці параметри виділяють інструмент шифрування серед шуму різних систем кібербезпеки на ринку. Тепер, маючи власний досвід випробування та дослідження найкращих інструментів шифрування які я відібрав, я впевнений, що моє рішення розроблено для захисту мобільного зв’язку, ваших локальних та хмарних баз даних серверів і встановлення високих стандартів безпеки даних, одночасно зменшуючи витоки даних або несанкціонований доступ.

Хоча вищезазначені функції надали мені необхідні Переваги, забезпечивши безперебійний та безпечний обмін даними та створюючи сховища для моєї зручності, у компаній інший процес складання короткого списку відповідного інструменту. Перш ніж потрапити до цього списку, окресліть свої вимоги, бюджет, робочі процеси мережевої архітектури та існуючі рішення, щоб проаналізувати найбільш підходяще програмне забезпечення для шифрування.

Наведений нижче список містить справжні відгуки користувачів про програмне забезпечення для шифрування. Важливо зазначити, що постачальники, які пропонують безкоштовну пробну версію, також вважаються безкоштовними в

цьому контексті.

Щоб бути включеним до цієї категорії, рішення повинно:

– Захист даних і файлів за допомогою шифротексту.

– Підготуйте для шифрування дані, що зберігаються, дані, що передаються, або дані, що використовуються.

– Дозвольте користувачам вибирати та керувати своїми файлами в налаштуваннях шифрування.

3.2 Розробка структурної схеми

У даній роботі, для системи скремблювання цифрового сигналу на мобільних пристроях використовується алгоритм AES.

Структурна схема наведена на рисунку 3.1. З неї ми бачимо, що розроблена система складається з наступних структурних блоків.

– Дані, які передаються між мобільними пристроями.

– Блок шифрування за допомогою алгоритму AES для скремблювання цифрового сигналу на мобільних пристроях.

– Блок розшифрування за допомогою алгоритму AES для скремблювання цифрового сигналу на мобільних пристроях.

– Мобільний пристрій, на який надходять зашифровані сигнали.

Основним блок системи є блок шифрування AES. Розглянемо його більш детальніше.

Алгоритм шифрування AES працює наступним чином:

1. Дані для шифрування input, розбивається на блоки та копіюються до установочного масиву State, згідно визначеного правилу.

2. Формується сеансовий ключ Round Key з ключа шифрування Cipher Key, за допомогою функції KeyExpansion().

3. Визначається число раундів в залежності від довжини ключа 10, 12, або 14 разів.

4. Виконання операції шифрування, тобто виконання раундів шифрування визначену в пункті 3 кількість раз:

– застосування SubBytes(), тобто трансформації при шифруванні які обробляють State використовуючи нелінійну таблицю заміщення байтів (S-box), застосовуючи її незалежно до кожного байта State;

– застосування ShiftRows(), тобто трансформації при шифруванні, які обробляють State, циклічно зміщаючи останні три рядки State на різні величини;

– застосування MixColumns(), тобто трансформація при шифруванні яка бере всі стовпці State і змішує їх дані (незалежно друг від друга), щоб одержати нові стовпці;

– застосування AddRoundKey(), тобто трансформація при шифруванні, при якому Round Key XOR *c* State. Довжина RoundKey дорівнює розміру State (ті, якщо *Nb* = 4, то довжина RoundKey дорівнює 128 біт або 16 байт).

5. Формування блоку зашифрованих даних, для цього після завершення останнього раунду трансформації, State копіюється в output за визначеним правилом.

Алгоритм розшифрування AES працює наступним чином:

1. Дані для розшифрування input, розбивається на блоки та копіюються до установочного масиву State, згідно визначеного правилу.

2. Формується сеансовий ключ Round Key з ключа шифрування Cipher Key, за допомогою функції KeyExpansion().

3. Визначається число раундів в залежності від довжини ключа 10, 12, або 14 разів.

4. Виконання операції розшифрування, тобто виконання раундів шифрування визначену в пункті 3 кількість раз:

– застосування InvShiftRows(), яке призначене для трансформації при розшифруванні яка є зворотною стосовно ShiftRows(), тобто трансформації при шифруванні, які обробляють State, циклічно зміщаючи останні три рядки State на різні величини;

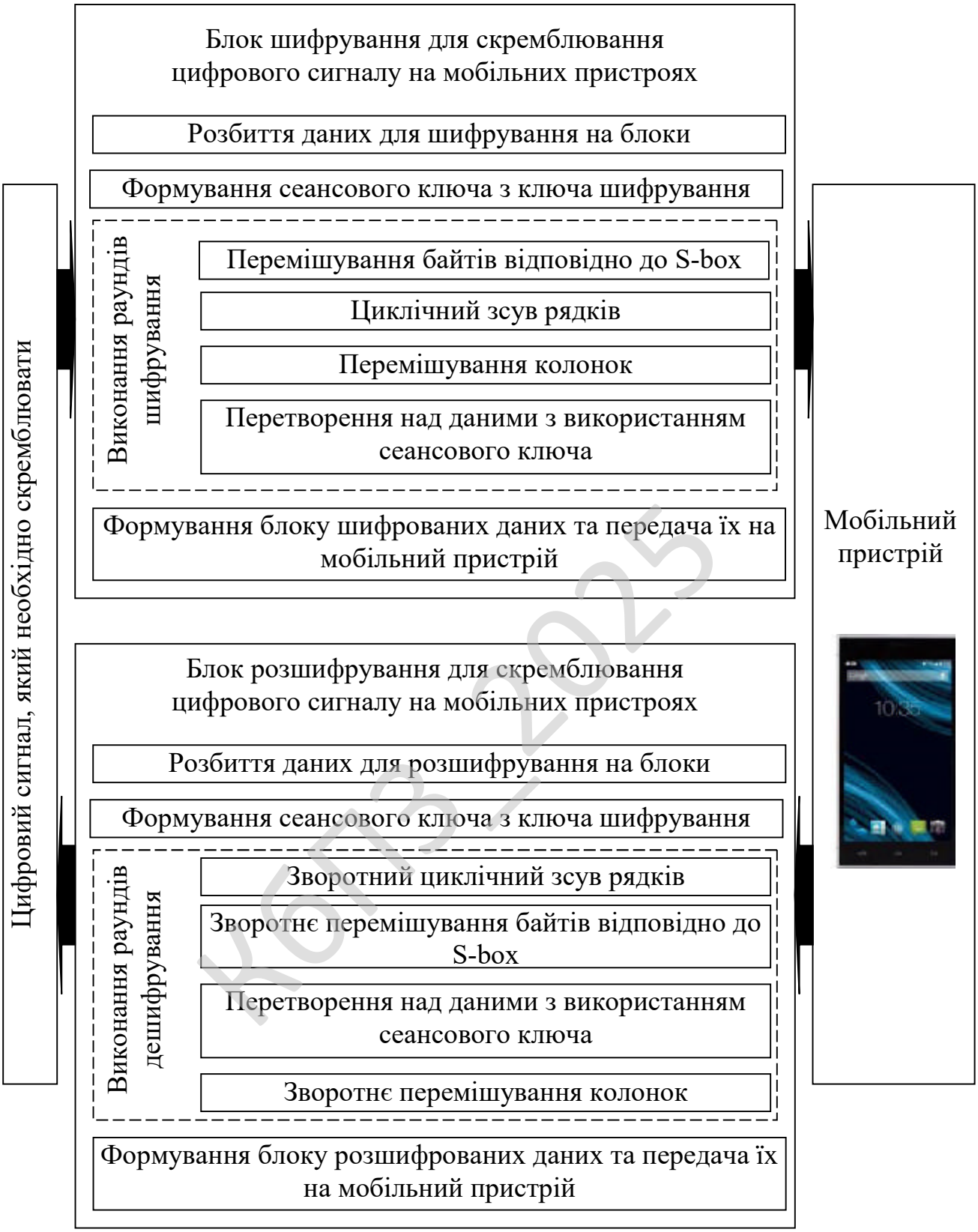

Рисунок 3.1 – Структурна схема системи

– застосування InvSubBytes(), яке призначене для трансформації при розшифруванні яка є зворотною стосовно SubBytes(), тобто трансформації при шифруванні які обробляють State використовуючи нелінійну таблицю заміщення байтів (S-box), застосовуючи її незалежно до кожного байта State;

– застосування InvAddRoundKey(), тобто трансформація при зворотному шифруванні, при якому Round Key XOR з State. Довжина RoundKey дорівнює розміру State (ті, якщо *Nb* = 4, то довжина RoundKey дорівнює 128 біт або 16 байт).

– застосування InvMixColumns(), яке призначене для трансформації при розшифруванні яка є зворотною стосовно MixColumns(), тобто трансформації при шифруванні яка бере всі стовпці State і змішує їх дані (незалежно друг від друга), щоб одержати нові стовпці.

5. Формування блоку роз зашифрованих даних, для цього після завершення останнього раунду трансформації, State копіюється в output за визначеним правилом.

3.3 Розробка функціональної схеми

На рисунку 3.2 зображена функціональна схема системи. Нижче розглянемо її більш докладно.

Функціональна схема складається з наступних блоків:

1. Головне вікно програми.
2. Блок розбиття мобільного пристрою на незашифровану частину, та зашифровану частину.
3. Блок зчитування та перевірки на легітимність паролю.
4. Блок підрахунку спроб введення некоректного паролю.
5. Блок шифрування за допомогою алгоритму AES для скремблювання цифрового сигналу на мобільних пристроях.
6. Блок дешифрування за допомогою алгоритму AES для скремблювання

цифрового сигналу на мобільних пристроях.

7. Блок гарантованого знищення інформації.

8. Блок допомоги та інформації про програму.

Розглянемо більш детально функціональні блоки програмного забезпечення.

Головне вікно програми

Головне вікно додатка призначене для доступу до усіх функцій програми й містить в собі:

– назву програмного модуля;

– рядок головного меню;

– панель інструментів;

– робочу область;

– статусний рядок стану.

Блок розбиття мобільного пристрою на незашифровану частину, та зашифровану частину

Дозволяє розбити накопичувач на відкриту й захищену частини. При виконанні даної операції на першу буде поміщений і невеликий додаток для доступу до другої (хоча перемикатися можна й за допомогою «загальної» утиліти).

Потім задаємо пароль звичайним чином і все готово. Єдиний недолік цієї схеми – оскільки обидві частини монтуються під одним ім'ям і просто перемикаються, одержати доступ відразу до обох неможливо.

Блок зчитування та перевірки на легітимність паролю

Блок зчитування та перевірки на легітимність паролю дозволяє зчитати пароль та порівняти його з тим, який збережений у програмі. Також є можливість заміни паролю, засобом введення старого паролю, та нового паролю з підтвердженням.

Рисунок 3.2 – Функціональна схема програмного забезпечення

Блок підрахунку спроб введення некоректного паролю

Призначення цього блоку заключається у тому, що пристрій автоматично блокується й гарантовано видаляється інформація після 10 невдалих спроб уведення пароля.

Блок шифрування за допомогою алгоритму AES для скремблювання цифрового сигналу на мобільних пристроях

Цей блок шифрує дані використовуючи алгоритм AES. Детальна робота цього алгоритму розписана у пункті 3.2.

Блок дешифрування за допомогою алгоритму AES для скремблювання цифрового сигналу на мобільних пристроях

Цей блок розшифрує дані використовуючи алгоритм AES. Детальна робота цього алгоритму розписана у пункті 3.2.

Блок гарантованого знищення інформації

Цей блок призначений для гарантованого знищення інформації, при неправильному введені паролю. З основу був вибраний алгоритм Гутмана, виходячи з наступних міркувань.

Всі програмні реалізації алгоритмів знищення інформації засновані на найпростіших операціях запису, тим самим відбувається багаторазовий перезапис інформації в секторах диска помилковими даними. Залежно від алгоритму це може бути випадкове число генератора псевдовипадкових чисел або фіксоване значення. Як правило, кожний алгоритм передбачає запис восьми бітових одиниць (#FF) і нуля (#00). В існуючих алгоритмах перезапис може виробляється від одного до 35 і більше раз. Існують реалізації з можливістю довільного вибору числа циклів перезапису.

Теоретично, найпростішим методом знищенні вихідного файлу є його повний перезапис байтом #FF, тобто бітовою маскою з восьми логічних одиниць (11111111), нулів або довільних чисел, тим самим виключивши його програмне відновлення стандартними засобами, доступними користувачеві. Однак з використанням спеціалізованих апаратних засобів, що аналізують поверхню магнітних носіїв і дозволяють відновити вихідну інформацію виходячи з показників залишкової намагніченості, існує ймовірність, що найпростіший перезапис не гарантує повноцінне знищення.

З метою виключення можливості відновлення й розроблені існуючі алгоритми знищення інформації:

– Найбільш відомий і розповсюджений алгоритм, застосовуваний в американському національному стандарті Міністерства оборони Do 5220.22-M. Варіант E відповідно до даного стандарту передбачає два цикли запису

псевдовипадкових чисел і один – фіксованих значень, залежних від значень першого циклу, четвертий цикл – верифікація записів. У варіанті ECE перезапис даних виробляється 7 разів – 3 рази байтом #FF, три #00 і один #F6.

– В алгоритмі Брюса Шнайра в першому циклі записується #FF, у другому – #00 і в п'яти циклах – псевдовипадкові числа. Уважається одним з найбільш ефективних.

– У найбільш повільному, але, на думку безлічі експертів, найбільш ефективному алгоритмі Питера Гутмана, існує 35 циклів, у яких записують усе найбільш ефективні бітові маски, даний алгоритм заснований на його теорії знищення інформації.

Таблиця 3.3 – Алгоритм Гутмана

Цикл	Дані	Цикл	Дані
1	Псевдовипадкові	19	#99
2	Псевдовипадкові	20	#AA
3	Псевдовипадкові	21	#BB
4	Псевдовипадкові	22	#CC
5	#55	23	#DD
6	#AA	24	#EE
7	#92 #49 #24	25	#FF
8	#49 #24 #92	26	#92 #49 #24
9	#24 #92 #49	27	#49 #24 #92
10	#00	28	#24 #92 #49
11	#11	29	#6D #B6 #DB
12	#22	30	#B6 #DB #6D
13	#33	31	#DB #6D #B6
14	#44	32	Псевдовипадкові
15	#55	33	Псевдовипадкові
16	#66	34	Псевдовипадкові
17	#77	35	Псевдовипадкові
18	#88		

– В алгоритмі, передбаченого американським національним стандартом NAVSO P-5239-26 для MFM-кодуємих пристроїв у першому циклі записується #01, у другому – #7FFFFF, у третьому – послідовність псевдовипадкових чисел, у четвертому проходить верифікація. У варіанті для RLL – кодуємих пристроїв даного алгоритму в другому циклі записується #27FFFFF.

– В алгоритмі, що описує німецький національний стандарт VSITR з першого по шостий цикл записуються послідовно байти #00 і #FF, у сьомому #AA.

– Існує думка про існування алгоритму, описаного Російським національним стандартом ДЕРЖСТАНДАРТ Р 50739-95, що передбачає запис #00 у кожний байт кожного сектора для систем з 4-6 класи захисту й запис псевдовипадкових чисел у кожний байт кожного сектора для систем 1-3 класу захисту. Однак даний стандарт містить лише формулювання «Очищення повинне вироблятися шляхом запису інформації, що маскує, до пам’яті при її звільненні перерозподілі», що не містить якої-небудь деталізації щодо порядку перезапису, кількості циклів і бітових масок. У той же час, існує діючий Керівний документ Держтехкомісії Росії «Автоматизовані системи. Захист від несанкціонованого доступу до інформації. Класифікація автоматизованих систем і вимоги по захисту інформації», виданий в 1992 році й ряд, що передбачає, вимог до механізму знищення інформації для систем певних класів захищеності. Зокрема, для класів 3А и 2А «Очищення здійснюється дворазовим довільним записом у область пам'яті, що звільняється, раніше використану для зберігання захищаємих даних (файлів)», для класу 1Г передбачений однократний перезапис.

– В алгоритмі Парагона перший цикл полягає в перезаписі унікальними 512-бітними блоками, використовуючи криптографічно безпечний генератор випадкових чисел, потім, у другому циклі кожний перезаписуваний блок переписується своїм двійковим доповненням, третій цикл повторює перший цикл із новими унікальними випадковими блокам, у четвертому циклі відбувається перезапис байтом #AA. Завершується знищення інформації циклом верифікації.

Як правило, для утруднення програмного відновлення інформації, перезапис інформації в окремому файлі відповідно до алгоритму знищення супроводжується установкою нульового розміру файлу і його перейменуванням, використовуючи довільний набір символів. Потім слідує видалення файлу з таблиці розміщення файлів.

Блок допомоги та інформації про програму

У цьому блоці знаходиться допомога по використанню програми, та інформацію про розробника, версію, та дату випуску програмного продукту.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.12. При детальному її розгляді можна побачити як саме проходе взаємодія у розробленій системі. Використовується модель проектування, графічне представлення «потоків» даних в інформаційній системі.

Діаграма взаємодії процесів використовується для візуалізації процесів обробки даних (структурне проектування). Для розробника вважається звичним спочатку креслити діаграму взаємодії процесів даних рівня контексту, завдяки чому буде показано взаємодію системи. Ця діаграма в подальшому підлягає уточненню шляхом деталізації процесів та потоків даних з метою показати систему що розробляється.

Діаграми потоків даних містять чотири типи елементів:

– Процеси які являють собою трансформацію даних в рамках описуваної системи.

– Сховища даних (репозиторії).

– Зовнішні по відношенню до системи сутності.

– Потоки даних між елементами трьох попередніх типів.

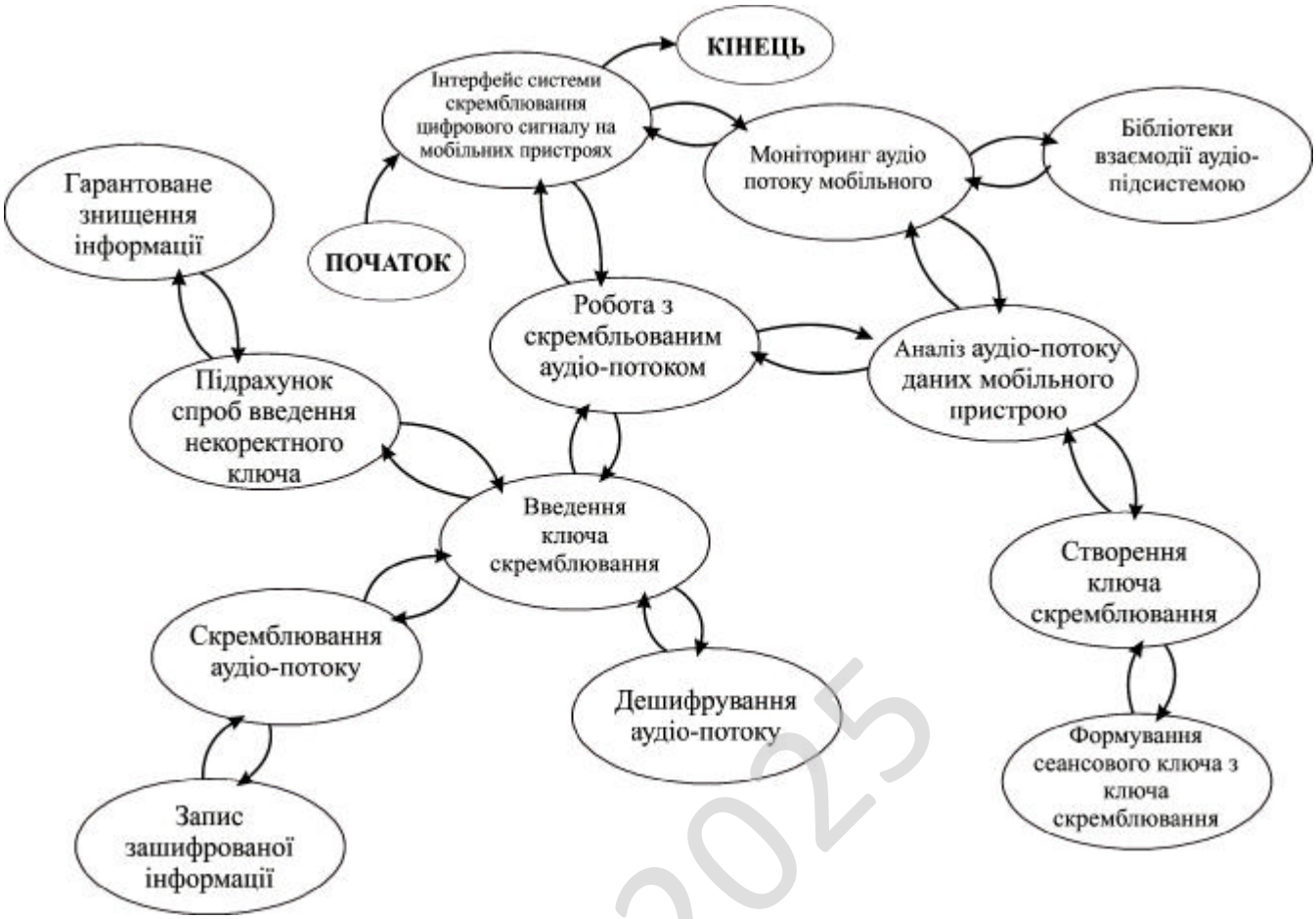

Рисунок 3.3 – Діаграма взаємодії процесів

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

4 РЕАЛІЗАЦІЯ ПРОЕКТУ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Під час роботи над магістерською дипломною роботою було створено блок-схеми. Перед їх розглядом необхідно провести роз’яснення який саме тип блок-схем використовується.

Блок-схема це представлення задачі для її аналізу або розв'язування за допомогою спеціальних символів (геометричних образів), які позначають такі елементи, як операції, потік, дані тощо. Блок вхідних та вихідних даних прийнято позначати паралелограмом, блок обчислень (обробки) даних – прямокутником, блок прийняття рішень – ромбом, еліпсом – початок та кінець алгоритму.

У інформаційних технологіях функціональна схема складається з функціональних блоків, які являють собою конструктивно відособлені частини (елементи або пристрої) автоматичних систем, які виконують певні функції. Функціональні блоки на схемі позначають прямокутниками, всередині яких надписують їх найменування відповідно до функцій, що виконуються. Зв'язки між функціональними блоками (внутрішні впливи) позначаються лініями зі стрілками, які вказують напрям впливів.

Функціональні схеми можуть виконуватися в укрупненому і розгорненому вигляді. У першому випадку на схемі зображають найважливіші блоки системи і зв'язки між ними.

У другому варіанті схема відображається більш детально, що полегшує її читання та ілюструє принцип роботи.

Основні елементи схем алгоритму це термінатор, процес, рішення, зумовлений процес (підпрограма), дані та з'єднувач.

Термінатор це елемент відображає вхід із зовнішнього середовища або вихід з неї (найчастіше застосування – початок і кінець програми). Всередині фігури записується відповідна дія.

Процес це виконання однієї або кількох операцій, обробка даних будь-якого виду (зміна значення даних, форми подання, розташування). Всередині фігури записують безпосередньо самі операції.

Рішення це показує рішення або функцію перемикального типу з одним входом і двома або більше альтернативними виходами, з яких тільки один може бути обраний після обчислення умов, визначених всередині цього елементу. Вхід в елемент позначається лінією, що входить зазвичай у верхню вершину елементу. Якщо виходів два чи три то зазвичай кожен вихід позначається лінією, що виходить з решти вершин (бічних і нижній). Якщо виходів більше трьох, то їх слід показувати однією лінією, що виходить з вершини (частіше нижній) елемента, яка потім розгалужується. Відповідні результати обчислень можуть записуватися поруч з лініями, що відображають ці шляхи.

Зумовлений процес (підпрограма) це символ відображає виконання процесу, що складається з однієї або кількох операцій, що визначені в іншому місці програми (у підпрограмі, модулі). Всередині символу записується назва процесу і передані в нього дані.

Дані це перетворення у форму, придатну для обробки (введення) або відображення результатів обробки (виведення). Цей символ не визначає носія даних (для вказівки типу носія даних використовуються специфічні символи).

З'єднувач це символ відображає вихід в частину схеми і вхід з іншої частини цієї схеми. Використовується для обриву лінії та продовження її в іншому місці (приклад: поділ блок-схеми, що не поміщається на листі). Відповідні сполучні символи повинні мати одне (при тому унікальне) позначення.

Блок-схеми є першоджерелами стратегії розвитку ПЗ. Тому від точності і детальної блок-схеми залежить результат всієї програми.

При виборі початкової точки відліку при побудові схем було враховано, що виходячи з вибору мови програмування і інших технічних засобів, програма буде об’єктно-орієнтована що вимагає оптимізації програми високого рівня, також те, що при розробці програми слід надати особливу увагу модулю системи скремблювання цифрового сигналу на мобільних пристроях.

На рисунку 4.1 зображена основна блок-схема програми, на рисунку 4.2 зображено роботу підпрограми.

З яких видно що робота основної програми складається з початкових етапів ініціалізації ПЗ, перевірки наявності ресурсів системи, блоку початку основного циклу з чеканням запиту від користувача в якому відбувається виклик підпрограми та останньої стадії – перевірка поточного стану з завершенням роботи розробленого ПЗ. При роботі підпрограми виконується основний функціонал системи з циклічними послідовностями, перевіркою поточного стану та поверненням в основну програму прапорів стану виконання.

Було використано підходи з використанням UML, це уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, називаної UML-моделлю. UML був створений для визначення, візуалізації, проектування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація.

UML може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки прикладних програм. Різні види діаграм які підтримуються UML, і найбагатший набір можливостей представлення певних аспектів системи робить UML універсальним засобом опису як програмних, так і ділових систем.

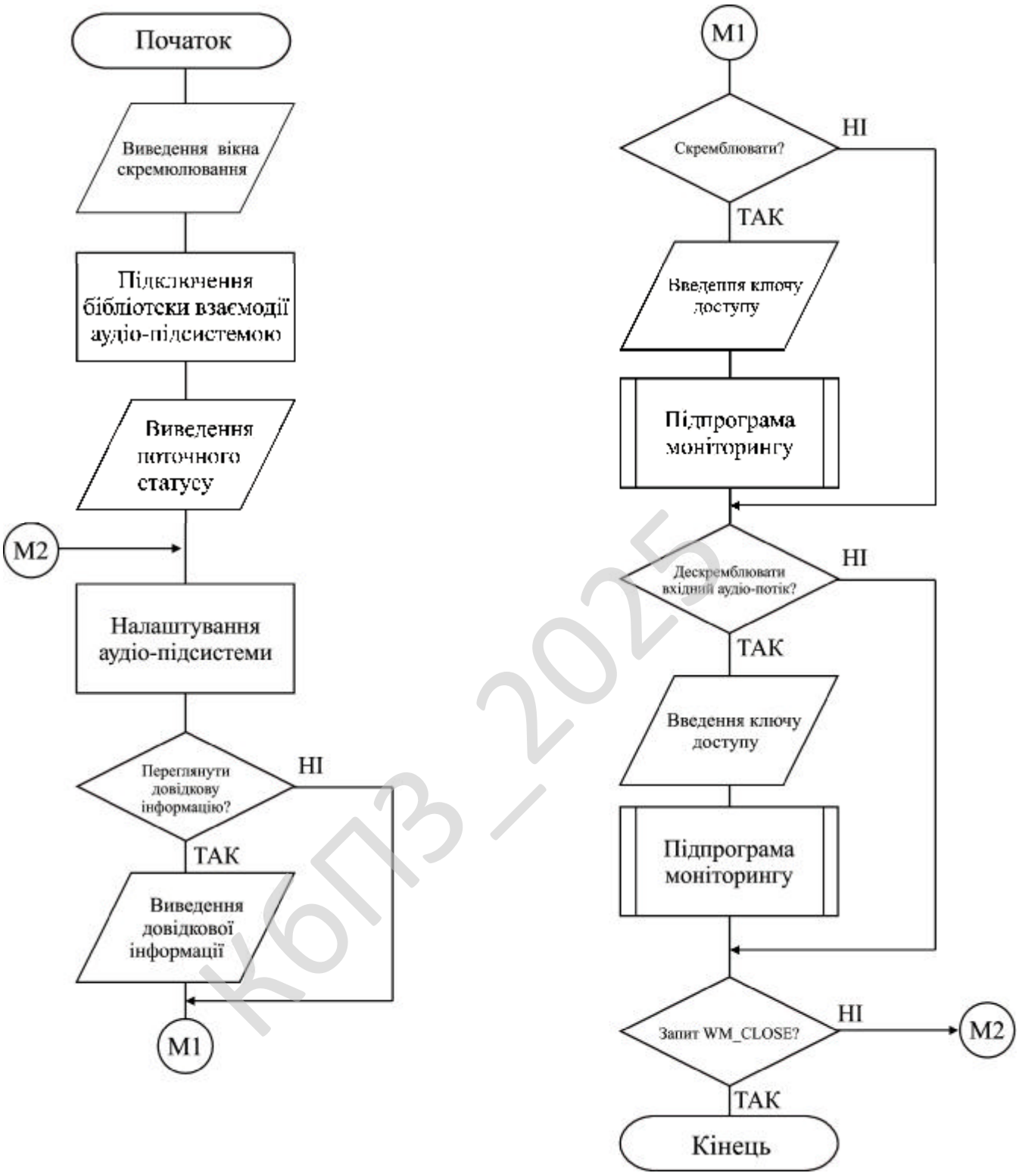

Рисунок 4.1 – Блок-схема основної програми

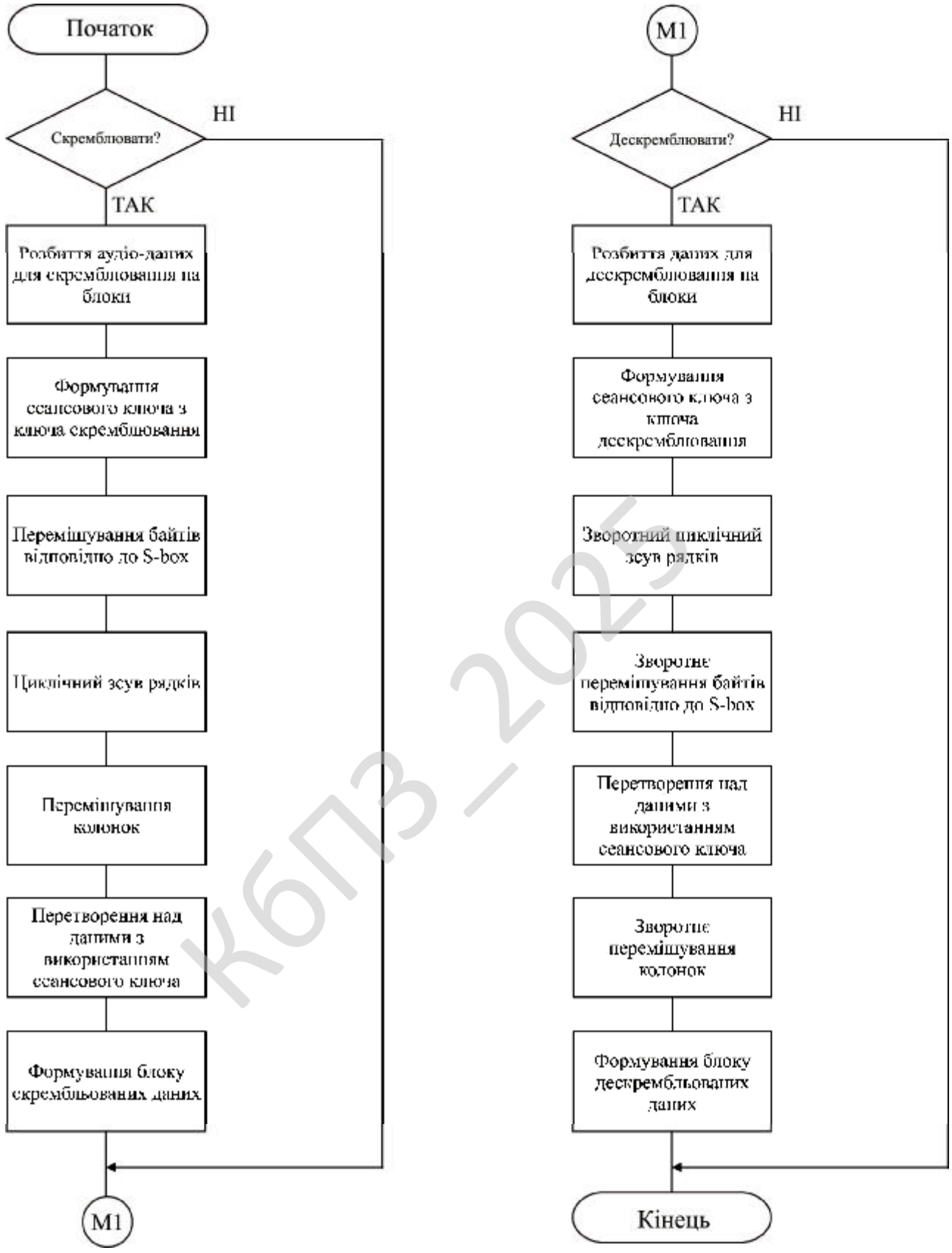

Рисунок 4.2 – Блок-схема роботи підпрограми

Діаграми дають можливість представити систему (як ділову, так і програмну) у такому вигляді, щоб її можна було легко перевести в програмний код. Основною причиною використання мови UML є спілкування розробників між собою.

Крім того, UML спеціально створювалася для оптимізації процесу розробки програмних систем, що дозволяє збільшити ефективність їх реалізації у кілька разів і помітно поліпшити якість кінцевого продукту.

UML прекрасно зарекомендувала себе в багатьох успішних програмних проектах. Засоби автоматичної генерації кодів дозволяють перетворювати моделі мовою UML у вихідний код об’єктно-орієнтованих мов програмування, що ще більш прискорює процес розробки. Практично усі CASE-засоби (програми автоматизації процесу аналізу і проектування) мають підтримку UML. Моделі розроблені в UML, дозволяють значно спростити процес кодування і направити зусилля програмістів безпосередньо на реалізацію системи.

Діаграми підвищують супроводжуваність проекту і полегшують розробку документації.

UML необхідний:

– Керівникам проектів, які керують розподілом завдань і контролем за проектом.

– Проектувальникам інформаційних систем які розробляють технічні завдання для програмістів.

– Бізнес-аналітикам, які досліджують реальну систему і здійснюють інжиніринг і реінжиніринг бізнесу компанії.

– Програмістам які реалізовують модулі інформаційної системи.

При модифікації системи об'єктний підхід дозволяє легко включати в систему нові об'єкти і виключати застарілі без істотної зміни її життєздатності. Використання побудованої моделі при модифікаціях системи дає можливість усунути небажані наслідки змін, оскільки вони не ламають структури системи, а тільки змінюють поведінку об'єктів.

Також при розробці магістерської дипломної роботи було використано наступні підходи UML: діаграма діяльності (діаграми поведінки типу); діаграма прецедентів (діаграми поведінки типу); Діаграма класів; Діаграма компонент; Діаграма об'єктів; Діаграма розгортання.

Діаграма діяльності. Це візуальне представлення графу діяльностей. Граф діяльностей є різновидом графу станів скінченного автомату, вершинами якого є певні дії, а переходи відбуваються по завершенню дій. Дія є фундаментальною одиницею визначення поведінки в специфікації. Дія отримує множину вхідних сигналів, та перетворює їх на множину вихідних сигналів.

Одна із цих множин, або обидві водночас, можуть бути порожніми. Виконання дії відповідає виконанню окремої дії. Подібно до цього, виконання діяльності є виконанням окремої діяльності, буквально, включно із виконанням тих дій, що містяться в діяльності. Кожна дія в діяльності може виконуватись один, два, або більше разів під час одного виконання діяльності. Щонайменше, дії мають отримувати дані, перетворювати їх та тестувати, деякі дії можуть вимагати певної послідовності.

Специфікація діяльності (на вищих рівнях сумісності) може дозволяти виконання декількох (логічних) потоків, та існування механізмів синхронізації для гарантування виконання дій у правильному порядку.

Діаграма прецедентів це діаграма, на якій зображено відношення між акторами та прецедентами в системі. Також, перекладається як діаграма варіантів використання.

Діаграма прецедентів є графом, що складається з множини акторів, прецедентів (варіантів використання) обмежених границею системи (прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання.

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із

системою за допомогою так званих варіантів використання. Варіант використання (use case) використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором.

При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

У мові UML є кілька стандартних видів відношень між акторами і варіантами використання:

– асоціації (association relationship);

– включення (include relationship);

– розширення (extend relationship);

– узагальнення (generalization relationship).

При цьому загальні властивості варіантів використання можуть бути представлені трьома різними способами, а саме – за допомогою відношень включення, розширення і узагальнення.

Відношення асоціації – одне з фундаментальних понять у мові UML і в тій чи іншій мірі використовується при побудові всіх графічних моделей систем у формі канонічних діаграм.

Включення (include) у мові UML – це різновид відношення залежності між базовим варіантом використання і його спеціальним випадком. При цьому відношенням залежності (dependency) є таке відношення між двома елементами моделі, при якому зміна одного елемента (незалежного) приводить до зміни іншого елемента (залежного).

Відношення розширення (extend) визначає взаємозв'язок базового варіанта використання з іншим варіантом використання, функціональна поведінка якого задіюється базовим не завжди, а тільки при виконанні додаткових умов.

Діаграма класів це статичне представлення структури моделі. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення.

Діаграма класів, також, може містити позначення для пакетів та може містити позначення для вкладених пакетів. Також, діаграма класів може містити позначення деяких елементів поведінки, однак їх динаміка розкривається в інших типах діаграм.

Діаграма класів (class diagram) служить для представлення статичної структури моделі системи в термінології класів об'єктно-орієнтованого програмування. На цій діаграмі показують класи, інтерфейси, об'єкти й кооперації, а також їхні відносини.

В UML існують наступні типи зв'язків які використовуються у діаграмі класів: Асоціації; Агрегація; Композиція.

Асоціації це якщо між двома класами визначена асоціація, то можна переміщатися від об'єктів одного класу до об'єктів іншого. Цілком припустимі випадки, коли обидва кінці асоціації відносяться до одного і того ж класу. Це означає, що з об'єктом деякого класу дозволено зв'язати інші об'єкти з того ж класу. Асоціація, що зв'язує два класи, називається бінарної. Можна, хоча це рідко буває необхідним, створювати асоціації, що зв'язують відразу кілька класів. Графічно асоціація зображується у вигляді лінії, що з'єднує клас сам з собою або з іншими класами.

Асоціації може бути присвоєно ім'я, яке описує природу відносини. Зазвичай ім'я асоціації не вказується, якщо тільки ви не хочете явно задати для неї рольові імена або у вашій моделі настільки багато асоціацій, що виникає необхідність посилатися на них і відрізняти один від одного. Ім'я буде особливо корисним, якщо між одними і тими ж класами існує кілька різних асоціацій.

Клас, що бере участь в асоціації, грає в ній деяку роль. По суті, це "обличчя", яким клас, що знаходиться на одній стороні асоціації, звернений до класу з іншого її боку. Можна явно позначити роль, яку клас грає в асоціації.

Часто при моделюванні буває важливо вказати, скільки об'єктів може бути пов'язано допомогою одного примірника асоціації. Це число називається

кратністю (Multiplicity) ролі асоціації та записується або як вираз, значенням якого є діапазон значень, або в явному вигляді.

Вказуючи кратність на одному кінці асоціації, ви тим самим говорите, що на цьому кінці саме стільки об'єктів повинно відповідати кожному об'єкту на протилежному кінці. Кратність можна задати рівною одиниці (1), можна вказати діапазон: "нуль або одиниця" (0..1), "багато" (0 .. *), "одиниця або більше" (1 .. *). Дозволяється також вказувати певне число (наприклад, 3). За допомогою списку можна задати і більш складні кратності, наприклад 0. . 1, 3..4, 6 .. *, що означає "будь-яке число об'єктів, крім 2 і 5".

Агрегація це проста асоціація між двома класами відображає структурний відношення між рівноправними сутностями, коли обидва класу знаходяться на одному концептуальному рівні і ні один не є більш важливим, ніж інший. Але іноді доводиться моделювати відношення типу «частина/ціле», в якому один з класів має більш високий ранг (ціле) і складається з декількох менших за рангом (частин).

Ставлення такого типу називають агрегацією; воно зараховане до відносин типу «має» (з урахуванням того, що об'єкт-ціле має кілька об'єктів-частин). Агрегація є окремим випадком асоціації і зображується у вигляді простої асоціації з незафарбованим ромбом з боку «цілого». Графічно агрегація представляється порожнім ромбом на блоці класу, і лінією, яка від цього ромба до міститься класу.

Композиція це більш суворий варіант агрегації. Відома також як агрегація за значенням.

Композиція має жорстку залежність часу існування екземплярів класу контейнера та примірників містяться класів. Якщо контейнер буде знищений, то весь його вміст буде також знищено. Графічно представляється як і агрегація, але з зафарбовані ромбиком.

Діаграма компонент в UML це діаграма, на якій відображаються компоненти, залежності та зв'язки між ними.

Діаграма компонент відображає залежності між компонентами програмного забезпечення, включаючи компоненти вихідних кодів, бінарні компоненти, та компоненти, що можуть виконуватись.

Модуль програмного забезпечення може бути представлено в якості компоненти. Деякі компоненти існують під час компіляції, деякі – під час компонування, а деякі під час роботи програми.

Діаграма компонент відображає лише структурні характеристики, для відображення окремих екземплярів компонент слід використовувати діаграму розгортування.

Компоненти об'єднуються разом використовуючи структурні зв'язки (assembly connector) щоб об'єднати інтерфейси двох компонент. Це ілюструє зв'язок типу «клієнт-сервер».

Структурна взаємодія – «зв'язок двох компонент, який передбачає, що один з них надає послуги, потрібні іншому компоненту».

При використанні діаграми компонент щоб показати внутрішню структуру компонента, клієнтські та серверні інтерфейси можуть утворювати пряме з'єднання з внутрішніми. Таке з'єднання називається з'єднанням делегації.

Діаграма об'єктів в UML це діаграма, що відображає об'єкти та їх зв'язки в певний момент часу. Діаграма об'єктів може розглядатись як окремий випадок діаграми класів, на якій можуть бути представлені як класи, так і екземпляри (об'єкти) класів. Схожою за змістом є діаграма взаємодії (collaboration diagram).

Діаграми об'єктів не мають власної нотації. Оскільки діаграми класів можуть відображати об'єкти, то діаграма класів, на якій відображено лише об'єкти, та не відображено класи, може вважатись діаграмою об'єктів.

Діаграма об'єктів відображає об'єкти та зв'язки в певний момент роботи програми. Об'єкти можуть містити інформацію про власні значення а не про описання. Для відображення загальних шаблонів об'єктів та зв'язків, що можуть багаторазово створюватись під час роботи програми, слід

використовувати діаграму взаємодії, яка може відображати характеристики об'єктів та зв'язків. Екземпляр діаграми взаємодії створює діаграму об'єктів.

Діаграма об'єктів не відображає еволюцію системи під час роботи. Натомість, слід використовувати діаграми взаємодії з повідомленнями, або діаграми послідовності.

Діаграма розгортання (deployment diagram) це діаграма в UML, на якій відображаються обчислювальні вузли під час роботи програми, компоненти, та об'єкти, що виконуються на цих вузлах. Компоненти відповідають представленню робочих екземплярів одиниць коду. Компоненти, що не мають представлення під час роботи програми на таких діаграмах не відображаються; натомість, їх можна відобразити на діаграмах компонент. Діаграма розгортання відображає робочі екземпляри компонент, а діаграма компонент, натомість, відображає зв'язки між типами компонент.

Розроблювана система скремблювання цифрового сигналу на мобільних пристроях виконує програмне перетворення бітового потоку для підвищення конфіденційності та зниження кореляції між сусідніми бітами.

Система працює на стороні мобільного пристрою як прототип логіки, яку у подальшому можна переносити в мобільні застосунки. У пояснювальній записці вихідний код на мові Python виконує роль еталонної реалізації алгоритмів і демонструє послідовність обробки сигналу.

Система працює з узагальненим цифровим сигналом у вигляді байтового потоку. Джерелом сигналу можуть бути аудіодані, відеофрагменти, телеметрія або текстові повідомлення, які вже перетворені у послідовність байтів. У всіх випадках ядро скремблювання працює з бітами, а не з високорівневими структурами. Це спрощує аналіз та дозволяє переносити алгоритм у інші мови програмування.

Архітектура системи

Архітектура системи складається з кількох логічних компонентів, які у вихідному коді реалізуються у вигляді класів і допоміжних функцій.

Перший компонент це клас конфігурації ScramblerConfig. Він зберігає параметри, які задає розробник або користувач. До них належить розмір регістра зсуву для генератора псевдовипадкової послідовності, набір відводів полінома, початкове значення регістра, розмір кадру у бітах та ймовірність бітової помилки у каналі мобільного звязку. Таке групування параметрів у єдину структуру спрощує зміну режимів роботи системи та уніфікує ініціалізацію всіх модулів.

Другий компонент це клас LFSR, який реалізує лінійний регістр зсуву з лінійним зворотним звязком. Саме цей клас генерує псевдовипадкову бінарну послідовність, що використовується для скремблювання сигналу. Алгоритм працює з цілим числом, яке у двійковому записі відповідає вмісту регістра. На кожному кроці функція next_bit обчислює біт зворотного звязку на основі зазначених відводів полінома, зсуває регістр і повертає вихідний біт. Правильний добір полінома забезпечує максимальний період послідовності, що наближається до 2^n мінус одиниця, де n розмір регістра. У програмі використовується поліном для шістнадцятибітного регістра. Така конфігурація дає період більше шістдесяти пяти тисяч бітів, що для більшості навчальних сценаріїв є достатнім.

Третій компонент це конвертер бітового потоку BitStreamConverter. Він перетворює байти у список нулів та одиниць і назад. Функція bytes_to_bits послідовно обробляє кожен байт і виділяє біти починаючи зі старшого. Функція bits_to_bytes виконує зворотне перетворення, накопичує біти у регістрі на вісім позицій і формує байт. Такий підхід забезпечує однозначність перетворення, що є критично важливим для правильного відновлення сигналу після скремблювання і дескремблювання.

Четвертий компонент це клас KeyManager. У мобільному контексті користувач вводить пароль або ключову фразу, а система перетворює її у числове значення, яке використовується як початковий стан LFSR. KeyManager виконує просте хешування ключової фрази у межах розміру регістра, що дозволяє задавати різні ключі без прямого введення двійкового стану регістра. Це робить систему більш дружньою до користувача і спрощує інтеграцію у мобільний

застосунок.

П’ятий компонент це клас SignalScrambler. Цей клас безпосередньо відповідає за скремблювання та дескремблювання цифрового сигналу. Внутрішній метод створює екземпляр LFSR на основі поточних параметрів конфігурації і генерує стільки псевдовипадкових бітів, скільки містить оброблюваний кадр даних. Функція scramble_bits виконує поелементну операцію виключного АБО між бітами сигналу і бітами псевдовипадкової послідовності. Функція descramble_bits застосовує той самий алгоритм, оскільки двократне застосування операції виключного АБО з однаковою послідовністю повертає початкові дані. Такий підхід робить відправника і отримувача симетричними, що відповідає практиці цифрових систем звязку.

Шостий компонент це клас MobileChannelSimulator. Він моделює вплив мобільного радіоканалу на переданий цифровий сигнал. Модель враховує випадкові бітові помилки, які виникають під час передачі. Для кожного біта використовується задана ймовірність інверсії. Це дозволяє оцінити, як скремблювання впливає на характеристики сигналу при наявності шуму і завад, що важливо для обгрунтування доцільності використання алгоритму у реальних мобільних системах.

Сьомий компонент це клас QualityMetrics. Він обчислює показники якості роботи системи. У програмі реалізуються два показники. Перший це бітова ймовірність помилки, яка визначається як відношення кількості відмінних бітів між оригінальним і прийнятим сигналом до загальної кількості бітів. Другий це оцінка бінарної ентропії послідовності. Для цього підраховується кількість нулів і одиниць і обчислюється ентропія Шеннона. Ентропія для випадкової послідовності прагне до одиниці. Порівняння ентропії вхідного і скрембльованого сигналу демонструє, наскільки розподіл бітів наближається до рівномірного, що служить непрямим підтвердженням ефективності скремблювання.

Восьмий компонент це функція run_demo, яка організовує типовий сценарій роботи системи для пояснювальної записки. Функція створює тестовий

повідомлення у байтовому вигляді, перетворює його у біти, виконує скремблювання, моделює передачу через зашумлений канал, виконує дескремблювання і оцінює якість відновленого сигналу. У результаті система виводить довжину сигналу, оцінку ентропії до і після скремблювання, а також бітову ймовірність помилки до і після дескремблювання. Для невеликих значень ймовірності бітової помилки система показує незначний рівень спотворень. При нульовій ймовірності помилки бітові послідовності до і після дескремблювання збігаються, що підтверджує коректність реалізації алгоритму.

Девятий компонент це точка входу програми. Якщо файл запускається безпосередньо, система створює конфігурацію з типовими параметрами, формує демонстраційний текстовий сигнал і викликає функцію run_demo. У реальному мобільному застосунку ці дії виконуються іншими модулями, які отримують байти від мікрофона, камери або мережевого стеку і передають їх у ядро скремблювання. Завдяки чіткій модульній структурі Python реалізація легко переноситься у інші мови, зокрема у мови, що використовуються у розробці мобільних систем.

Прийняті проектні рішення

Використання лінійного регістра зсуву як джерела псевдовипадкової послідовності пояснюється простотою реалізації і невеликими обчислювальними витратами, що важливо для мобільних пристроїв.

Операція виключного АБО є дуже швидкою за будь якого сучасного процесора, тому система не створює значного навантаження навіть при обробці довгих потоків даних.

Розмір регістра у шістнадцять бітів забезпечує період послідовності близько шістдесяти пяти тисяч бітів.

Це достатньо для скремблювання типового кадру даних мобільної системи. При потребі конфігурацію можна змінити і використати регістр більшого розміру для збільшення періоду.

Оцінка ентропії скрембльованого сигналу демонструє, що після

скремблювання розподіл нулів і одиниць стає ближчим до рівномірного, а значення ентропії наближається до одиниці. Це означає, що регулярні структури вхідного сигналу маскуються, а статистичний аналіз скрембльованих даних ускладнюється.

Одночасно система зберігає симетричність алгоритму та простоту реалізації дескремблювання, що є важливою перевагою для мобільних клієнтів.

Завдяки виділенню окремих класів для генератора, конвертера, скремблера, моделі каналу та метрик система залишається розширюваною. За потреби можна додати більш складний тип генератора, наприклад з використанням криптографічного алгоритму, зберігаючи той самий інтерфейс для скремблера. Такий підхід відповідає вимогам магістерської випускної кваліфікаційної роботи, оскільки демонструє як алгоритмічні, так і інженерні аспекти розробки програмної системи скремблювання цифрового сигналу на мобільних пристроях.

```
from dataclasses import dataclass
from typing import List, Iterable, Tuple
import random
import math
# Клас ScramblerConfig зберігає параметри конфігурації системи скремблювання
@dataclass
class ScramblerConfig:
    register_size: int = 16
    taps: Tuple[int, ...] = (16, 14, 13, 11)
    seed: int = 0xACE1
    frame_size_bits: int = 1024
    bit_error_probability: float = 0.0
# Клас LFSR реалізує лінійний регістр зсуву з лінійним зворотним звязком
class LFSR:
    def __init__(self, size: int, taps: Tuple[int, ...], seed: int) -> None:
        self.size = size
        self.taps = taps
        self.mask = (1 << size) - 1
        self.state = seed & self.mask
        if self.state == 0:
            self.state = 1
    def next_bit(self) -> int:
```

```
        xor_bit = 0
        for tap in self.taps:
            bit = (self.state >> (tap - 1)) & 1
            xor_bit ^= bit
        output_bit = self.state & 1
        self.state >>= 1
        self.state |= (xor_bit << (self.size - 1))
        self.state &= self.mask
        return output_bit
    def sequence(self, length: int) -> List[int]:
        bits: List[int] = []
        for _ in range(length):
            bits.append(self.next_bit())
        return bits
# Клас BitStreamConverter виконує перетворення між байтами та бітами
class BitStreamConverter:
    @staticmethod
    def bytes_to_bits(data: bytes) -> List[int]:
        bits: List[int] = []
        for byte in data:
            for i in range(7, -1, -1):
                bit = (byte >> i) & 1
                bits.append(bit)
        return bits
    @staticmethod
    def bits_to_bytes(bits: Iterable[int]) -> bytes:
        result = bytearray()
        accumulator = 0
        count = 0
        for bit in bits:
            accumulator = (accumulator << 1) | (bit & 1)
            count += 1
            if count == 8:
                result.append(accumulator)
                accumulator = 0
                count = 0
        if count > 0:
            accumulator <<= (8 - count)
            result.append(accumulator)
        return bytes(result)
# Клас KeyManager перетворює текстовий ключ у початковий стан регістра
class KeyManager:
```

```
    @staticmethod
    def passphrase_to_seed(passphrase: str, register_size: int) -> int:
        value = 0
        modulus = (1 << register_size) - 1
        for byte in passphrase.encode("utf-8"):
            value = (value * 31 + byte) & modulus
        if value == 0:
            value = 1
        return value
# Клас SignalScrambler реалізує скремблювання та дескремблювання бітового потоку
class SignalScrambler:
    def __init__(self, config: ScramblerConfig) -> None:
        self.config = config

    def _make_generator(self) -> LFSR:
        return LFSR(
            size=self.config.register_size,
            taps=self.config.taps,
            seed=self.config.seed,
        )
    def scramble_bits(self, bits: List[int]) -> List[int]:
        generator = self._make_generator()
        mask_bits = generator.sequence(len(bits))
        scrambled: List[int] = []
        for index, bit in enumerate(bits):
            scrambled_bit = bit ^ mask_bits[index]
            scrambled.append(scrambled_bit)
        return scrambled
    def descramble_bits(self, bits: List[int]) -> List[int]:
        return self.scramble_bits(bits)
# Клас MobileChannelSimulator моделює вплив мобільного каналу на бітовий потік
class MobileChannelSimulator:
    def __init__(self, bit_error_probability: float) -> None:
        self.bit_error_probability = bit_error_probability
    def transmit(self, bits: List[int]) -> List[int]:
        transmitted: List[int] = []
        for bit in bits:
            if random.random() < self.bit_error_probability:
                transmitted.append(1 - bit)
            else:
                transmitted.append(bit)
        return transmitted
```

```
# Клас QualityMetrics обчислює показники якості роботи системи
class QualityMetrics:
    @staticmethod
    def bit_error_rate(original: List[int], received: List[int]) -> float:
        length = min(len(original), len(received))
        if length == 0:
            return 0.0
        errors = 0
        for index in range(length):
            if original[index] != received[index]:
                errors += 1
        return errors / float(length)
    @staticmethod
    def estimate_binary_entropy(bits: List[int]) -> float:
        if not bits:
            return 0.0
        ones = sum(bits)
        zeros = len(bits) - ones
        entropy = 0.0
        for count in (zeros, ones):
            if count > 0:
                probability = count / float(len(bits))
                entropy -= probability * math.log2(probability)
        return entropy
# Функція run_demo демонструє послідовність роботи системи скремблювання
def run_demo() -> None:
    config = ScramblerConfig()
    message_text = (
        "Це демонстраційний приклад цифрового сигналу "
        "для системи скремблювання на мобільному пристрої"
    )
    payload = message_text.encode("utf-8")
    original_bits = BitStreamConverter.bytes_to_bits(payload)
    scrambler = SignalScrambler(config)
    scrambled_bits = scrambler.scramble_bits(original_bits)
    channel = MobileChannelSimulator(config.bit_error_probability)
    channel_bits = channel.transmit(scrambled_bits)
    descrambled_bits = scrambler.descramble_bits(channel_bits)
    recovered_bytes = BitStreamConverter.bits_to_bytes(descrambled_bits)
    ber_before = QualityMetrics.bit_error_rate(original_bits, channel_bits)
    ber_after = QualityMetrics.bit_error_rate(original_bits, descrambled_bits)
    entropy_original = QualityMetrics.estimate_binary_entropy(original_bits)
```

```
entropy_scrambled = QualityMetrics.estimate_binary_entropy(scrambled_bits)
print("Довжина повідомлення у бітах", len(original_bits))
print("Ентропія вхідного сигналу", round(entropy_original, 4))
print("Ентропія скрембльованого сигналу", round(entropy_scrambled, 4))
print("Бітова ймовірність помилки до дескремблювання", round(ber_before, 6))
print("Бітова ймовірність помилки після дескремблювання", round(ber_after, 6))
try:
    recovered_text = recovered_bytes.decode("utf-8")
except UnicodeDecodeError:
    recovered_text = "<не вдалося декодувати>"
print("Відновлене повідомлення")
print(recovered_text)
```

Розглянувши блок-схему програми та підпрограми шифрування дешифрування, перейдемо до розділу захисту розробленого програмного забезпечення.

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використати фіналіста конкурсу AES – шифр Rijndael. Він є нетрадиційним блоковим шифром, оскільки не використовує мережу Фейштеля для криптоперетворень. Алгоритм представляє кожний блок кодуємих даних у вигляді двовимірного масиву байт розміром 4x4, 4x6 або 4x8 залежно від установленої довжини блоку. Далі на відповідних етапах перетворення відбуваються або над незалежними стовпцями, або над незалежними рядками, або взагалі над окремими байтами в таблиці.

Всі перетворення в шифрі мають строге математичне обґрунтування. Сама структура й послідовність операцій дозволяють виконувати даний алгоритм ефективно як на 16-бітних так і на 64-бітних процесорах. У структурі алгоритму закладена можливість паралельного виконання деяких операцій, що на багатопроцесорних робочих станціях може ще підняти швидкість шифрування в 4 рази.

Алгоритм складається з деякої кількості раундів (від 10 до 14 – це залежить від розміру блоку й довжини ключа), у яких послідовно виконуються наступні операції:

ByteSub – Таблична підстановка 8x8 біт (рисунок 4.3).

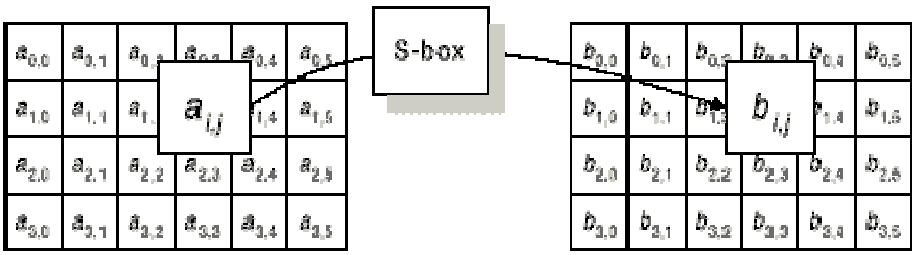

Рисунок 4.3 – Таблична підстановка 8x8 біт

ShiftRow – зрушення рядків у двовимірному масиві на різні зсуви (рисунок 4.4).

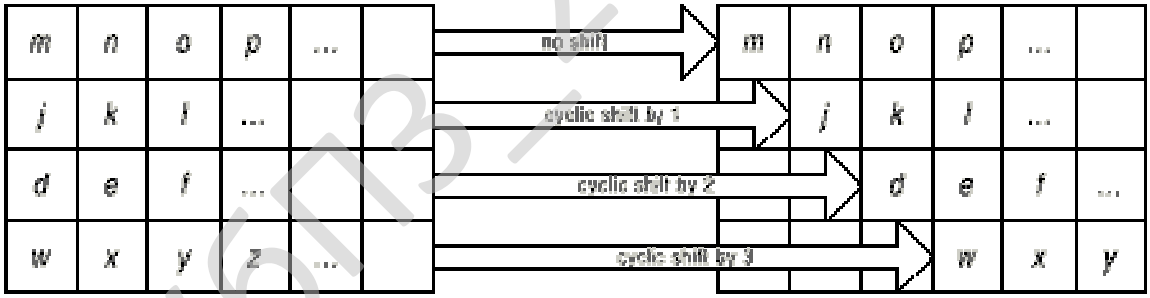

Рисунок 4.4 – Зрушення рядків у двовимірному масиві на різні зсуви

MixColumn – математичне перетворення, що перемішує дані усередині стовпця (рисунок 4.5).

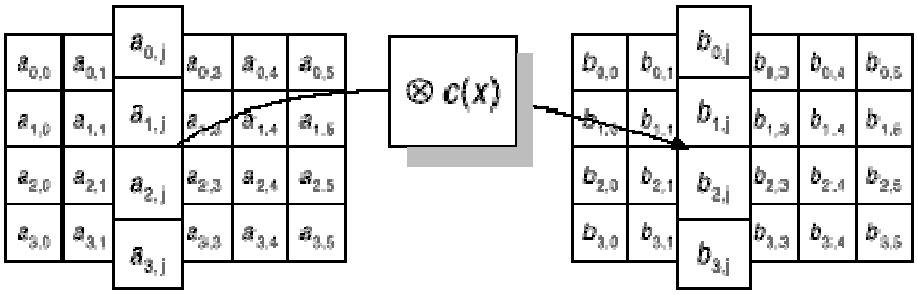

Рисунок 4.5 – Математичне перетворення, що перемішує дані усередині стовпця

AddRoundKey – додавання матеріалу ключа операцією XOR (рисунок 4.6).

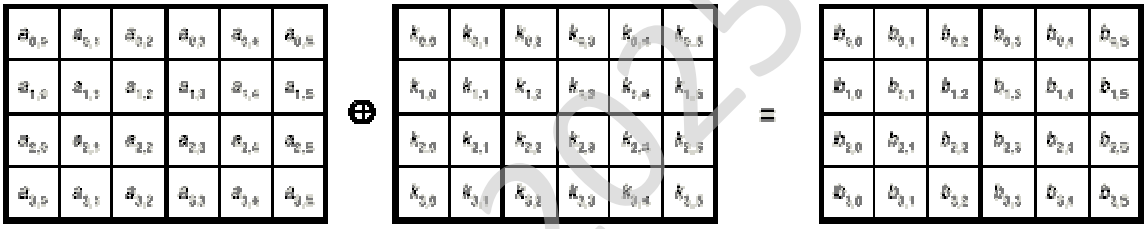

Рисунок 4.6 – Додавання матеріалу ключа операцією XOR

В останньому раунді операція перемішування стовпців відсутня, що робить всю послідовність операцій симетричною.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображено інтерфейс програмного забезпечення, розробленого у результаті виконання магістерської дипломної роботи. Розроблене програмне забезпечення системи скремблювання цифрового сигналу на мобільних пристроях складається з наступних функціональних блоків:

– Блоку графічного представлення аудіо-потоку.

– Вікна обрання групи .

– Вікно виведення результату роботи системи.

– Функціональних кнопок ПЗ: формування сеансового ключа; параметри скремблювання; налаштування аудіо-потоку; налаштування роботи системи.

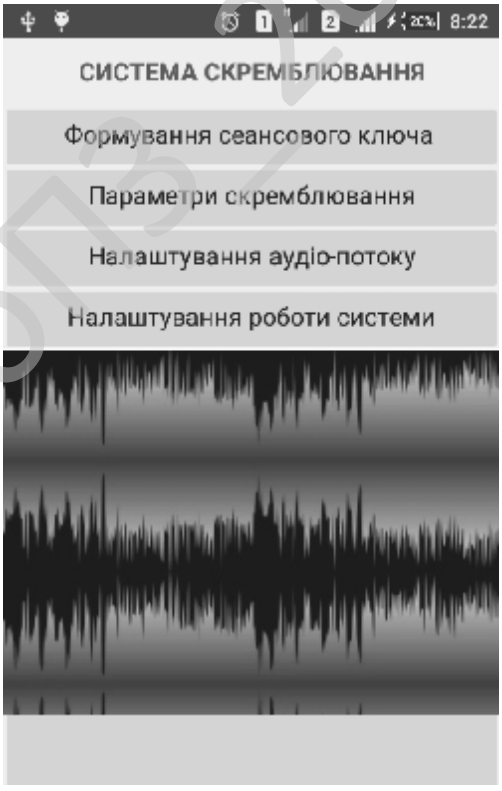

Рисунок 5.1 – Головне вікно розробленого ПЗ

Для перегляду короткої довідки про програму слід натиснути на основному вікні кнопку авторського права, після чого на екрані з’явиться вікно показане на рисунку 5.2.

Рисунок 5.2 – Вікно розробника ПЗ

Під час роботи над програмою було проведено тестування програмного забезпечення, тобто технічне дослідження, призначене для виявлення інформації про якість продукту відносно контексту, в якому воно має використовуватись.

Тестування включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою їх оцінки.

Проводилась оцінка:

– відповідності поставленим вимогам;

– правильна відповідь для усіх можливих вхідних даних;

– виконання функцій за прийнятний час;

– практичність;

– сумісність з ОС та стороннім ПЗ.

Оскільки число можливих тестів для програмних компонент практично нескінченне, тому стратегія тестування полягала в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів.

Як результат ПЗ тестувалось стандартним виконанням програми з метою виявлення помилок або інших дефектів.

Проводилось тестування форматом білої скриньки та чорної скриньки. Тестування форматом білої скриньки засноване на аналізі керуючої структури програми. Програма вважається повністю перевіреною, якщо проведено вичерпне тестування маршрутів (шляхів) її графа управління.

У цьому випадку формуються тестові варіанти, в яких:

– Гарантується перевірка всіх незалежних маршрутів програми.

– Знаходяться гілки True, False для всіх логічних рішень.

– Виконуються всі цикли (у межах їхніх кордонів та діапазонів).

– Аналізується правильність внутрішніх структур даних.

Недоліки тестування "білої скриньки":

– Кількість незалежних маршрутів може бути дуже велика.

– Повне тестування маршрутів не гарантує відповідності програми вихідним вимогам до неї.

– У програмі можуть бути пропущені деякі маршрути.

– Не можна виявити помилки, поява яких залежить від даних.

Переваги тестування "білої скриньки" пов'язані з тим, що принцип «білої скриньки» дозволяє врахувати особливості програмних помилок:

– Кількість помилок мінімально в «центрі» і максимально на «периферії» програми.

– Попередні припущення про ймовірність потоку керування або даних у програмі часто бувають некоректними. У результаті типовим може стати маршрут, модель обчислень за яким опрацьована слабо.

– При записі алгоритму програмного забезпечення у вигляді тексту на мові програмування можливе внесення типових помилок трансляції (синтаксичних та семантичних).

– Деякі результати в програмі залежать не від вихідних даних, а від внутрішніх станів програми.

Проводилось тестування чорної скриньки.

Основне місце програми тестів «чорної скриньки» – інтерфейс ПЗ. Відомі: функції програми. Досліджується: робота кожної функції на всій області визначення.

Ці тести демонструють:

– Як виконуються функції програми.

– Як приймаються вихідні дані.

– Як виробляються результати.

– Як зберігається цілісність зовнішньої інформації.

При тестуванні «чорної скриньки» розглядаються системні характеристики програм, ігнорується їхня внутрішня логічна структура. Вичерпне тестування, як правило, неможливе.

Наприклад, якщо в програмі 10 вхідних величин і кожна приймає по 10 значень, то кількість тестових варіантів становитиме 10^10. Тестування «чорної скриньки» не реагує на багато особливостей програмних помилок.

Тестування «чорної скриньки» (функціональне тестування) дозволяє отримати комбінації вхідних даних, які забезпечують повну перевірку всіх функціональних вимог до програми.

Програмний виріб тут розглядається як «чорна скринька», чию поведінку можна визначити тільки дослідженням його входів та відповідних виходів. При такому підході бажано мати:

– Набір, утворений такими вхідними даними, які призводять до аномалій у поведінці програми (назвемо його ITc).

– Набір, утворений такими вхідними даними, які демонструють дефекти програми (назвемо його OT).

Будь-який спосіб тестування «чорної скриньки» повинен:

– Виявити такі вхідні дані, які з високою ймовірністю належать набору ITc;

– Сформулювати такі очікувані результати, які з високою імовірністю є елементами набору OT.

Принцип «чорної скриньки» не альтернативний принципу «білої скриньки». Скоріше це доповнює підхід, який виявляє інший клас помилок.

Тестування «чорної скриньки» забезпечує пошук наступних категорій помилок:

– Некоректних чи відсутніх функцій;

– Помилок інтерфейсу;

– Помилок у зовнішніх структурах даних або в доступі до зовнішньої бази даних;

– Помилок характеристик (необхідна ємність пам'яті і т.д.);

– Помилок ініціалізації та завершення.

Обрано умови розповсюдження – Shareware.

Під умовно-безплатним програмним забезпеченням можна розуміти спосіб або метод розповсюдження комерційного ПЗ на ринку (тобто на шляху до кінцевого користувача), при якому випробувачеві пропонується обмежена за можливостями (не повнофункціональна або демонстраційна версія), терміном дії (тріал версія) або версія з вбудованим набридливим нагадуванням про необхідність оплати використання програми.

В угоді про використання (ліцензії для кінцевого користувача, EULA) також може бути обумовлена заборона на комерційне або професійне (не тестове) її використання.

Основний принцип умовно-безплатного ПЗ – «спробуй, перш ніж купити» (try before you buy). ПЗ що поширюється як умовно-безплатний, надається користувачам безоплатно. Звичайно користувач платить тільки за час завантаження файлів через Інтернет або за носій (CD диск, флешку, ключ). Протягом певного терміну, що становить зазвичай тридцять днів, він може користуватися програмою, тестувати її, освоювати її можливості.

Якщо після закінчення цього терміну користувач вирішить продовжити використання ПЗ, він зобов'язаний купити його (зареєструватися), заплативши авторові певну суму.

В іншому випадку користувач повинен припинити використання ПЗ та видалити його зі свого комп'ютера.

6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи скремблювання цифрового сигналу на мобільних пристроях.

Метою розробки є дослідження та програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях.

Об'єктом дослідження є процес скремблювання цифрового сигналу на мобільних пристроях.

Предметом дослідження є методи скремблювання цифрового сигналу на мобільних пристроях.

Методи дослідження базуються на методах теорії сигналів та теорії захисту інформації в мережі, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод скремблювання цифрового сигналу на мобільних пристроях.

– Розроблено вітчизняний продукт скремблювання цифрового сигналу на мобільних пристроях, який має більш широкі можливості, на відміну від існуючих аналогів.

7 МАРКЕТИНГОВЕ ТА ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ ІТ-ПРОЄКТУ

7.1 Визначення цільової аудиторії кінцевого готового продукту

Результати дослідження та програмної реалізації системи скремблювання цифрового сигналу на мобільних пристроях можуть бути цікавими компаніям, які працюють у сфері телекомунікацій і надають послуги мобільного зв'язку. Для них впровадження таких систем є важливим інструментом підвищення рівня захисту переданих даних, зменшення ризиків несанкціонованого доступу до каналів зв'язку та забезпечення конфіденційності користувацької інформації.

Особливий інтерес до результатів розробки можуть проявити підприємства, які працюють у сфері інформаційної безпеки та кіберзахисту. Результати дослідження будуть цінними також для державних установ, військових структур і правоохоронних органів, де питання безпечної передачі інформації мають стратегічне значення. У цих сферах необхідно гарантувати, що дані не можуть бути перехоплені або змінені сторонніми особами. В академічному середовищі результати дослідження можуть зацікавити науковців і студентів технічних спеціальностей, які займаються питаннями цифрової обробки сигналів, захисту інформації та бездротових технологій. Крім того, проєкт може становити інтерес для компаній, що займаються розробкою мобільного програмного забезпечення.

Таким чином, результати дослідження та програмної реалізації системи скремблювання цифрового сигналу на мобільних пристроях мають широкий спектр потенційних користувачів – від телекомунікаційних операторів і державних структур до приватних ІТ-компаній і освітніх закладів. Усі вони можуть отримати користь від підвищення безпеки передавання даних, зниження

ризиків кіберзагроз і створення надійної платформи для захищеного цифрового спілкування.

7.2 Оцінка привабливості шляхом застосування методів експертних оцінок

Оцінку привабливості проєкту програмної реалізації системи скремблювання цифрового сигналу на мобільних пристроях доцільно провести за допомогою методу експертних оцінок, який дозволяє об'єктивно визначити перспективність і конкурентоспроможність розробки. Для цього формується група з фахівців у галузях телекомунікацій, кібербезпеки, розробки мобільних додатків і маркетингу цифрових технологій. Кожен експерт аналізує проєкт за певними критеріями – рівень інноваційності, технічна реалізація, економічна ефективність, потреба ринку, потенціал масштабування та складність впровадження. Оцінювання проводиться за десятибальною шкалою, після чого обчислюється середньозважений показник, який характеризує загальну привабливість проєкту.

У процесі оцінювання фахівці, наприклад, визначають, що система має високий рівень технологічної новизни, оскільки здатна здійснювати скремблювання сигналу у реальному часі без значних затримок та втрат якості. За критерієм інноваційності вона може отримати середню оцінку 9 балів. З огляду на технічну реалізацію, яка базується на сучасних алгоритмах цифрової обробки сигналів та адаптованих криптографічних методах, експерти можуть виставити 8,5 бала. Економічна ефективність оцінюється з позиції потенційного комерційного використання – у цьому випадку середній бал становить 8, оскільки система може бути інтегрована в різні мобільні платформи та додатки з відносно низькими витратами на впровадження.

Критерій потреби ринку отримує одну з найвищих оцінок – близько 9 балів, адже у сучасних умовах постійного зростання кіберзагроз і потреби у

захищеному мобільному зв’язку попит на подібні рішення є стабільно високим. Потенціал масштабування оцінюється на рівні 8 балів, оскільки систему можна адаптувати для різних типів пристроїв і корпоративних середовищ. Найнижчу оцінку, наприклад 7,5 бала, може отримати критерій складності впровадження, оскільки інтеграція алгоритмів скремблювання у мобільні платформи вимагає високого рівня оптимізації та тестування для збереження стабільності роботи.

Після обробки результатів експертних оцінок розраховується середньозважений показник привабливості проєкту. Якщо взяти до уваги вагові коефіцієнти для кожного критерію (інноваційність – 0,2; технічна реалізація – 0,2; економічна ефективність – 0,15; потреба ринку – 0,2; масштабованість – 0,15; складність впровадження – 0,1), то підсумковий інтегральний показник становитиме приблизно 8,4 бала з 10 можливих. Такий результат свідчить про високий потенціал продукту як з точки зору ринкової привабливості, так і з технічної перспективи.

Отримане значення означає, що розробка системи скремблювання цифрового сигналу на мобільних пристроях є доцільною для подальшої комерціалізації. Високі оцінки за параметрами інноваційності та актуальності підтверджують, що рішення відповідає сучасним тенденціям у сфері захисту даних. Експерти також наголошують на потенціалі розширення функціональності – зокрема, інтеграції з корпоративними платформами безпеки або застосування в урядових структурах. Таким чином, метод експертних оцінок демонструє, що проєкт має не лише технічну, але й стратегічну цінність, а його впровадження здатне принести значну користь у забезпеченні безпечних комунікацій у цифровому середовищі.

7.3 Вибір методу оцінки вартості ПЗ

Для оцінки вартості програмної реалізації системи скремблювання цифрового сигналу на мобільних пристроях доцільно застосувати витратний

метод, оскільки він дає можливість найбільш точно визначити реальну собівартість створення програмного продукту. Цей метод ґрунтується на підрахунку фактичних витрат, необхідних для розроблення, тестування, впровадження та підтримки системи. До розрахунків включаються витрати на оплату праці розробників, придбання або оренду технічного обладнання, програмні інструменти, витрати на шифрувальні модулі, ліцензії, технічне забезпечення й навчання користувачів. Такий підхід особливо ефективний на етапі створення продукту, коли ще немає сформованої клієнтської бази чи стабільних продажів. Він дозволяє обґрунтовано визначити стартову вартість системи, необхідну для подальшого формування ціни реалізації.

Додатково для підвищення точності доцільно комбінувати витратний метод із ринковим підходом. Це означає порівняння результатів розрахунків із вартістю аналогічних рішень, які вже представлені на ринку програмного забезпечення для мобільної безпеки та шифрування даних. Такий аналіз дозволяє оцінити конкурентоспроможність ціни, визначити потенційний ціновий діапазон і спрогнозувати реакцію цільової аудиторії на пропоновану вартість. Ринковий підхід також допомагає з’ясувати, які додаткові функції можуть підвищити комерційну привабливість продукту без значного збільшення витрат на його розроблення.

На етапі виходу продукту на ринок, коли з’являються реальні дані про продажі, ефективно використовувати також дохідний метод оцінки вартості. Він базується на прогнозуванні майбутніх прибутків, які може забезпечити система після впровадження. Цей підхід враховує потенційну кількість користувачів, обсяг ліцензійних продажів або підписок, а також витрати на технічну підтримку та оновлення. Використання дохідного методу дозволяє визначити термін окупності проєкту, рівень рентабельності інвестицій і очікувану ефективність від комерційної експлуатації.

Таким чином, найдоцільніше застосувати комбінований підхід, у якому основою є витратний метод, доповнений ринковим і дохідним аналізом. Така

стратегія дозволяє отримати повну фінансову картину проєкту, визначити реальну собівартість створення системи, оцінити її ринкову конкурентність і спрогнозувати економічний ефект від подальшої реалізації. Комбінація методів забезпечує не лише точність оцінки, а й гнучкість під час ухвалення управлінських рішень щодо ціноутворення, інвестицій і масштабування програмного продукту.

7.4 Розрахунок економічної ефективності від впровадження реалізованого ПЗ як фактору його привабливості

Підприємство планує впровадити систему скремблювання цифрового сигналу для підвищення рівня захисту мобільних комунікацій між співробітниками. Раніше обмін інформацією відбувався через стандартні мобільні засоби без додаткового шифрування, що створювало ризики витоку конфіденційних даних. Вхідні дані та розрахунки зведемо до таблиці 7.1.

Розрахунок економічного ефекту

1. Зменшення фінансових втрат від інцидентів безпеки:

(6 – 1) × 50 000 = 250 000 грн/рік

2. Економія на технічному обслуговуванні зв’язку:

200 000 – 150 000 = 50 000 грн/рік

3. Додатковий ефект від підвищення продуктивності праці:

Завдяки стабільному та захищеному зв’язку скорочено простої на 10% часу

Економія робочого часу: 100 осіб × 2 год/тиждень × 200 грн × 52 тижні = 208 000 грн/рік

Загальний економічний ефект: зменшення втрат від інцидентів -250 000 грн/рік, економія на обслуговуванні – 50 000 грн/рік, підвищення продуктивності – 208 000 грн/рік, разом економічний ефект -508 000 грн/рік.

Термін окупності (PP) – 350 000 / 508 000 =0,69 року (~8 місяців).

Таблиця 7.1 – Економічна ефективність від реалізації

Показник	**До впровадження**	**Після впровадження**	**Ефект / Економія**
Кількість мобільних користувачів	100 осіб	100 осіб	—
Кількість інцидентів несанкціонованого доступу на рік	6	1	–5
Середні збитки від одного інциденту (витік інформації, простої, відновлення)	50 000 грн	5 000 грн	–45 000 грн
Річні витрати на технічне обслуговування зв’язку	200 000 грн	150 000 грн	–50 000 грн
Вартість розробки та впровадження системи	—	—	350 000 грн

Коефіцієнт економічної ефективності (Е) – (508 000 / 350 000) × 100%=145%

Додаткові (немонетарні) вигоди: підвищення рівня захищеності корпоративної інформації, зниження ризику витоку комерційної та персональної інформації, підвищення довіри партнерів та клієнтів до компанії, створення

основи для подальшої інтеграції із системами корпоративної безпеки, підвищення репутації підприємства як інноваційно орієнтованого.

Впровадження системи скремблювання цифрового сигналу на мобільних пристроях є економічно ефективним і доцільним, оскільки інвестиції окупаються менш ніж за рік, а коефіцієнт економічної ефективності становить понад 140%.

Окрім прямого фінансового ефекту, система забезпечує довгострокові стратегічні переваги – підвищення безпеки, зниження ризиків інформаційних атак і покращення якості внутрішніх комунікацій.

7.5 Пропозиція алгоритму просування проєкту розробки ПЗ

Алгоритм просування проєкту програмної реалізації системи скремблювання цифрового сигналу на мобільних пристроях доцільно розпочати з етапу аналітичної підготовки. На цьому етапі варто провести детальний аналіз ринку мобільних технологій і рішень у сфері інформаційної безпеки, виявити основних конкурентів, оцінити їхні переваги й недоліки, а також дослідити актуальні тенденції у використанні засобів шифрування та захисту даних. Особливо важливо визначити цільові сегменти користувачів, для яких питання конфіденційності комунікацій є критичними, – корпоративний сектор, державні структури, військові організації та приватні користувачі, які потребують захисту персональної інформації. Ретельно проведене аналітичне дослідження стане основою для формування стратегії просування продукту на ринку.

Наступним етапом має бути розроблення позиціонування продукту та формування унікальної торгової пропозиції. Для успішного просування потрібно чітко визначити, які саме переваги система надає користувачеві: наприклад, безперервне шифрування сигналу в реальному часі, відсутність затримок у передачі даних, низьке енергоспоживання або простота інтеграції в наявні мобільні платформи.

Після цього необхідно перейти до етапу комунікації та створення іміджу продукту. Важливо забезпечити інформаційну присутність системи у професійному середовищі, на тематичних форумах, у медіа, що спеціалізуються на інформаційній безпеці та мобільних технологіях.

Наступним кроком має стати розбудова партнерських каналів реалізації. Для цього варто налагодити співпрацю з компаніями, які вже працюють у сфері інформаційної безпеки, постачальниками корпоративних IT-рішень, розробниками мобільних додатків і дистриб'юторами захищеного програмного забезпечення.

Важливим елементом просування має стати цифровий маркетинг. Необхідно створити офіційний сайт продукту з чітким описом функцій, технічних характеристик і переваг. Варто забезпечити активність у професійних спільнотах, просування у соціальних мережах, використання таргетованої реклами для залучення цільової аудиторії.

Завершальним етапом алгоритму є формування системи моніторингу результатів просування та адаптації стратегії. Необхідно постійно аналізувати ефективність маркетингових заходів, рівень зацікавленості клієнтів, кількість завантажень або ліцензійних придбань. На основі зібраних даних можна оперативно вносити зміни до рекламних кампаній, оновлювати продукт, розширювати функціональність або змінювати цінову політику. Такий підхід забезпечить гнучкість стратегії просування, дозволить утримувати конкурентні позиції на ринку й поступово збільшувати комерційний потенціал системи скремблювання цифрового сигналу на мобільних пристроях.

7.6 Оптимізація каналів збуту та шляхів реалізації ПЗ

Оптимізацію каналів збуту проєкту програмної реалізації системи скремблювання цифрового сигналу на мобільних пристроях доцільно розпочати з концентрації на чітко визначених сегментах ринку. Необхідно зосередити увагу

на тих користувачах, для яких безпека комунікацій є пріоритетною – це корпоративні клієнти, державні структури, військові та дипломатичні установи, а також великі підприємства, що працюють із конфіденційними даними. Така сегментація дозволить точніше спрямовувати маркетингові зусилля, скоротити витрати на рекламу і досягти більшої ефективності при формуванні довгострокових партнерських відносин із клієнтами.

Подальша оптимізація може бути досягнута через розширення партнерських каналів реалізації. Співпраця з компаніями, що надають послуги у сфері мобільної безпеки, інтеграторами корпоративних IT-рішень і постачальниками програмного забезпечення дозволить масштабувати проєкт без значних витрат на власну мережу збуту.

Важливою складовою оптимізації є впровадження електронних каналів продажу. Створення офіційного сайту продукту з можливістю придбання ліцензії онлайн, демонстрацією функціональних можливостей і відеоінструкціями значно спростить процес знайомства потенційного користувача з системою. Використання маркетплейсів програмного забезпечення, спеціалізованих платформ для розробників і кібербезпекових рішень забезпечить доступ до міжнародного ринку та дозволить залучити нових клієнтів без значних витрат на фізичну дистрибуцію.

Доцільно також оптимізувати канали збуту шляхом упровадження гнучкої моделі ліцензування. Розробка декількох тарифних планів – для приватних користувачів, малого бізнесу та великих організацій – забезпечить глибше охоплення ринку.

Особливу увагу слід приділити розвитку комунікацій із клієнтами після продажу. Нарешті, оптимізація шляхів реалізації має ґрунтуватися на постійному аналізі ефективності кожного каналу.

7.7 Визначення ключових факторів успіху конкретного проєкту

Ключовим фактором успіху проєкту програмної реалізації системи скремблювання цифрового сигналу на мобільних пристроях є висока технологічна якість і надійність програмного продукту. Система має забезпечувати стабільне шифрування сигналу в реальному часі без затримок і втрати якості зв'язку, що є критично важливим для користувачів, які покладаються на безперебійну комунікацію. Використання сучасних криптографічних алгоритмів, оптимізованих під мобільні платформи, дозволяє гарантувати високий рівень захисту інформації навіть у випадку спроб несанкціонованого доступу. Надійність роботи, сумісність із різними операційними системами та ефективність використання ресурсів пристрою формують основу довіри до продукту та визначають його конкурентоспроможність на ринку. Не менш важливим чинником є інноваційність рішення, що проявляється у здатності поєднати потужні механізми шифрування з простотою використання. Визначальну роль у досягненні успіху відіграє рівень довіри до продукту з боку користувачів і партнерів. У сфері інформаційної безпеки навіть найкраща технологія може залишитися нереалізованою без репутаційної підтримки. Важливою складовою успіху є чітко сформована бізнес-модель і стратегія комерціалізації. Професійність і компетентність команди розробників також є одним із вирішальних факторів успіху. Окремої уваги заслуговує підтримка після впровадження – своєчасні оновлення, консультації користувачів і цілодобова технічна допомога. Таким чином, успіх проєкту програмної реалізації системи скремблювання цифрового сигналу на мобільних пристроях залежить від гармонійного поєднання технологічної досконалості, інноваційності, надійності, довіри користувачів і професіоналізму команди. Високий рівень безпеки, зручність використання, ефективна комерційна стратегія та постійний розвиток продукту створюють умови для стабільного зростання його ринкової вартості та широкого впровадження у сфері мобільних комунікацій.

8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

8.1 Вступ

Вимога щодо впровадження заходів з охорони праці передбачається, зокрема, статтею 13 Закону України «Про охорону праці». У відповідності з цим законом, кожна компанія, в рамках якої реалізуються трудові відносини, зобов'язана вжити всіх необхідних заходів з охорони праці та розробити відповідні документи. Правильний підхід до організації охорони праці на виробництві дає працівникам почуття стабільності, захищеності їх прав та інтересів, уваги з боку керівництва. Налагоджена система охорони праці також знижує плинність кадрів, що позитивно впливає на стабільність роботи підприємств.

Аналізуючи умови працівників ІТ-сфери, на перший погляд, може здатися, що працівники сфери інформаційних технологій не схильні до ризиків на виробництві, та якщо більш глибоко розглянути умови і специфіку праці фахівців сфері ІТ-індустрії, можна виявити ряд факторів які будуть мати негативний сплив на стан охорони праці, так і на самого it-фахівця. Сюди можна віднести як невідповідність освітлення, так і високий рівень шуму, що негативно позначатимуться як на емоційному так і на фізичному стані фахівця, призводитимуть до зниження ефективності праці та виробничих травм. Також, важливим моментом охорони праці ІТ-фахівця є врахування його психологічних можливостей (швидкість реакції, особливості пам'яті та уваги, емоційний стан тощо).

Для того, щоб забезпечити ефективну роботу ІТ-фахівця, потрібно враховувати та максимально компенсувати такі негативні фактори як: надмірне нервово-емоційне навантаження, довготривалі статичні перевантаження, обмежена рухова активність. Всі ці чинники призводить до різноманітних

відхилень у стані здоров'я, зокрема до перевтоми, зниження фізичної та розумової працездатності, неврозів, захворювань серцево-судинної системи тощо.

Метою даного розділу є огляд конкретних умов праці спеціаліста у сфері ІТ-індустрії. Завданнями для даного розділу є: аналіз умов праці на робочому місці фахівця it-індустрії, розробка конкретних рекомендацій щодо покращення умов праці фахівців ІТ-індустрії, огляд пожежної безпеки на ІТ-підприємстві та розрахунок системи загального штучного освітлення виробничого приміщення де працюють ІТ-фахівці.

8.2 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста

Розглянемо умови праці у приміщенні, в якому працюють програмісти. Геометричні розміри приміщення наведено у таблиці 8.1.

Таблиця 8.1 – Розміри приміщення

Найменування	Значення, м
Ширина	3
Довжина	4,6
Висота	3

Таблиця 8.2 – Площа та обсяг приміщення, на одного працюючого*

Геометрична характеристика	Одиниця виміру	Нормативне значення*	Фактичне значення
Площа, S	$м^2$	не менше 6.0	6,9
Об'єм, V	$м^3$	не менше 20.0	20,7

* Згідно ДСанПіН 3.3.2.007-98 (Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин).

У зазначеному приміщенні працюють двоє людей. За даними, які наведено у табл. 8.1, та табл. 8.2, можна зробити висновок, що площа та об'єм приміщення у розрахунку на одно робоче місце програміста не відповідають нормативним вимогам ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [2], але відповідають нормативним вимогам Наказу Міністерства соціальної політики України № 207, від 14.02.2018 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями» та НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин»). Таним чином можна зробити висновок, що санітарно-гігієнічні умови праці на робочому місці програміста відповідають вимогам.

Температура повітря в приміщенні визначається впливом температури зовнішнього повітря і тепловою енергією, яка виділяється всередині приміщення. Джерелами виділення теплоти в даному приміщенні є електроустаткування, освітлювальні прилади, а також люди. У світлий час доби джерелом надлишкового тепла є сонячна радіація. Згідно Постанови № 42 від 01.12.1999 Головного державного санітарного лікаря України, робота, виконувана в даному приміщенні, відноситься до категорії Іа. В цьому випадку людина витрачає енергії до 120 ккал у годину. Вологість повітря в приміщенні визначається впливом багатьох факторів, серед яких: вологість атмосферного повітря, виділення вологи людьми (при диханні та випарами з поверхні шкіри).

Мікроклімат повітряного середовища в приміщенні характеризується запиленістю та загазованістю повітря. Мікроклімат приміщення визначається діючим на організм людини поєднанням, вологості, температури, швидкості руху повітря та інтенсивності теплового випромінювання. Аналіз мікроклімату складається з визначення зазначених вище факторів і порівняння результатів із встановленими нормами.

У таблиці 8.3 наведено оптимальні та фактичні значення параметрів мікроклімату як для категорії ваги робіт Іа, так і розглянутого приміщення. У приміщеннях, де встановлено ЕОМ, рекомендується застосування тільки оптимальних значень показників мікроклімату.

Проведений аналіз показує, що показники мікроклімату в приміщенні відповідають установленим нормам. Штучне опалення застосовується у холодний період року.

В літню пору застосовується кондиціонер.

Таблиця 8.3 – Оптимальні і фактичні значення параметрів мікроклімату

Пора року	Оптимальні для Іа			Фактичні		
	Температура, °С	Воло-гість,%	Швидкість повітря, м/с	Температура, °С	Воло-гість%	Швидкість повітря, м/с
Холодна	22-24	40-60	0,1	22-24	40-55	0,12
Тепла	23-25	50-70	0,1	24-25	50-65	0,9

Для боротьби з пилом робляться регулярні провітрювання та вологі прибирання приміщенні.

У приміщенні знаходяться наступні джерела шуму: принтер Xerox WorkCentre 3025BI (3025VBI), електродвигуни вентиляторів ЕОМ.

Одним з найважливіших факторів, які впливають на ефективність трудової діяльності людини, та попереджають травматизм і професійні захворювання програмістів є освітлення на робочому місці.

З 2019 року діють Державні будівельні норми України "Природне і штучне освітлення" – ДБН В.2.5-28:2018 [1], у яких прописані вимоги до використання всіх освітлювальних приладів, у тому числі світлодіодних.

Працю працівника, який постійно працює за комп'ютером, згідно ДБН В.2.5-28:2018 [1], можна віднести до роботи з малою точністю (найменший розмір об'єкта розрізнення від 1 до 5 мм) V-го розряду зорової роботи, з великою

контрастністю об'єкта розрізнення (символів на екрані дисплея), з темним тлом (під розряд зорової роботи В). Приміщення можна віднести до 1-ої групи приміщень, у яких проводиться розрізнення об'єктів зорової роботи при фіксованому напрямку лінії зору того, що працює на робочу поверхню. Для такого типу приміщень і розряду зорової роботи нормоване значення коефіцієнта природної освітленості (КПО) робочої поверхні (при поєднаному, спільному освітленні), повинен становити не більше 1,5%, освітленість при штучному висвітленні повинна становити 300 Лк [1], Крім того, все поле зору повинно бути освітлено достатньо рівномірно – це основна гігієнічна вимога. Оскільки яскраве світло на ділянці периферийного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності, ступінь освітлення приміщення і яскравість екрану комп'ютера повинні бути приблизно однаковими.

8.3 Розробка заходів з умов поліпшення охорони праці

Згідно аналізу умов праці в розглянутому приміщенні, ми одержали наступні результати:

– розмірі приміщення, у розрахунку на одному працюючого, відповідають нормативам;

– мікроклімат відповідає нормативному значенню;

– акустичні умови роботи не перевищують нормативних значень;

Таким чином можна припустити, що основною причиною можливого зниження працездатності програміста є психофізіологічний фактор, тому основна пропозиція буде така: дотримання позитивної психологічної атмосфери в колективі та регламентованого режиму праці та відпочинку, організація робочого місця з урахуванням ергономічних вимог.

Рекомендовані заходи: регулярні періодичні наочні огляди персоналом шляхів для евакуації людей із приміщення, відповідно до плану евакуації (який повинен розташовуватись на видному місці у приміщенні), включення до

колективного договору мінімально можливого вмісту аптечок з обов'язковою наявністю масок-клапанів, або іншого спорядження для штучного дихання. Регулярна періодична перевірка параметрів заземлення та занулення (вимірювання опору ланцюга).

Регулярна наочне знайомство персоналу із шляхами для евакуації людей із приміщення відповідно до плану евакуації, забезпечення розподільних щитів спеціальними розетками з заземлюючими контактами; організація заземлення всіх приладів і пристроїв, які працюють при напрузі вище 36 В.

Так як при ураженні електричним струмом у людини може статися фібриляція шлуночків серця, в організації бажано мати дефібрилятор і підготовлений персонал для роботи з ним.

8.4 Пожежна безпека

Вимоги до пожежної безпеки на підприємстві неухильно повинен дотримуватися кожен співробітник, а організаційна складова при цьому покладається на посадових осіб за відповідним рішенням керівництва і прописується в посадових інструкціях і положеннях по структурним підрозділам.

Зокрема, вказуються конкретні території, ділянки, зони, об'єкти, цілі будівлі і їх частини, поверхи, на яких відповідального співробітника повинне проводити такі організаційні роботи.

Відповідальні особи зобов'язуються розробити, впровадити та підтримувати в певному інструкцією і положенням на ввірених їм об'єктах протипожежний режим і інструкції відповідно до вимог, викладених в нормативних актах.

Передбачено також створення підрозділу добровільної пожежної охорони та пожежно-рятувальної команди в його складі.

Встановлений режим включає порядки з описом місць спеціального призначення та правила їх користування та утримання, наприклад:

– евакуаційних шляхів;

– так званих «курилок»;

– місць складування продукції та сировини;

– стоянки транспорту.

Також встановлюється порядок роботи та технічного обслуговування:

– вентиляційного устаткування;

– засобів пожежогасіння і захисту від загорянь;

– нагрівальних приладів;

– електрообладнання.

Розробляються і впроваджуються правила роботи з відкритим вогнем і горючими матеріалами. Створюються графіки проходження інструктажів з пожежної безпеки співробітників, а також порядок і терміни перевірок знань пожежно-технічного мінімуму, в тому числі, тих працівників, які відповідальні за цю ділянку роботи на підприємстві. При цьому можуть передбачатися внутрішні лекції, семінари, тренінги та практичні заняття на підприємстві, а також зовнішні – на базі спеціалізованих навчальних центрів з професійними викладачами.

Важливою складовою протипожежного режиму на будь-якому об’єкті є розробка і впровадження порядку дій при виникненні пожежі. Неодмінно має бути план евакуації, описано, як повинні відключатися електроустановки, що і в якій послідовності необхідно робити співробітникам.

Відповідно, для кожного об’єкта, кожного приміщення (крім коридорів, санвузлів, басейнів і подібних приміщень), окремих видів робіт складаються інструкції, за якими повинен працювати персонал, залучений на певних ділянках і в виконанні окремих видів робіт. За інструкціями проводиться навчання (інструктаж) персоналу з подальшим контролем знань.

Детально про те, як розробити протипожежний режим, прописати порядки та інструкції, пояснюють на тематичних курсах і семінарах [2].

8.5 Розрахункова частина

Система освітлення робочого місця користувача ПК має відповідати наступним вимогам (рис. 8.1).

Рисунок 8.1 – Вимоги до системи освітлення робочого місця користувача ПК

Проведемо розрахунок штучного освітлення за методом коефіцієнта використання світлового потоку для приміщення, ширина якого складає 6,4 м, довжина – 8,3 м, висота – 3,2 м.

У зазначеному приміщенні працює 8 людей.

Для того, щоб визначити потрібну кількість світильників, які повинні забезпечити нормований рівень освітленості, визначимо світловий потік, що падає на робочу поверхню за формулою:

$$F=E\cdot S\cdot K\cdot Z/n,$$

де: F – світловий потік, що розраховується, Лм;

E – нормована мінімальна освітленість, Лк; *E = 300 Лк;*

S – площа освітлюваного приміщення.

K – коефіцієнт запасу, що враховує зменшення світлового потоку лампи в результаті забруднення світильників в процесі експлуатації (його значення залежить від типу приміщення і характеру робіт, що проводяться в ньому, в нашому випадку *K = 1,5);*

Z – відношення середньої освітленості до мінімальної (зазвичай приймається рівним 1.1... 1.2, в нашому випадку *Z = 1,1*);

n – коефіцієнт використання світлового потоку, (відношення світлового потоку, що падає на розрахункову поверхню, до сумарного потоку всіх ламп і обчислюється в долях одиниці; залежить від характеристик світильника, розмірів приміщення, забарвлення стін і стелі, що характеризуються коефіцієнтами відбиття від стін ($\rho_{стін.}$) і стелі ($\rho_{стелі}$), значення коефіцієнтів дорівнюють $\rho_{стін}$ = 50% і $\rho_{стелі}$ = 50%.

Обчислимо індекс приміщення за формулою:

$$i=S/(h\cdot (A+B)),$$

де:

S – площа приміщення, *S* = 53,1 *м*^{2}*;*

h – розрахункова висота підвісу, *h* = 3,2 *м. (співпадає з в*исотою стелі, оскільки лампи освітлення закріплюються на стелі)*;*

A – ширина приміщення, *A = 6,4 м;*

B – довжина приміщення, *B* = 8,3 *м.*

Підставимо всі значення у формулу та визначимо індекс приміщення: i=1,1.

Знаючи індекс приміщення, за знаходимо *n = 0,46 (з табличних даних коефіцієнтів використання світлового потоку (n) світильників з відповідним типом лампам) [8].* Підставимо всі значення у формулу, визначимо світловий потік: F=57161,7 Лм.

Для розрахунку будемо використовувати стельові *світлодіодні панелі* Призма-72 6400К, світловий потік яких $F_л$ = 7200 *Лм.*

Число ламп визначається по формулі:

$$N=F/F_л$$

де:

F – світловий потік,

$F_л$ – світловий потік однієї лампи.

Підставимо всі значення у формулу та визначимо індекс приміщення:

$$N= 57161,7/ 7200=7,9 \text{ шт.}$$

Приймаємо необхідну кількість *світлодіодних світильників* 8 шт.

Висновки до розділу

Дотримання всіх необхідних умов праці не лише сприяє збереженню здоров'я працівників, а також підвищує ефективність виробництва в цілому.

З цих міркувань було здійснено аналіз умов праці, призначеного для праці програмістів, проведено розгляд небезпечних та шкідливих факторів, що негативно впливають на програмістів під час роботи. Виконано розрахунок штучного освітлення, як одного з ключових факторів впливу на працездатність та здоров'я програміста. Розроблено заходи з охорони праці.

9 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти, призначено для системи скремблювання цифрового сигналу на мобільних пристроях.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти наведені теоретичне узагальнення й рішення наукового завдання дослідження методів скремблювання цифрового сигналу на мобільних пристроях.

Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем скремблювання цифрового сигналу на мобільних пристроях.

– Досліджена система скремблювання цифрового сигналу на мобільних пристроях.

– На основі отриманих результатів досліджень створена програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях.

Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання скремблювання цифрового сигналу на мобільних пристроях.

Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об’єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Python. Дана мова програмування дозволяє найбільш ефективно обробляти дані. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Android.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм AES.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

Проведено маркетингове та економічне обґрунтування ІТ-проєкту, що дозволило визначити ключові фактори успіху даного проекту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Андрусик Б.М. Дослідження та програмна реалізація системи скремблювання цифрового сигналу на мобільних пристроях // Збірник праць молодих науковців ЦНТУ. – Вип. 15. – Кропивницький: ЦНТУ, 2025.

2. Alex Matrosov, Eugene Rodionov, Sergey Bratus. Rootkits and Bootkits. No Starch Press. 2019. 450 p.

3. Петрик В.М., Присяжнюк М.М., Аль-Файюмі Халед та ін. «Системи інформаційної зброї та технології інформаційної війни»: підручник / Петрик В.М., Присяжнюк М.М., Аль-Файюмі Халед, Жарков Я.М., Смірнов О.А, Буравченко К.О., Давидюк А.В., Кононович В.Г., Корчинский В.В., Кудирко В.М., Фесенко А.О.; за заг. ред. В.М. Петрика, М.М. Присяжнюка.– К.: Видавничий центр “Кафедра”, 2025.– 320 с.

4. Усік, П.С., Смірнова, Т.В., Буравченко, К.О., Смірнов, О.А., Улічев, О.С., Смірнов, С.А. «Дослідження технологій забезпечення кібербезпеки банківських систем з використанням штучного інтелекту». *Кібербезпека: освіта, наука, техніка.* 2025. Том 1 № 29. С.704–716, 2025

5. Kuznetsov, O., Frontoni, E., Kuznetsova, K., Arnesano, M., Smirnov, O. «A secure biometric authentication architecture for blockchain-driven cyber-physical systems». *Security and Privacy of Cyber Physical Systems Emerging Trends Technologies and Applications*, 2025, pp. 193–224.

6. Kuznetsov, O., Atzeni, G., Arnesano, M., Randieri, C., Smirnov, O. «Secure IoT-based smart wheelchair system: From implementation to security enhancement strategy». *Security and Privacy of Cyber Physical Systems Emerging Trends Technologies and Applications*, 2025, pp. 225–257.

7. Kuznetsov, O., Smirnov, O., Kuznetsova, T., Shaikhanova, A., Svatowsky, I. «Privacy-utility trade-offs in IoT networks: A comparative analysis of differential privacy mechanisms for sensor data aggregation». *Security and Privacy of Cyber*

Physical Systems Emerging Trends Technologies and Applications, 2025, pp. 589–622.

8. Lakhno, V., Malyukov, V., Smirnov, O., Bebeshko, B., Chubaievskiy, V., Zhumadilova, M., Malyukova, I., Smirnov, S. «Multifactorial Model for Targeted Attacks Counteracting Within the Framework of a Multi-Step Quality Game with Fuzzy Information». *8th International Symposium on Intelligent Informatics, ISI 2023*, 2025. vol 389. pp 377-389. Springer, Singapore.

9. Kuznetsov O., Frontoni E., Kuznetsova Y., Smirnov O., Moskovchenko I. «Trust-Based Security Architecture for Edge Computing: A Simulation Study of Dynamic Trust Evolution and Attack Detection». *CEUR Workshop Proceedings,* 2024, 3909, pp. 227–241.

10. Kuznetsov, O., Frontoni, E., Kryvinska, N., Chevardin, V., Smirnov, O. «Wireless Network Encryption Stream Ciphers, Computational Modeling, and Security Analysis». *Computational Modeling and Simulation of Advanced Wireless Communication Systems*, 2024, pp. 379–402.

11. Kuznetsov, O., Frontoni, E., Kryvinska, N., Smirnov, O., Imoize, G.L. «Computational Modeling of Enhanced Spread Spectrum Codes for Asynchronous Wireless Communication». *Computational Modeling and Simulation of Advanced Wireless Communication Systems*, 2024, pp. 403–447.

12. Ткаченко, О., Ільєнко, А., Улічев, О., Мелешко, Є., Смірнов, О. «Правові засади поширення інформаційних впливів в соціальних мережах». *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»,* 2024. № 2(26), С. 170–188.

13. Смірнова Т.В., Конопліцька-Слободенюк О.К., Буравченко К.О., Смірнов С.А., Кравчук О.В., Козірова Н.Л., Смірнов О.А. «Дослідження технологій забезпечення кібербезпеки хмарних сервісів IaaS, PaaS та SaaS». *Кібербезпека: освіта, наука, техніка.* 2024. №4(24), С. 6-27.

14. Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для

безпеки». *Кібербезпека: освіта, наука, техніка.* 2024. №3(23), С. 111-131.

15. Батрак О., Смірнова Т., Гнатюк В., Одарченко Р., Смірнов О. «Дослідження показників ефективності функціонування та перспектив розвитку систем IP-телефонії». *Підводні технології,* 2024, № 13, с. 28-35.

16. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, 2023, 3628, pp. 106-115.

17. Akhalaia, G., Iavich, M., Iashvili, G., Prysiazhnyy, D., Smirnova, T. «Secure Encrypted Connection on Georgian Website». *CEUR Workshop Proceedings,* 2023, 3550, pp. 313-320.

18. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». *Advanced Information Systems*, 2023, 7(2), pp. 49-56

19. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings,* Volume 3530, 2023, pp. 256-265.

20. Kuznetsov, O., Kandiy, S., Frontoni, E., Smirnov, O. «Trade-offs in Post-Quantum Cryptography: A Comparative Assessment of BIKE, HQC, and Classic McEliece». *CEUR Workshop Proceedings,* Volume 3504, 2023, pp. 1-11.

21. Smirnov, O., Lakhno, V., Akhmetov, B., Chubaievskyi, V., Khorolska, K., Bebeshko, B. «Selection of a Rational Composition of İnformation Protection Means Using a Genetic Algorithm». *In: Rajakumar, G., Du, KL., Vuppalapati, C., Beligiannis, G.N. (eds) Intelligent Communication Technologies and Virtual Mobile Networks. Lecture Notes on Data Engineering and Communications Technologies*, vol 131. 2023. Springer, Singapore. pp. 21-34.

22. Смірнова Т.В., Гнатюк С.О., Бердибаєв Р.Ш., Сидоренко В.М., Жигаревич О.К., «Система корелювання подій та управління інцидентами кібербезпеки на об’єктах критичної інфраструктури». *Кібербезпека: освіта,*

наука, техніка, №*3*(19), 2023, С. 176-196.

23. Смірнов О.А. Козлов Я.О., Смірнова Т.В. «Дослідження застосування SIEM-систем для забезпечення кібербезпеки та захисту інформації». *II Міжнародна науково-практична Інтернет-конференція «Інновації та перспективні шляхи hозвитку інформаційних технологій (ІПШРІТ-2023)»* м.Черкаси 6 грудня 2023 року – Черкаси: ЧДТУ.– 2023. – С.251-252.

24. Козлов Я.О., Смірнова Т.В., Смірнов О.А. «Дослідження SIEM-систем для забезпечення кібербезпеки». *VII міжнародна науково-практична конференція "Інформаційна безпека та комп'ютерні технології" до 30-ти річчя кафедри кібербезпеки та програмного забезпечення,* м. Кропивницький. 1 листопада 2023 р. – Кропивницький: ЦНТУ. – 2023. – С. 26.

25. Козлов Я.О., Козірова Н.Л., Смірнов О.А. «Дослідження структури та принципу роботи SIEM-системи». *VII міжнародна науково-практична конференція "Інформаційна безпека та комп'ютерні технології" до 30-ти річчя кафедри кібербезпеки та програмного забезпечення,* м. Кропивницький. 1 листопада 2023 р. – Кропивницький: ЦНТУ. – 2023. – С. 59.

26. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А., Коваленко А.С. «Дослідження нормативних документів та галузевих стандартів розробки програмного забезпечення комп'ютерних систем управління АЕС, важливих для безпеки». *Системи управління, навігації та зв'язку*, 2023, вип. 2(72), С. 170-178.

27. Smirnov, O., Neskorodieva, T., Fedorov, E., Rudakov, K., Neskorodieva, A. «Method Detection Audit Data Anomalies on Basis Restricted Cauchy Machine» *CEUR Workshop Proceedings,* Volume 3187, 2022,

28. Smirnov O.A., Al-Oraiqat A.M., Ulichev O.S., Meleshko Ye.V., Al-Rawashdeh H.S., Polishchuk L.I. «Modeling strategies for information influence dissemination in social networks». *Journal of Ambient Intelligence and Humanized Computing* Volume 13, Issue 5. Springer, Cham. 2022, pp. 2463-2477.

29. Смірнова Т.В., Гнатюк С.О., Сидоренко В.М., Юдін О.Ю.,

Сидоренко С.Ю., «Модель визначення критичності галузевих інформаційно-телекомунікаційних систем». *Проблеми інформатизації та управління,* № 2(70). 2022. С. 28-37.

30. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Смірнов С.А., Поліщук Л.І., «Дослідження стійкості до диференціального криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку,* 2022, № 3(69). С. 93-98.

31. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Поліщук Л.І., Смірнов С.А. «Дослідження статистичної стійкості та швидкісних характеристик запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Вісник Хмельницького національного університету. Серія: «Технічні науки»,* № 2 (307). С. 46-52. 2022.

32. Смірнов О.А., Смірнова Т.В., Константинова Л.В., Смірнов С.А., Якименко Н.М., «Дослідження стійкості до лінійного криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 1(67). С. 84-89.

33. Smirnov O., Kuznetsov A., Zhora V., Onikiychuk A., Pieshkova O. «Hiding Messages in Audio Files Using Direct Spread Spectrum». *11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2021,* Cracow, Poland, 22-25 September 2021. P. 414-418

34. Smirnov O., Kuznetsov A., Lokotkova I., Kuznetsova T., Florov S., Lebid O. «Using Orthogonal Signals to Hide Information in Images». *4 IEEE International Conference on Advanced Information and Communication Technologies (AICT) – 2021*, Lviv, Ukraine, September 21-25, 2021. P. 255-260.

35. Smirnov O., Kuznetsov A., Girzheva O., Kiian A., Nakisko O., Kuznetsova T. «Advanced Code-Based Electronic Digital Signature Scheme». *2020*

IEEE International Conference on Problems of Infocommunications Science and Technology, PIC S and T 2020, Kharkiv, 6 October 2020-9 October 2020, P. 358-362.

36. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova K. «Data hiding scheme based on spread sequence addressing». *CEUR Workshop Proceedings* Volume 2805, 2020, Pages *44-58.*

37. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». *International Journal of Computing*; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.

38. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». *CEUR Workshop Proceedings*. Volume 2740, 2020, Pages 102-114.

39. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346.

40. Smirnov O., Kuznetsov A., Arischenko A., Chepurko I., Onikiychuk A., Kuznetsova T. «Pseudorandom sequences for spread spectrum image steganography». *CEUR Workshop Proceedings* Volume 2654, 2020, Pages 122-131.

41. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New technique for data hiding in cover images using adaptively generated pseudorandom sequences». *CEUR Workshop Proceedings* Volume 2654, 2020, Pages 1-14.

42. Smirnov O., Lutsenko M., Kuznetsov A., Kiian A., Kuznetsova T., «Biometric cryptosystems: overview, state-of-the-art and perspective directions». *Lecture Notes in Networks and Systems*, vol 152. Springer, Cham. 2021, pp 66-84.

43. Smirnov O., Kuznetsov A., Onikiychuk A., Makushenko T., Anisimova O., Arischenko A. «Adaptive pseudo-random sequence generation for spread spectrum image steganography». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Ukraine, Kyiv, May 14-18. 2020. P.

161-165.

44. Smirnov O., Kuznetsov A., Kiian A., Babenko V., Perevozova I.,Chepurko I. «New Approach to the Implementation of Post-Quantum Digital Signature Scheme». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Ukraine, Kyiv, May 14-18. 2020. P. 166-171.

45. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

46. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

47. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136.

48. Smirnov, O., Shekhanin, K., Kuznetsov, A., Krasnobayev, V. «Detecting Hidden Information in FAT». *International Journal of Computer Network and Information Security (IJCNIS)*. Vol. 12, No. 3, 2020. PP.33-43.

49. Smirnov, O., Kuznetsov, A., Gorbacheva, L., Babenko, V., «Hiding data in images using a pseudo-random sequence», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 646-660.

50. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

51. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T.,

Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019 . P.517-522.

52. Smirnov, O., Ulichev, O., Meleshko, Y., Khokh, V., Goncharenko, I. «Method of Choosing Objects for Informational Influence in Social Networks during Information Campaign Based on the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, Vol 2588, P. 215-227, 2019.

53. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». *CEUR Workshop Proceedings*, Vol 2588, P. 90-106, 2019.

54. Smirnov, O., Kuznetsov, A., Kiian, A., Gorbenko, Y., Cherep, O., Bexhter L. «Code-based Pseudorandom Generator for the Post-Quantum Period», *2019 IEEE International Conference on Advanced Trends in Information Theory (IEEE ATIT 2019).* 18.12.19-20.12.19 Kyiv Ukraine. P. 204 – 209.

55. Smirnov, O., Kuznetsov, A., Nariezhnii, O., Stelnyk, S., Kokhanovska, T., Kuznetsova T., «Side Channel Attack on a Quantum Random Number Generator», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019*; Metz; France; 18 - 21 September 2019. P.713-718.

56. Kuznetsova, T., «Code-Based Schemes for Post-Quantum Digital Signatures», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019;* Metz; France; 18-21 September 2019. P. 707-712.