ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ

Навчально-науковий інститут заочно дистанційної форми навчання
Форма навчання заочна
Кафедра комп'ютерних наук та інформаційних технологій

Допускається до захисту
Завідувач кафедри
_____ Олена ОЛЬХОВСЬКА
(підпис)

«___»_____202_ р.

КВАЛІФІКАЦІЙНА РОБОТА

на тему

РОЗРОБКА МОДЕЛІ ЗАХИСТУ ІНФОРМАЦІЇ WIFI МЕРЕЖІ

зі спеціальності 122 Комп'ютерні науки освітня
програма «Комп'ютерні науки» ступеня магістра

Виконавець роботи Личкін Володимир Володимирович
_____«____»_____ 202_ р.
(підпис)

Науковий керівник к. ф.-м. н., доцент, Черненко Оксана Олексіївна
_____«____»_____ 202_ р.
(підпис)

Рецензент

ПОЛТАВА 2023

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

Умовні позначення, символи, скорочення, терміни	Пояснення умовних позначень, скорочень, символів
AP (Access Point)	Точка доступу — це пристрій, який з'єднує бездротові пристрої з провідною мережею Інтернету.
SSID (Service Set Identifier)	Це унікальний ідентифікатор бездротової мережі (Wi-Fi). Кожна бездротова мережа має свій власний SSID, який ідентифікує її у мережі.
SSL (Secure Socket Layers)	SSL) і є криптографічним протоколом, призначеним для забезпечення безпечного обміну даними між веб-браузером користувача і веб-сайтом. SSL забезпечує шифрування даних, які передаються між клієнтом і сервером, що унеможливлює несанкціонованому третьому особі доступ до чутливої інформації, такої як особисті дані або дані банківської картки.
WEP (Wired Equivalent Privacy)	Протокол захисту даних при передачі по радіоканалу.

ВСТУП

В сучасному цифровому світі, де бездротовий доступ до мережі став невід'ємною частиною нашого повсякденного життя, важливість розробки надійної моделі захисту для Wi-Fi мережі визнається як критичний аспект забезпечення інформаційної безпеки. Захист Wi-Fi стає викликом через постійний розвиток технологій та зростання кількості пристроїв, які підключаються до бездротових мереж.

Важливість розробки надійних захисних механізмів стає ключовою у забезпеченні приватності особистих даних, запобіганні несанкціонованому доступу і захисті від різноманітних атак, що можуть використовувати вразливості мережі.Забезпечення безпеки Wi-Fi не лише захищає від потенційних кіберзагроз, але й сприяє збереженню ефективності мережі та забезпеченню надійності зв'язку між пристроями. [2]

Мета кваліфікаційної роботи – розробка програмного забезпечення моделі захисту для бездротових мереж.

Об’єкт розробки – розроблена модель, алгоритм моделі, засіб захисту інформації та ключів доступу.

Предмет розробки – модель захисту для бездротових мереж.

Кваліфікаційна робота складається з чотирьох розділів. У першому розділі розглянуто постановку задачі для втілення моделі захисту. В другому розділі описано про види загроз для бездротових мереж та Алгоритм шифрування AES. В третьому розділі описано про засоби захисту інформації мереж Wi-Fi та складено алгоритм моделі. У четвертому розділі описано процес створення моделі та описана інструкція користувача. Обсяг пояснювальної записки: 41 стор., в т.ч. основна частина – 36 стор., джерела – 20 назв.

1. ПОСТАНОВКА ЗАДАЧІ РОЗРОБКИ

Одним з важливих завдань кваліфікаційної роботи є розробка програмного забезпечення моделі захисту WIFI мережі, яка шифрує ключі мережі WIFI з використанням алгоритму шифрування AES.

Для досягнення мети визначаються наступні завдання:

1. Вивчаються наявні вирішення в сфері інформаційної безпеки у wifi мережах.
2. Розробляється модель та її дизайн.
3. Складається алгоритм.
4. Складається блок схема.
5. Розробляється функціональна частина моделі.
6. Проводиться тестування роботи моделі.
7. Опис роботи та програмної реалізації даної моделі, інструкція користувача.

Методологія дослідження включає описовий підхід, методи аналізу та синтезу, експериментальний метод і метод узагальнення. Щодо наукової новизни цієї розробки, варто відзначити розробку інструменту захисту інформації в радіомережах. Застосування цього інструменту суттєво підвищить рівень інформаційної безпеки.

Практичне значення даної розробки виявляється в тому, що отримані теоретичні та практичні результати рекомендуються для використання в організаціях, які передають конфіденційну інформацію через радіоканал і мають високі вимоги до безпеки.

2. ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1. Види загроз для бездротових мереж

Актуальною проблемою використання бездротових мереж є їх захист та способи захисту даних в них, оскільки комунікаційні сигнали при їх розповсюдженні через радіоефір доступні для перехоплення.

Існує кілька форм загрози безпеці в бездротових мережах. Так, хакери можуть викрасти дані, отримавши неавторизований доступ до мережі, і навіть порушити роботу мережі.

Моніторинг трафіку. Досвідчений хакер або навіть випадковий снупер (snooper – відстежувач, перехоплювач) може відстежити пакети даних в незахищеній бездротовій мережі, використовуючи відповідні програмні засоби за допомогою яких можна повністю розшифрувати вміст пакетів даних із бездротової мережі.

Неавторизований доступ. Також можна здійснити моніторинг виконуваних в мережі програм і без особливих зусиль, якщо не прийнято належних запобіжних заходів, отримати доступ до бездротової мережі, знаходячись поза приміщенням, де вона функціонує.

За допомогою сучасних операційних систем можна легко встановлювати під’єднання до бездротових мереж, особливо до загальнодоступних. Коли комп’ютер (ноутбук) під’єднаний до бездротової локальної мережі, його власник отримує доступ до будь-якого іншого комп’ютера (ноутбука), що під’єднаний до тієї самої бездротової локальної мережі. Якщо на комп’ютері не встановлений персональний брандмауер, то хто завгодно може отримати доступ до даних такого комп’ютера (ноутбука).

Навіть якщо в бездротовій мережі задіяні механізми захисту, істотною загрозою є під’єднання до підставної точки доступу (rogue access point).

Атака типу «людина всередині». Завдяки використанню механізмів шифрування і аутентифікації підвищується безпека бездротової мережі, проте, досвідчені хакери відшукують слабкі місця, знаючи, як працюють протоколи мережі. Певну небезпеку представляють атаки типу «людина всередині» (man-in-the-middleattacks): хакер розміщує фіктивний пристрій між легальними користувачами і бездротовою мережею. Наприклад, при здійсненні стандартної атаки типу «людина всередині» використовується протокол перетворення адрес (address resolution protocol (ARP)), який використовується у всіх мережах Ethernet. Хакер, маючи необхідне програмне забезпечення, може, скориставшись ARP, отримати контроль над бездротовою мережею. За допомогою ARP відправляються запити (як в провідниковій, так і в бездротовій мережі) через мережеву плату з метою виявлення іншої фізичної або MAC адреси, куди має прийти даний запит.

Проблема, яка виникає при використанні протоколу ARP, полягає в тому, що є небезпека для системи захисту даних за допомогою спуфінга (spoofing (спуфінг) – імітація з’єднання, отримання доступу обманним шляхом). Можна імітувати з’єднання між комп’ютерами, посилаючи на один з комп’ютерів через підставний мережевий пристрій фіктивний ARP запит, що містить IP-адресу дійсного мережевого пристрою іMAC-адресу підставного. Це приведе до того, що на всіх комп’ютерах мережі автоматично відновляться ARP-таблиці, які будуть містити помилкові дані. В результаті за допомогою комп’ютерів передаватимуться пакети до підставного пристрою, а не до дійсної точки доступу або маршрутизатора. Це і є класична атака типу «людина всередині», в результаті якої можна отримати доступ до управління сеансами зв'язку користувача, отримати паролі, важливі дані і навіть можна отримати доступ до корпоративних серверів.

Для запобігання такого роду атак з використанням спуфінга ARP розробники пропонують захищені ARP (secure ARP, SARP). Цей ARP забезпечує спеціальний захищений канал зв’язку («тунель») між кожним

клієнтом і бездротовою точкою доступу або маршрутизатором, за допомогою якого ігноруються всі ARP-відповіді, не пов'язані з клієнтом, що знаходиться на другому кінці цього каналу зв’язку. Але можна встановити SARP на клієнтських пристроях (комп’ютерах, ноутбуках), забезпечивши захист мережі від атак типу «людина всередині».

Атака типу «Відмова в обслуговуванні». Атака типу «відмова в обслуговуванні» (denial of service, DoS) – це атака, в результаті якої бездротова мережа стає недоступною або її робота блокується. Можливість такої атаки потрібно враховувати при створенні і використанні бездротових мереж. Серйозність DoS-атаки залежить від того, до яких наслідків може привести вихід з ладу бездротової мережі.

Одним з різновидів DoS-атакє метод «грубої сили» (bruteforce attack). Масове розсилання пакетів в мережі, при якому використовуються всі ресурси мережі, в результаті чого мережа переповнюється і блокується – це і є варіант DoS-атаки, виконаний за методом «грубої сили».

Іншим методом припинення роботи більшості бездротових мереж, особливо тих, в яких використовується метод виявлення мережі, є використання сильного радіосигналу, що «глушить» всі інші. Проте спроба проведення атаки на мережу з використанням сильного радіосигналу може виявитися вельми ризикованою, оскільки для проведення такої атаки потрібний потужний передавач, який повинен розташовуватися в безпосередній близькості від приміщення, в якому розгорнута бездротова мережа. Власник мережі може виявити цей передавач, використовуючи засоби виявлення, що входять до складу мережевих аналізаторів. Іноді «відмова в обслуговуванні» бездротової мережі виникає внаслідок ненавмисних дій.

Найбільш дієвим захистом від DoS-атаки розробка і дотримання таких правил безпеки:

1. встановлення та оновлення брандмауерів;
2. постійне оновлення антивірусних програмних засобів;

3. встановлення останніх оновлень, за допомогою яких ліквідовують недоліки в системі безпеки операційної системи;
4. використання довгих паролів;
5. від’єднання мережевих пристроїв, які не використовуються.

Універсального способу протидії DoS-атакам всіх типів не існує. Тому, якщо в результаті атаки бездротова мережа все ж таки вийшла з ладу, слід забезпечити перехід до пакетного опрацювання даних за допомогою дротової мережі.

2.2. Алгоритм шифрування AES

У 1998 році NIST (National Institute of Standards and Technology) оголосив конкурс на створення алгоритму симетричного шифрування, алгоритм отримав назву AES (Advanced Encryption Standard). Алгоритм планувалося прийняти як стандарт Сполучених Штатів Америки замість застарілого на той час стандарту DES (Digital Encryption Standard), що був американським стандартом з 1977 року. Необхідність у прийнятті нового стандарту була викликана невеликою довжиною ключа DES (56 біт), що дозволяло успішно застосовувати метод прямого перебору ключів для злому DES. Крім того, архітектура DES була орієнтована на апаратну реалізацію, і програмна реалізація алгоритму на платформах з обмеженими ресурсами не давала достатньої швидкодії. Модифікація TripleDES володіла достатньою довжиною ключа, але при цьому була ще повільніше. 2 січня 1997 року NIST оголошує про намір вибрати наступника для DES. Конкурс був оголошений 12 вересня 1997р. Свій алгоритм могла запропонувати будь-яка організація або група дослідників. NIST опублікував всі дані про тестування кандидатів на роль AES і зажадав від авторів алгоритмів повідомити про базові принципи побудови алгоритмів, о константантах, що використовуваних в них, таблицях для замін (S-box) і т.п. На відміну від ситуації з DES, NIST при виборі AES не став опиратися на секретні і, як наслідок, заборонені до публікації дані про дослідження

алгоритмів-кандидатів. Вимоги до кандидатів на новий стандарт були наступними:

• блоковий шифр;

• довжина блоку дорівнює 128 бітам;

• ключі довжиною 128, 192 і 256 біт. Додатково кандидатам рекомендувалося:

• використовувати операції, які можуть бути легко реалізовані як апаратно (в мікрочіпах), так і програмно (на персональних комп'ютерах і серверах);

• орієнтуватися на 32-розрядні процесори

• не ускладнювати без необхідності структуру шифру для того, щоб всі зацікавлені сторони були в змозі самостійно провести незалежний кріптоаналіз алгоритму і переконатися, що в ньому не закладено жодних недокументованих можливостей.

Крім того, алгоритм, який претендує на те, щоб стати стандартом, повинен поширюватися по всьому світу не на ексклюзивних умовах і без плати за користування патентом. Алгоритм AES насамперед повинен пропонувати високу ступінь захисту, мати просту структуро і високу продуктивністю. Вже на рівні внутрішньої архітектури він повинен володіти надійністю, достатньою для того, щоб протистояти майбутнім спробам його злому. Було проведено конкурс серед алгоритмів шифрування на роль AES, 2 жовтня 2000 було оголошено, що переможцем конкурсу став алгоритм Rijndael і почалася процедура стандартизації. 28 лютого 2001 року було опубліковано проект, а 26 листопада 2001 року AES був прийнятий як стандарт FIPS 197. AES не тотожний Rijndael, тому що оригінальний алгоритм Rijndael підтримує більш широкий діапазон довжин ключів і блоків. В AES розмір ключа фіксований і дорівнює 128 біт, а в Rijndael підтримуються різна довжин ключів - від 128 до 256 біт, з кроком 32 біта. AES оперує з блоком даних 16 байт, а Rijndael дозволяє вибирати розмір блоку. Rijndael - швидкий і компактний алгоритм з

простою математичною структурою. Він продемонстрував хорошу стійкість до атак на реалізацію, при яких намагаються декодувати зашифроване повідомлення, аналізуючи зовнішні прояви алгоритму, в тому числі рівень енергоспоживання і час виконання. Алгоритму притаманний внутрішній паралелізм, що дозволяє забезпечити ефективне використання процесорних ресурсів. Далі під AES можна розуміти Rijndael з ключем в 128 біт і блоком даних 16 байт.

Алгоритм представляє кожен блок кодованих даних у вигляді двовимірного масиву байт розміром 4x4, 4x6 або 4x8 в залежності від встановленої довжини блоку. Далі на відповідних етапах перетворення виробляються або над незалежними стовпцями, або над незалежними рядками, або взагалі над окремими байтами в таблиці.

Всі перетворення в шифрі мають суворе математичне обґрунтування. Сама структура і послідовність операцій дозволяють виконувати даний алгоритм ефективно як на 8-бітних так і на 32-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 певної кількості раундів (від 10 до 14 - це залежить від розміру блоку і довжини ключа), в яких послідовно виконуються наступні операції:

ByteSub - таблична підстановка 8x8 біт (рис.3.1)

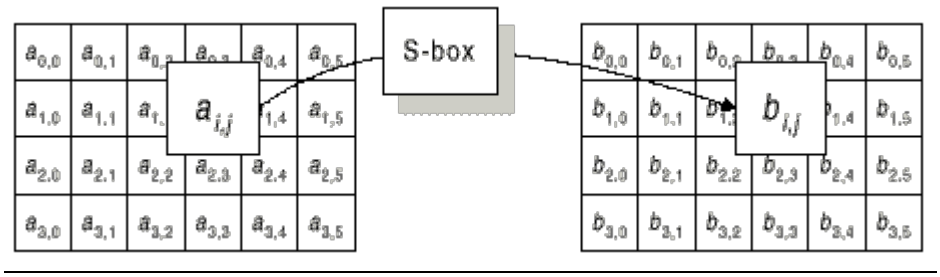

Рисунок 3.1 - Таблична підстановка 8x8 бит

ShiftRow – зсув рядків в двовимірному масиві на різні зміщення (рис.32),

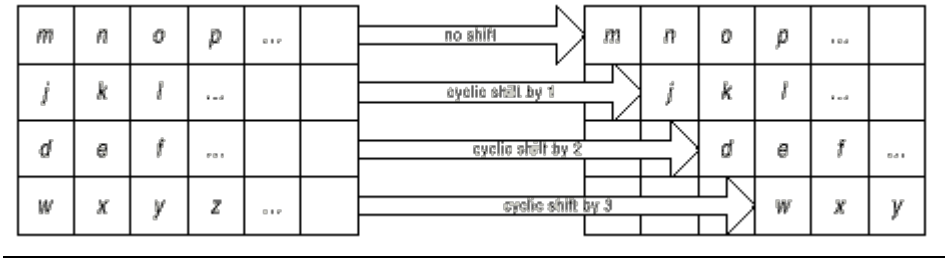

Рисунок 3.2 – Зсув рядків в двовимірному масиві

MixColumn – математичне перетворення, перемішуючи дані всередині стовпчика (рис.3.3),

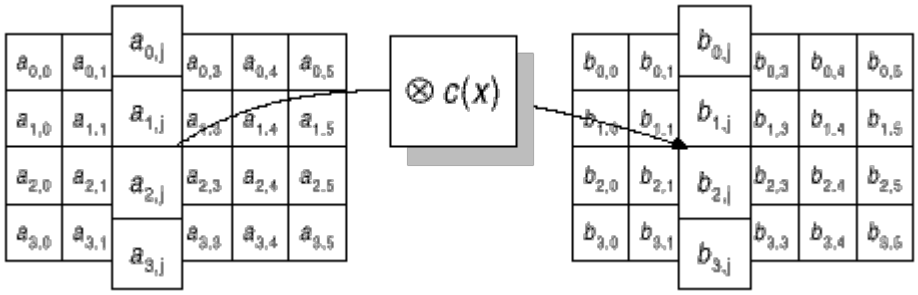

Рисунок 3.3 – Перемішування даних

AddRoundKey – додавання матеріалу ключа операцією XOR (рис.3.4).

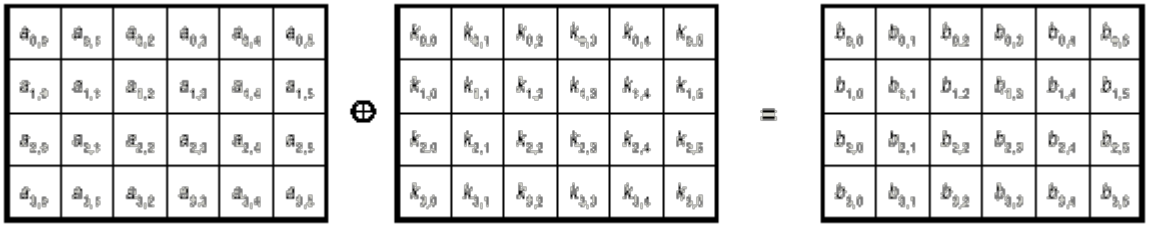

Рисунок 3.4 – Додавання ключа

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

2.3. Порівняльний аналіз алгоритмів DES та AES

Можна виділити дві основні характеристики, які можуть дати уявлення про відмінність одного алгоритму від іншого: це криптостійкість і швидкість шифрування / дешифрування. Оцінка таких характеристик є складним завданням, оскільки вимагає значних витрат на перевірку продуктивності алгоритму при різних настройках і різних типах даних

Для початку виділимо кроки, які використовуються у звичайній моделі шифрування:

• Відправник хоче відправити вітальне повідомлення одержувачу.

• Вихідне повідомлення, яке також називається відкритим текстом, перетворюється в випадкові біти, відомі як зашифрований текст, з використанням ключа і алгоритму. Алгоритм, що використовується може видавати різні вихідні дані кожного разу, коли його використовують, в залежності від значення ключа.

• Зашифрований текст передається по середовищу передачі.

• На стороні одержувача зашифрований текст перетворюється назад у вихідний текст з використанням того ж алгоритму і ключа, який використовувався для шифрування повідомлення.

Як показує проста модель для алгоритму шифрування важливими параметрами є: швидкість, розмір блоку і розмір ключа, що впливає на криптостійкість, а також можливості модернізації алгоритму та ін.

Для порівняння з алгоритмом DES був обраний алгоритм AES, який є більш сучасним блоковим алгоритмом шифрування.

Як було зазначено в розділі 2: DES був першим стандартом шифрування. DES використовує 56-бітний ключ і відображає 64-бітний вхідний блок в 64-бітний вихідний блок. Ключ насправді виглядає як 64-бітове кількість, але один біт в кожному з 8 октетів використовується для непарної парності в кожному октеті. На сьогоднішній день розроблено значна кількість методів атак, які показали слабкі сторони алгоритм DES, що зробило його небезпечним, але вплинуло на розвиток інших алгоритмів шифрування.

AES, також відомий як алгоритм Rijndael є симетричним блоковим шифром, який може шифрувати блоки даних довжиною 128 біт з використанням симетричних ключів 128, 192 або 256. AES був введений для заміни DES. У процесі шифрування AES використовує 10, 12 або 14 раундів. Розмір ключа, який може бути 128 192 або 256 біт, залежить від

кількості раундів. Для забезпечення безпеки AES використовує перестановки, мікшування і додавання ключів, при цьому кожен раунд AES, крім останнього, використовує, як мінімум, чотири перетворення.

У таблиці 3.1 наведено порівняльне дослідження по дев'яти факторам між DES і AES, які включають довжину ключа, тип шифру, розмір блоку, розвиненість, стійкість до криптоаналізу і таке інше.

Таблиця 3.1 – Порівняльний аналіз DES та AES

№	Параметр	DES	AES
1	Тип шифрування	Симетричний	Симетричний
2	Метод побудови	мережа Фейстеля	Підстановочно-перестановочна мережа
3	Год разробки	1977	2000
4	Довжина ключа	56 біт	128, 192 або 256 біт
5	Размір блоку	64 біта	128, 192 або 256 біт
6	Рівень безпеки	не відповідає поточним вимогам	Вважається безпечним
7	Криптостійкість	Вразливий до диференціального і лінійного криптоаналізу	Стійкий до диференціальних, лінійних, інтерполяційних і квадратурних атак
8	Вариації ключей	2^{56}	2^{128}, 2^{192} или 2^{256}
9	Час, необхідний для перевірки всіх можливих ключів зі швидкістю 50 мільярдів ключів в секунду	Для 128 бітного ключа: $5*10^{21}$ лет	Для 56 бітного ключа: 400 днів

Нижче наведені результати аналізу досліджень алгоритмів DES і AES, які були проведені в різних напрямках: криптостійкість, час шифрування-дешифрування, обсяг пам'яті і ін.

Одним з бажаних властивостей будь-якого алгоритму шифрування є те, що невелика зміна у відкритому тексті або ключі має привести до значних змін в зашифрованому тексті. Зокрема, зміна одного біта відкритого тексту або одного біта ключа повинно приводити до зміни багатьох бітів зашифрованих текстів (так званий лавинний ефект). У

таблиці 3.2 представлені результати оцінки лавинного ефекту для алгоритмів, що підлягають порівнянню. Результати показують, що в зміни для AES алгоритму практично в два рази вище, ніж для DES. Вимоги до обсягу пам'яті, необхідної для реалізації алгоритмів, відрізняються незначно таблиця 3.3. Обидва алгоритми є компактними.

Таблиця 3.2 – Лавинний ефект

Алгоритм	1-бітна зміна ключа, зберігаючи постійний текст	1-бітна зміна у відкритому тексті, зберігаючи ключ постійним
DES	30	34
AES	64	71

Таблиця 3.3 – Вимоги до пам'яті, необхідної для реалізації алгоритмів

Алгоритм	Необхідний розмір пам’яті , кБ
DES	12,8
AES	10,6

В таблиці 3.4 та на рисунку 3.5 наведені результати часових витрат на роботу алгоритмів для різної довжини повідомлень. Результати показують, що зі збільшенням обсягу даних алгоритм DES вимагає значних (на порядок) часових витрат на шифрування.

Таблица 3.4 – Часові витрати на роботу алгоритму для різної довжини повідомлення

Довжина повідомлення, кБ	Часові витрати, с	
	DES	AES
0,0013	0,3	2,46
0,636	4,8	4,86
2,90	57	12
6,17	245	22,88
8,21	438	29,56

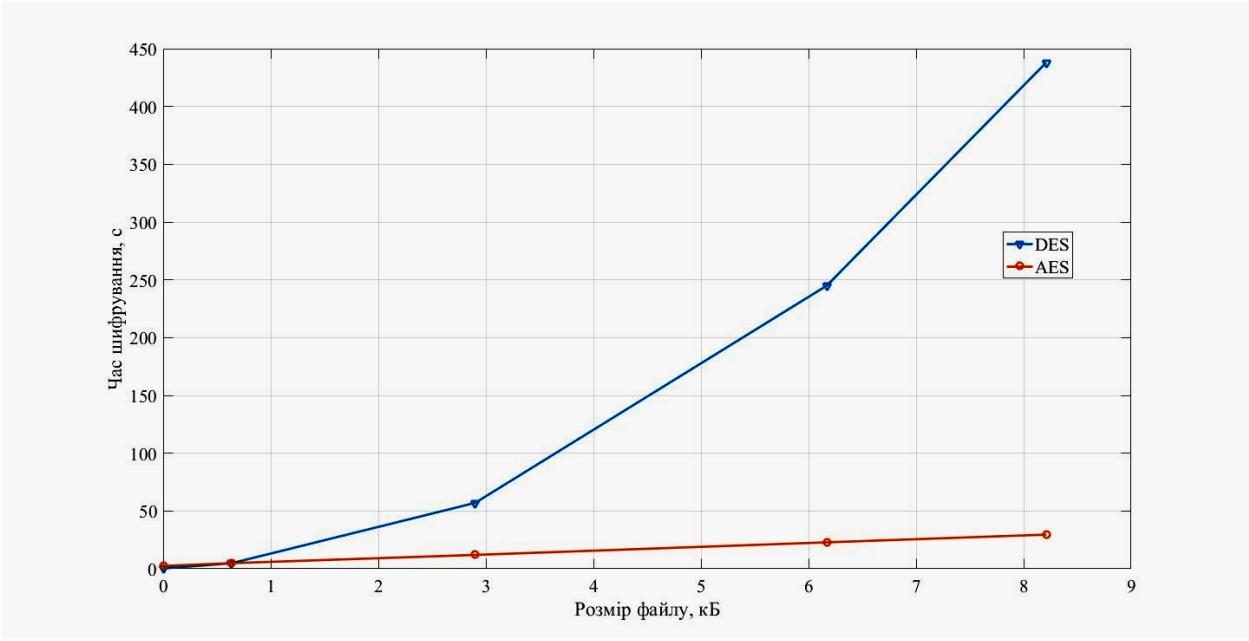

Рисунок 3.5 – Часові витрати на роботу алгоритму для різної довжини повідомлень

Наведені нижче дані і графік (табл. 3.5 і рис. 3.6) відображають ефективність алгоритмів AES і DES проти атаки виду Brute Force (повний перебір). З представлених даних видно, що AES надає більш надійний захист від перебору, ніж DES.

Таблиця 3.5 – Часові витрати на работу прямого перебору (Brute Force) у випадку різної довжини ключа

Довжина ключа, біт	Часові витрати на работу прямого перебору, с	
	DES	AES
8	0,73	1,04
16	7,19	74,75
24	1330,84	3081
32	2900,65	8758,65
40	5768,67	19141,12
48	9765,44	41750,54
56	25789,12	154277,87
64	60789,12	554277,87

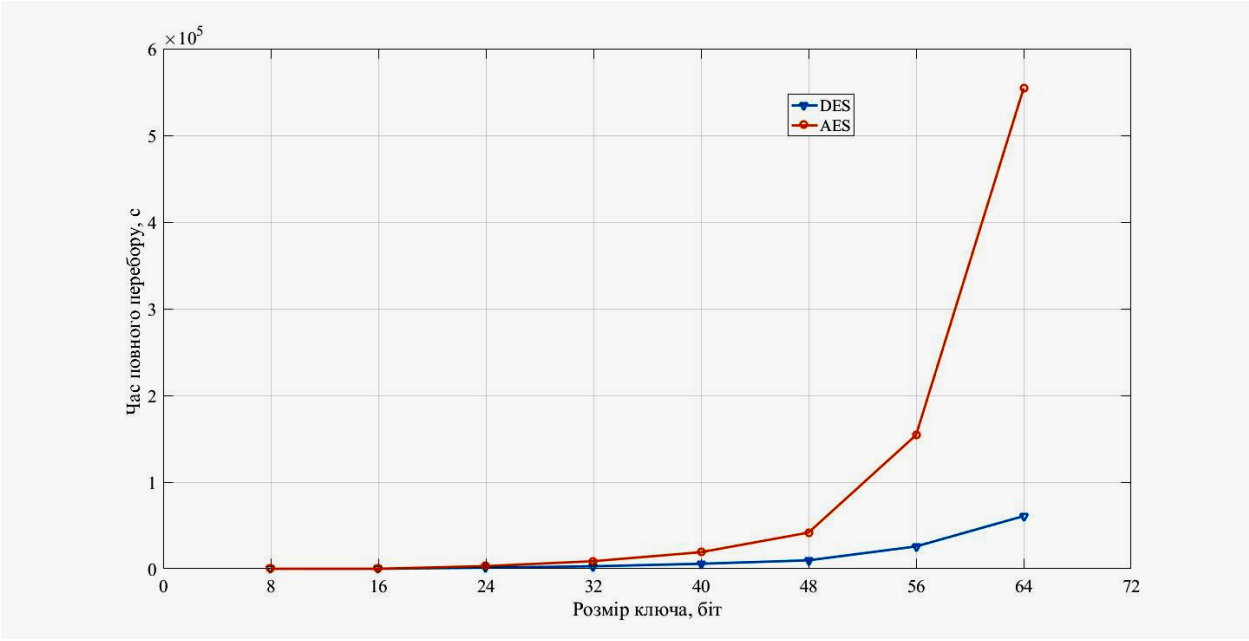

Рисунок 3.6 - Часові витрати на роботу прямого перебору у випадку різної довжини ключа

Результати порівняльного аналізу показують, що на малих обсягах даних алгоритм DES не поступається алгоритме AES за швидкістю. Зі збільшенням обсягу даних DES вимагає великих тимчасових і енергетичних витрат. На сьогоднішній день алгоритм DES вже не відповідає стандартам безпеки для передачі даних, однак, ефективно застосовується для систем де вимагається передача невеликого обсягу даних

3. ТЕОРЕТИЧНА ЧАСТИНА

3.1. Засоби захисту інформації мереж Wi-Fi

Для захисту бездротової мережі Wi-Fi використовуються наступні засоби: Протокол шифрування WEP — це протокол шифрування, що використовує досить нестійкий алгоритм RC4 на статичному ключі. Існує 64 -,128-,256 - і512-бітовешифрування. Чим більше біт використовується для зберігання ключа, тим більше можливих комбінацій ключів, а відповідно більш висока стійкість мережі до злому. Частина WEP-ключа є статичною (40 біт у випадку64-бітногошифрування), а інша частина (24 біта) - динамічною (вектор ініціалізації), вона змінюється в процесі роботи мережі. Основною вразливістю протоколу WEP є те, що вектори ініціалізації повторюються через деякий проміжок часу, і зламнику буде потрібно лише обробити ці повтори і обчислити по них статичну частину ключа. Для підвищення рівня безпеки також додатково до WEP-шифрування використовується стандарт 802.1x або VPN;

Протокол шифрування WPA — це більш стійкий протокол шифрування, ніж WEP, хоча використовується той самий алгоритм RC4. Більш високий рівень безпеки досягається за рахунок використання протоколів TKIP і MIC. TKIP (Temporal Key Integrity Protocol) - протокол динамічних ключів мережі, які змінюються досить часто. При цьому, кожному пристрою також привласнюється ключ, що теж змінюється. MIC (Message Integrity Check) - протокол перевірки цілісності пакетів, захищає від перехоплення пакетів і їх перенаправлення.

Також існує можливість використання 802.1x та VPN, як і у випадку з протоколом WEP. Існує 2 види WPA:

1. WPA-PSK(Pre-SharedKey) - для генерації ключів мережі і для входу в мережу використовується ключова фраза. Оптимальний варіант для домашньої або невеликої офісної мережі;

2. WPA-802.1x- вхід у мережу здійснюється через сервер аутентифікації. Оптимально для мережі великої компанії;

Протокол WPA2 - удосконалення протоколу WPA. На відміну від WPA, використовується більш стійкий алгоритм шифрування AES. За аналогією з WPA, WPA2 також ділиться на два типи: WPA2-PSKiWPA2-802.1x;

Слід звернути увагу на протоколи стандарту безпеки 802.1x. До них відносяться:

EAP (Extensible Authentication Protocol) - протокол розширеної аутентифікації. Використовується спільно з RADIUS - сервером у великих мережах;

TLS (Transport Layer Security) - протокол, який забезпечує цілісність і шифрування переданих даних між сервером і клієнтом, їх взаємну аутентифікацію, запобігаючи перехопленню і підміну повідомлень;

RADIUS (Remote Authentication Dial-InUser Server) - сервер аутентифікації користувачів за логіном і паролем;

VPN (Virtual Private Network) - віртуальна приватна мережа. Цей протокол спочатку був створений для безпечного підключення клієнтів до мережі через загальнодоступні Інтернет-канали. Принцип роботи VPN - створення так званих безпечних «тунелів» від користувача до вузла доступу або сервера. Хоча VPN спочатку був створений не для Wi-Fi, його можна використовувати в будь якому типі мереж. Для шифрування трафіку в VPN найчастіше використовується протокол IPSec.

Також існує де-кілька засобів додаткового захисту мереж Wi-Fi. Серед них слід відмітити наступні:

Фільтрація за MAC адресою. MAC адреса - це унікальний ідентифікатор пристрою (мережного адаптера), «зашитий» в нього виробником. На деякому обладнанні, можливо, задіяти цю функцію і дозволити доступ в мережу необхідним адресам. Це створить додаткову перешкоду зламнику;

Приховування SSID .SSID - це ідентифікатор бездротової мережі. Більшість обладнання дозволяє його приховати, таким чином, при скануванні мережі видно не буде. Але це не дуже серйозна перепона, якщо хакер використовує більш просунутий сканер мереж, ніж стандартна утиліта в Windows; заборона доступу до налаштувань точки доступу або роутера через бездротову мережу. Активувавши цю функцію можна заборонити доступ до налаштувань точки доступу через Wi-Fi мережу, однак, це не захистить мережу від перехоплення трафіку або від проникнення да неї.

3.2. Алгоритм моделі

1. Запуск моделі. Користувачу, щоб розпочати працювати, необхідно запустити модель шифрування ключа Wi-Fi мережі. Після запуску моделі, користувачу відкриється вікно, на якому буде розміщено декілька полів для вводу даних та декілька кнопок управління.
2. Ввід ключа. Першим кроком, необхідно ввести з клавіатури ключ Wi-Fi мережі в поле “Введіть ключ ключа Wi-Fi”.
3. Ввід пароля шифрування. Потім, користувачеві необхідно ввести пароль з 16-ти символів, для шифрування ключа Wi-Fi мережі.
4. Процес шифрування. Після заповнення полів, користувач натискає кнопку “Зашифрувати”, після цього, виконається алгоритм AES і ключ Wi-Fi мережі зашифрується. Потім, користувач зможе безпечно зберігати зашифрований ключ на будь-якому носії, записавши зашифрований ключ та пароль від шифрування в зручному текстовому редакторі.
5. Процес дешифрування. Для того, щоб розшифрувати зашифрований ключ, користувачу необхідно ввести з клавіатури раніше збережений зашифрований ключ WIFI в полі “Введіть зашифрований ключ WI-FI”.
6. Встановлення паролю для шифрування. Користувачу необхідно ввести з клавіатури раніше збережений пароль шифрування ключа, зробити це можна в полі “Введіть пароль для шифрування (16 символів)”.
7. Розшифровка. Потім, користувач натискає на кнопку “Розшифрувати”, модель розшифровує ключ WI-FI мережі за допомогою алгоритму шифрування AES та виводиться розшифрований ключ у полі “Розшифрований ключ WI-FI”.

8. Значення у байтах. Додатково, реалізовано можливість переглянути зашифрований ключ WI-FI мережі в байтах. Користувачу, необхідно просто двічі клацнути по полі зашифрованого ключа.
9. Виведення підказок. Якщо користувач введе в поле паролю шифрування пароль менше ніж 16 символів, йому виведеться спливаюча підказка.
10. Закрити модель. Для завершення роботи з моделлю, користувачеві необхідно натиснути кнопку “закрити”, після закриття, інформація, яка була введена в текстові поля не зберігається.

Алгоритм AES

Алгоритм шифрування AES (Advanced Encryption Standard) - це симетричний блочний шифр, який використовується для захисту інформації. AES працює з блоками даних розміром 128 біт і ключами розміром 128, 192 або 256 біт.

1. Ініціалізація ключа:
   - Вибирається ключ довжиною 128, 192 або 256 біт.
2. Розбиття тексту на блоки:
   - Вхідний текст розбивається на блоки розміром 128 біт.
3. Виконується операція XOR між блоком тексту і ключем раунду (перший раунд використовує початковий ключ).
4. Кожен раунд (крім останнього) складається з чотирьох етапів: SubBytes, ShiftRows, MixColumns і AddRoundKey. Останній раунд не виконує етап MixColumns.
5. Замінюється кожен байт у блоці на відповідний байт у S-Box таблиці заміни.
6. Зсувається кожен рядок блоку на фіксовану кількість байт вліво (1-й рядок - на 1 байт, 2-й - на 2 байти і т.д.).

7. Кожен стовпець блоку перетворюється за допомогою лінійної операції над байтами.

8. Виконується операція XOR між обробленим блоком і ключем раунду.

9. Виконується SubBytes, ShiftRows і AddRoundKey, але без MixColumns.

10. Об'єднуються всі зашифровані блоки для отримання кінцевого зашифрованого тексту.

3.3. Блок-схема роботи моделі

Нижче проілюстровано блок-схему роботи алгоритму шифрування моделі (рис. 3.1)

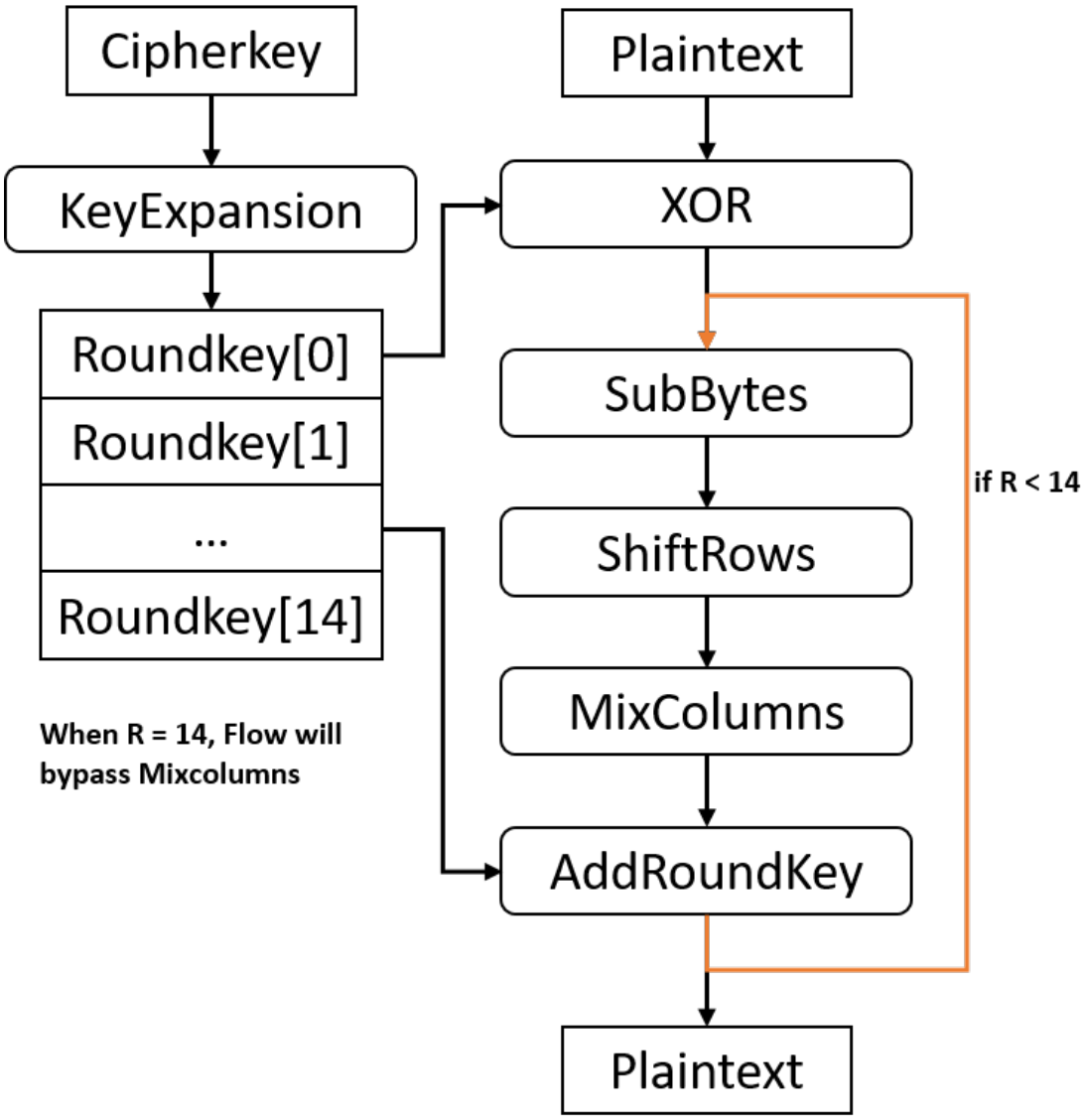

Рисунок 3.1 - блок-схема роботи алгоритму шифрування моделі.

4. ПРАКТИЧНА ЧАСТИНА

4.1. Обґрунтування вибору програмних засобів

Вибір програмних засобів, зокрема мови програмування C#, може бути обґрунтований різними факторами, залежно від конкретного завдання та вимог проекту. Ось кілька аргументів, які можуть впливати на вибір C#:

1. Широкий екосистема та інтеграція зі стеком Microsoft:

- C# розроблена Microsoft, і це забезпечує високий рівень інтеграції з іншими продуктами та технологіями Microsoft, такими як .NET Framework, ASP.NET, Azure, SQL Server та інші.
- Розробники можуть використовувати Visual Studio, потужне інтегроване середовище розробки (IDE), яке підтримує C#.

2. Мова з високим рівнем абстракції:

- C# є мовою програмування з високим рівнем абстракції, що дозволяє розробникам концентруватися на бізнес-логіці та рівні додатку, замість деталей пам'яті чи низькорівневих операцій.

3. Об'єктно-орієнтована мова:

- C# є повністю об'єктно-орієнтованою мовою програмування, що полегшує організацію коду, його перевикористання та підтримку модульності.

4. Безпека та контроль типів:

- C# має систему строгих типів, що дозволяє виявляти помилки на етапі компіляції, а не під час виконання, що сприяє безпеці та стабільності програм.

5. Підтримка мультиплатформенності:

- Сучасні версії C# і .NET (зокрема .NET Core та .NET 5+) надають можливість розробки мультиплатформенних застосунків, які можуть працювати на різних операційних системах.

6. Активна спільнота та ресурси:

- C# має велику та активну спільноту розробників, яка надає підтримку, ресурси, бібліотеки та інструменти для вирішення різних завдань.

7. Можливості для розробки веб-застосунків та служб:

- За допомогою ASP.NET розробники можуть створювати веб-застосунки та служби, використовуючи C# для бекенду.

Загалом, вибір C# може бути обґрунтований його потужними можливостями, широкою підтримкою та інтеграцією в екосистему Microsoft, що робить його відмінним вибором для розробки різноманітних додатків.

4.2. Опис програмної реалізації

Модель шифрування ключів Wi-Fi мережі розроблено з використанням мови програмування C# у середовищі розробки додатків Visual Studio. Проект створено за допомогою Windows Forms. (рис. 4.1)

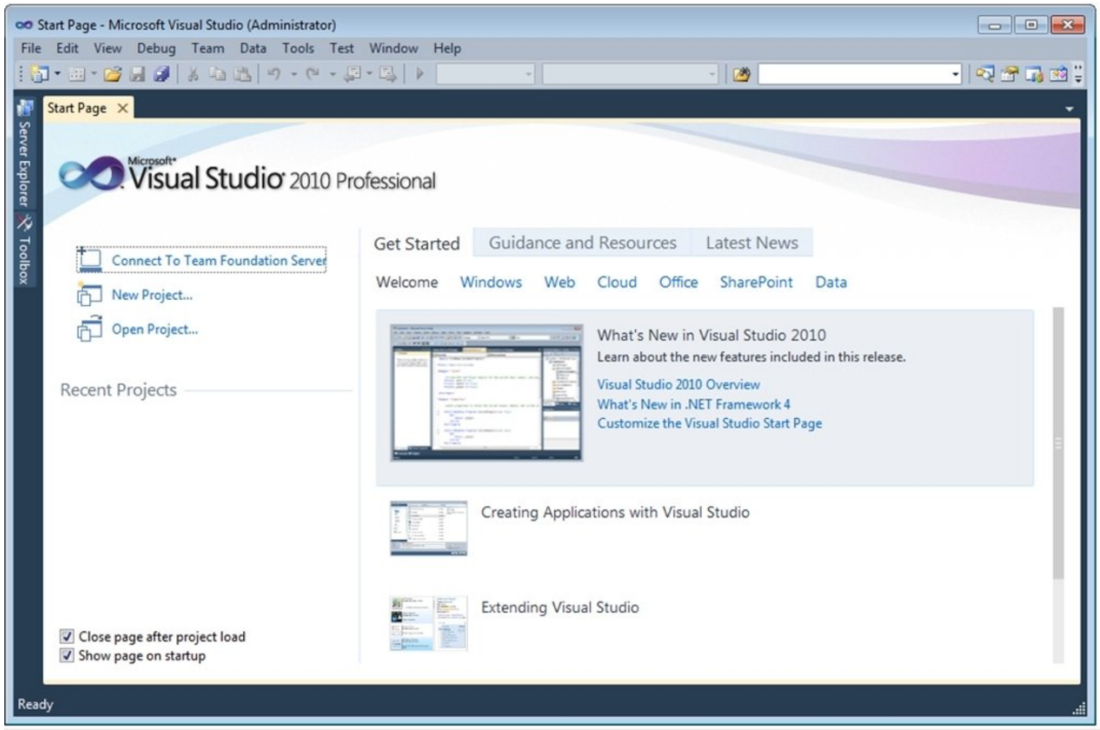

Рисунок 4.1 – процес створення проекту C#

Потім, потрібно придумати та розробити зручний головний інтерфейс моделі (рис. 4.2)

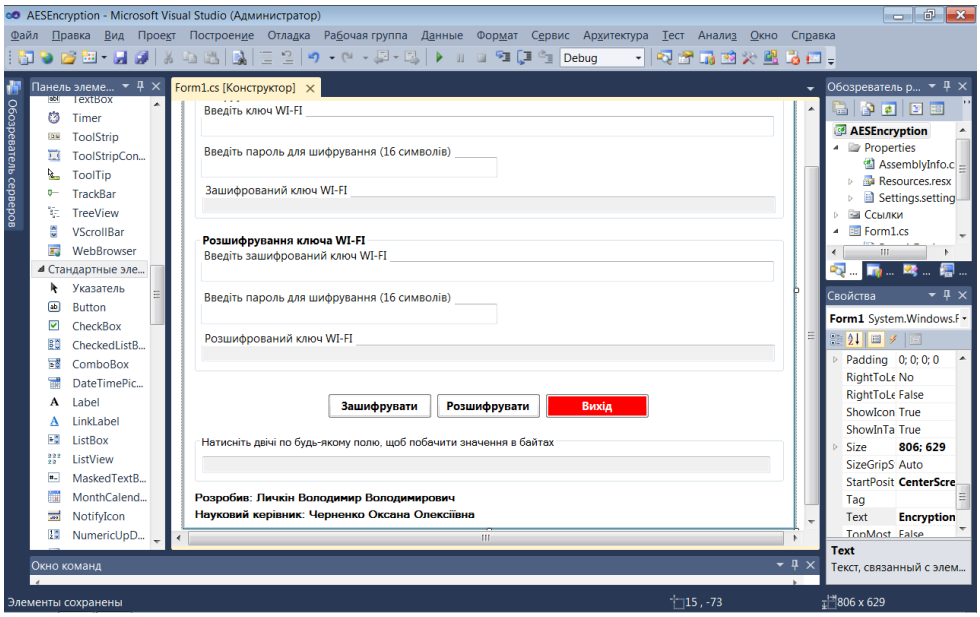

Рисунок 4.2 – головний інтерфейс моделі

Після розробки графічного інтерфейсу, потрібно написати код моделі, щоб кожен елемент працював та виконував свою функцію. (рис. 4.3)

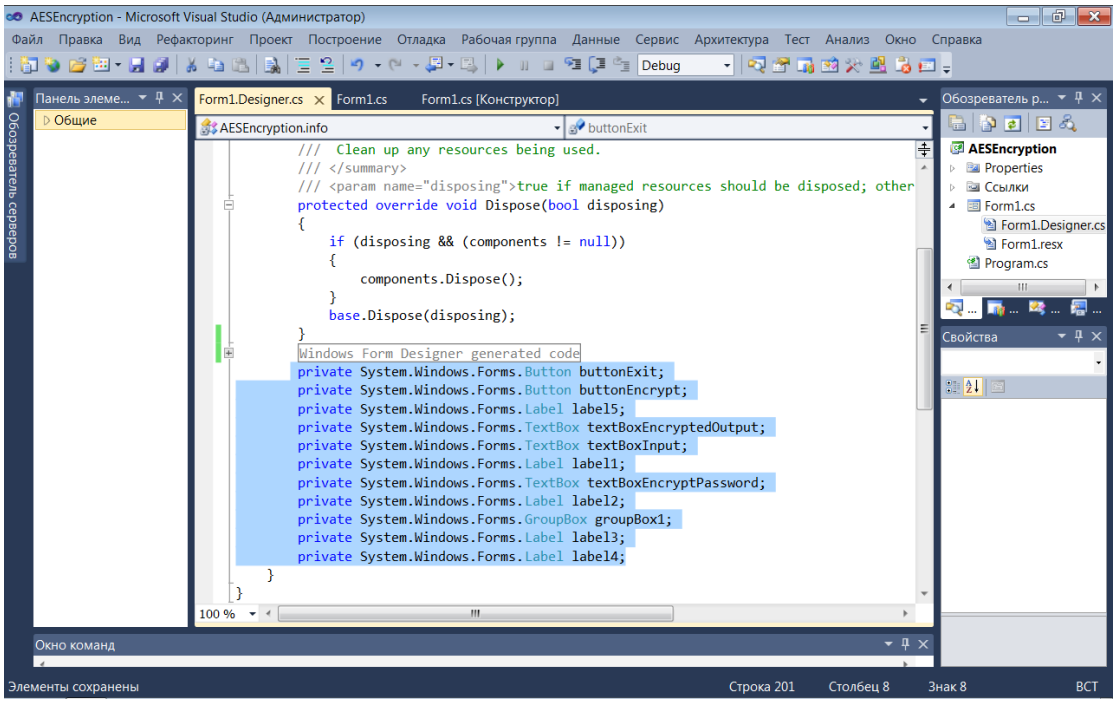

Рисунок 4.3 – Кількість елементів моделі

Далі, розробляється код, для операції з шифруванням ключа за допомогою алгоритму AES (Advanced Encryption Standard). (рис. 4.4)

```
51          private void textBoxInput_TextChanged(object sender, EventArgs e)
52          {
53              textBoxEncryptPassword.Enabled = textBoxInput.Text.Length > 0;
54              buttonEncrypt.Enabled = textBoxInput.Text.Length > 0;
55          }
56
57          private void textBoxEncryptPassword_TextChanged(object sender, EventArgs e)
58          {
59              textBoxInput.Enabled = textBoxInput.Text.Length > 0;
60              buttonEncrypt.Enabled = textBoxInput.Text.Length > 0;
61          }
62
63          private void textBoxEncrypted_TextChanged(object sender, EventArgs e)
64          {
65             // textBoxDcryptPassword.Enabled = textBoxInput.Text.Length > 0;
66              //buttonDecrypt.Enabled = textBoxInput.Text.Length > 0;
67          }
68
69          private void textBoxDcryptPassword_TextChanged(object sender, EventArgs e)
70          {
71             // textBoxEncrypted.Enabled = textBoxInput.Text.Length > 0;
72              //buttonDecrypt.Enabled = textBoxInput.Text.Length > 0;
73          }
74
75          private string Encrypt(string plainText, string Password, byte[] IV)
76          {
77              byte[] Key = Encoding.UTF8.GetBytes(Password);
78
79              // Create a new AesManaged.
80              AesManaged aes = new AesManaged();
81              aes.Key = Key;
82              aes.IV = IV;
83
84              MemoryStream memoryStream = new MemoryStream();
85              CryptoStream cryptoStream = new CryptoStream(memoryStream, aes.CreateEncryptor(), CryptoStreamMode.Write);
86
87              byte[] InputBytes = Encoding.UTF8.GetBytes(plainText);
88              cryptoStream.Write(InputBytes, 0, InputBytes.Length);
89              cryptoStream.FlushFinalBlock();
90
91              byte[] Encrypted = memoryStream.ToArray();
92              // Return encrypted data
93              return Convert.ToBase64String(Encrypted);
94          }
95
96          private string Decrypt(string plaintext, string Password, byte[] IV)
97          {
98              byte[] Key = Encoding.UTF8.GetBytes(Password);
99
100             // Create a new AesManaged.
101             AesManaged aes = new AesManaged();
102             aes.Key = Key;
103             aes.IV = IV;
104
105             MemoryStream memoryStream = new MemoryStream();
106             CryptoStream cryptoStream = new CryptoStream(memoryStream, aes.CreateDecryptor(), CryptoStreamMode.Write);
Restricted Mode  ⊗ 0 ⚠ 0  0
```

Рисунок 4.4 – реалізація алгоритму AES.

Якщо детальніше розглянути програмний код моделі, то з самого початку, рядки *using Statements* вказують на те, які простори імен і класи використовуються в програмі.

using System;

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Security.Cryptography;
using System.IO;
```

Далі, прописаний статичний вектор ініціалізації (IV) для використання в алгоритмі AES.

```
public partial class Form1 : Form
{
    private static byte[] IV = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
    public Form1()
    {
        InitializeComponent();
    }
```

Конструктор класу форми, який ініціалізує компоненти форми. Обробник події Exit Button Event Handler для кнопки "Exit", яка закриває форму при натисканні. Обробник події Encrypt Button Event Handler для кнопки "Encrypt", який викликає метод Encrypt для зашифрування введеного тексту. Обробник події Decrypt Button Event Handler для кнопки "Decrypt", який викликає метод Decrypt для дешифрування введеного зашифрованого тексту.

```
    private void textBoxInput_TextChanged(object sender, EventArgs e)
    {
        textBoxEncryptPassword.Enabled = textBoxInput.Text.Length > 0;
        buttonEncrypt.Enabled = textBoxInput.Text.Length > 0;
```

```
    }

    private void textBoxEncryptPassword_TextChanged(object sender,
EventArgs e)
    {
        textBoxInput.Enabled = textBoxEncryptPassword.Text.Length > 0;
        buttonEncrypt.Enabled = textBoxEncryptPassword.Text.Length > 0;
    }
    private void textBoxEncrypted_TextChanged(object sender, EventArgs e)
    {
        textBoxDcryptPassword.Enabled = textBoxEncrypted.Text.Length > 0;
        buttonDecrypt.Enabled = textBoxEncrypted.Text.Length > 0;
    }
    private void textBoxDcryptPassword_TextChanged(object sender,
EventArgs e)
    {
        textBoxEncrypted.Enabled = textBoxDcryptPassword.Text.Length > 0;
        buttonDecrypt.Enabled = textBoxDcryptPassword.Text.Length > 0;
    }
```

4.3. Опис роботи моделі

Після запуску моделі шифрування, користувач бачить перед собою вікно, на якому розміщено сім полів для введення даних та три кнопки. (рис. 4.5)

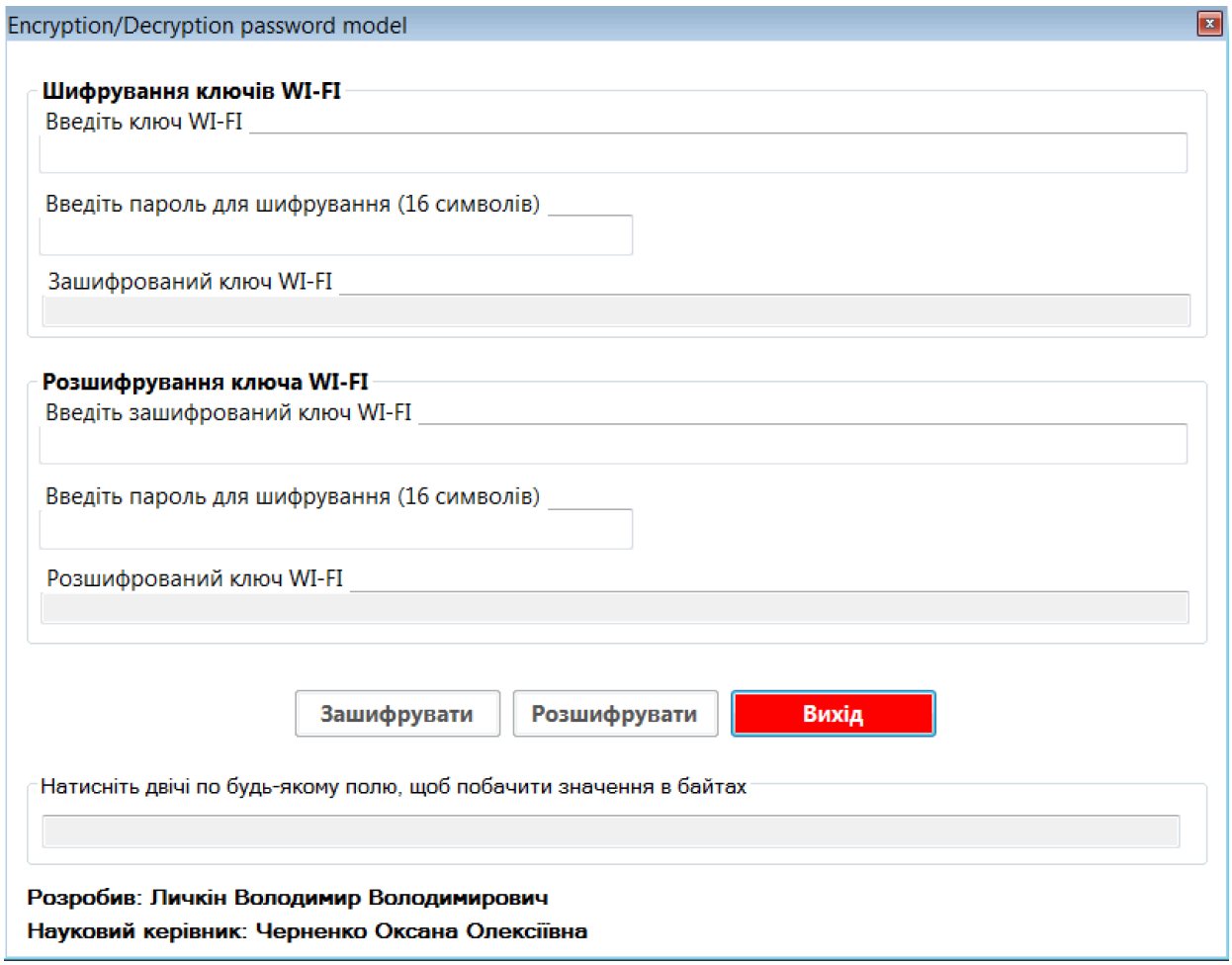

Рисунок 4.5 – Інтерфейс моделі

Користувачу в першому полі необхідно прописати wifi ключ, який необхідно зашифрувати. Потім, користувачеві потрібно в другому текстовому полі встановити пароль шифрування і натиснути кнопку “зашифрувати”. Після цих кроків, в третьому полі користувач отримає зашифрований ключ, його необхідно скопіювати та зберегти в надійному місці разом з паролем, для того щоб потім дешифрувати. (рис. 4.6)

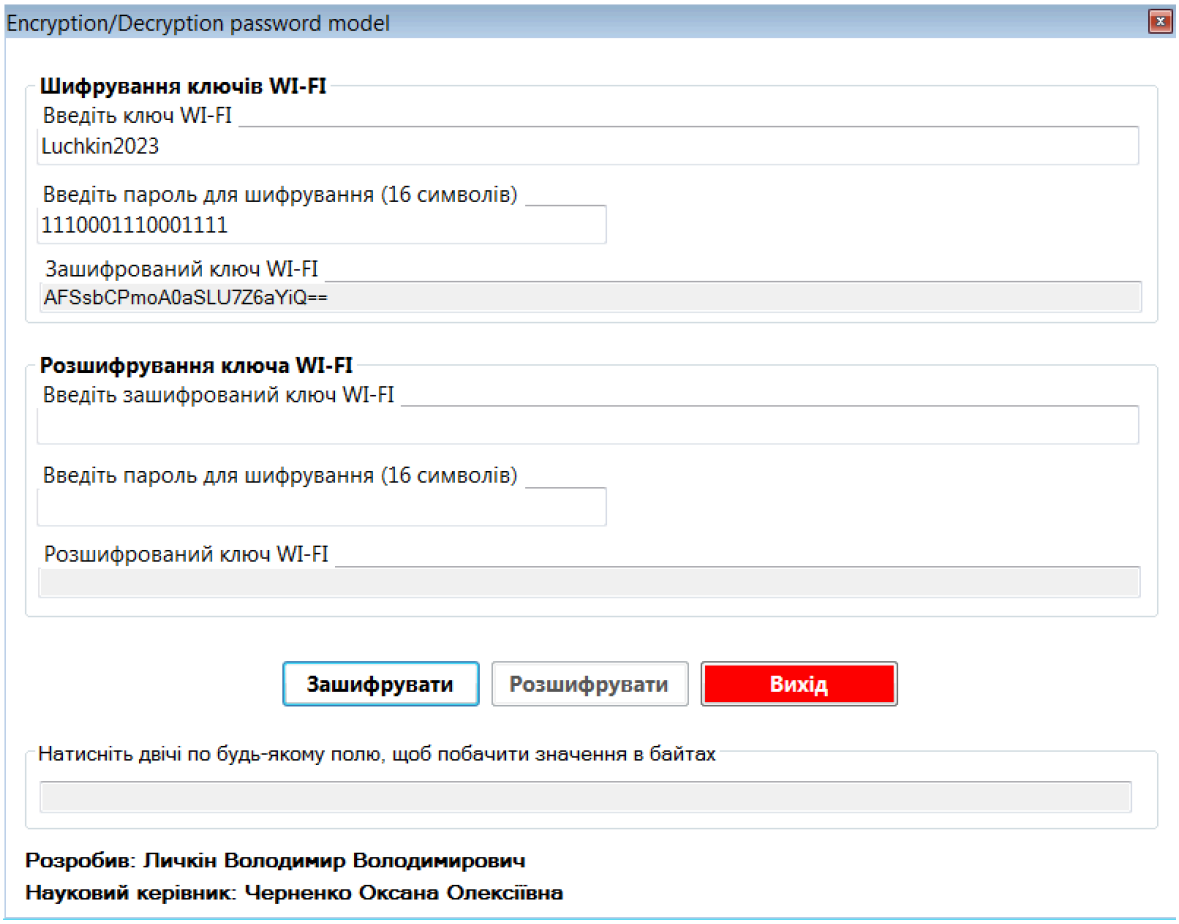

Рисунок 4.6 – Процес шифрування ключа за допомогою алгоритму AES.

В будь-який момент, при потребі у користувача є можливість розшифрувати зашифрований ключ WIFI. Для того, щоб розшифрувати зашифрований ключ, користувачу необхідно ввести з клавіатури раніше збережений зашифрований ключ WIFI в полі “Введіть зашифрований ключ WI-FI” (рис. 4.7)

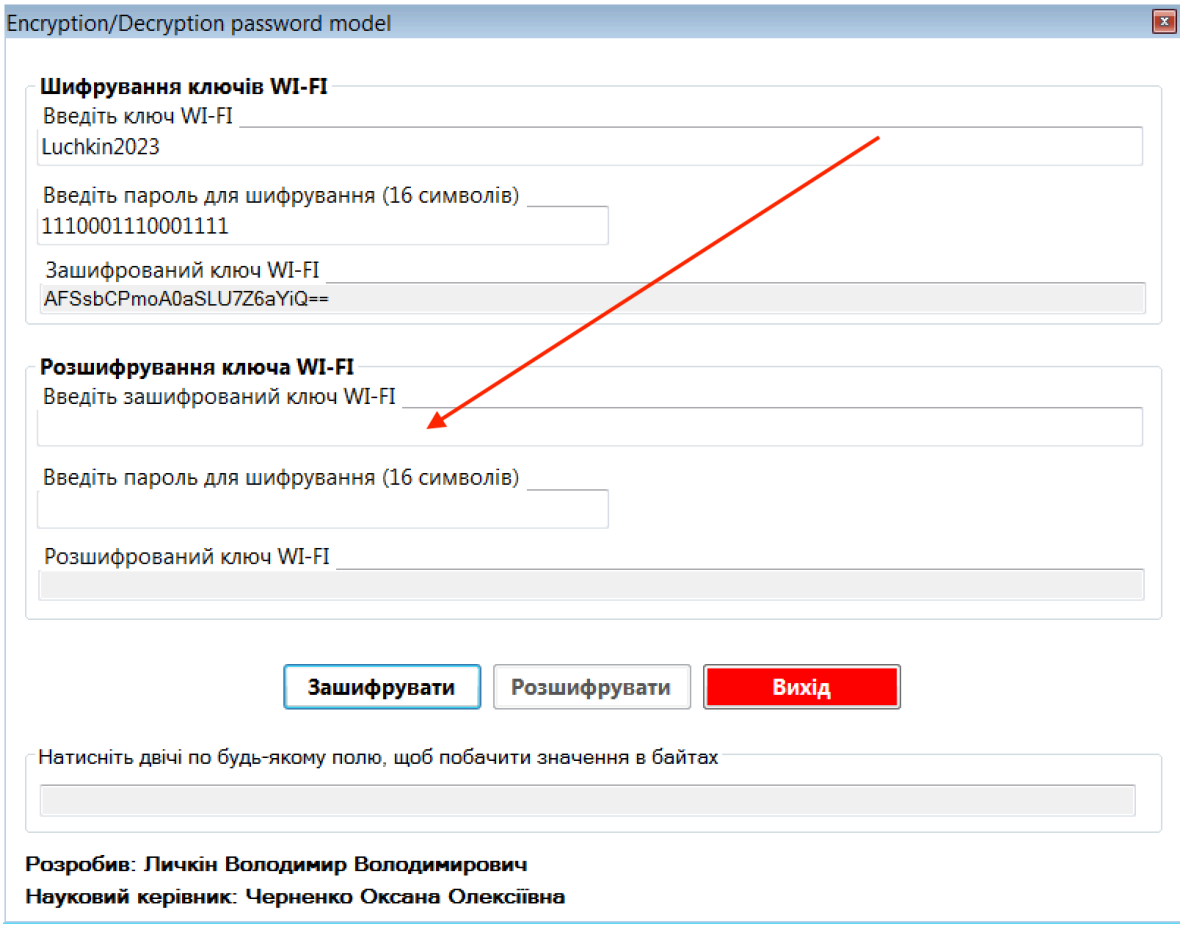

Рисунок 4.7 – опис поля для введення зашифрованого ключа.

Наступним кроком користувачу необхідно ввести з клавіатури раніше збережений пароль шифрування ключа, зробити це можна в полі “Введіть пароль для шифрування (16 символів)” (рис. 4.8)

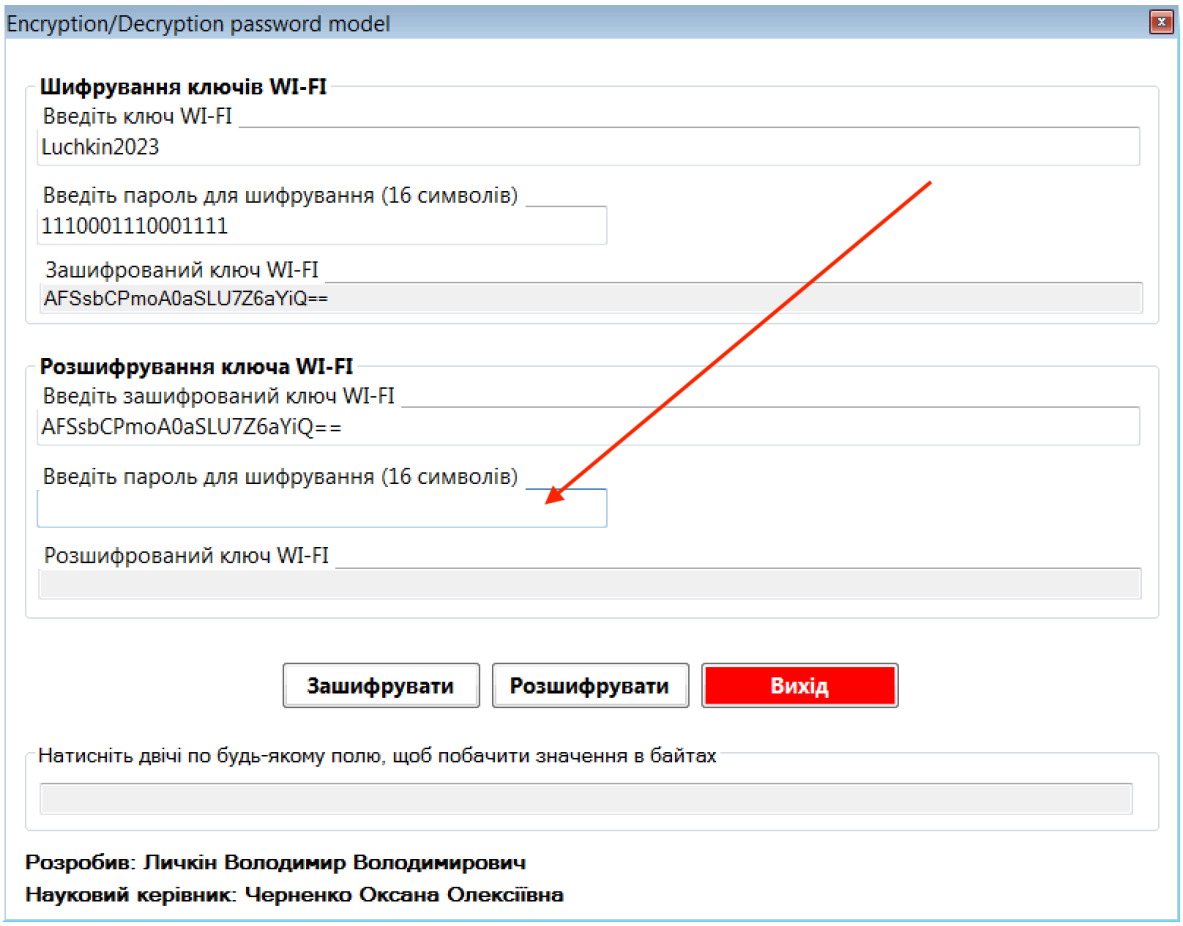

Рисунок 4.8 – опис поля для введення паролю зашифрованого ключа.

Потім, користувач натискає на кнопку “Розшифрувати”, модель розшифровує ключ WI-FI мережі за допомогою алгоритму шифрування AES та виводиться розшифрований ключ у полі “Розшифрований ключ WI-FI” (рис. 4.9)

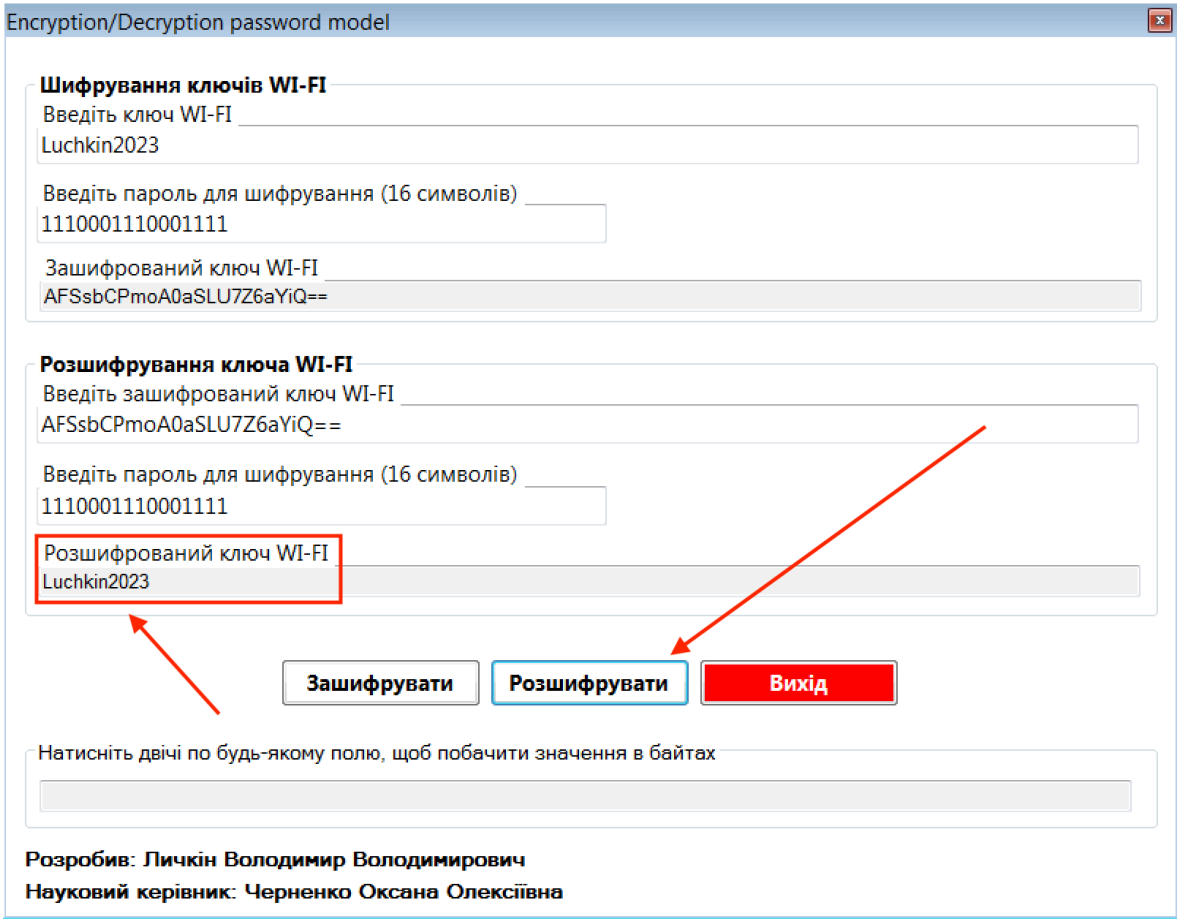

Рисунок 4.9 – Розшифрований ключ WI-FI.

Додатково, реалізовано можливість переглянути зашифрований ключ WI-FI мережі в байтах. Користувачу, необхідно просто двічі клацнути по полі зашифрованого ключа. (рис. 5.0)

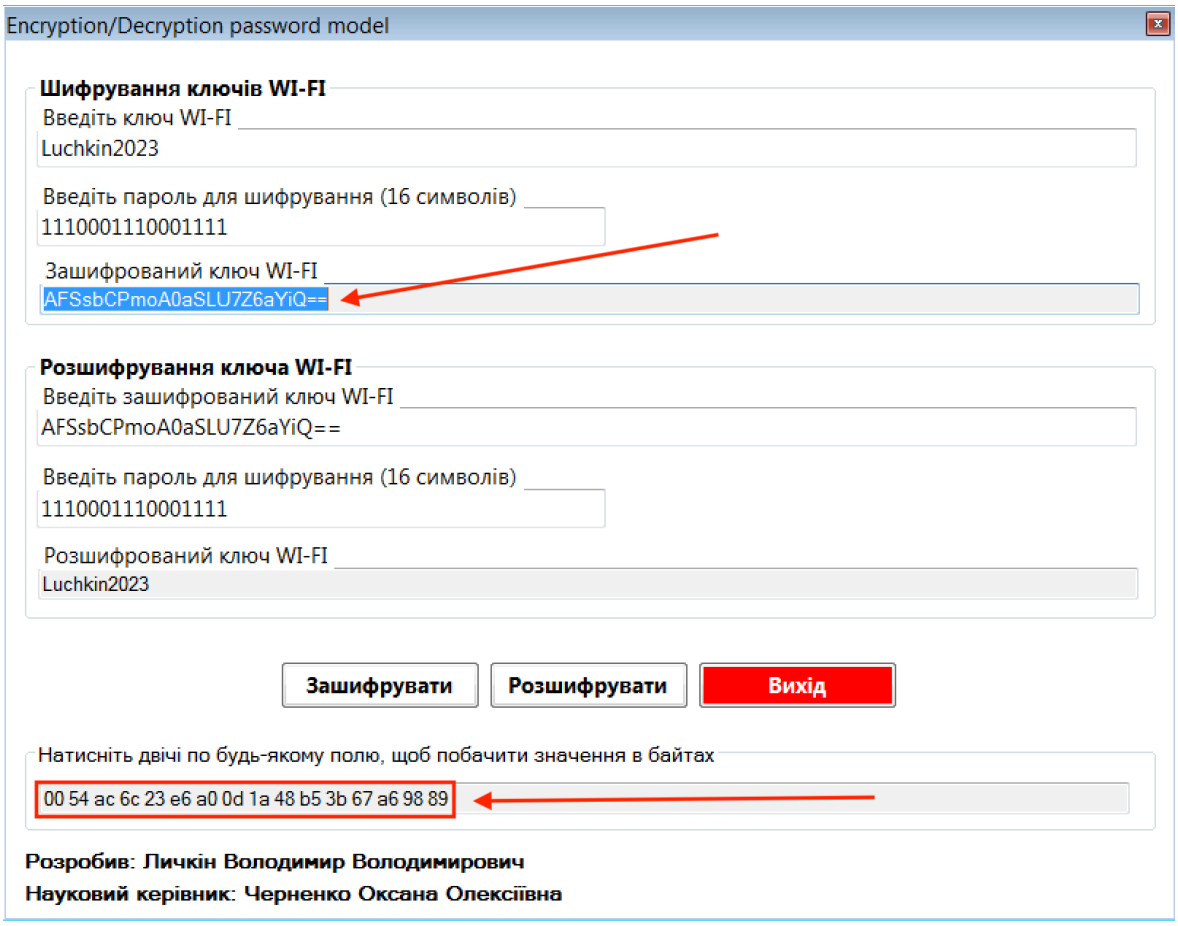

Рисунок 5.0 – Значення в байтах.

Якщо користувач введе в поле паролю шифрування пароль менше ніж 16 символів, йому виведеться спливаюча підказка. (рис. 5.1)

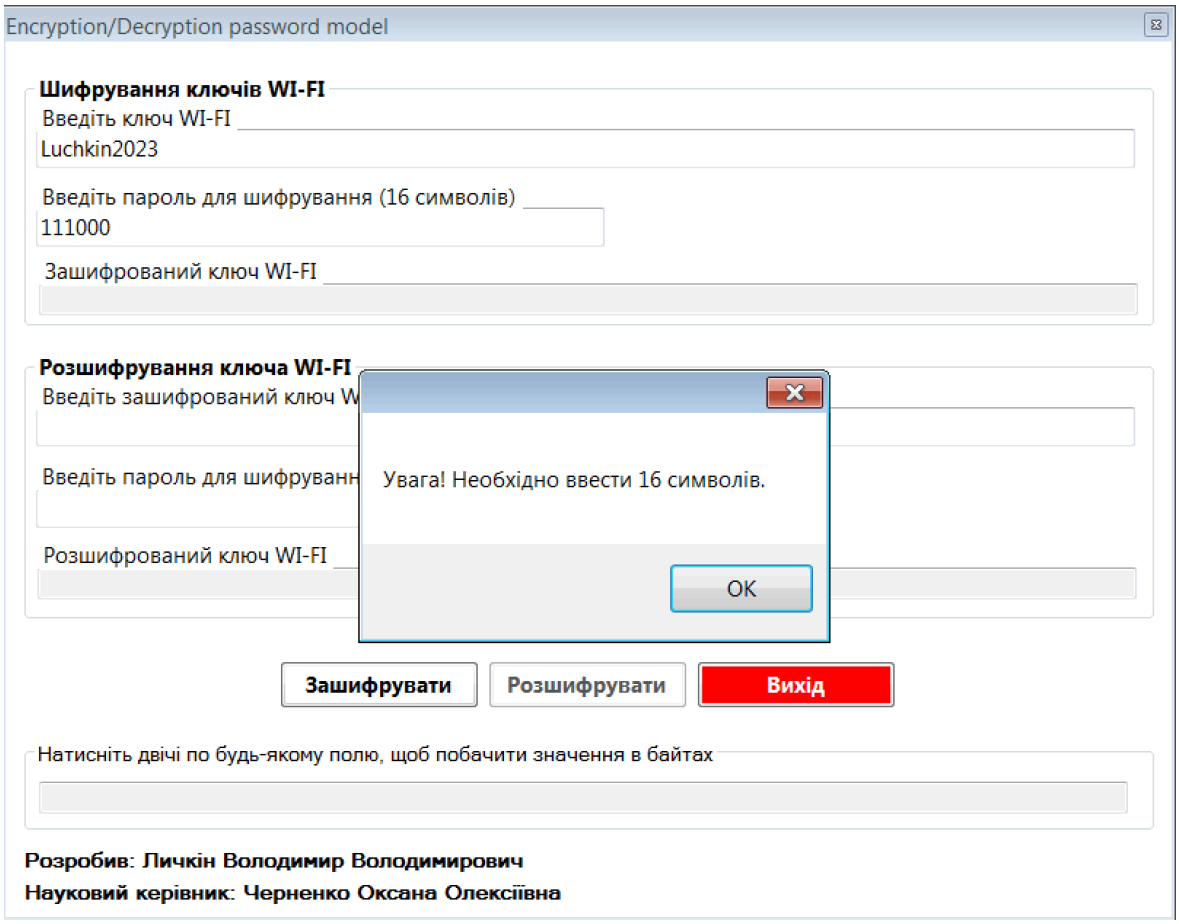

Рисунок 5.1 – Вивід спливаючого повідомлення.

ВИСНОВКИ

Отже, в сучасному світі, насиченому високотехнологічними засобами та мережевими взаємодіями, важливість захисту інформації стає критичною. За нинішнього рівня цифрового розвитку зростає ймовірність кіберзагроз, що вимагає ретельного управління інформаційною безпекою.

Зловмисники в мережі постійно вдосконалюють свої методи, ставлячи під загрозу як особисті дані, так і важливу корпоративну інформацію. Поглиблений аналіз кіберзагроз вказує на необхідність розвитку та вдосконалення заходів безпеки для забезпечення стійкості перед цими викликами.

Забезпечення інформаційної безпеки вимагає постійного оновлення стратегій та застосування передових технологій. Необхідно вдосконалювати процеси аутентифікації, шифрування та моніторингу.

Управління інформаційною безпекою несе відповідальність не тільки для приватних компаній, але й для урядових органів та громадськості в цілому. Співпраця всіх сторін є ключем до успішного протистояння кіберзагрозам та забезпечення сталої інформаційної безпеки.

В результаті виконання кваліфікаційної роботи, було розроблено програмний продукт, модель, за допомогою якої можна зашифрувати ключ WIFI мережі.

Кваліфікаційну роботу успішно завершено, створено модель захисту інформації wifi мережі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Черненко О. О. Курсовий проєкт із фаху: методичні рекомендації щодо оформлення пояснювальних записок до курсового проєкту для студентів спеціальності 122 Комп'ютерні науки освітня програма «Комп'ютерні науки» ступеня бакалавра, магістра / О. О. Черненко. – Полтава : ПУЕТ, 2022. – 58 с. – 1 електрон. опт. диск (CVD-ROM).
2. Захист мереж Wi-Fi [Електронний ресурс]. – Режим доступу: https://wiki.cuspu.edu.ua/index.php/%D0%97%D0%B0%D1%85%D0%B8%D1%81%D1%82_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6_Wi-Fi
3. Засоби захисту інформації мереж Wi-Fi [Електронний ресурс]. – Режим доступу: https://studfile.net/preview/9649992/page:2/
4. Обґрунтування методів захисту мереж WI-FI [Електронний ресурс]. – Режим доступу: https://elartu.tntu.edu.ua/bitstream/lib/28797/2/TPARP_2019_Protsyk_P_P-Reference_to_protection_105-107.pdf
5. ОСНОВИ МОВИ ПРОГРАМУВАННЯ C# [Електронний ресурс]. – Режим доступу: https://moodle.znu.edu.ua/pluginfile.php/514308/mod_resource/content/4/Lec2ukr.pdf
6. Захист локальної мережі [Електронний ресурс]. – Режим доступу: https://infotel.ua/zahist-lokalnoi-merezhi
7. Захист даних в WiFi мережах [Електронний ресурс]. – Режим доступу: http://isearch.kiev.ua/ru/searchpracticeru/-internetsecurity-ru/828-security-wifi
8. Алгорим шифрування AES [Електронний ресурс]. – Режим доступу: https://learn.ztu.edu.ua/pluginfile.php/181600/mod_resource/content/1/%D0%9B%D0%B5%D0%BA%D1%86%D1%96%D1%8F5.pdf
9. Що таке AES-шифрування [Електронний ресурс]. – Режим доступу: https://memory.net.ua/info/aes-shifruvannja

10. AES (Advanced Encryption Standard instructions) [Електронний ресурс]. – Режим доступу: https://www.chaynikam.info/ukr/cpu_aes.html
11. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання: ДСТУ 7.1-2006. – [Чинний від 2007-07-01]. – К. : Держспоживстандарт України, 2007. – 47 с.
12. Microsoft Visual Studio [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Microsoft_Visual_Studio
13. C# development with Visual Studio [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/?view=vs-2022
14. C# [Електронний ресурс]. – Режим доступу: https://www.w3schools.com/cs/cs_getstarted.php
15. Visual Studio and C# [Електронний ресурс]. – Режим доступу: https://www.halvorsen.blog/documents/programming/csharp/csharp.php
16. Створення програми на C# в Visual Studio [Електронний ресурс]. – Режим доступу: https://learn.ztu.edu.ua/mod/page/view.php?id=9976
17. Getting Started With C# on Visual Studio Code [Електронний ресурс]. – Режим доступу: https://medium.com/@adebiyiadedotun9/getting-started-with-c-on-visual-studio-code-5a76dcca1110
18. Randolph Visual Studio 2010 for professionals // Randolph, Nick, Gardner, David, Minutillo, Michael, Anderson, Chris.: Trans. with English - M.: LLC "I.D. Williams", 2011. - 1184 p.
19. ReSharper: The Visual Studio Extension for .NET [Електронний ресурс]. – Режим доступу: https://www.jetbrains.com/resharper/
20. Install Visual Studio [Електронний ресурс]. – Режим доступу: https://www.csharptutorial.net/csharp-tutorial/install-visual-studio/

ДОДАТОК А. КОД ПРОГРАМИ