Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ДОПУСТИТИ ДО ЗАХИСТУ:
завідувачка кафедри кібербезпеки
та захисту інформації
_____Наталія ЛУКОВА-ЧУЙКО
«14» червня 2022р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

дипломної роботи
бакалавра
(назва освітнього ступеня)

галузь знань 12 Інформаційні технології
(шифр і назва галузі знань)

спеціальність 125 Кібербезпека
(код і назва спеціальності)

освітня програма Кібербезпека
(назва освітньої програми)

на тему: «Дослідження симетричних алгоритмів шифрування на прикладі потокового алгоритму А5»

Виконавець: студентка IV курсу, групи КБ-41

_____ **Катерина ВЕНГРИНОВСЬКА**
(підпис) (прізвище ім'я по-батькові)

	Прізвище, ініціали	**Підпис**
Керівник	Андрій ФЕСЕНКО	

Нормоконтроль	Олександр ТОРОШАНКО	

Київ 2022

Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ЗАТВЕРДЖЕНО:
завідувачка кафедри кібербезпеки
та захисту інформації
_____Наталія ЛУКОВА-ЧУЙКО
«01» листопада 2021 р.

ЗАВДАННЯ
на виконання дипломної роботи

спеціальності 125 Кібербезпека
(код і назва спеціальності)

освітньої програми Кібербезпека
(назва освітньої програми)

Студентці **КБ-41** **Венгриновська Катерина Володимирівна**
(група) (прізвище ім'я по-батькові)

Тема дипломної роботи Дослідження симетричних алгоритмів шифрування на прикладі потокового алгоритму А5

1. ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Тема дипломної роботи затверджена на засіданні кафедри кібербезпеки та захисту інформації протокол №5 від 29.10.2021 р.

2. ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Структура симетричного шифрування та принцип роботи алгоритмів, алгоритми забезпечення безпеки в системі GSM, структура та принцип потокових алгоритмів

3. ЗМІСТ РОЗРАХУНКОВО-ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

Симетричні алгоритми шифрування, класифікація симетричних алгоритмів шифрування, блокові алгоритми шифрування, потокові алгоритми шифрування, сімейство алгоритмів А5, структура та принцип роботи алгоритму А5, відмінності алгоритмів сімейства А5, програмна реалізація алгоритму шифрування А5/1

4. ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Практична цінність Реалізація потокового алгоритму шифрування A5/1 та визначення сфери застосування

5. ДАТА ВИДАЧІ ЗАВДАННЯ

Дата видачі завдання: 01 листопада 2021 року

Завдання видав _____ (підпис) Андрій ФЕСЕНКО (ініціали, прізвище)

Завдання прийняв до виконання _____ (підпис) Катерина ВЕНГРИНОВСЬКА (ініціали, прізвище)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів робіт	Строки виконання робіт (початок-кінець)	Відмітка про виконання
1	Уточнення постановки задачі	01.11.2021 – 30.01.2022	*виконано*
2	Аналіз літератури	31.01.2022 – 20.02.2022	*виконано*
3	Розгляд класифікації симетричних алгоритмів шифрування	21.02.2022 – 06.03.2022	*виконано*
4	Дослідження блокових алгоритмів	07.03.2022 – 20.03.2022	*виконано*
5	Дослідження потокових алгоритмів	21.03.2022 – 03.04.2022	*виконано*
6	Розгляд сімейства алгоритмів A5	04.04.2022 – 17.04.2022	*виконано*
7	Дослідження структури та принципу роботи алгоритмів A5	18.04.2022 – 01.05.2022	*виконано*
8	Розгляд важливих складових алгоритму A5/1 для його реалізації	02.05.2022 – 15.05.2022	*виконано*
9	Програмна реалізація алгоритму A5/1	16.05.2022 – 01.06.2022	*виконано*
10	Оформлення пояснювальної записки	02.06.2021 – 08.06.2021	*виконано*
11	Підготовка до захисту	09.06.2021 – 13.06.2022	*виконано*

Завдання видав _____ (підпис) Андрій ФЕСЕНКО (ініціали, прізвище)

Завдання прийняв до виконання _____ (підпис) Катерина ВЕНГРИНОВСЬКА (ініціали, прізвище)

Термін подання дипломної роботи до ЕК 08 червня 2022 року

УДК 004.056.5

РЕФЕРАТ

Пояснювальна записка дипломної роботи складається зі вступу, трьох розділів, загальних висновків, списку використаних джерел та додатку. Основний текст займає 59 сторінок, включає в себе зміст, вступ, три розділи дипломної роботи, висновки та список джерел. Крім того, робота містить 2 додатки із загальною кількістю сторінок 10. У пояснювальній записці дипломної роботи міститься 12 рисунків і 2 таблиці.

Метою роботи є розроблення моделі захищеного каналу передачі інформації в безпілотному літальному апараті з використанням симетричного потокового алгоритму А5/1.

Для досягнення зазначеної мети поставлено наступні завдання:

- розглянути симетричні алгоритми шифрування;
- розглянути потоковий алгоритм шифрування А5;
- програмно реалізувати алгоритм шифрування А5/1;
- розробити модель технічної реалізації захищеного каналу передачі інформації в безпілотному літальному апараті використовуючи алгоритм А5/1.

Об'єктом дослідження є процес захисту інформації за допомогою симетричних алгоритмів шифрування та захист інформації у мобільних системах за допомогою потокового алгоритму шифрування.

Предметом дослідження є симетричні алгоритми шифрування та безпосередньо потоковий алгоритм шифрування А5, його структура, принцип роботи та застосування.

Практичною цінністю отриманих результатів є реалізація потового алгоритму шифрування А5/1 для забезпечення захищеного каналу зв'язку між безпілотним літальним апаратом та пунктом управління.

Ключові слова: симетричні алгоритми шифрування, блокові алгоритми шифрування, потокові алгоритми шифрування, криптостійкість, вразливості.

ЗМІСТ

ВСТУП

Актуальність. В сучасному світі проблема конфіденційності даних досі залишається відкритим питанням. На сьогоднішній день винайдено багато елементів захисту інформації, які повинні забезпечувати безпеку базовим властивостям інформації, тобто цілісність, конфіденційність та доступність. Перша властивість інформації відповідає за незмінність даних впродовж їх життєвого циклу, друга властивість – за унеможливлення отримання даних сторонніми особами, третя властивість – передбачає отримання інформації у будь-який момент користувачем за відносно прийнятний час. Невід'ємною частиною для забезпечення цих властивостей є шифрування даних. Насамперед це захист від порушення конфіденційності інформації.

Шифрування умовно можна розділити на дві групи – це симетричні алгоритми шифрування та асиметричні алгоритми шифрування. Перший тип широко використовується, оскільки забезпечує відносно високу ефективність та може застосовуватися для будь-якого типу інформації. Взагалі шифрування є невід'ємною частиною нашого життя, воно використовується як у банківській сфері, в соціальних мережах, так і в системах мобільного зв'язку для шифрування інформації, що передається між користувачем системи та базовою станцією мережі. Також сьогодні є дуже важливою умовою створення захищеного каналу передачі даних для отримання інформації. Ця система повинна задовольняти потреби як для обміну військовою інформацією, так і цивільною. Для такої системи можуть бути використані потокові алгоритми шифрування даних, оскільки вони є швидкими та легкими в реалізації. Системи з захищеним каналом передачі даних є більш захищеними від прослуховування сторонніми особами, ніж ті, де передача інформації відбувається без шифрування. Для розшифрування інформації зловмиснику знадобиться певний час, ця умова і забезпечує секретність інформації. Тому для обміну повідомленнями слід використовувати захищені канали передачі даних.

Метою дипломної роботи є розроблення моделі захищеного каналу передачі інформації в безпілотному літальному апараті з використанням симетричного потокового алгоритму А5/1.

Для досягнення вказаної мети дипломної роботи поставлені такі завдання:

- розглянути симетричні алгоритми шифрування, а саме їх класифікацію, найвідоміших представників та принцип їх роботи, переваги та недоліки цього типу шифрування;
- розглянути потоковий алгоритм шифрування А5, а саме його історію створення, представників сімейства, сферу застосування, принцип роботи та відмінності алгоритмів, криптостійкість, відомі атаки, переваги та недоліки;
- програмно реалізувати потоковий алгоритм шифрування А5/1, описати сферу застосування потокового алгоритму;
- розробити модель технічної реалізації захищеного каналу передачі інформації в безпілотному літальному апараті використовуючи алгоритм шифрування А5/1.

Об’єкт дослідження – це процес захисту інформації за допомогою симетричних алгоритмів шифрування та захист інформації у мобільних системах за допомогою потокового алгоритму шифрування.

Предмет дослідження – це симетричні алгоритми шифрування та безпосередньо потоковий алгоритм шифрування А5, його структура, принцип роботи та застосування.

Методи дослідження, що використовуються у дипломній роботі: аналіз та порівняння.

Практична цінність дипломної роботи полягає у реалізації потокового алгоритму шифрування А5/1 для забезпечення захищеного каналу зв’язку між безпілотним літальним апаратом та пунктом управління.

РОЗДІЛ 1
СИМЕТРИЧНІ АЛГОРИТМИ ШИФРУВАННЯ

1.1 Криптографічні алгоритми

Криптографічний алгоритм – це послідовність процесів, що використовуються для шифрування та розшифрування інформації в криптографічних системах. Послідовність цих процесів гарантує, що доступ до інформації не отримають сторонні люди. Криптографічні алгоритми широко використовуються в сучасному житті, для шифрування даних, аутентифікації, цифрового підпису, безпечних фінансових транзакцій, обміну даними по незахищених каналах передачі інформації та у багатьох інших сферах.

Шифрування передбачає перетворення відкритого (доступного для читання) тексту у зашифрований (нечитабельний) текст, це необхідно для захисту даних від стороннього втручання. Для того щоб відновити первинний вигляд повідомлення необхідно розшифрувати дані. Як шифрування, так і розшифрування працюють за допомогою криптографічних алгоритмів.

Існує багато різноманітних типів криптографічних алгоритмів, які можна поділити на три групи: симетричні алгоритми, асиметричні алгоритми та хеш-функції.

Симетричні алгоритми шифрування – це алгоритми шифрування, що використовують один і той же самий ключ для шифрування та розшифрування інформації. Ключ повинен бути секретним. Основним призначенням симетричного шифрування є шифрування великих об’ємів даних з високою швидкістю. Даний алгоритм шифрування широко використовується для приховування інформації від небажаного використання сторонніми особами. Важливим є те, що отримувачу інформації відомий алгоритм шифрування та секретний ключ.

Асиметричні алгоритми шифрування – це алгоритми шифрування, що використовують два ключі для шифрування та розшифрування інформації, їх

називають відкритим та закритим ключем. Відкритий ключ використовується для шифрування інформації, а закритий ключ для розшифрування. Жоден з ключів не може бути обчислений з іншого за прийнятний час. Такий алгоритм шифрування також називають шифруванням з відкритим ключем. Ці алгоритми використовуються для систем аутентифікації користувачів, контролю цілісності даних, важливою сферою застосування є електронно-цифровий підпис. Оскільки використання двох ключів ускладнює процес шифрування та розшифрування, тому асиметричні алгоритми забезпечують більшу безпеку даних. Найбільш відомими алгоритмами асиметричного шифрування є алгоритм Діффі-Хеллмана та RSA.

Хеш-функція – це функція, що перетворює вхідні дані довільного розміру в дані фіксованої довжини, при цьому не використовується ключ, як в розглянутих алгоритмах шифрування, також називаються хешуванням. Важливою вимогою хеш-функцій є неможливість відтворити початковий текст знаючи відповідне йому хеш значення. Також неможливою є існування однакових хеш значень для різних відкритих даних. Широко використовуються для шифрування паролів та перевірки цілісності даних. Найбільш відомими хеш-функціями є: MD4, MD5, SHA256, SHA512.

1.2 Класифікація симетричних алгоритмів

Алгоритми, що належать до симетричного шифрування можна розділити на потокові та блокові алгоритми шифрування [1]. Головною відмінністю цих типів шифрування є розмір блоку інформації. В блоковому шифруванні одиницею кодування є блок, що складається з певної кількості байтів, а результат процесу шифрування підпорядковується усім вихідним байтам цього блоку шифрування, а в потоковому шифруванні – один біт та результат не підпорядковується раніше вхідному потоку інформації [2].

Потокові алгоритми шифрування використовуються в системах потокової передачі інформації, тобто тоді, коли передача певного повідомлення розпочинається і завершується в будь-який момент часу та передачі цієї інформації

випадковим чином може перериватися, а блокові алгоритми шифрування використовуються для пакетної передачі даних та для кодування файлів [2].

Одним з поширених представників потокових алгоритмів шифрування є скремблери. При такому типі шифрування отримується псевдовипадкова послідовність біт. Скремблери широко використовуються в сучасних системах цифрового зв'язку, що мають назву SDH (Synchronous Digital Hierarchy), на них покладається завдання генерації псевдовипадкової послідовності біт для подальшого її використання у системах передачі даних. Принцип роботи такого методу шифрування полягає у наступному [3]:

- виконання логічної операції XOR (додавання за модулем 2) двох величин: біт вхідного сигналу та біт псевдовипадкової послідовності, що згодом передається на вихід алгоритму шифрування;
- далі попередня дія виконується необхідну кількість разів для отримання зашифрованої послідовності біт, тобто знову виконується операція XOR біт вхідного сигналу та наступний біт псевдовипадкової послідовності.

Оскільки скремблери відносяться до симетричного типу шифрування даних, то розшифрування інформації відбувається у зворотному порядку. Згенерована алгоритмом псевдовипадкова послідовність використовуються циклічно [3].

1.2.1 Блокові алгоритми шифрування

Симетричний блоковий алгоритм шифрування передбачає багатократне виконання перетворення блоку даних з використанням секретного ключа шифрування. За підходами до реалізації цього перетворення виділяють такі види блокових шифрів: побудовані на основі мережі Фейстеля, SP-мережі (чергування процедур перестановок і підстановок), структури "Square" (квадрат) та операцій за модулем. Всі ці типи мають майже однакові характеристики швидкості шифрування та стійкості до відомих атак [4].

Мережа Фейстеля була вперше представлена в 1973 році в статті "Cryptography and Computer Privacy" (Криптографія і комп'ютерна безпека) журналу

Scientific American Горстом Фейстелем. Базова схема мережі Фейстеля складається з 2 гілок, проте при великому розмірі блоків шифрування (більше 128 біт) реалізація такої схеми на 32-розрядних системах більш складна, тому використовують модифікований варіант, що складається з 4 гілок [5].

Принцип роботи мережі Фейстеля на 2 гілки [5]:

- вхідний блок тексту для шифрування поділяється навпіл, таким чином утворюючи дві рівні частини L та R;
- для кожного раунду обраховується:

$$L_i = R_{i-1} \oplus F(L_{i-1}, k_{i-1}); \qquad (1.1)$$

$$R_i = L_{i-1}, \qquad (1.2)$$

де L та R – під блоки інформації, $i = 1, \dots, r$ – номер раунду, F – це деяка функція, k_{i-1} – ключ i-го раунду шифрування.

Виконання r раундів призводить до отримання шифрованої інформації L_r та R_r. Схема раунду перетворення мережі Фейстеля з 2 гілками наведена на рис. 1.1.

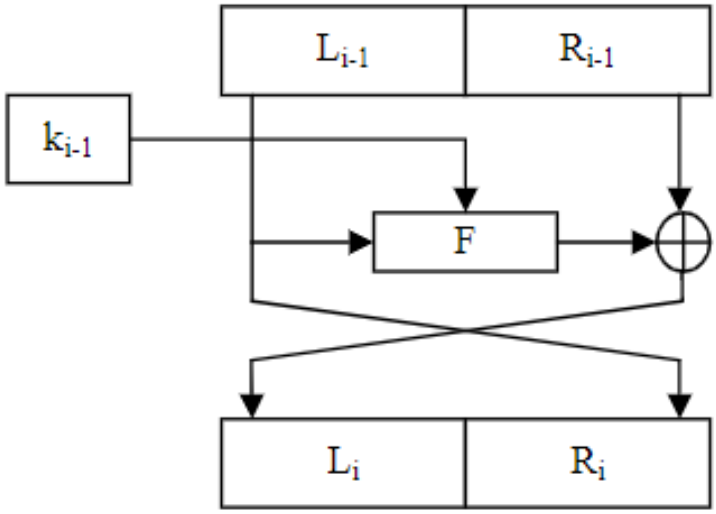

Рисунок 1.1 – Схема раунду перетворення мережі Фейстеля з 2 гілками

Мережа Фейстеля з 2 гілками покладена в основу більшості сучасних блокових алгоритмів шифрування. Основні перетворення даної мережі є оберненими, а процеси шифрування та розшифрування відмінні лише порядком використання раундового ключа [5]. Найбільш відомі та широко використовувані

блокові шифри: DES, Blowfish, Lucifer та інші. Важливим недоліком шифрів на основі мережі Фейстеля є відносно невисока швидкість шифрування та розшифрування, оскільки один раунд обробляє лише половину (мережа на 2 гілки) або частину (мережа на 4 гілки) блоку вхідного тексту, що потребує збільшення кількості ітерацій [4; 7].

SP-мережа (мережа замін-перестановок) приймає на вході блок відкритого тексту та ключ. Фрагмент вхідного блоку замінюється іншим фрагментом відповідно до таблиці замін S, а перестановки P залежать від ключа [4]. На рис. 1.2 зображена узагальнена схема раунду SP-мережі. Блокові шифри, що побудовані на основі SP-мереж, є стійкими до різних видів криптоаналізу, завдяки високому степеню нелінійності основних перетворень [8]. Алгоритми, що основані на SP-мережі: AES, Калина та інші.

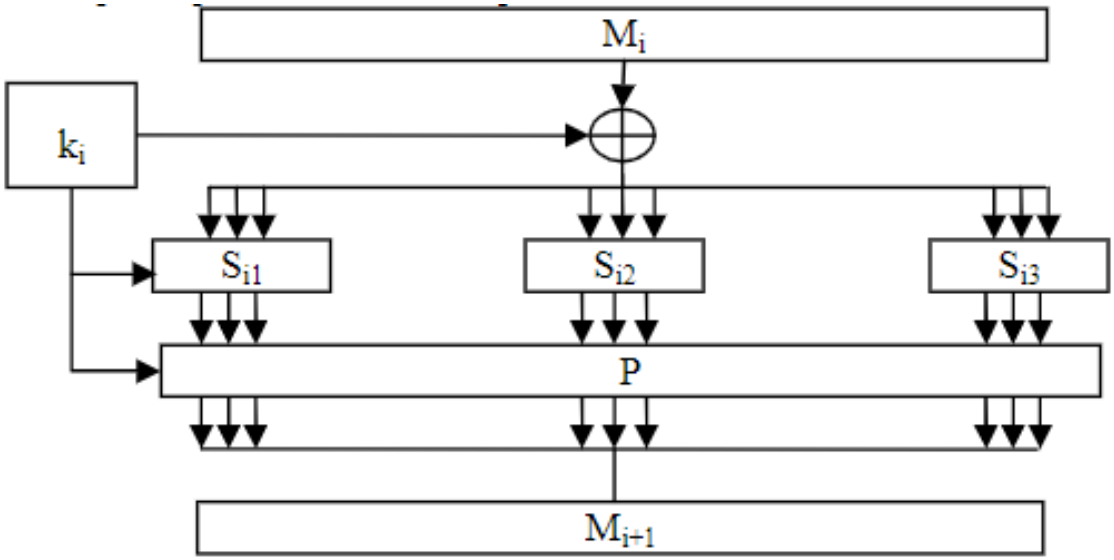

Рисунок 1.2 – Схема раунду перетворення на основі SP-мережі

Для блокових шифрів на основі структури “Square” (квадрат) характерною відмінністю є представлення блока у вигляді двовимірного байтового масиву. Криптографічні перетворення можуть виконуватись як над окремими байтами масиву, так і над його стовпчиками та рядками [6]. Перевагою шифрів на основі цієї структури є висока стійкість, проте використання табличних замін впливають на швидкість процесу шифрування. Найбільш відомим представником цієї групи алгоритмів є алгоритм AES.

Блокові шифри на основі арифметичних операцій за модулем використовують операції множення та додавання за модулем (XOR). Представниками цієї групи шифрів є: ABC, Caligo, MultiSwap, xmx та інші. Використання одного і того самого ключа для двох різних груп операцій на всіх раундах шифрування (xmx, MultiSwap) і відкритого значення модуля не забезпечує достатній рівень криптографічної стійкості шифру, проти атак, з використанням мультиплікативних диференціалів [9].

Сучасні блокові шифри повинні задовольняти такі вимоги: стійкість алгоритму шифрування повинна визначатися лише секретним ключем[10], можливість використання різного розміру секретного ключа [4], використання довжини блоку не менше 128 біт (2×64 біт) [4], хоча блок розміром 64 біт є прийнятним, проте для деяких алгоритмів з такою довжиною блоку є наявні теоретичні або навіть і практично реалізовані атаки.

Блокові симетричні шифри застосовуються для забезпечення конфіденційності та цілісності при обробці інформації в інформаційно-телекомунікаційних системах. Також використовуються як базовий елемент для побудови інших криптографічних алгоритмів, наприклад, генератори псевдовипадкових послідовностей [11].

До недоліків блокового шифрування можна віднести: всі блоки зашифровуються з використанням одного і того ж ключа шифрування; достатньо велика складність перетворень при шифруванні; складність, а іноді і неможливість розпараллелювання процесів криптографічних перетворень [11].

1.2.1.1 Алгоритми шифрування DES та Triple-DES

Data Encryption Standard (DES) – симетричний блоковий алгоритм шифрування опублікований Національним Інститутом Стандартів та Технологій (NIST) в 1973 р. Базується на мережі Фейстеля, оскільки алгоритм використовує 16 раундів мережі Фейстеля з різними ключами для кожного раунду. Вперше у листопаді 1976 року DES став затвердженим стандартом шифрування у США [12].

Для шифрування інформації алгоритм DES використовує відкритий текст розміром 64 біт та ключ довжиною 64 біт, проте 8 біт ключу використовується лише в якості контрольних бітів [13]. Кожен восьмий біт призначений для контролю парності інших бітів ключа [14]. Процес розшифрування інформації є оберненим процесу шифрування, тобто операції виконуються у зворотній послідовності.

Послідовність виконання алгоритму складається з таких етапів [12]:

- процес шифрування розпочинається з того, що блок вхідного тексту розміром 64 біт передається функції початкової перестановки (позначається як IP);
- виконується початкова перестановка (IP) вхідного тексту, що виконується згідно таблиці початкової перестановки;
- після початкової перестановки блок ділиться на дві частини, ліву (L_0) та праву (R_0);
- ліва та права частини (L_0 та R_0) проходять 16 раундів процесу шифрування інформації;
- ліва та права частини (L_0 та R_0) об'єднуються та над утвореним блоком інформації відбувається кінцева перестановка (IP^{-1}), що також як і початкова перестановка виконується згідно спеціальній таблиці, ці таблиці є інверсні одна одній;
- результатом процесу шифрування є шифрований блок інформації розміром 64 біт.

Необхідність використання 16 раундів шифрування пояснюється такими умовами [15]:

- 12 раундів є мінімальною кількістю для забезпечення належного рівня криптографічного захисту;
- використання 16 раундів в апаратній реалізації дозволяє повернути перетворений ключ у початковий стан для подальшого використання;
- перешкодити реалізацію атаки на блок шифрованих даних з двох сторін.

Процес шифрування інформації в алгоритмі DES можна поділити на такі кроки: генерація раундових ключів; перестановка й розширення; S-блок заміни; P-

блок перестановки; порозрядне підсумування в раунді (XOR) та заміна секцій раунду.

Необхідна кількість раундових ключів становить 16, оскільки кожний раунд використовує різний згенерований 48-бітовий ключ. На початку генерації раундових ключів необхідно видалити біти перевірки ключової послідовності. Ця процедура передбачає видалення бітів парності (08, 16, 24, 32, 40, 48, 56, 64 біти), вони не впливають на шифрування. Після видалення бітів парності, ключ поділяється на дві частини розмір кожної 28 біт. Над цими частинами відбувається зсув вліво на один або два біти, це залежить від номеру раунду та визначається спеціальною таблицею. Після циклічного зсуву дві частини зміненої ключової послідовності об'єднуються і утворюється блок розміром 56 біт. Далі над отриманим блоком відбувається перестановка та стиснення послідовності, що також виконується згідно даним таблиці, і результатом цих операцій є раундовий ключ k_i [15]. На рис. 1.3 зображена схема генерації раундових ключів.

Блок перестановки і розширення необхідний для того, щоб збільшити частину блоку R_1 довжиною 32 біт, до 48 біт, оскільки ключ k_i має довжину 48 біт, блок розділяється на 8 секцій по 4 біт, що потім розширюються до 6 біт. Для цього перетворення використовуються певні правила: вхідні біти 1, 2, 3 і 4 привласнюються бітам 2, 3, 4 і 5 на виході; 1 біт вихідної послідовності визначається за допомогою 4 біта попередньої секції; вихідний біт 6 визначається з біта 1 наступної секції [15].

S-блок використовується для змішування інформації, а P-блок для прямої перестановки з 32 біт на вході та на виході. Результат перетворення P-блока підсумовується за модулем 2 з L_{i-1} та отримується послідовність R_{i-1}^{L}. Наступним є процес заміни секцій раунду, що призначений для завершення процесу одного раунду алгоритму, результатом цієї операції є значення послідовностей L_i та R_i для наступного раунду [15].

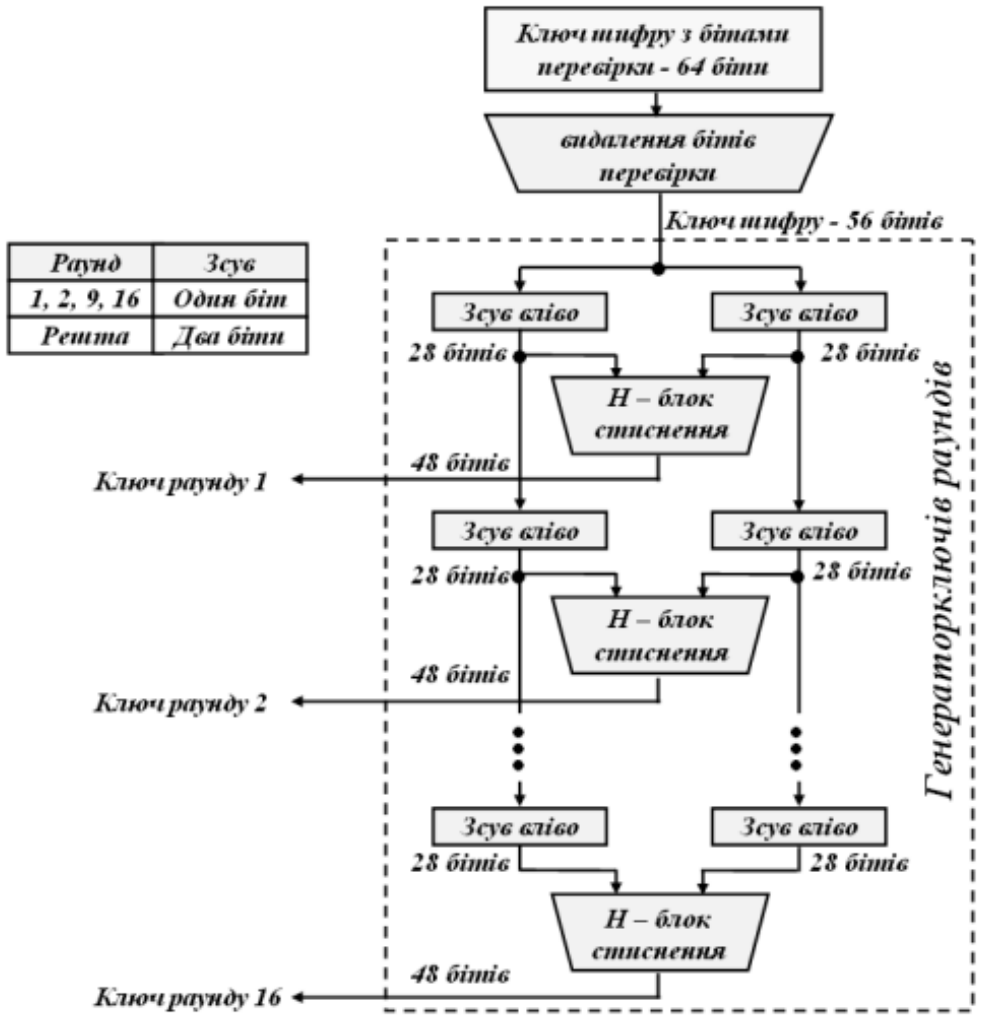

Рисунок 1.3 – Генерація раундових ключів

Використання DES має різні режими шифрування серед яких: електронна кодова книга (ECB), зчеплення блоків зашифрованих даних (CBC), зворотного зв’язку за зашифрованими даними (CFB), зворотного зв’язку за виходом (OFB), лічильника (CTR). Якщо кожен 64 бітовий блок шифрується окремо, то такий режим шифрування називається режимом електронної кодової книги (ECB) [16]. У режимі зчеплення блоків зашифрованих даних кожен блок залежить від попереднього [12]. У режимі зв’язку за зашифрованими даними (CFB) попередньо зашифрований текст використовується як вхідні дані і потім операцією XOR разом з відкритим текстом утворює наступний зашифрований текст [17]. Режим зворотного зв’язку за виходом (OFB) є подібний до режиму зв’язку за зашифрованими даними (CFB), окрім того, що вхідна послідовність алгоритму шифрування є виходом попередньої

послідовності [12]. У режимі лічильника (CTR) кожен блок відкритої послідовності отримується за допомогою операції XOR та зашифрованого лічильника [17].

Алгоритм DES вважається ненадійним алгоритмом шифрування через відносно малу довжину ключа 56 біт. В 1998 році під керівництвом Джона Гілмора разом з Electronic Frontier Foundation (EFF) побудували машину "Deep Crack", що може пройти через весь 56-розрядний ключовий простір за 4,5 дня, а 17 липня 1998 року вони оголосили, що зламали ключ за 56 годин. Комп’ютер використовував 27 плат, кожна з яких містить 64 мікросхеми і здатний тестувати 90 мільярдів ключів в секунду [16].

Загальна схема шифрування за допомогою блокового алгоритму шифрування DES показано на рис. 1.4.

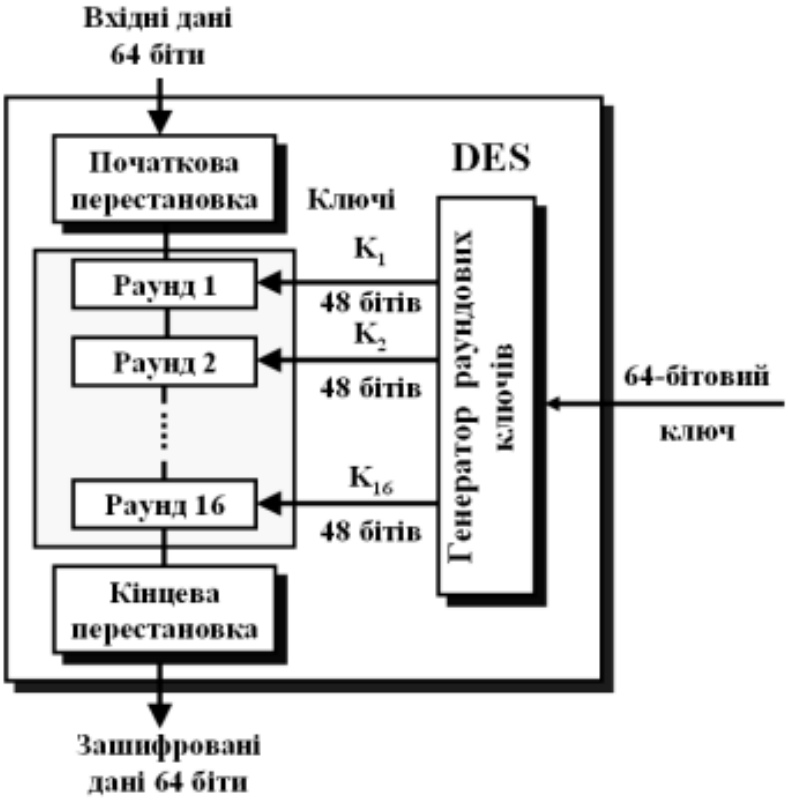

Рисунок 1.4 – Загальна схема шифрування DES

До переваг алгоритму DES можна віднести те, що цей алгоритм є стандартним, тобто на базі цього алгоритму можна не лише вивчати криптографічні методи, а й можна ускладнювати та створювати нові методи шифрування даних. Також перевагою можна зазначити те, що алгоритм можна виконувати у різних

режимах шифрування. Проте його головним недоліком є розмір ключа шифрування, що є занадто малим для задоволення потреб сучасного шифрування даних.

Triple-DES – симетричний блоковий шифр, що застосовує шифр DES тричі до кожного блоку даних. Для шифрування зазвичай використовується три ключі: k_1, k_2 та k_3. Перший ключ використовується для шифрування повідомлення, другий для розшифрування шифрованого повідомлення, проте другий ключ не є правильним ключем і тому просто перемішує дані, третій ключ використовується для отримання остаточно зашифрованого повідомлення. Також існує двоключовий варіант, в якому перший та третій ключ однакові [12; 16; 17]. Недоліком такого шифрування є те, що він в три рази повільніший за алгоритм DES, проте є більш безпечним. Основною перевагою цього алгоритму є те, що загальна довжина ключа складає 168 біт [18].

1.2.1.2 Алгоритм шифрування AES

Advanced Encryption Standard (AES) – симетричний блоковий шифр, також відомий як Rijndael, прийнятий як стандарт шифрування США. Оскільки шифр DES втрачав свою актуальність через публічне розшифрування, повільну швидкість програмної реалізації та недостатній розмір ключа шифрування, 1997 року NIST оголосили конкурс для вибрання нового стандарту шифрування, головні вимоги: блоковий алгоритм шифрування, довжина блоку якого становить 128 біт, ключі шифрування повинні задовольняти довжину 128, 192 та 256 біт. Переможцем цього конкурсу став алгоритм розроблений бельгійськими криптографами Йоном Деменом та Вінсентом Рейменом, початкові букви прізвищ утворюють назву шифру Rijndael (Rijmen, Daemen), AES був прийнятий як стандарт шифрування 26 травня 2002 року [19; 15].

Алгоритм AES базується на принципах SP-мережі та має структуру “Square” (квадрат). Кількість раундів в шифруванні AES залежить від довжини ключа: 128-бітовий ключ використовує 10 раундів, 192-бітовий ключ – 12 раундів, 256-бітовий ключ – 14 раундів. Кожен з цих раундів використовує інший 128-бітовий ключ раунду, який обчислюється на основі вихідного ключа AES [20]. Кожен ключ

шифрування має різну кількість можливих ключових комбінацій: 128-бітовий – $3{,}4x10^{38}$; 192-бітовий – $6{,}2x10^{57}$; 256-бітовий – $1{,}1x10^{77}$. Незважаючи на те, що довжина ключа шифрування може бути різною, блок шифрування залишається незмінним – 128 біт [21].

Процес алгоритму шифрування DES можна поділити на такі етапи [23]:

1) ключове розширення, використовується для обрахунку раундових ключів з початкового ключа шифрування;

2) додавання раундовго ключа, використовується для додавання (XOR) початкового раундового ключа до блоку даних;

3) SubBytes (Byte substitution – заміна байтів), використовується для заміни кожного байта іншим байтом на основі S-box блоку заміни для шифру AES;

4) ShiftRows (Row shifting – переміщення рядків), кожен із чотирьох рядків матриці зміщується вліво, правила зсуву: перший рядок не зсувається; другий рядок зсувається на один байт (одну позицію) вліво; третій рядок зсувається на дві позиції вліво; четвертий рядок зсувається на три позиції вліво; результатом є нова матриця, що складається з тих же 128 бітів (16 байтів), але зміщених відносно один одного [20];

5) MixColumns (Column mixing – перемішування стовпців), по суті є множенням матриці, кожен стовпець множиться на певну визначену матрицю, таким чином позиція кожного байта в стовпці змінюється, цей етап пропущений в останньому раунді [22];

6) додавання раундового ключа – до 128 біт даних додається (XOR) раундів ключ, якщо ця дія виконується в останньому раунді, то результатом виконання є зашифрований текст.

Етапи 3-6 виконуються в кожному раунді шифрування. Процес розшифрування відбувається в зворотному порядку.

Окрім безпеки алгоритму шифрування AES, що пов’язано з більшою довжиною ключа, є ще і інші переваги цього алгоритму. Дуже важливим є те, що AES відносно простий для розуміння, тому його реалізація є значно простіша та забезпечує високу швидкість шифрування та розшифрування. Також для

забезпечення вищого рівня криптографічної безпеки, алгоритм шифрування AES можна комбінувати з іншими протоколами безпеки та типами шифрування (WPA2, SSL) [21].

Актуальність шифрування AES можна пояснити тим, що наразі дослідники виявили лише теоретичні атаки на цей алгоритм шифрування. У 2009 році було виявлено атаку пов'язаних ключів (related-key attacks), цей тип криптоаналізу передбачає спостереження за тим як працює шифр з різними ключами, проте такі атаки можливі лише проти протоколів, що реалізовані неналежним чином. У цьому ж році відбулася атака з відомим ключем проти восьми раундової версії AES-128, проте це не несе ніяку загрозу для тих хто використовує стандартний AES-128, оскільки він складається з більшої кількості раундів. Через відсутність реалізованих атак на AES його можна вважати, на даний момент, не зламним, проте цей алгоритм може бути вразливий бічній атаці (side-channel attack). Ця атаки відбувається при витоку інформації, що в подальшому може застосовуватися для реалізації атаки [24].

Деякі приклади використання алгоритму AES: VPN (VPNs, що використовують AES-256 є NordVPN, Surfshark, ExpressVPN), Wi-Fi (може використовуватися шифрування AES разом з WPA2), мобільні додатки (Snapchat and Facebook Messenger), інструменти архівування та стиснення (використовують AES для запобігання витоку даних,7z, WinZip і RAR), парольні менеджери (LastPass) [21].

1.2.2 Потокові алгоритми шифрування

Потоковий алгоритм шифрування – це симетричний алгоритм шифрування, що шифрує один символ відкритого тексту за раз і незалежно один від одного, на відмінну від блокового алгоритму шифрування, де за один раз шифрується блок відкритого тексту фіксованого розміру. Шифр використовує нескінченний потік псевдовипадкових бітів як ключ, для реалізації безпечного потокового шифру, генератор псевдовипадкових бітів повинен бути непередбачуваним і ключі ніколи не повинні використовуватися повторно. Такий тип шифрування розроблений для

наближення до ідеалізованого шифру, що також відомий як одноразовий блокнот [25].

Для шифрування інформації за допомогою потокового шифру необхідно провести генерацію ключа, потім кожен біт відкритого тексту за допомогою математичної формули, що базується на операції XOR, та ключової послідовності перетворюється у шифрований текст, який отримувач не зможе зрозуміти без ключа. Лише з правильним ключем можлива операція розшифрування, що виконується у зворотному порядку [26].

Потоковий шифр може генерувати ключову послідовність двома способами і в залежності від цього поділяються на два типи: синхронні потокові шифри та потокові шифри, що самосинхронізуються (також називають асинхронні потокові шифри) [27].

У синхронних потокових шифрах блок ключового потоку генерується незалежно від попереднього зашифрованого та відкритого тексту повідомлення. Поширені режими потокового шифрування використовують генератори псевдовипадкових чисел для створення послідовності бітів та об’єднати з ключем, щоб сформувати потік ключів, що згодом за допомогою операції XOR буде об’єднаний з відкритим текстом для отримання шифрованого повідомлення [28].

Потоковий шифр, що самосинхронізується генерує блок ключового простору як функцію симетричного ключа та попереднього блоку шифрованого тексту фіксованого розміру [28]. Перевагою такого типу є те, що отримувач повідомлення на автоматичній основі синхронізується з генератором ключового потоку по отриманні фіксованого розміру шифрованого тексту, оскільки це зменшує складність відновлення, у тому випадку якщо до потоку інформації доданий або видалений символ [27].

Найпоширеніше використання потокових алгоритмів шифрування є шифрування інформації в каналах зв’язку. Перевагою такого алгоритму шифрування є його швидкість, вона зазвичай більша, ніж у інших шифрах. Також таким алгоритмам притаманна низька складність, що спрощує реалізацію шифру та вимоги до обладнання. Найкращий спосіб, щоб захистити інформацію шифровану

потоковим алгоритмом – це використання ключу лише один раз і ключі не мають бути пов’язані між собою [26].

Відомі потокові алгоритми шифрування: A3, A5, A8, RC4, SEAL та інші.

1.2.2.1 Алгоритм шифрування RC4

RC4 (Rivest Cipher 4) – потоковий алгоритм шифрування, що розроблений в 1987 році Роном Рівестом для компанії з безпеки мережі, також називають “Ron’s Code” (Код Рона). Цей шифр використовує ключ розміром 64 біт або 128 біт [29]. Ключовий вхід – це генератор псевдовипадкових бітів, що створює потік розміром 8 біт, що не можна передбачити без знання ключа введення; вихід генератора називається потоком ключів, він об’єднується по одному байту відкритого тексту та потоку ключів за допомогою операції XOR [30].

RC4 базується на понятті матриці станів, розмір цієї матриці 256 байтів. З цієї матриці випадково вибирається один байт, що і буде ключем шифрування. Алгоритм включає в себе два етапи: підготовчий та основний. На підготовчому етапі відбувається ініціалізація матриці стану *S* (таблиця замін), матриця ключа *K* та перестановка матриці стану, що залежить від значення байтів у *K[i]*. На основному етапі обчислюється псевдовипадкові числа *k* [15].

Ключ RC4 становить довільну послідовність байтів за яким створюється початковий стан шифру *S*, тобто перестановка всіх 256 байтів. Процес ініціалізації також називають алгоритмом ключового розкладу (KSA, Key-Scheduling Algorithm). Якщо довжина ключа шифрування рівна 256 байт, то значення копіюються в масив ключа *K*, в іншому випадку байти повторюються, поки не заповнять масив. У матриці станів відбувається перестановка елементів, що залежить від елементів матриці ключа. Далі генеруються псевдовипадкові числа, цей етап також називають алгоритм псевдовипадкової генерації (PRGA, Pseudo Random Generation Algorithm). Після цих кроків відбувається безпосередньо процес шифрування відкритого тексту, що зашифровується за допомогою ключового потоку *k* та операції XOR.

Розшифрування полягає в регенерації ключового потоку та додавання до нього зашифрованих даних за модулем 2 [15; 30].

Коротко описати алгоритм шифрування даних за допомогою RC4 можна так:

- користувач вводить дані та секретний ключ;
- генерація ключового потоку використовуючи алгоритми KSA та PRGA;
- виконання логічної операції XOR потоку ключа та відкритим текстом для створення шифрованого повідомлення;
- зашифрований текст надсилається отримувачу, що в подальшому розшифрує його та зможе прочитати вихідне відкрите повідомлення.

Алгоритм шифрування RC4 використовувався в SSL/TLS (Secure Socket Layer/Transport Layer Security) протоколах, стандарті бездротової локальної мережі IEEE 802.11 та протоколі безпеки Wi-Fi WEP (Wireless Equivalent Protocol). У 2015 році Internet Engineering Task Force (IETF) заборонила використання RC4 в протоколах TLS. Microsoft і Mozilla також опублікували рекомендації щодо обмеження застосування RC4 через вразливості. Застосування RC4 поступово припиняється з різних криптосистем через відомі атаки серед яких: Флюрер, Мантін і Шамір атака (Fluhrer, Mantin and Shamir attack, дозволяє відновити ключ з великої кількості повідомлень у цьому потоці), атака Кляйна, NOMORE атака та інші [31; 29; 33; 34].

До переваг RC4 можна віднести простоту використання та реалізації, швидкість виконання операцій, можна застосовувати для великих потоків даних. Недоліками є такі моменти: не надає аутентифікацію, вразливий до атаки з перевертанням бітів (bit-flipping attack) без використання сильного MAC, не реалізовується на невеликих потоках даних [32].

1.2.2.2 Загальні відомості про алгоритм шифрування СТРУМОК

У національному стандарті України представлений синхронний потоковий алгоритм шифрування під назвою СТРУМОК. Цей стандарт шифрування має назву ДСТУ 8845:2019 «Інформаційні технології. Криптографічний захист інформації.

Алгоритм симетричного потокового перетворення». Цей стандарт застосовується під час створення криптографічного захисту інформації в різних системах, а саме інформаційних, телекомунікаційних та інформаційно-телекомунікаційних [35].

СТРУМОК побудований за SNOW-2.0-подібною схемою, використовує 256-бітний вектор ініціалізації та 256-бітний чи 512-бітний секретний ключ, забезпечує високий та надвисокий рівні стійкості з урахуванням можливого застосування квантового криптографічного аналізу. Залежно від довжини ключа поділяється на два режими: режим із 256-бітним ключем (СТРУМОК-256) та режим із 512-бітним ключем (СТРУМОК-512) [35].

В 2016 році Олександр Кузнецов, Марія Луценко та Дмитро Іваненко представили шифр STRUMOK у роботі «Strumok Stream Cipher: Spesification and Basic Properties». Дана робота представлена на сайті IEEE Xplore та складається з 5 частин.

Структурні елементи генератора ключових потоків СТРУМОК є регістр зсуву з лінійним зворотним зв’язком та скінченний автомат, де виконується нелінійне перетворення [35].

1.3 Криптографічна стійкість та відомі атаки

Криптографічна стійкість – термін, що описує здатність криптографічної системи протистояти криптоаналізу; стійким вважається алгоритм, що для реалізації успішної атаки потребує величезних обчислювальних ресурсів злочинця та великої кількості перехоплених зашифрованих повідомлень для спроби створити систему дешифрування, яка виявиться неактуальною, оскільки для цього необхідно багато часу, а система захисту постійно удосконалюється [36].

Криптографічна стійкість залежить від багатьох факторів серед яких: секретність ключа шифрування; складність пошуку ключа шифрування; складність відтворити алгоритм дешифрування; існування уразливостей алгоритму та можливість їх використання; можливість реалізації атаки з відкритим текстом, суть

якої полягає в можливості розшифрувати все повідомлення, якщо вдалося розшифрувати фрагмент даних; знання властивостей відкритого тексту [37].

Симетричні алгоритми шифрування сприятливі до реалізації атак такого типу: атака на відомий відкритий текст (known-plaintext attacks), атака з вибраним відкритим текстом (chosen-plaintext attacks), диференціальний та лінійний криптоаналізи, атака грубої сили [38].

Атака грубої сили – це спроба зламати зашифроване повідомлення, перебираючи всі можливі ключі,поки не знайдеться вірний і дані можна буде розшифрувати. Для реалізації такого типу атаки необхідний лише відносно малий фрагмент зашифрованого тексту на якому будуть підбиратися ключі. Основний захист від такої атаки є використання ключа шифрування з великою довжиною, що збільшить можливий ключовий простір для перебору ключа. Сучасні алгоритми використовують ключ довжиною 128 біт і більше, що майже унеможливлює атаку такого типу [39]. Атаки грубої сили були реалізовані на шифр DES та AES, для цього було сконструйоване спеціальне апаратне забезпечення, у першому випадку – це DeepCrack (1998 р.), COPACABANA (2006 р.), а у другому – RIVYERA S3-5000 (2013 р.) [40].

Атака на відомий відкритий текст (known-plaintext attacks) – це атака у який криптоаналітик використовує блок відкритого тексту та відповідний блок зашифрованого тексту, метою є визначення ключа шифрування [41]. Ця атака ефективна для шифрів підстановки, у симетричних алгоритмах вона може бути корисною лише для певних блоків алгоритму, такі як блоки підстановки та перестановки.

Атака з вибраним відкритим текстом (chosen-plaintext attacks) – тип атаки, при якій зловмисник самостійно вибирає фрагмент відкритого та шифрованого тексту, аналізує їх порівнюючи дані та знаходить ключ шифрування [15].

Диференціальний криптоаналіз – це тип атаки проти блоковий шифрів в першу чергу, проте може застосовуватися також для потокових алгоритмів та хеш-функцій, метою цієї атаки є знаходження ключа шифрування. Ця атака є типом атаки з вибраним відкритим текстом. Застосовується для алгоритму DES [42].

Лінійний криптоаналіз використовує атаку на відомий відкритий текст для декількох шифрованих повідомлень з однаковим ключем шифрування, що створює уявлення про ймовірність певного ключа, чим більше проаналізованих повідомлень тим більша можливість знайти ключ шифрування [42].

1.4 Переваги та недоліки симетричного шифрування

Перевагою симетричних алгоритмів шифрування є простота. Такий тип шифрування легкий та зрозумілий для виконання, його можна відносно легко реалізувати апаратно та програмно, маючи необхідні знання та ресурсні можливості. Такий алгоритм шифрування є швидким, в порівнянні з асиметричним алгоритмом шифрування. Ще однією перевагою є використання менше комп'ютерних ресурсів порівняно з шифруванням з відкритим ключем. Також перевагою такого шифрування є безпека, оскільки сучасні блокові шифри використовують ключ з великою довжиною, що робить неможливим його підібрати, на підбір такого ключа можуть піти роки, що не є актуальним та корисним. Для великої частини представників симетричного шифрування не існує практичних атак, проте для деяких наявні теоретичні атаки. Такий тип шифрування є захищеним від квантового аналізу. Симетричні алгоритми шифрування можуть бути використані для шифрування інформації з великим об'ємом даних [44, 45].

Недоліком симетричного шифрування є проблема спільного доступу до ключа або його передача через канал зв'язку. Оскільки ключ шифрування однаковий і для процесу шифрування, і для процесу розшифрування, що є ще одним недоліком симетричного шифрування, то постає питання передачі ключа отримувачу. Тому для уникнення перехоплення ключа третьою стороною необхідно використовувати захищені канали передачі даних. Ще одним недоліком є те, що оригінальність та автентичність повідомлення не можливо гарантувати. Більша ймовірність витоку ключа шифрування ніж в асиметричному шифрування, оскільки для шифрування завжди необхідний секретний ключ, а для асиметричного шифрування потрібен лише відкритий ключ. Якщо безпека буде скомпрометована, то є ризик витоку

даних, оскільки знаючи ключ шифрування можна розшифрувати всю інформацію, що була зашифрована цим ключем [44; 45; 46].

Підхід до вибору алгоритму шифрування повинен бути комплексним та розглядатися враховуючи не лише переваги та недоліки певного алгоритму чи типу алгоритмів, а й враховувати особливості використання алгоритмів, особливості інформації, що підлягає шифрування, наявні ресурси та інше. Тому найпершим кроком для вибору алгоритму повинен бути розгляд властивостей інформації, розуміння власних потужностей обладнання, розуміння яким чином буде передаватися інформація та необхідний рівень її захисту. Після такого аналізу можна переходити до розгляду наявних алгоритмів шифрування, підбору виду шифрування, розгляд переваг та недоліків певного алгоритму та його сферу використання. Лише комплексний підхід до вибору алгоритму шифрування надасть бажаний результат та допоможе уникнути непередбачуваних обставин в ході виконання процесу.

Висновки за розділом 1

В першому розділі даної дипломної роботи було розглянуто характеристику та принципи побудови та функціонування симетричних алгоритмів шифрування. Головними результатами виконання завдань першого розділу є:

- проведено аналіз наявних криптографічних алгоритмів та подано головні характеристики та відмінності між ними, а також сфери застосування та найвідоміші представники;
- подана класифікація симетричних алгоритмів шифрування, що складається з блокових та потокових алгоритмів шифрування головною відмінністю між ними є розмір блоку інформації для шифрування;
- проведено аналіз блокових алгоритмів шифрування та їх поділ за підходами до реалізації шифрування, згідно цього принципу вони поділяються на: побудовані на основі мереж Фейстеля, SP-мереж (чергування процедур перестановок і підстановок), структури “Square” (квадрат) та операцій за модулем;

- детально розглянуто алгоритм шифрування DES та його модифікацію Triple-DES;
- розглянуто алгоритм шифрування AES;
- проведено аналіз потокових алгоритмів шифрування та представлено два типи шифрування, такий поділ обумовлений тим, що генерувати ключову послідовність можна двома способами і поділяються на синхронні потокові шифри та потокові шифри, що самосинхронізуються (також називають асинхронні потокові шифри);
- розглянуто алгоритм шифрування RC4, подано основні його властивості та алгоритм шифрування;
- розглянуто алгоритм шифрування СТРУМОК;
- подано ключові моменти від яких залежить криптографічна стійкість алгоритму шифрування та розглянуті можливі типа атак на алгоритм шифрування;
- приведені переваги та недоліки симетричних алгоритмів шифрування.

РОЗДІЛ 2
ПОТОКОВИЙ АЛГОРИТМ ШИФРУВАННЯ A5

2.1 Сімейство алгоритмів A5. Історія створення та розвитку

Алгоритм A5 – це потоковий симетричний алгоритм шифрування, що використовується у системі GSM (Group Special Mobile, система мобільного цифрового зв'язку), розроблений у 1989 році. Забезпечує безпеку та конфіденційність даних абонентів, що для обміну інформацією використовують телефон, який надалі використовує для своєї роботи базову станцію. Алгоритм побудований з використанням розробки французьких криптографів, що побудували схему потокового шифру лише для використання у військових цілях. Таким чином, для забезпечення безпеки даних в GSM використовують три алгоритми: A3, A5, A8. Перший з них відповідає за процес аутентифікації абонентів, другий – за процес потокового шифрування розмови, третій – генерує сеансові ключі [15].

Алгоритм A5 використовувався лише в країнах Європи, але згодом A5 перейменували в A5/1 та використовували в країнах Європи та США, а для інших країн алгоритм модифікували знизивши криптографічну стійкість та назвали A5/2. Також розроблено алгоритм A5/3, що базується на алгоритмі Касумі та використовується в мережах 3G. Також наявна модифікація A5/0, в ній відсутній процес шифрування [15]. У зв'язку з тим, що алгоритм має таку кількість модифікацій, проте всі ці алгоритми мають спільну ідею використання з деякими відмінностями, їх називають сімейством алгоритмів A5.

В алгоритмі A5/0 розмова абонентів не шифрується, тому передається відкритим текстом. Така розмова є практично незахищеною та легко може прослуховуватися сторонньою стороною [47].

Алгоритм A5/1 – це потоковий алгоритм шифрування даних, що складається з трьох регістрів зсуву з лінійним зворотним зв'язком x, довжина яких становить 19, 22, 23 біт [48]. Також в алгоритмі використовується логічна операція XOR,

об’єднання цієї операції та регістрів зсуву дозволяють спростити апаратну реалізацію та збільшити швидкість процесу шифрування.

Алгоритм A5/2 в даний час не використовується, причиною цього є те, що такий шифр розсекречується за 15 мс роботи сучасних обчислювальних машин. Розкриття шифру пов’язане з тим, що він був модифікований за рахунок зменшення стійкості алгоритму. В такому алгоритмі додано ще один регістр зсуву довжиною 17 біт, що управляє бітами в решті регістрів зсуву [49].

Алгоритм A5/3 – це блоковий алгоритм шифрування розроблений в 2002 році. Цей алгоритм має ряд вразливостей, проте успішних атак не було реалізовано. В основу даного алгоритму покладений алгоритм Касумі, що базується на алгоритмі MISTY (Mitsubishi) [53].

2.2 Структура та принцип роботи

Структуру та принцип роботи розглянемо на прикладі алгоритму A5/1. Оскільки алгоритм A5/1 є потоковим алгоритмом шифрування, то відкритий текст не ділиться на блоки сталого розміру. Алгоритм A5/1 передбачає створення потоку бітів використовуючи ключ довжиною 64 біт. Оскільки в GSM інформація передається кадрами величиною 228 біт, тому потоки шифрування зібрані в буфери по 228 бітів, щоб виконувати логічну операцію XOR [15]. Цей принцип показано на рис. 2.1.

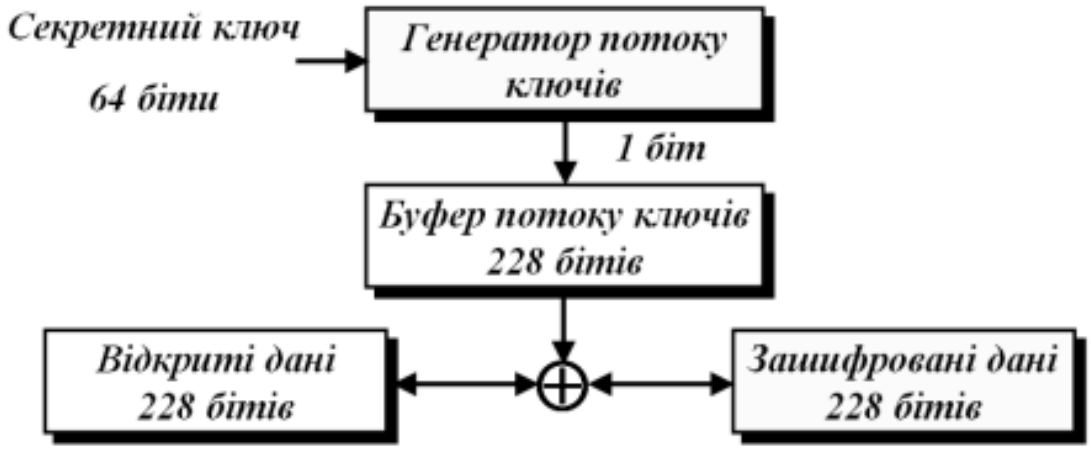

Рисунок 2.1 – Принцип побудови алгоритму A5/1

Цей рисунок показує, що для шифрування інформації необхідно виконати операцію XOR відкритих даних та буферу потоку ключів, а для розшифрування інформації навпаки, тобто до зашифрованих даних додати за допомогою операції XOR (за модулем 2) буфер потоку ключів і отримаємо початковий відкритий текст.

Визначення вихідної послідовності, шифрованого тексту, відбувається шляхом виконання операції XOR (додавання побітово за модулем 2) потоку відкритого тексту та послідовності біт, що також називається гамою. Від властивостей послідовності (гами) залежить захищеність системи, тому при ідеальних умовах кожен біт послідовності повинен бути незалежним значенням, а вся гама повністю випадковою, проте це ускладнює реалізацію. У практичних системах використовується ключ заданого розміру та послідовність генерується за допомогою цього ключа і є псевдовипадковою. Алгоритм A5 відноситься до потокових алгоритмів в яких генератор псевдовипадкової послідовності біт побудований на регістрах зсуву з лінійним зворотним зв'язком [15].

Регістр зсуву з лінійно зворотним зв'язком складається з послідовності біт заданої довжини (власне регістр) та зворотного зв'язку. Під час виконання кожного такту відбувається такий процес: крайній лівий біт, що є старшим бітом, витягується, таким чином гама зсувається вліво і додається молодший біт вправо, що є значенням функції зворотного зв'язку. Цією функцією є многочлен, що отримується в результаті виконання операції XOR визначених бітів регістру, ступінь многочлену вказує на номер біта. Витягнуті біти формують вихідну послідовність [15].

Регістр зсуву з лінійно зворотним зв'язком не є достатньо надійним для використання, оскільки піддається криптоаналізу. Для такого регістру основним показником є період псевдовипадкової послідовності [15].

На рис. 2.2 показана схема алгоритму A5/1, що складається з [48]:

- трьох регістрів, які позначені як R_1, R_2, R_3, довжина яких відповідно 19, 22, 23, многочлени представлені такими формулами 2.1 – 2.3:
  - для R_1: $x^{19} + x^{18} + x^{17} + x^{14} + 1$ (2.1)
  - для R_2: $x^{22} + x^{21} + 1$ (2.2)

- для R_3: $x^{23} + x^{22} + x^8 + 1$ (2.3)

- схему управління тактуванням.

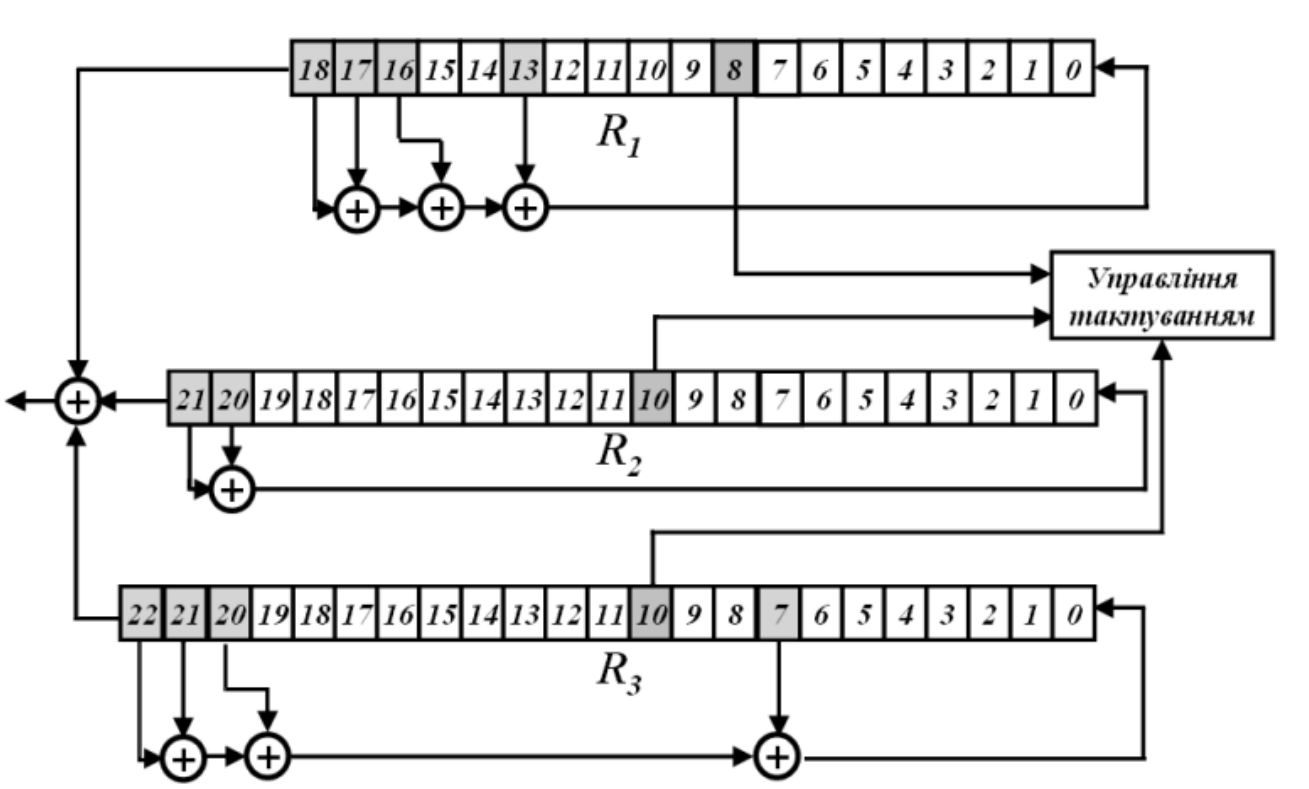

Рисунок 2.2 – Структура алгоритму A5/1

Управління тактуванням формується за допомогою бітів синхронізації, що є спеціальними для кожного регістру: для R_1 – 8, для R_2 – 10 та для R_3 також 10. Ці біти використовуються для обрахунку функції (2.4) [15]:

$$F = x \,\&\, y \mid x \,\&\, z \mid y \,\&\, z \qquad (2.4),$$

де & – булево *and;* | – булево *or; x, y, z* – біти синхронізації R_1, R_2, R_3 відповідно.

Якщо біт синхронізації дорівнює функції F у такому випадку регістр зсувається, вихідний біт системи є результатом додавання за модулем 2 вихідних бітів регістру [15].

2.3 Відмінності алгоритмів сімейства A5

В алгоритмі шифрування A5/2 додано ще один регістр розміром 17 біт (R_4), що призначений для керування іншими регістрами. Можна зазначити такі модифікації алгоритму A5/2 [50]:

- додано регістр R_4 розміром 17 біт;
- многочлен зворотного зв'язку R_4: $x^{17} + x^{12} + 1$; (2.5)
- R_4 здійснює управління тактуванням;
- для R_4 бітами синхронізації є біти 3, 7, 10;
- обчислюється мажоритарна функція *F*, що обраховується за формулою 2.4;
- R_1 зсувається якщо R_4 (10) = F;
- R_2 зсувається якщо R_4 (3) = F;
- R_3 зсувається якщо R_4 (7) = F;
- вихідний біт алгоритму визначається результатом логічної операції XOR, що виконується над старшими бітами регістрів та мажоритарних функцій певних регістрів.

Принцип алгоритму A5/2 показаний на рис. 2.3.

Алгоритм A5/3 використовує блоковий шифр Касумі. На виході алгоритму A5/3 отримуються два рядки ключової послідовності розміром 114 біт, один з них використовується для шифрування та дешифрування висхідної лінії зв'язку, а інший для шифрування та дешифрування низхідного каналу [51].

Головною відмінністю алгоритму A5/0 від іншого алгоритму цього сімейства є те, що він не використовує процес шифрування та дешифрування інформації.

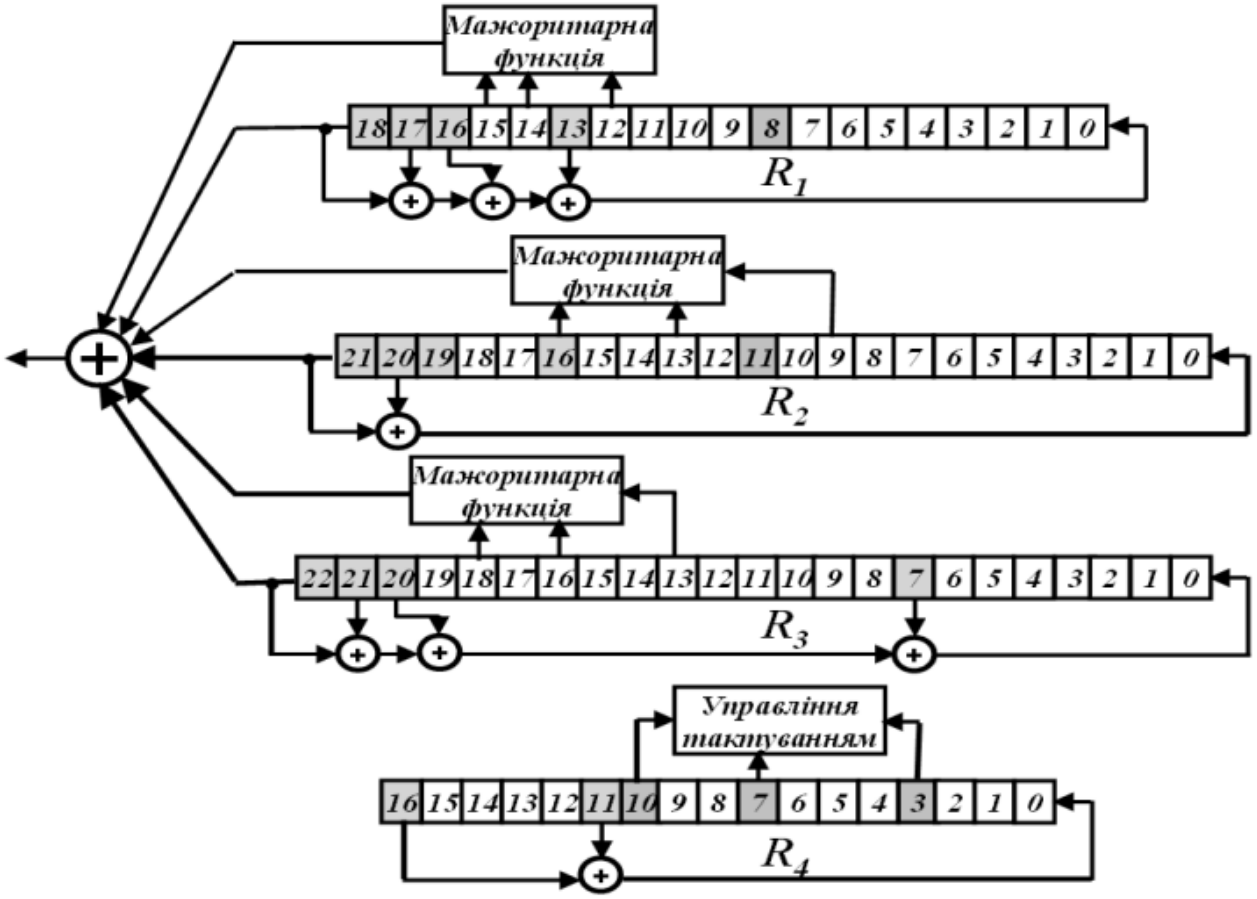

Рисунок 2.3 – Структура алгоритму A5/2

2.4 Криптостійкість, відомі уразливості та атаки

Для розроблення стандарту мобільного зв’язку GSM найважливішою умовою був захищений обмін між абонентами, що в свою чергу повинен бути стійким до атак, особливо до атак, що відбуваються в режимі реального часу. Проте в реальному життя прослуховуванням телефонних розмов займаються не лише злочинці, а й спецслужби, тому можна зробити висновок, що можливість підслуховування в цьому алгоритмі можлива і через це криптографічна стійкість алгоритму значно знижується. Також алгоритм A5/2 було модифіковано так, що криптографічна стійкість алгоритму неодмінно знизиться. Підсумовуючи можна сказати, що алгоритм A5/1 та A5/2 не є повністю захищеними від злому.

Після опублікування схеми та принципу роботи алгоритму, криптоаналітики та зловмисник почали роботу над пошуком вразливостей та використання їх для можливих атак. Серед знайдених вразливостей є [15]:

- 10 біт секретного ключа шифрування обов’язково занулені;
- окрім управління регістром зсуву відсутнє додаткове забезпечення захисту, наприклад, перехресні зв’язки між регістрами;
- надлишкова кількість службової інформації для шифрування та відомість цих даних криптоаналітику;
- на початку сеансу для перевірки з’єднання відбувається обмін порожніми повідомленнями, тобто нульовими, мається на увазі обмін за допомогою одного кадру;
- в A5/2 додано ще один регістр, що і здійснює керування регістрами.

Ці прогалини в безпеці алгоритму дають можливість побудови успішної атаки на систему.

Атака прямого перебору ключа передбачає складність виконання 2^{64}, оскільки сесійний ключ для шифрування має довжину 64 біти [15].

Атака Росса Андерсона полягає на вразливості занулення 10 біт секретного ключа, що знижує складність атаки до 2^{45}. Така атака заснована на певній ідеї, що полягає в наступному: для успішного зламування алгоритму необхідно правильно припустити вміст двох регістрів (R_1, R_2) та хоча б половину наповнення третього регістру, тобто R_3. Ця ідея дає змогу визначити тактування трьох регістрів алгоритму. Також ідея Андерсона використана в атаці за допомогою використання апаратних засобів Келлера і Зайтца (Keller and Seitz), таку атаку умовно можна поділити на два етапи: перший етап – визначення, другий етап – етап постобробки. На першому етапі визначається можливий претендент, що містить три регістри, а на другому етапі генерується перевірений претендент для успішної реалізації атаки [15; 52; 53].

Наступною атакою на алгоритм шифрування A5/1 є атака Голіча. Принципом цієї атаки є припущення першої половини кожного з трьох регістрів алгоритму. Припущені біти визначають тактування регістру протягом перших декількох циклів тактування. Потім виконується тактування алгоритму поки є припущені біти. Такий принцип дає те, що кожен вихідний біт алгоритму можна представити у вигляді лінійного рівняння внутрішніх бітів стану регістра, що складають верхню, тобто

другу частину наповнення трьох регістрів алгоритму. Потім продовжується отримування лінійних рівнянь за допомогою тактування послідовності, згодом система лінійних рівнянь розв'язується та отримуються необхідні дані для реалізації успішної атаки [53].

Атакою на алгоритм A5/2 є атака авторами якої є Вагнер та Голдберг. Суть цієї атаки полягає в тому, що для успішного зламування алгоритму необхідно лише правильно визначити початковий вміст регістру R_4. Вразливим є саме цей регістр, оскільки за його допомогою виконується управління тактуванням іншими регістрами алгоритму. Реалізації такої атаки виконується дуже швидко за допомогою наявного сучасного обладнання, оскільки складність виконання атаки порівняно з попередніми є низькою [15].

Також криптографами Шаміром, Бірюковим та Вагнером було представлено досить ефективний метод злому алгоритму шифрування A5/1 [15].

Таблиця 2.1 представляє скорочене подання наявних атак на алгоритм шифрування A5, де вказані основні характеристики.

Таблиця 2.1

Характеристики наявних атак на алгоритм шифрування A5

Атака	Головна ідея	Автор (якщо наявний)	Складність виконання атаки
Атака прямого перебору ключа	Ключем є згенерований сесійний ключ, що має довжину 64 біт та номер кадру (фрейму) вважається відомим	–	2^{64}
Атака на довжину ключа шифрування	Система шифрування ослаблюється за рахунок занулення 10 біт секретного ключа, довжиною 64 біт	–	2^{54}
Атака на	Визначення заповнення	Йован Голіч	2^{40}

Продовження таблиці 2.1

алгоритм шифрування A5/1	регістрів початкового стану за допомогою відомої частини послідовності (гами) довжиною 64 біт, що в свою чергу визначається з нульових повідомлень, які використовуються для перевірки встановлення з'єднання		
Атака на алгоритм шифрування A5/2	Для злому системи шифрування необхідно лише за допомогою прямого перебору знайти початковий вміст регістру, що управляє тактуванням	Вагнер, Голдберг	2^{17}
Атака на регістр зсуву	Використовує наявні вразливості у структурі регістру зсуву, часті перенавантаження цих регістрів у системі GSM та незворотні механізми руху регістрів	Шамір, Бірюков, Вагнер	2^{50}

2.5 Переваги та недоліки алгоритму

Алгоритм шифрування A5 наслідує певні переваги та недоліки як симетричного шифрування так і потокового алгоритму шифрування даних. Серед спільного можна вказати таку перевагу як простота алгоритму шифрування. Цей алгоритм шифрування даних відносно легко можна реалізувати апаратно. Головною причиною цієї переваги є використання регістру зсуву з лінійним зворотним зв'язком.

Також превагою цього алгоритму шифрування є висока швидкодія процесу виконання шифрування. Це досягається за допомогою використання двох

найпростіших та базових операцій, а саме: логічна операція XOR (додавання за модулем 2) та зсув регістру шифрування.

Алгоритми сімейства А5, особливо модифікації першопочаткового алгоритму шифрування, що наразі називається А5/1, були розроблені спеціально для задоволення потреб шифрування в системі мобільного зв'язку GSM. Оскільки вони призначені для використання у таких системах, то відповідно без цих алгоритмів шифрування система мобільного зв'язку не зможе забезпечити необхідне шифрування інформації своїх абонентів.

Недоліком є те, що алгоритм А5/2 був спеціально модифікований додаванням окремого регістру, що управляє тактуванням шифрування, що призвело до зниження криптостійкості цього алгоритму шифрування. Певним недоліком є занулені біти ключа, а саме 10 біт, що дозволяє побудувати на цій вразливості атаки та провести їх успішну реалізацію. Також окрім управління тактуванням відсутні будь-які інші зв'язки між регістрами, що також спрощує реалізацію атак, і відповідно знижує криптостійкість алгоритму.

Відмінності розглянутих потокових алгоритмів шифрування, а саме А5/1, RC4 та СТРУМОК представлено в табл. 2.2.

Таблиця 2.2

Порівняння потокових алгоритмів

Характеристика	А5/1	RC4	СТРУМОК
Довжина ключа	64 біт	40-2048 біт	256 біт, 512 біт
Шифрування	Біт-орієнтоване	Байт-орієнтоване	Біт-орієнтоване
Ядро шифрування	Регістр зсуву з лінійним зворотним зв'язком	Генератор псевдовипадкових біт	Регістр зсуву з лінійним зворотним зв'язком
Застосування	В системі GSM	SSL/TLS	В інформаційних, телекомунікаційних, інформаційно-телекомунікаційних системах
Відомі успішні атаки	Так	Так	Ні

Висновки за розділом 2

В другому розділі даної дипломної роботи було розглянуто алгоритм потокового симетричного шифрування A5, наведено характеристики та інші важливі відомості для досягнення мети дипломної роботи. Головними результатами виконання завдань другого розділу є:

- проведено аналіз сімейства алгоритмів A5, подано які алгоритми складають це сімейство та чому їх так називають;
- історію створення першопочаткового алгоритму, що зараз називається A5/1, та причину модифікацій та утворення похідних алгоритмів;
- застосування алгоритмів сімейства A5;
- принцип роботи алгоритму A5, що полягає в використанні операції XOR та регістру зсуву, що є найпростішими базовими операціями в шифруванні;
- розглянуто відмінності похідних алгоритмів сімейства;
- головними відмінностями алгоритмів є те, що у алгоритмі A5/0 відсутній процес шифрування, інформація передається без перетворень відкритих даних, тобто у незашифрованому вигляді, алгоритм A5/2 модифікований за допомогою додавання ще одного регістру, що керує процесом тактування у шифрування, алгоритм A5/3 є блоковим алгоритмом шифрування та базується на алгоритмі шифрування Касумі;
- розглянута криптографічна стійкість алгоритму та наведені наявні вразливості, на основі яких побудовані відомі атаки;
- надано перелік атак на алгоритм шифрування та подано головну ідею реалізації цих атак;
- наведені переваги та недоліки алгоритму шифрування A5.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ ПОТОКОВОГО АЛГОРИТМУ ШИФРУВАННЯ A5/1

3.1 Програмне середовище для реалізації алгоритму та його особливості

Для реалізації потокового алгоритму шифрування A5/1 використано програмне середовище від корпорації Microsoft, що являє собою інтегроване середовище розробки переважно використовується для програмного забезпечення,а саме Visual Studio 2017. Окрім розробки програмного забезпечення використовується для побудови веб-сайтів, веб-сервісів, веб-додатків та мобільних додатків [54; 55].

Для реалізації розробки програмного забезпечення Microsoft Visual Studio використовує такі платформи: Windows Forms, , Windows Store, Windows API, Windows Presentation Foundation та Microsoft Silverlight. Також програмне середовище містить редактор коду, який підтримує властивість IntelliSense, суть якої полягає в автоматичному доповненні написаного коду, тобто допомагає швидше та без помилок використовувати базові функції та необхідні відомі ключові слова з певних необхідних для роботи бібліотек. Також важливою можливістю Microsoft Visual Studio є можливість додавання сторонніх плагінів, що підвищують продуктивність роботи, розширюють можливий функціонал програмного середовища та сферу використання [55].

Перевагою Microsoft Visual Studio є те, що програмне середовище підтримує 36 різних мов програмування, окрім цього у редактора коду та відлагоджувача (дебагера, debbuger) є можливість працездатності для майже будь-якої мови програмування. Вбудованими мовами програмування є C, C++, C#, XML, XSLT, Visual Basic .NET та інші. Підтримка таких популярних мов програмування як Python, Ruby, Node.js та деяких інших можлива за допомогою використання

спеціальних плагінів, що підтримуються середовищем програмування Microsoft Visual Studio [55].

Для реалізації потокового алгоритму шифрування A5/1 використано мову програмування C. Вперше ця мова програмування бала використана у 1970-х роках завдяки Деннісу Річі, що створив цю мову програмування. Мова програмування C вважається мовою програмування загального призначення, проте найширше використовується при створенні операційних систем та драйверів для пристроїв, стеках протоколів, комп'ютерній архітектурі та інших системах [56].

На початку розвитку мови програмування C, вона використовувалася для створення утиліт, що працюють на платформі Unix, вже згодом вона стала однією з найбільш популярних та найбільш використовуваних мов програмування [56].

Багато мов програмування можна вважати наслідниками C, оскільки вони запозичили синтаксис, принцип та взагалі ідею. До таких мов програмування можна віднести C++, C#, Go, Java та інші. Багато структур та основних функцій вони побудували за алгоритмом мови C, а також мають схожий синтаксис [56].

Основний метод розширити функціонал та полегшити реалізацію власних програм є використання бібліотек. Цим поняттям позначають набір певних функцій для виконання певних завдань. Для підключення цих бібліотек до файлу з проектом використовуються заголовки, за допомогою яких зрозуміло програмі, які функції можна використовувати та як правильно це зробити [56].

В реалізації алгоритму використано дві бібліотеки: stdio.h та locale.h. Обидві бібліотеки є стандартними бібліотеками мови програмування C. Для сумісності мов програмування C та C++ можна використовувати ці ж бібліотеки програмування. Назва бібліотеки stdio.h є скороченням від англійської standard input/output header, що означає стандартний заголовок вводу/виводу. З назви бібліотеки відразу зрозуміло вміст її файлу, а саме оголошення функцій і типів та визначення макросів і констант, що використовуються для виконання таких операцій, як введення та виведення. В загальному функції, що описані в цьому заголовку можна поділити на дві групи: функції, що призначені для виконання операцій з файлами та функції, що призначені для виконання операцій введення/виведення. Бібліотека locale.h

використовується для вирішення завдань, що потребують рішення локалізації. Ця бібліотека надає дві головні функції: localeconv та setlocale. Перша функція призначена для забезпечення доступу до поточної локалі, у той час як друга функція призначена для зміни цієї локалі [57; 58; 59].

3.2 Основні принципи реалізації алгоритму

Як відомо, з попереднього розділу дипломної роботи, де детально було розглянуто принцип роботи алгоритму A5/1, передача інформації в даному потоковому алгоритмі виконується за допомогою кадрів (фреймів). Довжина таких кадрів становить 114 біт інформації. На початку роботи алгоритму регістри R_1, R_2 та R_3 обнуляються для отримання початкового стану регістрів. Також для справної роботи алгоритму необхідні такі ключові дані, як сеансовий ключ довжиною 64 біт, що створений алгоритмом A8, який також використовується для забезпечення безпеки системи мобільного зв'язку GSM, та номер кадру, що підлягає шифруванню довжиною 22 біт [15].

Важливим етапом реалізації алгоритму шифрування A5/1 є процес ініціалізації. Цей процес є обов'язковим до виконання для кожного кадру, що підлягає шифруванню та розшифруванню. Для цього етапу необхідні секретний ключ шифрування довжиною 64 біт та відповідний номер кадру довжиною 22 біт [15].

На етапі ініціалізації для справної роботи алгоритму необхідно виконати такі кроки [15]:

- першим кроком є обнулення регістрів зсуву, що також називаються регістри зсуву з лінійним зворотним зв'язком, для цього біти у трьох регістрах встановлюються в 0;
- далі відбувається змішування бітів ключа з відповідним значенням регістру, для цього виконується логічна операція XOR, або ж інша назва додавання за модулем 2, відповідного біта ключової послідовності з лівими бітами трьох регістрів, ця операція відбувається циклічно, стільки разів скільки довжина ключа,

на цьому етапі відбувається синхронізація регістрів зсуву R_1, R_2 та R_3, синхронізація регістрів відбувається за рахунок зсуву кожного регістру на один крок;

- наступним кроком є повторення попередньо операції, але на цей раз замість ключового потоку використовується потік кадрів довжиною 22 біт, на цьому етапі також відбувається виконання логічної операції XOR, або ж додавання за модулем 2, відповідного біта номеру кадру та лівих біт регістрів зсуву, цей процес також забезпечує синхронізацію регістрів зсуву R_1, R_2 та R_3;
- на цьому етапі відбувається виконання циклу 100 разів для синхронізації генератора псевдовипадкових чисел, для цього процесу використовується мажоритарна функція, яка застосовується для визначення регістру зсуву для якого буде відбуватися синхронізація.

Визначення мажоритарної функції є важливим етапом для роботи алгоритму шифрування A5/1, оскільки вона визначає процес синхронізації певного регістру зсуву з лінійним зворотним зв'язком. Обчислення цієї функції представлене формулою (2.4). В цю функцію передаються значення бітів синхронізації кожного з трьох регістрів зсуву та за допомогою булевих операцій обчислюється її значення. З формули можна сказати, що мажоритарна функція відповідає 1, якщо більшість переданих параметрів дорівнює 1, а якщо значення дорівнює 0, то це означає, що більшість бітів синхронізації регістрів дорівнює 0. Обрахунок цієї функції відбувається перед визначенням тактового імпульсу. Визначення регістру, що буде синхронізуватися виконується таким чином: якщо біт синхронізації певного регістру відповідає, тобто дорівнює значенню мажоритарно функції, то цей регістр зсувається, тобто синхронізується [15].

Потік бітів ключа створюється за допомогою генератора ключів, що відбувається використовуючи тактовий імпульс. Для початку виконується операція обчислення мажоритарної функції, а згодом створюється ключовий потік бітів. Далі відбувається визначення процесу синхронізації. Тобто за допомогою значення мажоритарної функції відбувається визначення регістрів синхронізації, процес визначення цих регістрів описаний вище [15].

3.3 Аналіз програмної реалізації алгоритму

Програмна реалізація представляє собою реалізацію алгоритму шифрування A5/1 за допомогою мови програмування C. Вихідний код програми представлений в Додатку А. Код програми призначений для використання у навчальних цілях, що і вимагає мета цієї дипломної роботи. Ця програмна реалізація призначена для вивчення властивостей, розуміння принципу роботи та аналізу потокових алгоритмів шифрування, що є класом симетричних алгоритмів на прикладі потокового алгоритму A5/1.

Програмна реалізація використовує ключ та відомий вказаний номер кадру. Ця реалізація видає результат шифрування даних від Абонента до Базової станції мобільної системи та навпаки, тобто від Базової станції до Абонента.

Програмна реалізація складається з незалежних функцій для простішого та більш зрозумілого процесу виконання алгоритму. Також в програмі використовуються дві стандартні бібліотеки мови C, а саме stdio.h та locale.h, призначення яких описано в цьому розділі у першому пункті.

Головна функція програми викликає функцію cipher(), де оголошено ключ та відомий номер кадру, також оголошено послідовність від Абонента до Базової станції та навпаки, що обчислюється спеціальною функцією a_to_b(). Також ця функція викликає функцію setkey(), яка використовує параметри заданого ключа та кадру. Функція cipher() виводить на екран початкові задані дані та отримані дані.

Також окрім описаних функцій програма виконує ініціалізацію трьох регістрів зсуву з лінійно зворотним зв’язком, визначає біти синхронізації, що використовуються для управління тактуванням алгоритму, це 8 біт для першого регістру, 10 біт для другого регістру та 10 біт для третього регістру, в подальшому ці біти регістрів будуть використовуватися для визначення процесу синхронізації. Окрім цих даних, визначаються біти тактування регістрів, що представлені визначеними многочленами, для першого регістру це біти 13, 16, 17, 18, для другого це біти 20 та 21, а для третього – 7, 20, 21 та 22. Також визначені вихідні біти, що використовуються для генерації вихідних результатів алгоритму.

Для обчислення мажоритарної функції використовується функція major(), головне призначення це визначення регістрів синхронізації. Для синхронізації регістрів відповідно до їх бітів синхронізації використовується функція sync(), де порівнюється значення біта синхронізації та обчисленої мажоритарної функції. Також для синхронізації всіх трьох регістрів зсуву, що використовується при ініціалізації ключа шифрування використовуються функція clockall().

Для налаштування генерації ключового потоку використовується функція setkey(), в якій спочатку обнуляються всі регістри, потім відбувається завантаження ключа в кожен регістр та відбувається логічна операція XOR з кожним бітом ключа та молодшим бітом регістру, тобто крайнім лівим на кожному такті. Така операція також виконується і для номеру кадру. Відповідно загружається номер кадру в кожен регістр та потім відбувається логічна операція XOR з кожним бітом кадру та крайнім лівим бітом регістру, тобто молодшим бітом. Для завершення генерація відбувається запуск регістру зсуву для 100 тактів зміщення ключа та кадру.

Для генерації вихідної послідовності ініціалюзується 15 байтовий буфер для послідовності від Абонента до Базової станції та 15 байтовий буфер для зворотного напрямку. Всього генерується 228 біт послідовності, проте вона розділена по 114 біт для кожного напрямку передачі інформації. Для цього процесу призначена функція a_to_b().

Для перевірки справності виконання алгоритму шифрування A5/1 використаємо загальновідомі дані, що були опубліковані під час першої реалізації алгоритму шифрування Марком Брікено, Яном Голдбергом та Девідом Вагнером. Для такої перевірки використовується номер кадру 0x134 та ключ 0x1223456789ABCDEF. Вихідна послідовність при таких даних повинна бути 0x534EAA582FE8151AB6E1855A728C00, 0x24FD35A35D5FB6526D32F906DF1AC0 [60]. Для перевірки програмної реалізації введемо початкові дані в код та запустимо виконання алгоритму. Результат шифрування алгоритму A5/1 в програмній реалізації показано на рис. 3.1.

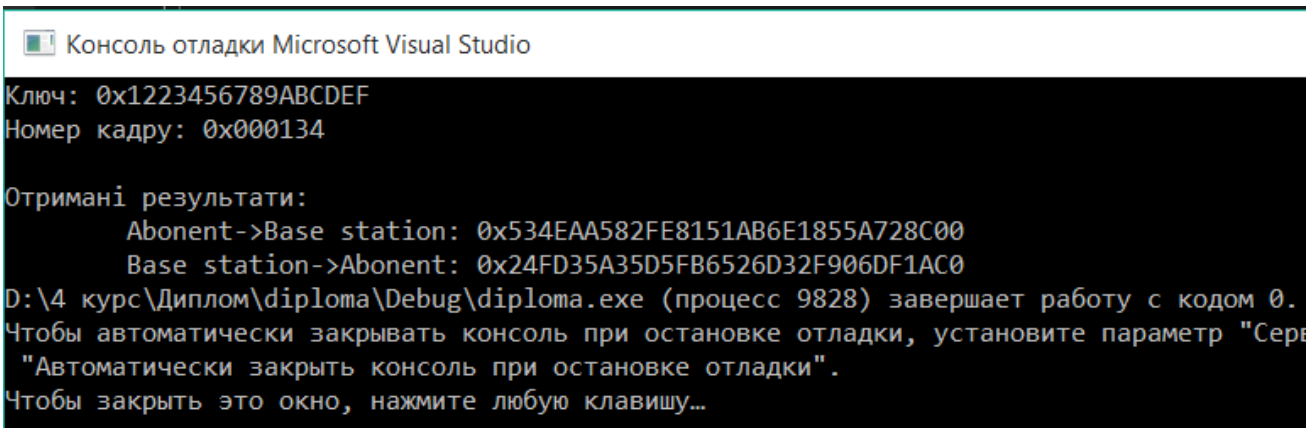

Рисунок 3.1 – Результат виконання алгоритму з тестовими даними

Для перевірки робити алгоритму з іншими даними використаємо довільний ключ та довільний номер кадру. Для прикладу візьмемо ключ 0x254315CD3611AADF та номер кадру 0x143. Результат виконання алгоритму з такими даними показано на рис. 3.2.

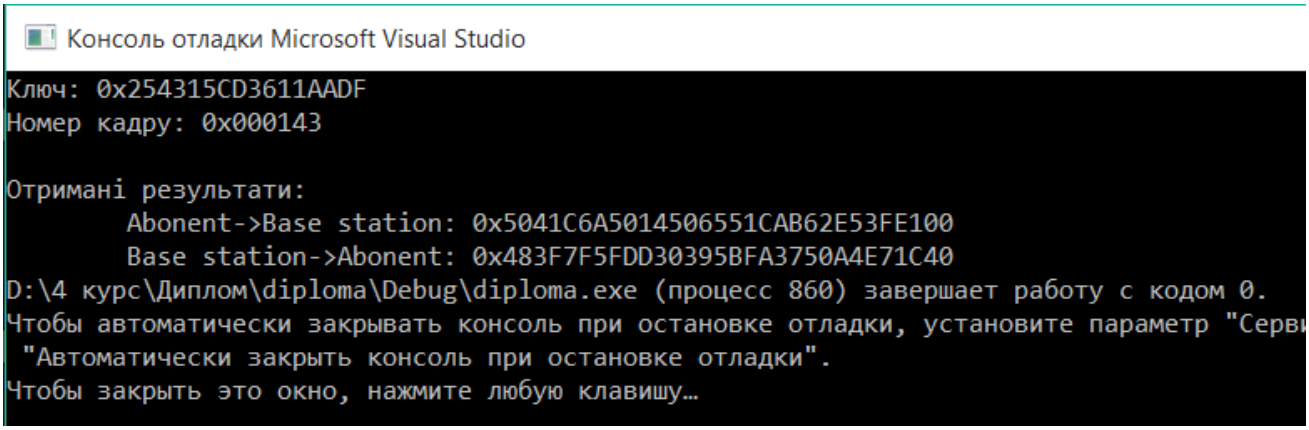

Рисунок 3.2 – Результат виконання алгоритму з довільними даними

Отримані результати послідовності від Абонента до Базової станції становить послідовність 0x5041C6A5014506551CAB62E53FE100, а в зворотному напрямку інформації становить 0x483F7F5FDD30395BFA3750A4E71C40. Оскільки результати виконання програми збігаються з наданими в роботі вчених та при виконання програми з довільними даними не виникло ніяких помилок та вихідні дані відрізняються можна зробити висновок, що програмна реалізація справна та може бути використана в навчальних цілях для розгляду принципу роботи потокового алгоритму шифрування A5/1.

Потоковий алгоритм шифрування A5/1 може бути використаний для забезпечення захищеного каналу зв’язку для обміну інформацією між безпілотним літальним апаратом (БПЛА) та пунктом управління. Саме цей алгоритм шифрування має ряд переваг серед інших потокових алгоритмів шифрування для забезпечення мети створення захищеного обміну інформацією між БПЛА та пунктом управління. Ці переваги проявляються у таких характеристиках як швидкість та простота. Алгоритм A5/1 є досить простим для розуміння та програмної реалізації, тому вимагає менших обчислюваних ресурсів апаратного забезпечення, крім того є достатньо швидким для передачі інформації. Ці переваги покривають недолік криптографічної стійкості алгоритму, оскільки більш важливим є швидка передача даних та невелика затратність апаратного забезпечення.

3.4 Моделювання захищеного каналу управління безпілотного літального апарату

Для прикладу розробки моделі технічної реалізації захищеного каналу передачі інформації в безпілотному літальному апараті за допомогою алгоритму потокового шифрування A5/1 використано таке апаратне забезпечення: БПЛА DJI AIR 2S та плату Arduino Due. Плата Arduino Due використовується для забезпечення програмної реалізації алгоритму A5/1 для захищеної передачі даних. Радіозв’язок БПЛА складається з таких компонентів: антени, приймач, відеопередавач, польотного контролеру та FPV камери. Схему БПЛА з основними компонентами показано на рис. 3.3.

Для створення захищеного каналу передачі даних БПЛА модифікується додаванням плати Arduino Due, з програмною реалізацією алгоритму A5/1. Ця плата забезпечує захищену передачу інформації. Схему БПЛА з основними компонентами після модифікації показано на рис. 3.4. Алгоритм A5/1 буде забезпечувати надійну передачу даних між БПЛА та пунктом управління завдяки його швидкості та простоті, ці переваги описано вище у пункту 3.3. Завдяки такій модифікацій пункт

управління зможе отримувати захищену інформацію за прийнятний час, що згодом буде використана для швидкого реагування на події.

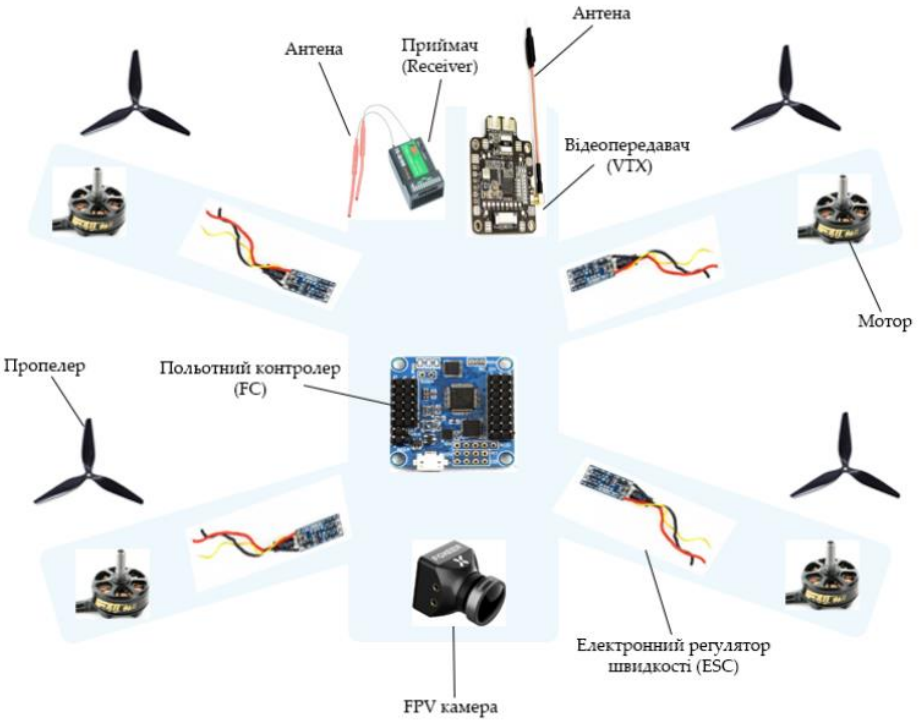

Рисунок 3.3 – Схема БПЛА до модифікації

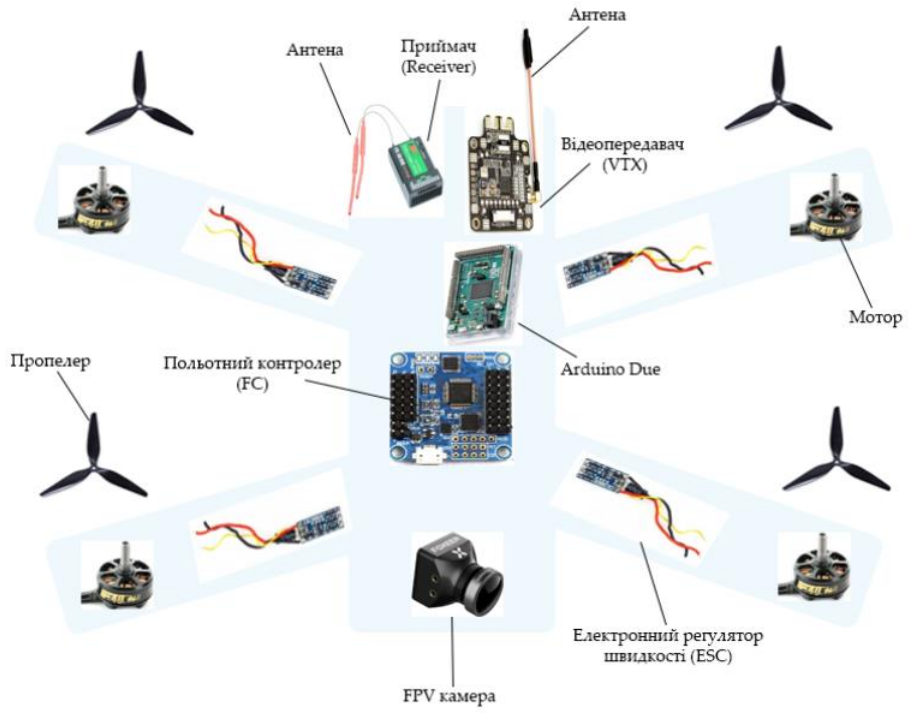

Рисунок 3.4 – Схема БПЛА після модифікації

Як зазначено вище, захищений канал передачі даних буде створений завдяки використанню потокового алгоритму шифрування, схему захищеного каналу зв'язку між БПЛА та пультом керування показано на рис. 3.5. Варто зазначити, що пультом керування в БПЛА DJI AIR 2S є мобільний пристрій, тому для розшифрування отриманої інформації необхідне спеціальне програмне забезпечення, що буде виконувати цю функцію.

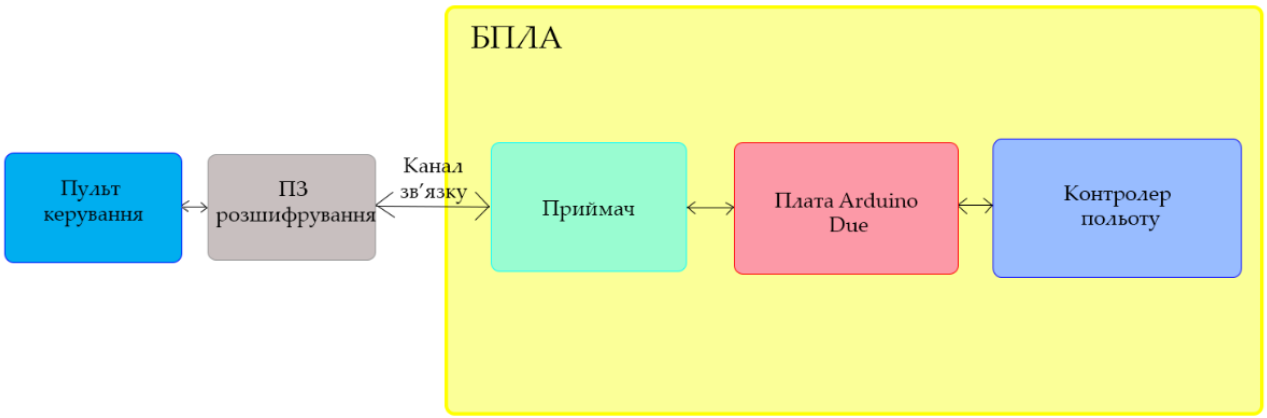

Рисунок 3.5 – Схема зв'язку між БПЛА та пультом керування

Висновки за розділом 3

У третьому розділі дипломної роботи було представлено програмну реалізацію потокового алгоритму шифрування A5/1. Головними результатами виконання завдань третього розділу є:

- вибір програмного середовища для реалізації алгоритму шифрування A5/1, цим продуктом є Visual Studio 2017 від розробника Microsoft, перевагою цієї платформи є підтримування багатьох поширених мов програмування, широкий спектр застосування, використання для різних потреб користувача, додаткові можливості для легшого та правильного розроблення продуктів, можливість підтримки мов не лише вбудованих, а й інших з використанням спеціально розроблених плагінів;
- вибір мови програмування для реалізації алгоритму шифрування A5/1, цією мовою програмування є мова C, яка є широко використовуваною та за допомогою цієї мови реалізовано багато продуктів, які є відомими та продуктивними, від цієї

мови програмування багато функцій і синтаксис успадкували новіші мови програмування;

- розглянуто основні принципи реалізації алгоритму шифрування A5/1, серед яких визначено такі ключові моменти: використання сесійного ключа довжиною 64 біт, номеру фрейму довжиною 22 біт, процес ініціалізації, обчислення мажоритарної функції, за допомогою якої визначаються регістри, що формують процес синхронізації;
- детальний аналіз реалізації алгоритму шифрування A5/1, опис роботи програми, що складається з декількох незалежних функцій, які допомагають полегшити процес реалізації шифру, розгляд ініціалізації змінних та інших компонентів коду;
- перевірка працездатності програми використовуючи загальновідомі дані, що були використані під час першої реалізації цього алгоритму шифрування, також перевірено чи програма працює для введених інших даних, оскільки ніяких помилок під час виконання алгоритму не було помічено, можна зробити висновок, що програмна реалізація працює справно;
- програмна реалізація алгоритму може бути використана для забезпечення захищеного каналу зв’язку між БПЛА та пунктом управління;
- розроблено модель технічної реалізації захищеного каналу передачі інформації в безпілотному літальному апараті використовуючи алгоритм шифрування A5/1.

ВИСНОВКИ

В дипломній роботі розроблено модель захищеного каналу передачі інформації в безпілотному літальному апараті з використанням симетричного потокового алгоритму A5/1. Головними результатами отриманими впродовж написання дипломної роботи є:

- розглянуто симетричні алгоритми шифрування, що поділяються на блокові та потокові алгоритми шифрування, найвідоміших представників та принцип їх роботи, переваги та недоліки цього типу шифрування;
- розглянуто потоковий алгоритм шифрування A5, а саме його історію створення, представників сімейства, сферу застосування, принцип роботи та відмінності алгоритмів, криптостійкість, відомі атаки, переваги та недоліки;
- програмно реалізовано потоковий алгоритм шифрування A5/1, описано сферу застосування потокового алгоритму;
- розроблено модель технічної реалізації захищеного каналу передачі інформації в безпілотному літальному апараті використовуючи алгоритм шифрування A5/1.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Погромська Г. С. Візуалізація шифрування файлів блочними симетричними алгоритмами з посиленням криптостійкості [Електронний ресурс] // Technics and technology. #15 – Режим доступу: https://repository.moippo.mk.ua/upload/17-06-20_49026591.pdf.

2. Криптографія: загальні визначення, класифікація. Асиметричні та симетричні криптоалгоритми, їх порівняння [Електронний ресурс]: Ляміна Уляна Методи сучасної криптографії. – Режим доступу: https://lyamina09.wordpress.com/2014/03/12/%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D1%96%D1%8F-%D0%B7%D0%B0%D0%B3%D0%B0%D0%BB%D1%8C%D0%BD%D1%96-%D0%B2%D0%B8%D0%B7%D0%BD%D0%B0%D1%87%D0%B5%D0%BD%D0%BD%D1%8F-%D0%BA%D0%BB/.

3. Скремблювання [Електронний ресурс]: Вікіпедія Вільна енциклопедія. – Режим доступу: https://uk.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B5%D0%BC%D0%B1%D0%BB%D1%8E%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F.

4. В. А. Лужецький, А. В. Остапенко, Аналіз алгоритмів симетричного блокового шифрування [Електронний ресурс] / Володимир Андрійович Лужецький, Аліна Василівна Остапенко // Міжнародний науково-технічний журнал “Інформаційні технології та комп’ютерна інженерія” № 3, 2012 – Режим доступу: https://itce.vntu.edu.ua/index.php/itce/article/view/95/102.

5. Мережа Фейстеля [Електронний ресурс]: Вікіпедія Вільна енциклопедія. – Режим доступу: https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%80%D0%B5%D0%B6%D0%B0_%D0%A4%D0%B0%D0%B9%D1%81%D1%82%D0%B5%D0%BB%D1%8F.

6. Гуменюк Л. М., Сидоренко Ю. В., Використання протоколу MTProto для створення сервісу обміну ключами [Електронний ресурс] / Гуменюк Л. М.,

Сидоренко Ю. В // XVI Міжнародної науково-практичної конференції аспірантів, магістрантів і студентів «Сучасні проблеми наукового забезпечення енергетики» – 2018 – Режим доступу: https://tef.kpi.ua/files/pdf/1tezi_tom2_1524728034.pdf.

7. Schneier B. Unbalanced Feistel Networks and Block-Cipher Design / B. Schneier, J. Kelsey // Counterpane Systems.

8. Kam J. Structured design of substitution-permutation encryption networks / J. Kam, G. Davida // IEEE Transactions on Computers. – 1979. – Vol. 28, No10. – P. 747.

9. Borisov N. Multiplicative differentials / N. Borisov, M. Chew, R. Johnson // Fast Software Encryption: 9th International Workshop on table of contents. – 2002. – Vol. 2365. – pp. 17-33.

10. Шеннон К. Работы по теории информации и кибернетике / К. Шеннон; переклад з англійської – М.: Изд-во иностранной литературы, 1963. – 830 с.

11. Д. Маковецький, П. Гаранюк, Аналіз методів та загальних властивостей генерування ключів для криптографічних додатків [Електронний ресурс] / Д. Маковецький, П. Гаранюк – 2014. – Режим доступу: https://www.ctp.uad.lviv.ua/images/ktd/31_makoveckiy.pdf.

12. What Is DES (Data Encryption Standard): DES Algorithm and Operation [Electronic resource]: Simplilearn. – Access: https://www.simplilearn.com/what-is-des-article.

13. Data Encryption Standard [Electronic resource]: Tutorials Point. – Access: https://www.tutorialspoint.com/cryptography/data_encryption_standard.htm.

14. Методи та алгоритми симетричної криптографії: Навч. посіб. / Кузнецов О. О., Євсеев С. П., Смірнов О. А., Мелешко Є. В., Король О. Г. – Кіровоград : Вид. КНТУ, 2012. – 316 с.

15. Прикладна криптологія: системи шифрування : підручник / О. Г. Корченко, В. П. Сіденко, Ю. О. Дрейс – Житомир : ДУТ, 2014. – 448 с.

16. J. Orlin Grabbe The DES Algorithm Illustrated [Electronic resource] / J. Orlin Grabbe // Laissez Faire City Times, Vol 2, No. 28. – Access: https://page.math.tu-berlin.de/~kant/teaching/hess/krypto-ws2006/des.htm.

17. What is the DES Algorithm? Understanding Data Encryption Standard [Electronic resource]: Intellipaat. – Access: https://intellipaat.com/blog/what-is-des-algorithm/.

18. How do I decrypt Triple DES encryption? [Electronic resource]: JANET-PANIC.COM World History Portal. – Access: https://janet-panic.com/how-do-i-decrypt-triple-des-encryption/.

19. Advanced Encryption Standard [Електронний ресурс]: Вікіпедія Вільна енциклопедія. – Режим доступу: https://uk.wikipedia.org/wiki/Advanced_ Encryption_Standard.

20. Advanced Encryption Standard [Electronic resource]: Tutorials Point. – Access: https://www.tutorialspoint.com/cryptography/advanced_encryption_standard.htm.

21. What is AES encryption and how does it work? [Electronic resource]: Cybernews. – Access: https://cybernews.com/resources/what-is-aes-encryption/.

22. Advanced Encryption Standard (AES) [Electronic resource]: GeeksforGeeks.– Access: https://www.geeksforgeeks.org/advanced-encryption-standard-aes/.

23. What Is AES Encryption? [The Definitive Q&A Guide] [Electronic resource]: Trenton Systems. – Access: https://www.trentonsystems.com/blog/aes-encryption-your-faqs-answered.

24. What is AES encryption and how does it work? [Electronic resource]: Comparitech. – Access: https://www.comparitech.com/blog/information-security/what-is-aes-encryption/.

25. An Introduction to Stream Ciphers vs. Block Ciphers [Electronic resource]: JSCAPE. – Access: https://www.jscape.com/blog/stream-cipher-vs-block-cipher.

26. Stream Cipher 101: Definition, Usage & Comparisons [Electronic resource]: Okta. – Access: https://www.okta.com/identity-101/stream-cipher/.

27. Потоковий шифр [Електронний ресурс]: Вікіпедія Вільна енциклопедія.– Режим доступу: https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%B2%D0%B8%D0%B9_%D1%88%D0%B8%D1%84%D1%80.

28. Block Cipher vs. Stream Cipher [Electronic resource]: Crashtest Security. – Access: https://crashtest-security.com/block-cipher-vs-stream-cipher/.

29. What Is RC4 and Why Is It A Vulnerability [Electronic resource]: Crashtest Security. – Access: https://crashtest-security.com/disable-ssl-rc4/.

30. RC4 Encryption Algorithm [Electronic resource]: GeeksforGeeks. – Access: https://www.geeksforgeeks.org/rc4-encryption-algorithm/.

31. What is RC4? Is RC4 secure? [Electronic resource]: Encryption consulting. – Access: https://www.encryptionconsulting.com/education-center/what-is-rc4/.

32. What is RC4 Encryption? [Electronic resource]: GeeksforGeeks. – Access: https://www.geeksforgeeks.org/what-is-rc4-encryption/.

33. Fluhrer, Mantin and Shamir attack [Electronic resource]: Wikipedia The free encyclopedia. – Access: https://en.wikipedia.org/wiki/Fluhrer,_Mantin_and_Shamir_attack.

34. Rivest Cipher 4 (RC4) [Electronic resource]: Infosec. – Access: https://resources.infosecinstitute.com/topic/rivest-cipher-4-rc4/.

35. Національний стандарт України: ДСТУ 8845:2019 Інформаційні технології. Криптографічний захист інформації. Алгоритм симетричного потокового шифрування від 01.10.2019.

36. Криптографічна стійкість шифр [Електронний ресурс]: Вікіпедія Вільна енциклопедія. – Режим доступу: https://uk.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D1%96%D1%87%D0%BD%D0%B0_%D1%81%D1%82%D1%96%D0%B9%D0%BA%D1%96%D1%81%D1%82%D1%8C.

37. Дробиш Р. С., Люта М. В., Аналіз існуючих методів захисту та шифрування інформації [Електронний ресурс] / Дробиш Р. С., Люта М. В. // II Всеукраїнська конференція здобувачів вищої освіти і молодих учених «Інноватика в освіті, науці та бізнесі: виклики та можливості». – Режим доступу: https://er.knutd.edu.ua/bitstream/123456789/19525/1/Innovatyka2021_V1_P250-253.pdf.

38. David R. Mirza Ahmad, Ryan Russell, Hack proofing your network (2nd ed.) Rockland, MA: Syngress.

39. Attacks on symmetric ciphers [Electronic resource]: Ten minute tutor. – Access: https://www.tenminutetutor.com/data-formats/cryptography/attacks-on-symmetric-ciphers/.

40. Chapter 7 Encryption and Attacks [Electronic resource]: Sandilands. – Access: https://sandilands.info/crypto/EncryptionandAttacks.html#x15-720007.5.

41. Attacks on Symmetric Key [Electronic resource]. – Access: https://www.cs.clemson.edu/course/cpsc424/material/Cryptography/Attacks%20on%20Symmetric%20Key.pdf.

42. Different Types of Cryptography Attacks [Electronic resource]: Infosectrain.– Access: https://www.infosectrain.com/blog/different-types-of-cryptography-attacks/.

43. GSM A5 Files Published on Cryptome [Electronic resource]: Cryptome. – Access: https://cryptome.org/jya/a51-pi.htm.

44. Advantages and Disadvantages of Symmetric and Asymmetric Key Encryption Methods [Electronic resource]: TechRejects - Helpful Tech Tips. – Access: https://techrejects.blogspot.com/2014/08/advantages-disadvantages-symmetric-asymmetric-key-encryption-methods.html.

45. What are the advantages and disadvantages of symmetric key cryptography? [Electronic resource]: Quora. – Access: https://www.quora.com/What-are-the-advantages-and-disadvantages-of-symmetric-key-cryptography.

46. What are the advantages of Symmetric Algorithms? [Electronic resource]: Tutorials Point. – Access: https://www.tutorialspoint.com/what-are-the-advantages-of-symmetric-algorithms.

47. І.В.Купріянчук, Я.М. Царівська, Методи захисту інформації в мережах GSM [Електронний ресурс] / І.В.Купріянчук, Я.М. Царівська // Науково-технічна конференції «Інформаційна безпека України» Київський національний університет імені Тараса Шевченка – 2016 – Режим доступу: http://docplayer.net/48254139-Informaciyna-bezpeka-ukrayini.html.

48. Alex Biryukov, Adi Shamir, David Wagner, Real Time Cryptanalysis of A5/1 on a PC [Electronic resource] / Alex Biryukov, Adi Shamir, David Wagner // Fast Software Encryption Workshop – 2000 – Access: https://cryptome.org/a51-bsw.htm.

49. Elad Barkan, Eli Biham, Nathan Keller, Instant Cipertext-Only Cryptanalysis of GSM Encrypted Communication [Electronic resource] / Elad Barkan, Eli Biham, Nathan Keller. – Access: https://cryptome.org/gsm-crack-bbk.pdf.

50. Прикладная криптология: методы шифрования : учебное пособие / Б. С. Ахметов, А. Г. Корченко, В. П. Сиденко и др. – Алматы : КазНИТУ имени К. И. Сатпаева, 2015. – 496 с.

51. A5 (шифр) [Електронний ресурс]: Вікіпедія Вільна енциклопедія. – Режим доступу: https://uk.wikipedia.org/wiki/A5_(%D1%88%D0%B8%D1%84%D1%80).

52. T. Anand, M. Shanmugam, B. Santhoshini, Rainbow table attack on 3rd generation GSM Telephony secure algorithm - A5/3 [Electronic resource] / T. Anand, M. Shanmugam, B. Santhoshini // International Journal of Recent Technology and Engineering (IJRTE), Volume-7, Issue-5S4, February 2019. – Access: https://www.ijrte.org/wp-content/uploads/papers/v7i5s4/E10170275S419.pdf.

53. Timo Gendrullis, Martin Novotn´y, Andy Rupp, A Real-World Attack Breaking A5/1 within Hours [Electronic resource] / Timo Gendrullis, Martin Novotn´y, Andy Rupp // Horst G¨ortz Institute for IT-Security, Ruhr-University Bochum, Germany. – Access: https://eprint.iacr.org/2008/147.pdf.

54. Timo Gendrullis Hardware-Based Cryptanalysis of the GSM A5/1 Encryption Algorithm [Electronic resource] / Timo Gendrullis // Departament of Electrical Engineering & Information Sciences Ruhr-University Bochum – 2008. – Access: https://www.yumpu.com/en/document/read/6469997/hardware-based-cryptanalysis-of-the-gsm-a5-1-encryption-algorithm.

55. Microsoft Visual Studio [Електронний ресурс]: Вікіпедія Вільна енциклопедія. – Режим доступу: https://uk.wikipedia.org/wiki/Microsoft_Visual_Studio.

56. Microsoft Visual Studio [Electronic resource]: Wikipedia The free encyclopedia. – Access: https://en.wikipedia.org/wiki/Microsoft_Visual_Studio.

57. C (programming language) [Electronic resource]: Wikipedia The free encyclopedia. – Access: https://en.wikipedia.org/wiki/C_(programming_language).

58. Stdio.h [Електронний ресурс]: Вікіпедія Вільна енциклопедія. – Режим доступу: https://uk.wikipedia.org/wiki/Stdio.h.

59. stdio.h [Електронний ресурс]: Вікіпедія Вільна енциклопедія. – Режим доступу: https://ru.wikipedia.org/wiki/Stdio.h.

60. locale.h [Електронний ресурс]: Вікіпедія Вільна енциклопедія. – Режим доступу: https://uk.wikipedia.org/wiki/Locale.h.

61. Nagendar Yerukala, V Kamakshi Prasad, and Allam Apparao Performance and Statistical Analysis of Stream Ciphers in GSM Communications [Electronic resource]/ Nagendar Yerukala, V Kamakshi Prasad, Allam Apparao // Journal of Communications Software and Systems (JCOMSS). – Access: https://www.researchgate.net/publication/338749303_Performance_and_Statistical_Analysis_of_Stream_ciphers_in_GSM_Communications.

ДОДАТОК А
ВИХІДНИЙ КОД ПРОГРАМНОЇ РЕАЛІЗАЦІЇ АЛГОРИТМУ A5/1

```
#include <stdio.h>
#include <locale.h>

//Ініціалізація 3 регістрів зсуву
#define R1INIT  0x07FFFF /* 19 біт, нумерація 0..18 */
#define R2INIT  0x3FFFFF /* 22 біт, нумерація 0..21 */
#define R3INIT  0x7FFFFF /* 23 біт, нумерація 0..22 */

 //Біти синхронізації для управління тактуванням
#define R1SYNC   0x000100 /* біт 8 */
#define R2SYNC   0x000400 /* біт 10 */
#define R3SYNC   0x000400 /* біт 10 */

 //Біти для тактування РЗЛЗЗ
 //* Визначаються такими многочленами
 //* x^19 + x^5 + x^2 + x + 1, x^22 + x + 1,
 //* x^23 + x^15 + x^2 + x + 1
#define R1TAPS  0x072000 /* біти 18,17,16,13 */
#define R2TAPS  0x300000 /* біти 21,20 */
#define R3TAPS  0x700080 /* біти 22,21,20,7 */

//Вихідні біти для генерації вихідних результатів
#define R1OUT   0x040000 /* біт 18 (старший біт) */
#define R2OUT   0x200000 /* біт 21 (старший біт) */
#define R3OUT   0x400000 /* біт 22 (старший біт) */
typedef unsigned char byte;
```

```
typedef unsigned long word;
typedef word bit;

//Обраховуємо парність 32 бітового слова, тобто обраховуємо суму його бітів за модулем 2
bit parnist(word x) {
	for (int i = 0; i < 31; i++)
	{
		x ^= x >> 1;
	}
	return x & 1;
}

//Синхронізуємо один регістр зсуву
word clockone(word reg, word init, word taps) {
	word t = reg & taps;
	reg = (reg << 1) & init;
	reg |= parnist(t);
	return reg;
}

//Оголосимо три регістри
word R1, R2, R3;

//Обчислимо мажоритарне значення 3 бітів синхронізації
bit major() {
	int sum;
	sum = parnist(R1&R1SYNC) + parnist(R2&R2SYNC) + parnist(R3&R3SYNC);
	if (sum >= 2)
```

```
            return 1;
        else
            return 0;
    }
```

//Синхронізуємо R1,R2,R3, відповідно до управління тактування бітами їх синхронізації

//Синхронізуємо регістри тоді коли біт синхронізації дорівнює мажоритарному значенню

```
    void sync() {
        bit maj = major();
        if (((R1&R1SYNC) != 0) == maj)
            R1 = clockone(R1, R1INIT, R1TAPS);
        if (((R2&R2SYNC) != 0) == maj)
            R2 = clockone(R2, R2INIT, R2TAPS);
        if (((R3&R3SYNC) != 0) == maj)
            R3 = clockone(R3, R3INIT, R3TAPS);
    }
```

//Синхронізуємо всі три регістри, ігноруючи біти синхронізації (використовується приініціалізації ключа)

```
    void clockall() {
        R1 = clockone(R1, R1INIT, R1TAPS);
        R2 = clockone(R2, R2INIT, R2TAPS);
        R3 = clockone(R3, R3INIT, R3TAPS);
    }
```

//Генеруємо вихідний біт, беручи вихідний біт з кожного регістру і додаємо їх за модулем 2

```
    bit getbit() {
```

```
        return parnist(R1&R1OUT) ^ parnist(R2&R2OUT) ^ parnist(R3&R3OUT);
    }

    //Робимо початкове налаштування A5/1 за допомогою ключа і 22-бітового номеру кадру
    void setkey(byte key[8], word frame) {
        int i;
        bit keybit, framebit;

        /* Спочатку всі три регістри заповнені нулями */
        R1 = R2 = R3 = 0;

        /* Загружаємо ключ в кожен регістр*/
         /*На кожному такті кожен біт ключа ^ з молодшим бітом регістру*/
        for (i = 0; i < 64; i++)
        {
            clockall(); /* завжди синхронізуємо і зсуваємо */
            keybit = (key[i / 8] >> (i & 7)) & 1; /* і-тий біт ключа */
            R1 ^= keybit; R2 ^= keybit; R3 ^= keybit;
        }

        /*Загружаємо номер кадру в кожний регістр*/
        /*На кожному такті, кожен біт кадру ^ з молодшим бітом кожного регістру*/
        for (i = 0; i < 22; i++)
        {
            clockall(); /* завжди синхронізуємо і зсуваємо */
            framebit = (frame >> i) & 1; /* і-тий біт кадру */
            R1 ^= framebit; R2 ^= framebit; R3 ^= framebit;
        }
```

```
        /* Запускаємо регістр зсуву для 100 тактів зміщення ключа і кадру*/
        /*При цьому вихідний генератор вимкнений, діє мажоритарне правило синхронізації*/
        for (i = 0; i < 100; i++)
        {
                sync();
        }

        /* Генератор налаштований */
}

//Генерація вихідної послідовності
//Генеруємо 228 біт послідовності
//114 біт від абонента до базової станції A->B
//114 біт від базової станції до абонента B->A
//Назначаємо 15 байтовий буфер для кожного напрямку
void a_to_b(byte AtoBkeystream[], byte BtoAkeystream[]) {
        int i;

        /* нулі для вихідного буферу */
        for (i = 0; i <= 113 / 8; i++)
        {
                AtoBkeystream[i] = BtoAkeystream[i] = 0;
        }

        //Генеруємо 114 біт в напрямку A->B
        for (i = 0; i < 114; i++)
        {
                sync();
```

```
            AtoBkeystream[i / 8] |= getbit() << (7 - (i & 7));
        }

        //Генеруємо 114 біт в напрямку B->A
        for (i = 0; i < 114; i++)
        {
            sync();
            BtoAkeystream[i / 8] |= getbit() << (7 - (i & 7));
        }
}

void cipher() {
        byte key1 = 0x25;
        byte key2 = 0x43;
        byte key3 = 0x15;
        byte key4 = 0xCD;
        byte key5 = 0x36;
        byte key6 = 0x11;
        byte key7 = 0xAA;
        byte key8 = 0xDF;
        byte key[8] = { key1, key2, key3, key4, key5, key6, key7, key8 };
        word frame = 0x143;
        byte AtoB[15], BtoA[15];
        int i, failed = 0;

        setkey(key, frame);
        a_to_b(AtoB, BtoA);

        printf("Ключ: 0x");
        for (i = 0; i < 8; i++)
```

```
        {
                printf("%02X", key[i]);
        }
        printf("\n");
        printf("Номер кадру: 0x%06X\n", (unsigned int)frame);
        printf("\n");
        printf("Отримані результати:\n");
        printf("        Abonent->Base station: 0x");
        for (i = 0; i < 15; i++)
        {
                printf("%02X", AtoB[i]);
        }
        printf("\n");
        printf("        Base station->Abonent: 0x");
        for (i = 0; i < 15; i++)
        {
                printf("%02X", BtoA[i]);
        }
}

int main(void) {
        setlocale(LC_ALL, "Rus");
        cipher();
        return 0;
}
```

ДОДАТОК Б
ОПУБЛІКОВАНА ПРАЦЯ ЗА ТЕМОЮ ДИПЛОМУ

Тези на V Міжнародну науково-практичну конференцію
«Проблеми кібербезпеки інформаційно-телекомунікаційних систем»
(PCSITS) 14 - 15 КВІТНЯ 2022, КИЇВ, Україна

«ДОСЛІДЖЕННЯ СИМЕТРИЧНИХ АЛГОРИТМІВ ШИФУРВАННЯ НА ПРИКЛАДІ ПОТОКОВОГО АЛГОРИТМУ А5»

Сучасний світ живе в епоху швидкого розвитку технологій безпровідного доступу, цьому сприяє прогрес в мікроелектроніці, що дозволяє випускати все більш складні і при цьому дешевші засоби безпровідного зв’язку. Мобільних телефонів по всьому світу стає значно більше, розвиток безпровідного зв’язку супроводжується неперервною зміною технологій в основі яких лежать стандарти стільникового зв’язку, а також стандарти передачі даних. Вони зокрема визначають рівень інформаційної безпеки, що вважається базовим і забезпечується комплексом програмно-апаратних засобів мережевого і абонентського обладнання.

GSM (Groupe Special Mobile, пізніше перейменований у Global System for Mobile Communications) – глобальний цифровий стандарт для мобільного стільникового зв’язку, що розроблений в кінці 1980-х рр., відноситься до систем другого покоління (2g) та підтримує чотири різні діапазони: 850,900,1800 та 1900 МГц. В мережі GSM поняття «безпека» означає не лише захист від несанкціонованого використання системи, а й забезпечення секретності переговорів абонентів. Багато вчених досліджували алгоритми захисту GSM, серед яких Oliver Damsgaard Jensen та Kristoffer Alvern Andersen, Thomas Stockinger та Jovan Dj. Golit.

В GSM стандарті з ціллю підвищення криптографічного захисту інформації використовується алгоритм аутентифікації, шифрування симетричним поточним шифром і алгоритм генерації ключів. Для виключення несанкціонованого використання ресурсів системи зв’язку вводяться і визначаються механізми

аутентифікації або перевірки достовірності абонента. Телефонна розмова в телекомунікаційній системі стандарту GSM передається в цифровій формі в вигляді послідовності кадрів. Кожний кадр, інформації, що передається, фактично шифрується своїм ключем шифрування, яке забезпечується поточним шифром. Основу системи безпеки GSM складають три алгоритми: А3 – алгоритм аутентифікації, А8 – алгоритм генерації криптоключа і А5 – алгоритм шифрування відцифрованої мови для забезпечення конфіденційності переговорів між абонентом і базовою станцією. Мобільні станції (телефони) оснащені смарт-картою, що містить алгоритми А3 та А8, а в самому телефоні наявний ASIC-чіп з алгоритмом А5. Базові станції також оснащені ASIC-чіпом з алгоритмом А5 і центром аутентифікації, що використовує алгоритми А3, А5, А8 для ідентифікації мобільного абоненту і генерації сеансового ключа. Для шифрування сигналу в GSM використовується алгоритми із сімейства А5, а саме А5/1 – сильна версія шифру та А5/2 – ослаблена версія.

А5 – це поточний алгоритм шифрування, що використовується для забезпечення конфіденційності даних, що передаються, між телефоном і базовою станцією в європейській системі мобільного цифрового зв’язку GSM. В цьому алгоритмі кожному символу відкритого тексту відповідає символ шифротексту. Текст не ділиться на блоки (як в блочному шифруванні) і не змінюється в розмірі. Для спрощення апаратної реалізації і, відповідно, збільшення швидкодії використовуються лише найпростіші операції: додавання за модулем 2 (XOR) та зсув регістру. Алгоритм А5 має чотири варіації: А5/0 (алгоритм в якому відсутнє шифрування), А5/1 (поточний шифр, що найбільш розповсюджений), А5/2 (аналог алгоритму А5/1, проте дешевший та має меншу криптостійкість, використовувався в країнах, що не входили до ЄС, в даний час не застосовується), А5/3 (блочний шифр, розроблений з метою удосконалення алгоритму А5/1 для третього покоління мобільних систем, також називається алгоритмом Касуми, має найвищий рівень криптостійкості). Певним недоліком алгоритму А5 можна вважати те, що рівень його криптостійкості був спрощений для доступу до інформації, що знаходиться в

захищеному каналі, розробниками алгоритму з ціллю навмисно послабити алгоритм доступу для спецслужб.

Отже, шифрування є невід’ємною частиною забезпечення конфіденційності, що підтверджує потоковий алгоритм шифрування A5,який використовується для забезпечення конфіденційності даних, що передаються, між телефоном і базовою станцією в європейській системі мобільного цифрового зв’язку GSM.