Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет __*Інфокомунікацій*_____
(повна назва)
Кафедра _____*Інформаційно-мережної інженерії*_____
(повна назва)

# КВАЛІФІКАЦІЙНА РОБОТА
## Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____

_____*Дослідження блокових шифрів для захисту даних в інфокомунікаційних мережах*_____
(тема)

Виконав:
студент _2_ курсу, групи __*ІМІм-22-2*___
_____*Шавлак А.В.*_____
(прізвище, ініціали)

Спеціальність __*172. Телекомунікації та радіотехніка*_____
(код і повна назва спеціальності)
Тип програми _____*освітньо-наукова*_____
(освітньо-професійна або освітньо-наукова)
Освітня програма _*Інформаційно-мережна інженерія*_____
( повна назва освітньої програми)

Керівник ____*ст.викл. Калюжний М.М.*_____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ _____*Безрук В.М.*_____
(підпис) (прізвище, ініціали)

2024 р.

Не містить відомостей, заборонених
до відкритого публікування

Керівник _____ /*М.М.Калюжний*

Студент _____ / *А.В. Шавлак*

Харківський національний університет радіоелектроніки

Факультет *Інфокомунікацій*

Кафедра *Інформаційно-мережної інженерії*

Рівень вищої освіти перший (бакалаврський)

Спеціальність *172. Телекомунікації та радіотехніка*
(код і повна назва)

Тип програми *Освітньо-професійна*
(освітньо-професійна або освітньо-наукова)

Освітня програма *Інформаційно-мережна інженерія*
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«_18_» *березня* 2024 р.

# ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові *Шавлаку Артему Володимировичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження блокових шифрів для захисту даних в інфокомунікаційних мережах*

затверджена наказом університету від 18 березня 2024 р. № 232 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 18 червня 2024 р.

3. Вихідні дані до роботи *Дослідити роль та специфіку використання криптографічних засобів для захисту даних в інфокомунікаційних мережах. Виконати аналіз особливостей побудови шифросистем та головні вимоги до них. Дослідити принципи та базові алхітектури симетричних блокових шифрів. За сукупністю показників ефективності серед найбільш поширених обрати такий, на базі якого, з низначними змінами у існуючу структуру, може бути побудовано нову шифросистему. Отримати програмну реалізації такої шифросистеми*

4. Перелік питань, що потрібно опрацювати в роботі *Вступ*

*1 . Місце та роль криптографічних систем у схемі захисту інформації*

*2. Огляд принципів реалізації та використання шифросистем*

*3. Дослідження шифросистеми DES*

*4. Побудова шифру на структурних засадах симетричної криптосистеми*

*Висновки*

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) _____
*слайди презентації в форматі Power Point (назва та мета роботи, місце та роль криптографічних систем у схемі захисту інформації, огляд принципів реалізації та використання шифросистем, дослідження шифросистеми DES, побудова шифру на структурних засадах блокового алгоритму , висновки)*

**КАЛЕНДАРНИЙ ПЛАН**

| № | Назва етапів роботи | Терміни виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Вступ | 20.03.2024 | виконано |
| 2 | Місце та роль криптографічних систем у схемі захисту інформації | 01.04.2024 | виконано |
| 3 | Огляд принципів реалізації та використання шифросистем | 10.04.2024 | виконано |
| 4 | Дослідження шифросистеми DES | 26.04.2023 | виконано |
| 5 | Побудова шифру на структурних засадах симетричної криптосистеми | 17.05.2024 | виконано |
| 6 | Висновки | 04.06.2024 | виконано |
| 7 | Оформлення пояснювальної записки | 07.06.2024 | виконано |

Дата видачі завдання *18 березня* 2024 р.

Студент _____
(підпис)

Керівник роботи _____ *ст. викл. Калюжний М.М.*
(підпис) (посада, прізвище, ініціали)

# РЕФЕРАТ

Пояснювальна записка: 60 с., 14 рис., 20 джерел, 4 додатки

TEA, AES, DES, Triple DES, MIPS, МЕРЕЖА ФЕЙСТЕЛЯ, РАУНДОВИЙ КЛЮЧ, ЛАВИННИЙ ЕФЕКТ, РАУНДОВА ФУНКЦІЯ, ДИФУЗІЯ

Об'єкт дослідження – блокові композитні симетричні шифри.

Мета роботи – дослідження можливості побудови шифросистеми з високою швидкістю шифротворення на засадах існуючої структури криптографічного алгоритму симетричного типу.

Розглядається специфіка побудови та застосування шифрів для захисту даних в інфокомунікаційних мережах. Обгрунтовується доцільність використання симетричних шифрів для обробки великих об’ємів даних. Аналізуються особливості архітектурних рішень з реалізації шифрів на базі S-P мережі та мережі Фейстеля. Досліджуються особливості побудови шифросистеми DES. На базі засад симетричних блокових систем виконується програмна реалізація шифру.

# THE ABSTRACT

Explanatory note: 60 p., 14 fig., 20 sources, 4 apps.

TEA, AES, DES, Triple DES, MIPS, FEISTEL NETWORK, ROUND KEY, AVALANCHE EFFECT, ROUND FUNCTION, DIFFUSION

The object of research is block composite symmetric ciphers.

The purpose of the work is to study the possibility of building a cipher system with a high rate of cipher creation on the basis of the existing structure of a symmetric type cryptographic algorithm.

The specifics of construction and application of ciphers for data protection in information communication networks are considered. The expediency of using symmetric ciphers for processing large volumes of data is substantiated. The peculiarities of architectural solutions for the implementation of ciphers based on the S-P network and the Feistel network are analyzed. Peculiarities of the construction of the DES cipher system are studied. Based on this, a software implementation of the specified cipher is created.

ЗМІСТ

С.

# ПЕРЕЛІК СКОРОЧЕНЬ

UTF-8 – (Unicode Transformation Format, 8-bit) – стандарт кодування символів Юнікоду;

ASCII – (American Standard Code For Information Interchange) – стандарт кодування символів;

IPS – (instructions per second) – кількість інструкцій за секунду;

VoIP – (Voice Over IP) – група сервісів, що надають послуги голосового зв’язку у IP-мережах;

VoD – (Video On Demand) – група сервісів, що надають послуги доставки потокового відео на запит користувача;

AoD – (Audio On Demand) — група сервісів, що надають послуги доставки потокового аудіо на запит користувача.

QoS – технологія контролю якості послуг, які надає інформаційно-комунікаційна мережа;

RSA – (Rivest, Shamir, Adleman) – асиметричний алгоритм шифрування;

AES – (Advanced Encrypting System) – симетричний алгоритм блочного шифрування;

DES – (Data Encrypting System) – симетричний алгоритм блочного шифрування;

XSL – (eXtended Sparse Linearization) – алгебраїчна атака, метод криптографічного аналізу.

# ВСТУП

Як свідчать статистичні дані [1-3], на поточний час соціум у загальносвітовому масштабі знаходиться у стадії поступового розширення масштабів інформаційно-мережної інфраструктури, збільшення кількості користувачів сервісів на базі мережі та кількості кінцевих терміналів.

Це, у свою чергу, викликано впливом певної сукупності факторів, серед яких:

- доробка (виявлення та усунення недоліків) існуючого технологічного базису;
- впровадження нових технологічних рішень, ККД яких перевищує передуючі;
- адаптація формату надання тих чи інших традиційних сервісів до вигляду, прийнятного для впровадження в інформаційно-комунікаційному просторі;
- здешевлення базових програмних та апаратних рішень в існуючому технологічному базисі, що пояснюється масовим характером їх впроваджень, ростом конкуренції у відповідних галузях та оптимізацією циклів виробництва.

Зазначена тенденція, у свою чергу, з одного боку створює умови для росту продуктивності бізнес-процесів фактично в усіх галузях діяльності, а з іншого боку – розширює можливості користувачів.

Разом з тим, у загальносвітовому масштабі тенденції до розширення інформаційно-мережної інфраструктури супутніми є:

- покращення функціональних та експлуатаційних характеристик інформаційних продуктів;
- ріст інтенсивності потоків даних у мережі;
- збільшення обчислювальної потужності вузлів.

Такі умови, на тлі розширення можливостей для користувачів сервісів, власників бізнесу та розробників, несуть у собі ймовірних ризиків, а саме – втрати або модифікації критичних даних, крадіжки даних з доступу/управління критичною інформаційною інфраструктурою.

Одними з головних чинників, що визначають існування ризиків, означених вище є наступні:

- більш широка потенційна доступність об’єктів інформаційної інфраструктури для зловмисників у наслідок її глибокої інтеграції у єдиний інформаційний простір [2];
- високі обчислювальні потужності, які є доступними для зловмисників;
- постійний розвиток методів зловмисного впливу.

За таких умов, у загальному випадку, для протидії інформаційним ризикам зазвичай застосовуються:

- превентивні дії, спрямовані на виявлення критичних ділянок систем захисту мережі з їх наступним усуненням;
- обмеження логічного доступу до тієї чи іншої інформації на базі політик безпеки та апаратно-програмного забезпечення, яке їх реалізує;
- обмеження фізичного доступу до тієї чи іншої інформації для осіб, що не мають відповідних привілеїв;
- захист даних на базі криптографічних перетворень.

У свою чергу, підхід на базі використання криптографічних перетворень заслуговує особливої уваги, так як [4]:

- являє собою відносно легкий з точки зору технічної реалізації захисту даних;
- може застосовуватися як самостійно, так і у вигляді додаткового засобу поряд з будь-яким з наведеними у переліку вище чи з усіма ними одночасно

Ключова ідея, закладена у захист інформації на базі криптографії полягає у тому, що за час, витрачений гіпотетичним зловмисником на злам шифрованих даних, навіть за умови доступу до значних обчислювальних потужностей, інформація застаріє а її актуальність та, власне, цінність, втратиться.

При цьому, беручи до уваги те, що реалізація криптосистеми на базі відомих алгоритмів не вимагає для реалізації значних витрат ресурсів, можна зробити висновок про ефективність та перспективність даного напрямку протидії інформаційним ризикам.

З іншого боку, сам факт того, що у широкому застосунку зараз знаходяться криптосистеми на базі найбільш поширених криптографічних алгоритмів, створює, у свою чергу, ряд додаткових ризиків, серед яких наступні:

- зловмиснику добре відомі архітектурні недоліки алгоритмів шифрування;
- розробники криптосистем, реалізуючи ті чи інші механізми, у протиріччі швидкодія/захищеність нерідко віддають перевагу швидкодії продукту, що створює потенційну небезпеку для шифрограм.

З вищесказаного виходить, що зараз усі питання, що стосуються дослідження, розробки та аналізу криптографічних алгоритмів є актуальними.

# 1. МІСЦЕ ТА РОЛЬ КРИПТОГРАФІЧНИХ СИСТЕМ У СХЕМІ ЗАХИСТУ ІНФОРМАЦІЇ

## 1.1 Загальна концепція захисту даних з обмеженим доступом

### 1.1.1 Класифікація даних за рівнем критичності щодо зловмисних впливів

Відповідно до чинної класифікації, з усього масиву даних, що зберігаються на носіях та надсилаються мережею, можна виділити [4]:

1. Дані, що мають гриф держтаємниці.
2. Конфіденційні дані.
3. Відкриті дані.

При цьому, захисту потребують перша та друга категорія з перелічених.

Так, гриф держтаємниці дає інформація, факт втрати чи модифікації якої створює загрозу на загальнодержавному чи регіональному рівнях і, у загальному випадку може нести у собі ризики відносно [4-6]:

- функціонування самого державного апарату та окремих його структур;
- фінансової безпеки держави та соціуму;
- стану довкілля;
- життя та здоров’я громадян та, у цілому, усіх осіб, що проживають на тій чи іншій території.

Такою інформацією володіють та оперують профільні служби та комітети, інші уповноважені органи державної влади. Захист даних цієї категорії забезпечується на найвищому рівні, при цьому застосовуються усі можливі механізми та засоби, перелік та регламент використання яких визначається та надалі впроваджується профільною комісією з інформаційної безпеки.

У свою чергу, поняття «конфіденційна інформація» поєднує у собі кілька окремих типів даних.

Втрата чи цілеспрямована модифікація таких даних є критичною для:

- окремої особи;

- групи осіб, поєднаних за ознакою географічної, професійної чи соціальної подібності;

- підприємства, установи чи галузі.

Зазвичай, до конфіденційних даних відносять [6]:

- особисту інформацію;
- комерційну таємницю;
- інформацію, що стосується функціонування об’єктів критичної інформаційної інфраструктури;
- дані щодо регламентів роботи галузі, підприємства чи його структурного підрозлілу, у разі втрати чи модифікації яких такий регламент буде порушено, що спричинить фінансові, репутаційні збитки та ін.

На відміну від інформації, яка відноситься до держтаємниці, захист даних, які вважаються конфіденційними, можуть здійснювати фахівці з інформаційної безпеки, що не є афільованими з профільними державними структурами.

У цілому тут, залежно від рівня потенційних ризиків, обсягу інформаціних ресурсів та побажання їх власника (власників), до захисту конфіденційних даних може бути залучено різних осіб (групи осіб). Це можуть бути:

- особи, що мають потрібний фах та необхідну сертифікацію відповідних державних органів;

- фахівці з інформаційної безпеки, які не мають сертифікації.

### 1.1.2 Ключові вимоги до захищеності даних

Незалежно від того, хто саме реалізує заходи з інформаційної безпеки, для будь-яких даних, що потребують захисту, має забезпечуватися [6]:

- конфіденційність;
- цілісність;
- доступність.

Тут властивість конфіденціності передбачає, що до інформації, яка захищається, мажуть мати виключно особи, які мають відповідні привілеї. При цьому, лише дана група осіб може виконувати дії з оновлення, модифікації, передачі, копіювання чи (за потреби) видалення такої інформації. Водночас, властивість цілісності гарантує, що інформація, до якої

отримують доступ авторизовані користувачі, не зазнала несанкціонованої модифікації чи псування, випадкового або цілеспрямованого.

Нарешті, властивість доступності вказує на те, що протягом усього часу, визначеного чинним регламентом, будь-яка уповноважена особа має змогу отримати доступ до інформації без будь-яких перешкод.

Далі розглянемо існуючі підходи до забезпечення захисту конфіденційних даних.

### 1.1.3 Рівні захисту даних

Як у випадку держтаємниці, так і для випадку конфіденційних даних, загальна схема захисту інформації містить у собі ряд ключових рівнів, а саме (рис. 1.1) [4, 6]:

- фізичний рівень;
- організаційний рівень;
- апаратно-програмний рівень.

У свою чергу, на фізичному рівні забезпечується блокування доступу неуповноваженим особам до:

- носіїв інформації;
- елементів мережевої інфраструктури;
- інструментарію управління;
- засобів забезпечення функціонування мережевої інфраструктури.

Зазвичай усе перелічене забезпечується за рахунок обмеження доступу до певної території чи приміщень для сторонніх осіб.

Разом з тим, на організаційному рівні визначається:

- ймовірні канали витоку інформації;
- уповноважені особи, що є відповідальними за існуючий регламент функціонування мережевої інфраструктури, захищеність даних та розслідування відповідних інцидентів;
- штатний та аварійний регламенти функціонування мережевої інфраструктури у цілому, а також його окремих складових;
- політики безпеки;
- ревні повноважень відповідальних осіб та рядових користувачів.

Приймаючи до уваги рішення, виконується застосування відповідних опцій для тих чи інших програмно-апаратних засобів, що забезпечують функціонал захисту даних.

При цьому, передбачається:

- створення (аудит) груп користувачів відповідно до необхідних привілеїв згідно з регламентом;
- створення «чорного» та «білого» списків внутрішніх вузлів, що мають або не мають повноваження виходу до зовнішньої мережі; у випадку його наявності – урахування можливих обмежень;
- створення «чорного» та «білого» списків вузлів зовнішньої мережі, для яких дозволено, або, навпаки, заборонено доступ до вузлів усередині мережі;
- імплементація засобів відстежування інцидентів, ведення статистики подій (у т.ч. як штатних, так і інцидентів), виявлення та блокування зловмисних впливів різного роду;
- вибір та застосування криптографічних засобів, вибір специфіки їх застосування;
- вибір та застосування апаратних засобів безпеки різних класів на рівні системи або на рівні окремих додатків (HASP, модулі довіреного завантаження тощо).

- створення «чорного» та «білого» списків внутрішніх вузлів, що мають або не мають повноваження виходу до зовнішньої мережі; у випадку його наявності – урахування можливих обмежень;
- створення «чорного» та «білого» списків вузлів зовнішньої мережі, для яких дозволено, або, навпаки, заборонено доступ до вузлів усередині мережі;
- імплементація засобів відстежування інцидентів, ведення статистики подій (у т.ч. як штатних, так і інцидентів), виявлення та блокування зловмисних впливів різного роду;
- вибір та застосування криптографічних засобів, вибір специфіки їх застосування;
- вибір та застосування апаратних засобів безпеки різних класів на рівні системи або на рівні окремих додатків (HASP, модулі довіреного завантаження тощо).

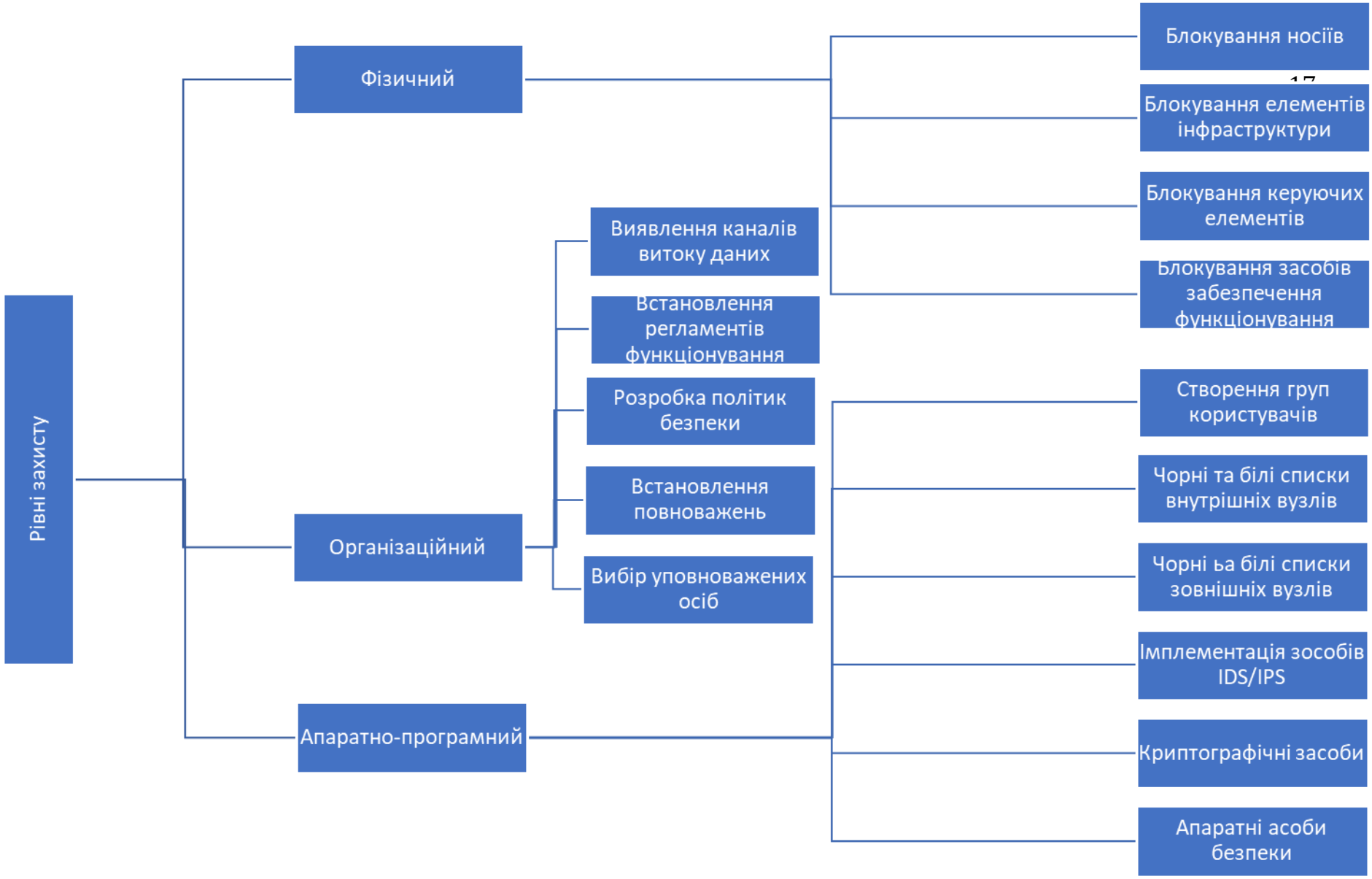


Рисунок 1.1 – Загальна схема захисту інформації

## 1.2 Специфіка використання криптографічних засобів

У свою чергу, у зазначеній схемі захисту даних, передбачено такі способи використання криптографічних засобів (рис.1.2) [7, 8]:

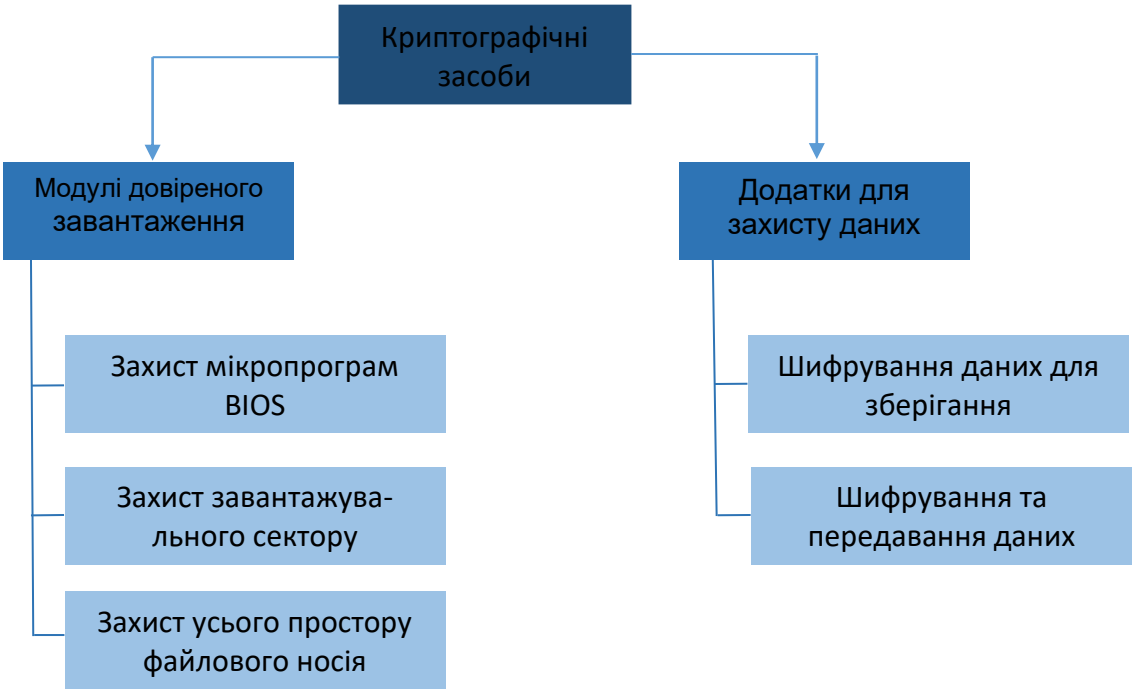


Рисунок 1.2 – Поширені сценарії використання криптографічних засобів

1. У складі модулів довіреного завантаження для захисту [7]:

- мікропрограм BIOS;
- завантажувального сектору;
- усього простору файлового носія.

2. Для захисту тих чи інших файлів, директорій чи розділів на базі окремих додатків.

На цей випадок додатково можна виокремити:

- додатки та утиліти, призначені для шифрування даних з їх подальшим зберіганням у файлосховищах, чи наступного передавання до віддалених вузлів;

- засоби, що поєднують у собі інструментарії комунікацій та шифрування; тобто, інформація, що підлягає передаванню, попередньо шифрується; при цьому, процедури шифрування та передачі являють собою послідовні кроки єдиного технологічного циклу.

При цьому, у рамках реалізації криптографічних засобів захисту даних можуть розглядатися такі, що в основі містять [4, 7]:

- асиметричні алгоритми шифрування;
- симетричні алгоритми шифрування;
- обидві вищеназвані архітектури.

Слід зазначити, що зараз додатки, які використовують виключно асиметричні алгоритми, мають обмежене поширення, як наслідок того, що:

- асиметричні алгоритми мають низьку продуктивність а відтак – можуть застосовуватися виключно для обробки невеликих обсягів даних (наприклад – ключів симетричного шифрування);
- на сьогодні існує велика кількість додатків, де забезпечується можливість використання як симетричних, так і асиметричних алгоритмів.

Водночас, досить широке поширення отримали додатки, що здатні одночасно надавати можливість використовувати як симетричні, так і асиметричні алгоритми шифрування.

Переважна більшість таких додатків надає користувачеві можливість вибору стосовно застосування тих чи інших підходів до побудови шифру та специфіки його застосування.

Зокрема, до таких додатків може бути віднесено засіб шифрування gpg, який набув широкого застосування у середовищі Unix [8].

Gpg являє собою утиліту, використання якої можливо у консольному режимі. При цьому, управління gpg може здійснюватися як з самої оболонки терміналу Linux/Unix, так і з Bash (Рис.1.3).

Незалежно від середовища застосування, Gpg надає можливість шифрування даних у таких режимах як:

- шифрування окремих файлів;
- шифрування директорій.

Також, незалежно від обраного режиму, шифрування може виконуватися на базі:

- алгоритмів симетричного шифрування;
- алгоритмів асиметричного шифрування;
- алгоритмів створення та інтеграції цифрового підпису.

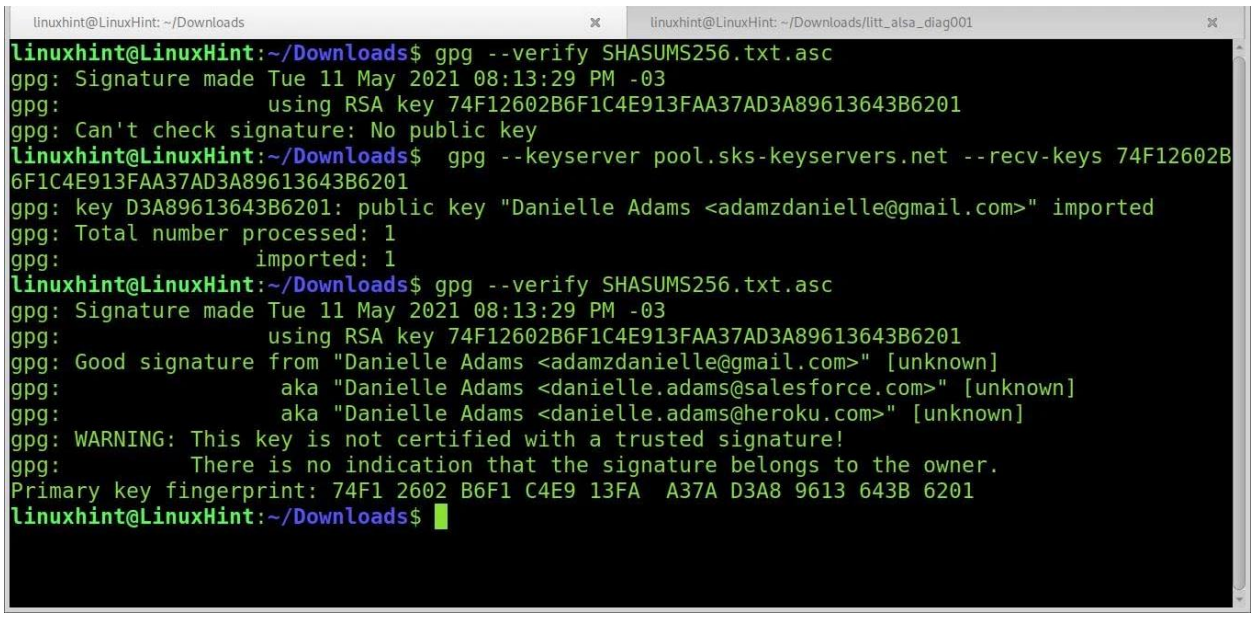


Рисунок 1.3 – Робочий діалог утиліти GPG

Окрім цього, передбачається можливість реалізації усього циклу шифротворення з використанням:

- парольної фрази;
- ключів шифрування.

При цьому, утиліта надає користувачеві здійснювати вибір між такими алгоритмами шифрування, як:

- RSA;
- DES;
- Blowfish;
- EL-GAMAL;
- Triple DES;
- AES та ін.

Окрім самого шифрування, передбачається також можливість функціонування у режимі цифрового підпису.

У свою чергу, дані, які було зашифровано на базі використанням утиліти GPG, надалі може бути або передано до іншого вузла відкритими каналами, або використано для зберігання на локальному носієві у зашифрованому вигляді.

Разом з тим, одним з прикладів додатків, які здатні одночасно реалізувати у собі функції як шифрування, так і надсилання даних, може вважатися Telegram. Для цього додатку передбачено можливість

використання додаткових механізмів шифрування окрім тих, які за замовчуванням передбачено системою.

Наприклад, таким механізмом є такий, що використовує хмарний пароль поряд з тими, що надаються системою (рис. 1.4) [9].

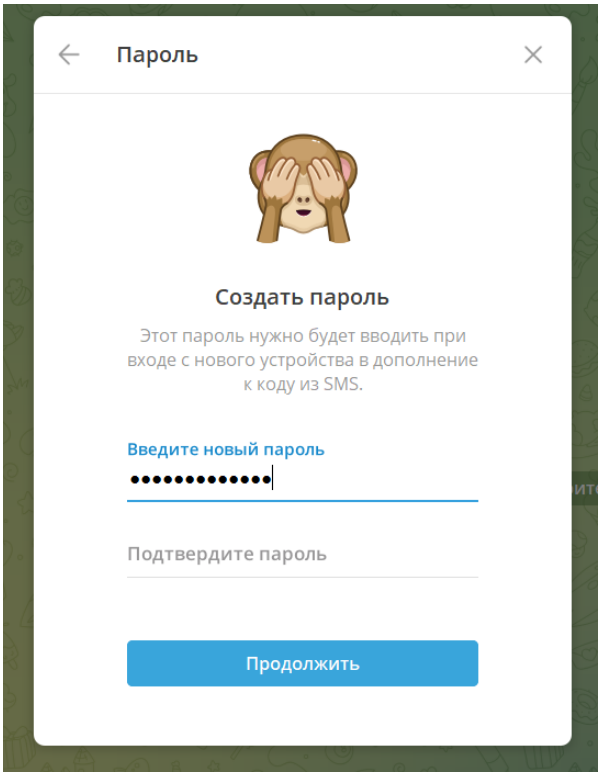


Рисунок 1.4 – Діалогове вікно встановлення додаткового хмарного паролю

Разом з тим, не дивлячись на формат наявних криптографічних засобів, необхідно взяти до уваги наступні закономірності [4, 10]:

- шифрування невеликого обсягу даних, або даних, які далі будуть зберігатися локально, може бути реалізовано на базі будь-якого з існуючих алгоритмів;
- у випадку, коли мова йде про шифрування з наступною передачею у мережу шифрованих даних, а тим паче – про шифрування і передачу у реальному часі – для виконання криптографічних перетворень безпосередньо самих даних використовуються алгоритми симетричного шифрування; для шифрування секретного ключа, у свою чергу, використовуються асиметричні криптосистеми.

У той же час, можливість використання виключно асиметричних алгоритмів у ході забезпечення безпечної передачі даних можлива у таких випадках:

- коли об’єм даних, що підлягають обробці, є відносно невеликим (порядку сотень кілобайт – одиниць мегабайт);
- наявні обчислювальні можливості забезпечують надзвичайно високу швидкість шифротворення;
- коли шифрування даних, що підлягають обробці, не має виконуватися у реальному часі;
- коли інформаційний потік шифрується не повністю, натомість шифруванню підлягають службові дані, які забезпечують його реконструкцію до вигляду, який може сприйматися на рівні приймача.

У будь-якому випадку, незалежно від типу шифра та режиму його використання, процедура шифрування виконується на рівні джерела відповідно до схеми на рис.1.5 [7].

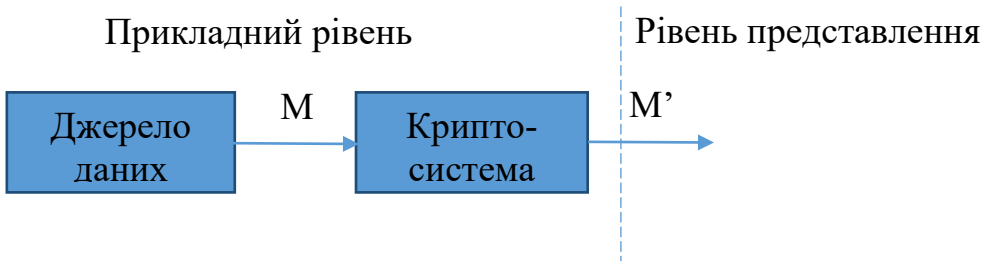


Рисунок 1.5 – Місце застосування криптографічних засобів у ході обробки даних

Як бачимо з рисунку 1.5, шифрування даних здійснюється на прикладному рівні моделі OSI після того, як повідомлення M формується джерелом.

Відповідно, на представницькому рівні далі виконується обробка вже шифрованих M’ даних.

# 2. ОГЛЯД ПРИНЦИПІВ РЕАЛІЗАЦІЇ ТА ВИКОРИСТАННЯ ШИФРОСИСТЕМ

Шифросистеми, які використовуються зараз, а також шифри класичних алгоритмів, мають загальні риси, які визначаються ключової архітектури.

## 2.1 Ключові архітектури шифрів

У загальному випадку, не дивлячись на велику кількість існуючих сьогодні шифросистем, можемо виділити дві головні архітектури шифрів, а саме [7, 10]:

- симетрична;
- асиметрична.

Для випадку шифрів симетричної архітектури загальна процедура криптографічного перетворення може бути продемонстрована наступним прикладом.

Нехай існує деяке вихідне повідомлення $M$ довільного змісту та довжини.

На базі даного повідомлення отримується шифр $M'$ відповідно до наступного принципу:

$$M' = \phi(M; K), \tag{2.1}$$

де $\phi$ - деяке функціональне перетворення (алгоритм шифрування), механізм якого частіше усього є загальновідомим;

$K$ - криптографічний ключ, який встановлює параметри виконання перетворення $\phi$; ключ не розголошується.

У свою чергу, зворотнє перетворення, або дешифрування, буде наступним:

$$M = \phi(M'; K). \tag{2.2}$$

Таким чином, у випадку симетричного шифру відновлення повідомлення $M$ виконується за алгоритмом $\phi$ з використанням того ж самого ключа $K$.

Розглянемо тепер приклад шифру асиметричної архітектури. У цьому разі аналітичний вираз для опису процедури шифрування може бути подано у наступному вигляді:

$$M' = \phi(M; K_1), \tag{2.3}$$

де $K_1$ - відкритий ключ.

При цьому, дешифрування виконується наступним чином:

$$M = \varphi(M'; K_2), \tag{2.4}$$

де $K_2$ - секретний ключ.

$\varphi$ - функціонал дешифрування.

Необхідно зазначити, що для випадку асиметричної криптосистеми, виходячи з особливостей її архітектури, для забезпечення стійкості утворених шифрів у загальному випадку має забезпечуватися виконання наступних умов [10, 11]:

$$\begin{cases} \varphi \neq \phi; \\ K_2 \neq K_1. \end{cases} \tag{2.5}$$

Тобто, усі функціональні перетворення – як шифрування, так і дешифрування, зазвичай виконуються за різними алгоритмами. При цьому, використовується різні ключі.

Дана архітектурна особливість забезпечує помітну перевагу асиметричних систем перед симетричними шифрами стосовно можливості спрощеного безпечного обміну ключами між передаючою та приймаючою сторонами.

Сутність процедури безпечного обміну ключами та наступного шифрування демонструється діаграмою на рисунку 2.1.

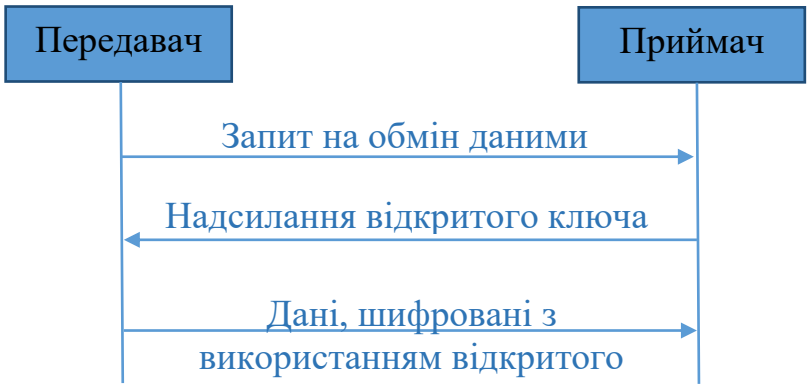


Рисунок 2.1 – Процедура надсилання відкритого ключа з наступним шифруванням в асиметричній схемі

Так, на першому кроці передавач ініціює запит на обмін даними.

У свою чергу, приймач, отримавши такий запит, надсилає передавачеві згенерований відкритий ключ $К_1$ [4, 6, 7].

Далі, отримавши такий ключ, передавач використовує його для шифрування повідомлення $М$ відповідно до виразу (2.3) та надсилає утворену шифрограму $М'$ відкритим каналом.

Приймач, одержавши шифрограму $М'$, реконструює повідомлення $М$ на базі принципу, поданого виразом (2.4).

Для цього використовується секретний ключ $К_2$, який генерується синхронно з відкритим.

Таким чином, відкритий ключ $К_1$ може вільно надсилатися незахищеним середовищем.

У загальному випадку, у разі перехоплення такого ключа зловмисником, його не може бути використано для відкриття шифрограми.

## 2.2 Різновиди блокових шифрів симетричної архітектури

На сьогодні майже усі існуючі симетричні шифри може бути віднесено або до блокового, або потокового типу [7, 12].

Водночас, у рамках блокових симетричних шифрів можна окремо виділити два головних їхніх різновиди – а саме – мережу Фейстеля та S-P мережу.

2.2.1 Мережа Фейстеля

Мережею Фейстеля являє собою спеціалізований механізм каскаду перетворень початкової послідовності Р біт повідомлення М. У ході таких перетворень здійснюється накладення значень, які було розраховано з однієї частини масиву, на інші його частини [12].

При цьому, структурна реалізація мережі Фейстеля передбачає можливість застосування єдиного алгоритма для здійснення як операцій шифрування, так і дешифрування.

У свою чергу, ключова відмінність між двома зазначеними процедурами полягає лише у різному порядку використання секретного ключа К.

Схему класичної мережі Фейстеля зображено на рисунку 2.2.

Передбачається, що у рамках мережі Фейстеля виконається обробка породжених початковим блоком $P_i$ незалежних між собою потоків даних. Такі потоки даних звуться гілками даних.

У той же час, параметрами мережі є величини $V(i)$. Такі параметри – похідні від матеріалу початкового секретного ключа.

Виходячи з того, як само у межах шифру було реалізовано механізм формування раундових ключів, функціонал алгоритму може передбачати використання як параметри $V(i)$, так і напряму сукупність раундових ключів, тобто, $\overline{К} = \{К(i)\}$.

У даному разі, на той випадок, коли опис алгоритму не є цілком формалізованим, тобто, наявна специфікація дозволяє варіювати варіантами реалізації деяких його технологічних етапів, може бути використано як параметри $V(i)$, так і ключі $К(i)$.

У ході перетворень даних у межах кожного раунду бере участь утворююча функція $Ф$.

Разом з тим, раундом, чи циклом мережі є технологічний етап, який містить у собі процедуру одноразового розрахунку величини $Ф$ утворюючої функції, та процедуру наступного накладання одержаного результату на іншу гілку.

При цьому виконується обмін гілок місцями.

Мережа Фейстеля, побудована за класичною схемою, на вхід приймає блоки вихідного повідомлення М та ключ К. Довжина кожного блоку при цьому складає 64 біта [7, 10].

Після того, як деякий блок $P_i$ відкритого повідомлення зчитується системою, здійснюється його поділ на дві частини однакової довжини з утворенням блоків $P_i^{(1)}$ і $P_i^{(2)}$.

Далі відносно таких блоків послідовно застосовується деяка кількість раундів шифрування, за результатом чого блоки об'єднуються, тим самим формуючи шифрований блок даних довжиною 64 біта.

У свою чергу, як для усього алгоритму, так і кожного раунду шифрування, початковими даними для шифротворення є:

- блоки половинного поділу $P_i^{(1)}$ і $P_i^{(2)}$, що формуються у підсумку усіх виконаних перетворень зі складу передуючого раунду;

- параметр $V_i$, який знаходиться за певним алгоритмом шляхом обробки початкового секретного ключа К; важливою умовою для того, щоб створити умови для росту криптостійкості утворюваного шифру, є дотримання зазначених далі умов, а саме:

$$V_1 \neq V_2 \neq ... \neq V_i \neq ... \neq V_r, \qquad (2.6)$$

іншими словами, має виконуватися умова нерівності усіх раундових ключів між собою, а також нерівність будь-якого з них з ключем К.

При цьому, не зважаючи на те, яку саме кількість раундів шифрування передбачено у складі того чи іншого шифру на базі мережі Фейстеля, усі раунди виконуються на базі однакової схеми.

Так, спершу права частина $P_i^{(2)}$ блоку даних, які шифруються, підлягає процедурі підстановки.

У ході означеної процедури до лівої частини блоку, тобто, ($P_i^{(1)}$), застосовується деяка раундова функція, після чого отриманий результат складається з правою частиною блоку, для цього застосовується операція XOR.

Слід зазначити, що раундова функція є незмінною для усіх раундів, тобто, передбачає виконання аналогічних перетворень даних у ході повного циклю шифрування.

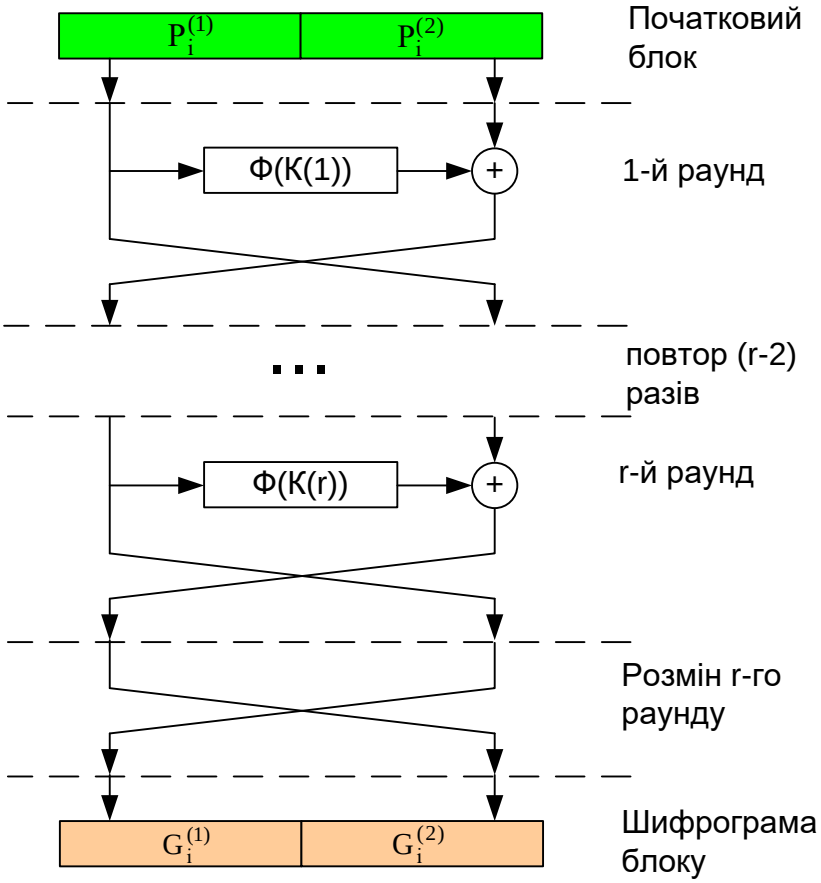


Рисунок 2.2 – Структура класичної мережі Фейстеля

Водночас, раундова функція є залежною від наявного параметру V(i), або раундового ключа K(i). Далі, після виконання підстановки, здійснюється процедура перестановка, завданням якої є обмін місцями лівої та правої частин блоку даних [12, 13].

Разом з тим, технологічний процес зворотнього перетворення даних, тобто, їх дешифрування, у рамках криптосистем, що були утворені на базі мережі Фейстеля, не має принципових відмінностей від процесу шифротворення.

Так, для реконструкції повідомлення M з шифру M′ застосовується аналогічний шифруванню алгоритм, проте на його вхід надсилаються шифровані дані.

У свою чергу, при цьому раундові ключі K(i), або параметри V(i), мають зворотню послідовність використання.

Зокрема, у ході першого раунду шифрування використовується параметр $V(r)$, для другого - $V(r-1)$ і т.д., до використання параметру $V(1)$ у ході останнього раунду.

Зазначена властивість розглянутої схеми шифрування відзначається зручністю. Це зумовлюється тим, що процес дешифрування раніше згенерованого шифру не потребує окремого алгоритму зворотної дії.

Виконаємо далі аналіз особливостей іншого різновиду блокових симетричних шифрів, а саме - S-P мережі.

### 2.2.2 S-P мережа

S-P мережа (Substitution-Permutation network), як одна з існуючих структурних основ побудови симетричного блокового шифру, містить у собі функціональні блоки двох типів, що реалізують підстановочні та перестановні операції, які багаторазово виконуються у ході шифротворення з певною черговістю [7, 14, 15]. Поєднання таких блоків утворюють своєрідні прошарки (рис.2.3).

Тут перший тип блоків - P-блоки (permutation, або перестановний блок). Лише один такий блок великої розрядності повністю формує P-прошарок.

У свою чергу, P-прошарку відповідає сукупність багатьох S-блоків низької розрядності, утворюючи S-прошарок.

При цьому, S-P мережа може містити у собі велику кількість прошарків S та P типу, що чергуються між собою.

Прикладами сучасних шифрів, що утворені базі S-P мережі, є, зокрема, Serpent, Anubis, Rijndael (AES) [7].

Зараз для реалізації S та P-блоків частіше за все застосовуються логічні або арифметичні операції та їх поєднання. Зокрема, будь-яка бінарна операція може стати основою S-блоку, тоді як деякі з них - основою P-блоку.

Перевагою такого підходу є легкість реалізації S та P функцій апаратно, що, у свою чергу, забезпечує високу швидкість усіх необхідних перетворень та високий рівень криптостійкості.

На початку роботи, отримавши вихідний блок повідомлення M та ключ K, шифр на базі SP-мережі виконує деяку кількість послідовних раундів шифрування. При цьому, кожен раунд містить у собі стадію підстановки (substitution stage), та стадію перестановки (permutation stage), які реалізуються, відповідно, S та P блоками [12].

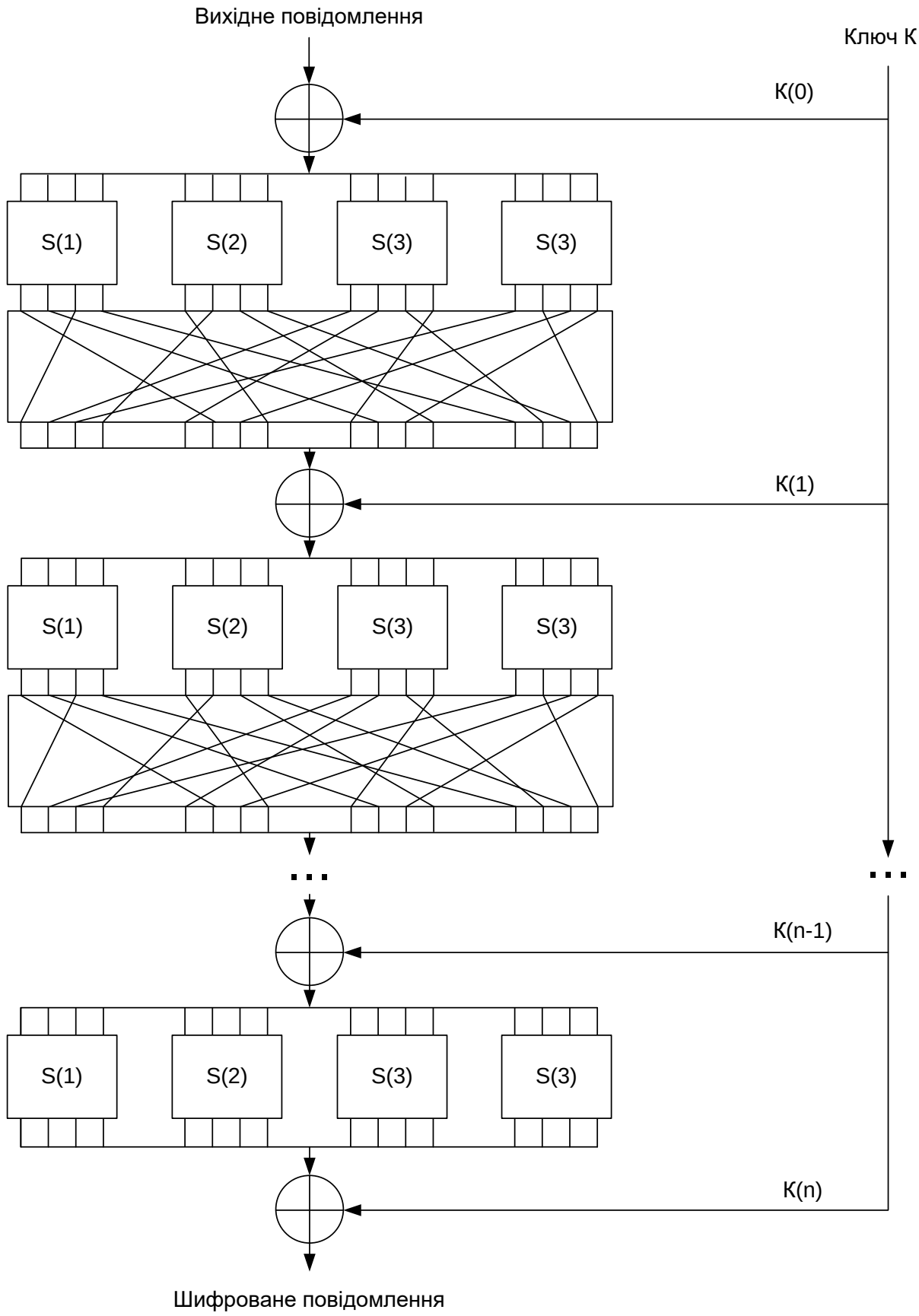


Рисунок 2.3 – Структурна схема S-P мережі

Архітектура S-P мережі передбачає застосування у складі S-прошарку великої кількості S-блоків невеликої розрядності. Це, у свою чергу, зумовлюється тим, що великий S-блок у ході шифрування вимагає значного обсягу пам'яті.

Разом з тим, на стадії перестановки виконується операція змішування біт ключа K з бітами надходячого повідомлення (це може бути або повідомлення безпосередньо від джерела, або частково шифровані дані з передуючого раунду обробки).

У свою чергу, таке змішування виконується за нелінійним законом. За рахунок цього дана процедура забезпечує умови виникнення властивості конфузії для шифрограми, тим самим збільшуючи потенційну криптостійкість.

З іншого боку, процедури перестановки являють собою лінійні перетворення. При цьому, у ході P-операцій виконується розподіл надмірність у межах усієї структури даних, за рахунок чого досягається властивість дифузії шифру [3][4].

Власне, сам S-блок виконує заміну невеликого вхідного блоку даних іншим блоком такого ж розміру.

У сутності, S-блок виконує операції нелінійного перетворення, за рахунок чого фактично виключається можливість виконання лінійного криптоаналізу.

Тут буде важливим зазначити, що операція заміни, яка реалізується на базі S-блоку, повинна бути зворотною, щоб гарантувати можливість розшифрування.

Також важливою властивістю S-блоку є можливість забезпечення т.з. «лавинного ефекту». Даний ефект проявлюється у тому, що зміна навіть одного біта вхідного повідомлення може вести до зміни усіх біт утвореної шифрограми [7, 12, 15].

Нарешті, блок перестановки, або P-блок, приймає на вхід результат перетворень S-блоків, виконує заміну позицій усіх прийнятих біт, після чого спрямовує одержаний таким чином результат на вхід S-блоку наступного раунду.

При цьому, P-блоку забезпечує можливість розподілу даних, що надійшли з одного S-блоку поточного $i-го$ раунду між великою кількістю S-блоків $(i+1)-го$ раунду.

Водночас, як і у випадку мережі Фейстеля, протягом кожного раунду шифрування застосовується окремий раундовий ключ $K(i)$. Формування кожного раундового ключа також виконується на базі початкового секретного ключа K.

## 2.3 Головні параметри, ключові переваги та недоліки шифрів різних архітектур

У цілому, результативність роботи будь-якої шифросистеми незалежно від архітектури, як зазначалося виразами (2.1)-(2.4), залежить від використовуваного алгоритму $\phi$ шифрування, та параметрів застосованого секретного ключа K.

При цьому, до головних характеристик будь-якого шифру, що у достатній мірі може характеризувати його ефективність, відносяться наступні [12]:

- криптографічна стійкість;
- пропускна здатність алгоритму (інтенсивність шифротворення);
- алгоритмічна складність.

### 2.3.1 Криптографічна стійкість алгоритму

Даний параметр у загальному випадку може розглядатися у двох трактуваннях, а саме:

- як час, що витратися зловмисником для того, щоб з шифрограми $M'$ відновити початкове повідомлення $M$;
- як обсяг обчислень, які має бути виконано для реконструювання повідомлення $M$ з шифру $M'$.

Якщо розглядати криптографічну стійкість з позиці необхідної кількості обчислень, тоді її величину буде вимірюватися у кількості MIPS, де $1\,\text{MIPS} = 10^6$ IPS. У свою чергу, це є показником кількості інструкцій за секунду (instructions per second).

Щоб приблизно оцінити ймовірність зламу ряду поширених шифросистем, спочатку переглянемо відповідні значення параметру IPS, як показано у таблиці 2.1 [12].

Таблиця 2.1 – Величина криптографічної стійкості деяких шифросистем, подана у вигляді параметру MIPS

| Алгоритм | Довжина ключа, біт | Криптографічна стійкість, MIPS |
|---|---|---|
| DES | 56 | $7\times10^{16}$ MIPS |
| Triple DES | 168 | $\approx10^{34}$ MIPS |
| DESX | 184 | $\approx10^{38}$ MIPS |
| IDEA | 128 | $\approx10^{33}$ MIPS |
| TEA | 128 | $9{,}6\times10^{11}$ MIPS |
| Rijndael (AES) | 128, 192, 256 | На поточний момент стійкий до класичних методів криптоаналізу |

У свою чергу, швидкодія сучасних процесорів знаходиться у діапазоні, що незначно перевищує 10000 MIPS.

Наприклад, процесору з ARM-архітектурою Cortex-A9 (Dual core) відповідає швидкодія на рівні 7500 MIPS при 1,5 GHz на ядро.

Водночас, для іншого процесору, а саме - PS3 Cell BE, може бути забезпечено швидкодію порядку 10240 MIPS при тактовій частоті ядер 3,2 GHz.

Тобто, як свідчить аналіз наведених даних, існуючий технологічний базис не може створити умови для ефективного зламу перелічених шифросистем.

Наприклад, злам шифросистеми AES, виходячи з існуючих обчислювальних можливостей, може потребувати близько 7900 років у випадку, коли для зламу будуть використовуватися класичні методи криптоаналізу.

Разом з тим, як свідчить ряд досліджень, [14, 15] алгоритм не є потенційно стійким до зламу за методами:

- атака за часом. Тут ураховується те, що різні операції потребують різного часу для реалізації;
- XSL-атака (eXtended Sparse Linearization), або алгебраїчна атака, можливість реалізації якої зумовлюється простою алгебраїчною структурою шифру [15];
- атака за сторонніми каналами. Так, на базі методу DFA (Differential Fault Analysis) секретний ключ було повністю відновлено за $2^{32}$ операції [16, 17].

2.3.2 Пропускна здатність алгоритму

Даний параметр визначає діапазон та особливості можливого застосування шифросистеми.

Так, беручи до уваги даний параметр, зараз усі існуючі шифросистеми, не залежно від архітектури та особливостей реалызації, у загальному випадку може бути поділено на:

- системи реального часу;
- системи з затримкою.

При, цьому, у рамках першого типу шифросистем може бути виокремлено:

- системи для обробки інтерактивного трафіку;
- системи, які застосовуються для шифрування трафіку реального часу, який не потребує забезпечення гарантованого рівня пропускної здатності.

Належність того чи іншого алгоритму до певної категорії може бути визначено на базі даних, наведених табл. 2.2 [12, 13].

Таблиця 2.2 – Рівень пропускної здатності деяких поширених шифросистем

| Алгоритм | Пропускна здатність, Мбіт/сек |
|---|---|
| DES | 1161,31 |
| Triple DES | 407,40 |
| DESX | 776,12 |
| IDEA | 225,55 |
| TEA | 394,08 |
| Rijndael (AES) | 1950,03 |

Аналіз таблиці 2.2 показує, що:

- алгоритми DES та Rijndael є найбільш продуктивними з позиції швидкодії; відповідно, на їх базі може бути реалізовано шифрування у реальному часі деяких типів трафіку «важкого» типу – відеоряду, даних інтерактивного та критичного типів тощо;
- шифри IDEA та TEA може бути застосовано для офлайн-обробки даних.

2.3.3 Алгоритмічна складність

Даний показник визначає можливість реалізації алгоритму шифрування на базі апаратних систем з невисокими обчислювальними можливостями. У загальному випадку вважається, що алгоритмічна складність шифросистеми у цілому корелюється з її пропускною здатністю.

Тобто, знаючи пропускну здатність алгоритму, можна надати якісну оцінку його складності.

Для оцінки складності алгоритму береться до уваги кількість логічних операцій, які мають виконатися для завершення одного повного циклу шифрування.

Виходячи з цього, найбільш раціональними з посеред вищеперелічених можуть вважатися шифри, знову ж таки, DES та Rijndael.

При цьому, у рамках DES-шифрування найбільш складною операцією є складання за модулем 2 (XOR), яка виконується відносно блоку довжиною 64 біта.

У свою чергу, серед усіх операцій, які містить шифр Rijndael у своєму складі, найбільш складною є обчислення поліномів у полі Галуа.

Отже, підводячи підсумок за результатами розгляду шифрів різних архітектур, попередньо можемо зазначити наступне:

- з точки зору швидкодії (пропускної здатності) алгоритму, найбільш ефективними серед розглянутих алгоритмів шифрування є DES та Rijndael (AES);
- з позиції алгоритмічної складності найбільш перспективними також можна вважати алгоритми DES та Rijndael;
- за показником криптографічної стійкості найбільш ефективними є алгоритми Rijndael, Triple DES та DESX.

Таким чином, за більшістю головних показників шифросистеми, найбільш ефективною з розглянутих є Rijndael (AES). Разом з тим, дана шифросистема є повністю формалізованою.

Тобто, зміна будь-якої частини структури шифру може привести або взагалі до нівелювання його ефективності, або до необхідності вносити відповідні зміни в усі модулі шифросистеми, що являє собою складне комплексне завдання.

Отже, беручи до уваги вимоги технічного зававння – а саме – можливість внесення змін до базового алгоритму, який характеризується високими показниками за рядом головних параметрів – теоретично

придатним для модифікації попередньо можемо вважати DES (зокрема, тому, що DESX та Triple DES мають в основі класичний DES).

Для підтвердження чи спростування означених висновків далі виконаємо більш глибоке дослідження даної шифросистеми.

# 3. ДОСЛІДЖЕННЯ ШИФРОСИСТЕМИ DES

## 3.1 Загальні відомості про шифросистему

Data Encryption Standard, або DES, являє собою блокову композитну шифросистему симетричної архітектури [7, 16].

Сьогодні DES характеризується досить широким застосуванням, та, фактично, є лідером з поширеності серед додатків, які реалізують функції симетричного шифрування.

У свою чергу, чинниками, що сприяли розповсюдженості даної шифросистеми, сьогодні є такі, як перелічено далі:

- використання єдиного ключа розмірністю 56 біт;
- можливість використання для реконструкції шифрованих повідомлень будь-якого іншого додатку, що підтримує алгоритм DES;
- той факт, що відносна простота алгоритму створює умови для забезпечення високої швидкості обробки даних;
- DES-шифрування забезпечує досить високий рівень криптостійкості;
- відновлення даних, шифрованих на базі DES, є процедурою, зворотньою процедурі шифрування; така процедура реалізується за рахунок виконання повторів операції шифрування у зворотньому порядку (попри очевидність зазначених дій, перетворення таким чином виконуються не завжди).

Свого часу, DES був рекомендований у якості засобу для шифрування конфіденційної інформації (окрім тієї, що відносилася до категорії державної таємниці) у державних закладах США.

Оскільки алгоритм DES не передбачає виконання будь-яких складних та ресурсоємних обчислень, за рахунок цього забезпечується однаково ефективне шифрування для систем з різними рівнями обчислювального потенціалу апаратної частини.

Архітектурний базис даної шифросистеми складає мережа Фейстеля.

## 3.2 Дослідження алгоритмічної реалізації DES

У сутності, загальна процедура шифротворення зводиться до виконання початкової перестановки біт 64-бітового блоку, реалізації 16 циклів шифрування та, врешті решт, реалізації обраної перестановки біт, як показує рисунок 3.1.

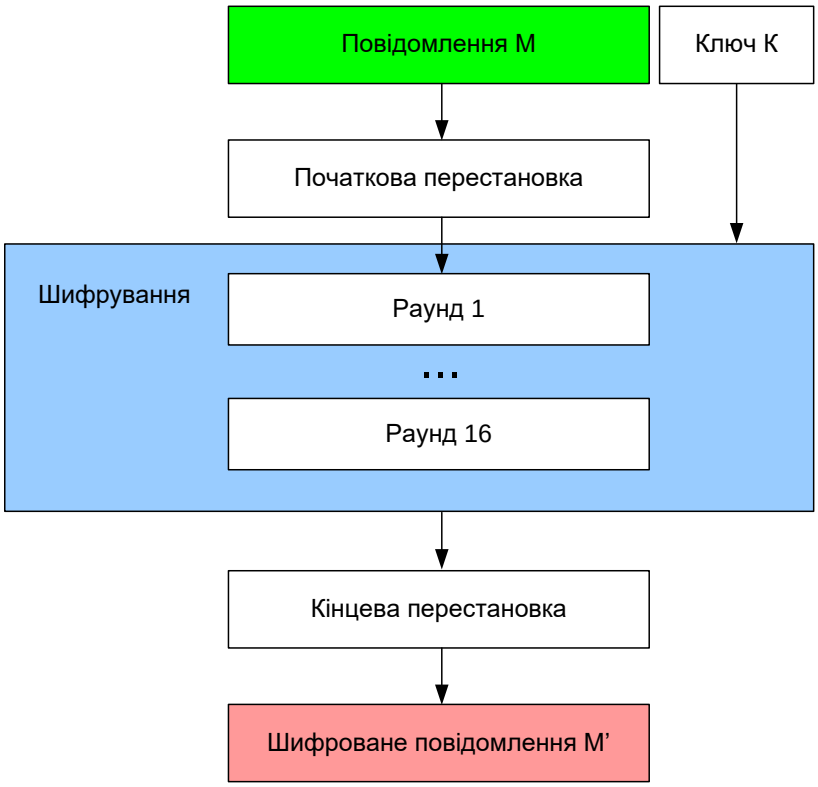


Рисунок 3.1 – Узагальнена схема шифрування, передбачена алгоритмом DES

У ході побудови криптограми DES використовує ряд таблиць шифрування.

При цьому, усі таблиці шифрування, що використовуються, є стандартизованими. Звідси виходить, що зазначені таблиці мають бути включеними у будь-яку реалізацію методу незмінно.

Водночас, усі можливі перестановки, а також коди у таблицях шифрування розробниками підібрано таким чином, щоб зробити процес дешифрування за рахунок підбору ключа максимально важким та ресурсоємким.

Більш детальну структуру алгоритму DES приведено на рис.3.2.

Згідно алгоритму, обробка виконується на рівні блоків по 8 байт.

При цьому, кожен черговий 64-бітовий блок T, який зчитується з повідомлення M, попередньо трансформується з використанням матриці початкової перестановки, або IP-матриці.

У результаті застосування IP-матриці встановленим чином, змінюється порядок початкових біт, що у спрощеному вигляді може бути подано як $T(0) = IP(T)$.

Далі, відносно одержаної за результатом виконання етапу початкової перестановки послідовності $T(0)$ виконується процедура поділу навпіл, з формуванням двох нових послідовностей довжиною 32 біта кожна: $L(0)$ – утворена лівими, або старшими бітами, та $R(0)$, яку формують праві (молодші) біти.

Далі виконується послідовність з 16 раундів шифрування. Водночас, перетворення, які здійснюються у ході кожного з них, може бути описано виразами [7, 12, 16]:

$$L(i) = R(i-1) \tag{3.1}$$

$$R(i) = L(i-1) \text{ xor } f(R(i-1), K(i)), \tag{3.2}$$

де $R(i-1)$ – послідовність довжиною 32-біта, утворена після виконання $(i-1)$-го раунду шифрування;

$f$ – шифруюча функція раунду;

$K(i)$ - раундовий ключ шифрування (i-го раунду), який має розмір 48 біт, утворений на базі з 64-бітового секретного ключа $K$.

У свою чергу, після виконання 16-го раунду шифрування утворюються послідовності $R(16)$ та $L(16)$ (без виконання процедури перестановки), які далі конкатенують між собою, формуючи єдину послідовність $R(16)L(16)$, що має довжину 64 біта.

Наступним кроком обробки є перестановка позицій біт послідовності $R(16)L(16)$ відповідно до матриці $IP^{-1}$ зворотніх перестановок.

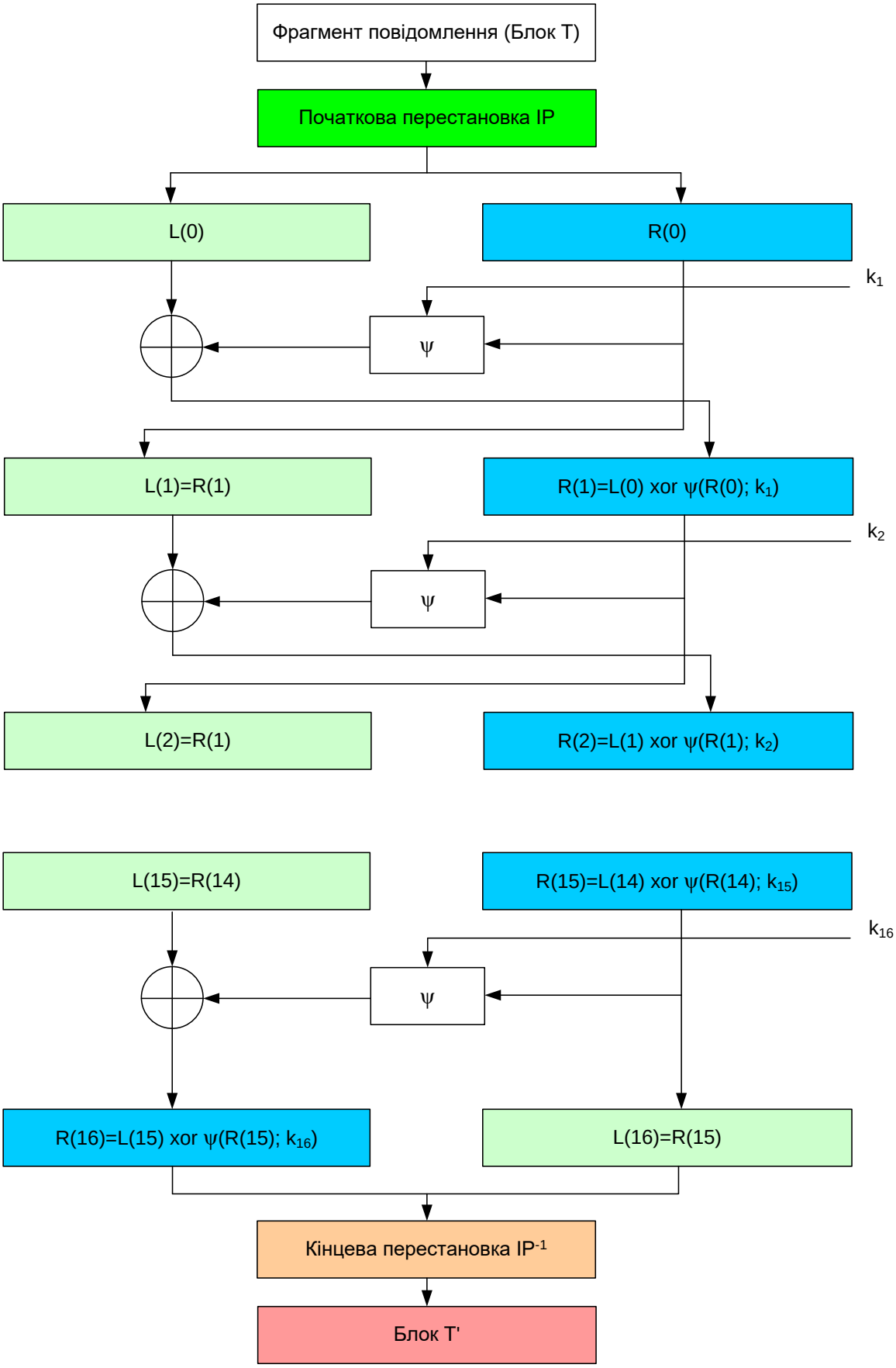


Рисунок 3.2 - Структура алгоритму шифрування DES

Окремо розглянемо зазначену раніше функцію шифрування f(R(i-1),K(i)). Графічна інтерпретація даної функції демонструється на рисунку 3.3.

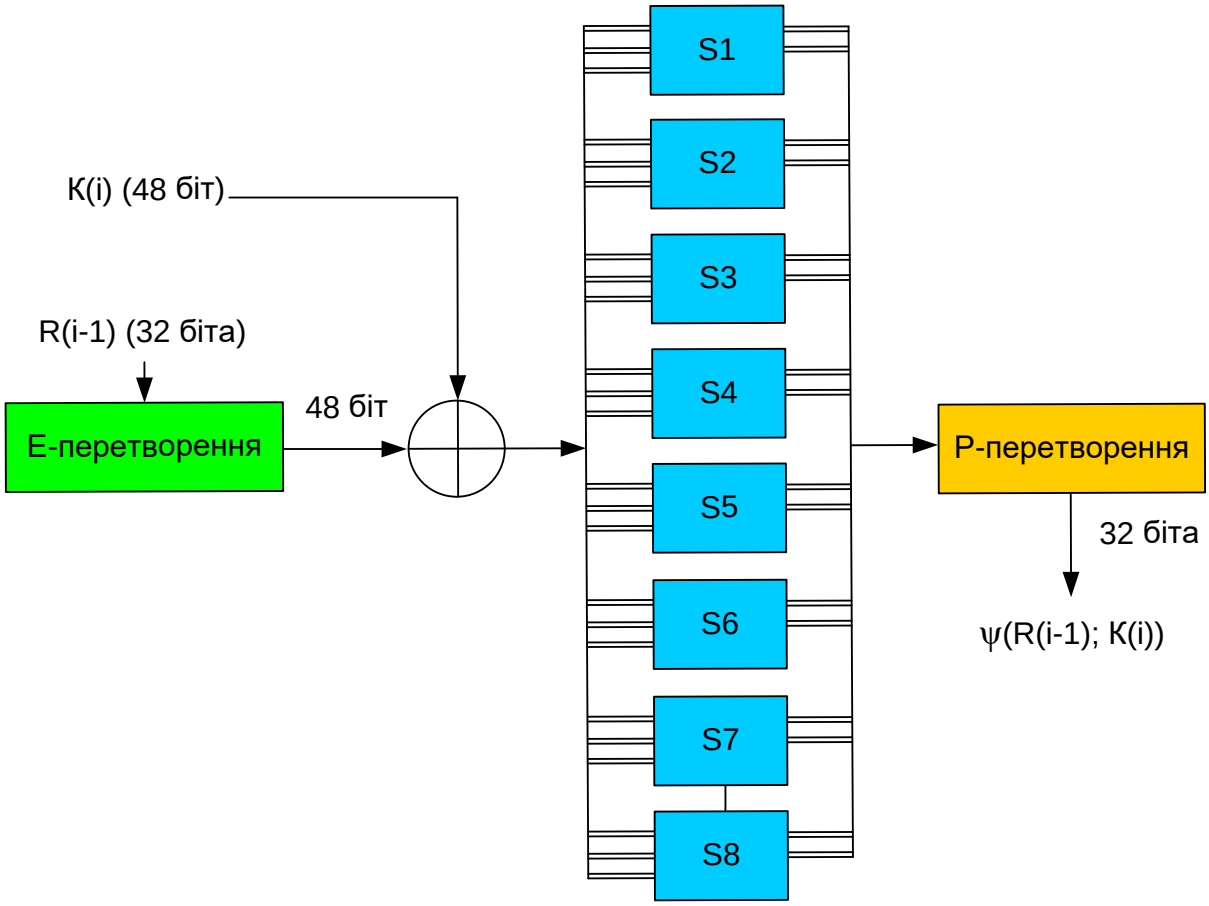


Рисунок 3.3 – Схема реалізації раундової функції

Як свідчить аналіз схеми на рис 3.3, тут для розрахунку f(R(i-1),K(i)) додатково застосовуються такі механізми, як:

- механізм розширення (E-перетворення), за використання якого 32-бітову послідовність, що йде на обробку, конвертується до послідовності довжиною 48 біт;
- підстановка (S-блоки – S1-S8), у результаті чого блоки 6 біт конвертуються у 4-бітні;
- перестановка, або P-перетворення, яке реалізує P-блок, оперуючи бітами у послідовності довжиною 32 біта.

При цьому, механізм розширення встановлює табличну залежність між бітами початкової та розширеної послідовностей. Сформована у результаті E-процедури послідовність довжиною 48 біт далі підлягає процедурі

накладання ключа K(i) такої ж довжини. Відносно одержаної у підсумку даної операції послідовності далі виконується поділ на сегменти довжиною 6 блоків, таким чином утворюючи 8 блоків, а саме $B(1) - B(8)$. Аналітичний вигляд даної процедури є таким:

$$E(R(i-1)) \oplus k_i = B(1)B(2)...B(8) \qquad (3.3)$$

Разом з тим, функції підстановки S-блоків, з S1 по S8, визначаються таблично. При цьому, принцип їх реалізації є незмінним за будь-яких обставин, як і E-процедури.

У підсумку, фінальним механізмом у складі функції шифрування є перестановка біт у межах утвореної послідовності, який реалізується на базі P-блоку. Дану функцію також реалізовано у вигляді табличного співставлення початкових та результуючих позицій біт.

Аналітичний вигляд даного перетворення ілюструє наступний вираз:

$$\psi(R(i-1); k_i) = P(S1(B(1)),...S8(B(8)) \qquad (3.4)$$

Окремо виконаємо аналіз функціонування механізму генерування раундових ключів K(i) при $i = \overline{1,16}$ довжиною 48 біт кожен з початкового 64-бітового секретного ключа K (65 початкових + 8 перевірочних біт на позиціях 8, 16, 24, 32, 40, 48, 56 та 64 відповідно).

Спочатку зазначимо, що для кожного раунду застосовується окремий ключ K(i), відмінний від інших.

При цьому, першим технологічним етапом у ході підготовки ключів є видалення перевірочних (контрольних) біт. Далі виконується операція перестановки інших біт в одержаній послідовності. Таку операцію реалізує функція підготовки ключа (G-перетворення), яка, у свою чергу, також задає табличну відповідність вихідному та фінальному порядку слідування біт. Результатом даного перетворення є послідовність G(K) біт.

Далі, одержана на передуючому кроці послідовність G(K) ділиться на дві рівні частини з виокремленням сегментів C(0) та D(0), довжина кожного з яких складає 28 біт. Водночас, блок C(0) утворюється бітами з 57 по 36 початкового ключа шифрування K з таблиці G-перетворення. У свою чергу,

блок D(0) формують біти, які належать діапазону позицій 63 - 4 вихідного ключа K. Маючи сформовані сегменти C(0) та D(0), далі здійснюється операція рекурсивного визначення C(i) та D(i) для $i = \overline{1;16}$.

Власне, сама процедура рекурсивного розрахунку полягає у виконання циклічного циклічного зсуву за напрямком вліво на одну або дві позиції. При цьому, величина зсуву також задається таблично (табл.3.1), беручи до уваги номер поточного раунду [].

Таблиця 3.1 – Визначення номіналу зсуву залежно від раунду шифрування у процесі формування раундового ключа

| Раунд | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Зсув, біт | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 |

Далі відносно даних, які отримуються за результатом виконання наведеної процедури, застосовуються операція перемішування.

У свою чергу, операція перемішування (H-перетворення) також є стандартизованою у межах DES та задається таблично.

Врешті решт, ключ K(i) для і-го раунду утворюватиметься на базі біт, що мають позиції з 14 по 32 послідовності C(i)D(i). Це еквівалентно виразу:

$$k_i = H(C(i)D(i)). \qquad (3.5)$$

Сам алгоритм побудови раундових ключів демонструється на рисунку 3.4. Таким чином, підводячи підсумок за результатами виконання дослідження закономірностей реалізації шифросистеми DES, можемо стверджувати про наступне:

- шифросистема DES, попри необхідність виконання значної кількості операцій у ході кожного з 16 раундів шифрування здатна забезпечити швидкістю генерації шифрограми на рівні не менш, ніж 800 Кбіт/с;
- загальний алгоритм, за яким побудовано досліджувану шифросистему, є алгоритмічно простим; при цьому, найбільш ресурсоємкою процедурою у ході DES-шифрування є складання за модулем 2;

- не дивлячись на те, що криптографічна система DES є повінстю формалізованою, її архітектура дозволяє долати обмеження, внесені формалізацією, та вносити ті чи інші зміни у структуру ширфу;
- з позиції можливості внесення змін та подальшої реалізації за власним сценарієм, або виключно слідування загальній архітектурі алгоритму, DES являє собою зручний та ефективний базис для розробки модифікованої ширфосистеми.

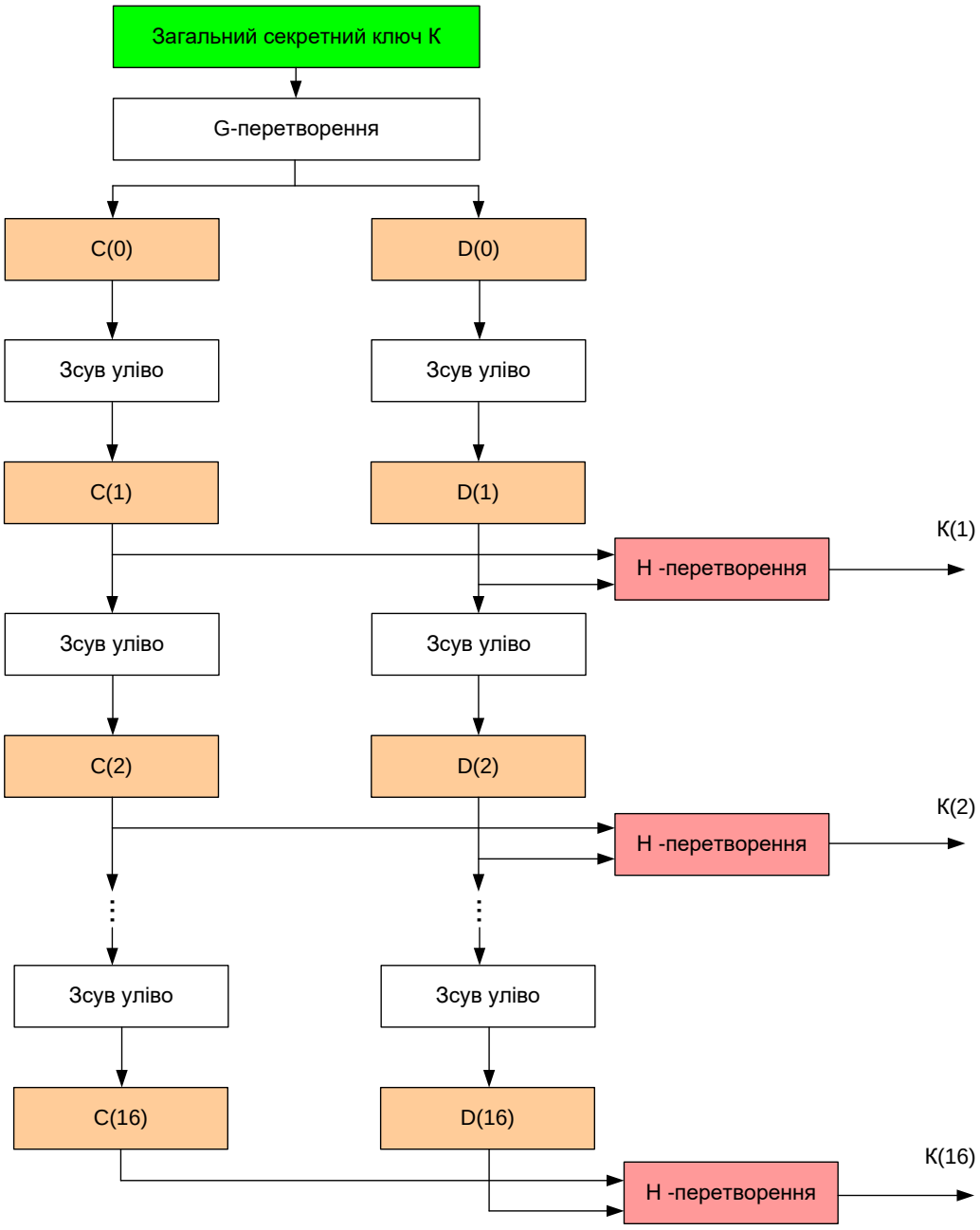


Рисунок 3.4 – Схема формування раундових ключів DES на базі загального ключа K

Отже, DES, та існуюча архітектура даного шифру, може використовуватися, як основа для побудови власної криптосистеми симетричного блокового композитного типу.

При цьому, пропонується використати загальну ідеологію шифру, вносячи зміни у механізми реалізації раундової функції та механізми формування раундових ключів.

# 4. ПОБУДОВА ШИФРУ НА СТРУКТУРНИХ ЗАСАДАХ БЛОКОВОГО АЛГОРИТМУ

## 1.1 Умови побудови та тестування шифру на базі загальної ідеології симетричних блокових систем

### 4.1.1 Апаратний базис

Проектування та подальша програмна реалізація шифру, утвореного на загальних принципах стандарту шифрування DES та у рамках структури мережі Фейстеля, виконувалася робочій станції PC-архітектури з наступними параметрами:

- 12 GB RAM;
- CPU 11th Gen Intel(R) Core (TM) i5-1155G7, 4x2,5 GHz cores;
- PCIe-4 SSD 512 GB.

### 4.1.2 Програмний базис

Розробка програмного модулю виконувалася у середовищі Windows 11 Home, версії 22H1, архітектура x64.

У якості засобу розробки використано середовище Python 3.10 [18].

## 2.2 Загальна архітектура створюваного шифру

У загальному випадку, пропонується, зберігаючи загальну ідеологію мережі Фейстеля, виконати побудову шифру, де з архітектури Фейстеля, та зі стандарту DES запозичено:

- загальний сценарій блокової обробки;
- наявність обробки на рівні раундової функції;
- процедури поділу та заміни місць для половин блоку, а також механізм доповнення помідомлення до довжини, що є кратною довжині блоку.

Разом з тим, зміні підлягають:

- механізми перестановок;
- кількість раундів;

- механізм виокремлення раундових ключів;
- механізми змішування рядків/стовпців;
- розмір блоку;
- формат обробки даних – передбачається виконувати шифрування на бінарному рівні.

При цьому, стосовно до нашого випадку пропонується обрати довжину блоку, яка є рівною 1 байт. У свою чергу, обробка суб-блоків (L та R відповідно) виконується на рівні 4 біти (рис.4.1).

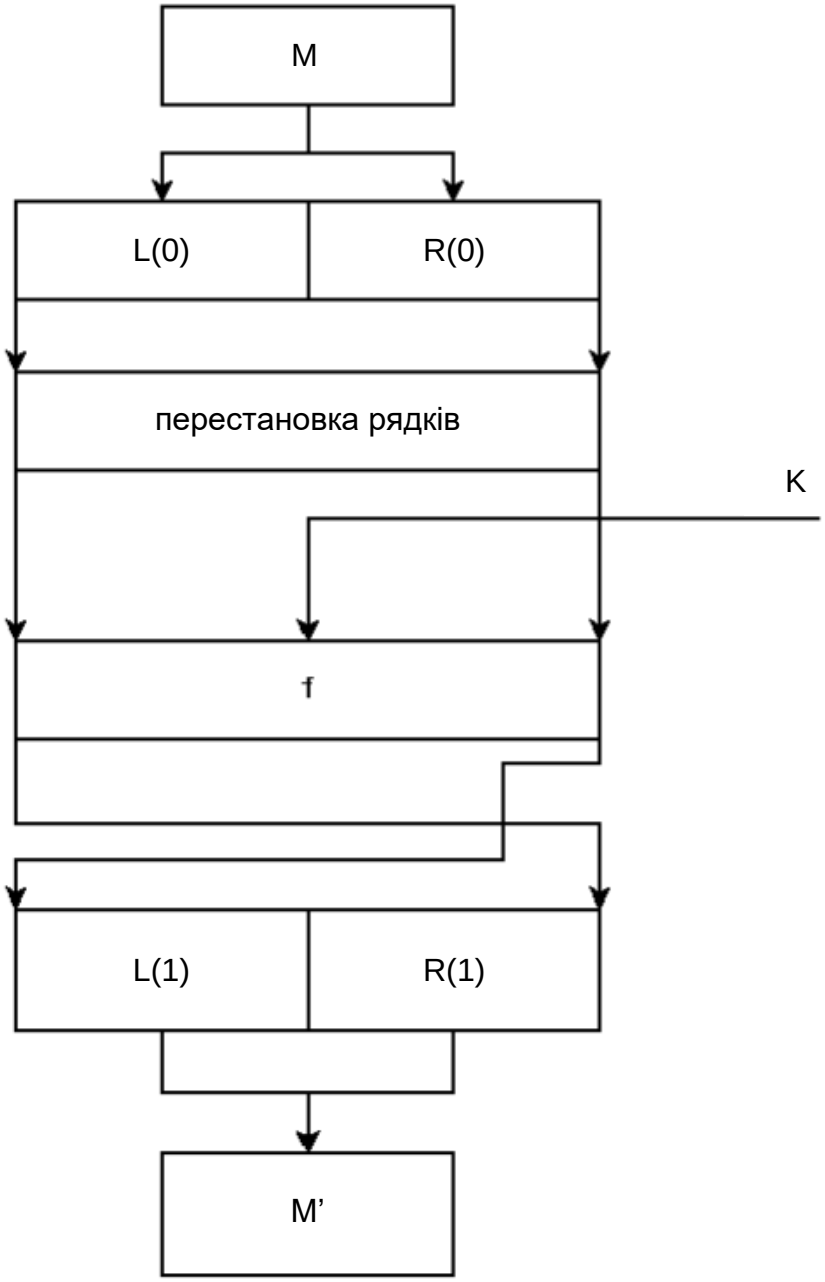


Рисунок 4.1 – Загальна схема створюваного шифру

Разом з тим, у якості раундової функції пропонується використати:

- механізм перестановки біт кожного з суб-блоків;
- циклічний зсув рядків/стовпців;
- спліт-механізм накладання секретного ключа.

Далі, у рамках обраного власного базису перетворень, виконаємо спробу побудови шифру на базі структури Фейстеля за умови, що:

- попередньо виконується спроба побудови шифру у складі 2-х раундів, що, за потреби, за результатами дослідження далі може бути скореговано у бік збільшення обсягу раундів;
- попередньо механізм виокремлення раундових ключів з загального секретного ключа не реалізується; ключ застосовується на першому раунді шифрування; за потреби (незадовільні результати шифрування), відповідно до розширення кількості раундів шифрування. може бути запроваджено механізм отримання раундових ключів.

У свою чергу, передбачається така послідовність операцій у ході циклу шифрування, як показано далі:

1. Вихідне повідомлення M довільної довжини n перетворюється до двійкового вигляду.

2. Виконується поділ повідомлення M на відрізки, кратні за довжиною $m^2$ кожен (за замовчуванням приймаємо m=2).

3. За необхідності, повідомлення доповнюється до розміру, кратного $m^2$.

4. Перетворення відрізків довжиною $m^2$ на двовимірний масив, фактично, утворюючи тим самим блок mxm.

5. Змішування рядків/стовпців утвореного блоку.

6. Накладання ключа довжиною m/2 на половину рядків/стовпців за деяким правилом.

7. Застосування циклічного зсуву відносно рядків/стовпців (якщо у п.5 виконувалася обробка за стовпцями, перетворення виконується на рівні рядків і навпаки).

При цьому, цикл дешифрування міститиме у собі аналогічний перелік етапів перетворення, які мають виконуватися у зворотному порядку

Таким чином, на першому етапі побудови шифру необхідно розробити механізми функціональних перетворень для кожного з зазначених вище технологічних етапів для циклів шифрування та дешифрування.

## 4.3 Кодова реалізація шифру

### 4.3.1 Розробка механізмів функціональних перетворень для циклу шифрування

У першу чергу, створюємо функцію, яка виконує перетворення довільного символьного тексту на двійковий масив. Для цього слугує наступний фрагмент коду:

```
def binary_text(text):
binary_text = ''
for char in text:
if char == ' ':
binary_text += '00100000' # двійковий опис символу
«пробіл»
else:
binary_text += format(ord(char), '08b')
return binary_text
```

Далі необхідно повідомлення довільної довжини розширити до розміру, що є кратним $m^2$, що виконує наступна функція:

```
def extend_text(binary_text, m):
 while len(binary_text) % (m ** 2) != 0:
binary_text += '0' # Текст доповнюється нулями
return binary_text
```

Після цього утворений таким чином двійковий масив ділиться на сегменти довжиною $m^2$, за що відповідає наступна функція:

```
def split_text_into_segments(text, m):
segments = []
for i in range(0, len(text), m ** 2):
segment = text[i:i + m ** 2]
```

```
segments.append(segment)
return segments
```

Далі створюємо функцію segments_to_2d_array, що ініціює двовимірний масив, та заповнює його відрізками $m^2$ за стовпцями:

```
def segments_to_2d_array(segments, m):
num_segments = len(segments)
num_columns = m
num_rows = (num_segments + m - 1) // m
array_2d = [['' for _ in range(num_columns)] for _
in range(num_rows)]
for i in range(num_rows):
for j in range(num_columns):
segment_index = i * num_columns + j
if segment_index < num_segments:
array_2d[i][j] = segments[segment_index]
return array_2d
```

Після цього необхідно утворений двовимірний масив сегментувати на блоки m x m розміром кожний. Виконання даної операції забезпечується впровадженням функції divide_into_blocks, лістинг якої продемонстрвано нижче:

```
def divide_into_blocks(array, m):
blocks = []
num_rows = len(array)
num_columns = len(array[0])
for i in range(0, num_rows, m):
for j in range(0, num_columns, m):
block = []
for x in range(i, min(i + m, num_rows)):
block_row = []
for y in range(j, min(j + m, num_columns)):
block_row.append(array[x][y])
block.append(block_row)
```

```
if block:
blocks.append(block)
return blocks
```

У свою чергу, механізм перестановлення елементів блоку, на рівні якого виконується обробка, забезпечується функцією permute_blocks. Дана функція, у сутності, вконує заміну місць першого і останнього елементів блоку:

```
def permute_blocks(blocks):
permuted_blocks = []
for block in blocks:
permuted_block = []
for row in block:
permuted_row = [row[-1]] + row[1:-1] + [row[0]]
permuted_block.append(permuted_row)
permuted_blocks.append(permuted_block)
return permuted_blocks
```

У ході реалізації наступної функції, а саме – функції накладання ключа, було використано адаптивній підхід.

У рамках цього, накладання ключа здійснюється лише на один з суб-блоків. При цьому, вибір того чи іншого суб-блока тут здійснюється на базі наступного правила:

- якщо різниця максимального та мінімального елементів суб-блоку є більшою, ніж половина максимального елементу, тоді ключ буде застосовано відносно першого суб-блоку;

- в інакшому випадку ключ накладається на другий суб-блок.

Окрім цього, у рамках раундової функції виконується також процедура циклічного зсуву елементів рядків відповідно з адаптивним принципом, розглянутим вище, а саме:

– якщо різниця максимального і мінімального елементів є більшою, ніж половина максимального елемента, виконується циклічний зсув вліво на 1 позицію;

- в інакшому випадку виконується циклічний зсув на 1 праворуч.

Виконання зазначених механізмів забезпечується функціями shift_blocks, та відповідно, apply_key:

```
    def shift_blocks(blocks):
    shifted_blocks = []
    for block in blocks:
    min_val = min(map(int, [''.join(row) for row in
block]))
    max_val = max(map(int, [''.join(row) for row in
block]))
    sum = max_val - min_val
    if sum > 0.5 * max_val:
    shift_direction = -1
    else:
    shift_direction = 1
    shifted_block = []
    for row in block:
    shifted_row = row[shift_direction:] +
row[:shift_direction]
    shifted_block.append(shifted_row)
    shifted_blocks.append(shifted_block)
    return shifted_blocks

    def apply_key(block, key):
    min_val = min(map(int, [''.join(row) for row in
block]))
    max_val = max(map(int, [''.join(row) for row in
block]))
    diff = max_val - min_val
    result_block = []
    for row in block:
    half = len(row) // 2
    result_row = []
    for i in range(len(row)):
    if diff > 0.5 * max_val:
    if i < half:
```

```
result_row.append(str(int(row[i]) ^ int(key[i])))
else:
result_row.append(row[i])
else:
if i >= half:
result_row.append(str(int(row[i]) ^ int(key[i])))
else:
result_row.append(row[i])
result_block.append(result_row)
return result_block
```

4.3.2 Розробка механізмів функціональних перетворень для циклу дешифрування

Хоча, у цілому, процедура дешифрування включає у себе перелік функціональних перетворень, що виконуються у зворотньому порядку, тут необхідно створити ряд специфічних механізмів, що виконують:

- доповнення блоку нульовими символами до необхідної довжини;
- поєднання блоків у єдиний масив;
- поєднання сегментів у суцільний рядок;
- відновлення тексту з двійкового формату.

Так, наступна функція з циклу дешифрування виконує допис нулів у блок, якщо його фактична довжина є недостатньою:

```
def fill_zeros(block, m):
    result_block = []
    for row in block:
    result_row = []
    for element in row:
    if len(element) < m**2:
    element = element.zfill(m**2)
    result_row.append(element)
    result_block.append(result_row)
return result_block
```

Після цього, за зворотнім сценарієм, необхідно поєднати окремі блоки у єдиний масив. Зазначену операцію реалізує відповідна функція:

```
def join_blocks(blocks):
join_array = []
for block in blocks:
for i in range(len(block)):
row = ''.join(str(cell) for cell in block[i])
join_array.append(row)
return join_array
```

У свою чергу, окремі сегменти поєднуються у суцільний рядок за допомогою наступної функції:

```
def join_segments(segments):
return ''.join(segments)
```

Далі виконується зворотнє перетворення бінарного масиву у текст:

```
def binary_to_text(binary_string):
text = ''
for i in range(0, len(binary_string), 8):
byte = binary_string[i:i + 8]
text += chr(int(byte, 2))
return text
```

### 4.4 Збірка головного модулю програми

Робочий модуль додатку, що реалізує функції шифрування та дешифрування, звертається до функцій, які було попередньо розроблено.

При цьому, для тестування функціональності додатку використано спрощений режим вибору параметру m відрізку, на базі якого далі формується блок, з вводом даних у вікно коду Python. Це саме стосується тестового повідомлення та ключа.

Повний текст головного модулю розміщено у Додатку Б.

## 4.5 Перевірка функціональності створеного шифру

Тест було виконано для найгірших умов, за мінімальної довжини блоку, коли m=2. Ша цей випадок ключ буде обмежено двома десятковими числами відповідно довжини блоку, що максимально зменшує стійкість. Свого часу саме недостатня довжина ключа зумовила вдалий злам DES [19]. Тобто, за таких умов маємо змогу зробити висновок стосовно потенціалу шифру – зокрема, про його функціональність у зазначених екстремальних умовах – і як наслідок – зробити висновок щодо доцільності використання для нього ключів більшої довжини, так як алгоритм надає таку змогу.

У ролі тестового повідомлення було використано текстовий рядок довільного змісту, а саме:

*Hello myuuo ytkfkkfkjhu*

При цьому, у якості ключа було використано пару чисел [199, 56] що відповідає випадку m=2.

Результат шифрування та відновлення зазначеного повідомлення демонструється на рисунку 4.2.

```
Entire text:  Hello myuuo ytkfkkfkjhu

Ciphered message:
[['0100', '815'], ['0110', '162']]
[['186', '1100'], ['86', '1100']]
[['0110', '1168'], ['0010', '199']]
[['86', '1101'], ['87', '1001']]
[['87', '0101'], ['87', '0101']]
[['0110', '1168'], ['0010', '199']]
[['0111', '816'], ['0111', '163']]
[['0110T', '820'], ['0110', '169']]
[['86', '1011'], ['86', '1011']]
[['0110', '169'], ['0110', '820']]
[['86', '1010'], ['86', '1000']]
[['87', '0101']]

Deciphered message:  Hello myuuo ytkfkkfkjhu
```

Рисунок 4.2 – Результат виконання шифруючих да дешифруючих перетворень на базі реалізованого шифру

Результат аналіз рисунку 4.2 дозволяє зробити висновки про наступне:

- утворений шифр  являє собою масив пар десяткових чисел, який також може бути подано у визгялі неструктурованої послідовності;
- кожна пара чисел являє собою окремий шифрований символ;
- у структурі шифрру може бути візуалізовано окремі структурні закономірності, характерні тексту на «природній» мові; даний ефект є властивим багатьом стандартним засобам шифрування – зокрема, RSA, та не свідчить про неефективність криптографічного засобу; для його усунення достатньо виконати передуюче змішування символів;
- для шифрування достатньо двох умовних раундів, при цьому збільшення їх кількості у даному випадку не є необхідним;
- оскільки результат шифрування не містив критичних ознак вихідного повідомлення, його функціональність за найгірших умов доведено. Відтак, його стійкість може бути підсилено використанням ключа більшої довжини.

# ВИСНОВКИ

У ході опрацювання технічного завдання до кваліфікаційної роботи, з самого початку було виконано:

- огляд ролі та специфіки застосування криптографічних методів для захисту даних;
- дослідження принципів реалізації криптографічних систем та їх особливостей, включаючи розгляд ключових вимог до них та сутність кожної окремої вимоги.

За результатами дослідження специфіки застосування методів шифрування та існуючих вимог до них, а також, керуючись вимогами технічного завдання було виявлено, шо:

- з точки зору забезпечення максимальних рівнів криптографічної стійкості, пропускної здатності за умови одночасної простоти алгоритмічної реалізації шифросистеми, найбільш прийнятним для використання є шифр Rijndael (AES);
- шифросистема DES незначно поступається Rijndael за показниками алгоритмічної складності та пропускної здатності, та суттєво програє за рівнем стійкості;
- з іншого боку, якщо брати до уваги можливість модифікації шифру шляхом збереження його архітектури, з одночасною зміною окремих функціональних блоків, для цього випадку використання шифросистеми Rijndael не є доцільним, так як система є повністю формалізованою а відтак – внесення зміни у будь-який функціональний блок незмінно викликає необхідність модифікації інших блоків;
- архітектура DES, утворена на базі мережі Фейстеля, попри свою суттєву формалізацію, може бути використана у якості базису для побудови блокової шифросистеми.

Виходячи з вищезазначеного, було побудовано блокову симетричну щифросистему на базі мережі Фейстеля. Така шифросистема містить у собі окремы ознаки  DES, однак, на відміну від зазначеного криптографічного алгоритму, не використовує систему таблиць перестановок.

Головна відмінність реалізованої шифросистеми, окрім відмови від табличних перестановок, полягає у змісті перетворень, частина з яких виконуэться залежно від особливостей змісту оброблюваних блоків повідомлення.

Таким чином, усі пункти технічного завдання виконано повністю.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Microsoft security report [Електронний ресурс] – Режим доступу: https://microsoft.com/securityinsights.

2. Antivirus and Cybersecurity Statistics & Facts 2021 [Електронний ресурс] – Режим доступу: https://wethegeek.com/antivirus-statistics-facts/

3. Microsoft unsecurity solutions [Електронний ресурс] Режим доступу: http://www.seattlepi.com/business/275780_msftsuit29.html

4. Шаньгін В.Ф. Захист інформації в розподілених корпоративних мережах і системах [Текст]: підручник / В. Ф. Шаньгін, А. В. Соколов. – ДМК Прес, 2002. – 656 с.

5. Шенон К. Теория связи в секретных системах / пер. с англ. В. Ф. Писаренко // Работы по теории информации и кибернетике / Под редакцией Р. Л. Добрушина и О. Б. Лупанова. — М. : Издательство иностранной литературы, 1963. — 829 с.

6. Andress Jason. Foundations of Information Security: A Straightforward Introduction. No Starch Press. 2019. - 248 p.

7. Мао В. Современная криптография: Теория и практика / пер. Д. А. Клюшина — Вильямс, 2005. — 768 с.

8. GPG - Encrypt and Decrypt Files with a Key on Linux [Електронний ресурс] Режим доступу: https://www.tecmint.com/gpg-encrypt-decrypt-files/

9. Cloud Telegram Password: How to Set Up, Install and Use [Електронний ресурс] Режим доступу: https://t9gram.com/p/cloud-telegram-password

10.Рябко Б.Я. Основы современной криптографии и стеганографии [Электронный ресурс] : [монография] / А.Н. Фионов, Б.Я. Рябко. — М.: Горячая линия – Телеком, 2010.— 233 с.

11. Rivest R., Shamir A., Adleman L. A method for obtaining digital signatures and public-key cryptosystems // Commun. ACM - [New York]: Association for Computing Machinery, 1978. - Vol. 21, Iss. 2. - P. 120-126.

12. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы и исходный код на С. – М.: Вильямс, 2016. – 1024 с.

13. Cryptology ePrint Archive [Електронний ресурс] – Режим доступу: https://eprint.iacr.org/

14. Ferguson N., Schroeppel R. and Whiting D. A simple algebraic representation of Rijndael. Selected Areas in Cryptography, Proc. SAC 2001,

Lecture Notes in Computer Science #2259. — Springer Verlag, 2001. — P. 103—111.

15. Rijndael block cipher [Електронний ресурс] – Режим доступу: https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf

16. Block Ciphers — Introduction and overview [Електронний ресурс] – Режим доступу: http://cacr.uwaterloo.ca/hac/about/chap7.pdf

17. Dhiman Saha, Debdeep Mukhopadhyay, Dipanwita RoyChowdhury. A Diagonal Fault Attack on the Advanced Encryption Standar (англ.) // Cryptology ePrint Archive. — 2009

18. Python Tutorial [Електронний ресурс] – Режим доступу: https://www.pythontutorial.net/

19. Schindler W., Koeune F., Quisquater J-J. Improving Divide and Conquer Attacks against Cryptosystems by Better Error Detection/Correction Strategies. Proc. of 8th IMA International Conference on Cryptography and Coding :— 2001. — P. 245—267. — doi:10.1.1.13.5175