МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Факультет: **ННІ Каразінський банківський інститут**
Кафедра: **Інформаційних технологій та математичного моделювання**
Спеціальність: **122 Комп'ютерні науки**
Освітня програма: **Комп'ютерні науки та інформаційні технології в бізнесі**

Група: АК-21М **денна форма навчання**

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему:

«ДОСЛІДЖЕННЯ ЗАСТОСУВАННЯ ТРАДИЦІЙНИХ СЕМИТРИЧНИХ КРИПТОСИСТЕМ ДЛЯ БАНКІВСЬКИХ ДОДАТКІВ»

ЗА НАКАЗОМ № 4601-5/3045 ВІД 25 ВЕРЕСНЯ 2024 РОКУ

здобувача вищої освіти **Ечкенка Кирила Васильовича**

Робота допущена до захисту в ЕК
протокол кафедри ІТММ № 4 від 30.11.2024 р.

Завідувач кафедри ІТММ
к. п. н., доцент
_____**Н. І. Стяглик**

Науковий керівник
Ph.D з «Комп'ютерних наук»
_____**Д. М. Ковальчук**

м. Харків 2024 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В. Н. Каразіна

Факультет навчально-науковий інститут “Каразінський банківський інститут”
Кафедра інформаційних технологій та математичного моделювання
Рівень вищої освіти перший (магістерський)
Спеціальність 122 Комп’ютерні науки
Освітня програма Комп’ютерні науки та інформаційні технології в бізнесі

ЗАТВЕРДЖУЮ

Завідувач кафедри

Н. І. Стяглик

“25” вересня 2024 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ (ПРОЕКТ)

Ечкенка Кирила Васильовича
(прізвище, ім’я, по батькові студента)

1. Тема роботи «Дослідження застосування традиційних семитричних криптосистем для банківських додатків»

Керівник роботи Ph.D з «Комп’ютерних наук» Ковальчук Д.М.

затверджені наказом по університету від “25” вересня 2024 року №4601-5/3045

2. Строк подання студентом роботи 20 листовада 2024 р.

3. Перелік питань, які потрібно розробити:
У розділі 1: Провести аналіз банківських додатків та визначення загроз до їх безпеки.
У розділі 2: Провести аналіз симетричних методів криптографії, дослідити основні алгоритми симетричного шифрування.
У розділі 3: Дослідити застосування традиційних симетричних криптосистем; розробити програмну реалізацію обраних симетричних крипто алгоритмів.

4. План роботи

№ з/п	Назви етапів роботи
1	Вибір здобувачем теми кваліфікаційної магістерської роботи
2	Затвердження плану і завдання кваліфікаційної магістерської роботи
3	Здача кваліфікаційної магістерської роботи керівнику
4	Підпис кваліфікаційної магістерської роботи у керівника
5	Підпис кваліфікаційної магістерської роботи у нормо контролера
6	Допуск завідувачем кафедри до захисту кваліфікаційної магістерської роботи
7	Захист кваліфікаційної магістерської роботи

5. Дата видачі завдання 25 вересня 2024 року

Студент _____ К. В. Ечкенко
Підпис інніціали, прізвище

Керівник роботи _____ Д. М. Ковальчук
підпис інніціали, прізвище

РЕФЕРАТ
НА КВАЛІФІКАЦІЙНУ МАГІСТЕРСЬКУ РОБОТУ
«ДОСЛІДЖЕННЯ ЗАСТОСУВАННЯ ТРАДИЦІЙНИХ СЕМИТРИЧНИХ КРИПТОСИСТЕМ ДЛЯ БАНКІВСЬКИХ ДОДАТКІВ»
Ечкенко Кирил Васильович

Кваліфікаційна магістерська робота містить 62 сторінки, 3 таблиця, 12 рисунків, список літератури з 25 найменувань.

Об'єктом дослідження є алгоритми симетричного шифрування**.**

Предметом дослідження є процес дослідження застосування традиційних алгоритмів симетричного шифрування для банківських додатків.

Мета кваліфікаційної магістерської роботи полягає у дослідження традиційних симетричних криптосистем.

Завданнями кваліфікаційної магістерської роботи є:

- провести аналіз банківських додатків та визначення загроз до їх безпеки;
- провести аналіз симетричних методів криптографії;
- дослідити основні алгоритми симетричного шифрування;
- дослідити застосування традиційних симетричних криптосистем;
- розробити програмну реалізацію алгоритмів, що дають змогу шифрувати дані за допомогою обраних шифрів.

Актуальність дослідження: розвиток технологій, таких як мобільні банківські додатки та онлайн-платформи, створює нові виклики для забезпечення безпеки даних. Сучасні симетричні криптосистеми можуть забезпечити необхідний рівень захисту для цих платформ. Їх дослідження стає важливим для розвитку комплексних рішень у сфері кібербезпеки та досягнення більшого рівня захисту.

За результатами дослідження: сформовано основні теоретичні аспекти використання алгоритмів шифрування та дешифрування даних методами підстановки в інформаційно-телекомунікаційних системах.

Практична новизна: розроблено коди на мові програмування Python, що дозволяють шифрувати та дешифровувати дані класичними симетричними криптоалгоритмами. Проведено порівняльний аналіз класичних симетричних криптоалгоритмів.

Одержані результати можуть бути використані: при створені надійної системи безпеки, що захищає банківський додаток від загроз, мінімізує ризики витоку даних і забезпечує безпеку фінансових операцій клієнтів.

КЛЮЧОВІ СЛОВА: БЕЗПЕКА БАНКІВСЬКИХ ДОДАТКІВ, СИМЕТРИЧНІ КРИПТОАЛГОРИТМИ, AES, DES, 3DES, TWOFISH, BLOWFISH, RC4, PYTHON.

ABSTRACT
AT QUALIFICATION MASTER'S WORK
«RESEARCH OF THE APPLICATION OF TRADITIONAL SEMITRICAL CRYPTOSYSTEMS FOR BANKING APPLICATIONS»
Kirill Echkenko

The qualifying master's work contains 62 pages, 3 tables, 12 figures, a list of literature from 25 titles.

The object of research is symmetric encryption algorithms.

The subject of the study is the process of researching the application of traditional symmetric encryption algorithms for banking applications.

The purpose of the master's thesis is to study traditional symmetric cryptosystems.

The tasks of the qualifying master's thesis are:

- analyze banking applications and identify threats to their security;
- analyze symmetric cryptography methods;
- to study the main algorithms of symmetric encryption;
- investigate the use of traditional symmetric cryptosystems;
- to develop a software implementation of algorithms that enable data encryption using selected ciphers.

Relevance of the study: The development of technologies such as mobile banking applications and online platforms creates new challenges for data security. Modern symmetric cryptosystems can provide the necessary level of protection for these platforms. Their research becomes important for the development of complex solutions in the field of cyber security and achieving a higher level of protection.

According to the results of the research: the main theoretical aspects of the use of data encryption and decryption algorithms by substitution methods in information and telecommunication systems were formed.

Practical novelty: codes in the Python programming language have been developed that allow data to be encrypted and decrypted using classical symmetric crypto-algorithms. A comparative analysis of classical symmetric crypto-algorithms is carried out.

The obtained results can be used: when a reliable security system is created, which protects the banking application from threats, minimizes the risks of data leakage and ensures the security of financial transactions of customers.

KEY WORDS: SECURITY OF BANKING APPLICATIONS, SYMMETRIC CRYPTO ALGORITHMS, AES, DES, 3DES, TWOFISH, BLOWFISH, RC4, PYTHON.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧОК, СИМВОЛІВ І ТЕРМІНІВ

AES – Advanced Encryption Standard;

API – application programming interface;

DDoS-атаки – denial-of-service attack;

DES – Data Encryption Standard;

3DES – Triple DES;

DNS – Domain Name System;

DSA – Digital Signature Algorithm;

ECC – Elliptic Curve Cryptography;

2FA – двофакторна аутентифікація;

HMAC – Hash-Based Message Authentication Code;

HSM – Hardware Security Module;

MFA – багатофакторна аутентифікація

PGP – Pretty Good Privacy;

PKI – Public Key Infrastructure;

RC4 – Rivest Cipher 4;

RSA – Rivest-Shamir-Adleman;

SDLC – Software development lifecycle;

TLS/SSL – Transport Layer Security / Secure Sockets Layer;

VPN – virtual private network.

ВСТУП

З моменту своєї появи банки завжди приваблювали злочинців. У останні десятиліття, з розвитком систем електронного банкінгу, питання безпеки фінансових технологій та банківського сектору залишається надзвичайно актуальним. Сьогодні значна частина злочинів пов’язана з використанням автоматизованих систем обробки інформації банку. Тому при розробці банківського програмного забезпечення надзвичайно важливо приділяти особливу увагу безпеці банківських додатків.

У банківських програмних системах зберігається та обробляється конфіденційна інформація, що вимагає особливої уваги до її безпеки. Важливо забезпечити захист персональних даних клієнтів, інформації про фінансові операції та внутрішні процеси банку.

Отже, дослідження використання традиційних симетричних криптосистем у банківських додатках є ключовим елементом для забезпечення безпеки фінансових операцій. Симетричні криптосистеми застосовуються для шифрування даних, що передаються між клієнтами та банківськими системами, з метою захисту конфіденційної інформації від несанкціонованого доступу. Вони забезпечують швидке та ефективне шифрування, проте мають певні обмеження, що потребує їх поєднання з іншими криптографічними методами для досягнення вищого рівня безпеки.

Актуальність роботи. Розвиток технологій, таких як мобільні банківські додатки та онлайн-платформи, створює нові виклики для забезпечення безпеки даних. Сучасні симетричні криптосистеми можуть забезпечити необхідний рівень захисту для цих платформ. Їх дослідження стає важливим для розвитку комплексних рішень у сфері кібербезпеки та досягнення більшого рівня захисту.

Метою роботи є дослідження традиційних симетричних криптосистем.

Завдання дослідження:

– провести аналіз банківських додатків та визначення загроз до їх безпеки;

– провести аналіз симетричних методів криптографії;

– дослідити основні алгоритми симетричного шифрування;

– дослідити застосування традиційних симетричних криптосистем;

– розробити програмну реалізацію алгоритмів, що дають змогу шифрувати дані за допомогою обраних шифрів.

Об’єктом дослідження є алгоритми симетричного шифрування.

Предметом дослідження є процес дослідження застосування традиційних алгоритмів симетричного шифрування для банківських додатків.

У вступі представлено актуальність роботи, сформульовано мету та відповідні завдання, об’єкт та предмет дослідження, наведено загальну структуру роботи.

У першому розділі «Аналіз банківських додатків та визначення загроз до їх безпеки» наведені загальні характеристики банківських додатків, розглянуто теоретичні основи кібербезпеки банківських додатків.

У другому розділі «Аналіз симетричних методів криптографії» проведено аналіз симетричних методів криптографії. За результатами дослідження: сформовано основні теоретичні аспекти використання основних алгоритмів симетричного шифрування.

У третьому розділі «Дослідження застосування традиційних симетричних криптосистем» практично досліджено застосування традиційних симетричних криптосистем у банківських додатках. Розроблена програмна реалізація симетричних криптоалгоритмів, що дають змогу шифрувати дані за допомогою обраних шифрів.

Одержані результати можуть бути використані: при створені надійної системи безпеки, що захищає банківський додаток від загроз, мінімізує ризики витоку даних і забезпечує безпеку фінансових операцій клієнтів.

Висновок висвітлює інформацію щодо підсумків дослідження, його наукової та практичної значущості, можливі перспективи подальшого розвитку.

РОЗДІЛ 1
АНАЛІЗ БАНКІВСЬКИХ ДОДАТКІВ ТА ВИЗНАЧЕННЯ ЗАГРОЗ ДО ЇХ БЕЗПЕКИ

1.1. Загальна характеристика банківських додатків

Банківські додатки розробляються для надання клієнтам можливості управляти своїми фінансами, здійснювати транзакції, отримувати інформацію про рахунки, кредитні картки та інші фінансові послуги. Вони сприяють спрощенню фінансових операцій і покращують доступність банківських послуг [1, 2].

Банківські додатки дозволяють клієнтам здійснювати різноманітні фінансові операції, такі як перекази між рахунками, оплата товарів і послуг, погашення кредитів. Користувачі отримують актуальну інформацію про стан своїх рахунків, баланс, історію транзакцій, а також деталі кредитних продуктів і інших послуг, що пропонує банк. Додатки надають можливість подання заявок на кредити, перегляд умов кредитування, пропонують умови для інвестування в різні фінансові інструменти. Завдяки додаткам банки забезпечують більш оперативне та персоналізоване обслуговування, надаючи клієнтам доступ до різних сервісів в режимі реального часу. Сучасні банківські додатки дозволяють налаштовувати інтерфейс і функціонал відповідно до індивідуальних потреб та вподобань користувачів [2, 3].

Банківські додатки виконують наступні функції:

1. Управління рахунками:
   - перегляд балансу рахунків (поточних, депозитних);
   - доступ до історії транзакцій (виписки по рахунках);
   - отримання електронних виписок і рахунків.
2. Здійснення платежів:
   - перекази між власними рахунками або на рахунки інших користувачів;

- оплата комунальних послуг, кредитів та інших платежів;
- можливість сканування QR-кодів для миттєвих платежів.

3. Кредитування:
   - подання заявок на кредити (позики, автокредити, іпотеки);
   - управління існуючими кредитами (перегляд залишку, графіка платежів);
   - отримання інформації про відсоткові ставки та умови кредитування.
4. Інвестиційні послуги:
   - доступ до інформації про інвестиційні продукти (акції, облігації, фонди);
   - можливість покупки та продажу фінансових інструментів;
   - аналіз та моніторинг інвестиційного портфеля.
5. Бюджетування та фінансовий моніторинг:
   - інструменти для складання бюджету та відстеження витрат;
   - налаштування фінансових цілей та планів (збереження, інвестування);
   - отримання звітів про фінансову активність у зручному форматі.
6. Безпека:
   - багатофакторна аутентифікація для доступу до додатку;
   - шифрування даних під час передачі та зберігання;
   - можливість блокування картки або рахунку у разі втрати чи крадіжки.
7. Сповіщення та нагадування:
   - оперативні сповіщення про транзакції, платежі, знижки або акції;
   - нагадування про майбутні платежі, терміни погашення кредитів.
8. Додаткові послуги:
   - підключення до програм лояльності, акцій та спеціальних пропозицій;
   - використання чат-ботів або онлайн-підтримки для консультацій;

- доступ до фінансових новин і аналітики.

Таким чином, банківські додатки виконують ключову роль у спрощенні та покращенні взаємодії клієнтів з банківськими установами, надаючи їм широкий спектр фінансових послуг у зручному форматі. Важливою метою є захист конфіденційної інформації та фінансових даних клієнтів. Банківські додатки реалізують різні методи безпеки, такі як шифрування, аутентифікація користувачів та моніторинг підозрілої активності.

1.2. Реалізація банківських додатків

Банківські додатки реалізовані на різних платформах, що забезпечує зручний доступ до фінансових послуг для користувачів [3, 4]. Основні платформи банківських додатків включають:

1. Мобільні платформи:

- iOS - додатки для пристроїв Apple (iPhone, iPad) розробляються з урахуванням специфіки операційної системи iOS, зазвичай доступні через App Store;
- Android - додатки для пристроїв на базі Android (смартфони, планшети) можуть мати більшу різноманітність завдяки відкритій природі платформи. Вони зазвичай розміщуються в Google Play Store.

2. Веб-додатки - інтерактивні інтерфейси, доступні через веб-браузери, дозволяють користувачам управляти своїми рахунками та здійснювати фінансові операції з будь-якого комп'ютера або пристрою, підключеного до Інтернету. Веб-додатки можуть бути оптимізовані для використання на різних браузерах.

3. Гібридні додатки – додатки, які поєднують елементи нативних мобільних додатків і веб-технологій. Вони можуть працювати на різних платформах, забезпечуючи користувачам доступ до фінансових послуг незалежно від типу пристрою.

4. Додатки для носимих пристроїв. Деякі банки пропонують

спеціалізовані додатки для смарт-годинників або інших носимих пристроїв.

5. API та інтеграції. Багато банківських систем забезпечують API (інтерфейси програмування додатків) для інтеграції з сторонніми додатками або платформами, що дозволяє надавати додаткові послуги, такі як фінансове управління, аналітика та інше.

6. Кросплатформні рішення. Використання технологій, які дозволяють створювати додатки, що можуть працювати на кількох платформах одночасно. Це досягається за допомогою фреймворків, таких як React Native, Flutter або Xamarin.

Ці різні платформи надають користувачам можливість вибирати найбільш зручний для них спосіб доступу до банківських послуг.

Банківські додатки часто інтегруються з системами обробки платежів, бухгалтерського обліку, управління ризиками та іншими внутрішніми і зовнішніми платформами для забезпечення комплексного обслуговування клієнтів. Інтеграція банківських додатків з іншими системами дозволяє розширити їх функціонал та забезпечити користувачів додатковими послугами. Взаємодія банківських додатків з різноманітними зовнішніми системами та сервісами допомагає підвищити зручність, швидкість обробки даних і задовольнити потреби сучасних клієнтів. Основні напрямки інтеграції включають:

1. Інтеграція з платіжними системами: Visa, MasterCard, Apple Pay, Google Pay дозволяє користувачам прив'язувати свої банківські картки до мобільних гаманців, що спрощує безконтактну оплату.

2. Відкритий банкінг (Open Banking) дозволяє користувачам за допомогою API об’єднувати дані з різних банківських рахунків у одному додатку, отримуючи повну картину своїх фінансів.

3. Інтеграція з системами лояльності та кешбеку дозволяє користувачам накопичувати бали або отримувати кешбек за певні покупки.

4. Взаємодія з кредитними бюро та системами скорингу дає можливість банкам перевіряти кредитну історію клієнтів, що спрощує процес

отримання кредитів. Системи скорингу можуть надавати інформацію для попередньої оцінки кредитоспроможності.

5. Інтеграція з державними системами для надання послуг, таких як оплата податків, комунальних послуг, штрафів або перевірка соціальних виплат.

6. Інтеграція з біржами та інвестиційними платформами (інвестування в акції, облігації, фонди та інші активи).

7. Інтеграція з системами захисту та верифікації - співпраця із системами біометричної верифікації та кібербезпеки для підвищення рівня безпеки. Це включає технології розпізнавання обличчя, відбитків пальців, а також інструменти моніторингу підозрілої активності.

8. Інтеграція з чат-ботами та системами підтримки для отримання швидкої відповіді на питання та звернення за допомогою до служби підтримки:

9. Інтеграція з бухгалтерськими та ERP-системами для бізнес-клієнтів.

10. Інтеграція з соціальними мережами для авторизації через соціальні мережі.

1.3. Структура банківських додатків

Структура банківських додатків зазвичай будується на багаторівневій архітектурі, яка забезпечує зручність, масштабованість і безпеку [4, 5]. Основні компоненти структури банківських додатків включають:

1. Інтерфейс користувача (Frontend):

– мобільний та веб-інтерфейси. Додаток доступний як через мобільні платформи (Android, iOS), так і через веб-браузери. Інтерфейс має бути інтуїтивно зрозумілим і забезпечувати легкий доступ до основних функцій, таких як перевірка балансу, перегляд транзакцій, переказ коштів, оплата рахунків тощо;

– UX/UI дизайн. Інтерфейс повинен бути оптимізований для зручності користувачів та відповідати сучасним вимогам дизайну.

2. Середній рівень:

– серверна логіка додатку - обробляє бізнес-логіку та координує взаємодію між інтерфейсом користувача та базами даних. На ньому розміщені основні функції додатка, такі як аутентифікація користувачів, обробка транзакцій, управління платежами, підписка на послуги тощо;

– API та інтеграційні сервіси для інтеграції з іншими системами, такими як платіжні шлюзи, сторонні сервіси або системи відкритого банкінгу.

3. База даних (Data Layer):

– база даних клієнтів і транзакцій. На цьому рівні зберігаються всі дані, включаючи особисту інформацію клієнтів, історію транзакцій, налаштування користувачів та інші важливі дані. Бази даних мають бути захищені за допомогою шифрування та доступ до них повинен бути суворо обмежений;

– система управління базами даних (DBMS) - SQL або NoSQL, в залежності від обсягів та специфіки даних.

4. Безпековий рівень:

– аутентифікація та авторизація - використання багатофакторної аутентифікації (MFA) для доступу користувачів, зокрема за допомогою біометрії, одноразових паролів або PIN-кодів, щоб підвищити рівень безпеки;

– шифрування даних;

– модулі для виявлення шахрайства - алгоритми та системи моніторингу, які автоматично аналізують транзакції для виявлення підозрілої активності.

5. Інтеграційний рівень:

– API для сторонніх інтеграцій;

– системи обміну даними. Використовуються інструменти обміну даними - XML, JSON або протоколи RESTful для забезпечення

взаємодії між різними системами в реальному часі.

6. Аналітичний та звітний рівень:

– системи збору та аналізу даних для збирання інформації про користувачів, транзакції, поведінкові дані та інші метрики, які використовуються для покращення обслуговування клієнтів, підвищення рівня безпеки, а також для формування статистичних звітів;

– інструменти для прийняття рішень. Аналітичні системи та модулі прогнозування допомагають банку приймати обґрунтовані рішення на основі зібраних даних (наприклад, для аналізу ризиків, прогнозування відтоку клієнтів та сегментації).

7. Підтримка та служба обслуговування (Support Layer):

– чат-боти та система звернень - онлайн-консультанти для забезпечення швидкої допомоги клієнтам;

– сповіщення та повідомлення. Додаток надсилає користувачам повідомлення про важливі події, такі як транзакції, зміни залишку на рахунку, сповіщення про підозрілу активність тощо.

Ця багаторівнева структура банківських додатків забезпечує високий рівень безпеки, функціональності та адаптивності до потреб клієнтів, а також дозволяє банкам легко масштабувати додаток, впроваджувати нові функції та підтримувати інтеграцію з іншими системами [5, 6].

1.4. Безпека банківських додатків

Безпека банківських додатків важлива складова їх розробки та експлуатації, оскільки вони обробляють конфіденційну інформацію та фінансові дані користувачів. Забезпечення безпеки — це безперервний процес, що включає впровадження нових технологій і методів захисту, а також адаптацію до сучасних загроз. Це дозволяє підтримувати високий рівень довіри користувачів і захищає їх фінансові та особисті дані [6, 7].

Для забезпечення високого рівня захисту застосовуються різні методи,

технології та практики. Основні компоненти системи безпеки банківських додатків включають:

1. Шифрування даних:

– шифрування на рівні передачі даних (дані, що передаються між додатком та сервером, шифруються за допомогою SSL/TLS протоколів, що запобігає перехопленню та зміні даних у процесі передачі);

– шифрування даних у базах даних (дані, що зберігаються в додатку або на сервері, шифруються за допомогою сучасних алгоритмів (наприклад, AES-256), що забезпечує їх безпеку навіть у разі компрометації серверу);

– шифрування локальних даних (дані, що зберігаються на пристрої користувача (наприклад, кеш або тимчасові файли), також шифруються для зменшення ризику їх витоку).

2. Аутентифікація та авторизація:

– багатофакторна аутентифікація (MFA) (для входу в додаток або підтвердження важливих операцій користувачам необхідно пройти декілька рівнів перевірки, наприклад, поєднання пароля, біометричної інформації (відбиток пальця або розпізнавання обличчя) та одноразового коду, що надходить через SMS або електронну пошту);

– система авторизації доступу (користувачі отримують доступ лише до інформації та функцій, які відповідають їх ролям і рівням доступу, що обмежує можливість компрометації системи у разі зламу облікового запису).

3. Захист від шкідливого програмного забезпечення:

– антивірусні системи та системи захисту від шкідливого програмного забезпечення (банківські додатки регулярно перевіряються на наявність шкідливого коду. Крім того, вбудовуються засоби захисту від модифікації коду або сторонніх втручань);

– контроль середовища виконання (виявлення потенційної загрози спроб запуску на зламаних пристроях).

4. Виявлення та запобігання шахрайству:

– аналіз поведінки користувачів (використання алгоритмів машинного навчання для аналізу дій користувача (місцезнаходження, час, послідовність дій), що дозволяє виявити аномалії, які можуть вказувати на шахрайство);

– системи попередження про підозрілі операції, тобто надсилання сповіщень користувачам про незвичні транзакції або підозрілу активність;

– ліміти на транзакції для нових акаунтів, щоб зменшити ризик від великих шахрайських транзакцій.

5. Захист API та інтеграцій з іншими системами:

– безпечна інтеграція API – це використання стандартів безпеки при розробці API, захист відкритих інтерфейсів програмування (API) від несанкціонованого доступу через використання токенів, шифрування та контролю доступу;

– OAuth та JWT для автентифікації API-запитів, це забезпечує надійний спосіб контролю доступу до API, що важливо для інтеграції з сторонніми сервісами;

– блокування доступу до підозрілих IP-адрес - впровадження систем, які автоматично блокують запити з підозрілих або відомих шкідливих IP-адресів.

6. Захист від мережевих атак:

– виявлення та запобігання DDoS-атак - використання технологій для виявлення та блокування DDoS-атак, щоб запобігти перевантаженню системи та порушенню доступності додатку;

– впровадження анти-DDoS технологій - використання рішень для виявлення і запобігання DDoS-атак, що допомагає швидко реагувати на аномальні обсяги трафіку та захищати систему від перевантажень;

– захист мережевого трафіку - застосування міжмережевих екранів та систем захисту, що дозволяє відслідковувати та аналізувати

мережевий трафік на наявність підозрілих дій;

– інструменти для виявлення та запобігання мережевих атак (IDS/IPS) - IDS виявляють аномалії в мережевому трафіку та діях користувачів, попереджаючи про можливі загрози, а IPS автоматично блокують загрози, якщо вони виявлені, запобігаючи потенційним зломам та витокам даних.

7. Оновлення та патчі:

– регулярні оновлення безпеки (для закриття вразливостей і виправлення помилок),

– використання сучасних бібліотек і стандартів (для уникнення відомих вразливостей).

8. Моніторинг та журнали подій:

– журнали аудиту (дії користувачів та системи записуються в спеціальні журнали, які допомагають виявити підозрілу активність та аналізувати потенційні інциденти);

– системи моніторингу безпеки (банки впроваджують системи моніторингу, що аналізують поведінку користувачів для виявлення нетипових або підозрілих дій, таких як незвичні транзакції або спроби доступу з нових пристроїв або локацій).

9. Управління сесіями:

– автоматичний вихід із системи (якщо користувач неактивний протягом певного часу, система автоматично завершує сесію);

– токени сесій з обмеженим терміном дії (для кожної сесії видається унікальний токен з обмеженим терміном дії).

10. Тестування безпеки та оцінка вразливостей:

– пенетраційне тестування (регулярне проведення тестів на проникнення дозволяє банкам виявити слабкі місця в додатку та вжити відповідні заходи);

– аналіз коду на вразливості (для виявлення можливих слабких місць у коді додатка, таких як SQL-ін'єкції, XSS тощо).

11. Відновлення після збоїв і резервне копіювання:

– резервне копіювання даних (для відновлення в разі збоїв чи втрати даних);

– план дій на випадок інцидентів (відновлення даних, перевірку безпеки та інформування користувачів у разі компрометації).

Ці заходи спрямовані на забезпечення комплексного захисту банківських додатків від різноманітних загроз, зокрема кібератак, шахрайства та витоку даних.

1.5. Типи загроз для банківських додатків

Існує кілька типів загроз для банківських додатків, які можуть поставити під загрозу конфіденційність, цілісність та доступність даних користувачів [8]. Банківські додатки стикаються з наступними видами загроз:

1. Фішинг (Phishing) - атаки спрямовані на отримання конфіденційної інформації (логінів, паролів, номерів кредитних карток) шляхом обману користувачів. Зловмисники можуть створити фальшивий вебсайт або мобільний додаток, що імітує офіційний банківський додаток, а також обман через підроблені електронні листи або SMS.
2. Шкідливе програмне забезпечення (Malware):

a) троянські програми - спеціально розроблені програми, які проникають на пристрій користувача, часто під виглядом законного програмного забезпечення та надають зловмисникам доступ до банківської інформації;

b) кейлогери - програми, що записують натискання клавіш, дозволяючи зловмисникам отримати дані для входу до банківських додатків;

c) фінансові трояни - це спеціальний тип троянського програмного забезпечення, який призначений для крадіжки фінансової інформації, включаючи номери карток та PIN-коди.

3. Атаки на мережевий рівень:

a) атаки типу “людина посередині” (Man-in-the-Middle) - перехоплення або зміна даних, що передаються між користувачем та сервером банку, через небезпечні Wi-Fi мережі або недостатньо захищені мережеві канали;

b) DDoS-атаки (розподілені атаки на відмову в обслуговуванні) - спрямовані на виведення з ладу серверів або банківських систем шляхом перевантаження їх великим обсягом запитів. Це може зробити систему недоступною для користувачів, що створює негативний досвід і перешкоджає доступу до послуг;

c) DNS-спуфінг - перенаправлення користувача на фальшивий сайт банку через маніпуляцію з DNS-записами.

4. Атаки на веб- та мобільні додатки

a) SQL-ін’єкції - впровадження шкідливого SQL-код у поля вводу на сайті або додатку, щоб отримати доступ до бази даних банку;

b) XSS (міжсайтовий скриптинг) - вставлення шкідливих скриптів у веб-сторінки, що можуть викрасти дані користувачів або обійти заходи безпеки;

c) зворотна інженерія мобільних додатків - декодування або маніпулювання кодом додатку з метою виявлення вразливостей або викрадення даних.

5. Атаки на рівні API, через вразливі точки в API можуть призвести до витоку даних, маніпуляцій з транзакціями або доступу до конфіденційної інформації користувачів.

6. Шахрайські транзакції - використання крадених облікових записів для проведення шахрайських операцій, таких як несанкціоновані перекази або оплата рахунків.

7. Атаки методом грубої сили (Brute Force Attacks) - використання програми для автоматизованого підбору паролів до облікових записів користувачів, що небезпечно для акаунтів, які мають слабкі або загальноприйняті паролі.

8. Експлойти нульового дня (Zero-Day Attacks) - використання вразливостей в програмному забезпеченні, які ще не були виявлені або виправлені розробниками. Такі атаки можуть бути надзвичайно небезпечними, оскільки у банку може не бути швидких рішень для захисту від таких загроз.

9. Зловмисне використання ботів для збирання інформації про рахунки, проведення несанкціонованих транзакцій або створення великої кількості запитів для зниження продуктивності банківських додатків.

10. Атаки на технології захисту:

a) злом застарілих алгоритмів шифрування, які можна розшифрувати за допомогою сучасних технологій;

b) перехоплення даних при слабкому шифруванні - використання недостатньо надійних шифрувальних алгоритмів для захисту даних при передачі між сервером і клієнтським додатком.

11. Відсутність безпеки на етапі розробки - недостатнє тестування під час розробки, відсутність своєчасного оновлення та виправлення помилок у коді.

12. Внутрішні загрози:

a) зловмисні дії працівників (співробітники з доступом до конфіденційних даних можуть бути залучені до шахрайства або крадіжки даних);

b) необережне поводження з даними (ненавмисне розголошення конфіденційної інформації через необережні дії працівників або недостатній рівень навчання щодо безпеки).

13. Атаки на біометричні дані – спроба обійти біометричні методи автентифікації, наприклад, шляхом обману сенсорів або використання фальшивих відбитків пальців чи фотографій обличчя для входу в додаток.

Щоб захиститися від загроз, банки застосовують комплексний підхід до безпеки, який включає регулярне оновлення програмного забезпечення, використання передових методів шифрування, багатофакторну

автентифікацію, навчання користувачів, а також інструменти для виявлення та запобігання загрозам у реальному часі. Це забезпечує надійний захист додатків від сучасних загроз та зміцнює довіру користувачів до фінансових послуг банку [8, 9].

Основні загрози для банківських додатків та підходи, які допомагають захистити додатки і мінімізувати потенційні ризики наведені в таблиці 1.1.

Таблиця 1.1.

Типи загроз для банківських додатків та шляхи вирішення

Тип загрози	Опис загрози	Шляхи вирішення
Шкідливе програмне забезпечення	Банківські трояни, кейлогери, віруси, що викрадають дані або порушують роботу додатка.	Антивірусний захист, регулярні оновлення систем, обмеження доступу до програм з неперевірених джерел, використання програм для сканування на віруси і шкідливе ПЗ.
Фішинг та соціальна інженерія	Обманом отримані конфіденційні дані через підроблені сайти, електронні листи, телефонні дзвінки тощо.	Впровадження багатофакторної аутентифікації (MFA), навчання користувачів розпізнаванню фішингових атак, регулярні перевірки підозрілої активності.
Атаки на мережевий рівень	Перехоплення трафіку між користувачем і сервером, DDoS-атаки, DNS-спуфінг для перенаправлення на фальшиві сайти.	Використання шифрування TLS/SSL, фільтри для захисту від DDoS, моніторинг мережевого трафіку, регулярні оновлення DNS записів, впровадження системи попередження про загрози на мережевому рівні.

Продовження таблиці 1.1.

Тип загрози	Опис загрози	Шляхи вирішення
SQL-ін’єкції та XSS-атаки	Впровадження шкідливого коду в бази даних або на вебсторінки, що дозволяє викрадати дані.	Валідація введених даних, використання підготовлених запитів, регулярне тестування на проникнення, впровадження WAF для захисту від атак на вебдодатки.
API-атаки	Використання уразливих API для доступу до даних або проведення несанкціонованих дій.	Впровадження OAuth, JSON Web Tokens (JWT) для захисту API, регулярний моніторинг API, застосування авторизації та аутентифікації для API-запиту.
Внутрішні загрози	Співробітники можуть викрасти або ненавмисно розголосити конфіденційні дані.	Контроль доступу за ролями, логування дій співробітників, регулярні перевірки та моніторинг доступу до конфіденційних даних, програми навчання з безпеки.
Незахищені пристрої користувачів	Вразливості на пристроях користувачів (смартфони, ПК), що можуть дати доступ до облікових записів.	Застосування MFA, постійне оновлення додатків, використання антивірусного ПЗ, обмеження доступу для ненадійних пристроїв, поради користувачам щодо безпечного використання.
Злом або компрометація шифрування	Використання застарілих або слабких алгоритмів шифрування, що можуть бути зламані.	Використання сучасних алгоритмів шифрування (AES-256), постійне оновлення системи шифрування, обмеження доступу до ключів шифрування, впровадження практик безпечного управління ключами.

Продовження таблиці 1.1.

Тип загрози	Опис загрози	Шляхи вирішення
Шахрайські транзакції та аномальна активність	Несанкціоновані транзакції, що свідчать про шахрайство або злом облікового запису.	Використання алгоритмів штучного інтелекту для виявлення аномальної активності, автоматичне блокування підозрілих транзакцій, оповіщення користувачів про незвичайну активність, запровадження лімітів на транзакції для нових облікових записів.
Недостатня безпека на етапі розробки	Відсутність безпеки на всіх етапах розробки призводить до випуску додатків з вразливостями.	Інтеграція безпеки на всіх етапах життєвого циклу розробки (SDLC), регулярне тестування на безпеку, використання безпечних практик кодування, навчання розробників принципам безпеки, проведення аналізу коду та тестів на проникнення.

Сучасні банківські додатки повинні забезпечувати багаторівневий захист, використовуючи надійні криптографічні методи, багатофакторну автентифікацію та регулярне оновлення програмного забезпечення для усунення вразливостей.

Комплексні заходи допомагають мінімізувати ризики, пов'язані з кіберзагрозами, забезпечують стабільну та надійну роботу банківських додатків і підвищують рівень довіри клієнтів до фінансових послуг.

РОЗДІЛ 2
АНАЛІЗ СИМЕТРИЧНИХ МЕТОДІВ КРИПТОГРАФІЇ

2.1. Характеристика криптографічної системи

Криптографічна система — це сукупність методів, протоколів і алгоритмів, що використовуються для забезпечення конфіденційності, цілісності та автентичності даних. Вона працює за принципами шифрування і дешифрування, захищає інформацію від несанкціонованого доступу та забезпечує її цілісність і захищеність під час передачі або зберігання [10, 11].

Основні цілі криптографічної системи представлені на рис. 2.1.

Рисунок 2.1. – Цілі криптографічної системи

Ці цілі разом створюють надійний захист для інформації, мінімізуючи ризики, пов’язані з порушенням безпеки. Криптографічний захист є важливим для банківських додатків та інших систем, де захист даних має першочергове значення [11, 12].

Криптографічні системи широко застосовуються в різних сферах:

– банківські та фінансові транзакції для захисту даних клієнтів і запобігання шахрайству;

– інтернет-комунікації для захисту конфіденційності та забезпечення захищених з'єднань;

– електронний документообіг для забезпечення автентичності та цілісності документів;

– захист даних у хмарних сервісах для безпечного зберігання інформації.

Основні компоненти криптографічної системи:

– алгоритми шифрування та дешифрування. Алгоритми шифрування перетворюють вихідні дані (текст) в шифрований вигляд, який виглядає як випадковий набір символів. Дешифрування виконує зворотне перетворення, щоб відновити початкові дані;

– ключі шифрування - це секретні або публічні значення, які використовуються для процесу шифрування та дешифрування. Тип ключів і спосіб їх використання залежать від типу криптографічної системи (симетричної чи асиметричної);

– методи генерації ключів - це процедури для створення надійних і унікальних ключів, які гарантують захищеність даних;

– протоколи захисту даних - це стандарти й інструкції, що регламентують використання криптографічних методів для забезпечення безпеки, наприклад, протоколи SSL/TLS для захищених інтернет-з’єднань.

Типи криптографічних систем:

1. Симетричні криптосистеми використовують один ключ для шифрування та дешифрування даних. Вони високошвидкісні, підходять для великих обсягів даних, але складність полягає у передачі ключів, оскільки обидві сторони повинні мати доступ до того самого ключа.

2. Асиметричні криптосистеми використовують пару ключів: відкритий ключ для шифрування і закритий ключ для дешифрування. Вони

підходять для безпечного обміну даними між користувачами, які не мають попереднього спільного ключа, але повільніші в порівнянні з симетричними, тому асиметричні методи частіше використовуються для шифрування ключів, а не самих даних.

3. Гібридні криптосистеми поєднують симетричну та асиметричну криптографію, щоб скористатися перевагами обох методів. Зазвичай, в асиметричній системі шифрується тільки ключ для симетричного шифрування, а вже самі дані шифруються симетричним методом Використовуються в багатьох сучасних системах безпеки, включаючи SSL/TLS, PGP.

2.2. Криптографічні методи і технології

Криптографія — це наука про методи захисту інформації шляхом її перетворення в таку форму, яка робить її недоступною для несанкціонованого доступу. Вона включає в себе різні техніки шифрування, створення та перевірки цифрових підписів, а також засоби забезпечення цілісності, автентичності та конфіденційності даних [12, 13].

Шифрування (Encryption) — процес перетворення інформації у формат, який неможливо прочитати без відповідного ключа [13].

Хешування (Hashing) — процес перетворення даних у фіксовану довжину представлення (хеш), яке використовується для перевірки цілісності та створення цифрових підписів [13].

Цифрові підписи (Digital Signatures) — використовуються для підтвердження автентичності та цілісності повідомлень або документів [13].

Протоколи аутентифікації та авторизації — застосовуються для перевірки особи або системи, що намагається отримати доступ до певних ресурсів.

Криптографічні методи — це різні техніки, які використовуються для забезпечення конфіденційності, цілісності, автентичності та неможливості

відмови від даних. Вони охоплюють широкий спектр алгоритмів і підходів, які застосовуються для захисту інформації [13].

Основні криптографічні методи:

– симетричне шифрування (Symmetric encryption) — використовується один і той самий ключ для шифрування та дешифрування. Основний принцип — доступ до інформації має лише той, хто має ключ. Найбільш популярні алгоритми: AES**,** DES, 3DES [14, 15];

– асиметричне шифрування (Asymmetric encryption) — використовуються два різні ключі: публічний (для шифрування) і приватний (для дешифрування). Це дозволяє здійснювати захищену комунікацію без необхідності обміну секретними ключами. Приклад**:** RSA**,** ECC [15, 16];

– хешування (Hashing) — це процес перетворення вхідних даних (наприклад, повідомлення або файлу) у фіксовану довжину хеш (скорочену репрезентацію даних). Хеш-функції застосовуються для створення унікальних представлень даних, які використовуються для перевірки їх цілісності. Приклад: SHA**,** MD5, RIPEMD [15, 16];

– цифрові підписи (Digital Signatures) — використовують асиметричне шифрування для підписання та перевірки автентичності електронних повідомлень або документів. Підпис дозволяє перевірити, чи не було змінено повідомлення після його підписання і чи справжній підписувач. Застосовується в електронному документообігу, фінансових транзакціях і комунікації. Приклад: RSA**,** DSA, ECDSA**.**

Криптографічні технології — це набір протоколів, алгоритмів та інструментів, які забезпечують безпеку даних, автентичність, конфіденційність та цілісність інформації у цифровому світі [17].

Основні криптографічні технології:

– TLS/SSL (Transport Layer Security / Secure Sockets Layer) — протоколи для захисту передачі даних між клієнтом і сервером, які використовують як симетричне, так і асиметричне шифрування для захисту даних під час передачі, що захищає дані від перехоплення третіми

сторонами. TLS — сучасна версія протоколу, що прийшла на заміну SSL. Зазвичай використовує гібридне шифрування, яке забезпечує як безпеку, так і ефективність передачі даних;

– PKI (Public Key Infrastructure) — інфраструктура відкритих ключів. Це система для керування публічними та приватними ключами, цифровими сертифікатами та їх сертифікацією, також для створення, розповсюдження та перевірки цифрових сертифікатів і ключів. PKI дозволяє організаціям використовувати цифрові сертифікати для автентифікації користувачів, шифрування інформації та забезпечення цілісності даних;

– блокчейн (Blockchain) — технологія, що використовує криптографічні методи для створення незмінних записів (блоків) транзакцій. Дані зберігаються у вигляді послідовності блоків, кожен блок містить хеш попереднього блоку, що забезпечує цілісність і безпеку даних. Застосовується у криптовалютах, децентралізованих фінансах (DeFi), управлінні ланцюгом постачань та інших сферах;

– двофакторна аутентифікація (2FA) — метод захисту, що використовує два фактори для підтвердження особи користувача (наприклад, пароль і смс-код). Забезпечує підвищений рівень захисту доступу до облікових записів, банківських додатків та інших критичних систем;

– PGP (Pretty Good Privacy) - технологія, яка використовується для шифрування електронної пошти та файлів**.** Вона використовує комбінацію симетричного та асиметричного шифрування для захисту даних. Застосовується для забезпечення конфіденційності та аутентифікації в електронній пошті та обміні повідомленнями;

– HMAC (Hash-Based Message Authentication Code) - це метод автентифікації повідомлень, який використовує хеш-функцію разом з секретним ключем для забезпечення цілісності та автентичності даних. Застосовується у мережевих протоколах (таких як TLS) для захисту переданих даних від змін;

– HSM (Hardware Security Module) технології апаратного

шифрування - це пристрої, які забезпечують захист та управління криптографічними ключами та виконання криптографічних операцій на апаратному рівні. Застосовуються в банківських та державних установах, де є високі вимоги до безпеки зберігання ключів.

2.3. Методи шифрування

Методи шифрування є ключовими для забезпечення безпеки інформації в криптографії. Вони використовуються для перетворення відкритих даних у захищену форму, яка може бути прочитана лише після дешифрування. Основні методи шифрування можна розділити на симетричне та асиметричне шифрування [16, 17].

У симетричному шифруванні використовується один і той самий ключ для шифрування та дешифрування. Обидві сторони, які обмінюються інформацією, повинні мати однаковий ключ для доступу до зашифрованих даних, що забезпечує швидкість, але вимагає надійного способу передачі ключа. Такий метод ефективний з точки зору швидкості, особливо при роботі з великими обсягами даних, але має і певні недоліки, головний з яких — передача ключа. Для того щоб уникнути перехоплення, сторони повинні мати безпечний спосіб обміну ключем, що складно забезпечити, особливо у відкритих мережах [17, 18].

Схема симетричного шифрування представлена на рисунку 2.2.

Відправник і одержувач отримують спільний згенерований ключ, який використовуватимуть для шифрування і дешифрування даних.

Відправник використовує симетричний алгоритм шифрування (наприклад, AES) і застосовує спільний ключ для перетворення відкритого тексту в шифротекст. Результат шифрування — шифротекст, що не має зв'язку з вихідним відкритим текстом і є непридатним для прочитання без ключа. Відправник надсилає зашифроване повідомлення (шифротекст) одержувачу через відкритий або захищений канал зв’язку.

Одержувач, маючи спільний ключ, отримує шифротекст і використовує цей ключ для дешифрування. Дешифрування здійснюється за тим самим алгоритмом, але в зворотному напрямку, повертаючи шифротекст у вихідний відкритий текст. Результат: Одержувач отримує вихідний текст у його первісній формі.

Рисунок 2.2 – Схема симетричного шифрування

Основні алгоритми симетричного шифрування:

- DES (Data Encryption Standard) — алгоритм, який застосовувався раніше, але широко відомий алгоритм, який використовує 56-бітний ключ для шифрування. Сьогодні його вважають незахищеним через короткий розмір ключа [19];
- 3DES (Triple DES) — модифікація DES, що використовує три різних ключі для шифрування даних, що забезпечує більшу безпеку [20];
- AES (Advanced Encryption Standard) — сучасний стандарт шифрування, який є на даний момент одним з найнадійніших для захисту даних, з різними розмірами ключа: 128, 192 і 256 біт [21];
- RC4 (Rivest Cipher 4) — поточний шифр, що працює за принципом потоку бітів. Його використовують у деяких старих протоколах, таких як SSL/TLS, але через уразливості він більше не вважається безпечним [22];
- Blowfish та Twofish — шифри, які використовують змінну довжину ключа до 448 біт, забезпечуючи надійне шифрування і високу швидкість [23,

24].

Переваги симетричного шифрування - це висока швидкість шифрування і дешифрування та простота реалізації для захисту великих обсягів даних, але виникає проблема з безпечною передачею ключа між сторонами.

Симетричні криптосистеми застосовуються там, де необхідне швидке шифрування, особливо для великих обсягів даних:

– захист даних у мережах (для забезпечення конфіденційності даних у VPN, мережевих протоколах, таких як SSL/TLS);

– банківські транзакції (для захисту даних, що передаються між банком і клієнтом, забезпечення конфіденційності та захисту від несанкціонованого доступу);

– шифрування файлів і баз даних (для захисту конфіденційних даних під час їх зберігання).

Асиметричне шифрування — це метод криптографії, що використовує пару ключів: публічний ключ для шифрування даних і приватний ключ для їх розшифрування. Відмінністю цього методу є те, що публічний ключ доступний усім, і він не може бути використаний для розшифровки, а лише для шифрування. Приватний ключ, який зберігається в таємниці, дозволяє розшифрувати дані, зашифровані публічним ключем [25].

Схема асиметричного шифрування представлена на рисунку 2.3.

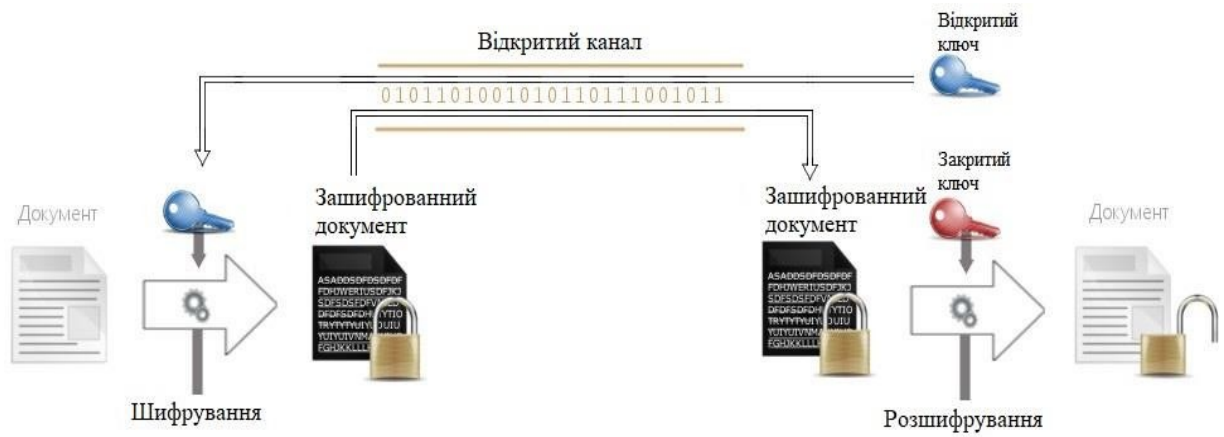

Рисунок 2.3. – Схема асиметричного шифрування

Генерується пара ключів: публічний та приватний. Публічний ключ передається відкрито, а приватний зберігається у таємниці власником.

Відправник надсилає зашифроване повідомлення, використовує публічний ключ отримувача для шифрування тексту. Оскільки публічний ключ є загальнодоступним, будь-хто може його використовувати для шифрування повідомлення. Відправник передає зашифрований текст отримувачу. Навіть якщо зловмисник перехопить повідомлення, без приватного ключа він не зможе його розшифрувати.

Одержувач використовує свій приватний ключ для розшифровки отриманого повідомлення, що було зашифроване публічним ключем. Приватний ключ є унікальним і відомий тільки отримувачу, тому лише він може розшифрувати дані.

Популярні алгоритми асиметричного шифрування:

– RSA (Rivest-Shamir-Adleman) — алгоритм, який широко використовується для шифрування даних і цифрових підписів. Його надійність базується на складності факторизації великих чисел.

– ECC (Elliptic Curve Cryptography) — алгоритм, який використовує еліптичні криві для створення ключів і забезпечує високу безпеку при коротших ключах.

– ElGamal — алгоритм, який використовується для шифрування та цифрових підписів і ґрунтується на складності обчислення дискретного логарифма.

– DSA (Digital Signature Algorithm) — використовується для створення цифрових підписів і працює на основі дискретного логарифмування.

Переваги асиметричного шифрування – це безпечний обмін ключами, оскільки приватний ключ ніколи не передається та використання цифрових підписів для перевірки автентичності.

Недоліки: повільніше в порівнянні з симетричним шифруванням, оскільки потребує більших обчислювальних ресурсів та вимагає складніших

обчислень, що проблематично для ресурсомістких систем.

Асиметричне шифрування широко використовується у цифрових сертифікатах**,** SSL/TLS-з'єднаннях**,** VPN-захисті та інших протоколах для безпечної передачі інформації через Інтернет.

Гібридне шифрування — це метод, що поєднує симетричне і асиметричне шифрування для забезпечення ефективної та безпечної передачі даних. У такій системі асиметричне шифрування використовується для безпечної передачі ключа симетричного шифрування, а симетричне шифрування застосовується для швидкого шифрування самих даних. Це дозволяє отримати переваги обох методів: швидкість симетричного шифрування та безпечний обмін ключами завдяки асиметричному шифруванню.

Роботи гібридного шифрування виконуються за наступною послідовністю. Відправник генерує випадковий сеансовий (симетричний) ключ для шифрування повідомлення, який використовується лише для поточної сесії або передачі даних та шифрує повідомлення, використовуючи симетричний алгоритм шифрування (наприклад, AES).

Сеансовий ключ шифрується за допомогою публічного ключа отримувача через асиметричний алгоритм шифрування (наприклад, RSA). Це гарантує, що лише власник відповідного приватного ключа може розшифрувати сеансовий ключ. Відправник передає одержувачу два компоненти: зашифроване повідомлення та зашифрований сеансовий ключ.

Отримувач використовує свій приватний ключ для розшифрування сеансового ключа, отриманого від відправника. Одержувач використовує сеансовий ключ для розшифрування повідомлення, закодованого симетричним шифруванням.

Переваги гібридного шифрування – це безпечний обмін ключами, тому що асиметричне шифрування захищає сеансовий ключ, дозволяючи безпечно передавати його навіть через незахищені мережі. Також швидке шифрування даних, тому що симетричне шифрування забезпечує швидку обробку великих

обсягів інформації.

Гібридне шифрування використовують:

– SSL/TLS (захист інтернет-з'єднання в протоколах HTTPS), де асиметричне шифрування використовується для обміну сеансовим ключем, а симетричне — для шифрування трафіку;

– PGP (Pretty Good Privacy) — забезпечує безпеку електронної пошти, використовуючи гібридне шифрування для шифрування повідомлень і ключів.

Порівняння основних характеристик симетричного**,** асиметричного та гібридного шифрування наведено в таблиці 2.1.

Таблиця 2.1.

Порівняння основних характеристик методів шифрування

Характеристика	Симетричне шифрування	Асиметричне шифрування	Гібридне шифрування
Тип ключа	Один секретний ключ для шифрування та розшифровки	Пара ключів: публічний для шифрування, приватний для розшифровки	Комбінація симетричного та асиметричного ключів
Швидкість шифрування	Висока	Низька через складні обчислення	Висока (шифрування даних), середня (обмін ключами)
Безпека обміну ключами	Низька (потрібен безпечний канал для передачі ключа)	Висока (публічний ключ можна передавати відкрито)	Висока (асиметричне шифрування захищає обмін ключами)

Продовження таблиці 2.1.

Характеристика	Симетричне шифрування	Асиметричне шифрування	Гібридне шифрування
Складність обчислень	Низька	Висока	Середня (асиметричне шифрування лише для сеансового ключа)
Переваги	Швидке, підходить для шифрування великих обсягів даних	Захищений обмін ключами, підтримка цифрових підписів	Оптимізоване за швидкістю і безпекою
Недоліки	Проблеми з передачею ключа	Повільне, ресурсомістке	Складніша реалізація через комбінацію двох методів
Застосування	Локальне шифрування файлів, захист баз даних	Цифрові підписи, сертифікати, SSL/TLS	Інтернет-протоколи (HTTPS, PGP), передача файлів
Приклади алгоритмів	AES, DES, RC4	RSA, ECC, ElGamal	SSL/TLS, PGP, S/MIME

Кожен метод має своє призначення і використовується в залежності від вимог до безпеки та ефективності конкретної системи.

Симетричне шифрування — швидке і ефективне, але має проблему з безпекою ключа.

Асиметричне шифрування — забезпечує безпеку без обміну секретним

ключем, але повільніше за симетричне.

Гібридні методи — комбінують сильні сторони симетричних та асиметричних методів для досягнення балансу між швидкістю та безпекою.

2.4. Структура систем шифрування

Структура систем шифрування складається з компонентів, що забезпечують захист даних на різних етапах їх обробки та передачі. У загальному вигляді, структура систем шифрування складається з таких елементів:

1. Алгоритми шифрування та розшифровки:

- алгоритм шифрування — процес, який перетворює відкритий текст у шифротекст на основі ключа. Відомі алгоритми :симетричний (AES, DES) та асиметричний (RSA, ECC);
- алгоритм розшифровки — процес, який перетворює шифротекст назад у відкритий текст. Використовує той самий ключ (симетричне шифрування) або парний ключ (асиметричне шифрування).

2. Ключі шифрування:

- секретний ключ (для симетричного шифрування) — єдиний ключ для шифрування та розшифровки;
- публічний і приватний ключі (для асиметричного шифрування) — публічний ключ використовується для шифрування, а приватний для розшифровки. Пара ключів генерується для кожного користувача, який бере участь у передачі даних;
- сеансовий ключ (у гібридному шифруванні) — одноразовий симетричний ключ, який використовується для шифрування даних в одній сесії та захищений асиметричним методом.

3. Керування ключами (Key Management) включає генерацію, розподіл, зберігання, ротацію і знищення ключів. Від керування ключами залежить загальна безпека системи шифрування.

Інфраструктура відкритих ключів (PKI) є прикладом системи керування ключами для асиметричних методів, що забезпечує генерацію та зберігання сертифікатів і ключів.

4. Хешування використовуються для створення унікального відбитка даних, що забезпечує цілісність інформації. Завдяки цьому методу можна перевірити, чи дані не були змінені під час передачі або зберігання, що є особливо важливим для банківських і фінансових систем. Поширені алгоритми хешування включають SHA-256, MD5.

5. Цифровий підпис використовується для підтвердження автентичності відправника та цілісності повідомлення. Підпис генерується за допомогою приватного ключа відправника, і його можна перевірити за допомогою публічного ключа. Використовується в електронних транзакціях, сертифікації і електронній пошті.

6. Сертифікати та інфраструктура відкритих ключів (PKI):

– цифровий сертифікат — електронний документ, що підтверджує особу власника публічного ключа. Він підписується сертифікаційним центром;

– PKI (інфраструктура відкритих ключів) забезпечує управління сертифікатами і ключами, забезпечує довіру між учасниками.

7. Процеси автентифікації та авторизації:

– автентифікація — підтвердження особи користувача перед наданням доступу до даних;

– авторизація — визначає рівень доступу користувача до зашифрованих даних після успішної автентифікації.

Система шифрування є комплексною структурою, що складається з кількох взаємопов'язаних компонентів. Кожен із цих компонентів відіграє критичну роль у забезпеченні конфіденційності, цілісності, автентифікації та доступності даних, що є основними принципами інформаційної безпеки [25].

2.5. Основні алгоритми симетричного шифрування

Алгоритми симетричного шифрування використовують один спільний ключ для шифрування і розшифрування даних. Вони є основою для багатьох захищених систем завдяки своїй ефективності і швидкості обробки. Симетричне шифрування вважається менш ресурсоємним, ніж асиметричне, і підходить для роботи з великими обсягами даних.

Розглянемо шість поширених методів шифрування, які використовуються для захисту конфіденційних даних.

DES (Data Encryption Standard) (рис.2.4.) — це алгоритм симетричного шифрування, який був розроблений у 1970-х роках компанією IBM і пізніше стандартизований урядом США в 1977 році як Федеральний стандарт шифрування (FIPS PUB46). DES широко використовувався до початку 2000-х років, але через свої вразливості, такі як коротка довжина ключа, став застарілим [19].

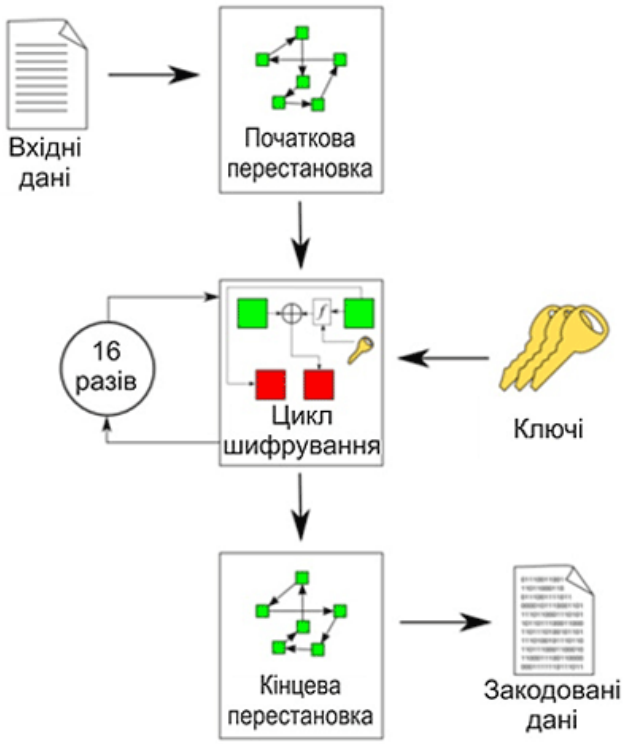

Рисунок 2.4. – Схема шифрування алгоритма DES

DES обробляє дані блоками, тобто дані діляться на фіксовані блоки по 64 біти, і кожен блок шифрується окремо. На відміну від потокових шифрів, блочні шифри працюють із цілими блоками, що дозволяє використовувати різні методи перестановок і замін для кожного блоку. Він використовує 56-бітний ключ, хоча фактично довжина ключа 64 біти, оскільки 8 бітів використовуються для перевірки парності. DES використовує Feistel-структуру, де дані обробляються за допомогою багатьох раундів (16 раундів), у кожному з яких здійснюється змішування та заміни частин даних за допомогою ключа.

Шифрування відбувається через поділ блоку даних на дві половини, де одна половина трансформується в іншу за допомогою функцій заміни та перестановки, і кожен раунд використовує різні частини ключа. За кожен з 16 раундів блоки даних зазнають перестановки, заміни та обробки через ключі, що залежать від початкового секретного ключа. Після виконання всіх раундів блоки об'єднуються та створюють зашифрований текст.

DES відомий своєю вразливістю до атак перебору ключів (brute-force), оскільки довжина ключа лише 56 біт. Через ці вразливості, DES сьогодні вважається незахищеним для використання в сучасних системах. Для посилення безпеки був розроблений варіант 3DES (Triple DES).

3DES (Triple DES) (рис. 2.5.) — це алгоритм симетричного шифрування, який є розширенням DES і використовує його триразове застосування для підвищення безпеки. 3DES використовує три послідовні операції DES для кожного блоку даних: шифрування, розшифрування, потім знову шифрування. В результаті для 3DES потрібно три різних ключі, кожен з яких має 56 біт. Дані спочатку шифруються за допомогою першого ключа, потім результат розшифровується другим ключем, а насамкінець, результат повторно шифрується третім ключем [20].

Підвищена безпека порівняно з DES завдяки потрійному застосуванню. 3DES є значно повільнішим, ніж сучасні алгоритми, такі як AES, оскільки потрійне шифрування вимагає більше ресурсів.

Рисунок 2.5. – Схема шифрування алгоритма 3DES

3DES все ще використовується в деяких застарілих системах і стандартах, але нові системи рекомендується будувати на основі AES, оскільки AES є швидшим, безпечнішим і рекомендованим сучасним стандартом шифрування.

AES (Advanced Encryption Standard) (рис. 2.6.) – це симетричний алгоритм блочного шифрування, який вважається одним із найбільш надійних і широко використовуваних методів шифрування. AES був розроблений для заміни застарілого алгоритму DES [21].

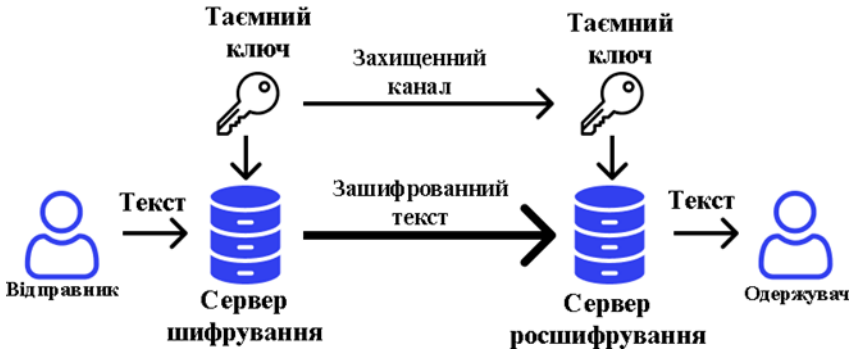

Рисунок 2.6 – Робота алгоритму AES

AES працює з фіксованими блоками даних розміром 128 біт, що робить його особливо придатним для обробки великих обсягів даних. AES підтримує довжини ключів 128, 192 і 256 біт. Довші ключі забезпечують більший рівень безпеки, але можуть впливати на швидкість роботи. Алгоритм шифрування

AES складається з кількох раундів (10 для 128-бітного ключа, 12 для 192-бітного і 14 для 256-бітного). Кожен раунд містить кілька операцій, включаючи заміну байтів, перестановку, зсув рядків і перемішування стовпців.

AES забезпечує високий рівень безпеки та стійкий до більшості відомих атак, що робить його стандартом для використання в урядових, фінансових і військових організаціях. Вважається, що для атаки «грубої сили» (перебору всіх можливих ключів) на AES-256 потрібно більше часу.

AES є ефективним і швидким навіть на обмежених ресурсах, що дозволяє використовувати його як на високопродуктивних серверах, так і на мобільних пристроях. AES використовується у протоколах безпеки, таких як SSL/TLS для захисту інтернет-з'єднань, WPA2 для безпеки Wi-Fi, а також у багатьох інших сферах, включаючи шифрування жорстких дисків, VPN-з'єднань та месенджерів.

Переваги: висока швидкість, стійкість до атак, підтримка широкого спектра застосувань. Недоліки: AES є симетричним алгоритмом, тому він потребує безпечного каналу для передачі ключа між сторонами.

Завдяки поєднанню надійності та швидкості AES залишається найпопулярнішим методом шифрування, який застосовується в різних системах безпеки по всьому світу.

Blowfish — це симетричний алгоритм блочного шифрування, розроблений Брюсом Шнайєром у 1993 році. Він призначений як альтернатива DES [23].

Blowfish працює з блоками даних розміром 64 біти, кожен блок обробляється незалежно. Blowfish підтримує змінну довжину ключа, від 32 до 448 біт, що дає гнучкість в налаштуванні рівня безпеки. Алгоритм Blowfish складається з 16 раундів перетворень, які включають XOR-операції, підстановки (S-блоки) та перестановки (P-блоки). Перед шифруванням кожен блок даних проходить через етап ініціалізації, на якому ключ змішується з P-блоками.

Blowfish є одним із найшвидших алгоритмів шифрування, особливо для програмного забезпечення, що працює на процесорах із обмеженими ресурсами. Він був розроблений для швидкого виконання на 32-бітних процесорах. Blowfish вважається безпечним для використання, якщо використовуються великі ключі. Однак, оскільки розмір блоку становить лише 64 біти, він менш стійкий до атак типу "день народження" порівняно з алгоритмами, що використовують більший блок (наприклад, AES із 128-бітним блоком). З часом його популярність зменшилася через появу більш безпечного та ефективного алгоритму AES. Blowfish є відкритим і не запатентованим алгоритмом, тому він доступний для вільного використання.

Twofish — це симетричний алгоритм блочного шифрування, розроблений Брюсом Шнайєром та його командою у 1998 році як потенційна заміна DES [24].

Twofish використовує блоки даних розміром 128 біт. Підтримує ключі змінної довжини — 128, 192 або 256 біт, що дає високий рівень безпеки. Twofish є досить швидким алгоритмом і оптимізований для програмної реалізації на різних процесорах. Він ефективний і для апаратної реалізації, хоча зазвичай AES вважається більш оптимізованим для сучасного обладнання. Використовує мережу Фейстеля з чотирьох циклів шифрування, що підвищує його стійкість до криптоаналітичних атак. Має стійкість до таких атак, як атака на основі диференціального та лінійного криптоаналізу.

Хоча Twofish не став офіційним стандартом AES, він досі використовується в деяких системах і програмах, особливо для забезпечення довготривалої безпеки даних. Застосовується у відкритому програмному забезпеченні, такому як PGP (Pretty Good Privacy), VeraCrypt, а також для захисту даних в деяких файлових системах.

Twofish використовує складніший розклад ключів, шифруючи дані за 16 раундів незалежно від розміру ключа шифрування. Він також загальнодоступний, як і його попередник Blowfish, але він набагато швидший і може бути застосований як до апаратного, так і до програмного

забезпечення.

RC4 (Rivest Cipher 4) — це потоковий алгоритм симетричного шифрування, розроблений Роном Рівестом у 1987 році. Він був дуже популярним через свою простоту й високу швидкість виконання, але згодом виявилися його вразливості, через що використання RC4 значно скоротилося.

RC4 обробляє дані як потік байтів, а не як блочні дані, тому він не потребує розбиття інформації на блоки. Довжина ключа може варіюватися від 40 до 2048 біт, хоча найчастіше використовуються ключі від 128 до 256 біт. RC4 був дуже популярним у таких протоколах, як: WEP (для захисту Wi-Fi мереж), оскільки ключ був дуже коротким і легко піддавався атаці та SSL/TLS (для забезпечення безпеки в інтернет-з'єднаннях) — використовувався до того, як були знайдені серйозні уразливості. Через виявлені проблеми з випадковістю у вихідному потоці шифрування RC4 більше не вважається безпечним для захисту конфіденційних даних. Застосовувався для шифрування даних у веб-протоколах, але поступово відходить у минуле через уразливості [22].

У банківській сфері симетричні алгоритми використовуються для шифрування транзакцій, зберігання конфіденційних даних клієнтів та захисту внутрішньої інформації. AES є стандартом для багатьох банківських систем завдяки високій стійкості та швидкості обробки.

2.6. Атаки на симетричні криптосистеми

Атаки на симетричні криптосистеми банківських додатків спрямовані на компрометацію секретних ключів, злам шифрованих даних або отримання доступу до конфіденційної інформації. Симетричні криптосистеми є швидкими та ефективними для шифрування великих обсягів даних, однак вони мають певні вразливості. Зловмисники можуть використовувати різні методи для злому таких систем, зокрема криптографічні атаки, атаки на зберігання та передачу ключів, а також атаки з використанням соціальної

інженерії [23-25].

Основні типи атак на симетричні криптосистеми:

1) Атака грубої сили (Brute Force Attack):

- зловмисник намагається підібрати ключ шляхом перебору всіх можливих комбінацій;
- симетричні криптосистеми з короткими ключами (наприклад, 56-бітний DES) можуть бути вразливими до таких атак, оскільки з часом обчислювальна потужність зросла, що робить можливим перебір комбінацій.

2) Криптоаналіз:

- це набір методів для математичного аналізу шифрів з метою знайти спосіб зламати їх, не перебираючи всі ключі;
- до основних видів криптоаналітичних атак належать:
  - лінійний криптоаналіз: аналізує статистичні закономірності шифротексту для виявлення залежностей між ключем та шифротекстом;
  - диференційний криптоаналіз: досліджує, як невеликі зміни в тексті впливають на шифротекст, що дозволяє знаходити інформацію про ключ.
- атаки лінійного і диференційного криптоаналізу особливо небезпечні для певних симетричних алгоритмів, якщо вони мають слабкі місця в структурі шифрування.

3) Атака на основі обраного шифротексту або відкритого тексту:

- Зловмисник має можливість отримати зашифрований текст для обраного відкритого тексту (або навпаки) та використовує цю інформацію для визначення ключа.
- Це може бути реалізовано, наприклад, якщо в додатку є відома структура або формат даних (наприклад, завжди однакові заголовки повідомлень або статичні поля).

4) Атаки на систему зберігання ключів:

- Зломи можуть бути спрямовані на фізичні або програмні сховища,

де зберігаються симетричні ключі.

- Якщо ключі не захищені належним чином, їх можна викрасти або перехопити, отримавши доступ до конфіденційних даних.

5) Атака повторного використання ключів (Key Reuse Attack):

- Симетричні криптосистеми можуть бути вразливими, якщо один і той самий ключ використовується для шифрування різних даних, особливо якщо дані мають схожі шаблони.
- Це може дозволити зловмиснику здійснити аналіз схожих фрагментів шифротекстів для визначення структури даних та зламати ключ.

6) Атака з допомогою аналізу часу (Timing Attack):

- Використання часових затримок в обчислювальних операціях шифрування і дешифрування для отримання інформації про ключ.
- Такі атаки часто застосовуються для зламу систем, де шифрування та дешифрування виконуються на апаратному рівні.

7) Атака на з’єднання або канал передачі даних:

- Перехоплення трафіку між клієнтами і банківським сервером у спробі виявити симетричний ключ.
- Зловмисник може спробувати модифікувати або повторно відправити пакети, щоб отримати доступ до інформації або змусити систему видати більше даних про ключ.
- Атака соціальної інженерії. Хоч вона і не пов’язана напряму з криптографічним зламом, але часто зловмисники використовують маніпуляції або шахрайство для отримання ключів чи паролів від працівників банку.

Методи захисту від атак на симетричні криптосистеми:

1) Використання довгих і складних ключів: чим довший ключ, тим складніше його підібрати за допомогою грубої сили. Наприклад, AES з ключем 256 біт вважається стійким до атак грубої сили.

2) Регулярна ротація ключів: часта зміна ключів знижує ризик їхньої компрометації.

3) Захист сховищ ключів: використання захищених модулів для зберігання ключів (наприклад, HSM — апаратні модулі безпеки).

4) Впровадження багаторівневого захисту: застосування декількох рівнів шифрування, щоб навіть у разі компрометації одного з рівнів загроза була мінімальною.

5) Використання випадкових значень (сольових значень): додавання випадкових даних до тексту перед шифруванням для зменшення можливості реалізації атак на основі повторного використання ключа.

6) Моніторинг аномальних дій: виявлення незвичайних запитів або підозрілої активності, яка може свідчити про кібератаку або спробу підібрати ключ.

7) Освітні програми для співробітників: навчання співробітників банку методам захисту від соціальної інженерії.

Захист симетричних криптосистем банківських додатків потребує комплексного підходу, що включає як вибір надійного алгоритму шифрування, так і дотримання кращих практик захисту ключів, підвищення обізнаності працівників та використання додаткових засобів безпеки. Ефективний захист дозволяє забезпечити безпеку фінансових даних клієнтів та зменшити ризик компрометації інформації.

РОЗДІЛ 3
ДОСЛІДЖЕННЯ ЗАСТОСУВАННЯ ТРАДИЦІЙНИХ СИМЕТРИЧНИХ КРИПТОСИСТЕМ

3.1. Програмна реалізація алгоритму AES

Розглянемо програмну реалізацію AES алгоритма на мові Python, для шифрування та дешифрування даних у банківському додатку. Програма використовує бібліотеку cryptography, яка надає сучасні методи шифрування для оцінки надійності та стійкості алгоритму (рис. 3.1). Цей код демонструє шифрування та розшифрування тексту за допомогою AES у режимі CBC (Cipher Block Chaining).

1) Генерація ключа: функція generate_aes_key генерує випадковий 32-байтний (256-бітний) ключ для AES-256.

2) Шифрування:

- encrypt_aes створює об'єкт шифру AES у режимі CBC і генерує випадковий ініціалізаційний вектор (IV) для забезпечення випадковості шифрування;
- вхідний текст кодується та додається заповнення (padding) для відповідності блоку AES (16 байт);
- повертається об'єднаний результат IV та шифротексту, щоб забезпечити можливість розшифрування.

3) Розшифрування:

- decrypt_aes розділяє IV та шифротекст і ініціалізує AES у режимі CBC з отриманим IV;
- після розшифрування заповнення (padding) видаляється, щоб повернути текст до початкового вигляду.

```python
from Crypto.Random import get_random_bytes
from Crypto.Util.Padding import pad, unpad

# Генерація 256-бітного ключа для AES
def generate_aes_key():
    return get_random_bytes(32)

# Шифрування даних
def encrypt_aes(plaintext, key):
    cipher = AES.new(key, AES.MODE_CBC)  # Використовуємо AES в режимі CBC
    iv = cipher.iv  # Ініціалізаційний вектор
    padded_data = pad(plaintext.encode(), AES.block_size)
    ciphertext = cipher.encrypt(padded_data)
    return iv + ciphertext  # Повертаємо IV разом з шифротекстом

# Розшифрування даних
def decrypt_aes(ciphertext, key):
    iv = ciphertext[:AES.block_size]  # Витягаємо IV з початку шифротексту
    actual_ciphertext = ciphertext[AES.block_size:]
    cipher = AES.new(key, AES.MODE_CBC, iv)
    padded_plaintext = cipher.decrypt(actual_ciphertext)
    return unpad(padded_plaintext, AES.block_size).decode()

# Приклад використання
if __name__ == "__main__":
    key = generate_aes_key()
    plaintext = "Це тестове повідомлення для шифрування AES."
    print("Оригінальний текст:", plaintext)

    # Шифрування
    ciphertext = encrypt_aes(plaintext, key)
    print("Шифротекст:", ciphertext.hex())

    # Розшифрування
    decrypted_text = decrypt_aes(ciphertext, key)
    print("Розшифрований текст:", decrypted_text)
```

Рисунок 3.1 – Програмна реалізація алгоритму AES

```
Output:

Оригінальний текст: Це тестове повідомлення для шифрування AES.
Шифротекст: 6523d5ed219d3002248b5726758496be933d510e339e2e0530a1dd6f904f4e278e4
Розшифрований текст: Це тестове повідомлення для шифрування AES.
```

Рисунок 3.2 – Результати шифрування та дешифрування алгоритмом AES

3.2. Програмна реалізація алгоритму Triple DES

Розглянемо програмну реалізацію алгоритму Triple DES (3DES) на Python, використовуючи бібліотеку pycryptodome, яка надає криптографічні

примітиви для реалізації 3DES, для шифрування та дешифрування даних (рис. 3.3).

```python
from Crypto.Cipher import DES3
from Crypto.Random import get_random_bytes
from Crypto.Util.Padding import pad, unpad

# Генерація 24-бітного ключа для Triple DES
def generate_3des_key():
    while True:
        key = get_random_bytes(24)  # 3DES потребує 24 байти для трьох 8-байтних ключів
        try:
            # Перевірка ключа для сумісності з 3DES
            des3_key = DES3.adjust_key_parity(key)
            return des3_key
        except ValueError:
            # Якщо ключ не підходить, генеруємо новий
            continue

# Шифрування даних
def encrypt_3des(plaintext, key):
    cipher = DES3.new(key, DES3.MODE_CBC)
    iv = cipher.iv  # Ініціалізаційний вектор (IV) для безпеки
    padded_data = pad(plaintext.encode(), DES3.block_size)
    ciphertext = cipher.encrypt(padded_data)
    return iv + ciphertext  # Повертаємо IV разом з шифротекстом

# Розшифрування даних
def decrypt_3des(ciphertext, key):
    iv = ciphertext[:DES3.block_size]  # Витягаємо IV з початку шифротексту
    actual_ciphertext = ciphertext[DES3.block_size:]
    cipher = DES3.new(key, DES3.MODE_CBC, iv)
    padded_plaintext = cipher.decrypt(actual_ciphertext)
    return unpad(padded_plaintext, DES3.block_size).decode()

# Приклад використання
if __name__ == "__main__":
    key = generate_3des_key()
    plaintext = "Це тестове повідомлення для шифрування Triple DES."
    print("Оригінальний текст:", plaintext)

    # Шифрування
    ciphertext = encrypt_3des(plaintext, key)
    print("Шифротекст:", ciphertext.hex())

    # Розшифрування
    decrypted_text = decrypt_3des(ciphertext, key)
    print("Розшифрований текст:", decrypted_text)
```

Рисунок 3.3 – Програмна реалізація алгоритму Triple DES

Output:

```
Оригінальний текст: Це тестове повідомлення для шифрування Triple DES.
Шифротекст: ccf368dbade27fb8bccc26c666d213411726fe8c20a6317cdce348f8ffc5e7d4ba18d646d4
Розшифрований текст: Це тестове повідомлення для шифрування Triple DES.
```

Рисунок 3.4 – Результати шифрування та дешифрування алгоритмом Triple DES

1) Генерація ключа: функція generate_3des_key генерує 24-байтний ключ для 3DES. Triple DES використовує три 8-байтні ключі, об'єднані в один 24-байтний ключ.

2) Шифрування: функція encrypt_3des використовує режим CBC (Cipher Block Chaining) для безпечнішого шифрування. IV (ініціалізаційний вектор) додається на початок шифротексту для забезпечення випадковості.

3) Розшифрування: функція decrypt_3des розділяє IV та шифротекст, щоб правильно ініціалізувати розшифрування. Після отримання розшифрованого тексту він депакетується до вихідного вигляду.

3.3. Програмна реалізація алгоритму Twofish

Алгоритм Twofish є симетричним блочним шифром, що розроблений Брюсом Шнайером та іншими вченими. Він має блок розміру 128 біт і підтримує довжину ключа 128, 192 або 256 біт. Однак, на відміну від AES, Twofish не є стандартом, хоча він претендував на участь у конкурсі AES. Для реалізації цього алгоритму можна використовувати бібліотеку pycryptodome, але вона не підтримує Twofish безпосередньо.

Замість цього, для реалізації Twofish, можна скористатися бібліотекою pycryptodomex або спеціальними бібліотеками, що реалізують Twofish (рис. 3.5).

1) Генерація ключа: функція generate_twofish_key генерує випадковий ключ довжиною 256 біт (32 байти). Twofish підтримує 128, 192 і 256-бітні ключі, але в цьому прикладі використовуємо найбільший доступний (256 біт).

2) Шифрування:

- encrypt_twofish виконує шифрування тексту за допомогою Twofish;
- вхідний текст кодується в байти та доповнюється (padding) до довжини блоку 16 байт, оскільки Twofish працює з блоками по 128 біт (16 байт).

3) Розшифрування: decrypt_twofish розшифровує шифротекст, після чого видаляє padding і повертає початковий текст.

```python
from twofish import Twofish
from Crypto.Random import get_random_bytes
from Crypto.Util.Padding import pad, unpad

# Генерація випадкового ключа для Twofish (256 біт)
def generate_twofish_key():
    # Twofish підтримує 128, 192 або 256 біт
    return get_random_bytes(32)

# Шифрування даних
def encrypt_twofish(plaintext, key):
    cipher = Twofish(key)
    # Twofish використовує блоки по 128 біт (16 байт)
    padded_data = pad(plaintext.encode(), 16)
    ciphertext = cipher.encrypt(padded_data)
    return ciphertext

# Розшифрування даних
def decrypt_twofish(ciphertext, key):
    cipher = Twofish(key)
    padded_plaintext = cipher.decrypt(ciphertext)
    return unpad(padded_plaintext, 16).decode()

# Приклад використання
if __name__ == "__main__":
    key = generate_twofish_key()
    plaintext = "Це тестове повідомлення для шифрування Twofish."
    print("Оригінальний текст:", plaintext)

    # Шифрування
    ciphertext = encrypt_twofish(plaintext, key)
    print("Шифротекст:", ciphertext.hex())

    # Розшифрування
    decrypted_text = decrypt_twofish(ciphertext, key)
    print("Розшифрований текст:", decrypted_text)
```

Рисунок 3.5 – Програмна реалізація алгоритму Twofish

```
Output:

Оригінальний текст: Це тестове повідомлення для шифрування Twofish.
Шифротекст: 64ffb49c0c993d15d81ff4f63e913c58
Розшифрований текст: Це тестове повідомлення для шифрування Twofish.
```

Рисунок 3.6 – Результати шифрування та дешифрування алгоритмом Twofish

Зведена інформація, щодо основних симетричних криптоалгоритмів наведена в табл. 3.1.

Таблиця 3.1

Порівняльна таблиця найбільш популярних симетричних криптоалгоритмів

Алгоритм	Розмір блоку	Розмір ключа	Швидкість	Безпека для банківських додатків	Стійкість до атак	Підтримка в банківських системах	Застосування
AES (Advanced Encryption Standard)	128 біт	128, 192, 256 біт	Висока	Висока	Дуже висока	Широко використовується у банківських додатках, рекомендовано для захисту транзакцій та даних	Стандарт для захисту даних, транзакцій і комунікацій
DES (Data Encryption Standard)	64 біт	56 біт	Повільний	Низька	Вразливий до атак перебору	Не використовується в сучасних банківських системах через слабкість ключа	Використовувався у старих системах, більше не рекомендується

Продовження таблиці 3.1.

3DES (Triple DES)	64 біт	112, 168 біт	Повільний	Середня безпека	Вразливий до атак на швидкість	Раніше використовувався в банківських додатках, зараз застарілий	Був використаний для підвищення безпеки DES, але зараз вже не рекомендується
Twofish	128 біт	128, 192, 256 біт	Висока	Висока	Висока стійкість до атак	Використовується в окремих банківських рішеннях, але менш поширений ніж AES	Альтернатива AES, застосовується у специфічних випадках
Blowfish	64 біт	32-448 біт	Висока	Середня безпека	Стійкий до атак перебору	Використовується в старих банківських додатках, не рекомендується для нових систем	Був використаний для шифрування транзакцій, зараз поступово відходить у минуле
RC4	Потоковий	40-2048 біт	Висока	Низька	Вразливий до криптоаналізу	Більше не використовується в банківських додатках через вразливості	Використовувався в SSL, WEP, але більше не використовується для безпеки

ВИСНОВКИ

Симетричні криптосистеми – це основа багатьох сучасних криптографічних рішень для забезпечення конфіденційності, цілісності та автентичності даних. У банківському секторі захист інформації є критично важливим, адже обробляються фінансові транзакції, персональні дані та конфіденційні відомості клієнтів. У роботі розглянуто основні симетричні криптосистеми, які використовуються для захисту банківських додатків, їхні переваги, недоліки та прикладні аспекти для досягнення високого рівня безпеки.

Симетричні криптосистеми використовують один ключ для шифрування та розшифрування, що робить їх ефективними для шифрування великих обсягів даних. Основні симетричні криптоалгоритми, які знаходять застосування в банківських додатках, включають DES, 3DES, AES, Blowfish та Twofish. Кожен з цих алгоритмів має свої переваги та недоліки, які визначають їх придатність для певних банківських завдань.

DES (Data Encryption Standard). Раніше DES широко використовувався в банківських додатках. Однак через довжину ключа (56 біт) він став вразливим до атак повного перебору. DES поступово витісняється з банківських застосунків, але став основою для багатьох інших алгоритмів, зокрема 3DES.

3DES (Triple DES). Удосконалений алгоритм DES, що шифрує дані тричі з різними ключами, досяг підвищеного рівня безпеки і донині використовується в фінансових транзакціях. 3DES підтримується багатьма банкоматами і POS-терміналами завдяки довгій історії використання у фінансовому секторі.

AES (Advanced Encryption Standard). AES став сучасним стандартом шифрування для банківських додатків завдяки високому рівню безпеки та можливості вибору довжини ключа (128, 192 або 256 біт). Він оптимізований для програмного та апаратного забезпечення і підходить для захисту даних у

мережевих транзакціях, зберігання інформації та мобільних банківських додатків.

Blowfish і Twofish. Алгоритми, що забезпечують високу швидкість та гнучкість завдяки варіативній довжині ключа. Twofish має деякі переваги для апаратної реалізації, але не є настільки широко поширеним, як AES чи 3DES.

RC4. Потоковий шифр, який раніше використовувався в деяких банківських системах для забезпечення швидкої обробки даних, але через відомі вразливості майже витіснений з фінансових додатків.

Основними перевагами симетричних алгоритмів є висока швидкість обробки та ефективність при роботі з великими обсягами даних, що робить їх ідеальними для транзакційних процесів, де час відповіді є важливим фактором. Однак існують і певні обмеження:

- проблема обміну ключами: оскільки для шифрування та розшифрування використовується один ключ, виникає проблема його безпечної передачі між сторонами. У банківських додатках цю проблему вирішують за допомогою гібридних схем, які поєднують симетричне та асиметричне шифрування;
- вимоги до довжини ключа: короткі ключі, як у DES, вразливі до атак перебором, що робить їх неприйнятними для довготривалого захисту конфіденційних даних.
- стійкість до сучасних атак: із зростанням обчислювальної потужності деякі традиційні алгоритми, такі як DES і RC4, стають менш надійними. AES із ключами 256 біт залишається найбільш прийнятним рішенням у банківських додатках завдяки своїй стійкості до криптоаналітичних атак.

У банківських додатках симетричні криптоалгоритми використовуються для шифрування даних під час транзакцій, захисту зберігання даних, автентифікації користувачів та в системах управління доступом. У роботі наведено приклади практичного застосування деяких із них:

- захист транзакцій. 3DES і AES часто використовуються для шифрування даних у POS-терміналах і банкоматах. Ці алгоритми забезпечують захист даних під час обміну між банкоматом та сервером банку, запобігаючи перехопленню та несанкціонованому доступу до конфіденційних даних клієнтів;

- захист зберігання даних. Банківські бази даних, що зберігають інформацію про клієнтів, транзакції та баланси рахунків, часто шифруються з використанням AES-256 для забезпечення довготривалого захисту інформації;

- автентифікація користувачів. Алгоритми симетричного шифрування застосовуються для генерації одноразових паролів (OTP) у процесі двофакторної автентифікації, що значно підвищує безпеку доступу до банківських послуг;

- віртуальні приватні мережі (VPN). VPN на основі AES використовуються для захисту віддаленого доступу до банківських систем співробітниками та партнерами банків, що забезпечує безпечний обмін даними в умовах глобальної мережі.

Для подолання обмежень симетричних алгоритмів банки часто використовують гібридні криптосистеми, поєднуючи симетричне та асиметричне шифрування. У такій системі асиметричний алгоритм (наприклад, RSA або ECC) використовується для безпечного обміну симетричним ключем, який надалі використовується для шифрування даних транзакцій. Це дозволяє зберегти високу швидкість обробки даних, властиву симетричним алгоритмам, і уникнути проблеми передачі ключа.

Отже, традиційні симетричні криптосистеми, зокрема 3DES та AES, продовжують грати важливу роль у захисті банківських додатків. Незважаючи на проблеми з обміном ключами та певні вразливості деяких застарілих алгоритмів, симетричні шифри залишаються ефективним рішенням для захисту конфіденційної інформації у фінансових транзакціях. Завдяки використанню гібридних підходів, банки можуть забезпечити

надійний захист даних, відповідний сучасним стандартам безпеки.

Застосування цих підходів допомагає створити надійну систему безпеки, що захищає банківський додаток від загроз, мінімізує ризики витоку даних і забезпечує безпеку фінансових операцій клієнтів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Тарнавський Ю. А. Технології захисту інформації : підручник для студентів спеціальності 122 «Комп'ютерні науки». – Київ : КПІ ім. Ігоря Сікорського, 2018. – 162 с.

2. Усік П. С., Буравченко К. О. Безпека банківських систем. Навчальний посібник. – Кропивницький: ЦНТУ, 2022. – 194 с.

3. Козіна Г. Л. Криптографія від історії до сучасних стандартів: навчальний посібник. – Запоріжжя: НУ «Запорізька політехніка», 2020. – 192 с.

4. Гребенюк А. М., Рибальченко Л. В. Основи управління інформаційною безпекою: навчальний посібник. – Дніпро: Дніпропетровський державний університет внутрішніх справ, 2020. – 144 с.

5. Закон України «Про захист інформації в інформаційно-телекомунікаційних системах», 1994. [Електроний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/80/94-%D0%B2%D1%80#Text.

6. ДСТУ 3396 0-96 Захист інформації. Технічний захист інформації. Основні положення. [Електроний ресурс]. – Режим доступу: https://tzi.com.ua/downloads/DSTU%203396.0-96.pdf.

7. Закон України «Про основні засади забезпечення кібербезпеки України», 2017. [Електроний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2163-19#Text.

8. Щур Н.О., Покотило О.А. Основи криптології: навч. посібник. – Житомир: Державний університет «Житомирська політехніка», 2021. – 120 с.

9. Горбенко І. Д. Гриненко Т. О. Захист інформації в інформаційно-телекомунікаційних системах: навчальний посібник. Ч.1. Криптографічний захист інформації. – Харків: ХНУРЕ, 2004. – 368 с.

10. Домарєв В.В., Швець В.А., Шестакова В.В. Організаційне забезпечення захисту інформації з обмеженим доступом. Навчальний посібник. – Київ: НАУ, 2006. – 108 с.

11. Зубок М. І. Безпека банківської діяльності: навчальний посібник. – Київ. : КНЕУ, 2002. – 190 с.

12. Хорошко В. О, Чередниченко В. С., Шелест М. Є. Основи інформаційної безпеки. – Київ: ДУІКТ, 2008. – 186 с.

13. Основні поняття криптології, 2023. [Електроний ресурс]. – Режим доступу: https://theorydatasecure.wordpress.com/лекції/лекція-№2/.

14. Даник Ю. Г., Воробієнко П. П., Чернега В.М. Основи кібербезпеки та кібероборони: підручник. – Одеса.: ОНАЗ ім. О.С. Попова, 2019. − 320 с.

15. Гончарова І.П. Кібербезпека в цифровому освітньому середовищі закладів професійної освіти: електронний навчальний курс. – Біла Церква: БІНПО ДЗВО «УМО» НАПН УКРАЇНИ, 2022. – 80 с.

16. Бурячок В. Л., Толубко В. Б., Хорошко В. О., Толюпа С. В. Інформаційна та кібербезпека: соціотехнічний аспект: підручник. – Київ: ДУТ, 2015. – 288 с.

17. Симетричні криптосистеми. [Електроний ресурс]. – Режим доступу: https://wiki.tntu.edu.ua/%D0%A1%D0%B8%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%BD%D1%96_%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8.

18. Шифрування з симетричними ключами. [Електроний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B7_%D1%81%D0%B8%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%BD%D0%B8%D0%BC%D0%B8_%D0%BA%D0%BB%D1%8E%D1%87%D0%B0%D0%BC%D0%B8.

19. Data Encryption Standard. [Електроний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Data_Encryption_Standard.

20. Triple DES. [Електроний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Triple_DES.

21. Advanced Encryption Standard. [Електроний ресурс]. – Режим

доступу: https://uk.wikipedia.org/wiki/Advanced_Encryption_Standard.

22. RC4. [Електроний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/RC4.

23. Blowfish. [Електроний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Blowfish.

24. Twofish. [Електроний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Twofish.

25. Блінцов В. С., Гальчевський Ю. Л. Математичні основи криптології. Навчальний посібник для студентів вищих навчальних закладів. – Миколаїв : Національний університет кораблебудування ім. адмірала Макарова, 2006. – 232 с.