МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ Скарга-Бандурова І.С.
«____»_____ 20__ р.

МАГІСТРСЬКА РОБОТА

НА ТЕМУ:

Дослідження стійкості криптографічних алгоритмів

Освітньо-кваліфікаційний рівень "Магістр"
Спеціальність 123 – "Комп'ютерна інженерія"

Науковий керівник роботи:	_____ (підпис)	В.С. Кардашук (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	Я.О. Критська (ініціали, прізвище)
Студент:	_____ (підпис)	А.О.Метьолкін (ініціали, прізвище)
Група:		КІ-17дм

Сєвєродонецьк 2019

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень магістр
Напрям підготовки
(шифр і назва)
Спеціальність 123 "Комп'ютерна інженерія"
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри комп'ютерної інженерії
д.т.н., доц. І.С. Скарга-Бандурова
«___» _____20__р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Метьолкіну Артему Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження стійкості криптографічних алгоритмів

керівник проекту (роботи) доц. Кардашук Володимир Сергійович
(прізвище, ім'я, по батькові, науковий ступінь)
затверджені наказом вищого навчального закладу від "18" 10 2018 р. № 220/48
2. Строк подання студентом роботи
3. Вихідні дані до роботи Матеріали переддипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз існуючих методів шифрування 2. Аналіз методів підвищення криптостійкості 3. Проведення криптоаналізу на прикладі алгоритму RSA та хеш-функцій 4. Оцінка параметрів підвищення криптостійкості

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
Основна частина	Кардашук В.С.		
Охорона праці та безпека в надзвичайних ситуаціях	Критська Я. О.		

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Отримання завдання	01.02.2018-01.03.2018	
2	Аналіз завдання, обробка отриманих даних	01.03.2018-01.04.2018	
3	Пошук літературних джерел	01.04.2018-15.04.2018	
4	Планування змісту проекту	15.04.2018-01.06.2018	
5	Аналіз та обрання методів та програмного забезпечення для проведення дослідження	01.06.2018-01.07.2018	
6	Розробка програм, проведення дослідження та проведення розрахунків	01.07.2018-01.08.2018	
7	Оформлення пояснювальної записки	01.08.2018-01.11.2018	
8	Оформлення розділу охорона праці	01.11.2018-25.12.2018	
9	Підготовка та подання магістерської роботи до захисту	25.12.2018-09.01.2019	

Студент _____ **Метьолкін А.О.**
(підпис) (прізвище та ініціали)

Науковий керівник _____ **Кардашук В.С.**
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Метьолкін А.О. Дослідження стійкості криптографічних алгоритмів.

Дана магістерська робота присвячена дослідженню методів підвищення криптографічного стійкості алгоритмів шифрування.

Для дослідження обрано алгоритм RSA та методи хешування MD5, SHA, Whirlpool і здійснене моделювання їх роботи за допомогою програмного продукту Cryptool 2. Здійснено атаки на алгоритм шифрування та злом хеш-функції за допомогою програмного засобу PasswordsPro 3.1.

Представлені та оглянуті інструменти для програмування систем шифрування та наведена їх програмна реалізація.

За результатами дослідження зроблені висновки про надійність методів шифрування та алгоритмів, що базуються на хеш-функціях, а також з'ясована залежність криптостійкості від побічних параметрів, таких як ключі шифрування та додаткове опрацювання алгоритму.

Ключові слова: криптографія, криптостійкість, криптоаналіз, хеш-функція, ключі шифрування, атака.

ABSTARCT

Metelkin A. O. Research of stability of cryptographic algorithms.

Given master`s work is devoted to research of methods of increase of cryptographic stability of algorithms of encryption.

For research algorithm RSA and hashing methods MD5, SHA, Whirlpool is chosen and simulation of their work with the aid of software Cryptool 2 is carried out. Attacks to algorithm of encryption and breaking open hash functions with the aid of software PasswordsPro 3.1 are carried out.

Presented and are considered tools for programming of the systems of encryption and their program realization is presented.

By results of research, conclusions of reliability of methods of encryption and algorithms based on hash functions are made, as well as dependence strong cryptography from collateral parameters, such as the keys of encryption and additional of processing of algorithm is found out.

The key words: cryptography, strong cryptography, cryptanalysis, hash function, the keys of encryption, attack.

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ,СКОРОЧЕНЬ І ТЕРМІНІВ ...6
ВСТУП ...7
РОЗДІЛ 1 ОГЛЯД ПУБЛІКАЦІЙ, АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПОСТАНОВКА ЗАДАЧІ ...9
1.1 Аналіз вимог до систем шифрування ...9
1.2 Основні поняття криптології ...9
1.3 Вимоги до криптосистем ...17
1.4 Симетричні криптосистеми ...18
1.4.1 Метод Цезаря ...19
1.4.2 Системи шифрування Віженера ...20
1.5 Криптосистем з відкритим ключом ...21
1.5.1 Алгоритм RSA ...22
1.5.2 Алгоритм Ель-Гамаля ...27
1.6 Криптоаналіз ...29
1.6.1 Атака за часом ...29
1.6.2 Запобігання атаки за часом ...30
1.6.3 Атака «Людина посередині» ...30
1.6.4 Повний перебір ...30
1.6.5 Запобігання атаки «повного перебору» ...31
1.6.6 Квантовий злом ...31
1.7 Висновки до розділу 1 і постановка задачі дослідження методів шифрування та їх криптографічної стійкості ...32
РОЗДІЛ 2 ЗАСОБИ ПІДВИЩЕННЯ КРИПТОГРАФІЧНОЇ СТІЙКОСТІ ТА ЇХ РЕАЛІЗАЦІЯ ...33
2.1 Цифровий підпис ...33
2.2 Залежність криптостійкості від ключа ...34
2.3 Підвищення криптостійкості за допомогою хеш-функції ...35
2.4 Алгоритм DSA ...37
2.5 Алгоритм DES ...39
2.6 Алгоритм MD5 ...42
2.7 Алгоритм SHA-1 ...45
2.8 Аналіз методів підвищення криптографічного стійкості ...47
2.9 Висновок до розділу 2 ...49
РОЗДІЛ 3 РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ ...50
3.1 Криптоаналіз алгоритму MD5 ...50

3.2 Криптоаналіз алгоритму RSA та хеш-функцій (цифровий підпис) 57
3.3 Реалізація атаки на алгоритм шифрування RSA за методом Ферма 67
3.4 Висновок до розділу 3 70
РОЗДІЛ 4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ. ЕКОЛОГІЯ 71
4.1 Загальні питання з охорони праці 71
4.1.1 Організаційно-технічні заходи з охорони праці 71
4.2 Аналіз стану умов праці 72
4.2.1 Вимоги до приміщень 72
4.2.2 Вимоги до організації місця праці 72
4.3 Виробнича санітарія 73
4.3.1 Аналіз небезпечних і шкідливих факторів при експлуатації системи 73
4.3.2 Пожежна безпека 75
4.3.3 Електробезпека 76
4.4 Гігієнічні вимоги до параметрів виробничого середовища 76
4.4.1 Мікроклімат 76
4.4.2 Освітленість 77
4.5 Заходи з організації виробничого середовища і попередження виникнення надзвичайних ситуацій 79
4.6 Охорона навколишнього природного середовища 82
4.6.1 Загальні дані з охорони навколишнього природного середовища 82
4.7 Висновок до розділу 4 83
ВИСНОВКИ 84
ВИКОРИСТАНІ ДЖЕРЕЛА 86
ДОДАТОК А Код програми для обрахування хешу MD5 89
ДОДАТОК Б Код програми шифратора-дешифратора RSA 90

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ,СКОРОЧЕНЬ І ТЕРМІНІВ

Хеш – перетворення тексту довільної довжини в текст фіксованої довжини

Хеш-функція – функція, яка втілює алгоритм для отримання хешу і виконує перетворення.

Атака – спроба розкриття конкретного шифру із застосуванням методів криптоаналізу.

ЕЦП – вид електронного підпису, отриманого за результатом криптографічного перетворення набору електронних даних, який додається до цього набору або логічно з ним поєднується і дає змогу підтвердити його цілісність та ідентифікувати підписувача.

ПК (Персональний комп'ютер) – настільна мікро-ЕОМ, що має експлуатаційні характеристики побутового приладу і універсальні функціональні можливості.

ЕОМ (Електронно-обчислювальна машина) – комплекс технічних, апаратних і програмних засобів, призначених для автоматичної обробки інформації, обчислень.

RSA (від прізвищ Rivest, Shamir і Adleman) - криптографічний алгоритм з відкритим ключем, який базується на обчислювальної складності задачі факторизації великих цілих чисел.

SHA (Secure Hash Algorithm) – алгоритм криптографічного хешування сімейства SHA

MD5 (Message Digest 5) – 28-бітний алгоритм хешування, для створення «відбитків» або дайджестів повідомлення довільної довжини і подальшої перевірки їх достовірності.

ВСТУП

На сьогоднішній день серед методів підвищення криптостійкості систем шифрування можна відзначити симетричні і асиметричні алгоритми, та комплексні методі із використанням хеш-функцій. При великій кількості методів та алгоритмів шифрування надійності не завжди приділяється належна увага. Але головною метою криптографії залишається забезпечення безпеки особистих даних при передачі, тому дослідження надійності і стійкості алгоритмів являється першочерговим завданням

В даній магістерській роботі досліджено питання криптостійкості на прикладі моделювання роботи алгоритму за допомогою програмного продукту Cryptool 2, проведення криптоаналізу за допомогою математичних методів методами і злом хеш-функцій за допомогою програмного засобу PasswordsPro 3.1.

Метою дослідження є підвищення криптостійкості вже існуючих алгоритмів шифрування за допомогою таких методів, як хеш-функції, ітерація ключів шифрування та інших, із проведенням подальшої порівняльної характеристики, для виявлення найнадійнішої сукупності методів.

Cryptool 2 підтримує усі відомі на сьогодні моделі шифрування. Функціональні блоки мають модулі для введення і виведення інформації в процесі роботи, що дає змогу на аналізування процесу шифрування та проведення аналітичної оцінки за результатами моделювання.

PasswordsPro 3.1 – програмний засіб, призначений зберігання паролів або для відновлення початкового тексту за їх хешем, за допомогою проведення атак. При проведенні атак, можливо зробити висновки щодо криптостійкості алгоритмів з хеш-функцій.

Удосконалено алгоритми шифрування шляхом синтезу із інструментами, котрі посилюють криптографічну стійкість, та за допомогою аналітичних, математичних та практичних методів аналізу перевірена надійність та стійкості методів і запропоновано для подальшого використання.

Практичне значення отриманих результатів полягає в тому, що основні наукові положення дисертації реалізовані у виді розрахункових моделей та програмних засобів, які утворюють прикладну інформаційну технологію для проведення аналізу криптографічних методів та виявлення їх недоліків. У свою чергу інші методи для перевірки існують окремо і надають результати тільки про певні аспекти алгоритмів шифрування і не достатньо чітко показують їх практичне значення криптостійкості і зазвичай стійкість методу при зломі або проведенні атак описано лише відносними значеннями.

За темою магістерської роботи з викладенням її основних результатів опубліковано 3 наукові праці, серед яких 1 стаття у наукових фахових виданнях України та 2 публікації у збірниках науково-практичних праць.

Магістерська робота складається із вступу, чотирьох розділів, висновків, списку використаних джерел і додатків. Загальний обсяг роботи складає 92 сторінки, з яких основний текст на 76 сторінках, список використаних джерел із 52 найменувань на 3 сторінках, додатки на 4 сторінках. Робота містить 8 таблиць, та 36 рисунків.

.

РОЗДІЛ 1
ОГЛЯД ПУБЛІКАЦІЙ, АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз вимог до систем шифрування

Безпека ланцюга залежить від найслабшої ланки: чим воно надійніше, тим ланцюг міцніше. У хорошій криптосистемі повинні бути досконально перевірені і алгоритм, і протокол, і ключі, і все інше. Якщо криптографічний алгоритм досить стійкий, а генератор випадкових чисел, який використовується для створення ключів, погано налаштований, то з великою ймовірністю він буде джерелом небезпеки. Якщо вдасться поліпшити генератор, але не будуть зачищати осередки пам'яті комп'ютера, після того як в них побував згенерований ключ, то стійкість даної системи буде під загрозою. Якщо використовуються стійкий криптографічний алгоритм і дійсно випадкові ключі, які акуратно стираються з пам'яті комп'ютера після того, як вони були використані, але перед шифруванням файл, в якому разом з вашою адресою та прізвищем вказані всі ваші доходи і інша подібна інформація, було отримана третьою особою, то незалежно від алгоритму дані будуть втрачені.

1.2 Основні поняття криптології

Криптологія розділяється на два основних напрямки - криптографію і криптоаналіз. Цілі цих двох напрямків прямо протилежні.

Криптографія в першу чергу спеціалізується на методах шифрування інформації – шифрування первинного тексту за допомогою алгоритму або криптоключа в шифрований текст [1].

Під конфіденційність розуміють можливість отримання інформації з перетворення масиву без знання додаткової інформації (ключа). Автентичність інформації складається в автентичності авторства і цілісності.

Криптоаналіз об'єднує математичні методи порушення конфіденційності і автентичності інформації без знання ключів [2].

Криптографія - наука про методи забезпечення конфіденційності даних, збереження даних (наприклад хеш-функції), аутентифікації за допомогою ключів і цифрових підписів, а також неможливості відмови від авторства.

Існує кілька способів, відповідно до яких можуть класифікуватися криптографічні системи. Розроблена класифікація дозволяє визначити схильність криптосистеми до різних

атак з боку противника, ідентифікуючи її відповідно до особливостей її реалізації. Пропонується розрізнити криптографічні засоби за критеріями (Рис. 1.1).

Рисунок 1.1 – Класифікація криптографічних засобів

Криптографічна система називається криптосистемою обмеженого використання, якщо її стійкість ґрунтується на збереженні в секреті самого характеру алгоритмів шифрування й розшифрування. Найпростішим прикладом такої системи можна вважати шифр Цезаря, у якому перетворення інформації зводиться до простої заміни кожного символу відкритого тексту третім, наступним за ним, символом алфавіту. Стійкість криптосистеми загального використання ґрунтується не на секретності алгоритмів шифрування і розшифрування, а на секретності деякого значення, яке називається її ключем. Такий ключ повинен вироблятися конкретними користувачами таким чином, щоб навіть розроблювач криптосистеми не міг розкрити її, не маючи доступу до ключа. Зберігаючи інформацію про алгоритм у секреті, можна забезпечити деяку додаткову безпеку. Дослідження надійності таких систем завжди повинні проводитися в припущенні, що потенційному противнику відома вся інформація про криптосистему, за винятком використовуваного ключа.

За класифікаційною схемою криптосистеми за ключами підрозділяються на 3 типи: Безключові, які не використовують ключі у процесі криптографічних перетворень. Одноключові, що використовують у своїх обчисленнях тільки секретний ключ. Двоключові, у яких на різних етапах обчислень застосовуються два види ключів: закриті (особисті) і відкриті. Ця класифікація є неповною, тому що в ній відсутні багатоключові криптосистеми. До цього типу можна віднести схеми поділу секрету, або (m, n)- граничні схеми. У такій системі

секретний ключ розбивається на n частин (де n – кількість учасників схеми поділу секрету) так, що за кожними m частинами можна відновити зашифровану інформацію ($m \leq n$). Отримані "частки" ключа розподіляються між усіма учасниками, після чого будь-які m учасників можуть спільно реконструювати зашифровану інформацію. В окремому випадку, коли $n = m$, для відновлення секрету необхідна присутність усіх учасників [1].

Здатність криптосистеми протистояти атакам криптоаналітика називається стійкістю. Кількісно стійкість виміряється як складність найкращого алгоритму, що приводить криптоаналітика до успіху із прийнятною ймовірністю. Універсальний метод прямого перебору множини всіх можливих ключів дозволяє одержати оцінку зверху для стійкості алгоритму шифрування. Відносний очікуваний безпечний час визначається як напівдобуток кількості відкритих ключів і часу, необхідного криптоаналітику для того, щоб випробовувати кожний ключ. Залежно від мети і можливостей криптоаналітика, змінюються й стійкість. Розрізняють стійкість ключа (складність розкриття ключа найкращим відомим алгоритмом), стійкість безключового читання, імітостійкість (складність нав'язування неправильної інформації найкращим відомим алгоритмом) і ймовірність нав'язування неправильної інформації. Аналогічно можна розрізняти стійкість власне криптоалгоритму, стійкість протоколу, стійкість алгоритму генерації й поширення ключів. Залежно від складності зламу алгоритми забезпечують різні степені захисту. За основу ставиться принципова можливість одержання з перехоплення деякої інформації про відкритий текст або використаний ключ. Існують безумовностійкі (або теоретичностійкі), доказовостійкій, приблизностійкі криптоалгоритми. Теоретично стійкі системи створюють шифртексти, що містять недостатню кількість інформації для однозначного визначення відповідних їм текстів (або ключів). У найкращому разі відкритий текст може бути локалізований у досить великій підмножині множини всіх відкритих текстів, і його можна лише «вгадати» з мізерно малою ймовірністю. Ніякий метод криптоаналізу, включаючи повний перебір ключів, не дозволяє не тільки визначити ключ або відкритий текст, але навіть одержати деяку інформацію про них. Алгоритм безумовностійкий, якщо відновлення відкритого тексту неможливе при будь-якому обсязі шифротексту, отриманого криптоаналітиком. Безпека безумовностійких криптоалгоритмів заснована на доведених теоремах про неможливість розкриття ключа [2].

Стійкість доказовостійких криптоалгоритмів визначається складністю розв'язання добре відомого математичного завдання, яке намагалися розв'язати багато математиків і яке є загальновизнано складним. Як приклад можна навести системи DH (Діффі – Хелмана) і RSA (Рівеста – Шаміра – Адельмана), засновані на складності дискретного логарифмування та розкладання цілого числа на множники відповідно. Підвищення стійкості в криптоалгоритмах досягається збільшенням розміру математичного завдання або її заміною, що, як правило,

тягне ланцюг змін в апаратурі, яка використовується для шифрування. Приблизностійкі криптоалгоритми засновані на складності розв'язання приватного математичного завдання, яке не зводиться до добре відомих завдань і яку намагалися розв'язати один або кілька чоловік. Прикладами можуть служити блокові шифри. Приблизностійкі криптоалгоритми характеризуються порівняно малою вивченістю математичних завдань, на яких базується їх стійкість. Однак такі шифри мають велику гнучкість, що дозволяє при виявленні слабких місць не відмовлятися від алгоритмів, а проводити їх доробку[3].

Розглянемо цю класифікацію в застосуванні до генераторів псевдовипадкових чисел. Для генерації ключової інформації, призначеної для використання в рамках симетричної криптосистеми, використовуються такі методи (у порядку зростання якості): Програмна генерація, що припускає обчислення чергового псевдовипадкового числа як функції поточного часу, послідовності символів, уведених користувачем, особливостей його клавіатурного почерку і т. д.; Програмна генерація, заснована на моделюванні якісного псевдовипадкового генератора з рівномірним законом розподілу; Апаратна генерація з використанням якісного псевдовипадкового генератора; Апаратна генерація з використанням генераторів випадкових послідовностей, побудованих на основі фізичних генераторів шуму і якісних псевдовипадкових генераторів. Кращий спосіб генерації множини випадкових бітів – витяг їх із природно випадкових подій реального світу. Часто такий метод вимагає наявності спеціальної апаратури, але можна реалізувати його й на комп'ютерах. У якості випадкових величин можна також розглядати інтервали між натисканнями клавіш клавіатури. Головний недолік подібних систем – можливі закономірності в послідовності, яка генерується. Використовувані фізичні процеси можуть бути випадкові, однак використання вимірювальних інструментів може призвести до появи проблем: зсуву, відхилення або кореляції між бітами. Обійти ці недоліки можна, використовуючи не один, а кілька випадкових джерел.

Відповідно до діючого на території України законодавства, якщо організація використовує несертифіковані в Україні криптографічні алгоритми шифрування й електронний цифровий підпис даних, вона не може вести обмін документами з державними установами (Постанова 26.11.2015 № 829 Про затвердження нормативно-правових актів з питань інформаційної безпеки [51]). Забезпечити юридичну значимість електронних документів при обміні ними між користувачами дозволить використання сертифікованих криптоалгоритмів. Класифікація криптоаналітичних атак Наведена на Рисунок 1.2 класифікація дозволяє розрізняти криптоаналітичні атаки і їх наслідки одночасно за декількома параметрами. Класифікація з доступу до відкритого й зашифрованого тексту Перш ніж класифікувати атаки, введемо ряд позначень: відкритий текст будемо позначати буквою M, шифртекст – C (у якості M може виступати будь-яка послідовність бітів: текстовий файл і

т. д.). Нехай для шифрування і розшифрування використовуються ключі k і $k/$ відповідно (у симетричній криптографії $k = k/$); позначимо функцію шифрування E_k, розшифрування – D_k. Тоді виконуються співвідношення $E_k(M) = C, D_k(C) = M$. Донедавна за критерієм доступу до відкритого і шифрованого тексту виділялося чотири основні типи криптоаналітичних атак (Рис. 1.2). Однак останнім часом одним із найактуальніших напрямів криптоаналізу стало здійснення атак, що використовують особливості реалізації та робочого середовища.

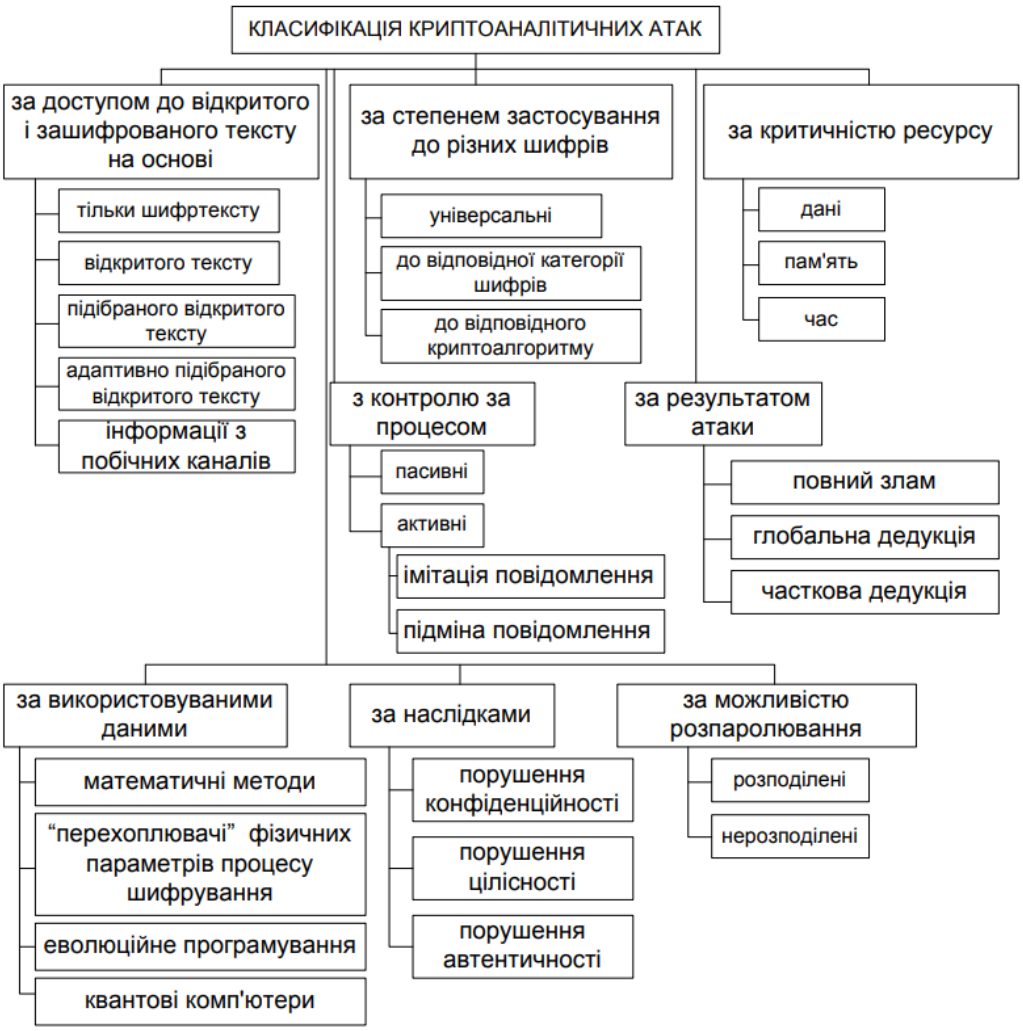


Рисунок 1.2 - Класифікація криптографічних атак

Атаки сторонніми або побічними каналами – це вид криптографічних атак, що використовують інформацію, отриману зі сторонніх або побічних каналів. У кожному випадку передбачається (згідно з фундаментальним допущенням Керкгоффа, що криптоаналітик знає використовуваний алгоритм шифрування. Атака на основі тільки шифротексту. Криптоаналітик розташовує шифротекстами $c_1, \ldots, c_m$, отриманими з невідомих відкритих текстів $m_1, \ldots, m_m$ різних повідомлень. Потрібно знайти хоча б один з $m_i \; i = i, \ldots, m$ (або

відповідний ключ k_i), виходячи з достатнього числа m криптограм, або переконатися у своїй нездатності зробити це. У якості окремих випадків можливий збіг ключів або збіг відкритих текстів.

Сфера інтересів криптоаналізу - дослідження можливості дешифрування інформації без знання ключів.

Атака на основі відкритого тексту. Криптоаналітик має у своєму розпорядженні пари $(m_1, c_1), \ldots, (m_m, c_m)$ відкритих і відповідних їм зашифрованих текстів. Потрібно визначити ключ k_i для хоча б однієї з пар. В окремому випадку, коли $k_1 = \ldots = k_m = k$, потрібно визначити ключ k або, переконавшись у своїй нездатності зробити це, визначити відкритий текст $m_m + 1$ ще однієї криптограми $c_m + 1$, шифрованої на тому ж ключі.

Атака на основі підібраного відкритого тексту відрізняється від попередньої лише тим, що криптоаналітик має можливість вибору відкритих текстів $m_1, \ldots, m_m$. Мета атаки та ж, що й попередньої. Подібна атака можлива, наприклад, у випадку, коли криптоаналітик має доступ до шифратора передавальної сторони.

Атака на основі адаптивно підібраного відкритого тексту. Це окремий випадок описаної вище атаки з використанням підібраного відкритого тексту. Криптоаналітик може не тільки вибирати використовуваний текст, що шифрується, але також уточнювати свій наступний вибір на основі отриманих раніше результатів шифрування.

Атака на основі інформації з побічних каналів. Криптоаналітик має інформацію, яка може бути отримана з обладнання шифрування й не є при цьому ні відкритим текстом, ні шифротекстом.

Атаки з побічних каналів, у свою чергу, класифікуються за такими типами:

- за процесом контролю алгоритму: пасивній активні;
- за способом доступу до модуля: агресивні, напівагресивні і неагресивні;
- за методом, застосованим у процесі аналізу: проста атака за побічним каналу, диференціальна атака за побічного каналу;

За видом використовуваного побічного каналу:

- атаки за часом виконання (Timing Attacks);
- атаки за енергоспоживанням (Power Analysis Attacks);
- атаки за помилками обчислень (Fault Attacks);
- атаки за електромагнітними випромінюванням (Electro-magnetic Analysis);
- атаки за помилками у каналі зв’язку (Error Message Attacks);
- атаки за кеш-пам’яттю (Cache-based Attacks);
- акустичні атаки (Acoustic Attacks); атаки за світовім випромінюванням (Visible Light Attacks).

Атаки з використанням відомого або підібраного відкритого тексту зустрічаються частіше, ніж можна подумати. Необхідною вимогою до криптоалгоритму є здатність протистояти таким атакам. Це означає, що розсекречення деякої інформації, що передається каналами зв’язку в шифрованому виді, не повинне приводити до розсекречення іншої інформації, шифрованої на цьому ж ключі. Крім того, зазначена вимога враховує особливості експлуатації апаратури й допускає деякі вільності з боку оператора або осіб, що мають доступ до формування засекреченої інформації. Атаки на основі підібраних текстів вважаються найнебезпечнішими.

Алгоритми можна розподілити на такі види:

- симетричні;
- криптосистеми з відкритим ключем;
- системи електронного підпису;
- системи з управляючими ключами.

Можливо виділити сфери застосування криптографія:

1) З відкритими каналами зв'язку. До них можна віднести телефонію, Інтернет, факс і т.д.
2) Операції з банківськими картами.
3) Робота із паролями користувачів (зберігання, опрацювання).
4) Передача документів за допомогою в середині мережі Інтернет.
5) Безпечне зберігання інформації на персональному комп'ютері.

Для шифрування/дешифрування береться текст, що складається з алфавіту, реальному чи абстрактному. Під цими поняттями маються на увазі наступне:

Алфавіт –дані, що складають повний набір літер чи символів, використовується для шифрування та дешифрування.

Текст – набір символів, що підпорядковані якомусь алфавіту.

Як приклади алфавітів, що використовуються в сучасних інформаційних системах можна навести наступні:

- алфавіт Z_{32}- 32 літери російського алфавіту і пробіл;
- алфавіт Z_{256} - символи, що входять в стандартні коди ASCII і KOI-8;
- бінарний алфавіт - Z_2= {0,1};
- восьмеричний алфавіт або шістнадцятковий алфавіт;

Шифрування – процес перетворення вихідного текст, що за допомогою алфавіту та ключів, замінюється шифрованих текстом [4].

Дешифрування - зворотний процес шифрування. На основі ключа шифрований текст перетвориться у вихідний [4].

Ключ - інформація, необхідна для формування із первинного тексту у шифрований.

Криптографічний система являє собою сімейство T перетворень відкритого тексту. Члени цього сімейства індексуються, або позначаються символом k; параметр k є ключем. Простір ключів K - це набір можливих значень ключа. Зазвичай ключ являє собою послідовний ряд букв алфавіту. Перетворення T_k визначається відповідним алгоритмом і значенням параметра k [5].

Криптографічні методи можна розбити на два класи:

- обробка інформації шляхом заміни і переміщення букв, при якому обсяг даних не змінюється, тобто звичайна заміна одного алфавіту іншим, відповідним йому;
- стиснення інформації за допомогою заміни окремих поєднань літер, слів або фраз.

За способом реалізації криптографічні методи можливі в апаратному та програмному виконанні.

Апаратні способи шифрування інформації застосовуються для передачі захищених даних по телекомунікаційної мережі.

Для реалізації шифрування за допомогою змішаного алфавіту виконується шляхом заміни окремих елементів або блоків в межах одного або декількох символів.

Програмні способи застосовуються для шифрування інформації, що зберігається на носіях (твердотільні накопичувачі, периферія). Це можуть бути дані різних інформаційно-довідкових систем, автоматизованих систем обробки даних, автоматизованих системам управління і ін. Програмні засоби шифрування зводяться до операцій перестановки, перекодування і складання за модулем 2 з ключовими словами (словом).

Криптосистеми поділяються на симетричні і з відкритим ключем.

У симетричних криптосистемах і для шифрування, і для дешифрування використовується один і той же ключ.

У системах з відкритим ключем використовуються два ключі - відкритий і закритий, які математично пов'язані один з одним. Інформація шифрується за допомогою відкритого ключа, що знаходиться у відкритому доступі, а розшифровується за допомогою закритого ключа, відомого тільки одержувачу повідомлення.

Терміни розподіл ключів і керування ключами відносяться до процесів системи обробки інформації, змістом яких є складання і розподіл ключів між користувачами.

Електронній (цифровій) підписом називається елемент, що приєднуються до тексту та виконує криптографічне перетворення, що дозволяє при отриманні тексту іншим користувачем перевірити авторство і достовірність повідомлення.

Крипостійкістю називається характеристика шифру - стійкість до дешифрованню без

знання ключа (тобто криптоаналізу). Є кілька показників криптостійкості, серед яких:

- загальна кількість можливих ключів;
- середній час, необхідний для злому методом повного перебору.

Ефективність шифрування з метою захисту інформації напряму залежить від степеню збереження у секреті ключів шифрування та стійкість самого алгоритму для шифрування.

Процес шифрування даних може здійснюватися як програмно, так і апаратно. Апаратна реалізація є дорожчою через вартість обладнання, але є свої плюси: висока продуктивність, простота, захищеність і т.д. Програмна реалізація більш практична і надає більшу гнучкість при використанні алгоритмів. Криптоаналіз може проводитися в сукупності з шифруванням. Для підвищення криптостійкості алгоритмів шифрування використовують методи, які посилюють їх, такі як конвертації (текст) за певними стандартами або використання булевої алгебри і т.д.

Зазвичай дані зберігаються на самому комп'ютері, або в пам'яті різних носіїв тоді криптографічні методи використовуються як інструмент для запобігання несанкціонованому доступу. Криптографічні зміни блокують передачу інформації між різними елементами системи. Більшість методів існуючих зараз можуть бути успішно використані для закриття інформації, але головною проблемою виступає їх надійність і доступність. Одними з головних методів шифрування є асиметричні методи шифрування, один із таких - алгоритм RSA.

1.3 Вимоги до криптосистем

Для сучасних криптографічних систем були висунуті такі загальноприйняті вимоги з криптостійкості та захисту інформації для даних систем:

- Процес дешифрування проводиться тільки при наявності ключів шифрування;
- Затрачений час на визначення використаного ключа шифрування за фрагментом шифрованого тексту і відповідного йому початкового тексту повинна бути досить великою для її реального здійснення
- Криптостійкість методу повинна бути максимальною для будь-якого типу атак, хоч при реалізації математичного методу чи комп’ютерним перебором можливих ключів.
- Знання алгоритму шифрування не повинно впливати на загальну надійність системи;
- Будь яка зміна чи виправлення ключів шифрування повинна вести до значного впливову на загальний шифрований текст;
- Структурні елементи алгоритму шифрування повинні бути незмінними;
- Хеш-функції та елементи цифрових підписів чи сертифікати мають бути надійно

захищені у системі чи шифротексті;

- Довжина тексту до шифрування повинно співпадати із вихідним після дешифрування;
- Не допускається легко помітно залежність між ключами шифрування та безпосередньо самим алгоритмом шифрування, так як це може бути критичним для загальної системи;
- Будь-який ключ з безлічі можливих повинен забезпечувати максимальний захист інформації;
- Алгоритм та технічна складова системи не повинна залежати ні від ключів та їх параметрів, ні від даних для шифрування.

1.4 Симетричні криптосистеми

У симетричних криптоалгоритмах для зашифровування і розшифрування повідомлення використовується один і той же блок інформації (ключ). Хоча алгоритм впливу на передані дані може бути відомий стороннім особам, але він залежить від секретного ключа, яким повинні володіти тільки відправник і одержувач.

Симетричні алгоритми шифрування виконує перетворення даних невеликого обсягу (1 біт або 32-128 біт), котрий повністю залежить від секретного ключа, тобто прочитати вихідне повідомлення можна тільки при наявності секретний ключа шифрування. Симетричні криптосистеми дозволяють на основі симетричних криптоалгоритмів кодувати і декодувати файли довільної довжини.

Характерна особливість симетричних блокових криптоалгоритмів - перетворення блоку вхідної інформації фіксованої довжини і отримання результуючого блоку того ж обсягу, але недоступного для прочитання стороннім особам, які не володіють ключем. Схему роботи симетричного блочного шифру можна описати функціями

$$C = E_K(M), M = D_K(C) \tag{1.1}$$

де M - вихідний (відкритий) блок даних, C - зашифрований блок даних.

Ключ K є параметром симетричного блокового криптоалгоритму і являє собою блок двійковій інформації фіксованого розміру. Вихідний M і зашифровані C блоки даних також мають фіксовану розрядність, рівну між собою, але необов'язково рівну довжині ключа K.

Найбільш поширені алгоритми шифрування простих замін і перестановок. І на рівні з ними існують такі криптографічні методи:

Багатоалфавитна підстановка - для заміни символів вихідного тексту використовується не один, а кілька алфавітів. Зазвичай алфавіти для заміни утворені із символів вихідного алфавіту, записаних в іншому порядку.

Перестановка - простий засіб криптографії. Частіш за все використовується в комбінації з іншими методами.

Гамування - полягає в «накладенні» послідовності, що складається з випадкових чисел, на відкритий текст. Послідовність випадкових чисел називається гамма-послідовністю і використовується для зашифровування і розшифрування даних. Підсумовування, зазвичай, виконується в будь-якому кінцевому полі

1.4.1 Метод Цезаря

Вкрай простий приклад симетричного шифрування - це шифр з підстановкою. За допомогою даного методу кожна частка інформації замінюється іншою. Найчастіше це досягається зміщенням символів даного алфавіту і наявний приклад – шифр Цезаря (Рис. 1.3). Алгоритм полягає в зміщенні алфавіту шифрування, а ключ - це число зміщення букв, для даного алфавіту.[6]

Підстановка визначається по таблиці заміщення, що містить пари відповідних букв "вихідний текст - шифрований текст" (Рис. 1.3).

Щоб прочитати такий текст, потрібен ключ, який дає змогу для розшифрування шляхом відношення алфавітів між собою. Але подібні алгоритми не є стійкими до частотного криптоаналізу шифру. Принцип посилення полягав у заміні не на одну букву, а на дві або більше. Подібні принципи дали основу для формування більш сучасних і більш досконалих криптосистем.

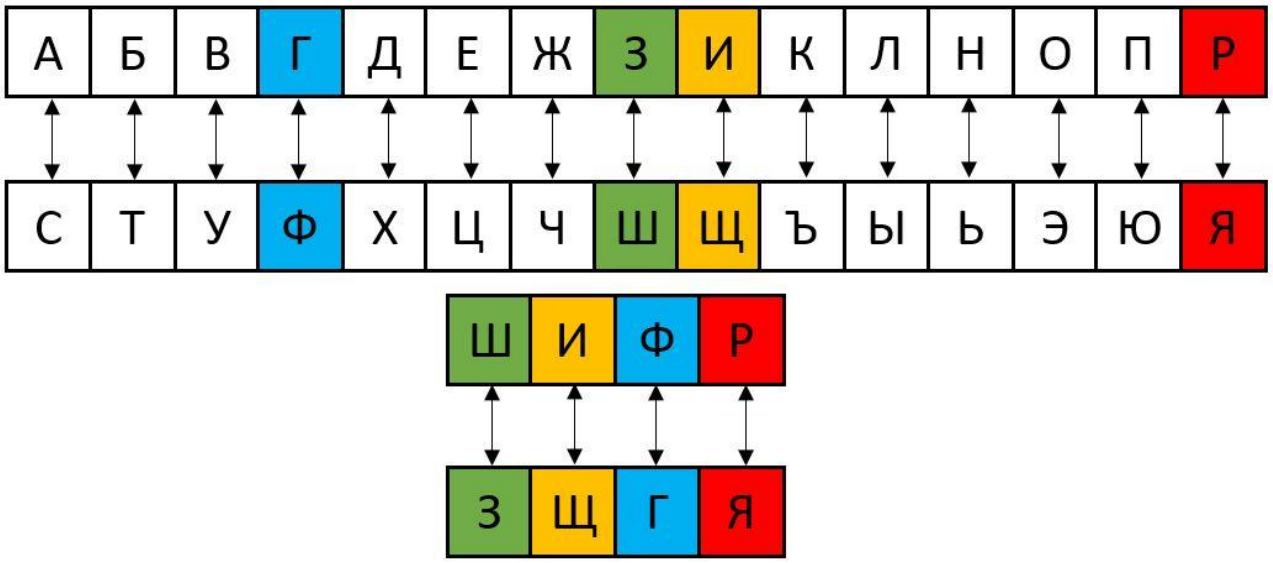


Рисунок 1.3 – Схема прикладу роботи шифру Цезаря із зміщенням на 15 символів

Шифр Цезаря може бути легко зламаний навіть в разі, коли зловмисник знає тільки зашифрований текст. Можна розглянути дві ситуації:

- Зловмисник для атаки використовує простий шифр підстановки, але не знає, що це - схема Цезаря.
- Зловмисник знає, що використовувався шифр Цезаря, але не знає значення зсуву.

Для першого сценарію – шифр може бути розпізнаний та зламаний, використовуючи ті ж самі методи що і для методів простої заміни та підстановки, за допомогою них визначається певна закономірність і враховується послідовність.

У другому випадку, злом шифру є навіть більш простим, всі варіанти можуть бути перевірені методом грубої сили.

Як альтернатива існують шифр Хілла і шифр Плейфера. Вони засновані на більш ніж однієї заміни символів, а саме 2-о кратної (код Плейфера), або N-кратної (шифр Хілла).

1.4.2 Системи шифрування Віженера

Суть алгоритму шифрування проста. Шифр Віженера - це послідовність шифрів Цезаря з різними значеннями зсуву.

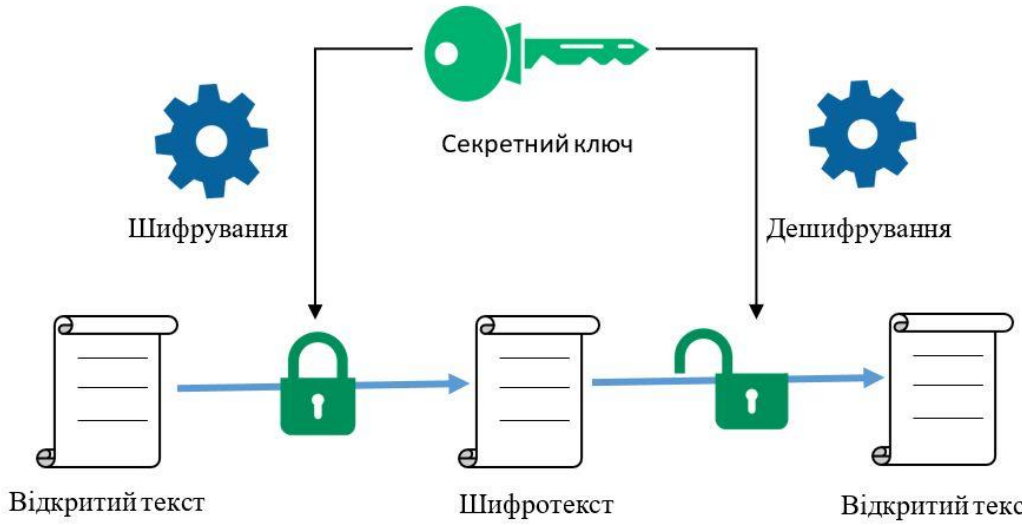


Рисунок 1.5 – Принцип роботи симетричного шифрування на основі ключа

В системі Віженера задається якась кінцева послідовність ключа:

$$k = (k_0, k_1, \dots, k_n) \tag{1.2}$$

яка називається ключем користувача, вона триває до нескінченної послідовності, повторюючи ланцюжок. Таким чином, виходить робочий ключ.

Якщо пронумерувати літери алфавіту від 0 до 32 (а → 0, б → 1, в → 2, …), то шифрування Віженера є можливим представити формулою:

$$C_i = (P_i + K_j) mod 33 \quad (1.3)$$

де K_j — j-та літера ключового слова.

Ключове слово повторюється, поки не отримано гаму, рівну довжині повідомлення.

Тобто, якщо зашифрувати фразу « Приклад шифру» із ключом «Ключ», ми отримаємо алгоритм, як на Рисунку 1.4

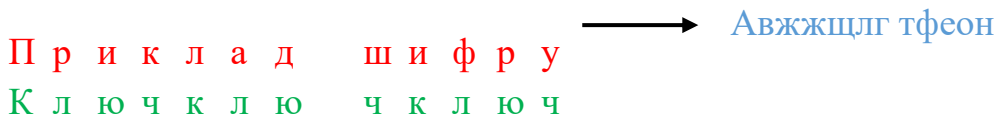


Рисунок 1.4 - Приклад роботи системи Віженера в українській мові

Крипостійкість методу різко зменшується зі зменшенням довжини ключа. Проте така система як шифр Віженера допускає нескладну апаратну або програмну реалізацію і при досить великій довжині ключа може бути використаний в сучасних інформаційних системах.

1.5 Криптосистем з відкритим ключом

У 1976 р був введений новий стандарт криптосистеми - система з відкритим ключем. Вона містить два ключі, відкритий і закритий, вибрані таким чином, що їх послідовне застосування до вихідного тексту ніяк не впливає на нього. Шифруюча процедура використовує за допомогою відкритого ключа, а процес дешифрації за допомогою закритого. Дешифрування коду без знання секретного ключа практично нездійсненна; зокрема, практично нерозв'язна задача без наявності усіх компонентів, такими як вхідний та вихідний текст, знаходження секретного ключа за відомим лише відкритим ключем. Основна перевага криптографії з відкритим ключем - спрощений механізм обміну ключами. При здійсненні комунікації по каналу зв'язку передається тільки відкритий ключ, що уможливлює використання для цієї мети звичайного каналу і усуває потребу в спеціальному захищеному каналі для передачі ключа.

Із введенням до експлуатації систем з відкритим ключем поняття про захист інформації, а разом з ним функції криптографії отримав значний прогрес, звертаючись на попередні алгоритми та системи шифрування даних. Якщо раніше основним завданням криптографічних систем вважалося надійне шифрування інформації, в даний час область застосування криптографії поширилися майже на усі сфери діяльності людей із комп’ютерними системами. Найбільш поширені функції криптографічних систем з відкритим ключем - шифрування і

цифровий підпис. Криптографія з відкритим ключем набула широкого поширення, і не тільки в шифруванні шпигунських і дипломатичних послань, її також використовують сайти з підтримкою протоколу HTTPS, месенджери, wifi-роутери, банківські системи і багато іншого. На основі асиметричної криптографії базується електронний підпис. Також на асиметричної криптографії побудований алгоритм блокчейна, на якому, в свою чергу побудовані всі криптовалюта, включаючи біткоіни

1.5.1 Алгоритм RSA

Наприкінці 1970-х інтерес до криптографії зріс як у держслужб, так і у звичайного населення. Так, криптосистема RSA, розроблена Ривестом (R), Шамір (S) і Адлеманом (A) в 1977 році, незважаючи на початкові перешкоди Агентства Національної Безпеки, зараз використовується практично скрізь. Вона володіє цікавими криптографічними властивостями, що використовуються у симетричних системах шифрування. У симетричній криптографії для шифрування і дешифрування використовується один ключ. Ці прості алгоритми ідеологічно описані ще в 50-х, але всім їм властивий один великий недолік, а саме необхідність каналу для передачі ключа шифрування. Цю проблему симетричного шифрування називають проблемою поширення ключів [7].

RSA - асиметричний алгоритми шифрування, є досить поширеним типом.

Алгоритм RSA включає в себе чотири етапи: генерація ключів, передача ключів, шифрування і розшифрування[8].

Алгоритм не є складним, через це можливе опрацювання великого обсягу даних, і формування як правило, розділене на обчислювальні блоки фіксованого довжини, кожен вхідний блок, незалежно не від чого шифрується окремо. Такі криптосистеми шифрування називаються блокові. Щоб підсилити самі методи шифрування, при цьому не деформуючи дані, використовуються спеціальні методи, які посилюють криптостійкість. Такі методи, як правило, працюють саме з ключами шифрування.

Дешифрування - зворотний процес шифрування. На основі ключа шифрований текст перетвориться у вихідний[4].

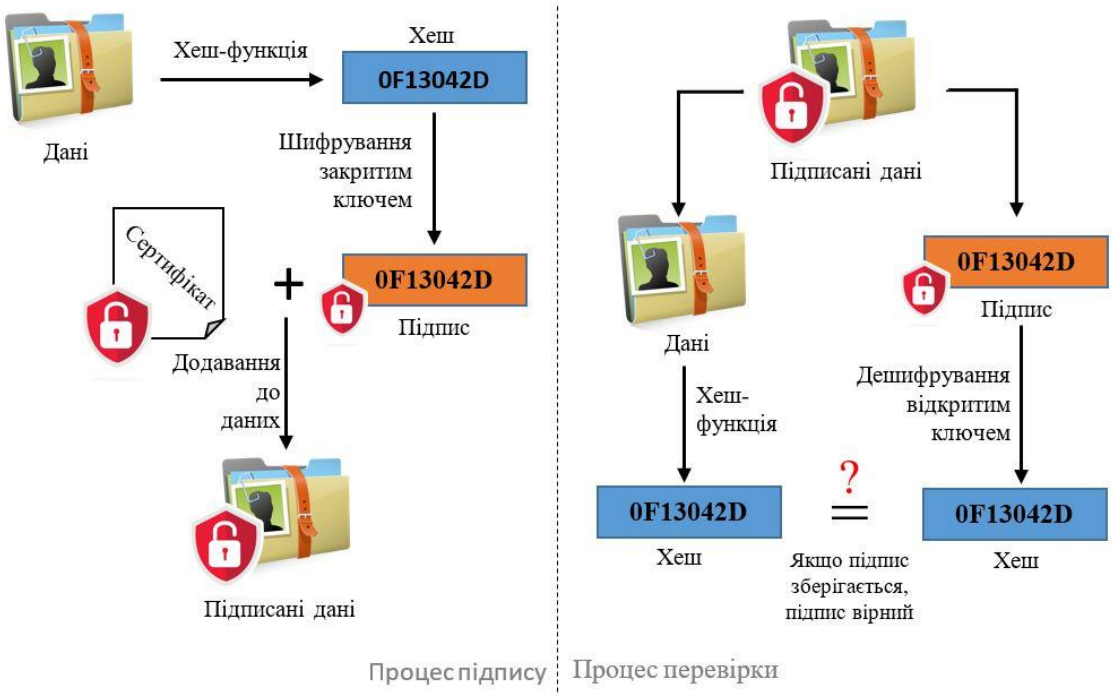


Рисунок 1.6 – Принцип роботи симетричного шифрування на основі ключа

Відкритий і закритий ключ між собою пов'язані певними математичними залежностями, тому знанням параметрів неможливий реалізувати підбір ключів шифрування за короткий час. Перевірити, чи не була інформація спотворена під час передачі по зашифрованому каналу можна за допомогою ЕЦП. В цьому випадку текст зашифрований закритим ключем і разом з відкритим текстом відправляється відправнику. Якщо проводити дешифрування відкритим ключем і текст співпадає з не зашифрованим текстом, то даний процес був коректний. На практиці закритим ключем зазвичай зашифрована хеш-функція документа. Електронний підпис крім іншого містить так званий сертифікат відкритого ключа, виданий довіреним центром сертифікації, який має дані по всіх відкритих ключів та їх користувачам. Сертифікат також повинен мати підпис. Важливо, що відкритий ключ довіреного центру повинен бути відомий заздалегідь, інакше його теж можна підробити (Рис. 1.6).

У криптографічного системі RSA кожен ключ складається з пари цілих чисел. Кожен учасник створює свій відкритий і закритий ключ самостійно. Закритий ключ кожен з них тримає в секреті, а відкриті можуть знаходитись у відкритому доступі. Відкритий і закритий ключі кожного учасника обміну повідомленнями в криптосистемі RSA утворюють «узгоджену пару» в тому сенсі, що вони є взаємно зворотними. Тобто для будь-яких допустимих пар відкритого і закритого ключів (p, s) існують відповідні функції шифрування $E_p\,(x)$ і розшифрування $D_s\,(x)$ такі, що для будь-якого повідомлення $m \in M$, де M - безліч допустимих повідомлень, $m = D_s\,(E_p\,(m)) = E_p\,(D_s\,(m))$ [8].

1.5.1.1 Генерація ключа

Генерація ключів RSA проводиться наступним чином:

За допомогою генерації ключів обирають прості числа p і q заданого розміру (як приклад, 1024 біта).Обчислюється їх добуток $n = p \cdot q$, яке називається модулем.

Обчислюється значення функції Ейлера від числа n [8]:

$$\varphi(n) = (p-1) \cdot (q-1) \tag{1.4}$$

Обирається ціле число e ($1 < e < \varphi(n)$), взаємно просте зі значенням функції $\varphi(n)$. Зазвичай в якості e беруть просте число для двійковій запису.

Число e називається відкритою експонентою. Час, необхідний для шифрування з використанням швидкого зведення в ступінь, пропорційно числу одиничних біт в e, в двоїчного запису цього числа.

Занадто малі значення e, наприклад 3, потенційно можуть послабити безпеку схеми RSA.

Обчислюється число d, мультиплікативно зворотне до числа e по модулю $\varphi(n)$, тобто число, яке задовольняє відношенню:

$$d \cdot e \equiv 1 \left(mod \varphi(n)\right) \tag{1.5}$$

Порівняння двох цілих чисел по модулю натурального числа m - математична операція, що дозволяє відповісти на питання про те, чи дають два обраних цілих числа при діленні на m один і той же залишок. Будь-яке ціле число при діленні на m дає один з m можливих залишків: число від 0 до $m-1$.

Методом Евкліда вирішує цілий ряд питань, пов'язаних з обчисленням простих чисел за допомогою рівняння:

$$e \cdot d + (p-1)(q-1) \cdot y = 1 \tag{1.6}$$

Невідомими параметрами в даному випадку будуть наступні змінні типу d і y. Метод обчислення таких чисел знаходить ціле безліч пар типу (d, y), кожна з яких є одним з рішень даного рівняння. Два числа типу (e, n) – публікується у відкритому доступ і є відкритим ключем. Число d зберігається в найсуворішому секреті - це і є закритий ключ, який дозволить читати всі послання, зашифровані за допомогою пари чисел (e, n).

Пара $\{e, n\}$ публікується в якості відкритого ключа RSA

Пара $\{d, n\}$ грає роль закритого ключа RSA і тримається в секреті.

Припустимо, що Відправник хоче передати Одержувачу інформацію за допомогою алгоритму RSA. Відправник повинен знати відкритий ключ отримувача для шифрування вихідного тексту, а Одержувач повинен використовувати свій закритий ключ для розшифрування отриманого шифрованого тексту. Але для передач цих ключів обом з учасників вимагаєте канал зв'язку надійний, але не обов'язково секретний. Закритий ключ отримувача ніколи нікому не передається. І тут же постає один з головних проблем в системі RSA - це потреба в закритій передачі ключів.

1.5.1.2 Шифрування / Дешифрування

Шифрування інформації – процес маскування первинної інформації; процес перетворення доступних даних на зашифровані та управління доступом до даних. Процес маскування здійснюється за допомогою алгоритму та алфавіту – набору цифр та символів, розшифрування якого можливе після використання ключів шифрування. Шифрування інформації використовується для приховання інформації, зазвичай використовується у службах безпеки, банках та інших комерційних підприємствах, що містять конфіденційні дані, але використовується також при передачі даних у інформаційних системах.

Припустимо, що відправник реалізує відправлення повідомлення m із шифрування.

Повідомленнями є цілими числами в інтервалі від 0 до $n-1$, тобто $m \in Z_n$. Алгоритм реалізується як:

- Відкритий ключ одержувача для шифрування $\{e, n\}$
- Береться повідомлення m.
- Виконується шифрування первинного тексту за допомогою ключа [8]:

$$c = E\ (m) = \ m^e mod(n) \tag{1.7}$$

Дешифрування - зворотний процес шифрування. За допомогою ключа шифрований текст перетвориться у вихідний. Дешифрування повідомлення можливе тільки з використанням закритого ключа, який відомий тільки самому адресату. Криптографічні системи з відкритим ключем використовують так звані незворотні або односторонні функції.

Алгоритм:

- Отримується зашифроване повідомлення c
- Закритий ключ $\{d, n\}$
- Використовується закритий ключ для дешифрування повідомлення [8]:

$$m\ = \ D\ (c) = \ c^d mod(n) \tag{1.8}$$

Тобто алгоритм реалізації шифрування та дешифрування можливо зобразити як Рисунку 1.7.

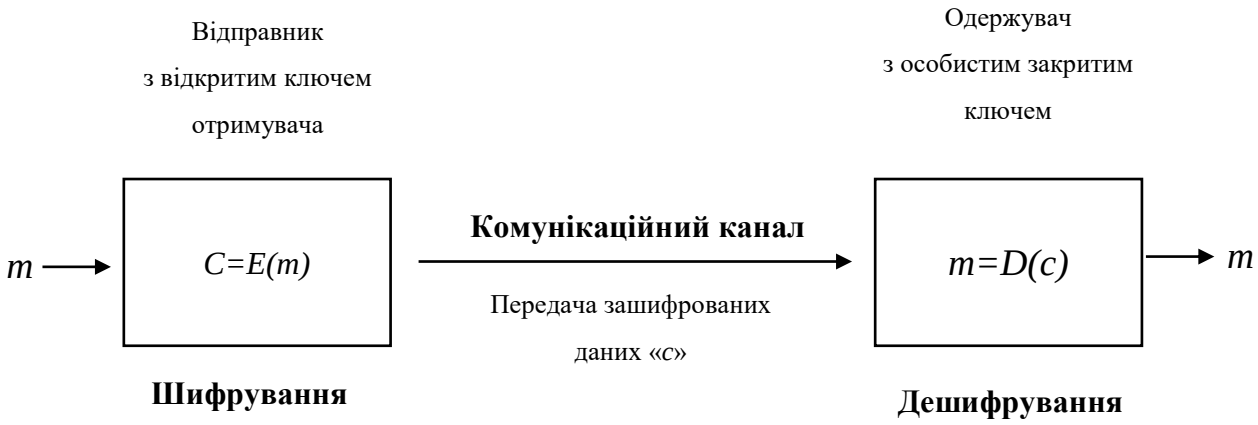


Рисунок 1.7 – Алгоритм роботи RSA

Стійкість алгоритму ґрунтується на складності обчислення зворотної функції до функції шифрування [8]:

$$c = E\ (m) = \ m^e mod(n) \tag{1.9}$$

Для обчислення m по відомим c, e, n потрібно знайти такий d, щоб

$$\mathrm{e} \cdot d \equiv 1\ \left(mod\varphi\ (n)\right) \tag{1.10}$$

тобто

$$d \equiv e - 1\ \left(mod\varphi\ (n)\right) \tag{1.11}.$$

Обчислення зворотного елемента по модулю не є складним завданням, проте зловмисникові невідомо значення функції Ейлера $\varphi\ (n)$. Для обчислення функції Ейлера від відомого числа n необхідно знати розкладання цього числа на прості множники. Знаходження таких множників і є складним завданням, а знання цих множників - «потайними дверми», яка використовується для обчислення d власником ключа. Існує безліч алгоритмів для знаходження простих співмножників, факторизації, найшвидший з яких на сьогоднішній день - загальний метод решета числового поля, швидкість якого для k-бітного цілого числа становить

$$exp\ \left(\left(c\ +\ o\ (1)\right)klogk\right) \text{ для деякого } c\ < 2 \tag{1.12}$$

У 2010 році групі вчених з Швейцарії, Японії, Франції, Нідерландів, Німеччини і США вдалося успішно обчислити дані, зашифровані за допомогою криптографічного ключа стандарту RSA довжиною 768 біт. Знаходження простих співмножників здійснювалося загальним методом решета числового поля. За словами дослідників, після їх роботи в якості надійної системи шифрування можна розглядати тільки RSA-ключі довжиною 1024 біта і більш. Причому від шифрування ключем довжиною в 1024 біт варто відмовитися в найближчі три-чотири роки. З 31 грудня 2013 року браузери Mozilla перестали підтримувати сертифікати що засвідчують центрів з ключами RSA менше 2048 біт.

Алгоритм RSA набагато повільніше, ніж AES (Advanced Encryption Standard) і інші алгоритми, що використовують симетричні блокові шифри.

При неправильної або неоптимальною реалізації або використання алгоритму можливі спеціальні криптографічні атаки, такі як атаки на схеми з малої секретної експонентою або на схеми із загальним обраним значенням модуля.

Через низьку швидкість шифрування (близько 30 кбіт/с при 512 бітному ключі на процесорі 2 ГГц), повідомлення зазвичай шифрують за допомогою більш продуктивних симетричних алгоритмів з випадковим сеансовим ключем (наприклад, IDEA, Serpent, Twofish), а за допомогою RSA шифрують лише цей ключ, таким чином реалізується гібридна криптосистема.

1.5.2 Алгоритм Ель-Гамаля

Асиметричний алгоритм, запропонований Ель-Гамалем вважається універсальний інструментом. Він виконує реалізує такі функції: шифрування даних, формування цифрового підпису та узгодження загального ключа. Також можливе впровадження модулів для схем перевірки пароля, докази ідентичності повідомлення і т.д. Безпека цього алгоритму, так само як і алгоритму Діффі-Хеллмана, ґрунтується на складності обчислення дискретних логарифмів. Цей алгоритм базується на схемі Діффі-Хеллмана, щоб сформувати загальний секретний ключ для абонентів, для передачі повідомлення, і потім повідомлення шифрується шляхом множення повідомлення на ключ.

Алгоритм реалізується шляхом піднесення до степеню за модулем великого простого числа P. І в разі шифрування, і в разі формування цифрового підпису кожному користувачеві необхідно згенерувати пару ключів. Обирається деяке велике просте число P і ціле число A, де зберігається відношення $1 < A < P$.

Повідомлення представляються цілими числами M з інтервалу $1 < M < P$.

Протокол передачі такого повідомлення M виглядає наступним чином: користувачі мають числа A і P і генерують випадкові великі числа: Ka, Kb, які не взаємодіють один-з-одним, і які задовольняють умові $1 < K < P$.

Далі обчислюється і передається відправнику число B, яке визначається послідовністю:

$$B = A^{Kb} mod(P) \tag{1.13}$$

Далі проводиться шифрування повідомлення M і відправляється отримана послідовність одержувачу:

$$C = M \cdot B^{Ka} \ mod(P) \tag{1.14}$$

Надалі розшифровується отримане повідомлення:

$$D = (A^{Ka})^{-Kb} \ mod(P), \qquad M = C \cdot D \ mod(P) \tag{1.15}$$

Для даного алгоритму шифрування ступінь захисту, що для алгоритму RSA з модулем N з 200 знаків, досягається вже при модулі P = 150 знаків. Це збільшує швидкість обробки інформації у 5-7. Але у даної системи відсутнє підтвердження автентичності повідомлень.

Криптостійкість даного алгоритму ідентична із системою захисту алгоритму RSA, а саме з модулем N з 200 знаків, досягається вже при модулі P з 150 знаків. Це дозволяє в 5-7 разів збільшити швидкість обробки інформації. Однак, в такому варіанті відкритого шифрування немає підтвердження автентичності повідомлень.

Для реалізації процедури підтвердження автентичності відправника і при цьому забезпечення при відкритому шифруванні по модулю простого числа P, як варіант був реалізований протокол передачі повідомлення M із підписом: абоненти знають числа A і P; у даний момент формується довільне число і зберігається без можливості доступу: Ka - задовольняє умові: $1 < Ka < P$ обчислює і передає одержувачу число B, яке визначається послідовністю:

$$B = A \, Ka \, mod \, (P) \tag{1.16}$$

Для повідомлення M $(1 < M < P)$: обрається випадкове число L $(1 < L < P)$, яке задовольняє умові $(L, P - 1) = 1$ обчислює число

$$R = A \, L \, mod \, (P) \tag{1.17}$$

Вираховується відношення щодо S

$$M = Ka * R + L * S \, mod \, (P) \tag{1.18}$$

передає підписане повідомлення $[M, R, S]$ і тоді провадиться перевірка коректності підпису:

$$A \, M = (B \, R) * (R \, S) mod \, (P) \tag{1.19}$$

Для даної системі із закритим ключем для підписування повідомлень є число X, а відкритим ключем і для реалізації ЕЦП виступає число B. Процедура перевірки підпису служить також і для перевірки коректного розшифрування, якщо повідомлення шифруються.

1.6 Криптоаналіз

Криптоаналіз - наука про методи знаходження вихідного змісту зашифрованою (прихованої) інформації, не маючи можливості отримання секретної інформації (ключу), необхідної для цього. У більшості випадків під цим мається на увазі злом шифру (коду). Вперше дане поняття було введено в 1920 році американським криптографом Вільямом Фрідманом [9].

Під поняттям «криптоаналіз» розуміється можливість знайти вразливі місця в криптографічному алгоритмі або протоколі. Не дивлячись на те що головною метою криптоаналізу залишилася незмінною з плином часу, способи криптоаналізу багато в чому змінились. Якщо раніше в професії криптоаналітика були, здебільшого, лінгвісти то в даний час основними криптоаналітиків вважаються математики. Криптографічний атака - це результат криптоаналізу обраного шифру. Якщо криптографічний атака була успішною, то таку атаку називають зломом або розкриттям.

Під стійкістю криптографічного алгоритму зазвичай розуміють кількість операцій, які необхідно виконати, щоб отримати секретний ключ використовуючи відкритий ключ. Від числа і характеру «елементарних» операцій безпосередньо, залежить час, необхідний для їх виконання.

1.6.1 Атака за часом

Атака за часом - це атака, яка використовується зловмисником в сторонніх каналах зв'язку, побудована на аналізі часу, який необхідний на виконання криптографічного алгоритму. Кожна обчислювальна операція вимагає певний час на виконання на персональному комп'ютері. Цей час може бути різним у залежності від представлених даних. Якщо зловмисник має точними вимірами часу для обраних операцій, то він може спробувати відновити вхідні дані [11].

Криптографічні алгоритми зазвичай витрачають різну кількість часу на обробку вхідних даних. Це може бути обумовлено не правильної оптимізацією обраного алгоритму, читанням даних з буфера обміну в оперативній пам'яті, командами процесора, які використовують різні часи для обчислення різних операцій. Характеристики продуктивності на пряму пов'язані як від розміру ключа шифрування, так і від вхідних даних.

1.6.2 Запобігання атаки за часом

Найбільш очевидний спосіб запобігти тимчасових атак - змоделювати алгоритм шифрування таким чином, що всі вироблені обчислення будуть виконуватися за рівний час. Однак створити такий ідеальний код досить складно, так як деякі обчислювальні процеси, такі як: читання з кеш, відгук системи, виконання потоків можуть сприяти появі тимчасових відхилень.

Інший спосіб полягає в тому, щоб зробити вимірювання часу настільки неточними, щоб атака стала неефективною. Наприклад, в програмному коді можна прописати тимчасові затримки, які матимуть випадкову довжину. Такий підхід є найбільш ефективним при боротьбі з таким видом атак [11].

1.6.3 Атака «Людина посередині»

Цей метод заснований на тому, що зловмисник підключається до каналу передачі даних, тим самим порушуючи криптографічний протокол. Він може активно втручатися в алгоритм передачі і видавати себе за одного з одержувачів. Так він може видаляти, змінювати і видавати неправдиву інформацію за дійсність. Припустимо, що користувач А намагається передати користувачеві В якусь зашифровану інформацію. Зловмисник С знає про структуру і властивості обраного методу шифрування і передачі даних. Для здійснення атаки на канал зв'язку зловмисник З представляється користувачеві А як користувач В і навпаки. Користувач А, не знаючи про це, намагається надіслати інформацію В, а насправді посилає її С. Об'єкт С, отримавши інформацію, і зробивши з неї деякі операції пересилає дані одержувачу В. Користувач В, в свою чергу, вважає, що інформація була отримана ним від користувача А [12].

1.6.4 Повний перебір

Повний перебір - метод вирішення задачі шляхом перебору всіх можливих варіантів. Складністю цього методу є кількість всіляких варіантів розв'язання задачі. Якщо кількість рішень занадто велике, то цей метод може не дати результатів у певного часу.

Оцінка криптостійкості шифрів якраз і ґрунтується на складності методу повного перебору рішень. В результаті шифр буде крипостійкість до атак якщо не буде виявлено алгоритм знаходження ключа за час менше ніж час, витрачений на повний перебір. Криптографічні атаки, які засновані на алгоритмі повного перебору, є найбільш універсальними, але дуже довгими.

1.6.5 Запобігання атаки «повного перебору»

Найкращим способом уникнути атак повного перебору є правильний вибір параметрів еліптичної кривої. Якщо використовувати в якості параметрів еліптичної кривої числа, довжини яких перевищують 512 біт, а в якості характеристики поля вибрати велике просте число то це може сприяти поліпшенню криптостійкості алгоритму. Звичайно ми втратимо трохи в часі, так як всі обчислення будуть виконуватися трохи довше, але дані обчислення будуть не настільки критичні, як небезпека злому шифру.

1.6.6 Квантовий злом

Найсерйознішою загрозою для сучасної криптографії є квантові комп'ютери і їх великі можливості.

Якщо модифікувати алгоритм Шора, щоб він міг використовуватися на квантових комп'ютерах, то можна без особливих зусиль вирішити проблему дискретного логарифмування. Отже, криптосистема, яка ґрунтується на еліптичних кривих буде під загрозою злому.

Але поки що це загроза не являється критичною, так як квантові комп'ютери зараз знаходяться лише в стадії розробки. Швидше за все, в найближчі десятиліття квантові комп'ютери, які можуть зламати вже існуючі шифри, не з'являться. Але навіть якщо і з'являться, то на даний момент вже активно розвивається такий розділ науки як квантова криптографія [13].

1.7 Висновки до розділу 1 і постановка задачі дослідження методів шифрування та їх криптографічної стійкості

У першому розділі було проведено аналіз існуючих методів шифрування та проведений первинний криптоаналіз їх складових. Розглянуто актуальність даної задачі та основні положення, щодо криптостійкості, реалізації методів шифрування та їх проблеми, котрі потрібно розкрити та надати можливі методи вдосконалення методів шифрування, шляхом виявлення надійних та економічно прийнятних і доцільних для використання методів.

Метою дослідження є підвищення криптостійкості вже існуючих алгоритмів шифрування за допомогою таких методів, як хеш-функції, ітерація ключів шифрування та інших. Необхідне подальше проведення порівняльної характеристики, для виявлення стійких методів, що задовольняють усім критеріям стійкості, економічність та складності реалізації.

Предметною областю даної магістерської роботи є криптографічні методи захисту інформації.

Об'єктом дослідження є криптографія.

Результатом дослідження має бути виявлення алгоритмів шифрування та набір методів для підвищення криптостійкості, котрі будуть задовольняти критеріям надійності, а саме стійкість до проведення атаки із знаходженням ключів шифрування чи первинного тексту і надійність ключа чи пари ключів шифрування. Стійкість та надійність методів буде визначена за допомогою аналітичних, математичних та практичних методів та запропоновані для подальшого використання.

РОЗДІЛ 2
ЗАСОБИ ПІДВИЩЕННЯ КРИПТОГРАФІЧНОЇ СТІЙКОСТІ ТА ЇХ РЕАЛІЗАЦІЯ

2.1 Цифровий підпис

Електронний цифровий підпис - це інформація, яка приєднується до іншої інформації (до підписується інформації) з метою визначення особи, яка підписала інформацію, а також факту незмінності інформації після її підписання. [14].

Даний інструмент використовується для таких цілей:

- гарантування цілісності та непохибності підписом, шляхом порівняння із наявним зразком;
- підтвердження авторства документа .

При реалізації підпису гарантується виконання властивостей:

- автентичність, тобто підтверджує особу власника підпису;
- доказовість приналежності власнику підпису, через неможливість підробки підпису;
- підпис є частиною документа і її неможливо перенести на інший документ;
- за наявності підпису документ не може бути відредагований;
- підпис незаперечна;
- електронно-цифровий підпис (ЕЦП) – елемент електронного документу, засобі доказу автентичності та цілісності документа. Він зберігає всі основні властивості звичайного підпису [15].

ЕЦП реалізується за допомогою таких методів:

- Реалізація шифрування електронного документа за допомогою симетричних алгоритмів. Реалізація проходить за допомогою третьої довіреної особи – арбітра. Авторизацією документа в даній схемі проводиться за допомогою шифрування документу секретним ключем і передача його арбітром.
- Використання асиметричних алгоритмів шифрування. Підписання документа за допомогою шифрування закритим ключем відправника.

Перший варіант став більш поширеною схемою реалізації ЕЦП - шифрування остаточного результату обробки електронного документу хеш-функцією за допомогою асиметричного алгоритму.

Існує багато методів реалізації електронного цифрового підпису такі, як груповий підпис, беззаперечний підпис, довірений підпис і ін. Ці методи можуть бути застосовані в залежності від завдання та степеню засекреченості, що вирішуються за допомогою електронних технологій передачі та обробки електронних документів.

2.2 Залежність криптостійкості від ключа

З розвитком систем шифрування та ЕЦП, тим рідше ідеться про незламність шифрів та «абсолютний захист» від атак, так як стає очевидним, що абсолютно незламних та захищених засобів шифрування не існує, тому питання криптоаналізу для систем лише складається зі швидкості злому, складності алгоритму шифрування та його економічної доцільності.

Якщо за основні метод злому брати атаку зі знанням початкового тексту, то знаходження закритого ключа не є великою проблемою, навіть дізнатися мето шифрування. Якщо навіть не брати до уваг спеціальні методи а спиратися тільки на методі звичайного перебору, то рішення цього завдання не є складним, це всього-лише тільки питання часу. І стійкість даного шифру залежить лише від обчислювальної техніки та розміром ключа [16].

Розмір ключа вимірюється в бітах і саме від його довжини залежить швидкість роботи алгоритму та тривалість злому даного ключа. Тому обрання довжини ключа залежить від мети застосування даного алгоритму шифрування. Наприклад, те, що може бути прийнятне для шифрування звичайного листування, не підходить для банківських операцій, або держслужб та служб пов’язаних із військовою безпекою.

Криптостійкість симетричних ключів вирахувати теоретично - не складно, якщо симетричний ключ не є надійним і його розмір становить приблизно 40 біт, тоді за допомогою звичайного перебору на доволі потужному комп’ютері буде займати біля доби, це досить затратний, але надійний метод. Якщо довжина ключа буде 64 біти, тоді для його знаходження необхідно мати кластер комп’ютерів і дана задача буде займати тижні, це дуже затратний метод у плані потужності та грошових затрат, але все ж практично можливий. Доволі крипостійким можна назвати шифрування із довжиною ключа 128 біт. На сьогодні вирахування подібного ключа займає декілька стонів років, тобто практично – неможливе та недоцільне, але все ж можливе, якщо мати деякі допоміжні дані, такі як засоби для ЕЦП та метод реалізації хеш-функції.

Для ключів несиметричного шифрування неможливо однозначно отримати дані про швидкість їх злому через складність вирахування самих ключів, тому для даного метода можливо казати тільки про теоретичну криптостійкість, але реальну їх стійкість можливо перевірити тільки дослідницьким методом і на практиці можливо дійти приблизно такого результату, а саме співставлення ключів шифрування у симетричному та несиметричному

шифруванні (Табл 2.1):

Таблиця 2.1 - Довжина симетричного і несиметричного ключів за однакового рівня безпеки, біт

Симетричний	Несиметричний
56	384
64	512
128	2304

Та найбільш докладно про теорії криптостійкість висунув голландець Керкгофф, який сформулював такі постулати, а саме, що під час розробки і застосування шифру треба зважати на те, що весь механізм шифрування, множина правил чи алгоритмів рано або пізно стає відомою опонентові, а стійкість шифру визначається тільки таємністю ключа, яка, у свою чергу, визначається його довжиною.

2.3 Підвищення криптостійкості за допомогою хеш-функції

Криптографічно стійка хеш-функція перетворює вихідний текст в один рядок так, що з цього рядка неможливо (або дуже складно) отримати вихідний текст. Хеш-функцію використовують, наприклад, там, де потрібно вводити пароль: фразу в рядку введення порівнюють не з самим паролем, а з його хеш-функцією.

Хеш-функція-перетворення тексту довільної довжини в текст фіксованої довжини визначається як:

$$H = hash(P) \tag{2.1}$$

, де P - пароль (відкритий текст), довжина P від 0 до нескінченності;

H-хеш (хешований текст), довжина $H = N$ біт (за умови що функція $hash$ повертає хеш-значення довжиною N біт) [17].

Для посилення криптографічної стійкості використовують ітераційний алгоритм отримання ключів.

Загальну схему ітераційного алгоритму можна представити таким чином представленим на Рисунку 2.1.

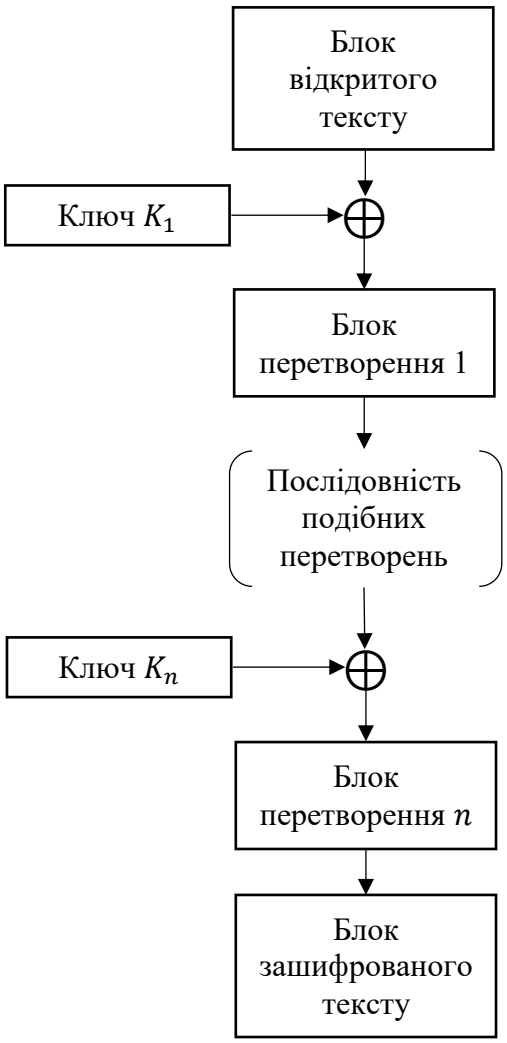


Рисунок 2.1 – Узагальнена схема ітераційних криптоалгоритмів

Ключі знаходяться за наступним алгоритмом:

$$\begin{gathered} K_1 = hash(PASSWORD) \\ K_2 = f(K_1) \\ \dots \\ K_n = f(K_{n-1}) \end{gathered} \tag{2.1}$$

, де f-функція перетворення ключа.

Якщо знати K_1, то можливо обчислити всі інші $K_i,\ i\ =\ 2..n.$

Зловмисник не знаючи первинного тексту (паролю), але якщо він має отриману конечну хеш-функцію, тоді можливо розшифрувати і отримати вхідний текст. Для підбору пароля методом жорсткого паролю, потрібно зробити підбір 2^s значень (S-розмір хеш-значення в бітах). Якщо хеш-функція має розмір у 64 біт, тоді потрібно зробити перебір 2^{64} значень.

Приймаємо, що комп'ютер в середньому може перебирати 1000000 паролів в секунду. Тоді знаходження первинного тексту за його хеш-значення максимально займе 213 503 982 днів.

Якщо використовувати інший метод злому, такий як пошуку колізій хеш-функцій, тоді кількість варіантів скоротиться в середньому в два раз, тобто виходить $2^{64/2} = 2^{32}$ значень і тоді на пошук відповідного хешу вийде всього лише приблизно 1,2 години.

2.4 Алгоритм DSA

DSA (Алгоритм цифрового підпису) - алгоритм з використанням відкритого ключа для створення електронного підпису, але не для шифрування. Секретне створення хеш-значення і можливість її публічної перевірки означає, що тільки один суб'єкт може створити хеш-значення повідомлення, але будь-хто може перевірити її коректність. Заснований на обчислювальної складності взяття логарифмів в кінцевих полях [18].

Цифровий підпис DSA - це пара великих чисел, в комп'ютерному поданні - бінарні рядки. Цифровий підпис обчислюється за певним алгоритмом (наприклад, DSA) з певними параметрами, такими, щоб можна було б перевірити цілісність даних і особистість, яка підписала повідомлення. DSA забезпечує можливість створення і перевірки підписів. Для створення цифрового підпису використовується закритий ключ, а для перевірки підпису - відкритий ключ, який відповідає (але не дорівнює) закритому ключу. Кожен користувач має парою ключів - відкритий і закритий ключ. Відкриті ключі вважаються загальнодоступними. А закриті ключі тримаються в секреті. Будь-хто може перевірити підпис користувача з використанням відкритого ключа даного користувача. Саму підпис може створити тільки володар закритого ключа.

Підписується не все повідомлення, а дайджест повідомлення (стисла версія підписуються даних, хеш-значення). Дайджест повідомлення є вхідним параметром алгоритму DSA. Цифровий підпис відправляється призначеному верифікатори разом з підписаними даними (часто називають повідомленням). Верифікатор перевіряє підпис повідомлення, використовуючи відкритий ключ відправника. У процесі перевірки повинна використовуватися та ж хеш-функція, що і у відправника. Хеш-функція задається в окремому стандарті, Secure Hash Standard (SHS), FIPS 180 (стандарт описує алгоритм SHA) [18].

Для підписування повідомлень необхідна пара ключів - відкритий і закритий. При цьому закритий ключ повинен бути відомий тільки тому, хто підписує повідомлення, а відкритий - будь-кому перевірити справжність повідомлення. Також загальнодоступними є параметри самого алгоритму.

1. Вибір хеш-функції $H(x)$. Для використання алгоритму необхідно, щоб підписується повідомлення було числом. Хеш-функція повинна перетворити будь-яке повідомлення в число

2. Вибір великого простого числа q, розмірність якого в бітах збігається з

розмірністю в бітах значень хеш-функції $H\ (x)$

3. Вибір простого числа p, такого, що $(p-1)$ ділиться на q. Розмірність p задає криптостойкость системи. Раніше рекомендувалася довжина в 1024 біта. В даний момент для систем, які повинні бути стійкими до 2010 (2030) року, рекомендується довжина в 2048 (3072) біта.

4. Вибір числа g такого, що його мультиплікативний порядок по модулю $p = q$. Для його обчислення можна скористатися формулою $g = h^{(p-1)/q}\ mod\, p$, де h - деяке довільне число, $h \in (1;\ p-1)$ таке, що $g \neq 1$. У більшості випадків значення $h = 2$ задовольняє цій вимозі.

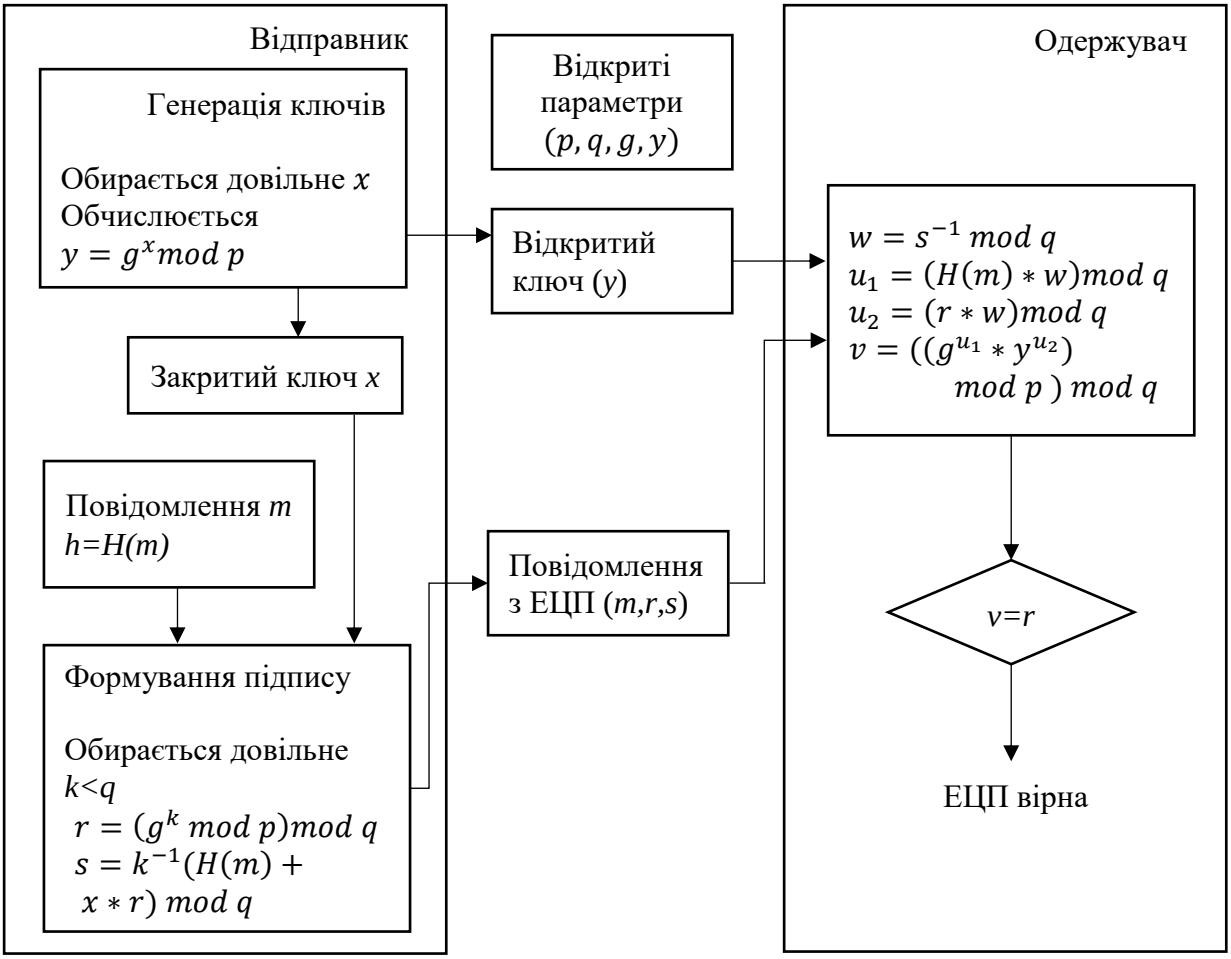


Рисунок 2.2 – Схема роботи алгоритму DSA

Процес підпису [18]:

1. Вибір випадкового числа $k \in (0, q)$
2. Обчислення $r = (g^k\ mod\ p) mod\ q$
3. Вибір іншого k, якщо $r = 0$
4. Обчислення $s = k^{-1}(H(m) +\ x * r)\ mod\ q$
5. Вибір іншого k, якщо $s = 0$
6. Підписом є пара (r, s)загальної довжини $2N$

Обчислювально складні операції це зведення в ступінь по модулю (обчислення $g^k \ mod \ p$), для якого існують швидкі алгоритми , обчислення хешу $H(x)$, де складність залежить від обраного алгоритму хешування і розміру вхідного повідомлення, і знаходження зворотного елемента $k^{-1} \ mod \ q$ використовуючи, наприклад, розширений алгоритм Евкліда або малу теорему Ферма у вигляді $k^{-1} \ mod \ q = \ k^{q-2} \ mod \ q$.

Процес перевірки відпису [18]:

1. Обчислення $w = s^{-1} \ mod \ q$
2. Обчислення $u_1 = (H(m) * w) mod \ q$
3. Обчислення $(r * w) mod \ q$
4. Обчислення $v = ((g^{u_1} * y^{u_2}) \ mod \ p \ ) \ mod \ q$
5. Підпис вірний, якщо $v = r$

При перевірці обчислювально складні операції це два зведення в ступінь g^{u_1} та y^{u_2}, обчислення хешу $H(x)$ та знаходження зворотного елементу $s^{-1} \ mod \ q$.

Будь-яку атаку на алгоритм можна описати так: зловмисник отримує все відкриті параметри підписи і якийсь набір пар (повідомлення, підпис) і намагається, використовуючи цей набір, створити дійсну підпис для нового повідомлення, що не представленого в наборі.

Ці атаки можна умовно розділити на дві групи - по-перше, зловмисник може спробувати відновити секретний ключ x, і тоді він відразу отримує можливість підписати будь-яке повідомлення, по-друге, він може спробувати створити дійсну підпис для нового повідомлення без прямого відновлення секретного ключа.

2.5 Алгоритм DES

Стандарт шифрування даних DES (DATA ENCRYPTION STANDARD) - блоковий шифр із симетричними ключами, розроблений Національним Інститутом Стандартів і Технології.

Прийняття стандарту шифрування DES стало потужним поштовхом до широкого застосування шифрування в комерційних системах. Введення цього стандарту - відмінний приклад уніфікації та стандартизації засобів захисту.

Для шифрування DES приймає 64-бітовий відкритий текст і породжує 64-бітовий зашифрований текст і навпаки, отримавши 64 біта зашифрованого тексту, він видає 64 біта розшифрованого. В обох випадках для шифрування і дешифрування застосовується один і той же 56-бітовий ключ [19] (Рисунок 2.3).

Рисунок 2.3 – Схема шифрування алгоритму DES

Процес шифрування складається з двох перестановок, які називають початковій і фінальній (кінцевої) перестановками, і 16 раундів Фейстеля. Кожен раунд використовує різні згенеровані 48-бітові ключі.

І кожний раунд реалізується як на Рисунку 2.4:

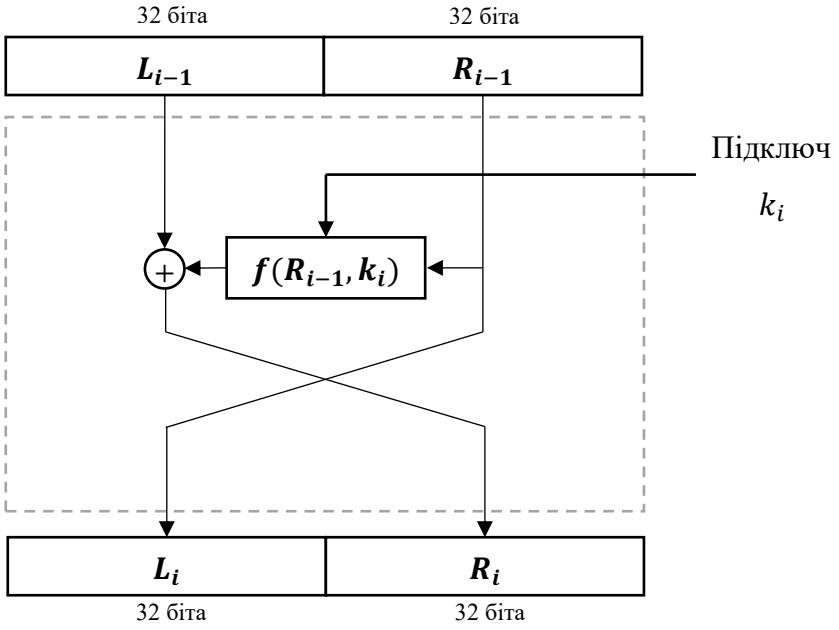


Рисунок 2.4 – Схема шифрування алгоритму DES

Раунд приймає напівблоки L_{i-1} та R_{i-1} від попереднього раунду (або початкового блоку перестановки) і створює напівблоки L_i та R_i для входу в наступний раунд (або кінцевий блок перестановки). Всі незворотні елементи зосереджені в функції $f(R_{i-1}, k_i)$.

DES створює 16 раундів ключів і k по 48 бітів з ключа k шифру на 56 бітів. Однак, щоб задати ключ шифру треба серед 56 бітів ключа додатково вписати 8 бітів в позиції 8,16, ..., 64 для перевірки парності таким чином, щоб кожен байт містив непарне число одиниць. За допомогою цієї операції виявляють помилки при обміні і зберіганні ключів.

З самого початку використання алгоритму криптоаналітики усього світу докладали багато зусиль для злому DES. Фактично DES дав небачений досі поштовх розвитку криптоаналізу. Вийшли сотні праць, присвячених різним методам криптоаналізу саме в додатку до алгоритму DES, а також деталей самого алгоритму і їх впливу на криптостійкість. Можна стверджувати, що саме завдяки DES з'явилися цілі напрямки криптоаналізу, такі як:

- лінійний криптоаналіз - аналіз залежності між відкритим текстом і шифротексту;

- диференційний криптоаналіз - аналіз залежності між співвідношеннями двох або більше відкритих текстів і відповідних їм шифротексту;

- криптоаналіз на пов'язаних ключах - пошук і аналіз залежності між шифротексту, отриманими на шуканому ключі і ключах, пов'язаних передбачуваним співвідношенням з шуканим ключем; проте DES виявився невразливий до даного виду атак, так як по ключовими розкладом циклічний зсув бітів ключа виконується на різну кількість позицій в різних раундах [20].

DES стійко витримав 20 років масового всесвітнього криптоаналізу - десятиліття криптоаналізу не привели до виявлення серйозних 14 вразливостей в алгоритмі. Основними результатами зусиль по злому DES можна вважати обчислення ключу шифру за допомогою методу лінійного криптоаналізу при наявності у атакуючого 247 пар «відкритий текст - шифротекст» або атаку, в якій ключ шифрування обчислювався методом диференціального криптоаналізу за умови, що атакуючий має 247 спеціально обраних пар «відкритий текст - шифротекст ».

Надалі ці атаки були кілька посилені (наприклад, атака лінійним Криптоаналіз при наявності 243 пар замість 247), з'являлися також нові види атак на DES (наприклад, атака, що дозволяє обчислити ключ високоточним опроміненням апаратного шифратора і подальшим аналізом помилок шифрування).

Однак, всі ці атаки вимагають наявності величезної кількості пар «відкритий текст - шифротекст», отримання яких на практиці є настільки трудомісткою операцією, що найбільш простий атакою на DES все ще можна вважати повний перебір ключів.

2.6 Алгоритм MD5

Хеш-функція призначена для згортки вхідного масиву будь-якого розміру в бітову рядок, для MD5 довжина вихідний рядки дорівнює 128 бітам. Хеш-функції використовуються, наприклад є два масиви, необхідно швидко порівняти їх на рівність, то хеш-функція може зробити це за вас, якщо у двох масивів хеші різні, то масиви гарантовано різні, а в разі рівності хеш - масиви швидше за все рівні [21].

Однак найчастіше хеш-функції використовуються для перевірки унікальності пароля, файлу, рядка і т.д. Наприклад, завантажуючи файл з інтернету або файлів архівів, часто можна побачити поруч з ним рядок виду b10a8db164e0754105b7a99be72e3fe5 - це і є хеш, прогнавши цей файл через алгоритм MD5 отримаєте такий рядок, і, якщо хеші рівні, можна з великою ймовірністю стверджувати що цей файл дійсно справжній (звичайно з деякими застереженнями, про буде ітись далі).

Використання MD5 дозволяє порівняти дайджест повідомлення з опублікованими, щоб переконатися, що дане повідомлення повністю збігається з оригінальним, тобто, не було пошкоджено або змінено [21].

Дана процедура порівняння називається "перевірка хеша" (hashcheck).

Найчастіше хеш-функції використовуються для перевірки унікальності пароля, файлу, рядка і тд. Наприклад, завантажуючи файл з інтернету, можна побачити поруч з ним рядок виду b10a8db164e0754105b7a99be72e3fe5 - це і є хеш, прогнавши цей файл через алгоритм MD5 ви отримаєте подібну рядок, і, якщо хеші рівні, можна з великою ймовірністю стверджувати що цей файл дійсно справжній.

Перевірочна сума MD5 (MD5 checksum) порожнього повідомлення:

$$d41d8cd98f00b204e9800998ecf8427e$$

На вхід алгоритму надходить вхідний потік даних, хеш якого необхідно знайти. Довжина повідомлення може бути будь-який (в тому числі нульовою). Запишемо довжину повідомлення в L. Це число ціле і не негативна. Кратність будь-яким числам не обов'язкова. Після надходження даних йде процес підготовки потоку до обчислень.

Алгоритм складається з п'яти кроків:

1) Append Padding Bits

У вихідний рядок дописують одиничний байт 0x80, а потім дописують нульові біти, до тих пір, поки довжина повідомлення не буде порівнянна з 448 по модулю 512. Тобто дописуємо нулі до тих пір, поки довжина нового повідомлення не буде дорівнює $L = (512 * N + 448)$,

де L – довжина повідомлення,

N - будь-яке натуральне число, таке, що цей вислів буде найближче до довжини блоку [22].

2) Append Length

Далі в повідомлення дописується 64-бітове представлення довжини вихідного повідомлення.

3) Initialize MD Buffer

На цьому кроці ініціалізується буфер.

$word\ A$ 01 23 45 67

$word\ B$: 89 $ab\ cd\ ef$

$word\ C$: $fe\ dc\ ba$ 98

$word\ D$: 76 54 32 10

Як можна помітити буффер складається з чотирьох констант, призначений для збору хешу.

4) Process Message in 16-Word Blocks

На четвертому кроці в першу чергу визначається 4 допоміжні логічні функції, які перетворять вхідні 32-бітові слова, в, як не дивно, в 32-бітові вихідні, які реалізують раунди шифрування [21].

1) $function\ F\ (X, Y, Z) = (X \cap Y) \cup (\bar{X} \cap Z)$

2) $function\ G\ (X, Y, Z) = (X \cap Z) \cup (\bar{Z} \cap Y)$

3) $function\ H\ (X, Y, Z) = X \oplus Y \oplus Z$

4) $function\ I\ (X, Y, Z) = Y \oplus (\bar{Z} \cup X)$

Також на цьому етапі реалізується так званий «білий шум» - посилення алгоритму, що складається 64 елементного масиву, що містить псевдовипадкові числа, залежні від синуса числа і:

$$T\ [i] = int(2^{32} * |\sin(i)|) \quad (2.2)$$

Далі починається «магія». Копіюємо кожен 16-бітний блок в масив $X\ [16]$ і виробляємо маніпуляції:

$AA = A$

$BB = B$

$CC = C$

$DD = D$

Вирівняні дані розбиваються на блоки (слова) по 32 біта, і кожен блок проходить 4 раунди з 16 операторів. Всі оператори однотипні і мають вигляд: $[abcd\ ksi]$, який визначається як:

$$A = B + \left((A + function\ (B, C, D) + X\ [k] + T\ [i]) <<< s\right) \quad (2.3)$$

, де A, B, C, D - регістри

$function\ (B, C, D)$ - одна з логічних функцій

$X\ [k]$ - k-тий елемент 16-бітного блоку.

$T\ [i]$ - i-тий елемент таблиці «білого шуму»

$<<<\ s$ - операція циклічного зсуву на s позицій вліво [22].

Тобто кожний раунд має вигляд (Рис. 2.5):

Раунд 1

```
/*[abcd k s i] a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s). */
[ABCD  0 7  1][DABC  1 12  2][CDAB  2 17  3][BCDA  3 22  4]
[ABCD  4 7  5][DABC  5 12  6][CDAB  6 17  7][BCDA  7 22  8]
[ABCD  8 7  9][DABC  9 12 10][CDAB 10 17 11][BCDA 11 22 12]
[ABCD 12 7 13][DABC 13 12 14][CDAB 14 17 15][BCDA 15 22 16]
```

Раунд 2

```
/*[abcd k s i] a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s). */
[ABCD  1 5 17][DABC  6 9 18][CDAB 11 14 19][BCDA  0 20 20]
[ABCD  5 5 21][DABC 10 9 22][CDAB 15 14 23][BCDA  4 20 24]
[ABCD  9 5 25][DABC 14 9 26][CDAB  3 14 27][BCDA  8 20 28]
[ABCD 13 5 29][DABC  2 9 30][CDAB  7 14 31][BCDA 12 20 32]
```

Раунд 3

```
/*[abcd k s i] a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s). */
[ABCD  5 4 33][DABC  8 11 34][CDAB 11 16 35][BCDA 14 23 36]
[ABCD  1 4 37][DABC  4 11 38][CDAB  7 16 39][BCDA 10 23 40]
[ABCD 13 4 41][DABC  0 11 42][CDAB  3 16 43][BCDA  6 23 44]
[ABCD  9 4 45][DABC 12 11 46][CDAB 15 16 47][BCDA  2 23 48]
```

Раунд 4

```
/*[abcd k s i] a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s). */
[ABCD  0 6 49][DABC  7 10 50][CDAB 14 15 51][BCDA  5 21 52]
[ABCD 12 6 53][DABC  3 10 54][CDAB 10 15 55][BCDA  1 21 56]
[ABCD  8 6 57][DABC 15 10 58][CDAB  6 15 59][BCDA 13 21 60]
[ABCD  4 6 61][DABC 11 10 62][CDAB  2 15 63][BCDA  9 21 64]
```

Рисунок 2.5 – Раунди MD5

Ну і в кінці підсумовуємо результати обчислень:

$A = A + AA$
$B = B + BB$
$C = C + CC$
$D = D + DD$

5) Output

Виводячи побайтово буфер ABCD починаючи з A і закінчуючи D отримаємо наш хеш.

Після закінчення циклу необхідно перевірити, чи є ще блоки для обчислень. Якщо так, то переходимо до наступного елементу масиву $(i + 1)$ і повторюємо цикл.

Цей алгоритм створює хеш і зміна хоча б одного елементу чи символу у первинному тексті кардинально змінює кінцевий результат. Але є один нюанс, це виникнення колізій.

Колізія хеш-функції – виникнення у пари незалежних об’єктів однакового хешу, тобто $H(x) = H(y)$

Колізії існують майже для будь-яких хеш-функцій, але від якості хеш-функцій частота їх виникнення приблизно мінімальна. У деяких окремих випадках, коли безліч різних вхідних даних звичайно, можна задати ін'єкційних хеш-функцію, за визначенням не має колізій.

2.7 Алгоритм SHA-1

Алгоритм SHA-1 (Secure Hash Algorithm Version 1 - безпечний алгоритм хешування, версія 1, був розроблений в далекому 1995 році. Деякі слабкі сторони SHA-1 були виявлені ще в 2005 році. Для вхідного повідомлення довільної довжини (до 2 ексабайт) алгоритм генерує 160-бітове хеш-значення (дайджест повідомлення) [23].

Алгоритм отримує на вході повідомлення максимальної довжини 264 біт і створює в якості виходу дайджест повідомлення довжиною 160 біт (Рис. 2.6).

Алгоритм складається з наступних кроків [23]:

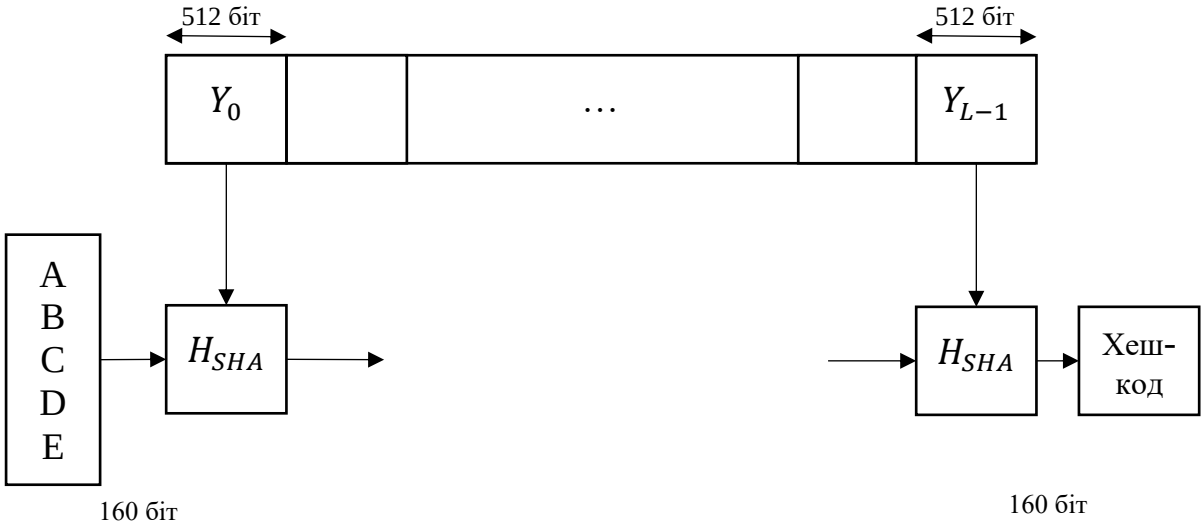


Рисунок 2.6 – Логіка виконання SHA-1

1) Додавання відсутніх бітів і вказівка довжини

Весь вихідний текст розбивається на блоки по 512 біт. У разі якщо довжина вихідного тексту не кратна 512 бітам, проводиться його вирівнювання за рахунок додавання в кінець біта зі значенням 1, m нульових бітів і 64-бітного представлення значення довжини вихідного повідомлення

2) Ініціалізація буфера

В алгоритмі використовується 160-бітний буфер для зберігання проміжних і остаточних результатів хеш-функції. Ініціалізується п'ять 32-бітових робочих змінних A, B, C, D, E:

$$A = 67\ 45\ 23\ 01_{16};$$
$$B = EF\ CD\ AB\ 89_{16};$$
$$C = 98\ BA\ DC\ FE_{16};$$
$$D = 10\ 32\ 54\ 76_{16};$$
$$E = C3\ D2\ E1\ F0_{16}.$$

Вектор ініціалізації, що подається на вхід 1-го раунду результат конкатенації

$$SHA_0 = A \| B \| C \| D \| E \tag{2.4}$$

3) Обробка повідомлення в 512-бітних блоках

Виконується обробка чергових 512 біт вихідного тексту. Для цього значення змінних A, B, C, D, E копіюються в змінні a, b, c, d, e і далі для t від 1 до 80 виконується перетворення значень даних змінних за схемою, зображеної на Рисунку 2.7.

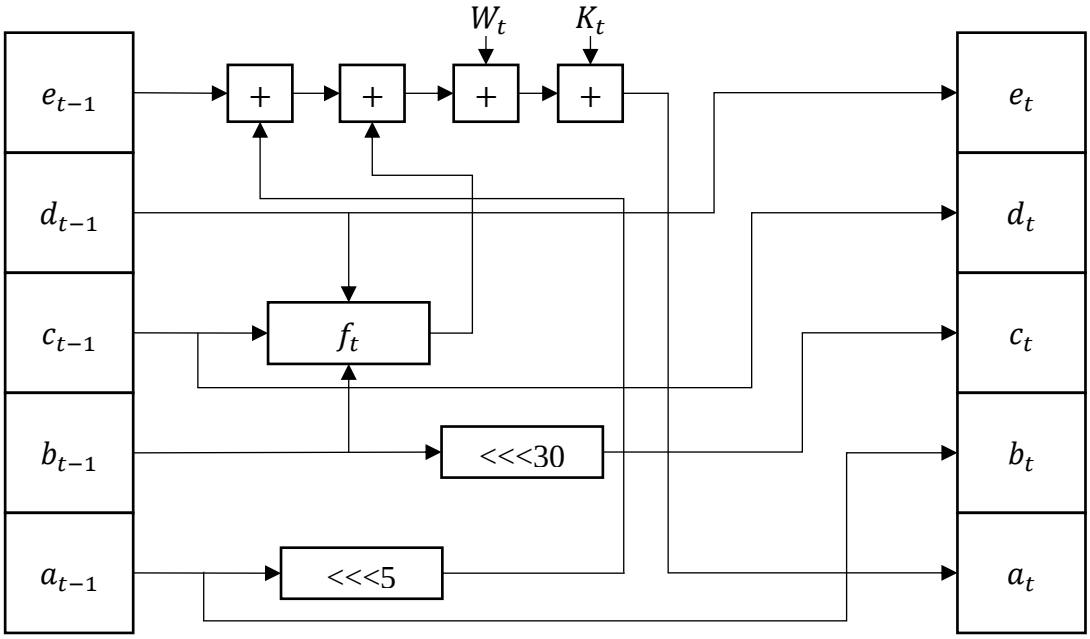


Рисунок 2.7 – Схема ітерації алгоритму SHA-1

Кожна з 80 ітерацій може бути записана наступним чином:

$TMP \leftarrow (a <<< 5) + f_t(b, c, d) + e + W_t + K_t;$

$e \leftarrow d;$

$d \leftarrow c;$

$c \leftarrow (b <<< 30);$

$b \leftarrow a;$

$a \leftarrow TMP;$

, де [+] - операція додавання за модулю 2^{32},

$f_t(X,Y,Z)$ - нелінійна функція, що має такий вигляд:

$$f_t(X,Y,Z)\begin{cases}(X\cap Y)\cup(\bar{X}\cap Z), & 0\le t<20;\\ X\oplus Y\oplus Z, & 20\le t<40;\\ (X\cap Y)\cup(X\cap Z)\cup(Y\cap Z), & 40\le t<59;\\ X\oplus Y\oplus Z, & 60\le t\le 79;\end{cases} \quad (2.5)$$

Параметр K_t приймає чотири різних значення в залежності від номера поточної ітерації:

$K_t = 5A82799916, \quad 0 < t < 19;$

$K_t = 6ED9EBA116, \quad 20 < t < 39;$

$K_t = 8F1BBCDC16, \quad 40 < t < 59;$

$K_t = CA62C1D616, \quad 60 < t < 79;$

W_t - одне з шістнадцяти 32-бітних слів 512-бітного блоку повідомлення при $0 < t < 15$ або значення, яке визначається відповідно до наступним виразом при $15 < t < 80$

$$W_t = (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16} <<< 1 \quad (2.6)$$

Значення змінних a, b, c, d, e незалежно один від одного складаються по модулю 2^{32} зі значеннями змінних A, B, C, D, E, в які потім і поміщаються отримані результати.

Етап 3 виконується до тих пір, поки не буде оброблений весь текст.

Після обробки останнього блоку тексту значення хеш-образу формується як $ABCDE$.

Криптоаналіз хеш-функцій спрямований на дослідження уразливості до різного виду атакам. Основні з них:

- знаходження колізій - ситуація, коли у пар повідомлень знаходиться одне і те ж хеш-значення.
- знаходження прообразу - вихідного повідомлення - по його хешу.

При вирішенні методом «грубої сили»:

- перше завдання вимагає в середньому $2^{160/2} = 2^{80}$ операцій, якщо використовувати атаку Днів народження.
- друга вимагає 2^{160} операцій.

Від стійкості хеш-функції до знаходження колізій залежить безпека електронного цифрового підпису з використанням даного хеш-алгоритму

2.8 Аналіз методів підвищення криптографічного стійкості

Проведемо аналіз найбільш поширених хеш-функцій.

Для пароля «Password» були отримані наступні хеш-значення (табл. 2):

Таблиця 2.2 - Отримання хешу для паролю «Password» за допомогою хеш-функцій

№ досліду	Хеш-функція	Хеш-значення
1	SHA-1	8be3c943b1609fffbfc51aad666d0a04adf83c9d
2	SHA-256	e7cf3ef4f17c3999a94f2c6f612e8a888e5b1026878e4e19398b23bd38ec221a
3	SHA-3	690ace5d07a99566b98f337abba767367756f78bc0a867c924b1f99c4bc469473a0cf69f108b329b44887a32abc23254d253d30db62538f0ebe72b0dabbed0c1
4	SHA-512	e6c83b282aeb2e022844595721cc00bbda47cb24537c1779f9bb84f04039e1676e6ba8573e588da1052510e3aa0a32a9e55879ae22b0c2d62136fc0a3e85f8bb
5	MD2	9dc7dd5f9b5f681a133e64c0089330e7
6	MD4	f15abd57801840f3348ddccafb677f6a
7	MD5	dc647eb65e6711e155375218212b3964
8	CRC-32	9abfd710
9	BASE-64	UGFzc3dvcmQ=
10	Tiger-128	78db95bcce175ab632095962774965d1
11	Whirlpool	fd07ba63996cdcfa6130ee82acec65da7487f51564bb7c6ead6dabc6b9e8eac974e5d852edc545804ae68fa46fc59d4789acab50bbc22b26fb24412f8dc11cde
12	DES(Unix)	Q37hnXsCsGjCU

Алгоритми CRC-32, Base-64, DES (Unix) використовувати не бажано, так як згенеровані хеш-функції замалий хеш, тобто дуже легко піддаються злому. Про алгоритми MD2 і MD4 не являються актуальними, як і попередні версії алгоритму SHA. Алгоритм Whirlpool попри те, що є надійним, є недоліки при обмеженій кількості «раундів» шифрування.

З претендентів на реальне використання можна розглядати такі алгоритми серії SHA-512, MD5 та Whirpool.

2.9 Висновок до розділу 2

У другому розділі оглянуто методи підвищення криптографічної стійкості методів шифрування та проведено первину оцінку засобів підвищення стійкості до методів криптоаналізу. На підставі отриманих даних було отримано наступні висновки:

При великій кількості засобів шифрування їх стійкість варіюється від складності алгоритму, тому допоміжних методів, що є доповненням до основного алгоритму шифрування, такі як електронно-цифровий підпис, хеш-функції та робота із ключами шифрування і за допомогою цих інструментів кардинально посилюється криптостійкість на відміну від первинного.

За допомогою проведеного аналізу можна виділити такий інструмент, як хеш-функції, а саме алгоритми серій SHA-512, Whirlpool та MD5, що значно посилюють криптостійкість та зберігають цілісність даних. Для більш детального аналізу будуть проведені розрахункові та аналітичні методи перевірки криптостійкості.

РОЗДІЛ 3
РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ

3.1 Криптоаналіз алгоритму MD5

Основні вимоги, що пред'являються до криптографічних хеш-функцій:

- хеш-функція використовується до повідомлення довільного розміру;
- велика швидкість обчислення хешу;
- від знання хеш-функції все одно майже неможливо знайти первинний текст, тобто прообраз P;
- низький поріг виникнення колізії, тобто збіг значення хешу у парі різних повідомлень;
- при відомому повідомленні P має бути важко знайти інше повідомлення P з таким же значенням хеш-функції [24];

MD5 (англ. Message Digest 5) - 128-бітний алгоритм хешування.

Використання MD5 дозволяє порівняти дайджест повідомлення з опублікованими, щоб переконатися, що дане повідомлення повністю збігається з оригінальним, тобто, не було пошкоджено або змінено.

Дана процедура порівняння називається "перевірка хеша" (hashcheck) [29].

Найчастіше хеш-функції використовуються для перевірки унікальності пароля, файлу, рядка і т.д. Наприклад, завантажуючи файл з інтернету, можна побачити поруч з ним рядок виду b10a8db164e0754105b7a99be72e3fe5 - це і є хеш, прогнавши цей файл через алгоритм MD5 буде отриманий подібний рядок, і, якщо хеші рівні, можна з великою ймовірністю стверджувати що цей файл дійсно справжній

Перевірочна сума MD5 (MD5 checksum) порожнього повідомлення:

d41d8cd98f00b204e9800998ecf8427e

Для проведення подальшого криптоаналізу необхідно реалізувати механізм роботи MD5, за основу буде братися розробка програмного забезпечення на мові програмування C#.

Для реалізації шифрування MD5 та інших існує бібліотека «System.Security.Cryptography»

Простір імен System.Security.Cryptography надає криптографічні служби, що включають безпечне кодування і декодування даних, а також цілий ряд інших функцій, таких як хешування, генерація випадкових чисел і перевірка справжності повідомлень.

Головним аспектом для даної задачі буде клас MD5CryptoServiceProvider, що обчислює значення хеша MD5 для вхідних даних за допомогою реалізації, що надається постачальником служб шифрування Crypto Service Provider. Хеш функції відображають бінарні рядки довільної довжини в короткі виконавчі рядки фіксованої довжини.

Також використовується метод ComputeHash() класу MD5CryptoServiceProvider, що обчислює хеш-значення для заданого об'єкту.

За допомогою даних інструментів була реалізована програма (Додаток А)

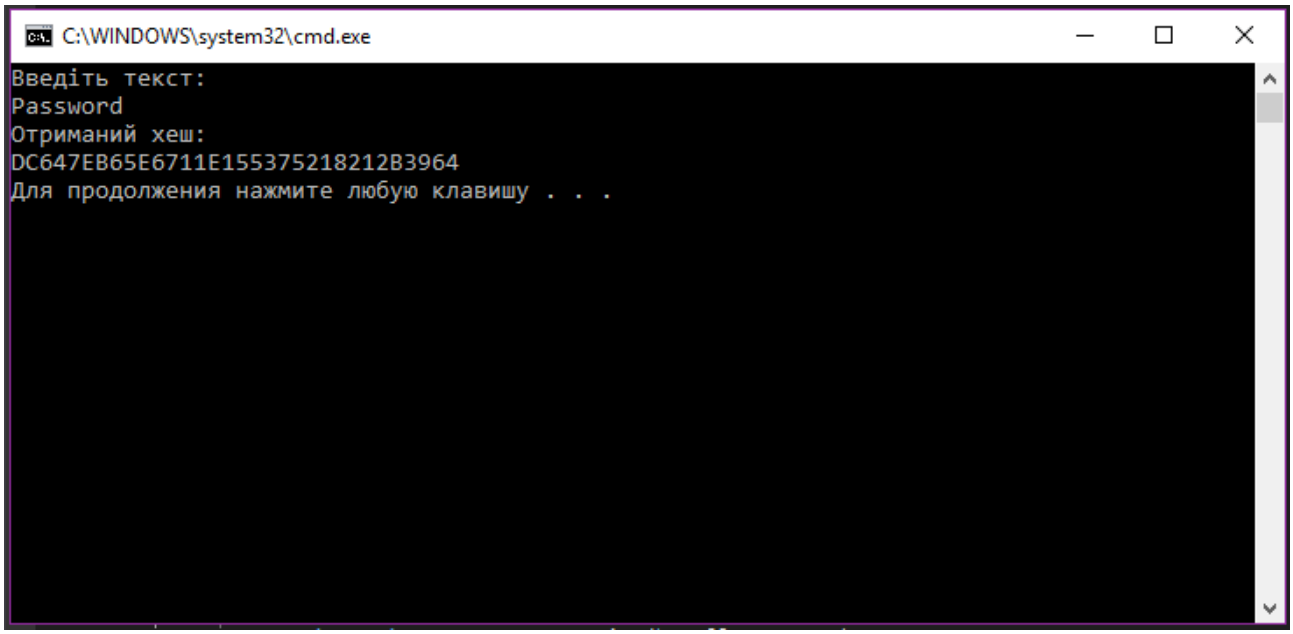


Рисунок 3.1 – Реалізація програми для обчислювання хешу за алгоритмом MD5

На даний момент існують кілька видів «злому» хешу MD5 - підбору повідомлення із заданим хешем;

- Перебір по словником;
- Повний перебір (Brute-force attack);
- Злом за райдужними таблицями(Rainbow attack).

У даному дослідженні розглядаються всі три способи злому коду MD5 за допомогою програмного забезпечення PasswordsPro v. 3.1.2.2.

PasswordsPro - програма, призначена для відновлення паролів за їхніми хешами. Розробник InsidePro Software. Програма підтримує кілька видів атак: атака повним перебором, атака по масці, проста атака по словниках, комбінована атака по словниках, атака по Rainbow-таблицях.

Було обрано декілька «паролів» і згенеровано з них хеш, після цього вони були збережені у текстовому файлі для подальшого опрацювання за допомогою програми PasswordsPro.

Таблиця 3.1 - Паролі та отримані хеші для них, за алгоритмом MD5

Пароль	Хеш
my_password	a865a7e0ddbf35fa6f6a232e0893bea4
0000	4a7d1ed414474e4033ac29ccb8653d9b
8805553535	c36ce47f1dbe88efa9655e5a7bb1286b
qwerty	d8578edf8458ce06fbc5bb76a58c5ca4
1q2w3e4r5t6y7u8i9o0p	c6b419d72d1664762ad3d5c500566c69
0123456789	781e5e245d69b566979b86e28d23f2c7
qwerty123456	48474f975022f960bc2afbe49be581e8
tRvVW50h	47e9458cd6a0398095c38925c249d2fd

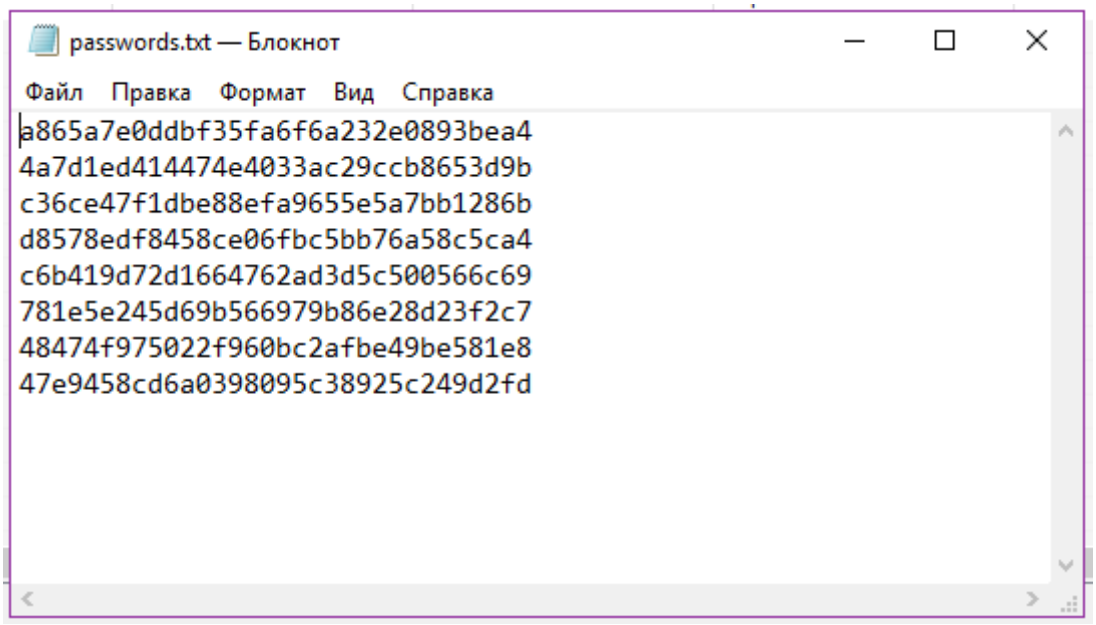


Рисунок 3.2 – Збережені хеші отримані за алгоритмом MD5

Надалі вони імпортуються до програми PasswordsPro для подальшої обробки (Рис 3.3-3.5).

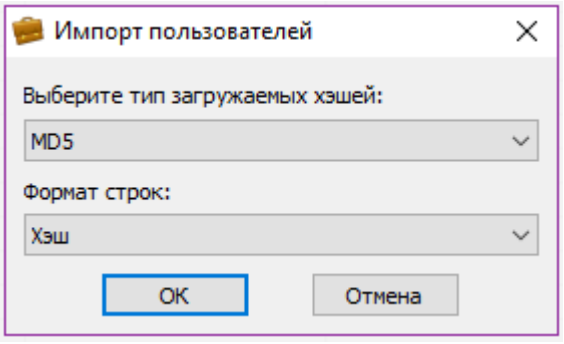


Рисунок 3.3 – Імпорт хешів до програми PasswordsPro

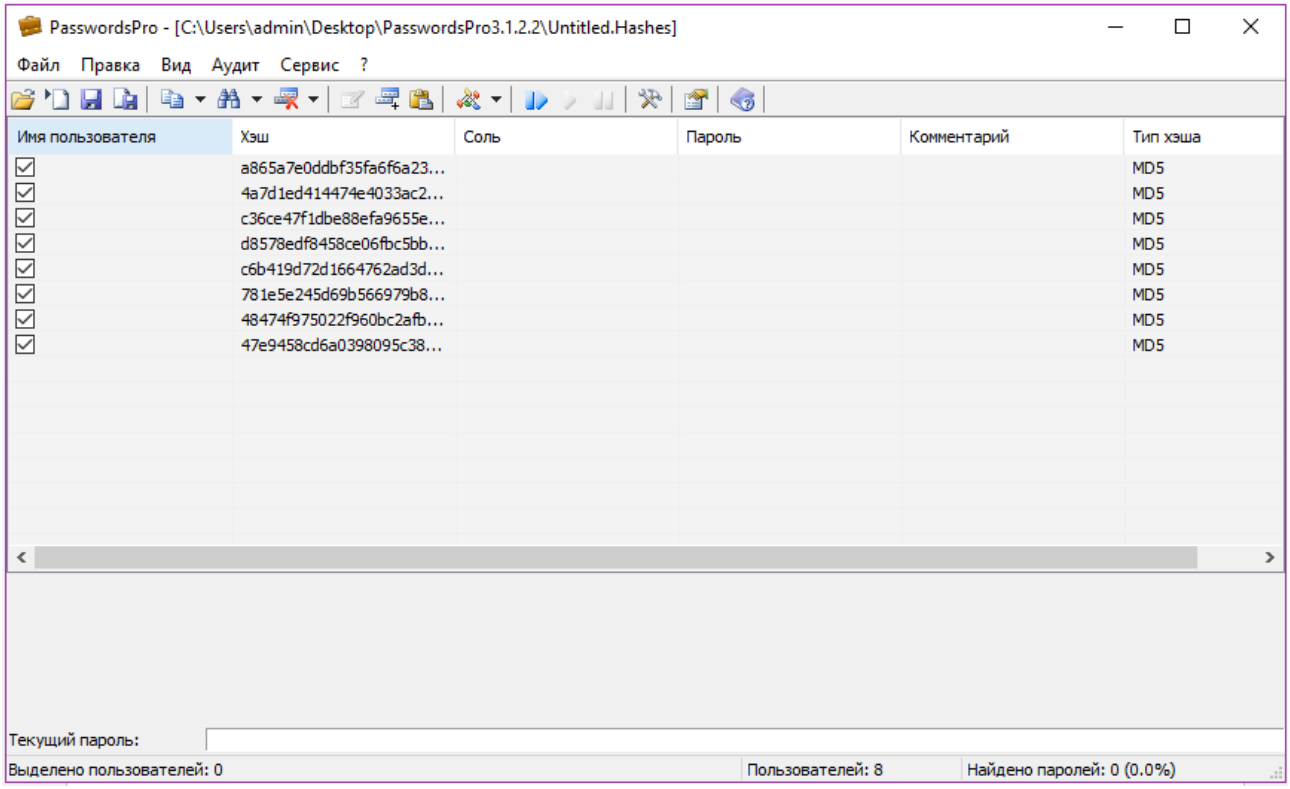


Рисунок 3.4 – Програма PasswordsPro із занесеними хешами

У досліді будуть використані основні типи атак:

1. Проста атака за словниками - в цій атаці відбувається проста перевірка хеш на паролі зі словників, зазвичай у словниках містяться найпростіші паролі типу "123", "qwerty", "99999" тощо, а також паролі, знайдені програмою раніше.

2. Brute Force Attack - це вичерпний пошук за всіма можливими паролями в певному діапазоні; наприклад, "аааааа" ... "zzzzzz", "0000" ... "9999" і т.д.

3. Rainbow Attack - ця атака намагається відновити паролі за допомогою попередньо розрахованих таблиць Rainbow.

Для першого виду атаки – Простої атаки за словниками існують готові словники з найпопулярнішими паролями а також їх комбінації. Основним недоліком такого типу атак є довгий час обробки, так як перебір всіх паролів дуже затрудняється через їх велику кількість та великий об’єм даних словників (Рис 3.5).

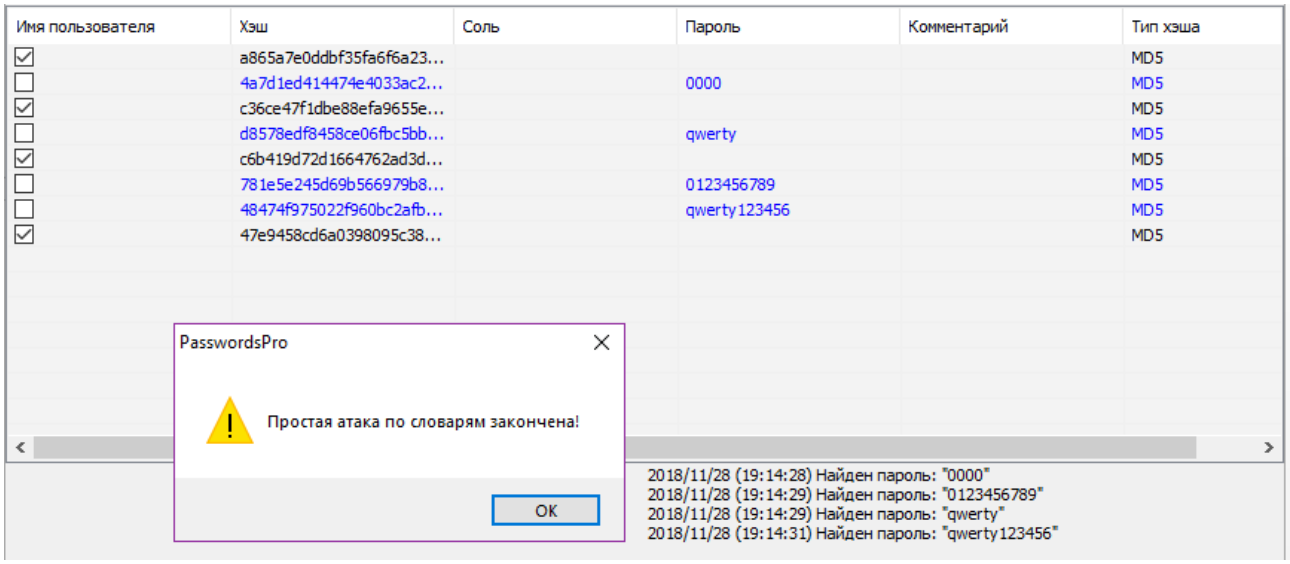


Рисунок 3.5 – Проведення звичайної атаки за словником

Із проведенням даної атаки можна побачити, що за хешами були знайдені первинні паролі, але можна помітити, що встановлені паролі були найбільш прості, тобто можна дійти висновку, що криптостійкість даних паролів дуже низька, так як шляхом звичайного співставлення хешів із найбільш поширених паролів дає позитивний результат.

Але при цьому можна дійти висновку, що при отримані хешу не з одного слова, такого як звичайний пароль, а для виразу, тексту чи файлу дану атаку використовувати не доцільно.

Brute-force є одним з найсильніших алгоритмом, тому що з його допомогою можливо зламати практично будь-які шифри, але недоліком такого типу атак є висока обчислювальна складність і дуже довгий процес перебору.

При цьому методи перебору по словнику і brute-force можуть використовуватися для злому хешу інших хеш-функцій (з невеликими змінами алгоритму).

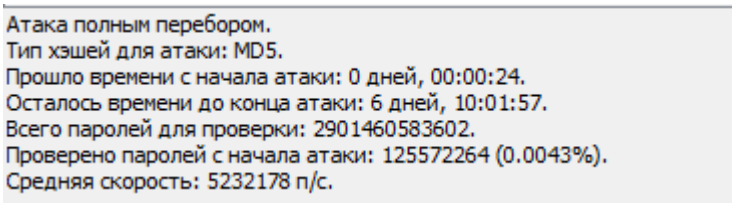


Рисунок 3.6 – Проведення атаки повним перебором

Атака грубою силою являється одним з найпопулярніших методів злому паролів. Тим не менш, вона також підходить для злому хешів при великому обсязі первинного тексту. Атаки методом перебору також можуть використовуватися для виявлення прихованих сторінок і контенту в веб-додатках. Ця атака, в основується на методі «удар і спроба», що виконується допоки не досягає успіху Ця атака іноді займає більше часу, але ймовірність успіху вище.

У традиційній атаці методом перебору реалізований пошук певної комбінації букв і цифр для послідовного відтворення необхідного «паролю». Однак цей традиційний метод займе більше часу, коли пароль досить довгий. Ці атаки можуть зайняти від декількох хвилин до декількох годин або декількох років в залежності від системи, що використовується та довжини пароля. (Рис 3.6).

Райдужна таблиця створюється побудовою ланцюжків можливих паролів. Кожен ланцюжок починається з випадкового можливого пароля, потім піддається дії хеш-функції і функції редукції. Ця функція перетворює результат хеш-функції в певний можливий пароль. Проміжні паролі в ланцюжку відкидаються і в таблицю записується тільки перший і останній елементи ланцюжків.

На відміну від попередніх, Rainbow attack вимагає попередньої підготовки райдужних таблиць, які створюються для заздалегідь певної хеш-функції.

Для даної функції було обрана програма Winrtgen v2.8 [52].

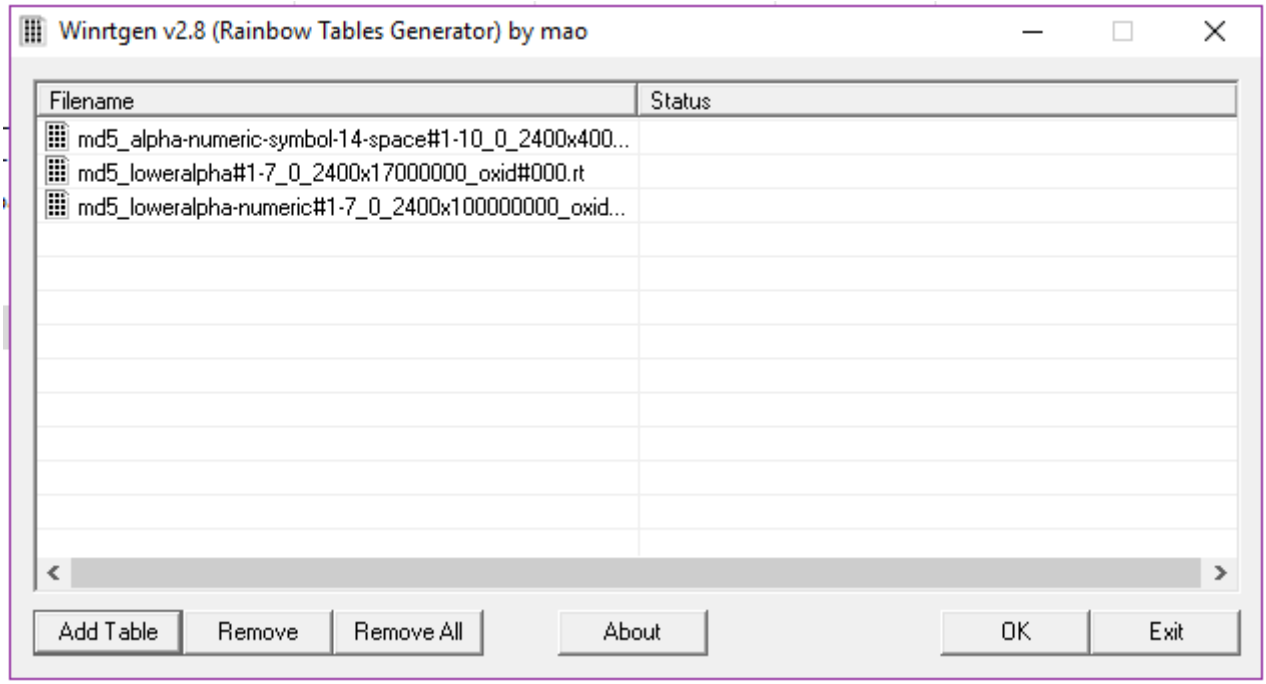


Рисунок 3.7 – Проведення атаки повним перебором

Winrtgen представляє графічний інтерфейс (Рис 3.7) для генератора Rainbow Table (формат RT). Він підтримує кілька підписів, в тому числі MD-4, MD-5, FastLM, SHA-256, ORACLE і CiscoPIX, що дозволяє користувачам створювати великі бази даних таблиць для дешифрування складних паролів.

Завдання може зайняти багато часу, в залежності від встановлених параметрів. Проте, Winrtgen використовує низьке завантаження ЦП і ОЗУ на відміну від інших подібних програм.

Створення таблиць вимагає часу і пам'яті (деякі більше сотень гігабайт), але вони дозволяють дуже швидко (у порівнянні зі звичайними методами) відновити вихідний пароль.

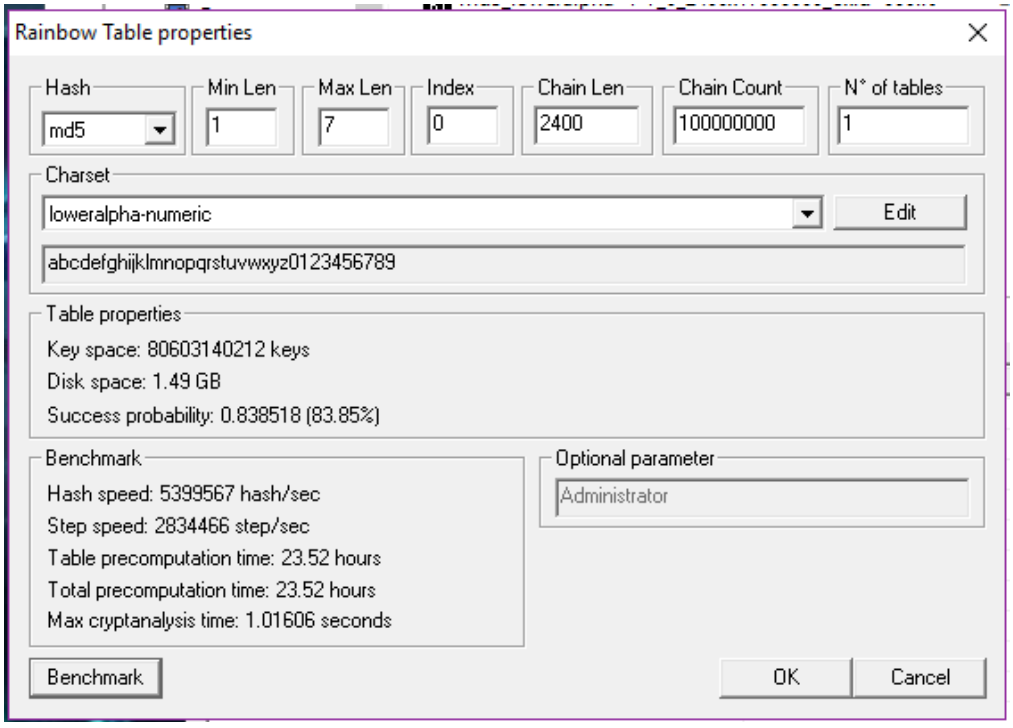


Рисунок 3.8 – Формування райдужної таблиці у програмі Winrtgen v2.8

Для первинного тесту для проведення атаки було обрано генерація таблиці із для злому MD5 алгоритму із параметрами – 7 символів, мінімальна кількість проходу алгоритму 2400 із набором символів цифр та літери нижнього регістру латиниці і як можна побачити при цих налаштуваннях генерування таблиці займе майже 24 години і важить 1,5 Гб і має більше 80 мільярдів комбінацій ключів (Рис 3.8).

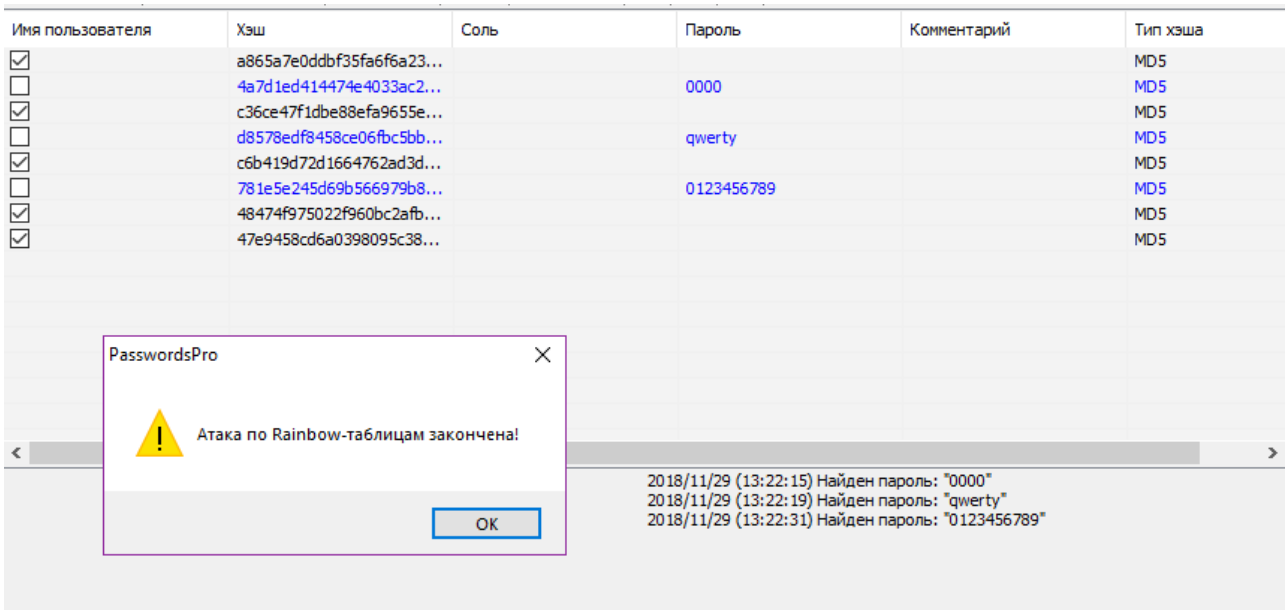


Рисунок 3.9 – Проведення атаки за допомогою райдужних таблиць

Є два великих недоліки при використанні райдужних таблиць. Головний недолік - для зберігання таблиць потрібно великий дисковий простір. Другий недолік - час, необхідний для генерації райдужних таблиці. Якщо таблиці не були заготовлені заздалегідь, вам будуть потрібні дні (і можливо місяці) для генерації таблиць з нуля.

Існує думка що зламати хеш MD5 неможливо, проте як можна бачити з результату, існує способи підбирають вихідне слово на основі хешу. Абсолютна більшість з них здійснює перебір по словнику, однак існують такі методи як Райдужні таблиці, він заснований на генеруванні безлічі хеш з набору символів, щоб по вийшла базі проводити пошук хеша.

Також у MD5, як у будь-якої хеш-функції, існує таке поняття як колізії - це отримання однакових хеш для різних вихідних рядків. І один із методів пошуку колізій показав свою ефективність, і був названий «тунелювання», що виник у 2006 році.

Тому для посилення можна використати такі надлаштувань для MD5:

- MD5 (HMAC) - (змішування з ключем для аутентифікації повідомлення) - алгоритм дозволяє хешувати вхідне повідомлення L з деяким ключем K, таке змішування дозволяє аутентифікувати підпис.
- MD5 (Base64) - тут отриманий MD5-хеш кодується алгоритмом Base64.
- MD5 (Unix) - алгоритм викликає тисячу разів стандартний MD5, для ускладнення процесу. Також відомий як MD5crypt.

Наступним кроком для перевірки криптоаналізу було обрано дослідження одного із найпопулярніших методів шифрування RSA у комплексі із методами хешування.

3.2 Криптоаналіз алгоритму RSA та хеш-функцій (цифровий підпис)

Даний метод шифрування буде базуватися на основі роботи такого інструменту, як HMAC.

HMAC (від англ. Hash-based message authentication code, код перевірки повідомлень, що використовує хеш-функції) - в криптографії, один з механізмів перевірки цілісності інформації, що дозволяє гарантувати, що даних не були відкориговані або деформовані сторонніми особами (наприклад як атака - людина посередині) [30].

Даний метод полягає у тому що у переданому повідомлені включений також хеш цього повідомлення. Тобто алгоритм приблизно такий [30]:

$$HMAC = E(m) \cap E\big(hash(m)\big) \text{ або } E\big(m \cap hash(m)\big) \quad (3.1)$$

, де $E(m)$ – шифрування повідомлення,

$hash(m)$ – хеш повідомлення.

Одним з найбільш поширених методів несиметричного шифрування/дешифрування є метод шифрування з відкритим ключем на основі алгоритму RSA, що заснований на використанні операції піднесення до степеню модульної арифметики. Алгоритм RSA можна представити у вигляді наступної послідовності для отримання ключів шифрування [8]:

$$n = p * q \tag{3.2}$$

, де p і q - два великих простих числа;

n - відкрита компонента ключа

На практиці для забезпечення криптостійкості системи величина цих чисел повинна бути довжиною не менше двохсот десяткових розрядів [8].

$$f(p,q) = (p-1)(q-1) \tag{3.3}$$

, де $f(p,q)$ - функція Ейлера.

$$e * d(|f(p,q)|) = 1 \tag{3.4}$$

де e - число, яке має бути взаємно простим із значенням функції Ейлера і меншим, ніж $f\ (p,q)$.

d – число, яке задовольняє співвідношенню.

Числа e і n приймаються в якості відкритого ключа. В якості секретного ключа використовуються числа d і n.

Підберемо три варіанти для значень p та q, та розрахуємо параметри закритого і відкритого ключа.

Для вибору можливе використання таких відкритих сервісів, як RSA Key Generator [25], що може відбирати p і q, чи навіть запропонувати пари ключів.

Для подальшого опрацювання були обрані такі пари простих чисел(табл. 3.2)

Таблиця 3.2 - Пари обраних простих чисел p та q

p	q
251	31
1523	113
165173	1231

Далі розрахуємо параметри відкритих та закритих ключів

Дана операція можливо частково реалізувати за допомогою калькуляторів ключів для RSA, такими як RSA Calculator [26].

На першому етапі вводяться первинні дані, а саме обрані p та q (Рис 3.10).

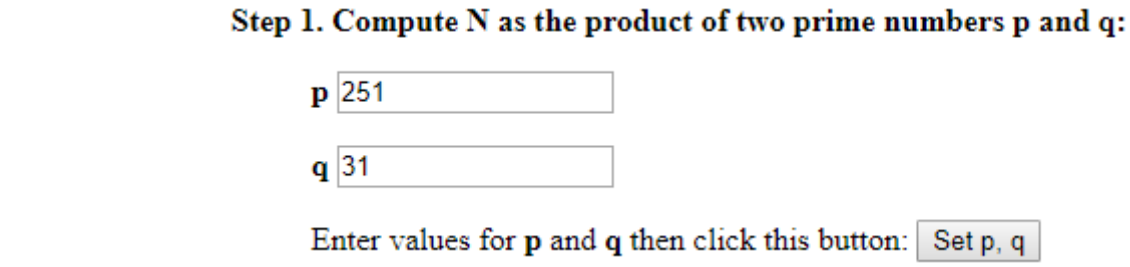


Рисунок 3.10 – Перший етап – занесення даних

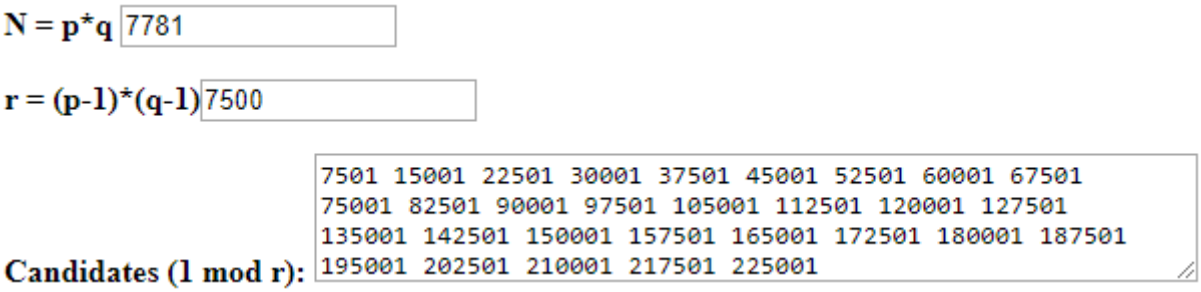


Рисунок 3.10 – Другий етап – обрахування даних

На другому етапі обраховуються другорядні дані такі, як N – відкрита компонента ключа, та r – функція Ейлера. Та видаються варіанти для обрання числа, що надалі буде виступати в обрахунку для ключів.

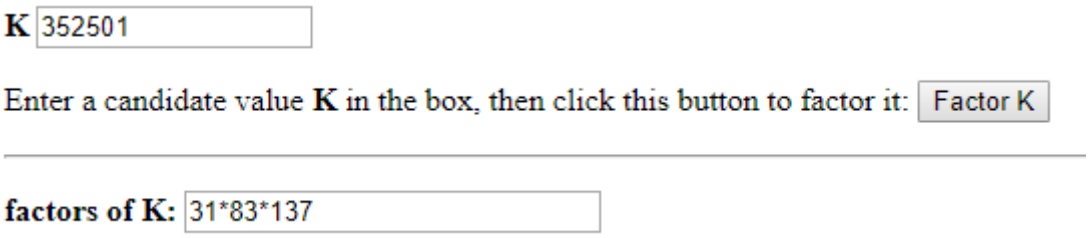


Рисунок 3.11 – Третій етап – розкладання обраного числа

На третьому етапі обирається число та розкладеться на множники

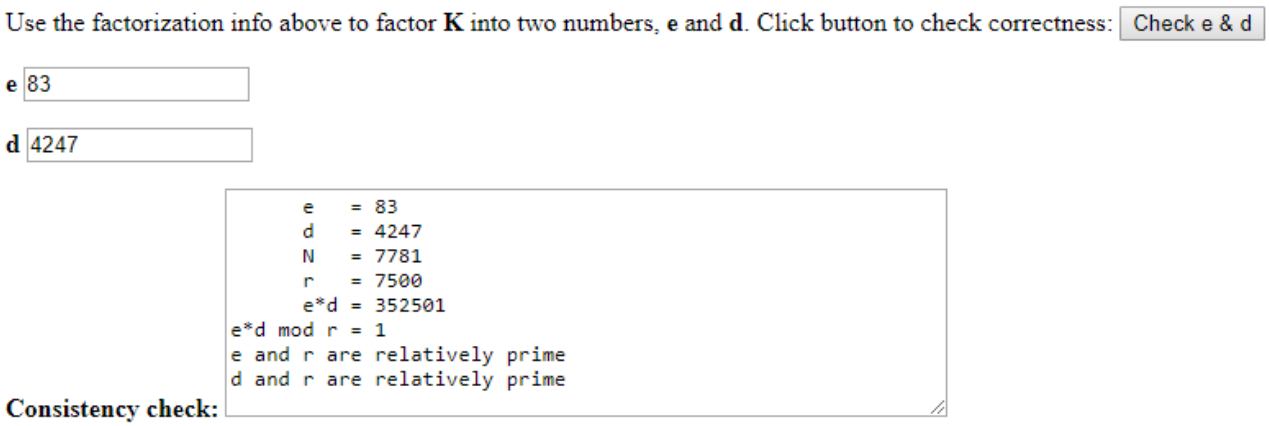


Рисунок 3.11 – Останні етап – формування ключів для шифрування

На четвертому етапі іде формування відкритого та закритого ключа шифрування та перевіряються ключі на їх простоту, тобто що вони діляться тільки на одинцю і себе.

Якщо результат задовільний то обчислені дані можливо використовувати для подальшого шифрування, а саме (e, n) – для пари відкритого ключа шифрування і (d, n) – для закритого.

Було розраховано для значень p та q , такі параметри закритого і відкритого ключа (табл. 3.3).

Таблиця 3.3 - Обчислені параметри ключів шифрування для p та q

Параметри p, q	Пара відкритого ключа e, n	Пара закритого ключа d, n
251, 31	83,7781	4247, 7781
1523, 113	251, 172099	29203, 172099
165173, 1231	647, 203327963	67511183, 203327963

Для перевірки працездатності алгоритму була розроблена програма (Додаток Б, Рисунок 3.12-3.14).

Дана програма була розроблена за допомогою мови програмування C# і за допомогою стандартних бібліотек.

Програма реалізує процес поступового шифрування та дешифрування RSA із розрахунком закритого та відкритого ключів з початкових даних – простих чисел p та q.

Процес шифрування починається із обрання простих значень p та q , і занесення даних (тексту) для шифрування у текстовий файл «in.txt» (Рис 3.12)..

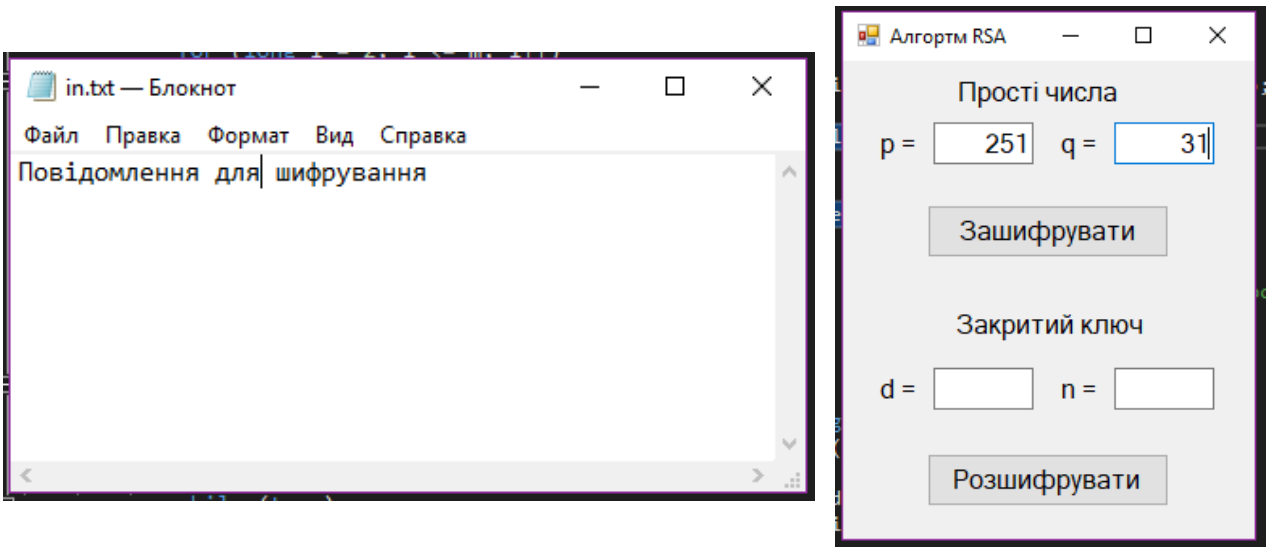


Рисунок 3.12 – Занесення даних до текстового файлу та форми

Після натискання кнопки «Зашифрувати» обраховуються закриті ключ для подальшого дешифрування та формується шифр у файлі «out1.txt» (Рис 3.13).

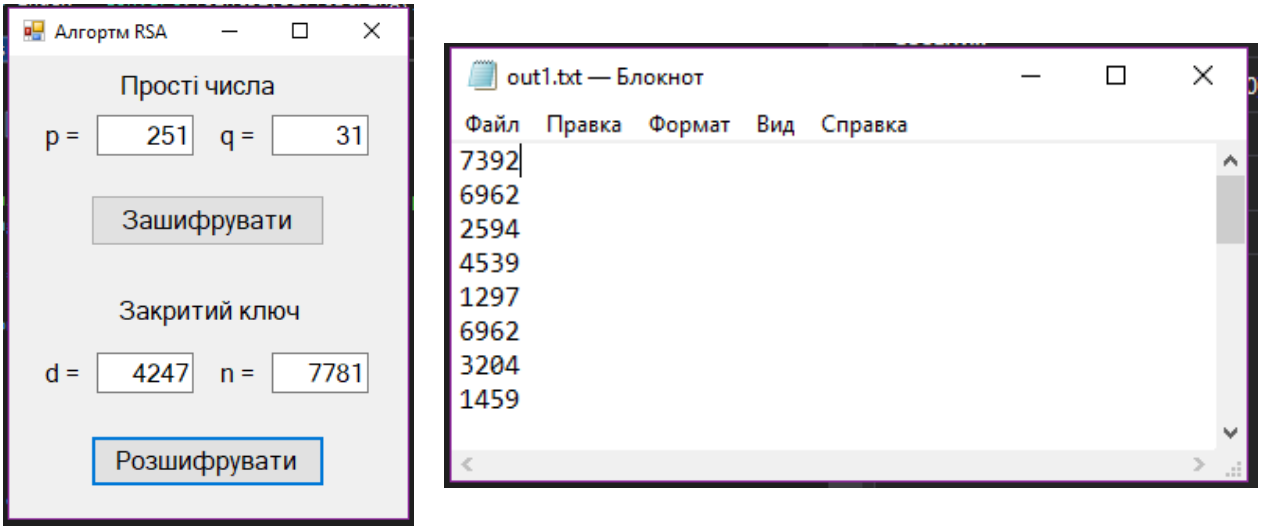


Рисунок 3.13 – Сформовані закриті ключі та файл із шифром

Для розшифрування використовується лише файл «out1.txt» із шифрованим повідомленням і пара закритих ключів

Після натискання кнопки «Розшифрувати» за введеними ключами реалізується процес дешифрування і отриманий текст заноситься до файлу «out2.txt» (Рис 3.14).

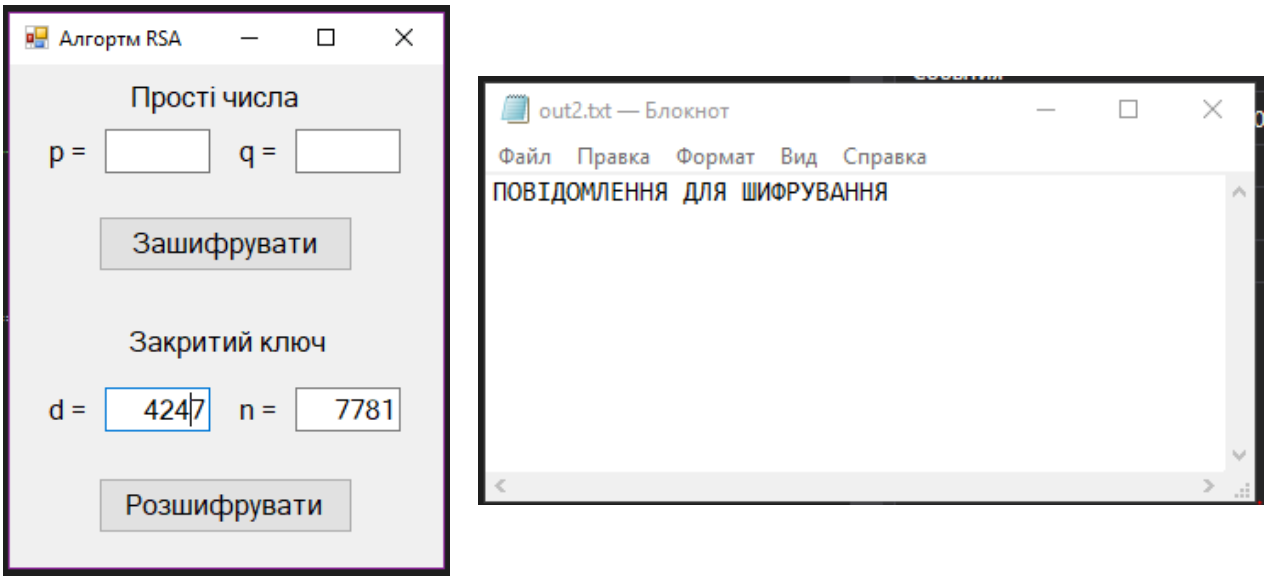


Рисунок 3.14 – Дані отримані по результату дешифрування

Наступним етапом буде проведення криптоаналізу алгоритму RSA за допомогою допоміжних програмних засобів, таких як Cryptool 2 [27], що працює із алгоритмами шифраторів, та допомагає змоделювати процес шифрування та отримувати необхідну статистику.

Cryptool 2 підтримує усі відомі на сьогодні моделі шифрування. Функціональні блоки мають модулі для введення і виведення інформації в процесі роботи, які в свою чергу можуть приєднуватися до інших функціональних блоків і обмінюватися інформацію між собою. Кожен блок має сценарій роботи і віртуалізації. Це дає можливість після складання всієї схеми запустити моделювання роботи. Після запуску моделювання кожен з блоків починає поступове завантаження з відображенням процесу у вигляді процентного виконання. По закінченню циклу на кожному блоці а також у вікні процесу відображається повний хід дій, в якому можуть бути присутні помилки та не стикування блоків шифрування та повідомлення у разі помилки при передачі інформації.

По-перше змоделюємо роботу алгоритма RSA (Рис. 3.15).

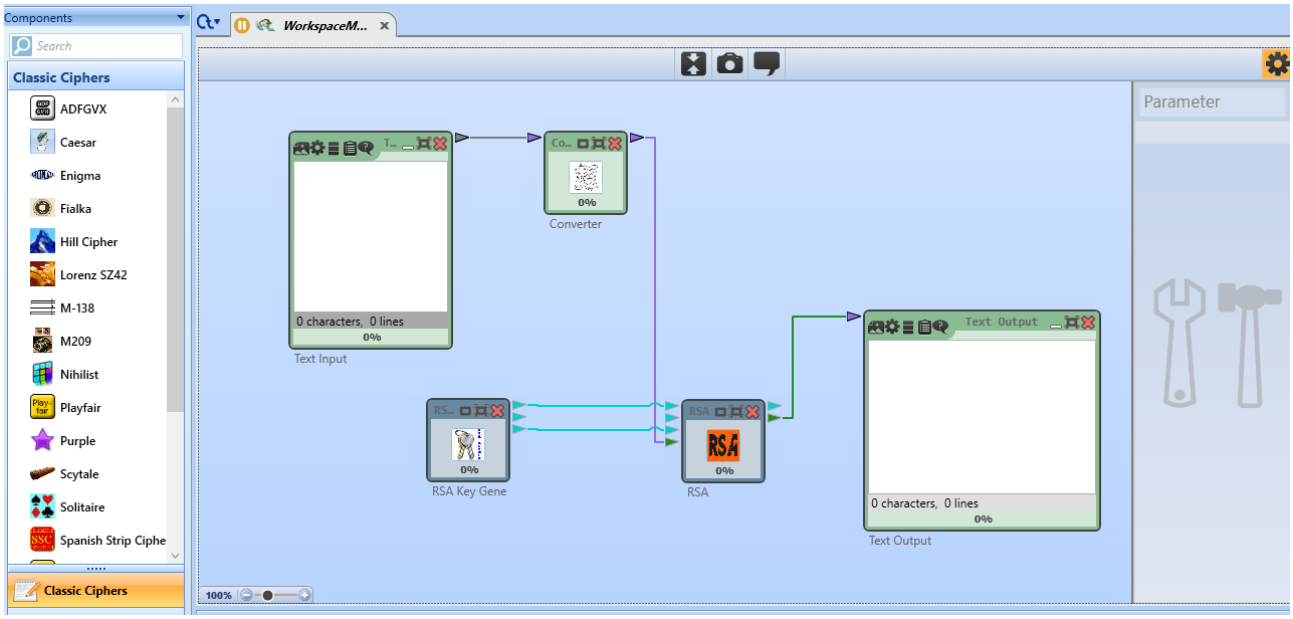


Рисунок 3.15 – Робоча область програми Cryptool 2 із моделлю роботи RSA алгоритму

Для реалізації алгоритму RSA було сформовано схему її роботи із областю вводу даних (відкритий текст), конвертор у строки, компонент роботи алгоритму RSA, генератор ключів для RSA, вихідний текст.

Як еталонний варіант проведемо шифрування із чистим алгоритмом RSA.

Для генератора ключів можливе введення своїх параметрів для закритого та відкритого ключа.

Занесемо до генератора ключів дані, що були отримані у результаті розрахунків(Рис. 3.16).

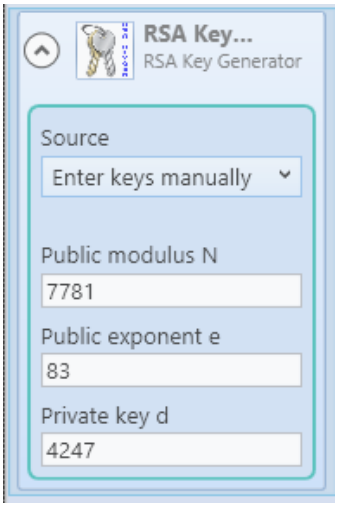


Рисунок 3.16 – Параметри ключів шифрування у RSA генераторі ключів

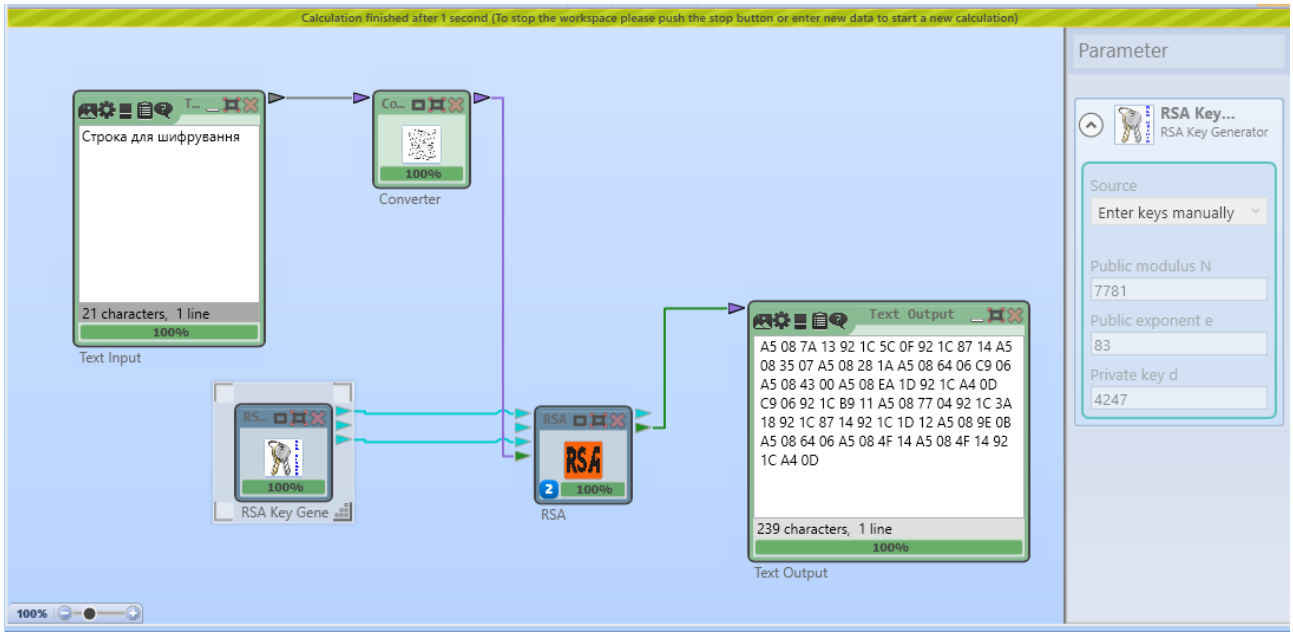


Рисунок 3.17 – Проведений процес шифрування за алгоритмом RSA

У результаті були отримані дані, а саме об’єм символів із початкових 21 було отримано 239 вихідних символів при шифруванні алгоритмом RSA.

Дані, що були отримані, будуть виступати як точка для порівняння для наступних експериментів.

Наступним кроком буде посилення даної системи шифрування шляхом введення у систему хеш-елементів різних алгоритмів.

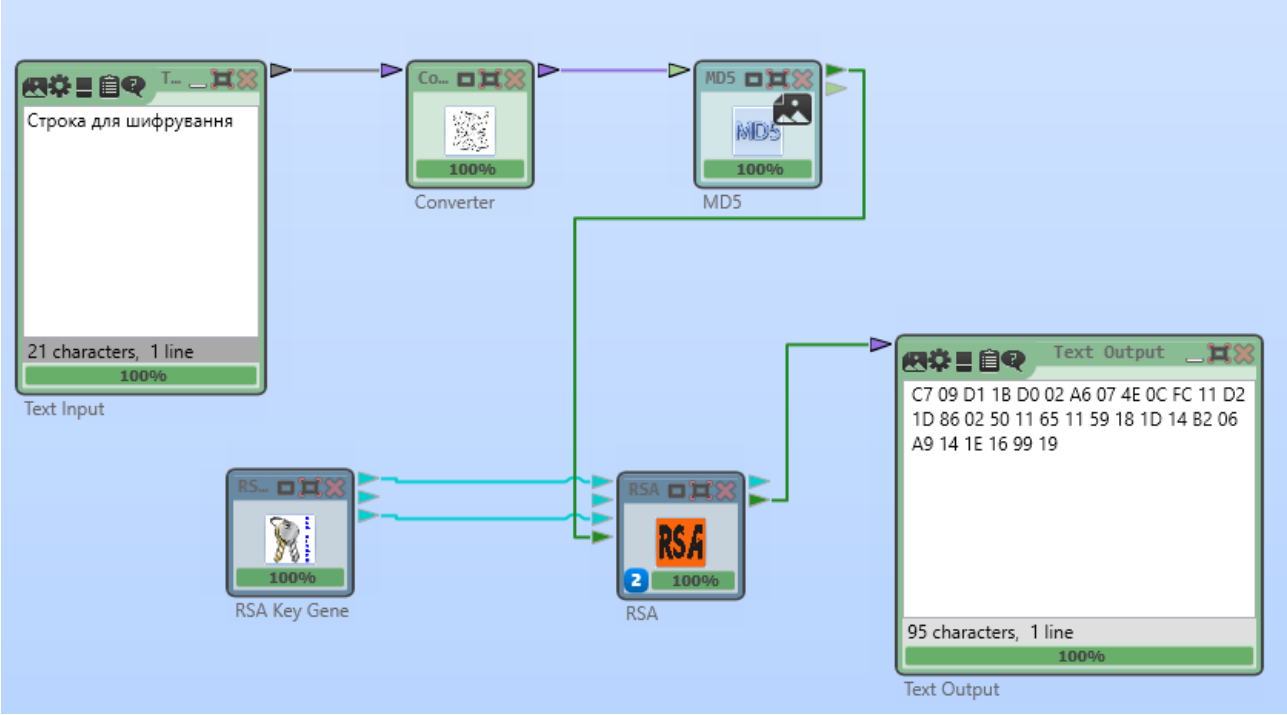


Рисунок 3.18 – Процес шифрування за алгоритмом RSA та хеш-функцією MD5

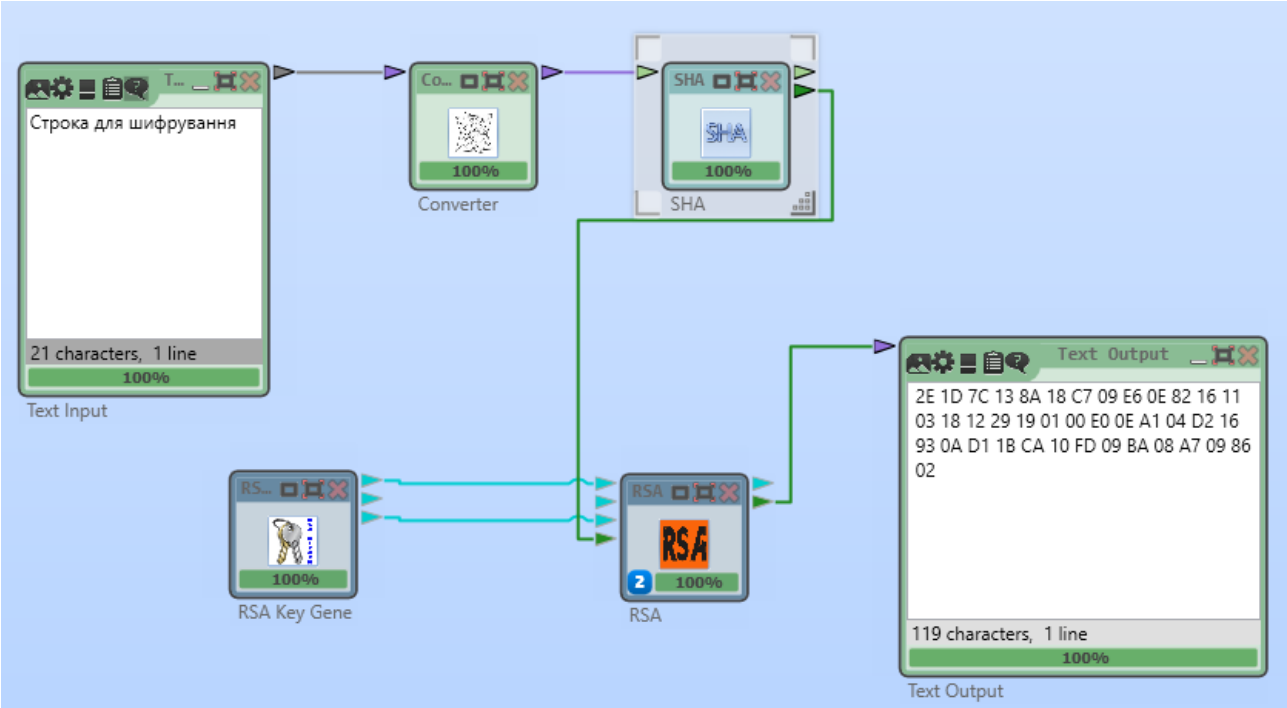


Рисунок 3.19 – Процес шифрування за алгоритмом RSA та хеш-функцією SHA-1

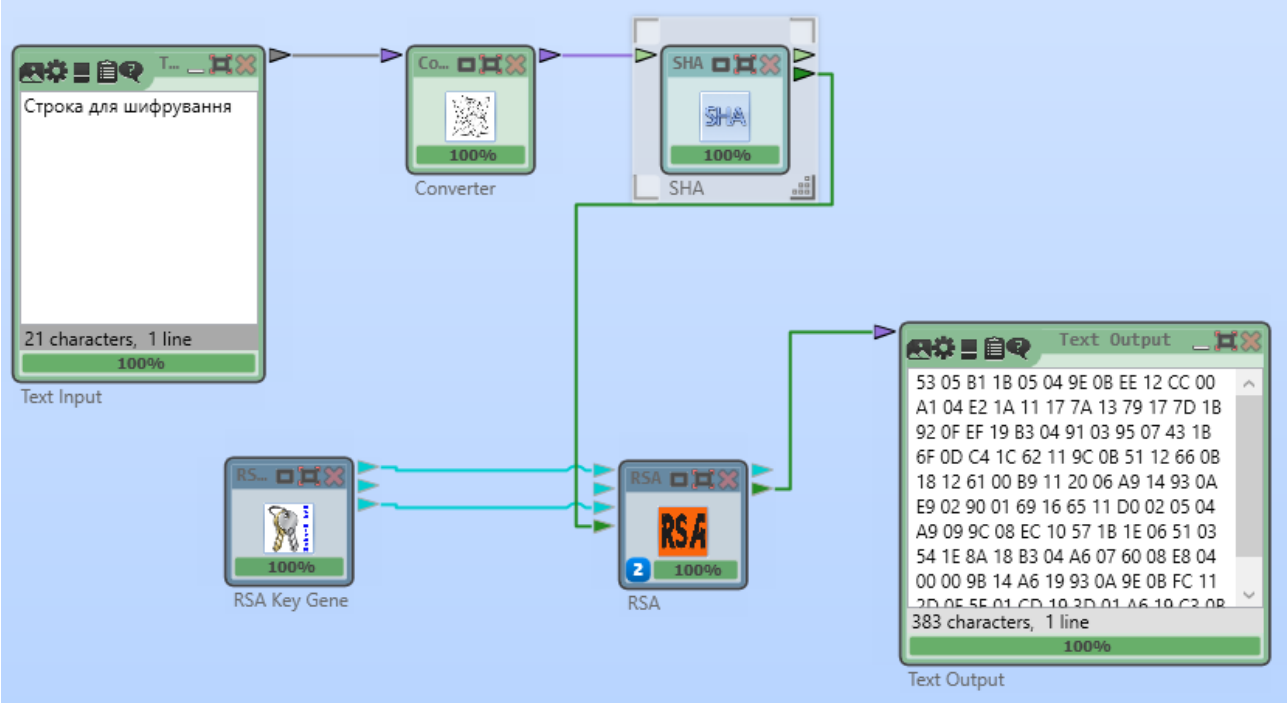


Рисунок 3.20 – Процес шифрування за алгоритмом RSA та хеш-функцією SHA-512

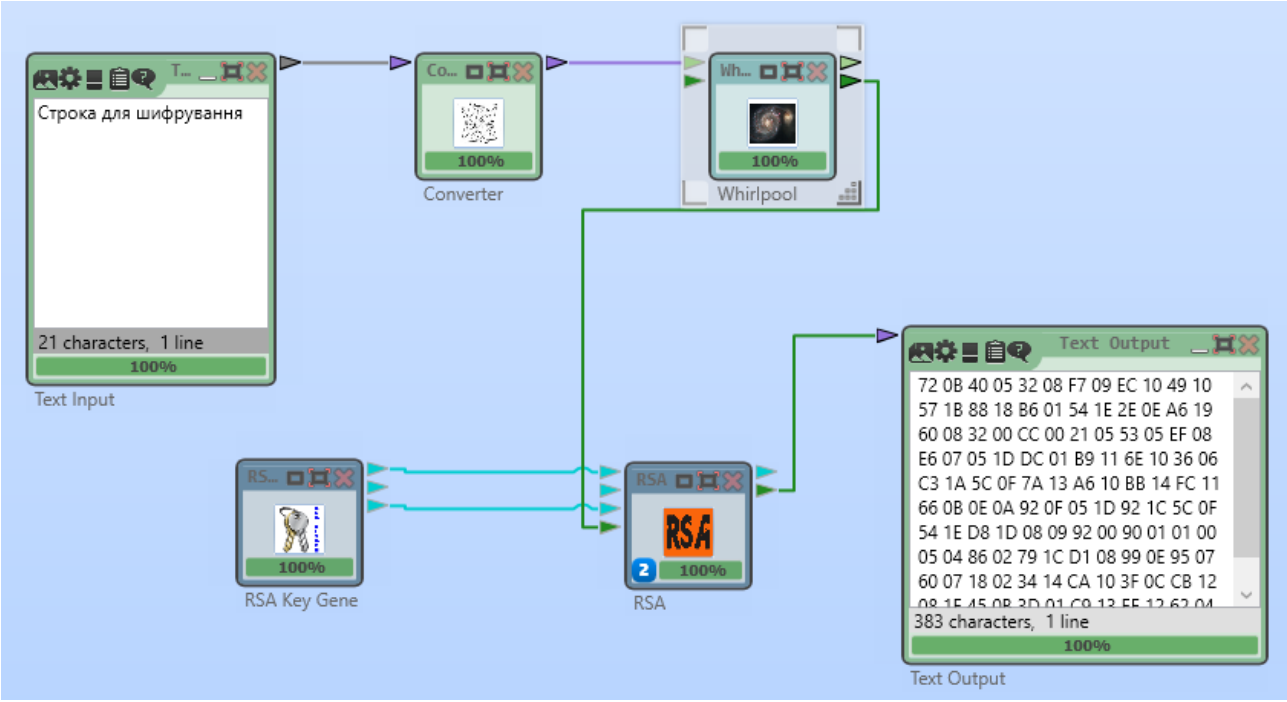


Рисунок 3.21 – Процес шифрування за алгоритмом RSA та хеш-функцією Whirlpool

Були отримані дані для ключів шифрування {83,7781} {4247, 7781}.

Наступним кроком буде моделювання для інших наборів ключів і для них були отримані наступні дані (табл. 3.3):

Таблиця 3.3 - Отримані дані для алгоритмів шифрування

Алгоритм	Кількість символів		
	1й набір ключів	2й набір ключів	3й набір ключів
RSA	239	179	167
RSA+MD5	95	68	71
RSA+SHA-1	119	89	80
RSA+SHA-512	383	287	263
RSA+Whirlpool	383	284	260

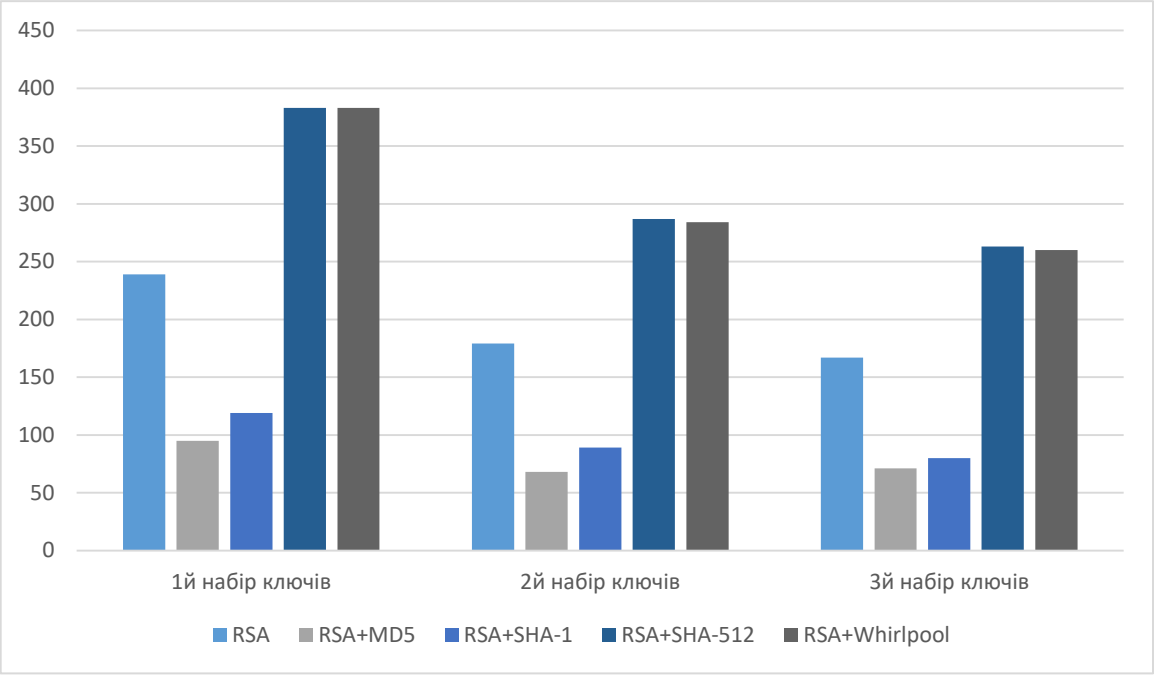


Рисунок 3.22 – Діаграма залежності кількості символів від ключів за різних методів

З Рисунок 3.22 видно, що при збільшенні розміру ключа найбільших змін отримує алгоритму RSA з використанням хеш-функцій SHA-512 та Whirlpool. Порівняльні дані показують ефективність використання алгоритмів шифрування у цілому. Як висновок можна зазначити, що при зміні ключів шифрування на більший за кількістю символів можна спостерігати незначне зменшення об’єму вихідних даних, що зменшує криптостійкість, але беручи до уваги принцип Керкгоффза про ключі шифрування стійкість до злому шифру не знаючи при цьому зростає прямо-пропорційно зростанню довжини ключа.

При використанні алгоритму RSA отриманий шифротекст є в 10 разів більший на відміну від початкового тексту, але його стійкість є занизькою. Хоча алгоритми RSA із хеш-функціями SHA-512 та Whirlpool доводять свою надійність, але через великий об’єм вихідних даних ці методи не можуть бути прийнятні у системах із обмеженою пам’яттю.

3.3 Реалізація атаки на алгоритм шифрування RSA за методом Ферма

Ідея алгоритму полягає в пошуку таких чисел a і b, що число, що факторизується n представимо у вигляді: $n = a^2 - b^2 = (a - b)(a + b)$. Цей метод не ефективний для факторизації великих чисел, так як має експонентну складність. Метод примітний тим, що реалізується без операції ділення, а тільки лише з операціями додавання і віднімання. Слід так само відзначити, що якщо $n = p\,q$, за умови того, що p і q - прості числа не сильно відрізняються за величиною, то метод Ферма факторизує n досить швидко [11].

Для проведення даної атаки припустимо, що в атакуючого мається число n, и тоді знаходиться натуральний дільник a:

Крок 1: Обчислити найменше ціле число s таке, що $s^2 \geq \sqrt{n}$, тобто $s = \lfloor\sqrt{n}\rfloor$.

Крок 2: Якщо $s^2 = n$, то $a = s$ та завершується алгоритм.

Крок 3: Взяти $x = s, l = x^2 - n$ і лічильник кроків $k = 0$.

Крок 4: Якщо l є повним квадратом, то обчислити $y = \sqrt{l}, a = x + y$ і закінчити алгоритм.

Крок 5: Обчислити $k = k + 1, x = x + 1, l = x^2 - n$. Перейти до пункту 4.

Інший дільник числа n дорівнює $b = \frac{n}{a}$.

Покажемо, що кількість кроків алгоритму не перевищує величини $y = \frac{a-b}{2}$

Маємо $x = s + k$. Тоді $k = x - s$. З огляду на, що $a = x + y$, отримаємо $k = x - s = a - y - s$. Так як справедлива нерівність $a > s > b$, маємо:

$$k = a - y - s < a - b - y = a - b - \frac{a-b}{2} = \frac{a-b}{2} \tag{3.5}$$

Тоді

$$k < \frac{a-b}{2} \tag{3.6}$$

Тепер, коли усі кроки розписані можливо зробити розрахунок на прикладі першого набору ключів - $p, q = 251{,}31$; $e, d, n = 83, 4247, 7781$

За початкові дані береться число $n = 7781$ і за допомогою методу факторизації Ферма розложмо на множники. Знаходимо найменше ціле число s таке, що $s^2 \geq \sqrt{7781}$, тобто $s = \sqrt{7781} \approx 89$. Далі задаючи $k = 0{,}1, \ldots$ обчислюємо $l = (s + k)^2 - n$. Якщо на якомусь етапі l є квадратом натурального числа, то процедуру зупиняємо.

Таблиця 3.4 - Реалізація методу факторизації Ферма

k	$k = s + k$	$l = x^2 - n$	$y = \sqrt{l}$	$a = x + y$	$b = x - y$
0	89	140	11.832	100.832	77.168
1	90	319	17.860	107.860	72.140
2	91	500	22.360	113.360	68.640
3	92	683	26.134	118.134	65.866
4	93	868	29.461	122.461	63.539
5	94	1055	32.480	126.480	61.520
6	95	1244	35.270	130.270	59.730
7	96	1435	37.881	133.881	58.119
8	97	1628	40.348	137.348	56.652
9	98	1823	42.696	140.696	55.304
10	99	2020	44.944	143.944	54.056
11	100	2219	47.106	147.106	52.894
12	101	2420	49.193	150.193	51.807
13	102	2623	51.215	153.215	50.785
14	103	2828	53.178	156.178	49.822
15	104	3035	55.090	159.090	48.910
16	105	3244	56.956	161.956	48.044
17	106	3455	58.779	164.779	47.221
18	107	3668	60.564	167.564	46.436
19	108	3883	62.313	170.313	45.687
20	109	4100	64.031	173.031	44.969
21	110	4319	65.719	175.719	44.281
22	111	4540	67.379	178.379	43.621
23	112	4763	69.014	181.014	42.986
24	113	4988	70.625	183.625	42.375
25	114	5215	72.214	186.214	41.786
26	115	5444	73.783	188.783	41.217
27	116	5675	75.332	191.332	40.668
28	117	5908	76.863	193.863	40.137
29	118	6143	78.377	196.377	39.623
30	119	6380	79.874	198.874	39.126
31	120	6619	81.357	201.357	38.643
32	121	6860	82.825	203.825	38.175
33	122	7103	84.279	206.279	37.721
34	123	7348	85.720	208.720	37.280
35	124	7595	87.149	211.149	36.851
36	125	7844	88.566	213.566	36.434
37	126	8095	89.972	215.972	36.028
38	127	8348	91.367	218.367	35.633
39	128	8603	92.752	220.752	35.248
40	129	8860	94.127	223.127	34.873
41	130	9119	95.493	225.493	34.507
42	131	9380	96.850	227.850	34.150
43	132	9643	98.198	230.198	33.802

44	133	9908	99.538	232.538	33.462
45	134	10175	100.871	234.871	33.129
46	135	10444	102.195	237.195	32.805
47	136	10715	103.513	239.513	32.487
48	137	10988	104.823	241.823	32.177
49	138	11263	106.127	244.127	31.873
50	139	11540	107.424	246.424	31.576
51	140	11819	108.715	248.715	31.285
52	141	12100	110.000	251.000	31.000

Тобто отримуємо за 53 кроки розложений $n = 7781 = 251 * 31$, а це значить, що за допомогою значення n було знайдене початкове значення p, q, що дорівнюють 251,31 відповідно.

А надалі щоб підібрати відповідні ключі шифрування вже не виникає складнощів.

Якщо не зважати на наявність даних у атакуючого компонентів для шифрування ,тоді криптостійкість методів шифрування є прийнятною і швидкість злому шифру залежить лише від самого методу шифрування та обладнання, з якого ведеться «злом» шифру.

3.4 Висновок до розділу 3

У третьому розділу проведено криптоаналіз обраного методу шифрування та допоміжних засобів для підвищення криптографічної стійкості.

Був проведений аналіз алгоритму шифрування RSA і за допомогою принципу Керкгоффза було висунуто теорію, щодо криптостійкості методу від опрацювання та величини ключів шифрування. Практично було реалізовано та проведено атаку за методом факторизації Ферма, що при наявності у атакуючого хоча б одного елементу алгоритма для шифрування, то можливе встановлення відповідності даних та знаходження первинних даних, такі як ключі шифрування, і з подальшим знаходженням шифротексту .

Проведення аналізу роботи хеш-функцій та електронного цифрового підпису показало свою надійність, але при використанні атак методами перебору за словниками та райдужних таблиць дало результат 50% вдалих атак, що призвело до злому і знаходження первинного тексту.

Проведено огляд та оцінка методів шифрування і розглянуті методи посилення стійкості методів шляхом поєднання із такими інструментами як хеш-функції. Проведені атаки методами факторизації та райдужних таблиць показали вразливість методів шифрування. За результатами досліджень були висунуті основні критерії для проведення оцінки криптографічної стійкості.

Для збільшення криптографічної стійкості та запобіганню зміни даних (атака «Людина посередині» та ін.), як приклад, було запропоновано використання поєднання алгоритму RSA та хеш-функції MD5. Для запобігання злому для симетричних та асиметричних методів шифрування рекомендовано посилення ключів шифрування за допомогою збільшення розміру ключів або їх додаткова обробка за допомогою методів, таких як ітерація ключа, якщо це можливо.

РОЗДІЛ 4
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ. ЕКОЛОГІЯ

У даному розділі проведено аналіз потенційних небезпечних і шкідливих виробничих факторів, причин пожеж. Розглянуто заходи, які дозволяють забезпечити гігієну праці та виробничу санітарію. На підставі аналізу розроблено заходи з техніки безпеки і рекомендації з пожежної профілактики.

Завданням даної роботи магістра було розглянути моделі та програмні засоби розробки клієнтської частини комп’ютерної системи моніторингу якості освіти. Так як в процесі проектування використовувався персональний комп'ютер (ПК), то аналіз потенційно небезпечних і шкідливих виробничих факторів виконується для робочого місця з ПК, на якому буде використовуватися розроблений засіб

4.1 Загальні питання з охорони праці

Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці. У законі України «Про охорону праці» визначається, що охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів і засобів, спрямованих на збереження життя, здоров'я і працездатності людини в процесі трудової діяльності .

4.1.1 Організаційно-технічні заходи з охорони праці

В організації проводиться навчання і перевірка знань з питань охорони праці відповідно до вимог «Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці» [31]. Також впроваджені організаційні заходи з пожежної безпеки - навчання і перевірку знань відповідно до вимог «НПАОП 0.00-4.12- 05 Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці» [32].

Обов'язковими вимогами враховано наступне:

- не слід допускати до роботи осіб, в установленому порядку не пройшли навчання, інструктаж і перевірку знань з охорони праці, пожежної безпеки та цих Правил.

- на підприємстві / організації, де експлуатуються ЕОМ з відео дисплейними терміналами (ВДТ) і периферійними пристроями (ПП), розробляється інструкція з охорони праці відповідно до «НПАОП 0.00-4.15-98 Про розробку інструкцій з охорони праці» [33].

4.2 Аналіз стану умов праці

Умови праці повинні задовольняти таким вимогам, які дали б можливість людині виконувати роботу без шкоди для здоров'я, без перевтоми і з високою продуктивністю. Для вибору показників умов праці при проведенні аналізу слід керуватися чинними в Україні «НПАОП 0.00-1.31-99 Правилами охорони праці під час експлуатації електронно-обчислювальних машин» [34].

Робота з інформаційною комп’ютерною системою буде проходити в приміщенні із комп’ютером, або сервером. Для даної роботи досить однієї людини, для якого надано робоче місце зі стаціонарним комп'ютером. Виконувана робота за ступенем тяжкості відноситься до категорії "легка 1б". До неї відносяться роботи, вироблені сидячи, стоячи або пов'язані з ходьбою, але не потребують систематичного фізичного напруження чи підняття і перенесення важких предметів.

4.2.1 Вимоги до приміщень

Згідно з «Санітарні норми мікроклімату виробничих приміщень» [35] розмір площі для одного робочого місця оператора персонального комп'ютера повинно бути не менше 6 кв. м, а обсяг - не менше 20 куб. м. Дане робоче місце має обсяг біля 60 м^3, площа -24 м^2, отже, дане приміщення повністю відповідає зазначеним нормам.

Для дотримання певного рівня мікроклімату в будівлі встановлено систему опалення та кондиціонування.

Для забезпечення потрібного рівня освітленості кімната має вікна і систему загального рівномірного освітлення, встановлено на стелі. Для дотримання вимог пожежної безпеки встановлено порошковий вогнегасник і систему автоматичної пожежної сигналізації.

4.2.2 Вимоги до організації місця праці

Робочий стіл на досліджуваному місці також містить досить простору для ніг. Крісло, використовується в якості робочого сидіння, є підємно-поворотним, має підлокітники і можливість регулювання по висоті і куту нахилу спинки, також воно м'яке і виконано з екологічної шкіри, що дозволяє працювати в комфорті. Екран монітора знаходиться на відстані 0.8 м, клавіатура має можливість регулювання кута нахилу 5-15°С. Отже, за всіма

параметрами робоче місце відповідає нормативним вимогам. Приміщення кабінету повинно знаходиться у будинку і має обсяг біля 60 $м^3$, площа -24 $м^2$. У цьому кабінеті обладнано три місця праці , з яких два укомплектовані ПК.

Температура в приміщенні протягом року коливається в межах 18-24°С, відносна вологість - близько 50%. Швидкість руху повітря не перевищує 0,2 м/с. Шум в лабораторії знаходиться на рівні 50 дБА. Система вентилювання приміщення - природна неорганізована, а опалення - централізоване.

Розміщення вікон забезпечує природне освітлення з коефіцієнтом природного освітлення не менше 1,5%, а загальне штучне освітлення, яке здійснюється за допомогою восьми люмінесцентних ламп, забезпечує рівень освітленості не менше 200 Лк.

У кабінеті є електрична мережа з напругою 220 В, яка створює небезпеку ураження електричним струмом. ПК і периферійні пристрої можуть бути джерелами електромагнітних випромінювань, аерозолів і шкідливих речовин (часток тонера, оксидів азоту і озону).

За ступенем пожежної безпеки приміщення належить до категорії В. Кабінет оснащений переносним вуглекислотним вогнегасником ВВК-5. Є аптечка для надання долікарської допомоги, а також в кабінеті роблять вологе прибирання і щодня провітрюють приміщення.

4.3 Виробнича санітарія

На підставі аналізу небезпечних і шкідливих факторів при експлуатації, пожежної безпеки можуть бути в подальшому вирішені питання необхідності забезпечення працюючих достатньою кількістю освітлення, вентиляції повітря, організації заземлення тощо.

4.3.1 Аналіз небезпечних і шкідливих факторів при експлуатації системи

Аналіз небезпечних і шкідливих виробничих факторів виконується в табличній формі (табл. 4.1). Роботу, пов'язану з ЕОП з ВДТ, в тому числі тих, які мають робочі місця, обладнані ЕОМ з ВДТ і ПП, виконують із забезпеченням виконання «НПАОП 0.00-1.31-99 Правилами охорони праці під час експлуатації електронно-обчислювальних машин»[34], які встановлюють вимоги безпеки до обладнання робочих місць, до роботи із застосуванням ЕОМ з ВДТ і ПП. Переважно роботи по проектам виконують в кабінетах чи інших приміщеннях, де використовують різноманітне електрообладнання, в тому числі персональні комп'ютери (ПК) і периферійні пристрої. Основними робочими характеристиками персонального комп'ютера є:

- робоча напруга U = + 220 + -5%;
- робочий струм I = 2А;
- споживана потужність P = 350 Вт.

Таблиця 4.1 - Аналіз небезпечних і шкідливих виробничих факторів

Небезпечні і шкідливі виробничі фактори	Джерела факторів (види робіт)	Кількісна оцінка	Нормативні документи
1	2	3	4
фізичні			
- підвищена температура поверхонь обладнання	експлуатація ЕОМ, принтерів, сканерів чи/або серверного обладнання для роботи	2	ДСН 3.3.6.042-99[35]
- підвищений рівень шуму на робочому місці	-//-	2	ДСН 3.3.6.037-99 [43]
- підвищений рівень електромагнітного випромінення	-//-	2	ГОСТ 12.1.006-84 [45]
- підвищений рівень напруги електричної мережі, замикання якої може відбутися через тіло людини	-//-	4	ГОСТ 12.1.030-81 [46] ГОСТ 13109-97 [47]
- підвищена напруженість електричного поля	-//-	2	ГОСТ 12.1.006-84 [45]
- недостатність природного світла	порушення умов праці (вимог до приміщень)	2	ДБН В.2.5-28:2015 [38]
- недостатнє освітлення робочої зони	порушення гігієнічних параметрів виробничого середовища	3	ДБН В.2.5-28:2015 [38]
- підвищена яскравість світла	порушення умов праці (організації місця праці-налагодження моніторів)	1	ДСанПіН 3.3.2.007-98 [48]
- понижена контрастність	-//-	1	ДСанПіН 3.3.2.007-98 [48]
психофізіологічні:			
- нервово-психічна перевантаження (розумове, перенапруження аналізаторів-зорових)	- пошук інформації для постановки теми; - пошук та аналіз аналогів і літератури; - пошук наявних технологій, моделювання та аналіз алгоритмів; - виконання роботи за темою диплома, тестування; - оформлення роботи	4	НПАОП 0.00-1.28-10 [49] ДСанПіН 3.3.2.007-98 [48]
- фізичні (статичне – сидіння)	порушення умов праці (організації місця праці-сидіння користувача,) та організації робочого часу - безперевна робота)	2	НПАОП 0.00-1.28-10 [49] ДСанПіН 3.3.2.007-98 [48]

Робочі місця повинні відповідати вимогам «ДСанПіН 2.2.7.029. Гігієнічні вимоги щодо поводження з промисловими відходами та визначення їх класу небезпеки для здоров'я населення». [40]. В умовах роботи з ПК виникають такі небезпечні і шкідливі фактори: несприятливі мікрокліматичні умови, освітлення, електромагнітні випромінювання, забруднення повітря шкідливими речовинами, шум, вібрація, електричний струм, електростатичне поле, напруженість трудового процесу та інше.

4.3.2 Пожежна безпека

Небезпека розвитку пожежі на обчислювальному центрі обумовлюється застосуванням розгалужених систем електроживлення ЕОМ, вентиляції та кондиціонування. Небезпека загорання пов'язана з особливістю комп'ютерів - з великою кількістю щільно розташованих на платі і блоках електронних вузлів і схем, електричних і комутаційних кабелів, резисторів, конденсаторів, напівпровідникових діодів і транзисторів. При відхиленні реальних умов експлуатації від розрахункових можуть виникнути пожежонебезпечні ситуації.

Наявність повного ізоляційного матеріалу, ймовірних джерел запалювання в вигляді електричних іскор і дуг, розгалуженість і недоступність роблять кабельні лінії місцем найбільш ймовірного виникнення і розвитку пожежі. Для зниження займистості і здатності поширювати полум'я кабелі покривають вогнезахисними покриттями.

Згідно «ГОСТ 12.1.044-89 ССБТ. Пожаровзрывоопасность веществ и материалов. Номенклатура показателей и методы их определения» [37] таке приміщення, площею 24 м^2, відноситься до категорії "В" (пожежонебезпечної).

Відповідно до норм первинних засобів пожежогасіння пропонується використовувати:

- повсть 1 1 м^2, кошму 2 × 1,5 м^2 або азбестове полотно 2 × 2 м^2 у кількості 1 шт.;

Виникнення пожежі можливо, якщо на об'єкті є горючі речовини, окислювач і джерела запалювання. Імовірність пожежної небезпеки приймається значною, якщо ймовірна взаємодія цих трьох чинників. Горючими компонентами є: будівельні матеріали для акустичної і естетичної обробки приміщень, перегородки, підлоги, двері, ізоляція силових, сигнальних кабелів і т.д.

Продуктами згоряння, що виділяються під час пожежі, є: окис вуглецю; сірчистий газ; окис азоту синильна кислота акромін; фосген; хлор та ін. При горінні пластмас, крім звичних продуктів згоряння, виділяються різні продукти термічного розкладання: хлорангідрідні кислоти, формальдегіди, хлористий водень, фосген, синильна кислота, аміак, фенол, ацетон, стирол «ГОСТ 12.1.044-89 ССБТ. Пожаровзрывоопасность веществ и материалов. Номенклатура показателей и методы их определения» [37].

4.3.3 Електробезпека

На робочому місці виконуються наступні вимоги електробезпеки: ПК, периферійні пристрої та обладнання для обслуговування, електропроводи і кабелі за виконанням та ступенем захисту відповідають класу зони по ПУЕ (правила улаштування електроустановок), мають апаратуру захисту від струму короткого замикання та інших аварійних режимів. Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування, виконана як окрема групова трьох провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів. Штепсельні з'єднання та електророзетки крім контактів фазового та нульового робочого провідників мають спеціальні контакти для підключення нульового захисного провідника. Електромережа штепсельних розеток для живлення персональних ПК, укладений по підлозі поруч зі стінами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання. Металеві труби і гнучкі металеві рукава заземлені. Захисне заземлення включає в себе заземлюючих пристроїв та провідник, що з'єднує заземлюючих пристроїв з обладнанням, яке заземлюється - заземлюючий провідник.

4.4 Гігієнічні вимоги до параметрів виробничого середовища

4.4.1 Мікроклімат

Мікроклімат робочих приміщень - це клімат внутрішнього середовища цих приміщень, який визначається діючої на організм людини з'єднанням температури, вологості, швидкості переміщення повітря. В даному приміщенні проводяться роботи, що виконуються сидячи і не потребують динамічного фізичного напруження, то для нього відповідає категорія робіт Іа. Отже оптимальні значення для температури, відносної вологості і рухливості повітря для зазначеного робочого місця відповідають «ДСН 3.3.6.042-99 Санітарні норми мікроклімату виробничих приміщень» [35] і наведені в табл. 4.2:

Дане приміщення обладнане системами опалення, кондиціонування повітря та припливно-витяжною вентиляцією. У приміщенні на робочому місці забезпечуються оптимальні значення параметрів мікроклімату: температури, відносної вологості і рухливості повітря відповідно до [35]. Рівні позитивних і негативних іонів в повітрі повинні відповідати [35]. Для забезпечення оптимальних параметрів мікроклімату в приміщенні проводяться перерви у роботі співробітників, з метою його провітрювання.

Таблиця 4.2 - Норми мікроклімату робочої зони об'єкта

Період року	Категорія робіт	Температура C^0	Відносна вологість %	Швидкість руху повітря, м/с
Холодна	легка-1 а	22 - 24	40 – 60	0,1
Тепла	легка-1 а	23 - 25	40 – 60	0,1

Контроль параметрів мікроклімату в холодний і теплий період року здійснюється не менше 3-х разів на зміну (на початку, середині, в кінці).

4.4.2 Освітленість

Світло є природною умовою існування людини. Воно впливає на стан вищих психічних функцій і фізіологічні процеси в організмі. Гарне освітлення діє тонізуюче, створює гарний настрій, покращує перебіг основних процесів вищої нервової діяльності. При поганому освітленні людина швидко втомлюється, працює менш продуктивно, виникає потенційна небезпека помилкових дій і нещасних випадків.

Освітленість приміщення має велике значення при роботі на ПЕОМ. Вона багато в чому визначається колірною і мережевий обстановкою. Відповідно до «ДБН В.2.5-28-2006 Природне і штучне освітлення» [38] для зменшеного поглинання світла стеля і стіни вище панелей (1,5-1,7м.). Якщо вони не облицьовані звукопоглинальним матеріалом, фарбуються білою водоемульсійною фарбою (коефіцієнт відбиття повинен бути не менше 0,7). Для забарвлення стіни панелей рекомендується віддавати перевагу світлим фарбам.

Основний потік природного світла при цій повинен бути зліва. Не допускається спрямування основного світлового потоку природного світла праворуч, ззаду і спереду працівника на ПЕОМ.

Природне освітлення, коли робочі місця з ПЕОМ розташовуються в один ряд по довжині приміщення на відстані 0,8 - 1,0 м від стіни з віконними прорізами, і екрани знаходяться перпендикулярно цієї стіни. Основний потік природного світла при цій повинен бути зліва. Не допускається спрямування основного світлового потоку природного світла праворуч, ззаду і спереду працює на ПЕОМ. Оптимальна відстань очей до екрана відео монітора повинна становити 60-70 см, допустимо не менше 50 см. Розглядати інформацію ближче 50 см не рекомендується.

У проекті, що розробляється передбачається використовувати суміщене освітлення. У світлий час доби буде використовуватися природне освітлення приміщення через віконні прорізи, в решту часу буде використовуватися штучне освітлення. Штучне освітлення

створюється газорозрядними лампами.

Штучне освітлення в робочому приміщенні передбачається здійснювати з використанням люмінесцентних джерел світла у світильниках загального освітлення. При експлуатації ЕОМ виконується зорова робота IV в розряд точності (середня точність). При цьому нормована освітленість на робочому місці (Ен) дорівнює 200 лк. Джерелом природного освітлення є сонячне світло.

Розрахунок освітлення. Для виробничих і адміністративних приміщень світловий коефіцієнт приймається не менш -1/8, в побутових - 1/10:

$$S_b = \left(\frac{1}{5} \div \frac{1}{10}\right) \cdot S_n, \tag{4.1}$$

де S_b – площа віконних прорізів, м^2;

S_n – площа підлоги, м^2.

$$Sn = a \cdot b = 4 \cdot 6 = 24 \text{ м}^2, \tag{4.2}$$

$$S = 1/8 \cdot 24 = 3 \text{ м}^2.$$

Приймаємо 2 вікна площею S=3 м^2 кожне.

Світильники загального освітлення розташовуються над робочими поверхнями у рівномірно-прямокутному порядку. Для організації освітлення в темний час доби передбачається обладнати приміщення, довжина якого становить 5 м, ширина 5 м, світильниками ЛПО2П, оснащеними лампами типу ЛБ (дві по 80 Вт) зі світловим потоком 5400 лм кожна.

Розрахунок штучного освітлення проводиться за коефіцієнтами використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні. Розрахунок кількості світильників n проводиться за формулою (4.3):

$$n = \frac{E \cdot S \cdot Z \cdot K}{F \cdot U \cdot M}, \tag{4.3}$$

де E – нормована освітленість робочої поверхні, визначається нормами – 300 лк;

S – освітлювана площа, м^2; S = 24 м^2;

Z – поправочний коефіцієнт світильника (Z = 1,15 для ламп розжарювання і ДРЛ; Z = 1,1 для люмінесцентних ламп) приймаємо рівним 1,1;

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації – 1,5;

U – коефіцієнт використання, що залежить від типу світильника, показника індексу приміщення і т.п. – 0,575

М – число люмінесцентних ламп в світильнику – 2;

F – світловий потік лампи – 5400лм (для ЛБ-80).

Підставивши числові значення в формулу (4.2), отримуємо:

$$n = \frac{300 * 24 * 1.1 * 1.5}{5400 * 0.575 * 2} \approx 1.9 \quad (4.4)$$

4.5 Заходи з організації виробничого середовища і попередження виникнення надзвичайних ситуацій

Розрахунок захисного заземлення (забезпечення електробезпеки будівлі).

Відповідно до класифікації приміщень за ступенем небезпеки ураження електричним струмом [50], приміщення в якому проводяться всі роботи належить до першого класу (без підвищеної небезпеки). Під час роботи використовуються електроустановки з напругою живлення 36 В, 220 В, і 360 В. Опір контуру заземлення повинен мати не більше 4 Ом.

Розрахунок проводять за допомогою методу коефіцієнта використання (екранування) електродів. Коефіцієнт використання групового заземлення η - це ставлення чинної провідності цього заземлення до найбільш можливої його провідності при нескінченно великих відстаней між його електродами. Коефіцієнт використання вертикальних заземлювачів $\eta_в$ у залежності від розміщення заземлювачів і їх кількості знаходиться в межах 0,4 ... 0,99. Взаємну екрануючого дії горизонтального заземлювача (сполучної смуги) враховують за допомогою коефіцієнта використання горизонтального заземлювача $\eta_с$.

Послідовність розрахунку:

1) Визначається необхідний опір штучних заземлювачів $R_{шт.з.}$:

$$R_{шт.з.} = \frac{R_д \cdot R_{пр.з.}}{R_{пр.з.} - R_д}, \quad (4.5)$$

де $R_{пр.з.}$ – опір природних заземлювачів; $R_д$ – допустимий опір заземлення.

Якщо природні заземлювачі відсутні, то $R_{шт.з.}=R_д$.

Підставивши числові значення в формулу (4.5), отримуємо:

$$R_{шт.з.} = \frac{4 \cdot 40}{40-4} \approx 4 \text{ Ом} \quad (4.6)$$

2) Опір заземлення в значній мірі залежить від питомого опору ґрунту ρ, Ом • м. Приблизне значення питомої опору глини приймаємо ρ=40 Ом•м (табличне значення).

3) Розрахункова питомий опір ґрунту, $\rho_{розр}$, Ом•м, визначається відповідно для вертикальних заземлювачів $\rho_{розр.в}$, і горизонтальних $\rho_{розр.г}$, Ом•м по формулі:

$$\rho_{розр..} = \psi \cdot \rho, \quad (4.7)$$

де ψ – коефіцієнт сезонності для вертикальних заземлювачів і кліматичної зони з

нормальною вологістю землі, приймається для вертикальних заземлювачів $\rho_{розр.в}$=1,7 і горизонтальних $\rho_{розр.г}$=5,5 Ом•м.

$$\rho_{розр.в} = 1,7 \cdot 40 = 68 \text{ Ом•м}$$
$$\rho_{розр.г} = 5,5 \cdot 40 = 220 \text{ Ом•м} \quad (4.8)$$

4) Розраховується опір розтікання струму вертикального заземлення $R_в$, Ом, по (4.5).

$$R_в = \frac{\rho_{розр.в}}{2 \cdot \pi \cdot l_в} \cdot \left(\ln \frac{2 \cdot l_в}{d_{ст}} + \frac{1}{2} \cdot \ln \frac{4 \cdot t + l_в}{4 \cdot t - l_в} \right), \quad (4.9)$$

де $l_в$ – довжина вертикального заземлювача (для труб - 2−3 м; $l_в$=3 м);

$d_{ст}$ – діаметр стрижня (для труб - 0,03−0,05 м; $d_{ст}$=0,05 м);

t – відстань від поверхні землі до середини заземлювача, яка визначається за формулою (4.10):

$$t = h_в + \frac{l_в}{2}, \quad (4.10)$$

де $h_в$ – глибина закладення вертикальних заземлювачів (0,8 м); тоді

$$t = 0,8 + \frac{3}{2} = 2,3 \text{ м} \quad (4.11)$$

$$R_в = \frac{68}{2 \cdot \pi \cdot 3} \cdot \left(\ln \frac{2 \cdot 3}{0,05} + \frac{1}{2} \cdot \ln \frac{4 \cdot 2,3 + 3}{4 \cdot 2,3 - 3} \right) = 18,5 \text{ Ом} \quad (4.12)$$

5) Визначається теоретична кількість вертикальних заземлювачів n штук, без урахування коефіцієнта використання $\eta_в$:

$$n = \frac{2 \cdot R_в}{R_д} = \frac{2 \cdot 18,5}{4} = 9,25 \quad (4.13)$$

І визначається коефіцієнт використання вертикальних електродів групового заземлення без урахування впливу сполучної стрічки $\eta_в$ =0,57 (табличне значення).

6) Визначається необхідна кількість вертикальних заземлювачів з урахуванням коефіцієнта використання $n_в$, шт:

$$n_в = \frac{2 \cdot R_в}{R_д \cdot \eta_в} = \frac{2 \cdot 18,5}{4 \cdot 0,57} = 16,2 \approx 16 \quad (4.14)$$

7) Визначається довжина сполучної стрічки горизонтального заземлювача $l_с$, м:

$$l_с = 1,05 \cdot L_в \cdot (n_в - 1), \quad (4.15)$$

де $L_в$ – відстань між вертикальними заземлювачами, (прийняти $L_в$ = 3м);

$n_в$ – необхідну кількість вертикальних заземлювачів.

$$l_c = 1,05 \cdot 3 \cdot (16-1) \approx 48 \text{ м} \quad (4.16)$$

8) Визначається опір розтіканню струму горизонтального заземлювача (сполучної стрічки) $R_г$, Ом:

$$R_г = \frac{\rho_{розр.г}}{2 \cdot \pi \cdot l_c} \cdot \ln \frac{2 \cdot l_c^{\,2}}{d_{см} \cdot h_г}, \quad (4.17)$$

де $d_{см}$ – еквівалентний діаметр смуги шириною b, $d_{см}$ =0,95b, b = 0,15 м;

$h_Г$ – глибина закладення горизонтальних заземлювачів (0,5 м);

l_c - довжина сполучної стрічки горизонтального заземлювача l_c, м

$$R_г = \frac{220}{2 \cdot \pi \cdot 48} \cdot \ln \frac{2 \cdot 48^2}{0,95 \cdot 0,15 \cdot 0,5} = 8,1 \text{ Ом} \quad (4.18)$$

9) Визначається коефіцієнт використання горизонтального заземлювача ηс, відповідно до необхідної кількості вертикальних заземлювачів $n_в$.

Коефіцієнт використання сполучної смуги η_c=0,3 (табличне значення).

10) Розраховується результуючий опір заземлюючого електрода з урахуванням сполучної смуги:

$$R_{заг} = \frac{R_в \cdot R_г}{R_в \cdot \eta_c + R_г \cdot n_в \cdot \eta_в} \le R_д. \quad (4.19)$$

Висновок: дане захисне заземлення буде забезпечувати електробезпека будівлі, так як виконується умова: $R_{заг}$ <4 Ом, а саме:

$$R_{заг} = \frac{18,5 \cdot 8,1}{18,5 \cdot 0,3 + 8,1 \cdot 16 \cdot 0,57} = 1,9 \le R_д \quad (4.20)$$

При виникненні пожеж при роботі на ПЕОМ від таких можливими джерел запалювання як:

- іскри і дуги коротких замикань;

- перегрів провідників, резисторів і інших радіодеталей ПЕОМ, від тривалого перевантаження і наявність перехідного опору;

- іскри при розмиканні і розмиканні ланцюгів;

- розряди статичної електрики;

- необережне поводження з вогнем, а також вибухи газоповітряних і пароповітряних сумішей.

Важливу увагу слід звернути на пожежну безпеку підприємства в цілому і окремих його приміщень. У приміщеннях не повинно накопичуватися сміття, непотрібну папір, мотлох та ін. Речі, які не використовуються у виробничому процесі. Наявний вільний аварійний вихід за межі приміщення в разі пожежі, бути передбачені вогнегасники. Вони повинні бути в робочому стані і перевірятися відповідно до норм. У приміщеннях повинна бути пожежна сигналізація, вогнегасник. У разі виникнення пожежі необхідно повідомити в найближчу пожежну частину, убезпечити інших працівників і по можливості прийняти кроки щодо запобігання можливих наслідків та усунення пожежі.

4.6 Охорона навколишнього природного середовища

4.6.1 Загальні дані з охорони навколишнього природного середовища

Діяльність за темою магістерської роботи в процесі її виконання впливає на навколишнє природне середовище і регламентується нормами діючого законодавства [39-42].

Основним екологічним аспектом в процесі діяльності за даними спеціальностями є процеси впливу на атмосферне повітря та процеси поводження з відходами, які утворюються, збираються, розміщуються, передаються на видалення (знешкодження), утилізацію, тощо в ІТ галузі.

Немає впливу на атмосферне повітря при нормальних умовах праці, бо в приміщенні не використовуються сканери, принтери та інші джерела викиду забруднюючих речовин в повітря робочої зони.

В процесі діяльності користувача виникають процеси поводження з відходами ІТ галузі. Види відходів, утворення, яких можливо:

- відпрацьовані люмінесцентні лампи - I клас небезпеки;
- батарейки та акумулятори (малі) -III клас небезпеки;
- змінні носії інформації - IV клас небезпеки;
- відпрацьований ізолюючий матеріал, дроти та кабелі - IV клас небезпеки;
- макулатура - IV клас небезпеки;
- побутові відходи - IV клас небезпеки.

4.7 Висновок до розділу 4

В результаті проведеної роботи було зроблено аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом написаному в кваліфікаційній роботі, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника. Приведені рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки. Була наведено розміри приміщення та наведено значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника, а також – наведені інструкції з охорони праці, техніки безпеки при роботі на комп’ютері.

ВИСНОВКИ

Метою магістерської атестаційної роботи було дослідження стійкості криптографічних алгоритмів. Оцінка існуючих методів шифрування та аналіз інструментів, взаємодіючих із шифротекстом і ключами шифрування.

Була розкрита актуальність проблеми забезпечення ефективної, надійної та безпечної передачі даних та збереження їх цілісності і пошук надійного захисту від деформування сторонніми особами.

У першому розділі роботи було проаналізовано теоретичні питання дослідження криптографічної стійкості методів шифрування та їх методи реалізації. Були розглянуті недоліки систем шифрування та основні види атак.

Була поставлена задача на дану магістерську роботу, а саме дослідження методів шифрування та інструментів підвищення криптостійкості, та постановка задачі, щодо проведення порівняльної характеристики, для виявлення методів шифрування, що задовольняють усім критеріям стійкості, економічності як фінансової так і технічної складової, та складності реалізації.

У другому розділі даної роботи були розглянуті основні методи підвищення криптографічної стійкості, такі як електронний цифровий підпис для контролю та аутентифікації даних, розглянутий принцип роботи хеш-функцій та основні їх алгоритми, та проведений первинний аналіз цих методів.

За результатами огляду для досліду були обрані хеш-функції серій RSA, SHA та Whirlpool для подальшого дослідження та проведення детального аналізу за допомогою розрахункових та аналітичних методів перевірки криптостійкості.

В третьому розділі даної дипломної роботи були описані заходи по вдосконаленню методів шифрування. Проведено аналіз крипостійкості алгоритму RSA та методів хешування за допомогою програмного засобу CrypTool 2 на базі якого було здійснена симуляція роботи алгоритмів та проведений аналіз їх взаємодії із ключам шифрування, за результатам якого було отримано залежність, щодо значного приросту криптостійкості при збільшені розміру криптографічних ключів. Проведений ряд атак на алгоритм MD5 за допомогою програми PasswordsPro 3.1, за результатами якого було доведено значний зріст рівня криптостійкості за допомогою впровадження до системи шифрування хеш-функції, котрі зберігають цілісність первинних даних та виступають як інструмент для аутентифікації при обміні даних у системі клієнт-сервер. Було надано інструменти та приклади реалізації систем шифрування RSA та хешування на прикладі MD5 алгоритму, за допомогою мови програмування C#. Проведення

атаки методом факторизації Ферма показало, що при наявності у третьої сторони (атакуючого) даних для шифрування та шифрованого повідомлення, можливий повний злом системи шифрування, що показує залежність криптостійкості системи також від наявності закритого каналу зв’язку між абонентами.

У четвертому розділі проведений аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом написаному в дипломній роботі, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника. Приведені рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки. Наведені розміри приміщення та визначені значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника, а також – наведені інструкції з охорони праці, техніки безпеки при роботі на комп’ютері, визначені основні екологічні аспекти та вплив на навколишнє природнє середовище та зазначені заходи щодо поводження з ними.

ВИКОРИСТАНІ ДЖЕРЕЛА

1. С. Гнатюк, Особливості криптографічного захисту державних інформаційних ресурсів/ С. Гнатюк, В. Кінзерявий, А. Охріменко. - Безпека інформації. - 2012. - № 1. - С. 68.
2. І.В. Лисенко,. Порівняльна характеристика можливостей програмних платформ і мов програмування з точки зору реалізації криптоалгоритмів / І.В. Лисенко, Ю.В. Трегуб - Системи управління, навігації та зв'язку. – 2017- випуск 1(41).
3. J. Gallier,J. Quaintance. Notes on Primality Testing And Public Key Cryptography Part1. – 2010.
4. Е. Мейволд. Безпека мереж. - 2-е видав. , редаг. — М.: Інтуіт, 2016. - 528 с.
5. Алферов А.Ю. Основы криптографии./ Алферов А.Ю., Зубов А.С. – М.: Наука, 2004- 423с.
6. Reinke E. C. Classical Cryptography // The Classical Journal — Classical Association of the Middle West and South, 1962. — Vol. 58, Iss. 3. — P. 113–121
7. Бабаш А.В. История криптографии / Бабаш А.В., Шанкин Г.П - Часть I. — М.: Гелиос АРВ, 2002. — 240 с.
8. Sara RobinsonStill. Guarding Secrets after Years of Attacks, RSA Earns Accolades for its Founders – June 2003. - SIAM News, Volume 36
9. Романец Ю.В., Защита информации в компьютерных системах и сетях / Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. / Под ред. В.Ф. Шаньгина. – 2-е изд., перераб. и доп. – М.: Радио и связь, 2001. – 376 с.
10. Аграновский А.В Практическая криптография. Алгоритмы и их программирование / Аграновский А.В., Хади Р.А. — М: Солон-Пресс, 2009. — 258 с.
11. Петров А. А. Компьютерная безопасность. Криптографические методы защиты. – М.: ДМК, - 2000г. -150 с.
12. Common injection attacks - Режим доступу: www. URL: http://www.webcitation.org/65XH8EqmO – 07.11.2018р.
13. Д.А. Кронберг. Квантова криптографія / Д.А. Кронберг, Ю.І. Ожигов, Довідковий посібник - 2012р. – 112с.
14. ЗАКОН УКРАЇНИ № 2155-VIII від 05.10.2017. Про електронний цифровий підпис.
15. ДСТУ 4145-2002 "Криптографічний захист ін- формації. Цифровий підпис, що ґрунтується на еліптич- них кривих. Формування та перевіряння"
16. Alfred J. Menezes, Handbook of applied cryptography / Alfred J. Menezes, Paul C.

van Oorschot, Scott A. Vanstone. — CRC-Press, 1996. — p. 32.

17. Брюс Шнайер. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. — М.: Триумф, 2002

18. Elgamal T. A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms/ Trans. Inf. — IEEE, 1985. — Vol. 31, Iss. 4. — P. 469–472.

19. А. П. Алферов Основы криптографии/ А. П. Алферов, А. Ю. Зубов, А. С. Кузьмин, А. В. Черемушкин - М.: Гелиос АРВ, 2002г. – 480с.

20. Семенов Ю. А. Алгоритм DES – Режим доступу: www. URL: http://book.itep.ru/1/intro1.htm - 07.12.2018р.

21. Методы и средства защиты компьютерной информации – Режим доступу: www. URL: http://www.volpi.ru/umkd/zki/index.php?man=1&page=20 – 08.12.2018

22. Stevens M., Lenstra A. K., Weger B. d. Chosen-prefix collisions for MD5 and applications // International Journal of Applied Cryptography — Inderscience Publishers, 2012. — Vol. 2, Iss. 4. — P. 322–359.

23. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке С. — М.: Триумф, 2002. — 816 с.

24. Криптографическая хеш-функция - Режим доступу: www. URL: https://ru.wikipedia.org/wiki/Криптографческая_хеш-функция

25. RSA Key generator - Режим доступу: http://www.brianveitch.com/cryptography/generate_rsa_small_keys.php - 10.10.2018р.

26. RSA Calculator - Режим доступу: www. URL: https://www.cs.drexel.edu/~jpopyack/IntroCS/HW/RSAWorksheet.html - 01.10.2018р.

27. Download CrypTool portal - Режим доступу: www. URL: https://www.cryptool.org/de/ct2-downloads - 01.10.2018р.

28. Факторзация целых чисел - Режим доступу: www. URL: https://ru.wikipedia.org/wikiФакторзация_целых_чисел - 01.10.2018р.

29. Stevens M., Lenstra A. K., Weger B. d. Chosen-prefix collisions for MD5 and applications // International Journal of Applied Cryptography — Inderscience Publishers, 2012. — Vol. 2, Iss. 4. — p. 322–359.

30. Stallings W. Cryptography and network security principles and practices // International Journal - Pearson Education, Inc, 2016 – 4th edition – p.268.

31. Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці– від 15 лютого 2005 р. за № 231/10511

32. НПАОП 0.00-4.12- 05 Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці.

33. НПАОП 0.00-4.15-98 Про розробку інструкцій з охорони праці

34. НПАОП 0.00-1.31-99 Правилами охорони праці під час експлуатації електронно-обчислювальних машин

35. ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень»

36. НАПБ Б.03.002-2007 Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою

37. ГОСТ 12.1.044-89 ССБТ. Пожаровзрывоопасность веществ и материалов. Номенклатура показателей и методы их определения

38. ДБН В.2.5-28:2015 Природне і штучне освітлення

39. Закон України Закон України «Про захист населення і територій від надзвичайних ситуацій техногенного та природного характеру».

40. ДСанПіН 2.2.7.029. «Гігієнічні вимоги щодо поводження з промисловими відходами та визначення їх класу небезпеки для здоров'я населення».

41. Закон України «Про металобрухт».

42. ДК 005-96 Державний класифікатор України. Класифікатор відходів.

43. ДСН 3.3.6.037-99 Санітарні норми виробничого шуму, ультразвуку та інфразвуку

44. ГОСТ 12.1.012-90 ССБТ Вібраційна безпека. Загальні вимоги

45. ГОСТ 12.1.006-84 ССБТ Електромагнітні поля радіочастот. Допустимі рівні на робочих місцях і вимоги до проведення контролю

46. ГОСТ 12.1.030-81 ССБТ Електробезпека. Захисне заземлення. Занулення (зі Зміною N 1)

47. ГОСТ 13109-97 Норми якості електричної енергії в системах електропостачання загального призначення

48. ДСанПіН 3.3.2.007-98 Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин.

49. НПАОП 0.00-1.28-10. Про затвердження правил охорони праці під час експлуатації електронно-обчислювальних машин

50. Студопедія - Класифікації приміщень за ступенем небезпеки ураження електричним струмом - Режим доступу: www. URL: https://studopedia.info/3-7460.html

51. Про затвердження нормативно-правових актів з питань інформаційної безпеки: постанова Національного банку від 26.11.2015 № 829

52. Download WinRTgen v.2.8.3 portal - Режим доступу: www. URL: http://www.oxid.it/downloads/winrtgen.zip - 01.10.2018р.

ДОДАТОК А

Код програми для обрахування хешу MD5

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Security.Cryptography;


namespace ComputeAHash_csharp
{
    class Class1
    {

        static void Main(string[] args)
        {
            string sSourceData;
            byte[] tmpSource;
            byte[] tmpHash;
            Console.WriteLine("Введіть текст: ");
            sSourceData = Console.ReadLine();
            //Створення масив байтів з вхідних даних
            tmpSource = ASCIIEncoding.ASCII.GetBytes(sSourceData);

            //Обчислювання хешу на основі вхідних даних
            tmpHash = new MD5CryptoServiceProvider().ComputeHash(tmpSource);
            Console.WriteLine("Отриманий хеш: ");
            Console.WriteLine(ByteArrayToString(tmpHash));
        }

        static string ByteArrayToString(byte[] arrInput)
        {
            int i;
            StringBuilder sOutput = new StringBuilder(arrInput.Length);
            for (i = 0; i < arrInput.Length; i++)
            {
                sOutput.Append(arrInput[i].ToString("X2"));
            }
            return sOutput.ToString();
        }


    }
}
```

ДОДАТОК Б

Код програми шифратора-дешифратора RSA

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

namespace protect_inf_LR1
{
    public partial class Form1 : Form
    {
        char[] characters = new char[] { '#', 'А', 'Б', 'В', 'Г', 'Ґ', 'Д', 'Е', 'Є', 'Ж',
                                         'З', 'И', 'І', 'Ї', 'Й', 'К', 'Л', 'М', 'Н', 'О',
                                         'П', 'Р', 'С', 'Т', 'У', 'Ф', 'Х', 'Ц', 'Ч', 'Ш',
                                         'Щ', 'Ь', 'Ю', 'Я', 'Ё', 'Ы', 'Ъ', 'Э', ' ', '1',
                                         '2', '3', '4', '5', '6', '7', '8', '9', '0' };

        public Form1()
        {
            InitializeComponent();
        }

        //зашифрувати (робота із файлами)
        private void buttonEncrypt_Click(object sender, EventArgs e)
        {
            if ((textBox_p.Text.Length > 0) && (textBox_q.Text.Length > 0))
            {
                long p = Convert.ToInt64(textBox_p.Text);
                long q = Convert.ToInt64(textBox_q.Text);

                if (IsTheNumberSimple(p) && IsTheNumberSimple(q))
                {
                    string s = "";
                    StreamReader sr = new StreamReader("in.txt");
                    while (!sr.EndOfStream)
                    {
                        s += sr.ReadLine();
                    }
                    sr.Close();
                    s = s.ToUpper();

                    long n = p * q;
                    long m = (p - 1) * (q - 1);
                    long d = Calculate_d(m);
                    long e_ = Calculate_e(d, m);

                    List<string> result = RSA_Endoce(s, e_, n);
                    StreamWriter sw = new StreamWriter("out1.txt");
                    foreach (string item in result)
                        sw.WriteLine(item);
                    sw.Close();
```

```
            textBox_d.Text = d.ToString();
            textBox_n.Text = n.ToString();

            Process.Start("out1.txt");
        }
        else
            MessageBox.Show("p или q - не простые числа!");
    }
    else
        MessageBox.Show("Введите p и q!");
}

//розшифрувати (робота із файлами)
private void buttonDecipher_Click(object sender, EventArgs e)
{
    if ((textBox_d.Text.Length > 0) && (textBox_n.Text.Length > 0))
    {
        long d = Convert.ToInt64(textBox_d.Text);
        long n = Convert.ToInt64(textBox_n.Text);

        List<string> input = new List<string>();
        StreamReader sr = new StreamReader("out1.txt");
        while (!sr.EndOfStream)
        {
            input.Add(sr.ReadLine());
        }
        sr.Close();
        string result = RSA_Dedoce(input, d, n);
        StreamWriter sw = new StreamWriter("out2.txt");
        sw.WriteLine(result);
        sw.Close();
        Process.Start("out2.txt");
    }
    else
        MessageBox.Show("Введите секретный ключ!");
}

//перевірка на простоту числа
private bool IsTheNumberSimple(long n)
{
    if (n < 2)
        return false;
    if (n == 2)
        return true;
    for (long i = 2; i < n; i++)
        if (n % i == 0)
            return false;
    return true;
}

//зашифрувати
private List<string> RSA_Endoce(string s, long e, long n)
{
    List<string> result = new List<string>();
    BigInteger bi;
    for (int i = 0; i < s.Length; i++)
    {
        int index = Array.IndexOf(characters, s[i]);
        bi = new BigInteger(index);
        bi = BigInteger.Pow(bi, (int)e);
        BigInteger n_ = new BigInteger((int)n);

        bi = bi % n_;
        result.Add(bi.ToString());
    }
```

```
            return result;
        }

        //розшифрувати
        private string RSA_Dedoce(List<string> input, long d, long n)
        {
            string result = "";
            BigInteger bi;

            foreach (string item in input)
            {
                bi = new BigInteger(Convert.ToDouble(item));
                bi = BigInteger.Pow(bi, (int)d);
                BigInteger n_ = new BigInteger((int)n);
                bi = bi % n_;
                int index = Convert.ToInt32(bi.ToString());
                result += characters[index].ToString();
            }
            return result;
        }

        //обчислення параметру d. повинен бути взаимно простим із m
        private long Calculate_d(long m)
        {
            long d = m - 1;
            for (long i = 2; i <= m; i++)
                if ((m % i == 0) && (d % i == 0))
                {
                    d--;
                    i = 1;
                }

            return d;
        }

        //обчислення параметру e
        private long Calculate_e(long d, long m)
        {
            long e = 10;
            while (true)
            {
                if ((e * d) % m == 1)
                    break;
                else
                    e++;
            }
            return e;
        }

    }
}
```

ДОДАТОК В

Слайди презентації

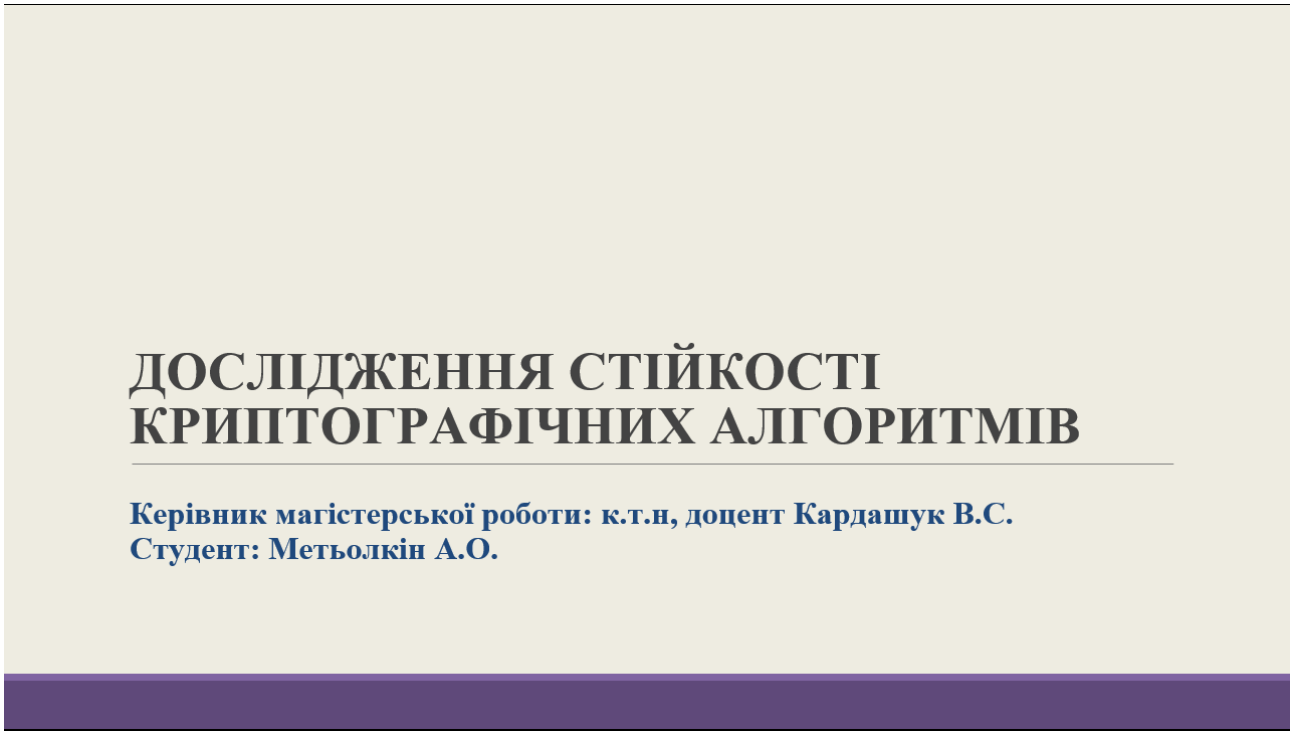


Рисунок В.1 – Слайд №1

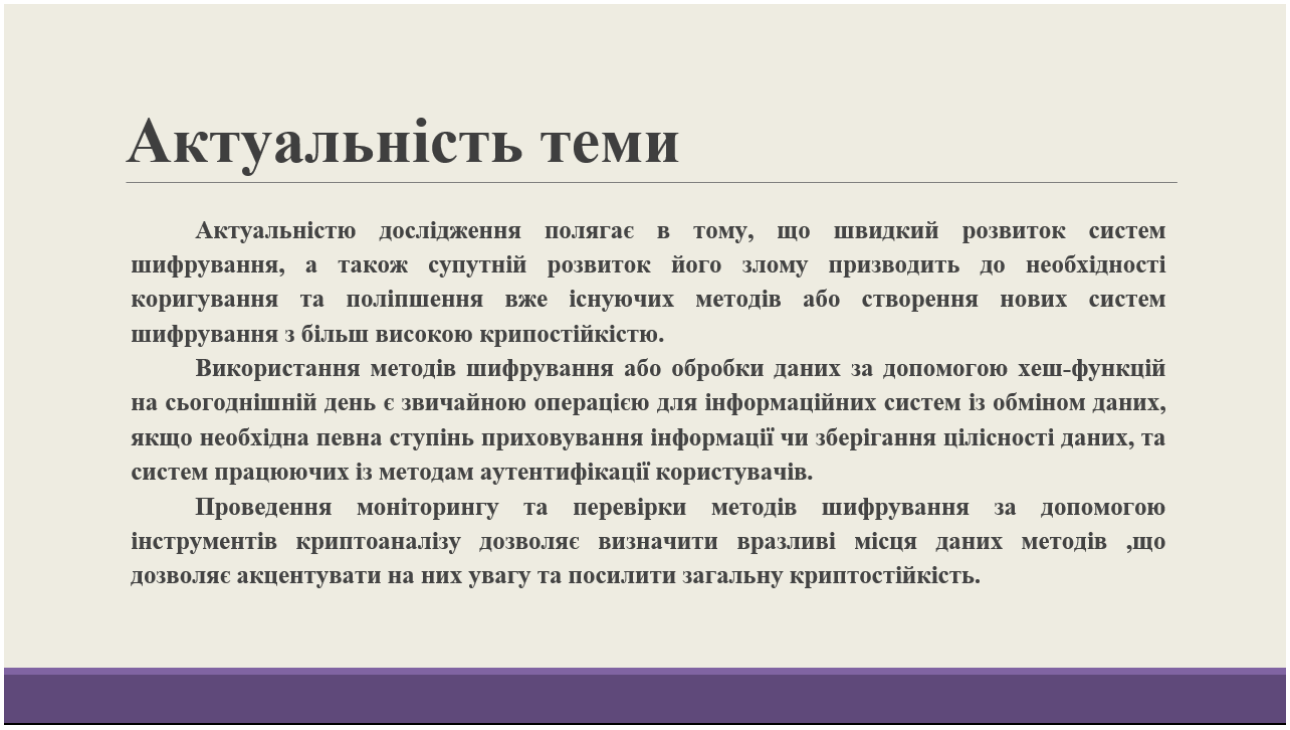


Рисунок В.2 – Слайд №2

Аналіз досліджень і публікацій

За основу для дослідження було взято такі дослідження та публікації: у роботі С. Гнатюка, В. Кінзерявого, А. Охріменко. Особливості криптографічного захисту державних інформаційних ресурсів - проаналізована робота алгоритмів шифрування та постановка питання щодо криптостійкості алгоритмів шифрування із відкритими та закритими ключами. У роботі І.В. Лисенко, Ю.В. Трегуба Порівняльна характеристика можливостей програмних платформ і мов програмування з точки зору реалізації криптоалгоритмів - розглянуті теоретичні матеріали методів шифрування і їх порівняльна характеристика. У роботі J. Gallier,J. Quaintance. Notes on Primality Testing And Public Key - розглянуто стандарти реалізації алгоритму шифрування RSA, формули для розрахунків ключів шифрування, основні вимоги до алгоритму та його первинний криптоаналіз.

Рисунок В.3 – Слайд №3

Мета дослідження

Підвищення криптостійкості вже існуючих алгоритмів шифрування за допомогою таких методів, як хеш-функції, ітерація ключів шифрування та інших, із проведенням подальшої порівняльної характеристики, для виявлення найнадійнішої сукупності методів, за допомогою аналітичних, математичних та практичних методів аналізу проведення перевірки надійність та стійкості методів і за результатами дослідження поради, щодо подальшої експлуатації даних методів.

Рисунок В.4 – Слайд №4

Постановка задачі

Для досягнення поставленої мети, в магістерській атестаційній роботі сформульовані та вирішені наступні задачі:

- **дослідження та аналіз програмних засобів реалізації методів шифрування;**
- **визначення оптимального методу шифрування, що буде задовольняти визначеним параметрам криптографічної стійкості для приватних мереж;**
- **використання програмних та математичних засобів криптоаналізу, для визначення надійності методів шифрування;**
- **розробка програм для реалізації алгоритмів шифрування і використання отриманих параметрів для подальших розрахунків;**
- **за результатами дослідження отримання оцінки стійкості до криптоаналізу методів та інструментів для роботи із ключами шифрування і шифротекстом, із подальшою рекомендацією по експлуатації даних методів.**

Рисунок В.5 – Слайд №5

Алгоритм шифрования RSA

RSA - асиметричний алгоритми шифрування, є досить поширеним типом.

Алгоритм RSA включає в себе чотири етапи: генерація ключів, передача ключів, шифрування і розшифрування.

Алгоритм не є складним, через це можливе опрацювання великого обсягу даних, і формування як правило, розділене на обчислювальні блоки фіксованого довжини, кожен вхідний блок, незалежно не від чого шифрується окремо. Такі криптосистеми шифрування називаються блокові. Щоб підсилити самі методи шифрування, при цьому не деформуючи дані, використовуються спеціальні методи, які посилюють криптостійкість. Такі методи, як правило, працюють саме з ключами шифрування.

Рисунок В.6 – Слайд №6

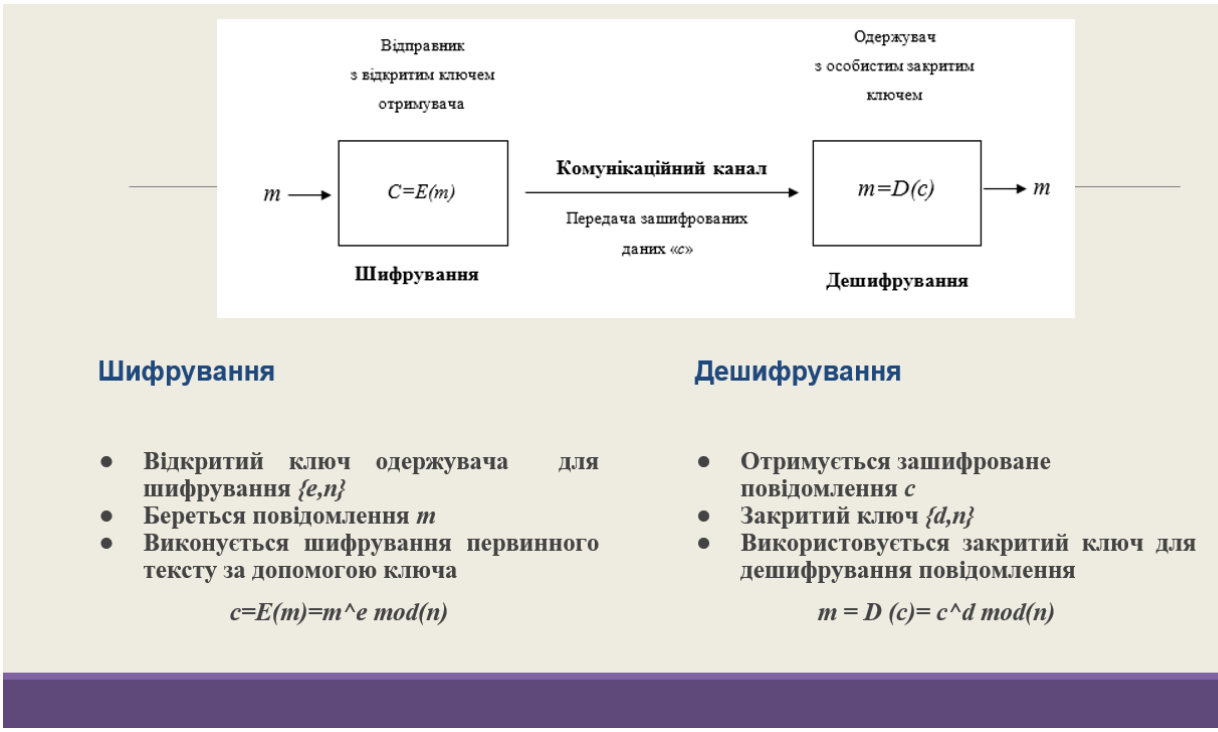


Рисунок В.7 – Слайд №7

Шифрування та методи ЕЦП

Відкритий і закритий ключ між собою пов'язані певними математичними залежностями, тому знанням параметрів неможливий реалізувати підбір ключів шифрування за короткий час. Перевірити, чи не була інформація спотворена під час передачі по зашифрованому каналу можна за допомогою ЕЦП. В цьому випадку текст зашифрований закритим ключем і разом з відкритим текстом відправляється відправнику. Якщо проводити дешифрування відкритим ключем і текст співпадає з не зашифрованим текстом, то даний процес був коректний. На практиці закритим ключем зазвичай зашифрована хеш-функція документа. Електронний підпис крім іншого містить так званий сертифікат відкритого ключа, виданий довіреним центром сертифікації, який має дані по всіх відкритих ключів та їх користувачам. Сертифікат також повинен мати підпис. Важливо, що відкритий ключ довіреного центру повинен бути відомий заздалегідь, інакше його теж можна підробити

Рисунок В.8 – Слайд №8

Процедура електронного підпису та застосування хеш-функцій

За основний засіб підвищення стійкості даних та системи шифрування у цілому використовується хеш-функції. Криптографічно стійка хеш-функція перетворює вихідний текст в один рядок так, що з цього рядка неможливо (або дуже складно) отримати вихідний текст. Хеш-функцію використовують, наприклад, там, де потрібно вводити пароль: фразу в рядку введення порівнюють не з самим паролем, а з його хеш-функцією

Рисунок В.9 – Слайд №9

Первинний аналіз хеш-функцій

Хеш-функція-перетворення тексту довільної довжини в текст фіксованої довжини.

Безпека хеш-функції може забезпечуватися складністю деякої математичної задачі при наявності доказів, що атаки, спрямовані на порушення вимог до неї, настільки ж складні, наскільки і рішення цього завдання.

№	Хеш-функція	Хеш-значення
1	SHA-1	8be3c943b1609fffbfc51aad666d0a04adf83c9d
2	SHA-256	e7cf3ef4f17c3999a94f2c6f612e8a888e5b1026878e4e19398b23bd38ec221a
3	SHA-3	690ace5d07a99566b98f337abba767367756f78bc0a867c924b1f99c4bc469473a0cf69f108b329b44887a32abc23254d253d30db62538f0ebe72b0dabbed0c1
4	SHA-512	e6c83b282aeb2e022844595721cc00bbda47cb24537c1779f9bb84f04039e1676e6ba8573e588da1052510e3aa0a32a9e55879ae22b0c2d62136fc0a3e85f8bb
5	MD2	9dc7dd5f9b5f681a133e64c0089330e7
6	MD4	f15abd57801840f3348ddccafb677f6a
7	MD5	dc647eb65e6711e155375218212b3964
8	CRC-32	9abfd710
9	BASE-64	UGFzc3dvcmQ=
10	Tiger-128	78db95bcce175ab6320959627749 65d1
11	Whirlpool	fd07ba63996cdcfa6130ee82acec65da7487f51564bb7c6ead6dabc6b9e8eac974e5d852edc545804ae68fa46fc59d4789acab50bbc22b26fb24412f8dc11cde
12	DES(Unix)	Q37hnXsCsGjCU

Рисунок В.10 – Слайд №10

Перехід до криптоаналізу

Алгоритми CRC-32, Base-64, DES (Unix) використовувати не бажано, так як згенеровані хеш-функції замалий хеш, тобто дуже легко піддаються злому. Про алгоритми MD2 і MD4 не являються актуальними, як і попередні версії алгоритму SHA. Алгоритм Whirlpool попри те, що є надійним, є недоліки при обмеженій кількості «раундів» шифрування.

З претендентів на реальне використання можна розглядати такі алгоритми серії SHA-512, MD5 та Whirpool.

Наступним кроком буде перевірка стійкості до атак хеш-функцій, на прикладі MD5.

Рисунок В.11 – Слайд №11

Криптоаналіз хеш-функцій

Найчастіше хеш-функції використовуються для перевірки унікальності пароля, файлу, рядка і т.д. Наприклад, завантажуючи файл з інтернету, можна побачити поруч з ним рядок виду b10a8db164e0754105b7a99be72e3fe5 - це і є хеш, прогнав цей файл через алгоритм MD5 буде отриманий подібний рядок, і, якщо хеші рівні, можна з великою ймовірністю стверджувати що цей файл дійсно справжній

Для проведення подальшого криптоаналізу необхідно реалізувати механізм роботи MD5, за основу буде братися розробка програмного забезпечення на мові програмування C#.

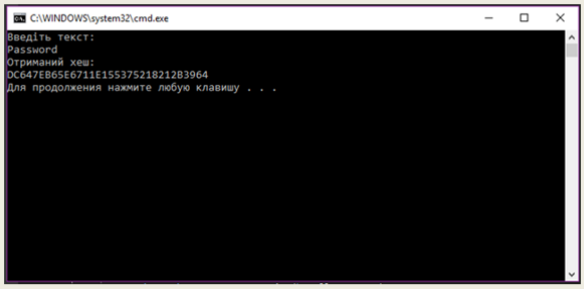
Для реалізації шифрування MD5 та інших існує бібліотека «System.Security.Cryptography»

Простір імен System.Security.Cryptography надає криптографічні служби, що включають безпечне кодування і декодування даних, а також цілий ряд інших функцій, таких як хешування, генерація випадкових чисел і перевірка справжності повідомлень.

Рисунок В.12 – Слайд №12

Також використовується метод ComputeHash() класу MD5CryptoServiceProvider, що обчислює хеш-значення для заданого об'єкту.

За допомогою даних інструментів була реалізована програма



Пароль	Хеш
my_password	a865a7e0ddbf35fa6f6a232e0893bea4
0000	4a7d1ed414474e4033ac29ccb8653d9b
8805553535	c36ce47f1dbe88efa9655e5a7bb1286b
qwerty	d8578edf8458ce06fbc5bb76a58c5ca4
1q2w3e4r5t6y7u8i9o0p	c6b419d72d1664762ad3d5c500566c69
0123456789	781e5e245d69b566979b86e28d23f2c7
qwerty123456	48474f975022f960bc2afbe49be581e8
tRvVW50h	47e9458cd6a0398095c38925c249d2fd

Було обрано декілька «паролів» і згенеровано за допомогою реалізованої програми з них хеш, після цього вони були збережені у текстовому файлі для подальшого опрацювання за допомогою програми PasswordsPro.

Рисунок В.13 – Слайд №13

У досліді будуть використані основні типи атак:

1. **Проста атака за словниками - в цій атаці відбувається проста перевірка хеш на паролі зі словників, зазвичай у словниках містяться найпростіші паролі типу "123", "qwerty", "99999" тощо, а також паролі, знайдені програмою раніше.**
2. **Brute Force Attack - це вичерпний пошук за всіма можливими паролями в певному діапазоні; наприклад, "аааааа" ... "zzzzzz", "0000" ... "9999" і т.д.**
3. **Rainbow Attack - ця атака намагається відновити паролі за допомогою попередньо розрахованих таблиць Rainbow.**

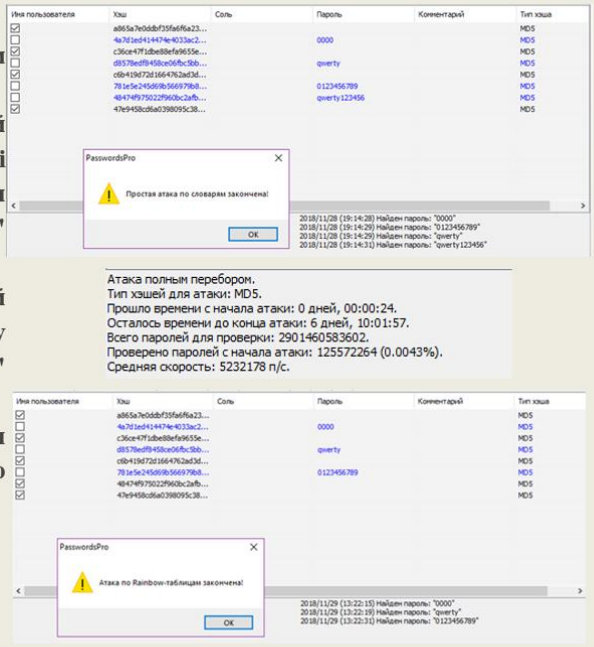


Рисунок В.14 – Слайд №14

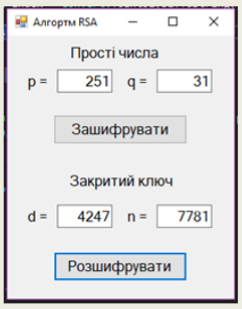
Криптоаналіз алгоритму RSA та хеш-функцій

Наступним кроком для перевірки криптоаналізу було обрано дослідження одного із найпопулярніших методів шифрування RSA у комплексі із методами хешування.

Було

Параметри p, q	Пара відкритого ключа e, n	Пара закритого ключа d, n
251, 31	83,7781	4247, 7781
1523, 113	251, 172099	29203, 172099
165173, 1231	647, 203327963	67511183, 203327963

критого ключа



Для перевірки працездатності алгоритму була розроблена програма.

Дана програма була розроблена за допомогою мови програмування C# і за допомогою стандартних бібліотек.

Програма реалізує процес поступового шифрування та дешифрування RSA із розрахунком закритого та відкритого ключів з початкових даних – простих чисел .

Рисунок В.15 – Слайд №15

Реалізація RSA алгоритму та хеш-функцій

Наступним етапом буде проведення криптоаналізу алгоритму RSA за допомогою допоміжних програмних засобів, таких як Cryptool 2, що працює з алгоритмами шифраторів, та допомагає змоделювати процес шифрування та отримувати необхідну статистику.

Для реалізації алгоритму RSA було сформовано схему її роботи із областю вводу даних (відкритий текст), конвертор у строки, компонент роботи алгоритму RSA, генератор ключів для RSA, вихідний текст.

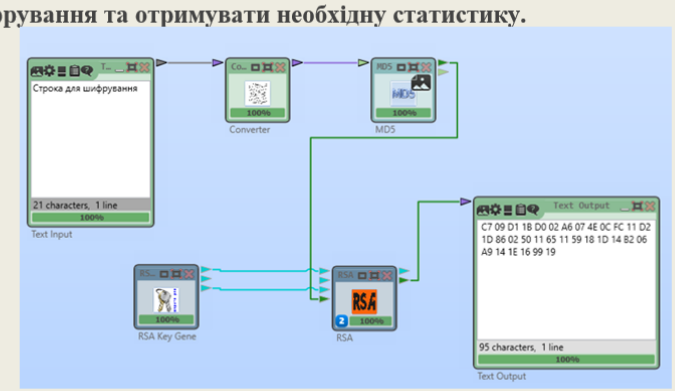


Рисунок В.16 – Слайд №16

У результаті були отримані дані, а саме об'єм символів із початкових 21 було отримано 239 вихідних символів при шифруванні алгоритмом RSA.

Наступним кроком буде посилення даної системи шифрування шляхом введення у систему хеш-елементів різних алгоритмів.

При використанні алгоритму RSA отриманий шифротекст є в 10 разів більший на відміну від початкового тексту, але його стійкість є занизькою. Хоча алгоритми RSA із хеш-функціями SHA-512 та Whirlpool доводять свою надійність, але через великий об'єм вихідних даних ці методи не можуть бути прийняті у системах із обмеженою пам'яттю.

Алгоритм	Кількість символів		
	1й набір ключів	2й набір ключів	3й набір ключів
RSA	239	179	167
RSA+MD5	95	68	71
RSA+SHA-1	119	89	80
RSA+SHA-512	383	287	263
RSA+Whirlpool	383	284	260

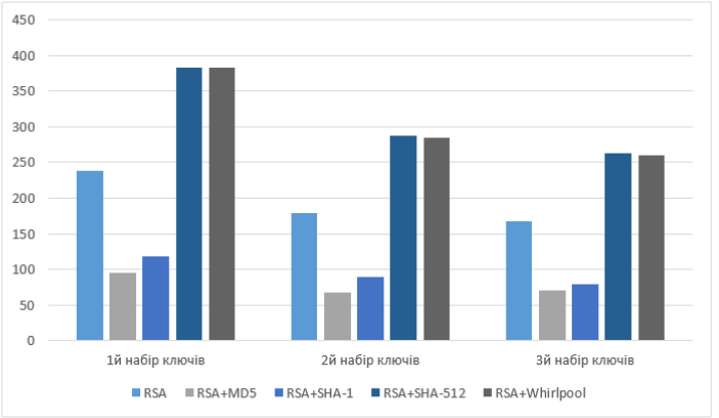


Рисунок В.17 – Слайд №17

Реалізація атаки на алгоритм RSA за методом Ферма

Атака була реалізована на прикладі першого набору ключів - *p,q=251,31 ; e,d,n=83,4247,7781*

За 53 кроки було знайдене початкове n, і отримані значення p та q, що дорівнюють відповідно 251, 31 .

А надалі щоб підібрати відповідні ключі шифрування вже не виникає складнощів.

Якщо не зважати на наявність даних у атакуючого компонентів для шифрування ,тоді криптостійкість методів шифрування є прийнятною і швидкість злому шифру залежить лише від самого методу шифрування та обладнання, з якого ведеться «злом» шифру.

Рисунок В.18 – Слайд №18

Висновки

Була розкрита актуальність проблеми забезпечення ефективної, надійної та безпечної передачі даних та збереження їх цілісності і пошук надійного захисту від деформування сторонніми особами.

Були описані заходи по вдосконаленню методів шифрування. Проведено аналіз криптостійкості алгоритму RSA та методів хешування за допомогою програмного засобу CrypTool 2 на базі якого було здійснена симуляція роботи алгоритмів та проведений аналіз їх взаємодії із ключам шифрування, за результатам якого було отримано залежність, щодо значного приросту криптостійкості при збільшенні розміру криптографічних ключів.

Проведений ряд атак на алгоритм MD5 за допомогою програми PasswordsPro 3.1, за результатами якого було доведено значний зріст рівня криптостійкості за допомогою впровадження до системи шифрування хеш-функції, котрі зберігають цілісність первинних даних та виступають як інструмент для аутентифікації при обміні даних у системі клієнт-сервер.

Рисунок В.19 – Слайд №19

Висновки

Було надано інструменти та приклади реалізації систем шифрування RSA та хешування на прикладі MD5 алгоритму, за допомогою мови програмування C#. Проведення атаки методом факторизації Ферма показало, що при наявності у третьої сторони (атакуючого) даних для шифрування та шифрованого повідомлення, можливий повний злом системи шифрування, що показує залежність криптостійкості системи також від наявності закритого каналу зв'язку між абонентами.

Практичне значення отриманих результатів полягає в тому, що основні наукові положення дисертації реалізовані у виді розрахункових моделей та програмних засобів, які утворюють прикладну інформаційну технологію для проведення аналізу криптографічних методів та виявлення їх недоліків. У свою чергу інші методи для перевірки існують окремо і надають результати тільки про певні аспекти алгоритмів шифрування і не достатньо чітко показують їх практичне значення криптостійкості і зазвичай стійкість методу при зломі або проведенні атак описано лише відносними значеннями.

Рисунок В.20 – Слайд №20