МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ
КАФЕДРА ІНФОРМАЦІЙНОЇ БЕЗПЕКИ ТА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

Методичні вказівки

до виконання лабораторних робіт

з дисципліни

«ПРОЕКТУВАННЯ СИСТЕМ КОМПЛЕКСНОГО ЗАХИСТУ ІНФОРМАЦІЇ»

для студентів напряму
6.050102 «Комп'ютерна інженерія»
всіх форм навчання

Затверджено на засіданні кафедри
інформаційної безпеки та
комп'ютерної інженерії,
протокол №2 від 25.09.2014 р.,
та Методичною радою ЧДТУ,
протокол № 1/ 2014-2015
від 10.11.2014 р.

Черкаси 2015

Укладач Куницька Світлана Юріївна, к.т.н., доцент

Рецензент Захарова М.В., к.т.н. доцент

Методичні вказівки з дисципліни «Проектування систем комплексного захисту інформації» для студентів напряму 6.050102 «Комп'ютерна інженерія» всіх форм навчання / Уклад.: С.Ю. Куницька ; М-во освіти і науки, Черкас. держ. технол. ун-т. – Черкаси : ЧДТУ, 2015. – 48с.

ЗМІСТ

Вступ

Методичні вказівки призначені для освоєння студентами напряму 6.050102 – комп'ютерна інженерія, основ криптографічного захисту інформації та різноманітних алгоритмів шифрування. Розглядаються питання захисту операційної системи.

Основною метою викладання даної дисципліни є ознайомлення з методами простого шифрування та електронного цифрового підпису, а також навчитися створювати та аналізувати шифротексти за допомогою написаного студентом програмного забезпечення.

Навчання студентів ведеться в формі лекцій, лабораторних, практичних та самостійних занять з використанням сучасних персональних комп'ютерів. Самостійні заняття зводяться до написання рефератів та підготовки доповідей.

Мета проведення лабораторних та практичних занять: одержання початкових навичок при використанні різноманітних методів та отримання результатів кодування або розкодування за допомогою написання різними мовами об'єктно-оріентованого програмування процесу шифрування тексту для захисту інформації.

В результаті проведення лабораторних та практичних занять студент повинен:

- знати лекційний матеріал з детальним описанням методики того чи іншого способу кодування інформації;
- вміти правильно використати можливості об'єктно-оріентованого програмування за допомогою належної мови;
- правильно та зрозуміло описати алгоритм шифрування;
- вміти показати процес розкодування інформації;
- показати отримані програмні результати.

1. Вимоги до виконання, оформлення та захисту лабораторних робіт

Лабораторні роботи, що представлені в методичних вказівках, призначені для набуття практичних навичок при використанні методів та отримання результатів кодування або розкодування за допомогою написаних різними мовами програмування процесу шифрування тексту для захисту інформації.

При виконанні кожної лабораторної роботи студент опрацьовує розділ з теоретичними відомостями, аналізує поставлену задачу, виконує її методами, вказаними в роботі. Мову програмування вказує викладач на свій розсуд.

Завдання полягає в написанні лістингу програми, де буде викладено методику шифрування тексту згідно лабораторної роботи.

Після виконання кожної лабораторної роботи студент оформлює звіт згідно ДСТУ 3008-95 *(«Документація. Звіти у сфері науки і техніки. Структура і правила оформлення)* на аркушах формату А4 у запропонованій нижче формі і захищає лабораторну роботу у викладача при наявності електронної версії лістингу програми.

Кожна лабораторна робота повинна мати:

- *титульний лист* з порядковим номером роботи та відображеною темою цієї роботи;
- *зміст (лист 2)* з наступними розділами та відповідними сторінками: вступ, теоретичні відомості, лістинг програми, результати виконання роботи або хід виконання та висновки;
- *вступ (лист 3)* включає в себе: назву теми, мету та завдання;
- *теоретичні відомості (лист 4)*обсягом описання 2-3 сторінки;
- *лістинг програми до лабораторної роботи (лист N);*
- *результати виконання або хід роботи (лист N);*
- *висновки (лист N)*.

В розділі «*Теоретичні відомості»* студент детально описує процес написання лістингу програми - всі процедури, модулі або макроси, які виконують поставлене завдання. Робить речове розширене описання лістингу

програми та обґрунтовує метод шифрування. Аналізує вибраний ключ для шифрування тексту, описує, процес розшифрування тексту та обґрунтовує отримані результати. Коротко описує методику шифрування.

Лістинг програми друкується 14 шрифтом на одній стороні листа в одну колонку.

В розділі *«Результати виконання роботи»* студент повинен представити всі допоміжні робочі вікна у друкованому вигляді, які отримані на кожному з етапів кодування або розкодування.

У *«Висновках»* повинно бути описано та обґрунтовано хід виконання лабораторної роботи обсягом на 1 сторінку.

По закінченні курсу лабораторних робіт студент представляє звіти оформлені належним чином по кожній з них, зібравши їх у єдину папку із загальним титульним аркушем *«Журнал звітів з лабораторних робіт по дисципліні ПСКЗІ»*.

2. Лабораторна робота №1

Тема: Розробка програмного забезпечення для визначення парольної ідентифікації**.**

Мета: Навчитися створювати та аналізувати табличні дані, обмежувати доступ до структур за допомогою аутентифікації.

2.1. Завдання

1. Створити 7 структур таблиць даних, які містять в собі мінімум 5 записів.

2. Визначити мінімум 5 учасників, 2 з яких мають доступ до 2-х структурних підрозділів, а інші тільки до одного.

3. Перевірка доступу учасника, за допомогою введення трьохкратного паролю.

4. Вивести результати роботи та створити дружній інтерфейс.

Мова програмування обирається на розсуд викладача.

2.2. Короткі теоретичні відомості

Основні етапи допуску в комп'ютерну систему:

1. ідентифікація;

2. встановлення автентичності (аутентифікація);

3. визначення повноважень для подальшого контролю і розмежування доступу до комп'ютерних ресурсів (авторизація).

Ідентифікація необхідна для вказівки комп'ютерній системі унікального ідентифікатора користувача, що звертається до неї, з метою виконання наступних захисних функцій:

- встановлення автентичності та визначення повноважень користувача та його допуск в комп'ютерну систему;

- контроль встановлених повноважень і реєстрація заданих дій користувача в процесі його сеансу, роботи після допуску даного користувача в КС;
- облік звернень до комп'ютерної системи.

Ідентифікація – привласнення суб’єктам або об’єктам доступу ідентифікатора або порівняння пред’явленого ідентифікатора з переліком привласнених ідентифікаторів.

Ідентифікація об’єкта - це його впізнання, ототожнення із чим-небудь. Якщо ж говорити про області інформаційних технологій, то даний термін звичайно означає встановлення особистості користувача. Цей процес необхідний для того, щоб система надалі змогла ухвалити рішення щодо видачі людині дозволу для роботи на комп’ютері, доступу до закритої інформації тощо. Таким чином, ідентифікація є одним з основних понять в інформаційній безпеці.

Парольна ідентифікація найбільш проста як у реалізації, так й у використанні. Суть її - кожен зареєстрований користувач системи одержує набір персональних реквізитів (звичайно використаються пари: логін-пароль). Далі при кожній спробі входу людина повинна вказати свою інформацію. Оскільки вона унікальна для кожного користувача, то на підставі її система й робить висновок про особистість та ідентифікує.

Недоліком парольної ідентифікації є значна залежність надійності ідентифікації від користувачів, точніше від обраних ними паролів. Фахівці в області інформаційної безпеки радять використати довгі паролі, що складаються з безладного сполучення букв, цифр і різних символів.

Сам ідентифікатор може являти собою послідовність будь-яких символів, і повинен бути заздалегідь зареєстрований в системі адміністратором служби безпеки. У процесі реєстрації адміністратором в базу еталонних даних системи захисту для кожного користувача заносяться такі елементи даних:

- прізвище, ім'я, по-батькові та, при необхідності, інші характеристики користувача;
- унікальний ідентифікатор користувача;
- ім'я процедури встановлення автентичності;
- використовувана для підтвердження автентичності еталонна інформація, наприклад, пароль;
- обмеження на використовувану еталонну інформацію, наприклад, мінімальний і максимальний час, протягом якого зазначений пароль буде вважатися дійсним;
- повноваження користувача з доступу до комп'ютерних ресурсів.

Парольні методи перевірки автентичності користувачів при вході в КС можна розділити на дві групи:

- методи перевірки автентичності на основі простого пароля;
- методи перевірки автентичності на основі динамічно мінливого пароля.

Процедура впізнання з використанням простого пароля може бути представлена у вигляді такої послідовності дій:

1. користувач надсилає запит на доступ до комп'ютерної системи і вводить свій ідентифікатор;
2. система запитує пароль;
3. користувач вводить пароль;
4. система порівнює отриманий пароль з паролем користувача, що зберігаються в базі еталонних даних системи захисту, і дозволяє доступ, якщо паролі збігаються; в іншому випадку користувач до ресурсів комп'ютерної системи не допускається.

Оскільки користувач може допустити помилку при введенні пароля, то системою має бути передбачено допустима кількість повторень для введення пароля.

Можна виділити наступні основні способи підвищення стійкості системи захисту на етапі аутентифікації:

- підвищення ступеня не тривіальності пароля;
- збільшення довжини послідовності символів пароля;
- збільшення часу затримки між дозволеними спробами повторного введення неправильно введеного пароля;
- підвищення обмежень на мінімальний і максимальний час дійсності пароля.

Імовірність підбору пароля зменшується також при збільшенні його довжини і часу затримки між дозволеними спробами повторного введення неправильно введеного пароля. Очікуваний час розкриття пароля T_r можна обчислити на основі наступної отриманої експериментально наближеною формули:

$$T_r \approx (As * T_v) / 2.$$

A - число символів в алфавіті, використовуваному для набору символів пароля;

S - довжина пароля в символах, включаючи пробіли та інші службові символи;

T_v - час введення пароля з урахуванням часу затримки між дозволеними спробами повторного введення неправильно введеного пароля.

Існують наступні методи парольного захисту, засновані на використанні динамічно мінливого пароля.

Методи модифікації схеми простих паролів

До методів модифікації схеми простих паролів відносять випадкову вибірку символів пароля і одноразове використання паролів.

При використанні першого методу кожному користувачеві виділяється досить довгий пароль, причому кожного разу для впізнання використовується не весь пароль, а тільки його деяка частина. У процесі перевірки автентичності система запитує у користувача групу символів по заданим порядковим номерам. Кількість символів і їх порядкові номери для запиту визначаються за допомогою датчика псевдовипадкових чисел.

При одноразовому використанні паролів кожному користувачеві виділяється список паролів. У процесі запиту номер пароля, який необхідно ввести, вибирається послідовно за списком або за схемою випадкової вибірки.

Недоліком методів модифікації схеми простих паролів є необхідність запам'ятовування користувачами довгих паролів або їх списків.

Функціональні методи

Серед функціональних методів найбільш поширеними є метод функціонального перетворення пароля, а також метод «Рукостискання».

Метод функціонального перетворення заснований на використанні деякої функції F, яка повинна задовольняти наступним вимогам:

- для заданого числа або слова X легко обчислити Y = F (X);
- знаючи X і Y складно або неможливо визначити функцію Y = F (X).

Користувачеві повідомляється:

- початковий пароль - слово або число X, наприклад число 31;
- функція F (X), наприклад, Y = (X mod 100) * D + W3, де (X mod 100) - операція взяття залишку від цілочисельного ділення X на 100, D - поточний номер дня тижня, а W - поточний номер тижня в поточному місяці;
- періодичність зміни пароля, наприклад, кожен день, кожні три дня або щотижня.

Значення криптографії виходить далеко за рамки забезпечення секретності даних. По мірі все більшої автоматизації передачі та обробки інформації та інтенсифікації інформаційних потоків її методи набувають унікальне значення. Відзначимо деякі сучасні напрями застосування криптографії:

1. Захист від несанкціонованого читання, або забезпечення конфіденційності інформації.
2. Захист від нав'язування помилкових повідомлень, як умисних, так і ненавмисних.
3. Ідентифікація законних користувачів .

4. Контроль цілісності інформації.
5. Аутентифікація інформації.
6. Електронний цифровий підпис.
7. Системи таємного електронного голосування .
8. Електронне жеребкування .
9. Захист документів і цінних паперів від підробки.

Захист від нав'язування помилкових повідомлень може бути забезпечений за допомогою так званого імітозахисту. Імітозахист - захист від нав'язування помилкових повідомлень шляхом формування , залежно від секретного ключа , спеціальної додаткової інформації , так званої імітовставки, яка передається разом з криптограмою, причому можуть використовуватися два варіанти: обчислення імітовставки або з відкритого тексту, або по шифртексту. Чим більше довжина імітовставки, тим менше ймовірність того, що спотворення шифртекста не буде виявлено одержувачем. Найпростішими прикладами імітовставок можуть служити будь-які надлишкові коди: код з перевіркою на парність, коди Хеммінга, циклічні коди і т. п.

Ідентифікація законних користувачів полягає в розпізнаванні користувачів, після чого їм надаються певні права доступу до ресурсів.

Контроль цілісності інформації - це виявлення будь-яких несанкціонованих змін інформації, наприклад, даних або програм, що зберігаються в комп'ютері. Імітозахист, по суті, є окремим випадком контролю цілісності інформації, що передається у вигляді шифртекста. У практичних додатках часто потрібно упевнитися, що деякі дані або програми не були змінені будь-яким несанкціонованим способом, хоча самі дані не є секретними і зберігаються у відкритому вигляді. Контроль цілісності інформації заснований на використанні кодів для виявлення та виправлення помилок, наприклад таких, як уже згадувані вище коди з перевіркою на парність, коди Хеммінга циклічні коди і т. п.

Аутентифікація інформації - встановлення санкціонованим одержувачем того факту, що отримане повідомлення послано санкціонованим відправником. Дотримання заздалегідь обумовленого протоколу - набору правил і процедур -

має забезпечити максимальну ймовірність цього факту. Очевидно, що при цьому контролюється і цілісність повідомлення на можливість підміни або спотворення. Прийнятий протокол повинен забезпечити протидію використанню потенційним порушником раніше переданих повідомлень. Цей напрям сучасної криптології дуже інтенсивно розвивається з моменту відкриття криптографії з відкритим ключем (асиметричної або двоключової криптографії) у середині 1970 -х років. Якщо робота Шеннона "Теорія зв'язку в секретних системах" 1949 р. заклала фундамент формування криптології як науки, то відкриття двоключової криптографіі ознаменувало собою її перехід в якісно нову фазу бурхливого розвитку і послужило основою для найбільш повного вирішення проблем аутентифікації інформації і розробки систем цифрового (електронного) підпису, які повинні надати юридичну силу документам та іншими повідомленнями, переданих електронним способом.

Електронний підпис грунтується на двоключових криптографічних алгоритмах, в яких передбачається використання двох ключів - відкритого і секретного. Ідея використання відкритого, тобто відомого всім користувачам криптосистеми і потенційному зловмиснику, ключа є фундаментальною, тому двоключові криптосистеми ще називають відкритими шифрами. Двоключові криптоалгоритми дозволяють забезпечити сувору доказовість факту складання того чи іншого повідомлення конкретними користувачами криптосистеми. Це засновано на тому, що тільки відправник повідомлення, який тримає в секреті деяку додаткову інформацію - секретний ключ - може скласти повідомлення зі специфічною внутрішньою структурою. Те, що повідомлення має структуру, сформовану за допомогою секретного ключа, перевіряється за допомогою відкритого ключа - процедура перевірки електронного цифрового підпису. Імовірність того, що деяке повідомлення, складене порушником, може бути прийняте за повідомлення, підписане несанкціонованим користувачем, оцінюється в близько 10-30%.

Відкритий ключ формується на основі секретного ключа, або обидва вони виробляються одночасно за спеціальними процедурами, причому визначення секретного ключа з відкритого є обчислювально-складним завданням, тобто

завдання, завідомо має рішення, але вимагає для його знаходження виконання надзвичайно великого числа операцій обчислювача: витрати часу на обчислення з залученням найсучасніших засобів можуть досягати десятиліть або століть.

Системи таємного електронного голосування будуються на базі двоключових алгоритмів, які використовують механізм сліпого підпису, тобто можливості підписати повідомлення без ознайомлення з його змістом. Такі системи мають великі перспективи для вдосконалення системи політичного управління сучасного суспільства з розвиненою інформаційною інфраструктурою.

Електронне жеребкування зводиться, наприклад, до реалізації нижче наведеного алгоритму .

1. Абонент *A* вибирає випадкове число x_a, двійкове представлення якого має, наприклад , 80 розрядів , обчислює значення деякої односторонньої функції $y_a = F(x_a)$ і повідомляє величину я абоненту *B*. Абонент *B* повинен вгадати, чи є число x_a парне або непарне.

2. Оскільки використовувана функція (відома і *B*) є односторонньою, то *B* не може за значенням a визначити x_a, тому він повинен вгадувати парність. Нехай абонент *B* стверджує , що x_a є парним і повідомляє про це абоненту *A*.

3. Абонент *A* повідомляє абоненту *B* число x_a .

4. Абонент *B* обчислює значення $y = F(x_a)$; якщо $y=a$, то *B* переконується , що його партнер дійсно надав для перевірки спочатку вибране число.

Очевидно, варіантів електронного жеребкування може бути запропоновано безліч, а практичне використання його застосовано до будь-яких спортивних жеребкувань, розіграшів, лотерей і т. п.

Криптографічний захист документів і цінних паперів від підробки є найбільш надійним сучасним способом припинення їх фальсифікації. Криптографічний захист від підробки може проводитися наступним чином: зчитується інформація про унікальні особливості даного носія інформації, формується цифровий паспорт, що включає зміст документа і інформацію про

мікроструктуру документа. Потім законний виробник документа, використовуючи свій секретний ключ, обчислює цифровий підпис паспорта і записує на носії паспорт і відповідний йому цифровий підпис.

Перевірка справжності документа виконується шляхом сканування мікроструктури матеріального об'єкта, на якому сформований документ, зчитування записаної на ньому інформації та перевірки цифрового підпису виготовлювача документа по відкритому ключу, який є загальнодоступним. Виготовлення фальшивого документа на іншому матеріальному об'єкті або модифікування змісту документа і його цифрового паспорта практично не здійсненні без знання секретного ключа, за допомогою якого формується підпис. Будь-яка підробка буде виявлена шляхом зчитування цифрового паспорта та цифрового підпису, суміщення паспорта із змістом документа та перевірка підпису з відкритого ключа.

2.3. Приклад до виконання лабораторної роботи №1

Теоретичні відомості

Після того, як користувач запускає розроблене програмне забезпечення, виконується запуск події завантаження форми. Дана подія потрібна зазвичай для того, щоб задати стандартні значення для змінних або елементів. В даному випадку виконується встановлення значення за замовченням для елемента comboBox1. Даний елемент дозволяє користувачу вибирати рівень доступу облікового запису, що був створений. Якщо не встановити значення за замовченням, то поле буде пусте та може створювати труднощі при створенні облікового запису (потрібно додавати додаткову перевірку для даного поля). Значення за замовченням для елемента – це 1 рівень доступу, але підрахунок елементів поля починається з 0, тому потрібно встановити значення 0.

Коли відбулося завантаження форми, програма більше не виконує ніяких дій та очікує натиснення кнопок. Для кожної кнопки створена подія.

Кнопка «Додати користувача» описана подією «button1_Click». Після натиснення кнопки програма виконує перевірку всіх існуючих видимих полів

форми. Якщо хоча б одне з обов'язкових полів пусте – користувач отримує повідомлення про помилку та подія завершає свою роботу. Щоб викликати повторно подію – потрібно повторно натиснути на кнопку. Якщо користувач ввів всі потрібні дані – програма починає процедуру реєстрації облікового запису в створеній структурній БД програми. В масив bd_user, який в даному випадку і є базою даних, що містить в собі користувачів, вноситься нове поле під порядковим номером bd_user_count. Дана змінна – це і лічильник кількості користувачів, і порядковий номер користувача в БД. Створивши нове поле, викликається метод класу, який має назву «addnew» та виконує функцію створення нового користувача. Даний метод приймає масив даних, в якому міститься вся інформація про користувача. Також остання змінна, яка входить до даного методу – це рівень доступу. Так як раніше вже описано, що елемент «comboBox1» містить поля, починаючи з 0, а потрібно починати відлік від 1, до індексу елементу потрібно додати одиницю. Створивши користувача, програма збільшує індекс кількості користувачів на 1 та оновлює інформацію форми про кількість користувачів (рис.1).

Після створення облікового запису, користувач проходить процедуру авторизації. Це потрібно для того, щоб надати обліковому запису користувача потрібний доступ до таблиць. Авторизація проводиться після натиснення кнопки «Авторизація» (виконується виклик події «button2_Click»). Спочатку оновлюється індекс користувача. За замовченням він дорівнює -1 (0 не може бути, так як ідентифікатор користувачів в БД починається з 0). Далі виконується стандартна перевірка заповнення полів. Якщо всі потрібні поля не заповнені – виконується зупинення події та виведення повідомлення про помилку. В іншому випадку виконується цикл по створеній БД, який шукає користувача, у якого співпадає логін та пароль. Перевірка користувача відбувається за допомогою методу «checkuser», який приймає вхідний параметр у вигляді масиву. Масив містить у собі дані користувача, які були введені в програму. Якщо програмі вдається знайти користувача, в змінну «userid» (індекс користувача) записується id користувача. Якщо користувача не знайдено – програма додає до кількості невдалих спроб авторизації одиницю та

проводить перевірку: якщо користувач зробив більше, ніж 2 помилки – викликаємо процедуру «resetauth», яка виконує блокування панелі авторизації на 1 хвилину (рис.2).

Надалі, якщо користувач знайдений – на екрані користувача з'являються поля заповнення, які потрібні для перевірки особи користувача – це номер телефону та дата народження (рис.3). Виконується аналогічний пошук аккаунту за даними параметрами. Якщо користувач знайдений – програма перевіряє його рівень доступу та в залежності від рівня, надає доступ до певних таблиць (наприклад, якщо рівень доступу 3 – користувач побачить тільки 6 та 5 таблицю, якщо рівень 2 – 6,5,4 та 3 і т. д.). Якщо ж користувач ввів хибні дані про свою особу – дана спроба зараховується як помилка введення та у разі 2-х невдалих спроб, виконується блокування панелі авторизації.

Опишемо процедуру блокування панелі авторизації, яка має назву «resetauth». Після виклику виконується виведення інформації про те, що авторизація заблокована на 1 хвилину. Виконується програмне змалювання області авторизації та нульове всіх введених раніше даних. Особливістю даної процедури є те, що виконується виклик таймеру, який працює 1 хвилину. Після того, як пройшла хвилина, програма виконує повторне виведення блоку авторизації.

Коротко опишемо клас «user», який в даній програмі є базою даних, яка зберігає інформацію про користувача. Даний клас має такі поля: ім'я користувача, пароль, дата народження, телефон та рівень доступу. Всі поля, крім останнього мають рядковий тип. Останнє поле числове (рис.4). Також клас має такі методи: метод перевірки логіну та паролю (який перевіряє поле ім'я користувача та пароль, якщо поля відповідають введеним даним – виводиться значення true, в іншому випадку – fasle), метод перевірки телефону та дати народження (який працює аналогічно попередньому методу), метод отримання інформації про аккаунт (виконується виведення з класу всіх методів) та метод внесення даних про користувача.

Так як всі дані вносяться в програму безпосередньо під час роботи програми (в т.ч. й користувачі), інформація в структурні таблиці також

заноситься під час роботи програми. Після успішного заповнення всіх вище описаних полів отримуємо повну інформацію про користувача (рис.4).

Після вдалої авторизації ми отримуємо доступ до бази даних, у відповідності до рівню доступу наданого користувачеві (рис.5).

Результати роботи

Створивши користувача, програма збільшує індекс кількості користувачів на 1 та оновлює інформацію форми про кількість користувачів. Інформується користувач про вдале створення аккаунту, виконується видалення всіх введених даних та розблокування панелі авторизації (у разі, якщо кількість користувачів у БД більше за 0):

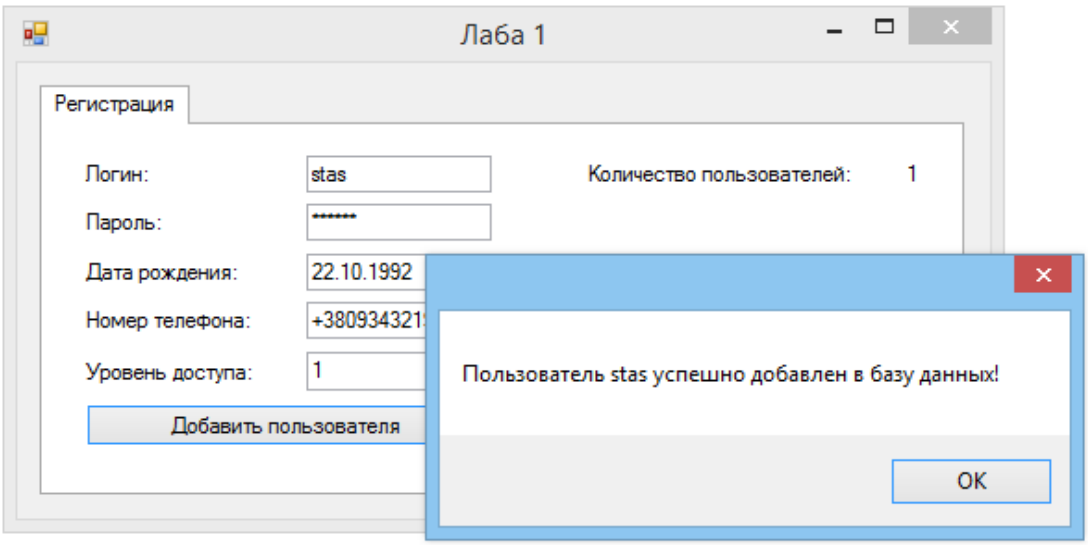

Рисунок 1 - Введення користувачів до БД

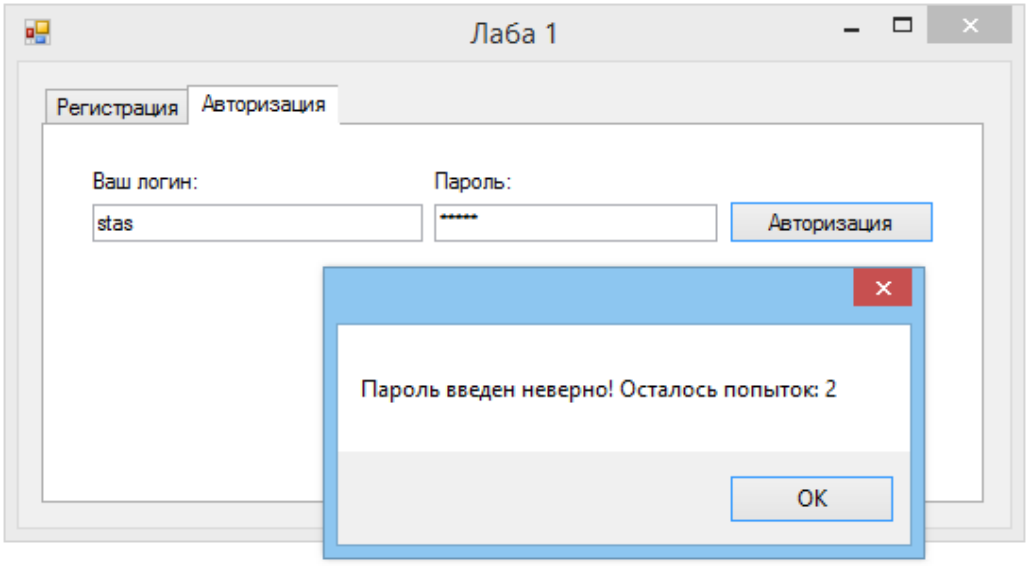

Рисунок 2 - Невдала спроба авторизації

Користувача не знайдено – програма додає до кількості невдалих спроб авторизації одиницю та проводить перевірку: якщо користувач зробив більше, ніж 2 помилки – викликаємо процедуру «resetauth», яка виконує блокування панелі авторизації на 1 хвилину (рис. 2).

На екрані користувача з’являються поля заповнення, які потрібні для перевірки особи користувача – це номер телефону та дата народження - для користувача, який знайдений:

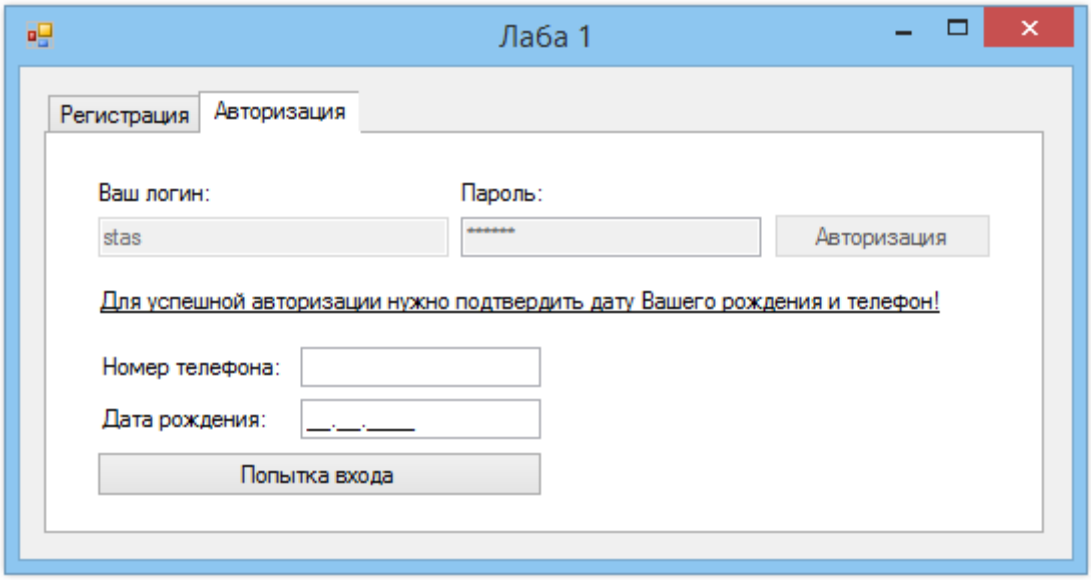

Рисунок 3 - Підтвердження власника за допомогою телефону та дати народження

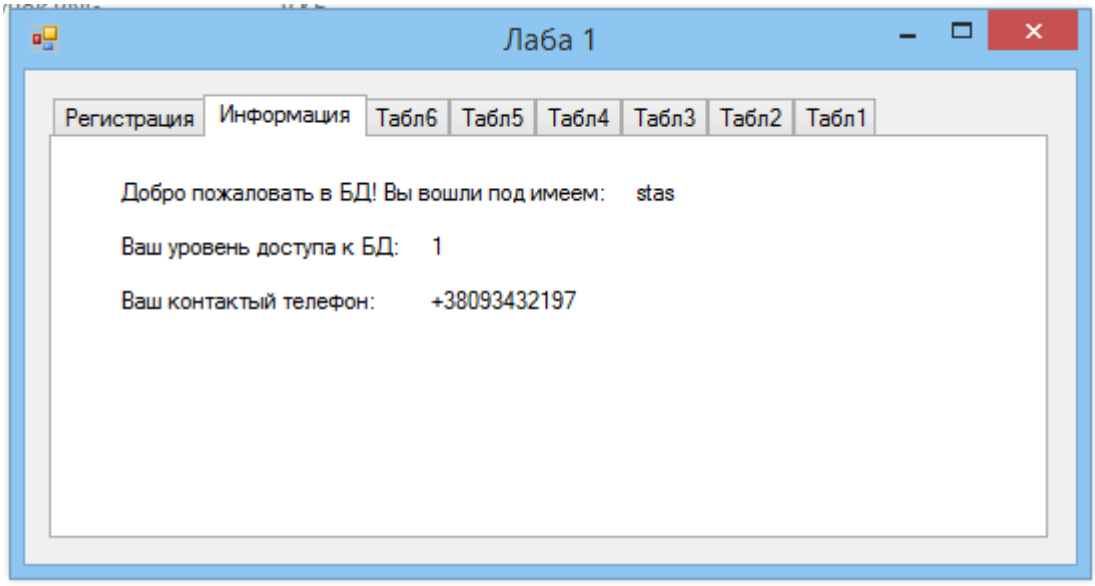

Рисунок 4 - Вдала авторизація та повний доступ до БД

Всі дані, в т.ч. й користувачі, та інформація по структурних таблицях вносяться в програму безпосередньо під час роботи програми. Після успішного заповнення всіх полів отримуємо повну інформацію про користувача (рис. 4).

Також після вдалої авторизації ми отримуємо доступ до бази даних, у відповідності до рівню доступу наданого користувачеві:

Рисунок 5 - Перегляд таблиць БД за допомогою створеного ПЗ

2.4. Контрольні питання

1. Які основні етапи допуску в комп'ютерну систему?
2. Скільки існує видів ідентифікації?
3. Сучасні напрями застосування криптографії?
4. Види ідентифікації?
5. Що таке імітозахист?
6. Що таке аутентифікація?
7. Які існують методи аутентифікації?
8. Які існують методи модифікації схеми простих паролів?
9. Види та призначення функціональних методів?

2.5. Список використаних джерел

1. Бондаренко М.Ф., Горбенко І.Д., Кравченко П.А., Мелецкий О.П. Аналіз та перспективи сучасних протоколів видання та генерації ключів для інфраструктури на базі ідентифікаторів. // Прикладная радиоэлектроника. – Харків: ХНУРЄ, 2007. - №3. – С.356.

2. Ричард Е. Смит. Аутентификация: от паролей до открытых ключей = Authentication: From Passwords to Public Keys First Edition . - М. : «Вильямс», 2002. - С. 432. - ISBN 0-201-61599-1.

3. Под редакцией А.А. Шелупанова, С.Л. Груздева, Ю.С. Нахаева. Аутентификация. Теория и практика обеспечения доступа к информационным ресурсам . = Authentication. Theory and practice of ensuring access to information resources . - М.: Гарячая линия - Телеком, 2009. - С. 552. - ISBN 978-5-9912-0110-0.

4. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, тексты на языке Ci = Applied Cryptography. Protocols, Algorithms and Source Code in C. - М.: Триумф , 2002 . - 816 с. - 3000 екз. - ISBN 5-89392-055-4.

5. Горбенко Ю.І., Тоцький О.С. Аналіз та вдосконалення криптографічних протоколів автентифікації та встановлення ключів між серверами ЛОМ. – ПР Рад. – Том8, №3. – С. 405-412.

6. Tardo J. and K. Alagappan SPX : Global Authentication Using Public Key Certificates. - M.California, 1991. - P. 232 - 244.

7. А.А. Гладких, В.Е. Дементьев. Базовые принципы информационной безопасности вычислительных сетей. - Ульяновск: УлГТУ, 2009. - С. 156.

8. ДСТУ 3008-95. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення.

3. Лабораторна робота №2

Тема: Захист інформації за допомогою **метода «Код Цезаря».**

Мета: Створити програмний лістинг, де користувач зможе виконувати шифрування відкритого тексту за допомогою методу «Код Цезаря».

3.1. Завдання

1. Написати процедуру для шифрування цитат видатного філософа за допомогою методу «Код Цезаря».
2. Створити дружній інтерфейс та отримані результати роботи вивести на екран.

Мова програмування обирається за згодою з викладачем.

3.2. Короткі теоретичні відомості

Шифр Цезаря - це вид шифру підстановки, в якому кожен символ у відкритому тексті замінюється буквою, що знаходиться на деяке постійне число позицій лівіше або правіше нього в алфавіті. Наприклад, у шифрі із зсувом 3, А була б замінена на Г, Б стане Д, і так далі.

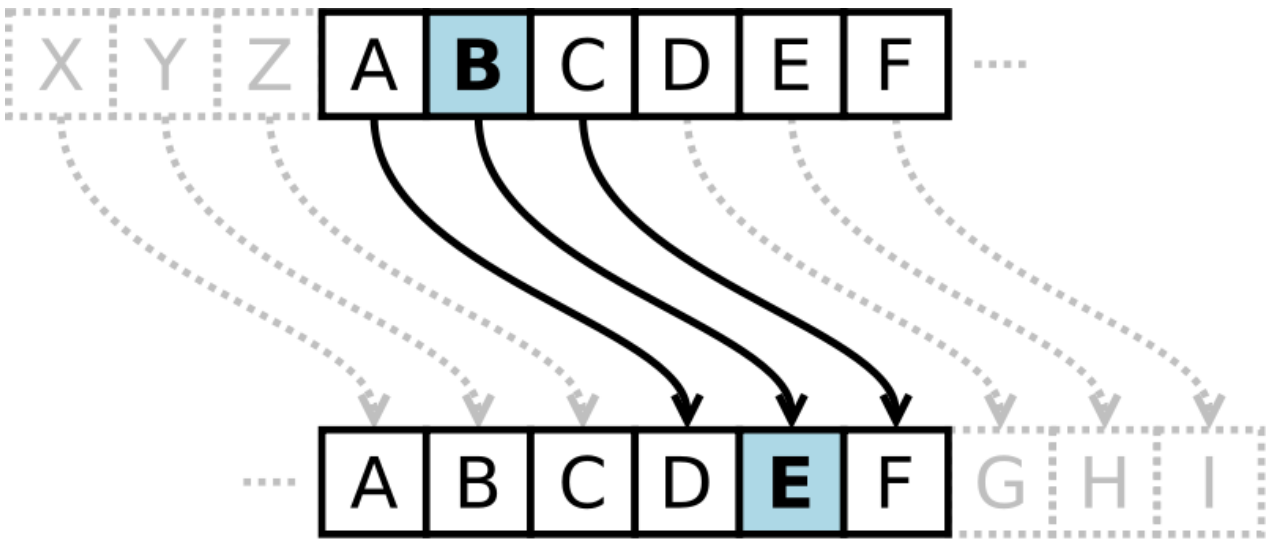

Рисунок 6 – Методика шифру із зсувом

Шифр названий на честь римського імператора Гая Юлія Цезаря, який використав його для секретного листування зі своїми генералами. Крок шифрування, що виконується шифром Цезаря, часто включається як частина більш складних схем, таких як шифр Віженера, і все ще має сучасний додаток в системі ROT13. Як і всі моноалфавітні шифри, шифр Цезаря легко зламуються і не має практично ніякого застосування на практиці.

Приклад 1. Припустимо, що використовуючи шифр Цезаря з ключем, який дорівнює 3, необхідно зашифрувати словосполучення *«ШИФР ЦЕЗАРЯ»*. Для цього потрібно змістити алфавіт так, щоб він починався з четвертої букви (Г). Отже, беручи вихідний алфавіт:

АБВГҐДЕЄЖЗИІЇЙКЛМНОПРСТУФХЦЧШЩЬЮЯ,

і зміщуючи всі літери вліво на 3, отримуємо відповідність:

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

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 А Б В

де Г=А, Ґ=Б, Д=В, і т. д.

Використовуючи цю схему, відкритий текст *«ШИФР ЦЕЗАРЯ»* перетворюється на *«ЮЙЧУ ЩЗЇГУВ»*. Для того, щоб одержувач повідомлення міг відновити вихідний текст, необхідно повідомити йому, що ключ — 3.

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

- зломщик знає (або припускає), що використовувався простий шифр підстановки, але не знає, що це - схема Цезаря ;
- зломщик знає, що використовувався шифр Цезаря, але не знає значення зсуву .

У першому випадку шифр може бути зламаний, використовуючи ті ж самі методи що і для простого шифру підстановки, такі як частотний аналіз і т.д. Використовуючи ці методи, зломщик, ймовірно, швидко помітить регулярність у вирішенні і зрозуміє, що використовуваний шифр - це шифр Цезаря.

У другому випадку, злом шифру є навіть простішим. Існує не так багато варіантів значень зсуву (26 для англійської мови), всі вони можуть бути перевірені методом грубої сили. Один із способів зробити це - виписати уривок зашифрованого тексту в стовпець всіх можливих зрушень - техніка, іноді звана як «завершення простого компоненту».

Приклад 2. Для зашифрованого тексту «EXXEGOEXSRGI» відкритий текст негайно розпізнається в четвертому рядку (табл.1).

Таблиця 1

Зсув	**Відкритий текст**
0	exxegoexsrgi
1	dwwdfndwrqfh
2	cvvcemcvqpeg
3	buubdlbupodf
4	attackatonce
5	zsszbjzsnmbd
6	yrryaiyrmlac
…	
23	haahjrhavujl
24	gzzgiqgzutik
25	fyyfhpfytshj

Інший спосіб застосування цього методу - це написати алфавіт під кожною буквою зашифрованого тексту, починаючи з цієї літери. Метод може бути прискорений, якщо використовувати заздалегідь підготовлені смужки з алфавітом. Для цього потрібно скласти смужки так, щоб в одному рядку утворився зашифрований текст, тоді в деякому іншому рядку ми побачимо відкритий текст.

Для звичайного тексту на природній мові, швидше за все, буде тільки один варіант декодування. Але, якщо використовувати дуже короткі повідомлення, то можливі випадки, коли можливі кілька варіантів

розшифровки з різними зрушеннями. Наприклад зашифрований текст *MPQY* може бути розшифрований як *«aden»* так і як *«know»* (припускаючи, що відкритий текст написаний англійською мовою). Точно так само *«ALIIP»* можна розшифрувати як *«dolls »* або як *«wheel»*, *«AFCCP»* як *«jolly»* або як *«cheer»*.

Багаторазове шифрування ніяк не покращує стійкість, так як застосування шифрів із зсувом a і b еквівалентно застосуванню шифру із зсувом $a + b$. У математичних термінах шифрування з різними ключами утворює математичну групу.

3.3. Приклад до виконання лабораторної роботи №2

Теоретичні відомості

Розроблена програма виконує шифрування відкритого повідомлення (передбачається робота тільки з літерами). Розглянемо детальніше принцип роботи програми.

Програма має поле, яке потрібне для того, щоб користувач вводив ключ. Тому потрібно зробити дозволеним введення тільки чисел. Для цього потрібно для даного поля прикріпити подію «KeyPress», яка буде спрацьовувати тільки в тому випадку, якщо користувач натискає клавішу та фокус курсору знаходиться в даному полі. Перевірка на введення чисел досить проста. Для цього потрібно використати метод «IsDigit» класу «Char» та передати в даний метод код клавіші, яка в даний момент натиснута. Код клавіші завжди знаходиться в змінній «*e*», яка передається автоматично створеній події. Якщо метод повертає значення false, потрібно виконати додаткову перевірку, щоб забезпечити користувачу можливість видалення тексту. Тому порівнюється змінна «*e*» та стандартний код клавіші BackSpace. Якщо код не співпадає – це значить, що введена не цифра і не натиснута клавіша видалення тексту. Тобто програма не допускає дану клавішу до введення.

Також потрібно при завантаженні програми виконати встановлення елементу за замовченням в блоці «comboBox1». Даний блок використовується для визначення принципу роботи програми: виконувати шифрування чи

розшифрування тексту. Якщо значення за замовченням не встановлене – потрібно виконувати додаткові перевірки в програмі. В даному випадку, значення за замовченням – це режим шифрування.

Після того, як користувач ввів потрібний текст та вибрав режим роботи програми, він повинен натиснути кнопку «Виконати дію» яка викличе подію «button1_Click». В даній події виконується перевірка потрібних полів - поля введення тексту та поля ключа (рис.7).

Якщо поля заповнені, програма виконує отримання значення ключа. Причому виконується конвертування ключа з тексту в цифру, так як всі поля форми повертають тільки рядкове значення. Також після запуску події виконується обнулення поля результату. Далі виконується цикл, який розбирає кожну літеру введеного тексту. Виконується перевірка отриманого символу. Якщо це не буква – символ не кодується. Якщо буква, та вибрано режим шифрування – знову виконується перевірка. В даному випадку це перевірка на регістр: нижній чи верхній. Якщо код літери в верхньому регістрі плюс ключ дорівнює більше коду літери «Z» виконується додаткова дія – це визначення літери, яка повинна слідувати за літерою «Z». Так як алфавіт починається від «A» то програма починає шукати букву від початку алфавіту за допомогою функції «mod». Якщо отриманий код літери менше коду літери «Z» - результат кодування записується в поле виведення результату.

Також виконуються аналогічні операції при розшифруванні. Але в даному випадку потрібно не додавати ключ, а віднімати (рис.8).

Результати роботи

При завантаженні програми встановили елемент за замовченням в блоці «comboBox1», який використовується для визначення принципу роботи програми: виконувати шифрування чи розшифрування тексту.

Якщо значення за замовченням не встановлене – потрібно виконувати додаткові перевірки в програмі. В даному випадку, значення за замовченням – це режим шифрування.

Користувач ввів потрібний текст та вибрав режим роботи програми, а потім натиснув кнопку «Виконати дію» яка викличе подію «button1_Click». Виконується перевірка поля введення тексту та поля ключа:

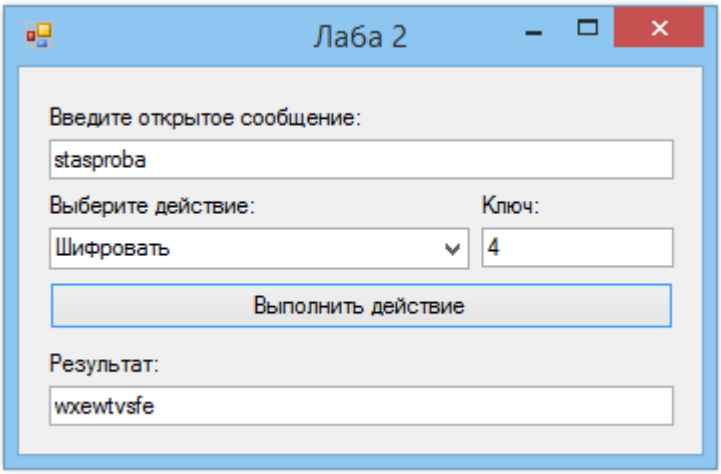

Рисунок 7 - Кодування інформації

При розшифруванні потрібно виконати дії аналогічні шифруванні, але не додавати ключ, а віднімати:

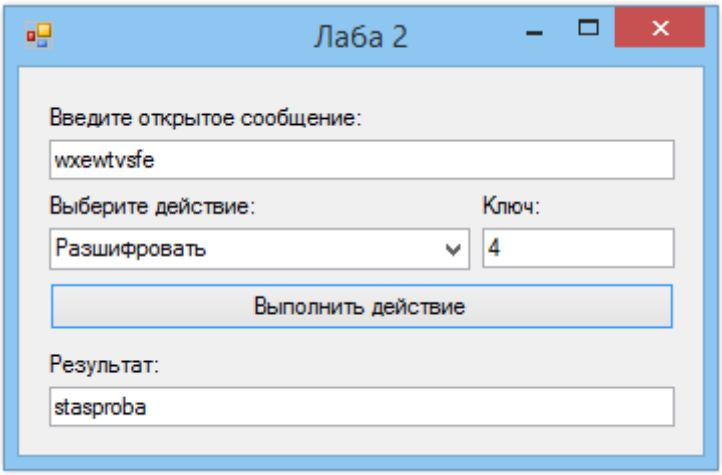

Рисунок 8 - Декодування інформації

3.4. Контрольні питання

1. В чому полягає методика шифру підстановки?
2. Що таке ключ, його використання?
3. Два способи застосування метода зсуву?
4. Як декодується текст?

3.5. Список використаних джерел

1. Бондаренко М.Ф. Горбенко И.Д., Потий А.В., Олешко О.И., Головашич С.А. Улучшенный стандарт симметричного шифрования 21 века: концепция создания свойства кондидатов. // Радиотехника. – 2000. – Вып. 114. – С. 5-14.
2. Аршинов М.Н. Садовский Л.Е. Коды и математика. - М.: Наука, 1983.
3. Березюк Н.Т., Андрущенко А.Г., Мощинський С.С. та ін. Кодування інформації (двійкові коди). - Харків: Вища школа, 1978.
4. Виноградов И.М. Основы теории чисел. - М.: Изд. Наука, 1982.
5. ДСТУ 4145-2002. Інформаційні технології. Криптографічний захист інформації. Цифровий підпис, що ґрунтується на еліптичних кривих. Формування та перевірка.
6. ДСТУ 3008-95. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення.
7. Диффи У., Хеллман М. Защищённость и имитостойкость. Введение в криптографию. - ТИИЭР. - 1979 . - Т.64, № 3. - С. 71-109.
8. Дориченко С.А., Ященко В.В. 25 этюдов про шифры. - М. : ТЕИС, 1994.
9. Бондаренко М.Ф., Горбенко Ю.І., Батюшко С.С. Аналіз існуючих ЕЦП від атак на звязаних ключах. // Прикладна радіоелектроніка. – 2005. – Том 5. №1. – С. 52-59.
10. Молдовян А.А., Молдовян Н.А., Советов Б.Я. Криптография. – СПб., 2000. – 218 с.
11. Саломан А. Криптография с открытым ключом. - М. : Мир, 1996.
12. Введение в криптографию / Под ред. В.В. Ященко. - М. : МЦНМО : "ЧеРо", 1999.

4. Лабораторна робота №3

Тема: Захист інформації за допомогою представника простого криптографічного шифрування - **таблиці Віженера**.

Мета: Програмно реалізувати захист інформації за допомогою таблиці Віженера.

4.1. Завдання

1. Написати процедуру для шифрування відомої цитати за допомогою таблиці Віженера.
2. Створити дружній інтерфейс та вивести результати роботи на екран.

Мова програмування обирається на розсуд викладача.

4.2. Короткі теоретичні відомості

Шифр Віженера (фр. Chiffre de Vigenère) - метод поліалфавітного шифрування буквеного тексту з використанням ключового слова.

У шифрі Цезаря кожна буква алфавіту зсувається на кілька рядків, наприклад, в шифрі Цезаря при зсуві +3 A стало б D, B стало б E і так далі. Шифр Віженера складається з послідовності декількох шифрів Цезаря з різними значеннями зсуву. Відносно латинського алфавіту таблиця Віженера складається з рядків по 26 символів, причому кожен наступний рядок зсувається на кілька позицій. Таким чином, в таблиці виходить 26 різних шифрів Цезаря. На різних етапах кодування шифр Віженера використовує різні алфавіти з цієї таблиці. На кожному етапі шифрування використовуються різні алфавіти, обрані в залежності від символу ключового слова.

Наприклад, припустимо, що вихідний текст має вигляд:

ATTACKATDAWN

Людина, надсилає повідомлення, записує ключове слово (*«LEMON»*) циклічно доти, поки його довжина не буде відповідати довжині вихідного тексту:

LEMONLEMONLE

Перший символ вихідного тексту A зашифрований послідовністю L, яка є першим символом ключа. Перший символ L шифрованого тексту знаходиться на перетині рядка L і стовпця A в таблиці Віженера. Так само для другого символу вихідного тексту використовується другий символ ключа, тобто другий символ шифрованого тексту X виходить на перетині рядка E і стовбцю T. Інша частина вихідного тексту шифрується подібним способом.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Рисунок 9 – Представлення таблиці Віженера

Оригінальний текст: **ATTACKATDAWN**

Ключ : **LEMONLEMONLE**

Зашифрований текст: **LXFOPVEFRNHR**

Розшифрування проводиться таким чином: знаходимо в таблиці Віженера рядок, відповідну першому символу ключового слова; в цьому рядку знаходимо перший символ зашифрованого тексту. Стовпець, в якому знаходиться даний символ, відповідає першому символу вихідного тексту. Наступні символи зашифрованого тексту розшифровуються подібним чином.

Якщо букви AZ відповідають числах 0-25, то шифрування Віженера можна записати у вигляді формули:

$$C_i \equiv (P_i + K_i) \mod 26 \quad (1)$$

тоді розкодування згідно формули:

$$P_i \equiv (C_i - K_i + 26) \mod 26 \quad (2)$$

Шифр Віженера «розмиває» характеристики частот появи символів у тексті, але деякі особливості появи символів у тексті залишаються. Головний недолік шифру Віженера полягає в тому, що його ключ повторюється. Тому простий криптоанализ шифру може бути побудований в два етапи:

- пошук довжини ключа. Можна аналізувати розподіл частот у зашифрованому тексті з різним проріджування. Тобто брати текст, що включає кожну 2 -у букву зашифрованого тексту, потім кожну 3-ю і т. д. Як тільки розподіл частот букв буде сильно відрізнятися від рівномірного (наприклад , по ентропії) , то можна говорити про знайдену довжині ключа.
- криптоаналіз. Сукупність л шифрів Цезаря (де л - знайдена довжина ключа), які окремо легко зламуються:
  - тести Фрідмана і Касіскі можуть допомогти визначити довжину ключа.
  - тест Касіскі спирається на те , що деякі слова, такі як «the » можуть бути зашифровані однаковими символами, що призводить до повторення груп символів в зашифрованому тексті.

Наприклад: повідомлення, зашифроване ключем ABCDEF, не завжди однаково зашифрує слово «crypto»:

Оригінальний текст: **CRYPTO**

Ключ : **ABCDEF AB CDEFA BCD EFABCDEFABCD**

Шифрований текст: **CSASXT IT UKSWT GQU GWYQVRKWAQJB**

Зашифрований текст в даному випадку не буде повторювати послідовності символів, які відповідають повторним послідовностям вихідного тексту. У даному зашифрованому тексті є кілька повторюваних сегментів, які дозволяють криптоаналітику знайти довжину ключа:

Оригінальний текст: **CRYPTO**

Ключ : **ABCDAB CD ABCDA BCD ABCDABCDABCD**

Шифрований текст: **CSASTP KB SIQUT GQU CSASTPIUAQJB**

Довші повідомлення роблять тест більш точним, так як вони включають в себе більше повторюваних сегментів зашифрованого тексту. У даному зашифрованому тексті є кілька повторюваних сегментів, які дозволяють криптоаналітику знайти довжину ключа:

Шифрований текст: **DYDUXRMHTVDVNQDQNWDYDUXRMHARTJ WNQD**

Відстань між повторюваними DYDUXRMH дорівнює 18, це дозволяє зробити висновок, що довжина ключа дорівнює одному зі значень: 18,9,6,3 або 2. Відстань між повторюваними NQD дорівнює 20. З цього виходить, що довжина ключа дорівнює 20 - або 10, або 5, або 4, або 2. Порівнюючи можливі довжини ключів, можна зробити висновок, що довжина ключа дорівнює 2.

4.3. Приклад до виконання лабораторної роботи №3

Теоретичні відомості

Перш ніж виконувати основні дії, програма повинна встановити для елементу comboBox1 поле за замовченням. Так як користувачу потрібно спочатку зашифрувати текст, потрібно встановити поле 0, яке одразу ж виконає вибір режиму шифрування даної програми. Всі ці дії виконуються при завантаженні форми та описуються в події «Form1_Load».

Розглянемо подію натискання кнопки «Виконати дії». Спочатку виконується очищення блоку результатів, створюється рядковий масив, який містить у собі алфавіт букв від «a» до «z», причому всі букви в нижньому регістрі. Наступним кроком потрібно отримати рядок та ключ в тимчасові змінні. Причому текст, який отримується, переводиться в нижній регістр (всі літери слів будуть маленькими). Виконується стандартна перевірка введення потрібних значень (ключа та слова). Якщо дані присутні – виконується циклічна генерація ключа – ключ повинен повторюватися стільки раз, скільки символів має текстове повідомлення. Після того, як отримано ключ, циклічно записуються всі літери вхідного слова в масив «ryadok_mas». Це потрібно для того, щоб було зручно працювати з текстом.

Перед виконанням шифрування потрібно перевірити символи слова та ключа на присутність їх в алфавіті. Якщо символ відсутній – він автоматично випадає з масиву символів. Далі отриманий ключ (розмір якого дорівнює розміру тексту) циклічно додається до символу тексту з урахуванням алфавіту (рис.10). Якщо в результаті складання отримується код символу, який більше 26 – потрібно вирахувати результат від ділення, таким чином отримавши код нового символу в алфавіті.

Щоб уникнути помилок при роботі з відкритим текстом та ключем, потрібно розробити події, які не дозволять вводити непотрібні символи до полів. Скористаємося типом події «KeyPress». Якщо користувач натискає цифри, або спеціальні символи, або управляючі клавіші (крім клавіші BackSpace), подія автоматично відміняє введення таких значень в потрібний блок.

Якщо режим роботи програми – це розшифрування, потрібно не додавати ключ, а віднімати його від символу відкритого тексту. Таким чином в блок виведення результату буде отримано потрібний результат (рис.11).

Результати роботи

Отриманий ключ (розмір якого дорівнює розміру тексту) циклічно додається до символу тексту з урахуванням алфавіту. Якщо в результаті

складання отримується код символу, який більше 26 – потрібно вирахувати результат від ділення, таким чином отримавши код нового символу в алфавіті:

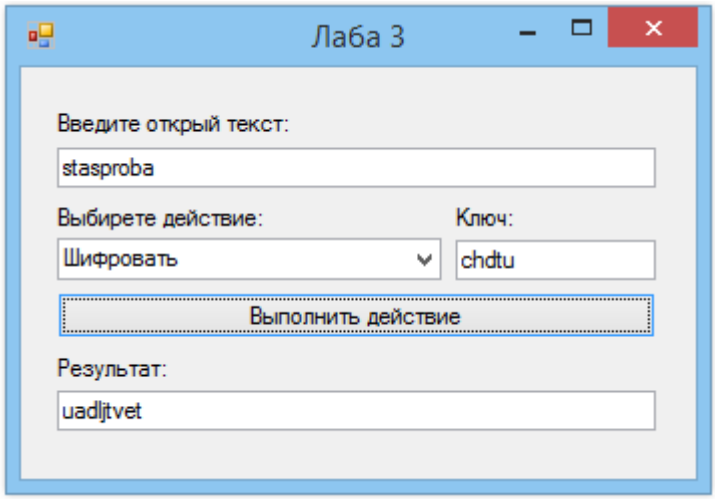

Рисунок 10 - Кодування інформації

Декодування відбувається за таким самим принципом:

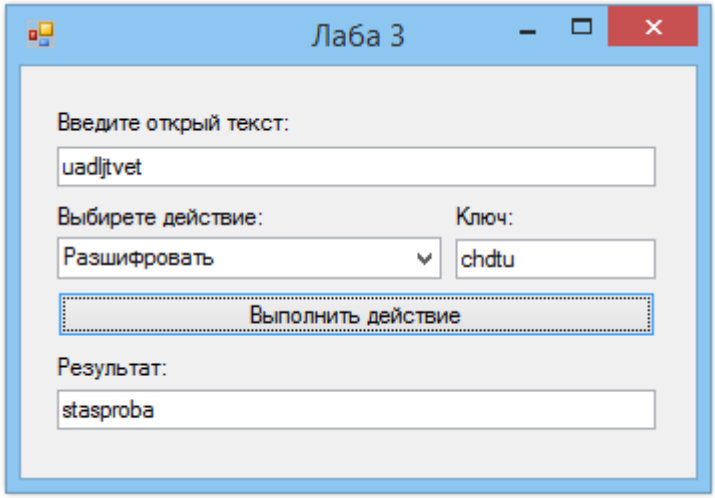

Рисунок 11 - Декодування інформації

4.4. Контрольні питання

1. В чому полягає метод поліалфавітного шифрування буквеного тексту?
2. Як використовується ключове слово?
3. Представлення таблиці Віженера?
4. Методика розшифрування тексту?

4.5. Список використаних джерел

1. Горбенко І.Д. Горбенко Ю.І. Прикладна криптологія. Теорія. Практика. Застосування: Монографія. – Харків: Видавництво «Форт», 2012. – 870 с.

2. Бондаренко М.Ф., Кравченко П.О. Комбінована інфраструктура відкритих ключів // Прикладная радиоэлектроника. – 2009. – Том 8.№3. – С. 327-330.

3. Борель Е. Вероятности, ошибки / Борель Е., Дейтель Р., Юрон Р. – 1972.

4. Березюк Н.Т., Андрущенко А.Г., Мощинський С.С. та ін. Кодування інформації (двійкові коди). - Харків : Вища школа, 1978.

5. Васильцов І.В. Атаки спеціального виду на крипто пристрої та методи боротьби з ними. / За науковою редакцією проф.. В.П. Широчина. – Кременець: Видавничий центр «КОГПІ», 2009. – 264 с.

6. ДСТУ 4145-2002. Інформаційні технології. Криптографічний захист інформації. Цифровий підпис, що ґрунтується на еліптичних кривих. Формування та перевірка.

7. ДСТУ 3008-95. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення.

8. Вембо Мао. Современная криптография. Теория и практика. / Пер. с англ. – М.: Изд. дом «Вильямс», 2005. – 768 с.

9. Василенко О.Н. Теоретико-числовые алгоритмы в криптографии. – Москва, 2003. – С. 325.

10. Петров А.А. Компьютерная безопасность. Криптографические методы защиты. – М.: изд. ДМК, 2000. – 445 с.

5. Лабораторна робота №4

Тема: Захист інформації за допомогою **методу «Решітка Кардано».**

Мета: Навчитися створювати програмне забезпечення, за допомогою якого користувач зможе виконувати шифрування відкритого тексту методом «Решітка Кардано».

5.1. Завдання

1. Написати процедуру для шифрування тексту за допомогою методу «Решітка Кардано».
2. Створити дружній інтерфейс та вивести результати роботи на екран.

Мова програмування обирається за згодою з викладачем.

5.2. Короткі теоретичні відомості

Решітка Кардано - інструмент кодування і декодування, що представляє собою спеціальну прямокутну (в окремому випадку - квадратну) таблицю-картку, частина осередків якої вирізана.

Щоб прочитати закодоване повідомлення, необхідно накласти решітку Кардано на текст потрібне число раз і прочитати літери, розташовані в вирізаних осередках.

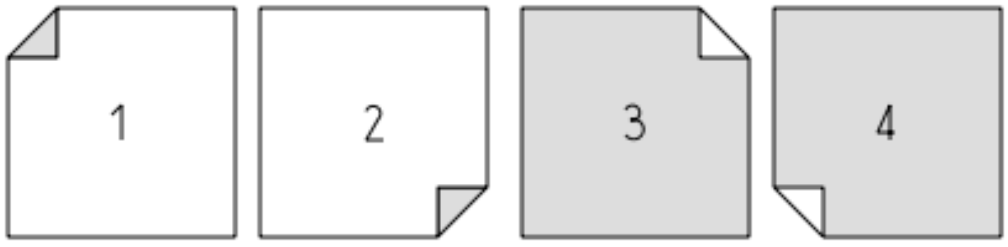

Рисунок 12 – Вигляд вирізаних осередків в решітці Кардано

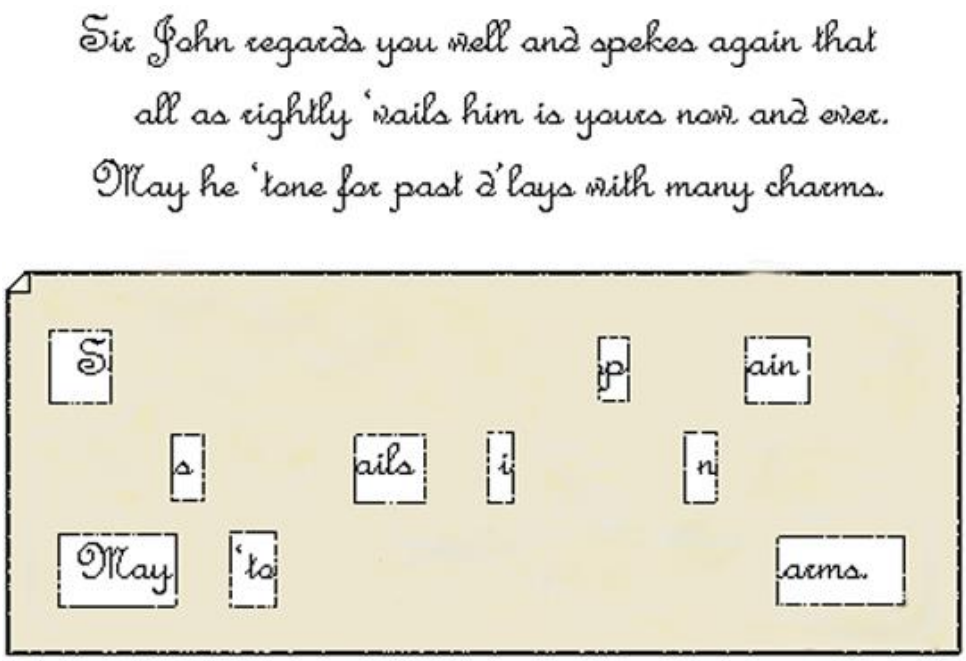

Рисунок 13 – Шифрування решіткою Кардано

Френсіс Бекон виділяв 3 особливості решітки Кардано, які можна сформулювати так:

1. Метод легкий в застосуванні.
2. Розшифрувати зашифрований текст для зловмисника - завдання практично нездійсненне.
3. Зашифрований текст не потрапляє під підозру.

Навіть якщо фахівець вважає повідомлення підозрілим, зашифрований текст може містити будь-яка буква. Тому, на практиці, єдине рішення - це отримати саму решітку.

Недоліками методу є те, що він повільний і вимагає наявності літературних навичок. Але найголовніше, що будь-який шифрувальний апарат може бути загублений, вкрадений або конфіскований. Таким чином, втратити одну решітку - значить втратити всю секретну переписку, шифровану за допомогою цієї решітки. Коли зашифроване повідомлення складено погано, воно виділяється неприродною мовою і постійно мінливим стилем. Фахівець може спробувати відновити решітку, якщо у нього є кілька примірників підозрілих повідомлень із листування. Коли повідомлення зашифровано добре, його важко виявити. Навіть якщо фахівець вважає повідомлення підозрілим,

зашифрований текст може містити будь безневинна буква. Тому, на практиці, єдине рішення - це отримати саму решітку.

5.3. Приклад до виконання лабораторної роботи №4

Теоретичні відомості

Шифрування методом решітки Кардано є класичною задачею роботи з квадратними матрицями. Для виконання шифрування потрібно розробити чотири матриці, які будуть відповідати таким положенням решітки: прямий обхід, 90 градусів, 180 градусів та 270 градусів. Причому потрібно розробити матриці так, щоб під час оберту всі поля квадрату були використанні (щоб запобігти втрати символів з тексту). В розроблених матрицях одиниця – це проріз в решітці, 0 – закрита ділянка. В даному випадку програма має матриці розмірністю 6 на 6. Тому можна зробити висновок, що вхідна цитата, яка буде кодуватися, повинна бути розміром в 36 символів. Також слід зазначити, що матриця, яку маємо і є ключем цього методу.

Розглянемо принцип кодування інформації. Кодування інформації в програмі виконує функція «getencrypt», яка приймає відкритий текст а повертає закритий. Спочатку створюємо тимчасовий масив, який буде зберігати в собі закодоване повідомлення та створюємо 4-и цикли, які по черзі будуть виконувати оберти решітки. Розглянемо приклад роботи решітки в режимі кодування під час прямого оберту. Циклічно перевіряючи решітку на наявність пустої клітинки і якщо вона знайдена – на її місці встановлюється і-й елемент відкритого тексту. Таким чином заповнюється тимчасовий масив (виконавши 4 оберти решітки). Закритий код отримується також циклічно: спочатку до результату вибираються символи першого рядку матриці, потім другого і т.д. (рис.14).

Принцип роботи режиму розшифрування аналогічний. Спочатку переносимо всі символи закодованого повідомлення до тимчасового масиву, та починаємо виконувати оберти решітки, вибираючи в пустих клітинках літери. Таким чином можна отримати відкрите повідомлення (рис.15).

В програмі також існує подія введення тексту в текстове поле. Виконується перевірка на кількість символів у рядку. Якщо більше 36 – блокується можливість введення.

Результати роботи

Циклічно перевіряється решітка на наявність пустої клітинки і якщо вона знайдена – на її місці встановлюється і-й елемент відкритого тексту. Так заповнюється тимчасовий масив виконавши 4 оберти решітки:

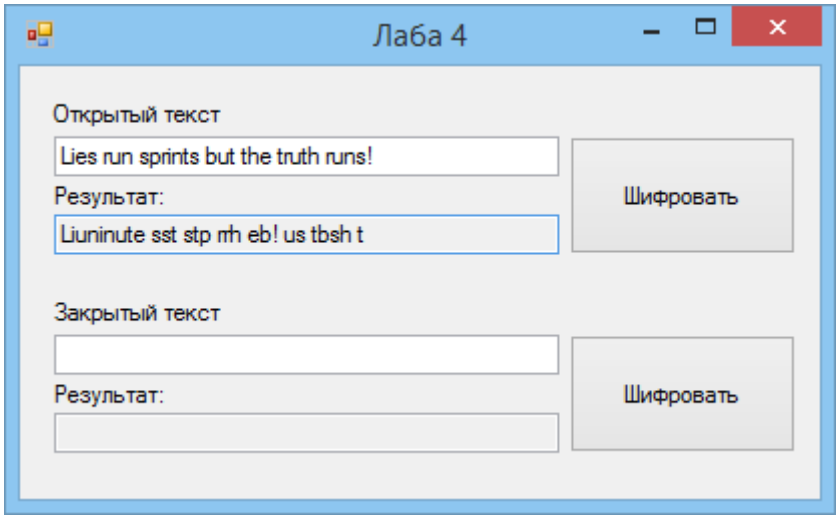

Рисунок 14 - Кодування інформації

Принцип розкодування тексту - спочатку переносимо всі символи закодованого повідомлення до тимчасового масиву та обертаємо решітку, вибираючи в пустих клітинках літери:

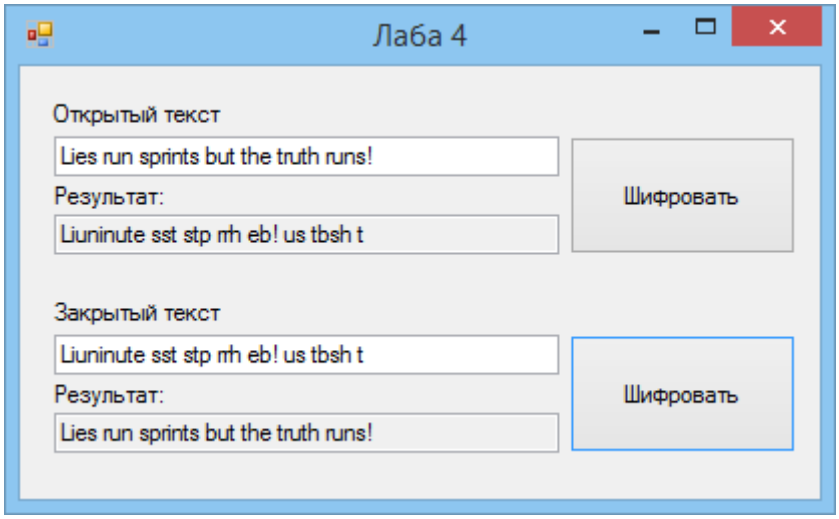

Рисунок 15 - Декодування інформації

5.4. Контрольні питання

1. Як виглядає інструмент для кодування та розкодування тексту – «Решітка Кардано»?
2. В чому полягає методика кодування за допомогою решітки Кардано?
3. Принцип роботи розшифрування?

5.5. Список використаних джерел

1. Шифр, тайнопись // Энциклопедический словарь Брокгауза и Эфрона: в 86 томах (82 т. и 4 доп.). - СПб, 1890-1907.
2. Горбенко И.Д., Потий А.В., Избенко Ю.А., Орлова С.Ю. Анализ схем поточного шифрования, представленных на европейский конкурс NESSIE // правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні. – К.: НТУ «КПІ», ДСТСЗІ СБУ, 2002. – Вып. 5. – С. 92-110.
3. ДСТУ 4145-2002. Інформаційні технології. Криптографічний захист інформації. Цифровий підпис, що ґрунтується на еліптичних кривих. Формування та перевірка.
4. Горбенко И.Д. Скрыпник Л.В., Головашич С.А., Гриненко Т.А. Стандарт симметричного шифрования 21 века: свойства, режимы работы, реализация // Радиотехника. Всеукр. межвед. научн.-техн. сб. 2001. – Вып. 119. - С. 22-35.
5. Молдовян А.А., Молдовян Н.А., Рад Б.Я. Криптография. - Серия "Учебники для вузов. Специальная литература". - СПб.: Издательство "Лань", 2000.
6. Клименко І.В., Линьов К.О., Горбенко І.Д., Онопарієнко В.В. Електрониий документообіг в державному управлінні. Навчальний посібник. – К.: НАДУ, 2009; Х.: Вид-во «ФОРТ», 2009. – 232 с.

6. Лабораторна робота №5

Тема: Захист інформації за допомогою **метода одноразового блокноту** – **шифру Вернама.**

Мета: Навчитися створювати програмне забезпечення, за допомогою якого користувач зможе виконувати шифрування відкритого тексту за методом одноразового блокноту.

6.1. Завдання

1. Написати процедуру для шифрування цитати за допомогою методу одноразового блокноту.
2. Вивести результати роботи та створити дружній інтерфейс.

Мова програмування обирається за згодою з викладачем.

6.2. Короткі теоретичні відомості

Шифр Вернама (англ. Verrnam Cipher, інша назва одноразовий блокнот - схема одноразових блокнотів) - система симетричного шифрування, вперше запропонована в 1882 році Ф. Міллером і заново винайдена в 1917 році співробітником AT & T Гілбертом Вернама.

Шифр Вернама є єдиною системою шифрування, для якої доведена абсолютна криптографічна стійкість. При цьому він є однією з найпростіших криптосистем.

Криптосистема була запропонована для шифрування телеграфних повідомлень, які представляли собою бінарні тексти, в яких відкритий текст представляється в коді Бодо (у вигляді п'ятизначних «імпульсних комбінацій»).

У цьому коді, наприклад, літера «А» мала вигляд (11000). На паперовій стрічці цифрі «1» відповідав отвір, а цифрі «0» - його відсутність.

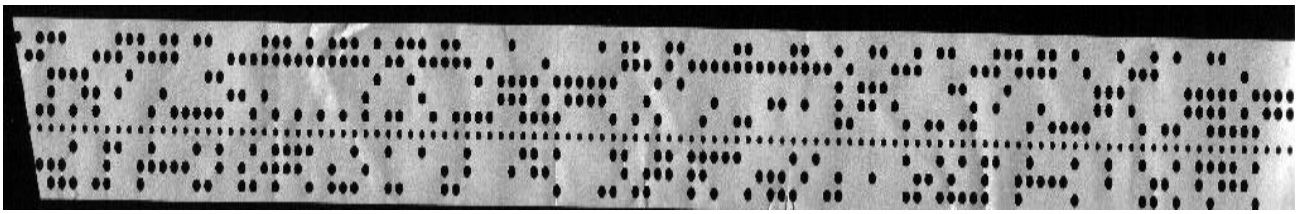

Рисунок 16 – Представлення шифрованих телеграфних повідомлень

Для творення шифртекста відкритий текст об'єднується операцією «виключаюче АБО» з ключем (званим одноразовим блокнотом або шифрблокнотом). При цьому ключ повинен володіти трьома критично важливими властивостями:

1. Мати випадковий рівномірний розподіл: P_ { до } (k) = 1 / 2 ^ { N }, де k - ключ, а N - кількість бінарних символів в ключі.
2. Збігатися за розміром із заданим відкритим текстом.
3. Застосовуватися тільки один раз.

Без знання ключа таке повідомлення не піддається аналізу. Навіть якби можна було перепробувати всі ключі, в якості результату було б отримано б всі можливі повідомлення даної довжини плюс колосальну кількість безглуздих дешифровок, що виглядають як безладне нагромадження букв. Але й серед осмислених дешифровок не було б ніякої можливості вибрати шукану. Коли випадкова послідовність (ключ) поєднується з невипадковою (відкритим текстом), результат цього (шифртекст) виявляється абсолютно випадковим і, отже, позбавленим тих статистичних особливостей, які могли б бути використані для аналізу шифру.

Недоліки шифру:

- Для роботи шифру Вернама необхідна істинно випадкова послідовність (ключ). За визначенням, послідовність, отримана з використанням будь-якого алгоритму, є не істинно випадковою, а псевдовипадковою. Тобто, потрібно отримати випадкову послідовність перестановок алгоритмічно (а, наприклад, використовуючи радіоактивний розпад, створюваний електронним генератором білого шуму, або інші досить випадкові події). Щоб зробити розподіл гранично близьким до рівномірного,

випадкова послідовність зазвичай пропускається через хеш-функцію на зразок MD5.

- Недоліком використання шифру Вернама є відсутність підтвердження автентичності та цілісності повідомлення. Одержувач не може упевнитися у відсутності ушкоджень або в дійсності відправника. Якщо третя сторона якимось чином дізнається повідомлення, вона легко відновить ключ і зможе підмінити послання на інше такої ж довжини. Вирішенням проблеми є застосування хеш -функції. Від відкритого тексту обчислюється хеш- функція, і її значення шифрується разом з повідомленням. При певній зміні повідомлення, значення хеш-функція зміниться. Таким чином, навіть якщо зловмисник придбав шифрблокнота , не знаючи алгоритм обчислення хеш-функції, він не зможе використовувати його для передачі інформації.
- Під рукою завжди необхідно мати достатню кількість ключів, які можуть знадобитися в подальшому для шифрування великих обсягів відкритого тексту. Реальний же обсяг тексту найчастіше важко оцінити заздалегідь, особливо це стосується дипломатичної та військової сфери, де ситуація здатна мінятися швидко і непередбачувана. Це може призводити до браку ключів, що може змусити шифрувальника або використовувати ключ(і) повторно, або повністю перервати шифрований зв'язок.
- Проблемою є захищена передача послідовності і збереження її в таємниці. Якщо існує надійно захищений від перехоплення канал передачі повідомлень, шифри взагалі не потрібні: секретні повідомлення можна передавати по цьому каналу. Якщо ж передавати ключ системи Вернама за допомогою іншого шифру (наприклад, DES), то отриманий шифр виявиться захищеним рівно настільки, наскільки захищений DES. При цьому, оскільки довжина ключа та ж, що і довжина повідомлення, передати його не простіше, ніж повідомлення.
- Шифроблокнота на фізичному носії можна вкрасти або скопіювати.

- Можливі проблеми з надійним знищенням використаної сторінки. Цьому схильні як паперові сторінки блокнота, так і сучасні електронні реалізації з використанням компакт - дисків або флеш -пам'яті.
- Шифр Вернама чутливий до будь-якого порушення процедури шифрування. Бували випадки, коли одна і та ж сторінка блокнота з різних причин застосовувалася двічі.

6.3. Приклад до виконання лабораторної роботи №5

Теоретичні відомості

Перед завантаженням програми виконується створення функцій, які потрібні для роботи з відкритим текстом. Це функція отримання одноразового ключа «getkey» та функція виконання шифрування тексту «blocnot». Розглянемо дані функції більш детальніше.

Функція отримання ключа виконує генерацію випадкового, унікального та одноразового ключа, кожен символ якого є символом так званого ASCII алфавіту. Після виклику функції, програма створює символьний масив типу Char, кількість елементів якого – це кількість символів у рядку, який входить в функцію. Вхідний параметр – це відкритий текст. Наступний крок – це генерація випадкових чисел, кожне з яких повинно входити в діапазон від 0 до 255. Отримавши число ми можемо отримати еквівалентний символ з таблиці ASCII. Результатом роботи функції є масив символів.

Функція шифрування тексту також досить проста. Створюємо змінну, яка буде приймати в себе закодоване повідомлення. Кодується повідомлення наступним чином: програма бере і-й елемент рядку, причому бере в форматі char та виконує складання по модулю з і-м елементом ключа. Ключ та текст повинні бути однакового розміру. Результатом є закодоване повідомлення, яке має таку саму кількість символів, як і вихідне повідомлення (рис.17).

Щоб виконати кодування або декодування тексту, потрібно перевірити, який режим роботи програми користувач вибрав. Якщо вибрав режим шифрування – потрібно запустити функцію отримання ключа та вивести його в форму. Якщо вибрано режим розшифрування – потрібно взяти ключ з форми.

Для декодування викликаємо функцію «blocnot», вхідними параметрами якої є масив символів ключа та масив символів тексту (рис.18).

Результати роботи

Кодується повідомлення наступним чином: програма вибирає в форматі char і-й елемент рядку та виконує складання по модулю з і-м елементом ключа. Ключ та текст повинні бути однакового розміру. Результатом є закодоване повідомлення, яке має таку саму кількість символів, як і вихідне повідомлення:

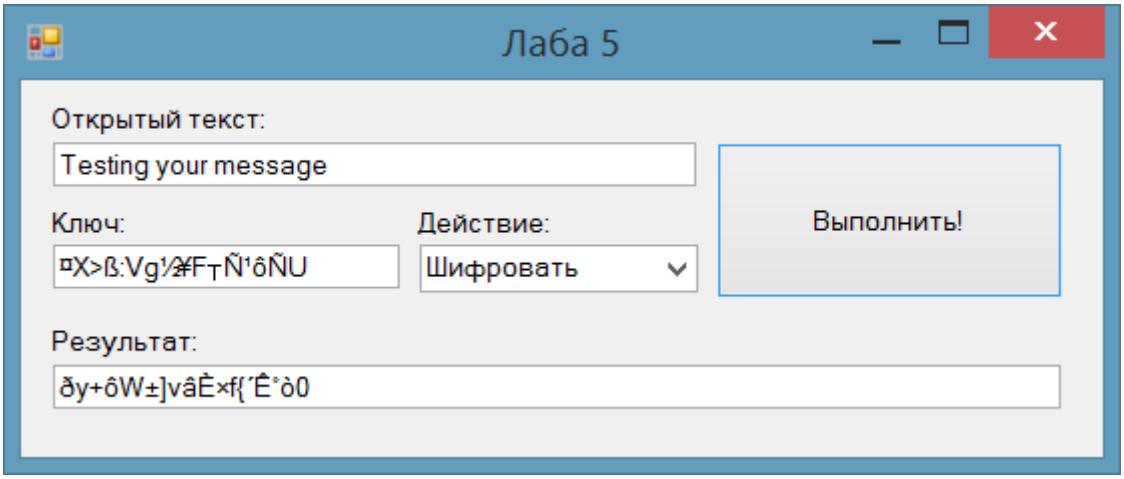

Рисунок 17 - Кодування інформації

Алгоритм шифрування та розшифрування однаковий, тому просто викликаємо функцію «blocnot», вхідними параметрами якої є масив символів ключа та масив символів тексту:

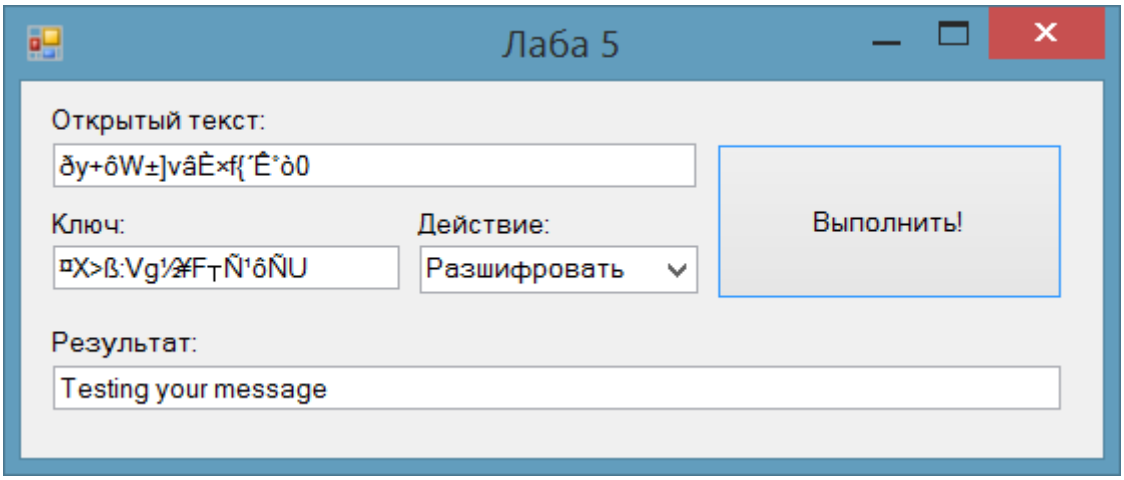

Рисунок 18 - Декодування інформації

6.4. Контрольні питання

1. Шифр Вернама як стійка система криптозахисту?
2. Важливі властивості ключа.
3. Як працює метод одноразового блокнота для шифрування телеграфних повідомлень?
4. Недоліки метода одноразового блокнота.

6.5. Список використаних джерел

1. Криптография / Под ред В.П. Шерстюка, Э.А. Применко, А.В. Бабаш, Г.П. Шанкин. – М.: СОЛОМОН-Р, 2002. – 512 с.
2. Фомичов В.М. – «Дискретная математика и криптология. Курс лекций» (рос.) // Москва Диалог МИФИ. - 2003. - С. 239 - 246.
3. Габидулин Е.М., Кшевецкий А.С., Колибельников А.И.– «Защита информации». - 2011. - С. 41 - 43.
4. Кузнецов А.А., Лисицкая И.В., Исаев С.А. Линейные свойства блочных симметричных шифров, представленных на украинский конкурс // Прикладная радиоэлектроника. – 2-11. – Том 10. - №2. – С. 135-140.
5. Кофман А. Введение в теорию нечётких множеств. Пер. с франц. / Под ред. Травкина. – М.: Радио и связь, 1982. – 432 с.
6. ДСТУ 3008-95. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення.
7. Мецезис А., Ван Аршот П., Ватсон С. Руководство по прикладной криптографии. - CRC Press, 1997, электронная копия. – 662 с.
8. Юдін О.К., Корченко О.Г., Конахович Г.Ф. Захист інформації в мережах передачі даних. - К.: Вид-во ТОВ «НВП» ІНТЕРСЕРВІС», 2009. – 716 с.