УДК 004.056.5 DOI: 10.30977/VEIT.2021.20.0.07

Аналіз та програмна реалізація модифікованого криптографічного шифру Вернама та шифру Цезаря

Фастовець В. І.[1]

[1]Харківський національний автомобільно-дорожній університет, Україна

Анотація. *В статті проведено аналіз та реалізовані практичні підходи до підвищення інтересу студентів, які навчаються на спеціальності «Кібербезпека», до курсу криптографії. Обрано гнучкі методи, що дозволяють різні підходи до навчання. Реалізовано модифікований симетричний шифр Вернама та шифр Цезаря з використанням концепції, що комбінує сучасні мови програмування та принципи криптографії, які студенти вивчають на профільних спеціальностях.*

Ключові слова: *шифр, криптографія, програмування, секретний ключ, стійкість, криптосистема, навчання.*

Вступ

Сьогодні у світі інформаційних технологій без кордонів та цифрової глобалізації питання інформаційної безпеки стоїть на порядку денному як ніколи актуально.

З огляду на величезну кількість передач даних у сучасному світі, підтримувати безпеку даних надзвичайно важливо та складно. У цій роботі пропонується дослідження та реалізація модифікованого криптографічного шифру Вернама та шифру Цезаря, які можна використовувати для захисту конфіденційної інформації [1-4].

Алгоритми програмно реалізовані на мові C++, їх дослідження проведено експериментально, і результати оцінки, а також порівняння з аналогічними роботами демонструють хороші якості запропонованої схеми.

У роботі розглянуто захист даних за допомогою криптографічних методів та алгоритмів, що забезпечує безпеку даних, що передаються, та конфіденційність, цілісність та доступність інформації. Криптографічні алгоритми діляться на дві категорії: криптографія із симетричним ключем, де один ключ використовується як для шифрування, так і для дешифрування, та криптографія з асиметричним ключем: коли використовуються два різні ключі, один використовується для шифрування, а другий – для дешифрування.

Аналіз публікацій

Постійне поліпшення інтернет зв'язку стало невід'ємною частиною життя, наприклад, на роботі або в навчальному закладі, і навіть при повсякденному використанні, наприклад, під час листування електронною поштою та обміну миттєвими повідомленнями.

З розвитком технологій зберігання та обміну даними різними способами по мережі від відправника до адресата постає питання захисту цієї інформації від загроз або перешкод. Захист інформації має здійснюватися за допомогою методів захисту, які забезпечують безпеку даних серед задіяних сторін [5-7].

Інформаційна безпека стає все більш важливим компонентом будь-якої інформаційної системи. Дані та інформація, що зберігаються на складах, магазинах, банках, підприємствах та проходять через канали зв'язку, повинні бути захищеними.

Використання криптографічних механізмів і протоколів може вирішити багато проблем безпеки (цілісність, конфіденційність, достовірність і неспростовність інформації).

Хоча спочатку криптографія виникла у військовій галузі, нині її використання широко поширилося на різні цивільні сфери, як-то інтернет-банкінг, державні установи, електронна комерція та соціальні мережі [8-10].

Розвиток криптографічних технологій має безпосередній вплив на економіку, соціальні та політичні аспекти суспільства. З одного боку, повсюдне використання криптографії сьогодні підвищує важливість курсу кібербезпеки в університеті. З іншої сторони, викладачі з кібербезпеки стикаються з деякими труднощами в ході читання цього курсу.

Кібербезпека тісно пов'язана з іншими науками; вона є перетином математики, інформатики, комп'ютерних мереж та обробки даних. Основи ж криптографії, однієї з найважливіших складових кібербезпеки, лежать глибоко в таких прикладних науках, як теорія чисел і абстрактна алгебра.

Однак окрім різної математичної підготовки, студенти мають ще і різні академічні інтереси. Щоб мати можливість вивчати криптографічні системи, студенти повинні мати сильні математичні знання, особливо в таких дисциплінах, як теорія чисел, абстрактна алгебра, теорія ймовірності і статистика [11-12].

Вища математика, лінійна алгебра є дуже важливими предметами. Але, якщо ми хочемо заохотити студентів криптографією, треба ефективніше використовувати всі аспекти IT кластеру. Ідеально для цього підходить саме реалізація алгоритмів та програм за допомогою мов програмування.

Перевага криптографії з симетричним ключем полягає в тому, що робота з цим методом дуже проста для користувачів, так як для шифрування використовується один ключ, як і для цілей дешифрування. Цей ключ має бути секретним і повинен бути відомим тільки відправнику та одержувачу і нікому іншому.

З іншого боку, у криптографії з відкритим ключем є два ключі. Один ключ називається відкритим ключем, який може бути доступний будь-кому, хто хоче зашифрувати повідомлення, а інший називається секретним ключем або закритим ключем, який повинен зберігатися тільки у одержувача.

На жаль, це викриває недоліки безпеки, так як цілісність шифрування повністю залежить від надійності пароля.

Хоча майже всі університети мають сучасні електронні лабораторії з мережевими комп'ютерами та доступом до Інтернету, практика викладання курсу криптографії в багатьох університетах не включає експериментування. Ця відсутність практичного застосування криптографії є основною причиною відсутності інтересу студентів до виконання практичних завдань [13-15].

Розвиток сучасних та найпопулярніших мов програмування надає можливість створювати як криптографічні десктопні додатки під операційну систему Windows так і мобільні застосунки під операційні системи IOS та Android. Засоби ж Web-програмування дозволяють реалізувати криптографічні методи на мові JS та Python. Скрипт-програми реалізовані на цих мовах програмування доступні користувачам з браузерів та месенджерів.

Мета та постановка задачі

Метою роботи є підвищення інтересу студентів, які навчаються на спеціальності «Кібербезпека», до курсу Криптографії. Дуже важливо і корисно проілюструвати, де і яким чином можливо створювати програмні реалізації методів шифрування/дешифрування, а також навести приклади модифікації існуючих алгоритмів.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

- провести аналіз та реалізувати практичні підходи до підвищення інтересу студентів до криптографії, методи повинні бути гнучкими та дозволяти різні підходи до навчання;
- реалізувати модифікований симетричний шифр Вернама;
- реалізувати шифр Цезаря.

Аналіз та програмна реалізація шифру Вернама

Шифр Вернама є системою симетричного шифрування, і був запропонований співробітниками AT & T Гильбертом Вернамом і Мейджором Джозефом Моборном в 1917 році. У 1949 роках була опублікована робота Клода Шеннона «Математична теорія криптографії» , де він довів абсолютну стійкість шифру Вернама і його висновком стало наступне твердження: «шифр Вернама - найбезпечніша криптосистема з усіх наявних».

Однак для того щоб шифр дійсно був стійким, необхідно виконання наступних трьох правил:

1. Ключ для шифрування вибирається випадковим чином.
2. Довжина ключа повинна дорівнювати довжині відкритого тексту.
3. Ключ повинен використовуватися тільки один раз.

У криптографії шифр Вернама називають також «схемою одноразових записників». На

практиці можна один раз фізично передати носій інформації з довгим істинно випадковим ключем, а потім, у міру необхідності, пересилати повідомлення. На цьому заснована ідея шифрозаписника: шифрувальник дипломатичною поштою або при особистій зустрічі забезпечується записником, кожна сторінка якого містить ключі. Такий же записник є і у приймаючої сторони. Використані сторінки знищуються.

Шифр Вернама базується на двійковій арифметиці. Основним об'єктом розгляду в даному методі шифрування є логічна операція XOR (виключне АБО). Таким чином, так як використовується двійкова система числення, то всі операції будуть здійснюватися над нулем (0) і одиницею (1). Логічна операція XOR на відміну від операції OR при логічному порівнянні 0 і 1 дає 1, при порівнянні 1 з 1 дає 0, а при 0 з 0 дає 0. Отже, якщо ми виконаємо операцію XOR над числами 10110 і 11010, то отримаємо: 10110 xor 11010 = 01100.

Далі, так як шифр працює з двійковою системою числення, необхідно пам'ятати, що букви - це всього лише деяка інтерпретація числа, тобто число є кодом символу деякої таблиці кодувань. Наприклад, найбільш популярні таблиці кодувань це: ANSI, ASCII і UTF (unicode). В кожній таблиці один і той же символ може мати різний код, тому необхідно використовувати одне і те ж кодування для шифрування і дешифрування. Таким чином, перед тим як здійснити шифрування, необхідно перевести всі символи в їх однозначну числову інтерпретацію.

Використовувати будемо власну систему кодування. При застосуванні шифру Вернама до письмового тексту, кожній букві використовуваного алфавіту дамо відповідний їй порядковий номер у двійковій системі числення. Наприклад, якщо ми використовуємо український алфавіт, то це буде виглядати так: а → 00000, б → 00001, в → 00010, г → 00011, ... я → 111111. Тим самим ми визначили свою таблицю кодування.

Після цього, написавши повідомлення і вигадавши ключ, кожен символ необхідно перетворити в його числове значення, відповідне до нашої таблиці кодування, і після цього здійснити операцію XOR над кожною відповідною парою. Так як даний метод шифрування є симетричним, отже застосувавши операцію XOR для кожної пари символів шифр-тексту (шифрограми) і ключа, ми отримаємо відкритий текст.

Програмну реалізація шифру Вернама було вирішено робити на мові програмування C++.

Припустимо, що у нас є рядок або масив символів - Str0. Це буде відкритий текст, який треба зашифрувати. Тепер нам треба визначити випадковий ключ з довжиною, що дорівнює довжині відкритого тексту. Скористаємося стандартною функцією генерації випадкових чисел rand для C++. Для реалізації шифрування опишемо цикл, в якому будемо посимвольно здійснювати операцію XOR. Але для цього нам знадобитися ще оголосити масив-приймач, в який ми помістимо зашифрований текст.

Програмна реалізація шифрування шифром Вернама:

```
void shifr_Vernam
{ int i,len; // Визначаємо необхідні змінні
len = strlen("Infomatiks");// визначаємо довжину рядка відкритого тексту
char *Str0 = new char[len]; // оголошуємо динамічний масив зазначеної довжини для відкритого тексту
char *key = new char[len];  //точно такий же масив оголошуємо для ключа
char *shStr = new char[len];  // оголошуємо масив-приймач для зашифрованого тексту
Str0=" Infomatiks";  // поміщаємо в масив відкритий текст
// визначаємо ключ випадковим чином
for(i = 0; i < len; i++)
key[i]=(char)rand()%255;
// безпосередньо саме шифрування
for(i = 0; i < len; i++)
shStr[i]= Str0[i]^key[i];
// для наочності виведемо на екран результат роботи
printf("\nOtkrytyi text: %s", Str0);
printf("\nZashifrovanyi text: %s", shStr);}
```

Для дешифрування ті самі дії проводимо для зашифрованого тексту.

Аналіз та програмна реалізація шифру Цезаря

Розглянемо систему шифрування, яка носить ім'я «шифр Юлія Цезаря». Знаний римський імператор і полководець, що жив в 1 столітті до нашої ери, використовував цей шифр в своєму листуванні.

Шифр Цезаря стосовно до української мови полягає в наступному. Кожна буква повідомлення замінюється на іншу, яка в українському алфавіті знаходиться від вихідної на три позиції далі.

Таким чином, буква А замінюється на Г, Б

на Д и так далі до букви Я, яка замінюється на В.

Для розшифрування повідомлення необхідно знати тільки сам алгоритм шифрування. Будь-яка людина, що знає спосіб шифрування, легко може розшифрувати секретне повідомлення. Таким чином, ключем в даному методі є сам алгоритм.

Спробуємо модифікувати шифр Цезаря для його вдосконалення. Можна було б спробувати розширити алфавіт з 33 до 36 символів і більше за рахунок включення розділових знаків, апострофа і пробілів. Це збільшення алфавіту замаскувало б довжину кожного окремого слова.

Припустимо, що букви зсуваються не на три знака вправо, а на n (0 <n <33). В цьому випадку в системі шифрування з'являється ключ - число n - параметр зсуву. Так як n може приймати різні значення, знання одного тільки алгоритму не дозволить противнику розшифрувати секретне повідомлення.

Противник, намагаючись знайти значення секретного ключа, зможе тільки проводити атаку по шифротексту способом послідовного перебору всіх можливих ключів (так званий метод "грубої сили").

При такому методі відбувається пошук осмисленого повідомлення. Таке допущення про одиничність рішення цілком обґрунтовано, коли вихідне повідомлення складено на одній з природних мов і містить більше п'яти-шести знаків. Але якщо повідомлення дуже коротке, можливих рішень може бути кілька. Єдине рішення також дуже важко знайти, якщо вихідне повідомлення, складається, наприклад, з цифр.

При методі послідовного перебору всі отримані варіанти будуть рівнозначні і зловмисник не зможе зрозуміти, яка саме комбінація істинна. Аналізуючи шифротекст, він не зможе знайти значення секретного ключа. Звичайно, у якийсь момент один з варіантів підійде до кода, але в настільки простий метод шифрування не можна розраховувати на більшу секретність.

На основі цих міркувань була побудована математична модель шифра Цезаря. Якщо співставити кожному символу алфавіту його порядковий номер (проводячи нумерацію з 0), то шифрування і дешифрування можна висловити формулами модульної арифметики:

$$y = (x + k) \bmod n;$$
$$x = (y - k + n) \bmod n,$$

де x – символ відкритого тексту, y – символ шифрованого тексту, n – потужність алфавіту, k – ключ.

З точки зору математики шифр Цезаря є окремим випадком афінного шифру.

Програмна реалізація шифрування / дешифрування шифром Цезаря для латинського алфавіту:

```
#include "stdafx.h"
#include <iostream>
#include <string>
#include <conio.h>
#include <stdlib.h>
#include <sstream>
#include <fstream>
using namespace std;
int main()
{int k; // Змінна вибору - шифрування/дешифрування
int shift; //Величина зсуву
string result = ""; // рядок - результат
cout<<" Введіть 1 для шифрування та для розшифрування 2\n"; cin>>k;
switch (k) //Якщо k
{ case 1: // Якщо вибрано шифрування
{ cout<<" Введіть значення зсуву для шифрування \n";
cin>>shift;
if (shift > 26)
shift = shift % 26; // обчислення зсуву
cout<<"Read of file...\n";
// читання файлу
string s; // Рядок, зчитаний з файлу
ifstream in("Test.txt");
getline(in,s);
cout<<"Текст файлу: \n"<<s<<endl;
in.close();
cout<<"Читання завершено!\n";
cout<<"Шифрування...\n";
for (int i = 0; i < s.length(); i++) {
// Якщо не латиниця
if (((int)(s[i]) < 65)||((int)(s[i]) > 122))
result += s[i];
// Якщо буква є рядковою
if (((int)(s[i]) >= 97) && ((int)(s[i]) <= 122))
// Якщо буква після зсуву виходить за межі алфавіту
{ if ((int)(s[i]) + shift > 122)
// Додавання в рядок результатів символа
result += (char)((int)(s[i]) + shift - 26);
// Якщо буква після зсуву виходить за межі алфавіту
else
// Додавання в рядок результатів символа
result += (char)((int)(s[i]) + shift); }
```

```
// Якщо буква є прописною
if (((int)(s[i]) >= 65) && ((int)(s[i]) <= 90)){
// Якщо буква після зсуву виходить за межі алфавіту
if ((int)(s[i]) + shift > 90)
// Додавання в рядок результатів символа
result += (char)((int)(s[i]) + shift - 26);
// Якщо буква може бути зсунута в межах алфавіту
else
// Додавання в рядок результатів символа
result += (char)((int)(s[i]) + shift);      }     }
cout<<"Шифрування завершено!\n";
cout<<"Результат:\n";
cout<<result; //Вивід результату
break;          }
case 2:
// Якщо вибрано дешифрування
{ cout<<" Введіть значення зсуву для розшифрування \n";
cin>>shift;
if (shift > 26)
shift = shift % 26;
cout<<"Прочитати файл...\n";
string s;
ifstream in("Test.txt");
getline(in,s);
cout<<"Текст файлу: \n"<<s<<endl;
 in.close();
cout<<"Читання завершено!\n";
cout<<"Розшифровка...\n";
for (int i = 0; i < s.length(); i++)
{ if (((int)(s[i]) < 65)||((int)(s[i]) > 122))
result += s[i];
if (((int)(s[i]) >= 97) &&
((int)(s[i]) <= 122))
{ if ((int)(s[i]) - shift < 97)
result += (char)((int)(s[i]) - shift + 26);
else
result += (char)((int)(s[i]) - shift);  }
if (((int)(s[i]) >= 65) &&
((int)(s[i]) <= 90))
{ if ((int)(s[i]) - shift < 65)
result += (char)((int)(s[i]) - shift + 26);
else
result += (char)((int)(s[i]) - shift);  }      }
cout<<"Розшифровка завершена!\n";
cout<<"Результат:\n";
cout<<result; //Вивід результату
break; }
default:
// Якщо помилкове значення
{  cout<<"Помилка\n";
break;  } }
getch();
return 0;  }
```

Висновки

У статті описано погляд на криптосистеми шифрування з боку мов програмування.

Наведені аналіз та реалізація модифікованого криптографічного шифру Вернама та шифру Цезаря з використанням концепції, що комбінує сучасні мови програмування та принципи криптографії, які студенти вивчають на профільних спеціальностях.

Використання криптографії як навчального засобу допоможе студентам розвивати свої навички в програмуванні та розуміти поняття кібербезпеки на прикладах реалізації модифікованого шифру Вернама та шифру Цезаря.

Конфлікт інтересів

Автор заявляє, що немає конфлікту інтересів щодо публікації цієї статті.

Література

1. Bhandari R., Kirubanand V. B. (2019) Enhanced encryption technique for secure iot data transmission. 9. 5. 3732–3738.
2. Chałupnik R., Kędziora M., Jóźwiak P., Jóźwiak I. (2020) Correspondent Sensitive Encryption Standard (CSES) Algorithm in Insecure Communication Channel in IEEE Systems Journal. 12. 4. 90–98.
3. Bhanot R., Hans R. (2015) A review and comparative analysis of various encryption algorithms. Int. J. Secur. its Appl. 9. 4. 289–306.
4. Din R., Bakar R., Utama S., Jasmis J., Elias S. J. (2019) The evaluation performance of letter-based technique on text steganography system. Bull. Electr. Eng. Informatics. 8. 1. 291–297.
5. Singh N. (2020) XOR Encryption Techniques of Video Steganography: A Comparative Analysis. Springer International Publishing 203–214.
6. Namasudra S. (2019) An improved attribute-based encryption technique towards the data security in cloud computing. Concurr. Comput. 31. 3. 1–15.
7. Ogiela M. R. (2019) Cognitive solutions for security and cryptography. Cogn. Syst. Res. 55. 258–261.
8. Hatkar S. S., Bhagyashri K. Pawar. (2016) Symmetric key algorithm using vernam cipher: VSA. Inventive Computation Technologies (ICICT). International Conference on. Vol. 3.
9. Zimmermann M., Staicu C. A., Tenny C., Pradel M. (2019) Small world with high risks: A study of security threats in the npm ecosystem in Proc. 28th USENIX Sec., Santa Clara. CA.
10. Nath A., Das S., Chakrabarti A. (2010) Data Hiding and Retrieval. Proceedings of IEEE International conference on Computer

Intelligence and Computer Network held at Bhopal.
11. Adamovic S., Sarac M., Stamenkovic D., Radovanovic D. (2018) The Importance of the Using Software Tools for Learning Modern Cryptography. International Journal of Engineering Education. 34. 1, 256–262.
12. Yang F., Zhong C., Yin M., Huang Y. (2009) Teaching Cryptology Course Based on Theory-Algorithm-PracticeApplication Mode. Proceedings of the First International Workshop on Education Technology and Computer Science. Wuhan. 468–470.
13. Adamovic S., Sarac M., Veinovic M., Milosavljevic M., Jevremovic A. (2014) An Interactive and Collaborative Approach to Teaching Cryptology, Journal of Educational Technology & Society. 17(1). 197–205.
14. Olejar D., Stanek M (1999) Some Aspects of Cryptology Teaching. Proceedings of the 1st World Conference on Information Security Education. Stockholm (Sweden). 1–9.
15. Song X., Deng H. (2009) Taking Flexible and Diverse Approaches to Get Undergraduate Students Interested in Cryptography Course. Proceedings of the First International Workshop on Education Technology and Computer Science. Wuhan. 490–494.

Фастовець Валентина Іванівна[1], к.т.н., доц.,
кафедри кафедра Інформатики та прикладної математики, тел. +380632840672,
e-mail: tinafast2013@gmail.com,
ORCID: 0000-0002-8428-747X.

[1]Харківський національний автомобільно-дорожній університет, 61002, Україна, м. Харків, вул. Ярослава Мудрого, 25.

Analysis and software implementation of the modified cryptographic Vernam cipher and the Caesar cipher

Abstract: Problem. *Modern cryptography is a very important part of cybersecurity and confidentiality of many operations. It covers almost all parts of our lives, from e-commerce to specialized education of students.* ***Goal****. The advantage of symmetric key cryptography is that working with this method is very easy for users, as one key is used for encryption, as well as for decryption purposes, and this key must be secret and should be known only to the sender and recipient and no one else. On the other hand, public key cryptography has two keys. Unfortunately, this exposes the inherent security flaws, as the integrity of the encryption depends entirely on the password. It was decided to consider the implementation of a modified symmetric Vernam cipher that avoids these problems, and its modification and experimental studies should further strengthen data protection.* ***Methodology.*** *Higher mathematics, linear algebra are very important subjects. But if we want to encourage students with cryptography, we need to use all aspects of the IT cluster more effectively. Ideal for this is the implementation of algorithms and programs using programming languages. It is very important and useful for students studying Cybersecurity to illustrate where and how it is possible to create software implementations of encryption / decryption methods.* ***Results.*** *The article presents the analysis and implementation of the modified cryptographic Vernam cipher and Caesar cipher using a concept that combines modern programming languages and the principles of cryptography, which students study in subject-oriented specialties.* ***Originality.*** *An original approach to teaching Cybersecurity students by implementing ciphers using applied programming is described.* ***Practical value****. Using cryptography as a learning tool will help students develop their programming skills and effectively understand the concept of cybersecurity in real-world examples.*

Keywords: *cipher, cryptography, programming, secret key, stability, cryptosystem, study.*

Fastovets Valentyna[1], Assoc. Prof, cand. eng. sc.,
e-mail: tinafast2013@gmail.com,
tel. +380632840672,
ORCID: 0000-0002-8428-747X.

[1]Kharkiv National Automobile and Highway University, 25, Yaroslava Mudrogo str., Kharkiv, 61002, Ukraine.