УДК 004.056.55
Ю.М. Щебланін, О.А. Курченко, Н.А. Гончаренко
DOI: 10.31673/2409-7292.2019.048287

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ЗАХИСТУ ІНФОРМАЦІЇ ВІД НЕСАНКЦІОНОВАНОГО ДОСТУПУ

В даній статті проведено детальний аналіз симетричних шифрування. На основі проведеного аналізу розроблено програмне забезпечення з використанням алгоритмів шифрування DES та AES.
Ключові слова: алгоритм, шифрування, криптографічні системи, шифр, ключ, програма.

Вступ. В сучасних умовах XXI століття проблеми інформаційної безпеки України стають все більш актуальними і вимагають постійного поглибленого вивчення та вдосконалення.

Зі стрімким розвитком інформаційних технологій найактуальнішою стає проблема забезпечення безпечного функціонування інформаційно-телекомунікаційних систем та безпеки інформації.

Для запобігання втрат та непередбачуваних наслідків від загроз безпеки інформації необхідно впровадження системи комплексного захисту інформації, яка включає в себе застосування криптографічних та технічних засобів захисту, а також виконання ряду організаційно-технічних та нормативно-правових заходів.

Інформаційна та кібернетична безпека має забезпечуватися на всіх рівнях діяльності держави, суспільства та людини.

На жаль, на сьогодні в Україні не у всіх сферах суспільного життя і в діяльності органів державної влади вирішено проблему захисту інформації через відсутність належного фінансування та матеріально-технічного забезпечення, недооцінки важливості розуміння у питанні інформаційної безпеки.

Постановка задачі. В даний час існує велика кількість систем захисту інформації, які є дуже поширеними і знаходяться в постійному розвитку.

Необхідність використання систем захисту інформації обумовлена рядом проблем, серед яких варто виділити:

1. промислове шпигунство;
2. крадіжка і використання;
3. несанкціонована модифікація інформації.

Всі ці проблеми необхідно вирішувати, тому виникає необхідність розробки нового програмного забезпечення для підвищення ефективності захисту інформації для подолання даних проблем.

Основна частина

Основними процедурами захисту інформації є **шифрування** і **розшифрування**, але також необхідно не забувати про можливість **дешифрування**.

Шифрування – це перетворення вихідного (інакше – відкритого) тексту T в шифрований текст S з використанням конфіденційних даних K відповідно до деякого алгоритму E (від англ. encryption – шифрування). Параметр K в криптографії називається ***ключем***. Ключ є тією інформацією, без якої неможливе відновлення початкового повідомлення. Рівняння шифрування записують у вигляді:

$$S = E(T, K) \text{ або } S = E_k(T).$$

Розшифрування – обернена відносно шифрування процедура D (від англ. decryption – розшифрування), в результаті виконання якої шифрований текст з використанням ключа перетвориться у вихідний:

$$T = D(S, K) \text{ або } T = D_k(S).$$

Конкретну процедуру відновлення інформації або розкриття шифру без знання секретних параметрів перетворень, називають **дешифруванням** (або **криптоаналітичною атакою**).

Також необхідно відмітити, що сам процес криптографічного перетворення даних може здійснюватися як програмно, так і апаратно. Апаратна реалізація відрізняється суттєво більшою вартістю, однак їй властиві й переваги: висока продуктивність, простота, захищеність і т.д.

Програмна реалізація більш практична, допускає відому гнучкість у використанні.

Алгоритми шифрування поділяються на дві великі групи:

1. Симетричне (традиційне шифрування).
2. Асиметричне (шифрування з відкритим ключем).

Шифрування методом перестановки. Шифр перестановки (перестановочний шифр) – це один з найпростіших видів блочного шифру, що часто використовується як приклад для наочного розуміння механізму роботи більш складних алгоритмів шифрування. Відноситься до групи симетричних криптографічних алгоритмів, легко зламується і не має майже ніякого застосування на практиці.

Реалізація шифру перестановки. Розглянемо принцип реалізації алгоритму простої перестановки на окремому прикладі. У загальному випадку текст вхідного повідомлення розбивається на блоки довжини m, де m – це довжина ключа. Нехай ключ в розглянутому шифрі має наступний вигляд:

1	2	3	4
2	4	1	3

У першому рядку таблиці вказані номери символів блоку по порядку, а у другому рядку вказані номери позицій, які повинні займати зазначені символи в зашифрованому блоці тексту.

Кодування здійснюється перестановкою букв. Таким чином, перший символ з вихідного блоку повинен бути переставлено на друге місце, другий на четверте, третій на перше, четвертий на третє. Дешифрування проводиться в зворотному порядку. На прикладі зазначеного ключа: другий символ з зашифрованого блоку ставимо на перше місце, четвертий на друге, перший на третє, третій на четверте.

При використанні будь-якого блочного шифру (шифр перестановки також не є винятком), може виникнути ситуація, коли текст не ділиться на рівні блоки довжини m, тобто залишок від ділення довжини тексту n на довжину ключа m не дорівнює нулю. У таких випадках довжину вихідного повідомлення збільшують на m – (n% m) символів, щоб воно ділилося на рівні блоки довжини m.

Відтворимо послідовність дій для реалізації шифру перестановки на програмному рівні за допомогою засобів мови C#. Загальна схема такого алгоритму наведена на рис. 1.

Пояснення. На початку в тілі класу оголошується змінна зі початковим значенням null під одновимірний масив, що містить ключ (другий рядок з таблиці ключа).

Далі слідують три версії перевантаженого методу *SetKey*, який зберігає в екземплярі класу значення ключа.

Перша версія методу (I) приймає на вхід рядок, в якому міститься ключ. Елементи ключа повинні бути відокремлені один від одного символом пробілу. За допомогою методу *Split* рядок поділяється на масив рядків.

Друга перевантажена версія методу *SetKey* (II) приймає на вхід масив ключа, елементи якого представлені рядками. Рядки конвертуються в цілі числа за допомогою статичного класу *Convert* і його методу *ToInt32()*.

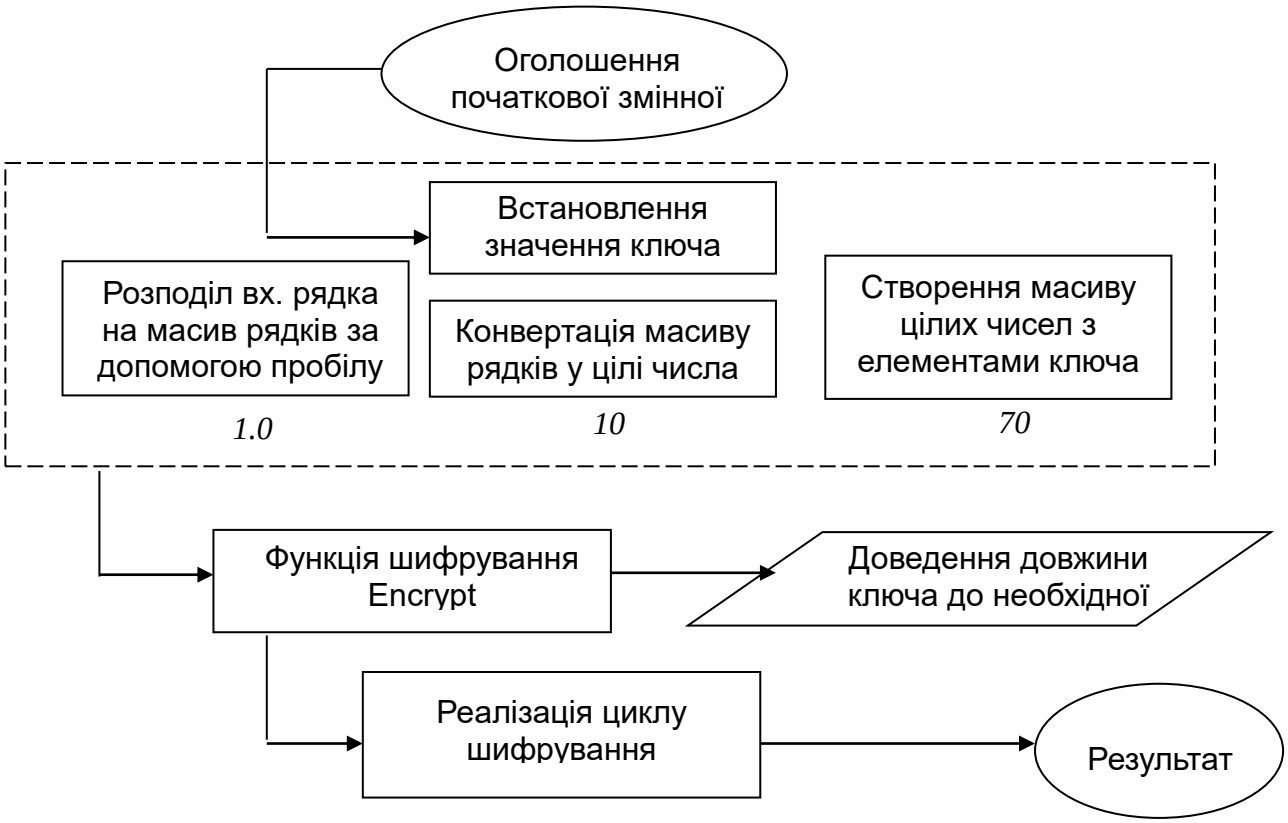

Рис. 1. Програмна реалізація алгоритму шифрування перестановкою

Третя версія (III) – вхідним аргументом є масив цілих чисел, у який копіюються елементи ключа. Необхідно зауважити, що посилання на масив ключа приходить в об'єкт класу ззовні, тому зовнішній код має можливість впливати на масив під час роботи самого класу. Тут найкращим варіантом вирішення проблеми є створення копії масиву з ключем для шифру перестановки.

Далі використовується метод *Encrypt*, який безпосередньо реалізує шифрування за допомогою перестановочного шифру. Аргумент методу – рядок з вихідним текстом.

На початку тіла методу довжина вхідного рядка доводиться до числа, яке без залишку ділитиметься на довжину ключа, за допомогою додавання в кінець рядка потрібної кількості символів з її початку.

Умова виходу з циклу хоч і не еквівалентна i <m – (n% m), але тим не менше працює і є більш функціональною – цикл триває до тих пір, поки довжина повідомлення не буде без остачі ділитися на довжину ключа. Коли поділ буде виконуватися без остачі (тобто i <0 дасть false), відбудеться вихід з циклу.

Після цього оголошується строкова змінна, в якій буде зберігатися результат – зашифрований текст:

Далі відбувається безпосередньо шифрування за допомогою циклу, що реалізує шифр перестановки. Розглянемо даний етап більш детально на конкретному фрагменті коду:

```
for (int i = 0; i < input.Length; i += key.Length)                    (1)
{                                                                     (2)
   char[] transposition = new char[key.Length];                       (3)
    for (int j = 0; j < key.Length; j++)                              (4)
      transposition[key[j] - 1] = input[i + j];                       (5)
   for (int j = 0; j < key.Length; j++)                               (6)
      result += transposition[j];                                     (7)
}                                                                     (8)
```

На кожній ітерації цього циклу створюється символьний масив *transposition* (рядок 3), в який заноситься зашифрований текст (переставлені символи) – символи в новому порядку, виходячи зі значення ключа.

Потім у внутрішньому циклі (рядки 4-5) перебираються всі символи блоку і відбувається безпосередньо шифрування (перестановка) за ключем (рядок 5).

Після цього в циклі (рядки 6-7) посимвольно зашифрований блок додається до кінця строкової змінної з результатом (рядок 7).

Метод дешифрування *Decrypt* за структурою ідентичний методу шифрування Encrypt, за винятком того, що в ньому відсутній код, який доводить вхідні рядок до потрібно довжини, оскільки зашифрований текст уже має необхідну довжину. І, звичайно, в даному методі відбувається дешифрування, тому механізм по перестановці символів інший:

transposition[j] = input[i + key[j] - 1];

На цьому шифрування перестановками завершено. В кінці з методу повертається результат шифрування/розшифрування.

Стандарт шифрування даних DES – блоковий шифр із симетричними ключами, розроблений Національним Інститутом Стандартів і Технології (NIST – National Institute of Standards and Technology) фахівцями фірми IBM, який вступив у дію в США 1977 року.

Розглянемо алгоритм реалізації шифру DES на уже знайомій мові C# в окремому проекті. Отож, щоб зашифрувати повідомлення алгоритмом DES, необхідно виконати наступну послідовність кроків:

1. довести вихідне повідомлення до такого розміру (в бітах), щоб воно без остачі поділялося на розмір блоку (128 біт);
2. розділити вихідне повідомлення на блоки;
3. довести довжину ключа до довжини половини блоку;
4. перевести ключ в бінарний формат (в нулі і одиниці);
5. провести над кожним блоком пряме перетворення мережею Фейстеля протягом 16-ти раундів. Після кожного раунду необхідно виконувати циклічний зсув ключа на задану кількість символів;
6. з'єднати всі блоки разом; таким чином отримаємо повідомлення, зашифроване алгоритмом DES.

Розшифровка DES проводиться за аналогією, при цьому використовується зворотне перетворення мережею Фейстеля.

Мережа Фейстеля використовується в алгоритмі DES для зашифровування (пряме перетворення мережею) та розшифрування (зворотне перетворення). Ці перетворення зображені на рис. 2 і 3 відповідно.

Розберемо один раунд прямого перетворення мережею Фейстеля. На і-й ітерації вихідний блок ділиться навпіл – ліва частина позначається L, права – R. Над R і ключем k_i обчислюється будь-яка обрана логічна функція f (ми будемо використовувати XOR). Потім виконується обчислення логічної операції над L і обчисленим раніше значенням функції (L xor f). Старе значення R переноситься в ліву частину блоку, а в праву частину заноситься значення L xor f. І остання операція раунду – потрібно виконати циклічний зсув ключа: $key_i + 1 = key_i >> Shift$ (при розшифровці $key_i - 1 = key_i << Shift$), де Shift – кількість символів, на яку необхідно циклічно зсунути ключ.

Через невелике число можливих ключів (всього 2^{56}) з'являється можливість їх повного перебору на швидкодіючій обчислювальній техніці за реальний час. У 1998 році Electronic Frontier Foundation, використовуючи спеціальний комп'ютер DES-Cracker, вдалося зламати DES за 3 дні. Згодом, щоб збільшити криптостійкість DES, з'являються кілька його модифікацій: double DES (2DES), triple DES (3DES), DESX, G-DES. В двох перших із них використовується збільшення довжини ключа до 112 та 168 біт відповідно, що безпосередньо впливає на стійкість проти злому для даного алгоритму [4].

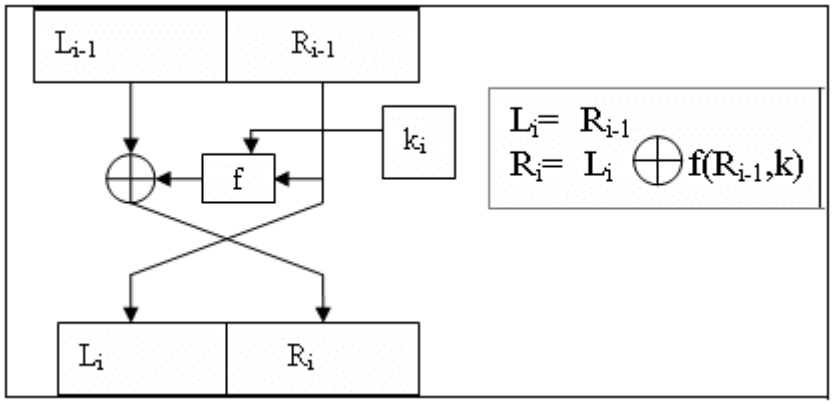

Рис. 2. Пряме перетворення мережею Фейстеля

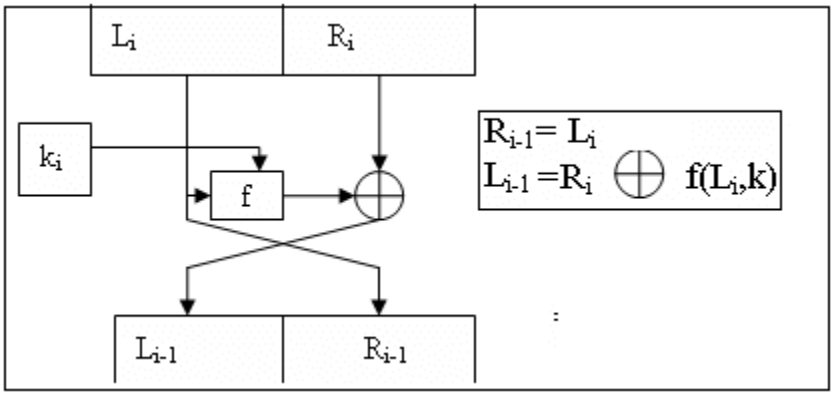

Рис. 3. Зворотне перетворення мережею Фейстеля

Стандарт шифрування даних AES (Rijndael) – симетричний алгоритм блочного шифрування (розмір блоку 128 біт, ключ 128/192/256 біт), прийнятий в якості стандарту шифрування урядом США за результатами конкурсу AES. Цей алгоритм добре проаналізований і зараз широко використовується, як це було з його попередником DES. Національний інститут стандартів і технологій США (NIST) опублікував специфікацію AES 26 листопада 2001 після п'ятирічного періоду, в ході якого були створені і оцінені 15 кандидатур. 26 травня 2002 року AES був оголошений стандартом шифрування. Станом на 2009 рік AES є одним з найпоширеніших алгоритмів симетричного шифрування [5].

Враховуючи таку поширеність даного алгоритму, в усіх високорівневих мовах програмування, що підтримують технології .NET Framework від Microsoft, є можливість використання готового простору імен System.Security.Cryptography. System.Security.Cryptography включає в себе перелік попередньо визначених класів, структур та інтерфейсів, що дозволяють реалізувати найпоширеніші типи алгоритмів шифрування на програмного рівні (в т.ч. DES та AES) без покрокового опису кожного етапу за допомогою циклів вручну. Нижче наведено фрагмент коду, який відповідає за реалізацію шифру AES на мові C#. При цьому на початку обов'язково необхідно під'єднати додаткову директиву *using System.Security.Cryptography.*

```
//Decrypt
private void buttonEncrypt_Click(object sender, EventArgs e)
{
if(radioButton3.Checked && textBox1.Text.Length > 0)
 {
    TripleDESCryptoServiceProvider triple = new TripleDESCryptoServiceProvider();
```

```
            UTF8Encoding u = new UTF8Encoding();
            MD5CryptoServiceProvider md5 = new MD5CryptoServiceProvider();
            triple.Key = md5.ComputeHash(u.GetBytes(textBox1.Text));
            triple.Mode = CipherMode.ECB;
            triple.Padding = PaddingMode.PKCS7;
            ICryptoTransform trans = triple.CreateEncryptor();
            encrypted = trans.TransformFinalBlock(u.GetBytes(inputTextBox.Text), 0,
u.GetBytes(inputTextBox.Text).Length);
            outputTextBox.Text = BitConverter.ToString(encrypted); }
             }
        //Encrypt
         private void buttonDecipher_Click(object sender, EventArgs e)
       {
        if(radioButton3.Checked && textBox1.Text.Length > 0)
          {
            TripleDESCryptoServiceProvider triple = new TripleDESCryptoServiceProvider();
            MD5CryptoServiceProvider md5 = new MD5CryptoServiceProvider();
            UTF8Encoding u = new UTF8Encoding();
            triple.Key = md5.ComputeHash(u.GetBytes(textBox1.Text));
            triple.Mode = CipherMode.ECB;
            triple.Padding = PaddingMode.PKCS7;
            ICryptoTransform trans = triple.CreateDecryptor();
            outputTextBox.Text = u.GetString( trans.TransformFinalBlock(encrypted, 0,
encrypted.Length));          }
        }
```

У червні 2003 року Агентство національної безпеки США постановило, що шифр AES є досить надійним, щоб використовувати його для захисту відомостей, що становлять державну таємницю (англ. Classified information). Для рівня SECRET було дозволено використовувати ключі довжиною 128 біт, для рівня TOP SECRET були потрібні ключі довжиною 192 і 256 біт [5].

Висновки. Таким чином, в статті розроблені та реалізовані на практиці алгоритми шифрування методом перестановки, DES та AES, два останніх з яких в повній мірі можуть забезпечити захист інформації від несанкціонованого доступу.

Перелік посилань

1. Задірака В.К., Кудін А.М., Людвиченко В.О., Олексюк О.С. Комп'ютерні технології криптографічного захисту інформації на спеціальних цифрових носіях: Навчальний посібник. Київ – Тернопіль: Підручники і посібники, 2007. – 272 с. www.hyade.com/base-pc/1281-978-966-07-1001-6.html

2. Гулак Г.М. Основи криптографічного захисту інформації: підручник / Г.М. Гулак, В.А. Мухачов, В.О. Хорошко, Ю.Є. Яремчук / – Вінниця : ВНТУ, 2011. – 199 с. (ISBN 978–966–641–430–7)

Електронні ресурси:

1. Герасименко Олена – Методи сучасної криптографії - WordPress.com https://herasymenko09blog.wordpress.com/
2. http://uk.wikipedia.org/wiki/Криптографія
3. http://uk.wikipedia.org/wiki/Симетричні_алгоритми_шифрування
4. https://uk.wikipedia.org/wiki/Data_Encryption_Standard
5. https://uk.wikipedia.org/wiki/Advanced_Encryption_Standard

Надійшла: 30.10.2019 Рецензент: д.т.н., доцент Гайдур Г.І.