Вісник ЛДУБЖД

Bulletin of Lviv State University of Life Safety

ISSN 2078-4643
https://journal.ldubgd.edu.ua/index.php/Visnuk
УДК 004.056.5:003.26:004.415.24
DOI: 10.32447/20784643.18.2018.04

А. Е. Лагун
Львівський державний університет безпеки життєдіяльності

ВИКОРИСТАННЯ СТЕГАНОГРАФІЧНИХ АЛГОРИТМІВ ДЛЯ ПРИХОВУВАННЯ ТЕКСТОВОЇ ІНФОРМАЦІЇ

Проблеми захисту інформації сьогодні надійно виконують сучасні криптографічні та стеганографічні системи. Криптографія захищає інформацію, перетворюючи її в незрозумілу форму завдяки використанню криптографічних алгоритмів і ключів. Стеганографія дає змогу приховати місцезнаходження секретної інформації.

В статті наведено результати досліджень стеганографічних алгоритмів, що приховують повідомлення у відкритому тексті.

Для процесу приховування використовують властивості текстового файлу-контейнера. Приховане повідомлення переводять у двійкову систему числення, а потім розставляють нулі та одиниці у відповідні місця текстового файлу за певними характеристиками.

Можливо два види приховування: вставки і заміни.

У випадку «вставки» приховане повідомлення додають до файлу-контейнера у вигляді символів, які невидимі під час перегляду текстового файлу. Тоді розмір заповненого контейнера буде більший за розмір порожнього контейнера.

При використанні «заміни» символи файлу-контейнера замінюють на інші символи, які майже не відрізняються від перших, наприклад, здійснюють заміну символів, які мають однаковий вигляд різними мовами. У цих випадках розміри порожнього і заповненого контейнера залишаються однаковими.

Розглянемо найпростіший спосіб приховування у вигляді вставки, який полягає у зміні кількості символів пропуску між словами текстового файлу. Нехай нулю відповідає один символ пропуску, а одиниці – два. Відповідно до вмісту прихованого тексту, один або два символи пропуску розміщуються у відкритому тексті.

Також автором розглянуто інший спосіб приховування, який використовує однаковий вигляд деяких символів алфавітів різних мов. Якщо розглянути символи української та англійської мов, то вигляд 18 символів кожної з цих мов є однаковим – 'a','c','e','i','o','p','x','A','B','C','E','H','I','K','O','P','T','X'. Під час приховування для значень нулів прихованого повідомлення початковий текст залишається таким самим, а для значень одиниць в тексті замінюють вказані символи на такі самі символи іншої мови. Одержані результати показують, що при використанні символів з різних мов для приховування потрібно файл-контейнер значно менших розмірів, ніж при кодуванні символів пропуску.

Ще один алгоритм використовує хвостові символи пропуску. Він формує заповнений контейнер із збільшеним розміром текстових рядків залежно від кількості символів пропуску прихованого повідомлення. Один символ прихованого повідомлення записується у два рядки текстового файлу. А саме двійкове подання кожного символу ділиться на дві частини по чотири біти, і в кінець кожного рядка записуються не більше 15 символів пропуску, кількість яких відповідає десятковому значенню кожної частини. Для забезпечення прихованості секретного повідомлення заповнений контейнер буде мати вигляд вирівняного за лівим краєм тексту.

Розглянуті алгоритми приховування повідомлення в текстовому контейнері можуть використовуватися для захисту конфіденційної інформації. Алгоритми, що використовують вставку невидимих символів, дають змогу приховати таку кількість інформації, яка відповідає кількості символів пропуску з певними характеристиками. Більшість алгоритмів заміни приховують набагато більше інформації, ніж алгоритми вставки, із збереженням величини порожнього контейнера. Зокрема, алгоритм, що замінює символи алфавіту однієї мови на символи алфавіту іншої, може приховати таку кількість інформації, яка визначається статистикою використаних мов.

Найбільшою проблемою використання текстових контейнерів є забезпечення їх стеганографічного захисту. Адже, якщо увімкнути режим перегляду недрукованих символів в текстовому редакторі, то якусь статистику розташування невидимих символів, доданих методами вставки, можна помітити і тоді декодувати приховане повідомлення. Приховане повідомлення за допомогою алгоритмів заміни є більш захищене, проте, якщо використати алгоритми стиснення до заповненого контейнера, то вся прихована інформація втратиться.

Ключові слова: стеганографічна система, прихована інформація, файл-контейнер, секретний ключ, хвостовий символ пропуску, вставка, заміна, алгоритм приховування, ASCII таблиця кодування, кількість інформації

Вступ

Сьогодні із розвитком обчислювальної техніки та систем передавання інформації питання захисту інформації є надзвичайно важливими. Цифрова інформація передається мобільними телефонами, комп'ютерними системами, соціальними мережами тощо. В усіх випадках необхідно забезпечувати захищеність інформації від зловмисників.

Завдання захисту інформації надійно виконують сучасні криптографічні та стеганографічні системи. Криптографія захищає інформацію, перетворюючи її в незрозумілу форму завдяки використанню криптографічних алгоритмів і ключів [1]. Стеганографія дає змогу приховати місцезнаходження секретної інформації, як правило, в об'єктах, які мають фізичну структуру. Такими об'єктами є нерухомі зображення, аудіо- та відео- файли. Отже, для криптографії відомо, що прихована інформація міститься в засекреченому повідомленні, а для стеганографії – не міститься.

У даній статті будуть розглянуті стеганографічні алгоритми, які дають змогу приховати текстові повідомлення в текстових файлах.

Огляд стеганографічних систем

Для формування прихованого каналу передавання інформації використовуються стеганографічні системи. В цих системах застосовують різні засоби і методи, що приховують таємну інформацію [2].

Вимоги до використання стеганографічних систем залежать від завдань комп'ютерної стеганографії. Вбудовування прихованої інформації виконують у випадках прихованої комунікації, захисту авторства з використанням цифрових водяних знаків та для тегування.

У випадку прихованої комунікації інформацію вбудовують у контейнер, відкрито передаючи його на приймальну сторону. Основною вимогою є непомітність прихованої інформації. Цифрові водяні знаки є невеликим обсягом інформації, яку вбудовують в контейнер з можливістю її подальшого виявлення. Основною вимогою є стійкість вбудованих даних до атак. При тегуванні приховане повідомлення вбудовують для зручності використання і воно містить корисну інформацію про вміст контейнера: заголовок, опис, розмір та інше.

Будь-яка стеганографічна система складається з таких елементів.

Приховане повідомлення, як правило, існує у цифровому вигляді і є послідовністю бітів і байтів. Основною вимогою при проектуванні стеганографічних систем є непомітність прихованої інформації. Приховане повідомлення вбудовується в контейнер – цифровий об'єкт, що має аналогову структуру (аудіо- файл, нерухоме зображення, відеопослідовність). Контейнери бувають порожніми (без вбудованого повідомлення) і заповненими. Заповнені контейнери коротко називають «стего».

Процес вбудовування відбувається за допомогою секретного ключа, який визначає місцезнаходження прихованого повідомлення в контейнері. Цей ключ використовується і на передавальній, і на приймальній стороні для вбудовування та видобування прихованого повідомлення. Структурну схему стеганографічної системи зображено на рис. 1 [3].

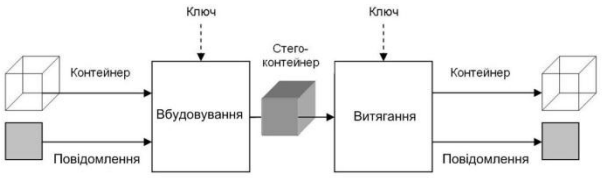

Рисунок 1 – Структурна схема стеганографічної системи

Дослідження стеганографічних алгоритмів, що приховують текстові повідомлення у відкритому тексті

Для приховування текстової інформації у текстовому файлі використовують властивості текстового файлу-контейнера. Приховане повідомлення переводять у двійкову систему числення, а потім розставляють нулі та одиниці у відповідні місця текстового файлу за певними характеристиками [5].

Можливо два види приховування: вставки і заміни.

У випадку «вставки» приховане повідомлення додають до файлу-контейнера у вигляді символів, які невидимі під час перегляду текстового файлу. Тоді розмір заповненого контейнера буде більший за розмір порожнього контейнера.

При використанні «заміни» символи файлу-контейнера замінюють на інші символи, які майже не відрізняються від перших, наприклад, здійснюють заміну символів, які мають однаковий вигляд різними мовами. У цих випадках розміри порожнього і заповненого контейнера залишаються однаковими.

Розглянемо найпростіший спосіб приховування у вигляді вставки, який полягає у зміні кількості символів пропуску між словами текстового файлу. Нехай нулю відповідає один символ пропуску, а одиниці – два. Відповідно до вмісту прихованого тексту, один або два символи пропуску розміщуються у відкритому тексті. Алгоритм приховування тексту наведено на рис. 2. На рис. 3 наведено порожній контейнер (test.txt) і заповнений контейнер з прихованим повідомленням «Стеганографія» (test2.txt).

В наведеному алгоритмі приховування відбувається з першого символу пропуску порожнього контейнера до закінчення бітів прихованого повідомлення. Тому ключем є перший символ пропуску контейнера. Алгоритм можна ускладнити, якщо для приховування задати ключ у вигляді двійкової псевдовипадкової послідовності. Тоді біти прихованого повідомлення розміщуються у символах пропуску контейнера, якщо, наприклад, біт ключа дорівнює одиниці, а якщо біт ключа – нуль, то символ пропуску контейнера ігнорується.

Розглянемо інший спосіб приховування, який використовує однаковий вигляд деяких символів алфавітів різних мов. Структуру алгоритму зображено на рис. 4, а результати роботи – на рис. 5.

Якщо розглянути символи української та англійської мов (змінна ***Letters***), то вигляд 18 символів кожної з цих мов є однаковим
'а','с','е','і','о','р','х','А','В','С','Е','Н','І','К','О','Р','Т','Х' **.**

Якщо потрібно приховати одиницю то відбувається заміна знайденого символу контейнера на символ іншої абетки (наприклад, символ ‘х’ української абетки (ASCII код 245) замінюється на символ ‘х’ англійської абетки (ASCII код 120)), а якщо приховується нуль, то ніяких замін не виконують.

З одержаних результатів випливає, що при використанні символів з різних мов для приховування потрібно контейнер значно менших розмірів, ніж при кодуванні символів пропуску.

Ще одним способом приховування інформації в текстовому файлі є метод вставки із використанням хвостових символів пропуску в кінці кожного рядка символів. Символи прихованого повідомлення кодуються аналогічно до попередніх випадків – за ASCII таблицею і переводяться в двійкову систему числення.

Взагалі кажучи, у відформатованих за розмірами сторінки файлах з розширенням doc та docx можливий випадок, що для документа не включено перенесення символів пропуску в наступний рядок. Тоді, починаючи з певного символу пропуску, в кінець рядка можна вставити довільну кількість символів пропуску. Теоретично цих символів можна вставити 255, що буде відповідати одному символу прихованого повідомлення згідно з ASCII таблицею. Проте виникає питання доцільності такої операції.

Якщо розглядати файл-контейнер у текстовому форматі (розширення txt), то при використанні для читання, наприклад, програми «Блокнот» закінчення рядка буде з’являтися лише при використанні символу перенесення на наступний рядок (абзацу). В цьому випадку символи прихованого повідомлення будуть вставлятися лише в кінці абзацу. Тоді кількість прихованих символів дорівнює (при використанні в кінці абзацу до 255 символів) кількості абзаців контейнера, що є дуже малим обсягом прихованого повідомлення.

У роботі [4] було розглянуто алгоритм вбудовування символів пропуску в кінець кожного рядка текстового контейнера. В цьому алгоритмі потрібно попередньо сформувати порожній файл-контейнер.

Було розроблено алгоритм, який формує заповнений контейнер з коригуванням розміру текстового рядка, залежно від кількості символів пропуску прихованого повідомлення. У цьому алгоритмі один символ повідомлення приховується у два рядки текстового файлу. А саме двійкове подання кожного символу ділиться на дві частини по чотири біти, і в кінець кожного рядка записуються не більше 15 символів пропуску, кількість яких відповідає десятковому значенню кожної частини (числу $1011_{(2)}$ відповідає 11 символів пропуску).

Для забезпечення прихованості секретного повідомлення заповнений контейнер буде мати вигляд вирівняного за лівим краєм тексту. Структуру алгоритму приховування зображено на рис. 6.

Алгоритм працює, доки у файлі-контейнері існує текст, оскільки після закінчення приховуваного повідомлення для збереження невидимості контейнер буде далі модифікуватися як вирівняний за лівим краєм текст.

Під час обробки кожного прихованого символу його ASCII код перетворюється в двійкову систему числення, а потім ці 8 біт поділяються на дві частини, кожен з 4 біт яких перетворюється в десяткову систему числення (значення ***n***).

Вважаємо, що довжина рядка становить 75 символів (зрозуміло, що ця довжина визначається типом вибраного шрифту). Потім замість останнього символу пропуску, який розташований на місці, меншому ніж 75-***n***, дописуються ***n*** символів пропуску. Після останнього вставленого символу пропуску дописують символ перенесення на наступний рядок. Символи контейнера, які розташовані після останнього символу пропуску в рядку запам’ятовуються в тимчасовій змінній, і вставляються у новий файл після символу перенесення. Далі процес продовжується аналогічно для другої частини з 4 біт приховуваного символу.

Якщо у файлі контейнері трапляється символ перенесення на наступний рядок, то замість нього у вихідний файл спочатку вставляється частина символів пропуску приховуваного символу, а потім – символ перенесення на наступний рядок.

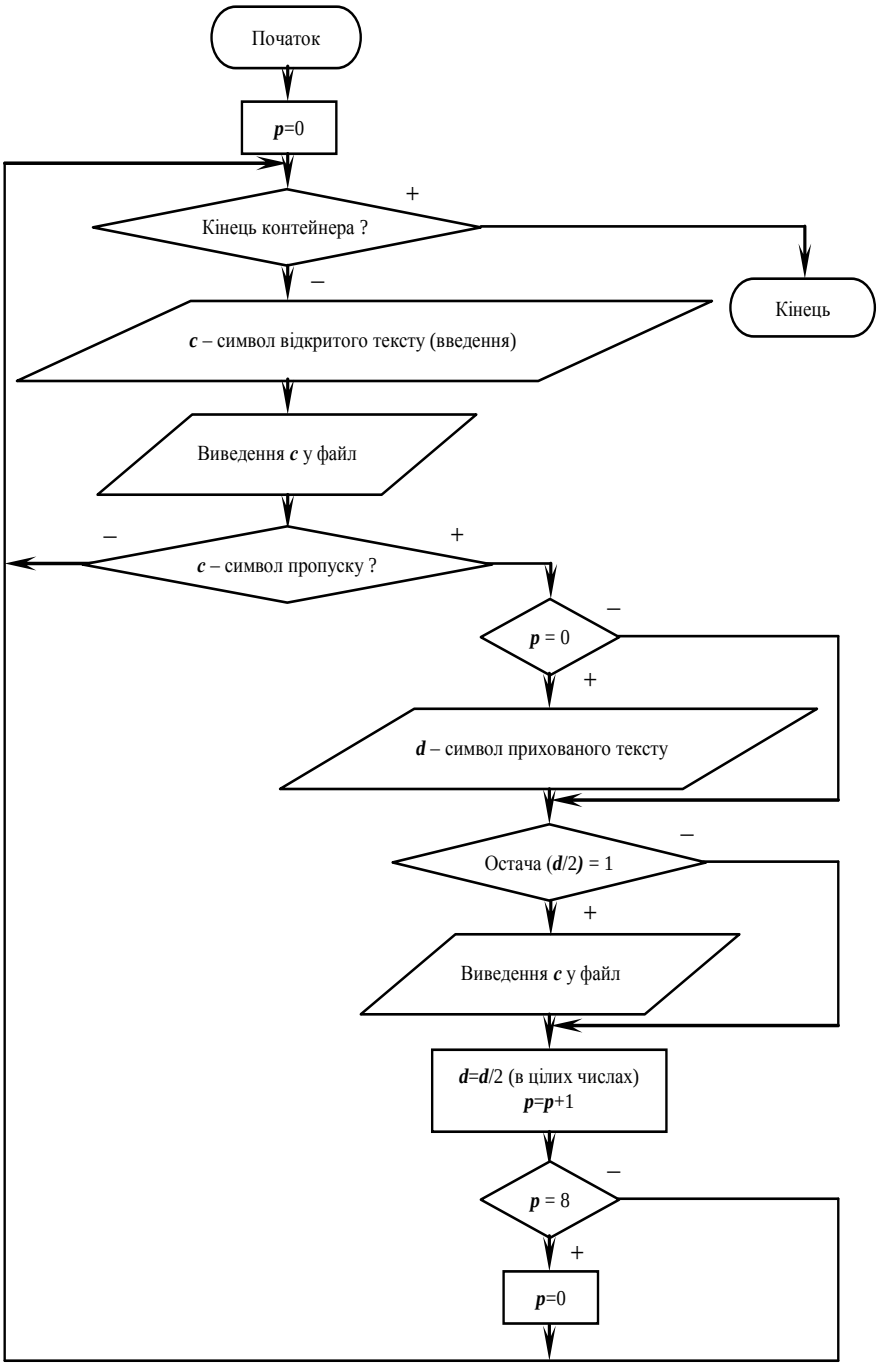

Рисунок 2 – Приховування з використанням символів пропуску

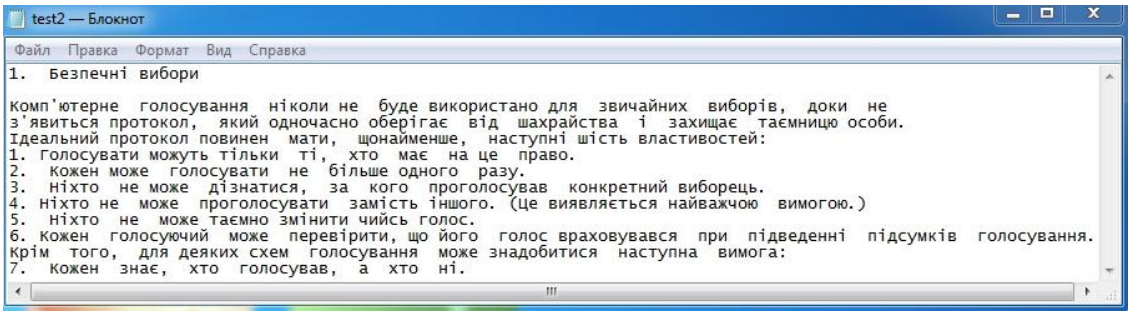

Рисунок 3 – Початковий текст і результат приховування

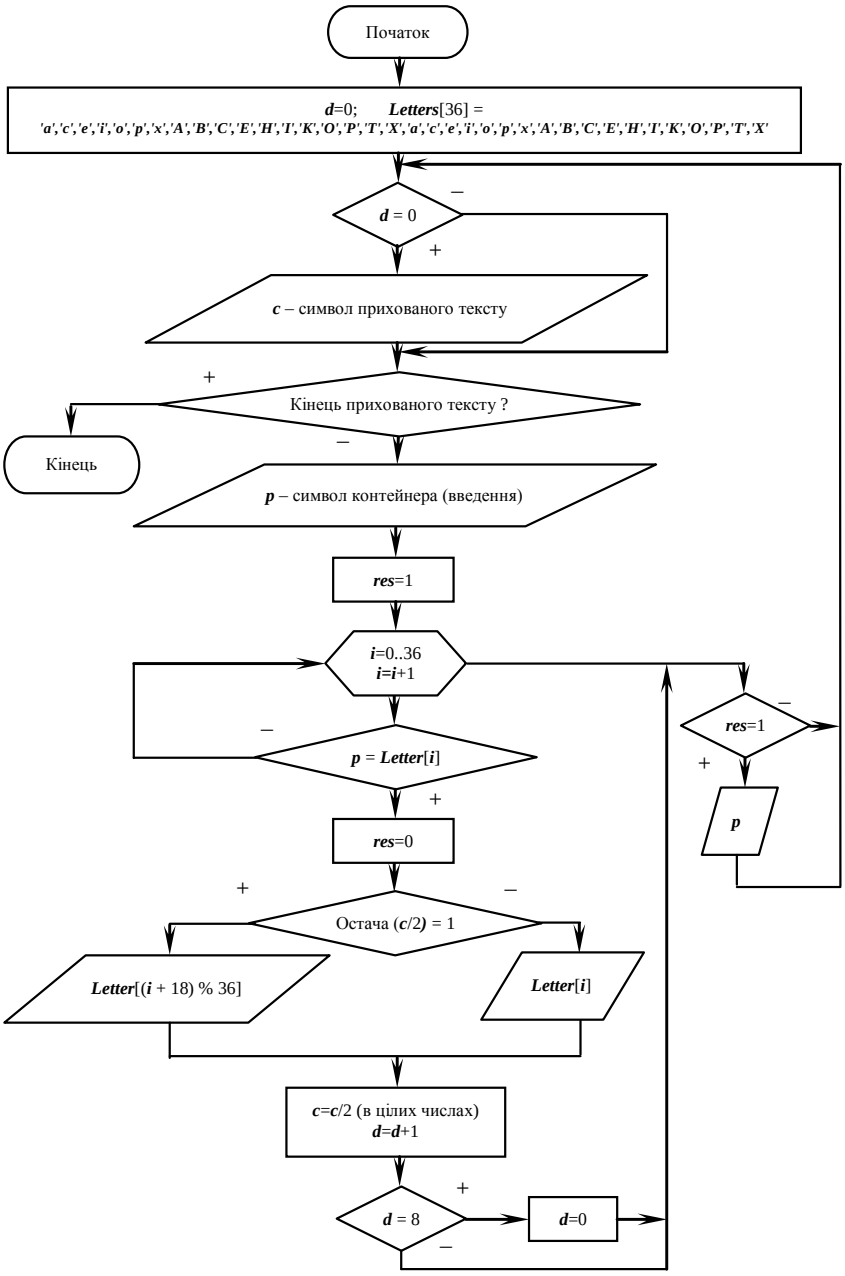

Рисунок 4 – Приховування з використанням символів різних мов

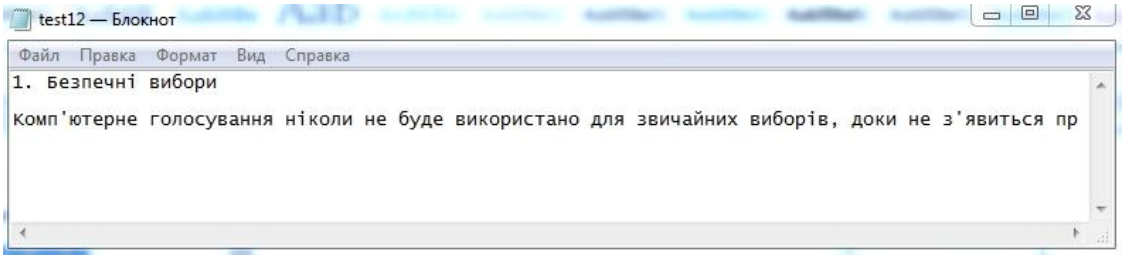

Рисунок 5 – Результат приховування повідомлення «Стеганографія»

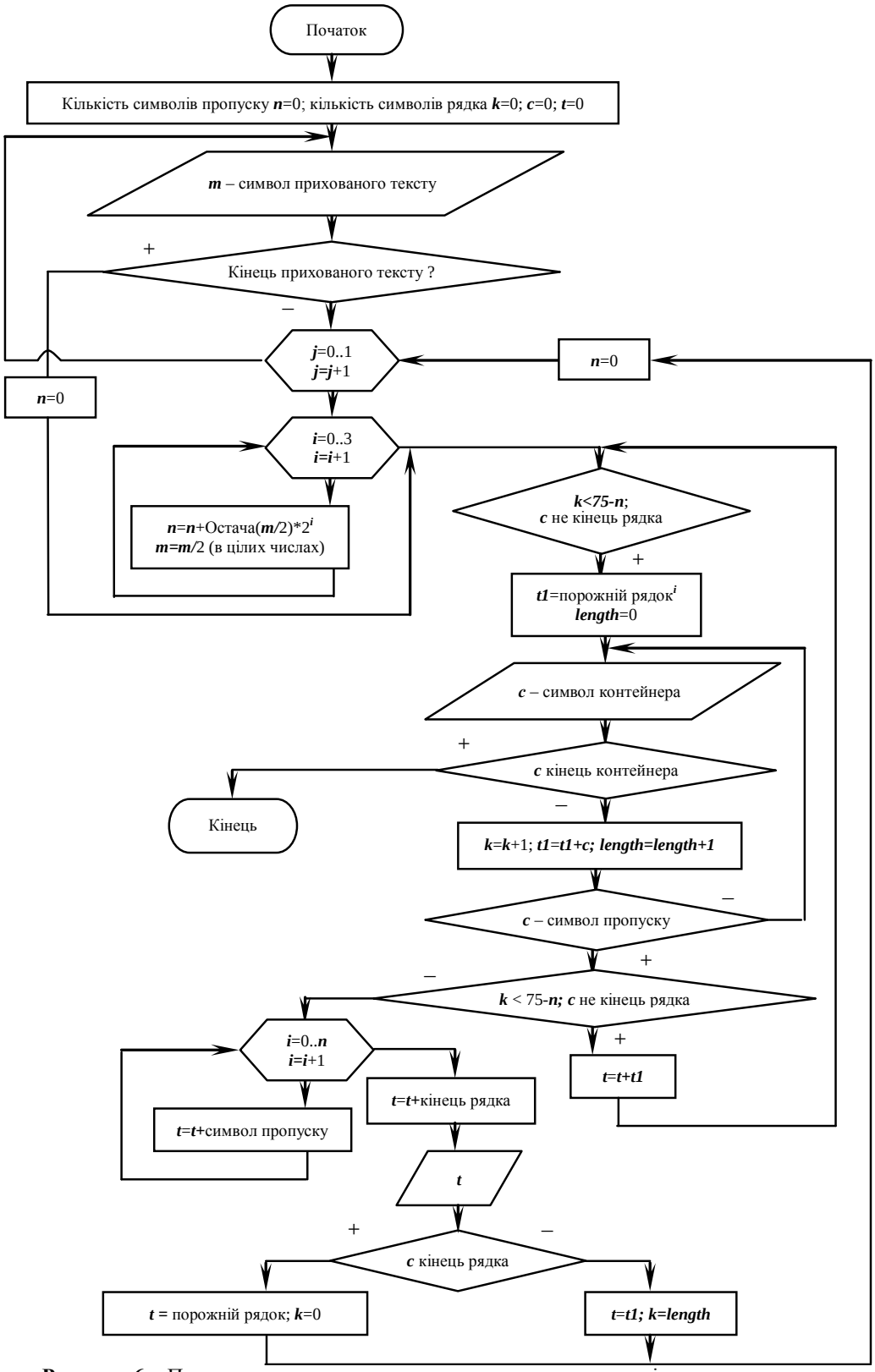

Рисунок 6 – Приховування з використанням хвостових символів пропуску

Таким чином, після закінчення алгоритму приховування з використанням хвостових символів пропуску, у заповненому контейнері в кінці двох рядків контейнера буде приховано один символ повідомлення. Зрозуміло, що кількість рядків контейнера має бути більшою за величину приховуваного повідомлення. Результат роботи алгоритму для повідомлення «Стеганографія» наведено на рис. 7. Наприклад, літера «***C***» в ASCII таблиці має код $209=11010001_{(2)}$, що під час приховування буде відповідати 1 символу пропуску (0001) в кінці першого рядка заповненого контейнера і 13 символам пропуску (1101) – в кінці другого рядка.

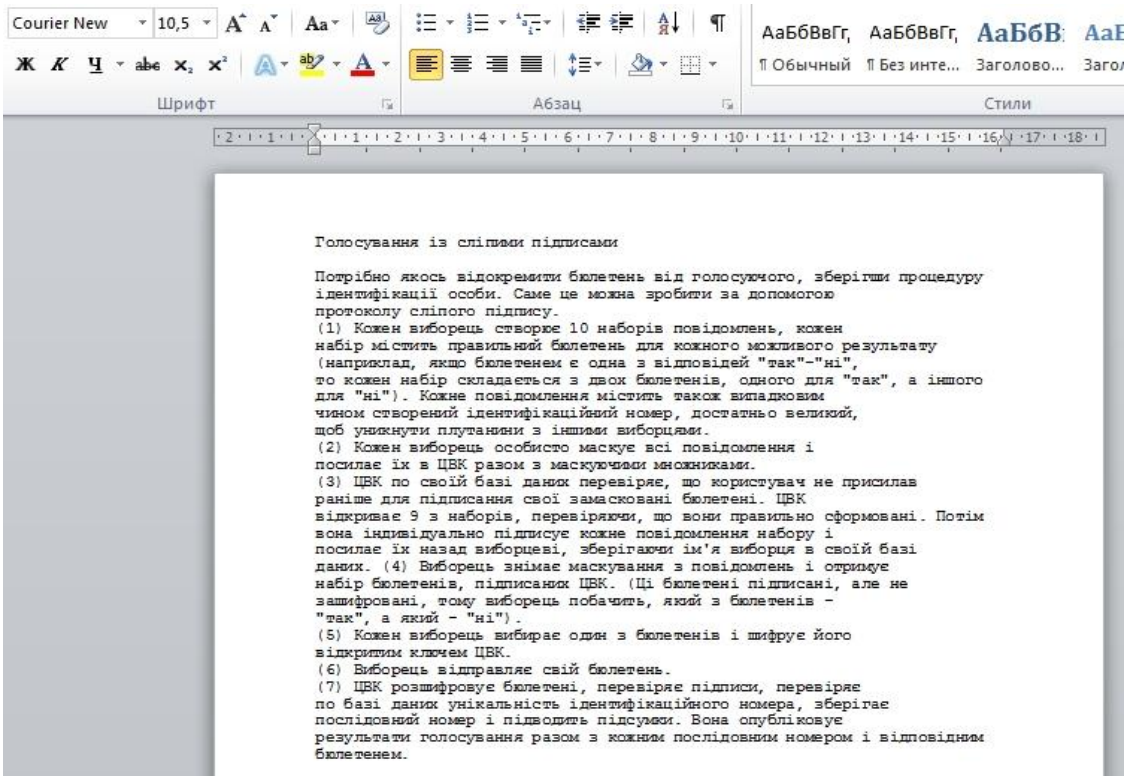

Рисунок 7 – Приховування тексту «Стеганографія» методом хвостових символів пропуску

Алгоритм приховування методом хвостових символів пропуску можна змінити, якщо в кінці кожного рядка вбудовувати один прихований символ за допомогою восьми недрукованих символів. Наприклад, двійкову одиницю замінювати на символ пропуску ‘ ’ (код 32), а двійковий нуль – на символ закінчення рядка ‘\0’ (код 10). Тоді в кінці кожного рядка заповненого контейнера буде міститися 8 невидимих символів, що визначатимуть двійковий код прихованого символу. У цьому алгоритмі кількість прихованих символів може бути в два рази більшою, ніж в попередньому алгоритмі. Проте кількість символів пропуску в алгоритмі приховування методом хвостових символів пропуску може змінюватися від 0 до 15, тобто є змінною.

Можливе також приховування, що використовує заміну, в якому форматування символів файлу-контейнера розглядають як бітову послідовність. Тоді форматування, яке відповідає нулю, візуально дуже близьке до форматування, що відповідає одиниці завдяки непомітності цієї різниці неозброєним оком. Наприклад, зміна палітри кольорів написання тексту (нуль – RGB(0, 0, 0), а одиниця RGB(1, 0, 0)), або зміна висоти літер тексту (нуль – 12 кеглів, а одиниця – 12,5). Останні алгоритми дають змогу приховати найбільше інформації в контейнерах, через те, що для приховування використовуються всі символи контейнера.

Висновки

Розглянуті алгоритми приховування повідомлення в текстовому контейнері можуть використовуватися для захисту конфіденційної інформації. Алгоритми, що використовують вставку невидимих символів, дають змогу приховати таку кількість інформації, яка відповідає кількості символів пропуску з певними ознаками. Більшість алгоритмів заміни дають змогу приховати набагато більше інформації із збереженням величини контейнера. Зокрема, алгоритм, що замінює символи алфавіту однієї мови на символи алфавіту іншої, може приховати кількість інформації, що визначається статистикою використаних мов. Наприклад, для англійського та українського алфавіту однакове написання мають 18 символів.

Найбільшою проблемою використання текстових контейнерів є забезпечення їх стеганографічного захисту. Адже, якщо увімкнути режим перегляду недрукованих символів в текстовому редакторі, то певну статистику розташування невидимих символів доданих методами вставки можна помітити і тоді декодувати приховане повідомлення. Приховане повідомлення за допомогою методів заміни є більш захищене, проте, якщо використати алгоритми стиснення до заповненого контейнера, то вся прихована інформація втратиться.

Список літератури:

1) Schneier B. Applied Cryptography: Protocols, Algorithms, and Source Code in C. 2nd ed. / B. Schneier. – New York : John Wiley and Sons, 1996.

2) Lagun A. Embedding of the hidden information with the use of Discrete Fourier Transform [Electronic resource] / A. Lagun, N. Kukharska //

Automatic Control and Information Technology (ICACIT'17) : 4th International Conference, 14-16 December 2017 : Proceedings. – Cracow, 2017. – 1 electr. opt. disk (CD-ROM).

3) Конахович Г. Ф. Компьютерная стеганография. Теория и практика / Г. Ф. Конахович, А. Ю Пузыренко. – К. : МК-Пресс, 2006. – 249 с.

4) Кухарська Н. П. Аналіз стеганографічних методів довільного інтервалу / Кухарська Н. П. – Вісник ЛДУ БЖД. – 2016. – № 14. – С. 7-16.

5) Иванов В. Текстовая стеганография: метод двойных пробелов между словами [Электронный ресурс] / Иванов В. – 2012. – Режим доступа : http://www.nestego.ru/2012/05/blog-post_12.html.

References:

1) Schneier, B. (1996). *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. John Wiley and Sons, New York, (USA).

2) Lagun, A. (2017). "Embedding of the hidden information with the use of Discrete Fourier Transform". *Automatic Control and Information Technology (ICACIT'17) : 4th International Conference*, 14-16 December 2017, Cracow, available at http://icacit2017.weebly.com.

3) Konakhovych, G. F. and Puzyrenko, A. Yu. (2006). *Computer steganography*. MK-PRESS, Kyiv (in Russ.)

4) Kukharska, N. P. (2016). Analysis of steganography methods of random interval. *Bulletin of Lviv State University of Life Safety, 14*, 7-16 (in Ukr.)

5) Ivanov, V. (2012). *Text steganography: the method of double spaces between word*. Retrieved from http://www.nestego.ru/2012/05/blog-post_12.html (in Russ.)

A. E. Lagun

USAGE OF THE STEGANOGRAPHIC ALGORITHMS FOR TEXT INFORMATION HIDING

Today cryptographic and steganographic systems provide the best information security of society. Cryptography transforms information into the incomprehensible form with using the cryptographic keys and algorithms. Steganography hides the secret information in unknown place of object.

The steganographic algorithms, which hide message in text container, are researched in the article.

For process of hiding are used the text file-container properties. The hide message converts to the binary numbers system. User puts ones or zeros into the defined places of text file-container. These places have special characteristics.

There may be two types of hiding: insertion and replacement.

In case of insertion the hiding message adds to file-container with using invisible characters in viewing mode of text file. Then the size of full container with hided message is bigger than size of empty container.

If used the replacing method then the characters of file-container replace to other characters that are almost the same as the first ones. For example, anyone is possible replacement of characters that have the same appearance in different languages. In this case the sizes of the empty and filled container remain the same.

One of the simplest hiding methods is insertion the variable quantity of the space characters between words of text file. Suppose, that zero of hidden message is coded by one space character and one - is coded by two space characters. Therefore, depending on hidden message one or two space characters are located in different places of the text.

Also, the author considers another hiding type, which uses the same view of some characters of different languages. If you look at the characters in Ukrainian and English, than the 18 characters in the each language is the same – 'a','c','e','i','o','p','x','A','B','C','E','H','I','K','O','P','T','X'. When hiding for the values of zeros in hidden message the file-container remains the same, and for the values of ones in hidden message the characters of language file-container replace to the same characters of another language (Ukrainian-English). The results of the algorithm work show us, that when using characters from different languages in the hiding process, the full file-container is much smaller than when encoding the space characters.

The last algorithm which is considered in work uses tail space characters. It forms a filled container with enlarged text strings depending on the number of space characters which the hidden message determines. One character of hidden message is written in two file-container text strings. In particular the binary representation of each character is divided into two parts with four bits, and at the end of each text string is written no more than 15 space characters. The number of space characters corresponds to the decimal value of each part. To ensure hiding of secret message full container has the form aligned to the left edge of the text.

Considered algorithms of hiding message in text container are used for the confidential information defense. Algorithms, which use insertion of invisible characters, allow hiding the amount of information that corresponds to the number of space characters with certain characteristics. The most of replacement algorithms hide more information than insertion algorithms. Also replacement algorithms do not change file-container size. For example, algorithm, which replace characters of different alphabets, hides such amount of information, which depends on the statistics of used languages.

The most problem of using text containers is providing its steganographic defense. In particular, if user enables the unprintable character view in a text editor, then could see the some statistic of location invisible symbols added by the insertion methods. Therefore decoding of hidden message is simplified. The hidden message with using replacement algorithms is more defensible, but using of compression algorithms to the full container deletes the hidden information.

Keywords: steganographic system, hidden information, file-container, secret key, tail space symbol, insertion, replacement, hidden algorithm, ASCII table, amount of information