№ 470. – С. 149–155. 4. Макар В.М., Матвійків О.М. Н-адаптивне моделювання на основі методу скінченних елементів. Стратегія згущення і адаптивні сітки. Частина 2 // Вісник Нац. ун-ту "Львівська політехніка". – 2003. – № 471. – С. 94–100. 5. Макар В.М., Матвійків О.М. Н-адаптивне моделювання на основі методу скінченних елементів. Частина 3: Результати моделювання на прикладі задачі Ляме // Вісник Нац. ун-ту "Львівська політехніка". – 2004. – № 471. – С. 94–100. 6. Лехницкий С.Г. Теория упругости анизотропного тела. – М.: Наука, 1977. – 416 с. 7. Григоренко А.Я., Дыяк И.И., Макар В.М. Решение пространственной динамической задачи теории упругости для анизотропных тел // Прикл. механика. – 1998. – Т. 34, № 5. – С. 24–31. 8. Zienkiewicz O.C. and Zhu J.Z. A simple error estimator and adaptive procedure for practical engineering analysis // Int. J. Numer. Meth. Eng. – 1987. – V. 24. – P. 337–357. 9. Коваль В., Макар В., Савула Я. Генерація адаптивних сіток з чотирикутних елементів у скінченноелементному аналізі // Вісник Львівського національного університету ім.І.Франка, серія "Прикладна математика та інформатика", 2002. 10. Zhu J.Z., Zienkiewicz O.C., Hinton E., Wu J. A new approach to the development of automatic quadrilateral mesh generation // Int. J. Numer. Meth. Eng. – 1991. – V. 32. – P. 849–866.

УДК 004.051

Т. Коротєєва, Є. Яворський
Національний університет "Львівська політехніка"

РОЗРОБЛЕННЯ ЕФЕКТИВНОГО ФОРМАТУ ДЛЯ СУМІСНОГО ВИКОРИСТАННЯ З LSB-МЕТОДОМ СТЕГАНОГРАФІЧНОГО КОДУВАННЯ У BMP-КОНТЕЙНЕРИ

Розглянуто практичний аспект створення наповнених стегоконтейнерів з використанням розробленого формату даних, які вони містять. Проаналізовано витрати часу на генерування даних у найменш впливових бітах зображення.

Ключові слова: стеганографія, зображення, біт, кодування.

The practical side of creating filled stego-containers using the developed format of crypted data was reviewed. Time spent on generating data in least significant bits of image was analyzed.

Key words: steganographia, picture, bit, coding.

1. Аналіз наявних засобів прикладної стеганографії.

Потреба в прихованому обміні інформацією постає практично у всіх галузях життя, де використовується комп'ютерна обробка даних, особливо у комп'ютерних мережах та там, де дуже важлива та потрібна захищеність. Стеганографія – це метод приховування інформації, коли сам факт наявності інформації стає невидимим. Методом стеганографії можна зашифрувати будь-яку інформацію у мультимедійному форматі. Використання стеганографічного кодування дає змогу широко застосувати можливості комп'ютерних систем для такого збереження даних, коли про сам факт наявності прихованої інформації не буде відомо.

Сьогодні більшістю програмних засобів для стеганографічного кодування даних керують служби безпеки, що ускладнює доступ до них. З наявних вільних застосувань та програмних пакетів під керуванням ОС "Windows" можна виокремити два засоби: SteganoG та Проект «Зоря». Дослідження обох проектів показали, що необхідно розвивати цю проблематику, щоб вдосконалити можливості існуючих засобів. Розглянемо по черзі переваги та недоліки вказаних продуктів.

SteganoG.

Переваги:

– портативність;

– порівняно велика кількість налаштувань.

Недоліки:

– складність у користуванні;

– можливість кодувати лише файли; втрачається інформація про них – назва, розмір.

Проект «Зоря».

Переваги:

– велика кількість методів шифрування;

– велика кількість налаштувань;

– гнучкість.

Недоліки:

– велика кількість проблемних моментів – повідомлення про критичні помилки, відсутність доступу до певних елементів керування;

– складність, що полягає у використанні незрозумілих назв функцій програми та дуже глибокій ієрархії налаштувань;

– можливість кодувати лише файли; втрачається інформація про них – назва, розмір.

Отже, спільною проблемою є складність та відсутність форматування даних.

2. Призначення стеганографічного кодування та збереження інформації.

Найпоширенішими контейнерами для кодування завжди були зображення. Один з відомих алгоритмів кодування у таких контейнерах – алгоритм LSB (least significant bit – найменш впливовий біт)[2]. Суть методу полягає у тому, що при зміні молодшого біта у мультимедійних потоках для неозброєного ока ніяких змін не відбувається. Його застосування подамо у такий спосіб:

– перебираємо кожен піксел зображення, – наприклад, піксел № 1:

R = 5Ah = 90 = 01011010

G = 39h = 57 = 00111001

B = F4h = 244 = 11110100

– змінюємо у кожному значенні молодший біт за бажанням. Наприклад, щоб зберегти число «5» у цьому пікселі, переведемо його у двійковий код – «101» і відповідно змінимо значення каналів:

R = 5Bh = 91 = 0101101**1**

G = 38h = 56 = 0011100**0**

B = F5h = 245 = 1111010**1**

Колір зміниться на величину, яку не розпізнають традиційні засоби сканування:

$$\Delta = \frac{3}{255} = 0,0117647.$$

Приклад зображення до і після кодування наведено на рис. 1.

Рис. 1. Порівняння зображень до і після стегокодування

Розмір файла також не зміниться, тому що ніяка додаткова інформація не записується.

Враховуючи той факт, що ці зображення в форматі bmp йдуть суцільним потоком, у котрому кожен байт або не використовується, або є частиною піксела, то можна розширити доступний для зберігання об'єм інформації, використавши всі байти. Код при цьому значно спрощується. Для забезпечення універсальності символи кодуються в форматі UNICODE.

В мережі Інтернет великого поширення набули псевдостеганографічні програми. Наприклад, такі, після використання яких вихідний розмір файла змінюється або погіршується читабельність вихідного файла. А це суперечить основному правилу стеганографії – факт кодування не повинен бути помітним.

Враховуючи це правило, стає очевидним, що розмір інформації, яку кодують, повинен бути значно меншим, ніж розмір інформації-приймача. Тобто mp3-файл з розміром 5 МБ неможливо зберегти без втрат у зображенні розміром 5 МБ[1] (рис. 2).

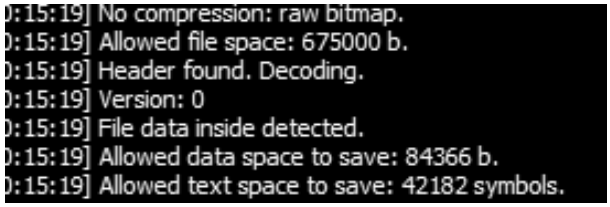


Рис. 2. Різниця у розмірі контейнера та стегоданих: розмір файла – 675000 байт, місце для кодування – 84366 байт

3. Опис алгоритмів

Вхідною інформацією для алгоритму кодування є файл із зображенням та текст, що буде використаний для кодування.

Основні кроки алгоритму STEGANOZ:

STEGANOZ_1: Завантажити вхідну інформацію.

STEGANOZ_2: Зчитати заголовок за версією WIN3.11. Визначити довжину зображення та кількість символів K, котрі можна зберегти у завантажене зображення. K = довжина/16. Встановити відповідне обмеження на поле введення.

STEGANOZ_3: У файлі для запису цикл від 1 до довжини тексту в буфері +1 для запису завершального нуля.

STEGANOZ_4: {Для поточного символу цикли від 1 до 16: { розбити на біти та занести у окремий масив.}

STEGANOZ_5: До кінця повторювати: {Зчитати 16 байт зображення та змінити молодший біт відповідно до масиву, сформованого в пункті **STEGANOZ_4**.}

Вхідною інформацією для алгоритму розкодування є файл із зображенням.

Основні кроки алгоритму STEGANOZU:

STEGANOZU_1: Завантажити вхідну інформацію.

STEGANOZU_2: Зчитати заголовок за версією WIN3.11. Визначити довжину зображення. Визначити кількість символів K, котрі можна зберегти у завантаженому зображенні. K = довжина/16. Встановити відповідне обмеження на поле введення.

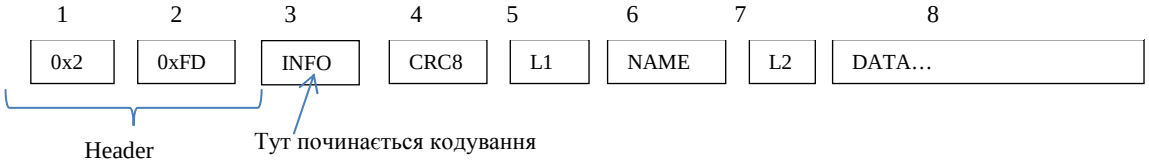
STEGANOZU_3: Цикл від 1 до K.

STEGANOZU_4: TEMP_BYTE = 0. Цикл від 1 до 16; {зчитати по одному байту, вибрати молодший біт і з 16 бітів сформувати 2-байтове число та завантажити його в TEMP_BYTE.

STEGANOZU_5: Якщо значення TEMP_BYTE = 0, то це означає кінець тексту. Вийти з циклу. В іншому разі додати символ у буфер.}

4. Опис формату даних та результатів

Щоб підвищити ефективність використання закодованих даних, розроблено такий формат збереження даних:



У таблиці розписано побітове представлення даних з назвою, розміром та детальним описом полів.

Опис формату

№	Назва поля	Розмір	Опис
1	Header1	1 б	Містить першу частину заголовка – число 0x2 (00000010b)
2	Header2	1 б	Містить другу частину заголовка – число 0xFD (11111101b). Разом з Header1 утворює константу SZ_HEADER_W = 0xFD02. Число вибране так, щоб максимально зменшити вірогідність його появи у некодованому контейнері [3].
3	Info	1 б	Байт інформації. Старші 4 біти – півзаголовок SZ_HEADER_N = 0x05 (0101b). Наступні два біти – версія формату. Поточна версія – 0 (00b). Перший біт (000000X0) містить булеве значення наявності парольного кодування даних. Нульовий біт (0000000X) містить «1», якщо дані – файл та «0», якщо дані – текст.
4	CRC8	1 б	З цього байта починається парольне кодування даних. Байт містить CRC8 контрольну суму даних у полі Data
5	NameLength	1 б	Містить довжину в байтах поля Name чи «0», якщо дані – текст.
6	Name	N б	Назва файла у форматі UNICODE 16 – 2 байти на символ. Без завершального нуля. Поле відсутнє, якщо дані – текст.
7	DataLength	4 б	Довжина власне даних.
8	Data	X б	Дані. Якщо це текст – завершується двома нулями (0x0000).

Отже, можна легко маніпулювати збереженою інформацією та надавати зручний доступ для читання, автоматично визначаючи, які дії можна виконати над збереженими даними [4].

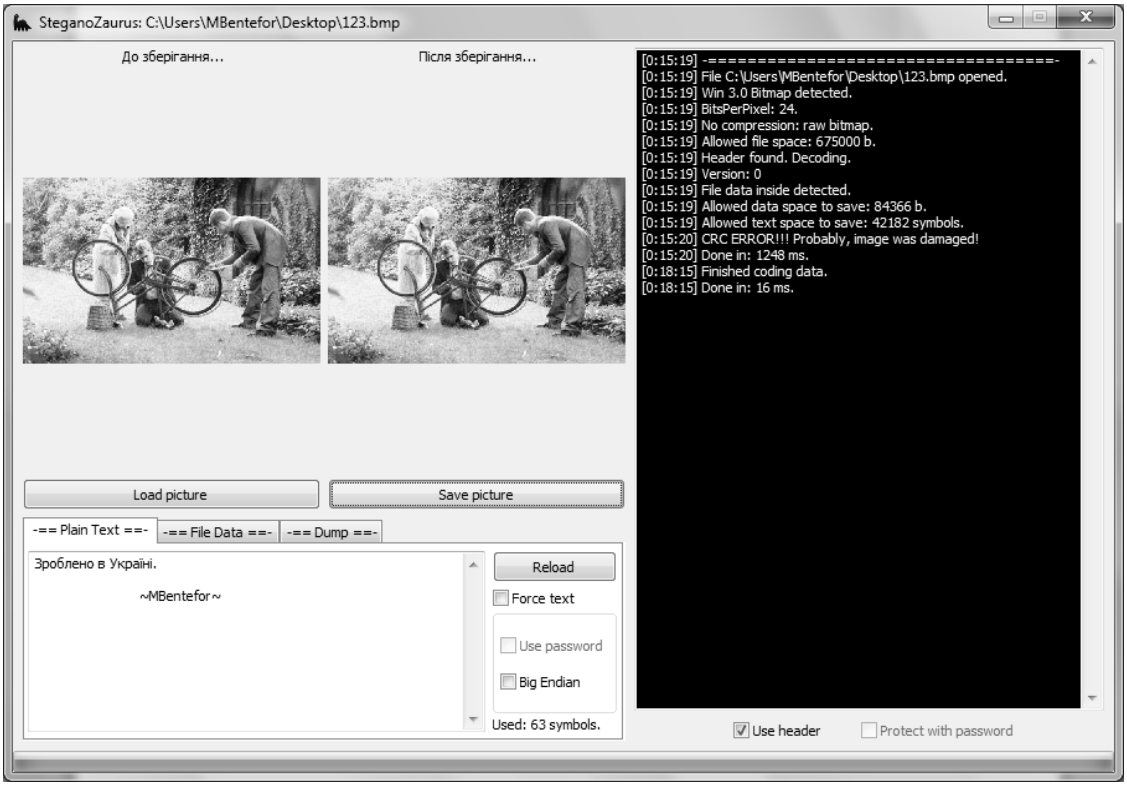


Рис. 3. Зовнішній вигляд ПП «SteganoZaurus»

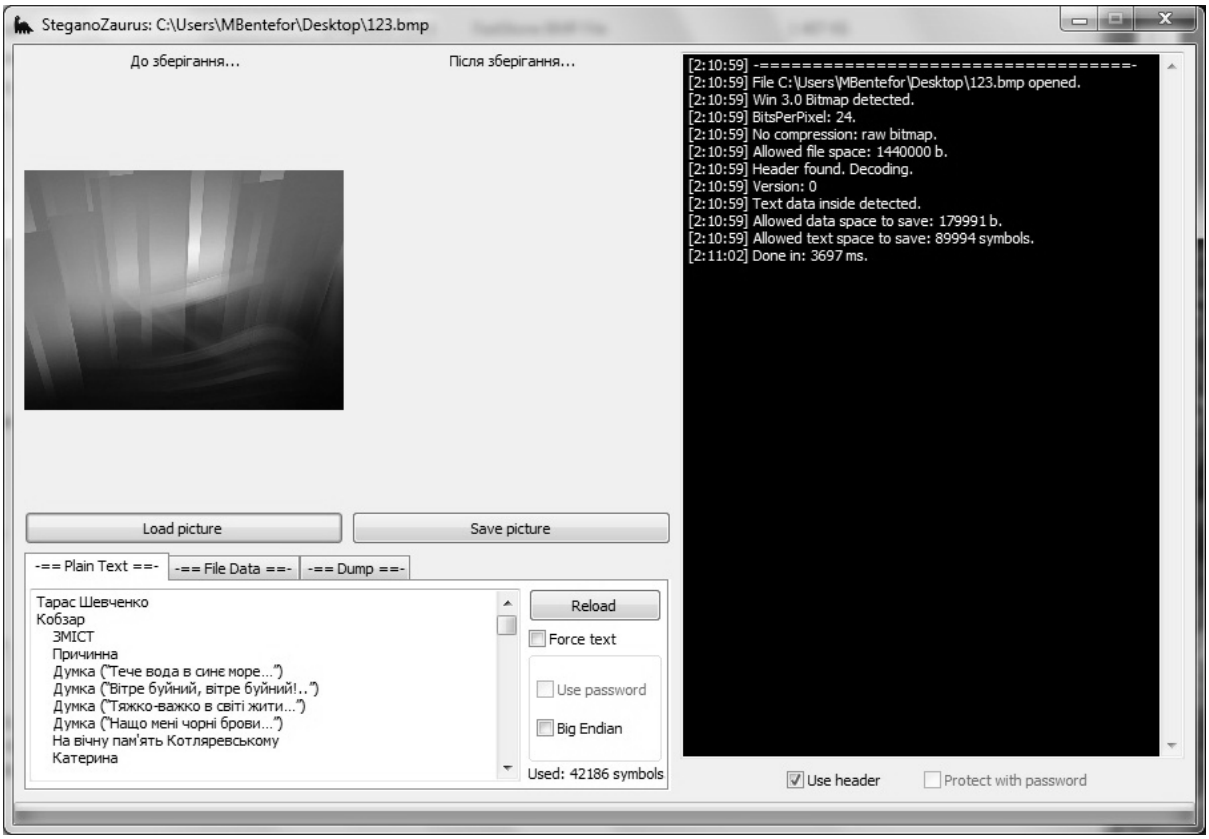


Рис. 4. Приклад кодування «Кобзаря» Т.Г. Шевченка

У результаті досліджень розроблено програму SteganoZaurus, котра в зрозумілому інтерфейсі дає змогу закодувати дані (текст чи файл) у зображення формату BMP та розкодувати їх. Програма надає можливість збереження дампу з файла (точної копії закодованої інформації, сукупності всіх молодших бітів) і ручного встановлення наявності заголовка. Чим більший розмір зображення, тим більше інформації можна у нього закодувати. Так, наприклад, якщо розмір зображення 600x375 пікселів, можна зберегти 42186 символів тексту. Для порівняння: кількість символів у «Кобзарі» Т.Г. Шевченка (1987 року видання) становить 600000, що дорівнює 300 сторінок тексту А4 формату. Отже, для його кодування потрібно лише 4 HD зображення.

На рис. 3 наведено приклад головного вікна розробленої програми «SteganoZaurus». Рис.4 ілюструє застосування програмного продукту на практиці – закодована у зображення частина «Кобзаря» Т.Г. Шевченка.

5. Висновки

Дослідження розробленого формату даних показали особливу зручність цього формату у практичному застосуванні в прикладних програмах. У подальших планах стосовно розвитку програми – можливість використовувати інші контейнери та парольний захист даних, також уникнути детектування наявності даних за допомогою аналізаторів.

1. Грибунин В.Г. Цифровая стеганография. – М. – 2008. – 272 с. 2. Пузыренко А.Ю. Конахович Г.Ф. Компьютерная стеганография. Теория и практика. – К.:МК-Пресс, 2006. – 286 с. 3. Cachin C. An Information-Theoretic model for Steganography. – Cambridge, 1998. 4.Bender W. – Techniques for data hiding // IMB Systems Journal. – 1996.