3. Аграновский А.В. Теоретические аспекты алгоритмов и классификации речевых сигналов / А.В.Аграновский, Д.А.Леднов. – М.: Радио и связь, 2004. – 164с.
4. Титов Ю.Н. Современные технологии распознавания речи / Ю.Н.Титов // Вестник ТГУ. – 2006. – Т.11. – Вип.4. –С.571-574.
5. Запрягаев С.А. Распознавание речевых сигналов / С.А.Запрягаев, А.Ю.Коновалов // Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии. – 2009. – №2. – С.37-46.
6. Малькова Е.С. Методы распознавания речи в задаче автоматизированного выявления дефектов произношения / Е.С.Малькова, О.А.Шабалина // Известия Волгоградского государственного технического университета. – 2015. – №2. – С.65-71.

Надійшла до редколегії 29.01.2018.

УДК 004.42 DOI 10.31319/2519-2884.32.2018.175

ДЕМЧЕНКО Ю.Ю., студент
БАБЕНКО М.В., к.т.н., доцент

Дніпровський державний технічний університет, м. Кам'янське

ВИКОРИСТАННЯ КОЛІРНОЇ МОДЕЛІ RGB ТА МЕТОДУ LSB ПРИ СТЕГАНОГРАФІЧНОМУ ЗАХИСТІ ІНФОРМАЦІЇ У ФАЙЛАХ ФОРМАТУ OFFICE OPEN XML

Вступ. Розглядаючи способи захисту інформації, доречно звернути увагу на стеганографічні методи. Сам термін «стеганографія» означає приховане повідомлення, яке повністю виключає можливість дізнатися про його існування третій особі. В якості сучасного прикладу можна привести випадок роздрукування контрактів з малопомітними викривленнями обрисів певних символів тексту на ЕОМ. Таким чином вносились дані про умови складання контракту, які необхідно було зашифрувати.

Комп'ютерна стеганографія ґрунтується на двох основних принципах [1]. По-перше, файли з оцифрованими зображеннями, а також аудіо- та відеофайли можна певною мірою змінити без втрати їх функціональності. По-друге, можливості людини розрізняти незначні зміни звуку або кольору досить обмежені. Стеганографічні методи дають можливість замінити несуттєві частки даних потрібною інформацією. Це означає, що сімейне фото може містити інформацію комерційного характеру, а файл з улюбленою мелодією – секретне повідомлення.

Проте найчастіше стеганографія застосовується для створення цифрових водяних знаків, які на відміну від звичайних можна виявити, лише використовуючи необхідне програмне забезпечення. Цифрові водяні знаки записуються у вигляді псевдовипадкових послідовностей сигналів шуму, які згенеровані на базі секретних ключів. Такі знаки забезпечують автентичність або недоторканість документа, дають можливість ідентифікувати власника або автора, перевірити права користувача або дистриб'ютора навіть в тому випадку, коли файл був спотворений або оброблений.

Щодо впровадження засобів програмно-технічного захисту в ІС, виділяють два головних способи:

1 – вбудований захист – механізми захисту розподілені за іншими компонентами системи або реалізуються у вигляді окремих складових ІС;

2 – додатковий захист – засоби захисту являють собою доповнення до основних

апаратних і програмних засобів комп'ютерної системи.

Другий спосіб є більш гнучким, його механізми при необхідності можна додавати та вилучати, але під час його реалізації можуть з'явитися проблеми забезпечення сумісності методів захисту між собою та з програмно-технічним комплексом інформаційної системи. Вбудований захист вважається більш надійним та оптимальним, але в той же час є жорстким, адже в нього складно внести зміни. Таким доповненням характеристик методів захисту зумовлюється те, що в реальній системі їх комбінують.

Існуючі алгоритми вбудовування секретної інформації поділяють на декілька груп:

1 – ті, які працюють з самим цифровим сигналом. До цієї групи відноситься метод LSB;

2 – «впаювання» таємної інформації. У цьому випадку відбувається накладення зображення, звуку або тексту, які необхідно приховати, поверх оригіналу. Досить часто застосовується для вбудовування ЦВЗ (цифровий водяний знак);

3 – використання можливостей файлових форматів. Сюди відноситься вкладення інформації в метадані або в інші зарезервовані поля файлу, які не використовуються.

За методом вбудовування інформації стеганографічні алгоритми поділяють на лінійні, нелінійні та інші.

Оскільки ми будемо вбудовувати приховані дані в текстовий файл, розглянемо методи, за допомогою яких це можна реалізувати [2].

1. Зміна регістру літер. Наприклад, нам треба сховати букву «А» в тексті «машина». Для цього ми беремо двійкове представлення коду символу «А» – «01010». Нехай для позначення біта, який містить одиницю, використовується символ верхнього регістру, а для нуля – нижнього. Результатом такого приховання буде «мАшИна». Закінчення «а» не використовується, так як для приховання цього символу потрібно було лише 5 біт, в той час як довжина рядка складає 6 символів. Таким чином вийшло, що остання літера – «зайва». Використовуючи такий підхід, можна сховати в текст довжиною N повідомлення з N/5 символів, що досить незручно.

2. Зміна кількості проміжків. Будемо вважати, що один проміжок відповідає біту «0», а два – «1». Програма отримує будь-який текст в якості контейнера і вкладає в нього повідомлення, замінюючи його біти на відповідну кількість проміжків. Важливу роль тут також відіграє і спосіб кодування символів. Треба отримати код символів оптимальної довжини, і щоб при цьому подвійний проміжок зустрічався якомога менше разів.

3. Line-shiftcoding. Змінюється відстань між рядками електронного тексту.

4. Word-shiftcoding. Змінюється відстань між словами тексту. Суть методу полягає в тому, що береться текст з різними відстанями між словами. Виділяється максимальна та мінімальна відстані, які позначаються відповідно 1 та 0, а інші відстані збільшують або зменшують до розмірів виділених. Окремим випадком цього методу являється метод зміни кількості проміжків, розглянутий вище.

5. Featurecoding. Внесення специфічних змін у шрифти окремих літер, наприклад, варіації довжини нижньої частини літери р.

Розглянуті вище методи досить легко вбудовуються в будь-який текст незалежно від його змісту, призначення та мови. Але, на жаль, такі методи легко зламуються, і секретна інформація може стати доступною третій особі. Також великим недоліком є те, що цими методами не можна передавати велику кількість прихованої інформації. Тому ми будемо використовувати інший метод, який має назву LSB.

LSB (Least Significant Bit, найменший значущий біт). Суть полягає в заміні найменш значущих бітів контейнера (аудіо-, відеофайл, зображення або текстовий файл) на біти повідомлення, яке необхідно приховати [3]. Оскільки можливості людського

ока розрізняти відтінки одного й того самого кольору досить обмежені, така заміна буде непомітною для людини. Саме на базі методу LSB і буде реалізовано алгоритм приховання таємної інформації, якому присвячена дана робота.

Постановка задачі. Засобами мови програмування C# розробити програмне забезпечення, за допомогою якого можна буде приховати таємну інформацію таким чином, щоб про її існування не дізнався будь-хто інший. Також необхідно забезпечити можливість витягнення секретного повідомлення з контейнера, в якому воно вже приховане. В якості контейнера (або сховища) для таємних даних ми будемо використовувати файл формату Office Open XML на прикладі документа Microsoft Office Word з розширенням docx. Чому саме docx, а не, наприклад, doc або txt? На це є декілька важливих причин. По-перше, файл з розширенням docx, на відміну від doc, являє собою zip-архів з XML-документами, який можна розпакувати та отримати всю необхідну інформацію: текст, зображення, таблиці тощо. Завдяки цьому досить легко вкладати та діставати приховані в нього дані. По-друге, docx – найбільш популярний і масовий формат, і його часте використання не буде викликати ні в кого сумнівів на предмет вкладених у ньому даних, що безсумнівно є великим плюсом у цій справі. По-третє, docx-файл важить значно менше, ніж його аналог з розширенням doc. Особливо ця різниця помітна в файлах, які містять велику кількість зображень або графіків. Для зберігання docx набагато зручніший, адже він займає мало місця на жорсткому диску.

Результати роботи. Як було вже сказано раніше, людське око не в змозі відрізнити незначні відтінки одного й того ж кольору. Цим можна вдало скористатися при побудові алгоритму вкладення таємних даних у контейнер. Суть цього алгоритму полягає в наступному. У нас є повідомлення, яке треба приховати в документ з розширенням docx. При цьому сам документ повинен вже містити у собі текстову інформацію. Від обсягу цієї інформації буде залежати обсяг тих даних, які ми зможемо в нього вкласти. Чим більше тексту містить документ, тим більше даних ми зможемо в нього приховати. Самі дані ми будемо вкладати в RGB канали кольору кожного текстового символу з цього файла. Для цього нам спочатку треба розпарсити документ Word, отримати з нього всі необхідні дані – текст та інформацію про кольори кожного з символів в форматі RGB. Потім отримані складові кольору потрібно перевести в двійкову систему числення і замінити молодші біти складових кольору бітами нашого повідомлення. Більш детально пояснимо це на прикладі:

Це 1 байт нашого повідомлення:

10 101 010

Це RGB кольори одного символу:

R: 11110**00**

G: 00001**000**

B: 11001**000**

Замінивши 2 молодших біта у каналі R та 3 молодших біти у каналах Gт а B, отримаємо наступний результат:

R: 11110**10**

G: 00001**101**

B: 11001**010**

Дана операція не внесе в колір помітних людському оку спотворень. Натомість вона допоможе нам вкласти рівно 1 байт нашого повідомлення в колір кожного символу вхідного файла. Тобто максимальна кількість байт (або символів), яку ми можемо приховати, буде дорівнювати кількості символів документу з розширенням docx, включаючи проміжки, табуляції, символи повернення каретки та переводу на новий рядок.

Аналогічним чином виконується і витягнення даних з контейнера. Для того, щоб отримати повідомлення, потрібно, як і в першому випадку, розпарсити документ Word,

отримати кольори текстових символів у форматі RGB і прочитати останні біти кожного каналу. Вони і будуть складати один байт (або символ) прихованих даних. Проробивши ці дії для всіх інших кольорів, ми отримаємо повністю текст секретного повідомлення.

На кафедрі «Програмне забезпечення систем» Дніпровського державного технічного університету (м. Кам'янське) розроблено програмний засіб, який реалізує вищеописаний алгоритм. Також у процесі роботи було написано власний парсер docx-документів, який на відміну від вже існуючих повністю задовольняє вимогам задачі і містить в собі лише необхідні функції, такі як зчитування тексту зі збереженням форматування, зчитування кольорів символів, які використовуються у файлі, і т. ін.

Сама структура додатку нескладна. На головній формі знаходиться 4 пункти меню. Нас будуть цікавити перші два з них: «Приховання даних» та «Витяг даних». Перший пункт відкриває форму, за допомогою якої можна вкласти дані в документ (рис.1).

Маємо декілька груп полів. У першій групі треба вибрати документ формату docx, в який ми збираємося приховати повідомлення, у другій – ввести його текст. Сам текст можна також імпортувати з іншого файла, який має розширення docx або txt. Для цього необхідно поставити відповідний чекбокс, а потім в діалоговому вікні вибрати потрібний нам документ. Зазначимо, що обсяг контейнера для вкладення повинен бути не менший, ніж кількість символів у самому повідомленні. В іншому випадку нам видасть відповідну помилку.

Другий пункт «Витяг даних» відкриває форму, за допомогою якої можна витягти вже приховані дані з контейнера (рис.2). Для цього нам необхідно вибрати файл з прихованим текстом і натиснути кнопку «Отримати повідомлення».

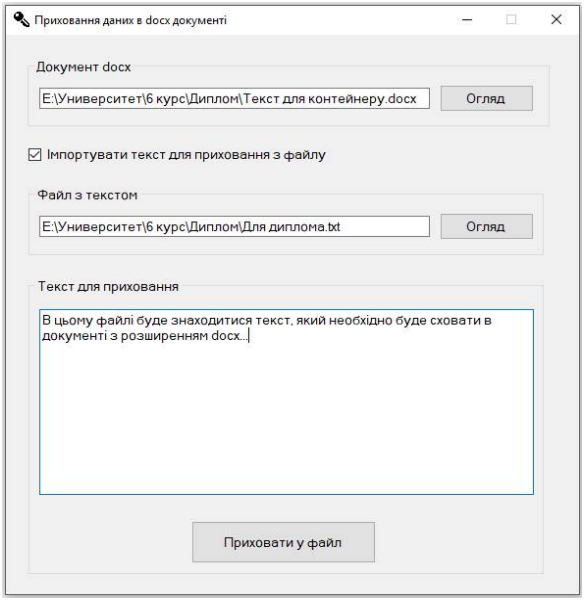

Рисунок 1 – Форма для вкладення даних у контейнер

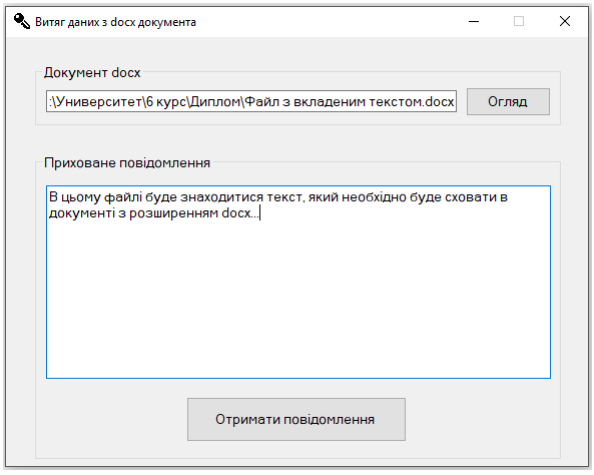

Рисунок 2 – Форма для витягнення даних з контейнера

Висновки. Оскільки даний проект орієнтований на сферу захисту даних, кожен, хто зацікавлений даною областю, отримає необхідний результат при використанні даного ПЗ. В процесі роботи розглянуто стеганографічні способи захисту інформації. На базі одного з них (метод LSB) з використанням колірної моделі RGB побудовано алгоритм вкладення прихованих даних в документ Microsoft Word з розширенням docx. Також засобами мови програмування C# було створено програмне забезпечення, яке повністю реалізує даний алгоритм.

Результати, отримані в цій роботі, будуть корисні у науково-технічній сфері та сфері захисту даних саме тому, що вони дадуть змогу як початківцям, так і професіоналам приховувати будь-які повідомлення без використання інших програмних засобів.

ЛІТЕРАТУРА

1. Конахович Г.Ф. Компьютерная стеганография. Теория и практика / Г.Ф.Конахович, А.Ю.Пузыренко. – К.: МК-Пресс, 2006. – 288с.
2. Грибунин В.Г. Цифровая стеганография / В.Г.Грибунин, И.Н.Оков, И.В.Туринцев. – М.: «Солон-Пресс», 2002. – 272с.
3. .Домарев В.В. Безопасность информационных технологий. Системный подход / В.В.Домарев. – К.: ООО "ТИД "ДС", 2004. – 992с.

Надійшла до редколегії 29.01.2018.

УДК 004.52

DOI 10.31319/2519-2884.32.2018.176

МІНЯЙЛО Я.О., студент
БАБЕНКО М.В., к.т.н., доцент
ЖУЛЬКОВСЬКИЙ О.О., к.т.н., доцент

Дніпровський державний технічний університет, м. Кам'янське

СТВОРЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОТРИМАННЯ НОТНОЇ ГРАМОТИ З МУЗИЧНИХ ФАЙЛІВ ФОРМАТУ MIDI

Вступ. Питання, пов'язані з програмуванням музики, «програмованою музикою», «музикою на основі розрахунків» тощо обговорюються досить тривалий час. Завдяки інтелектуалізації персональних комп'ютерів, наявності вбудованих систем аналітичних обчислень, великої кількості діалогових засобів роботи з табличними, текстовими, графічними, музичними об'єктами і т. д., а також у зв'язку з розвитком спеціального програмного забезпечення виникли реальні можливості синтезу композиції з теорією інформації, об'єднання музичних параметрів з акустичними за допомогою серійного комбінування.

Музичне програмування, яке передбачає детальне з'ясування нюансів уявлень про способи функціонування гармонії і тенденції її розвитку, зробило істотний внесок в розвиток сучасних уявлень про музичну гармонію. Дана робота присвячена вибору оптимального шляху перетворення музичних файлів у нотну грамоту завдяки аналізу музичних файлів формату MIDI за допомогою аспектів музикознавства, що допускають формалізований підхід з можливістю навчання гри на фортепіано будь-якої мелодії, що запущена за допомогою програми.

MIDI (англ. *Musical Instrument Digital Interface*, цифровий інтерфейс музичних інструментів) – стандарт передачі інформації між електронними музичними інструментами, розроблений 1983 року, що уможливлює комунікацію електромузичних інструментів, комп'ютера та іншого MIDI-сумісного обладнання, здійснювати з одного інструменту управління іншими.