УДК 004.72.056.52:003.27:004.438
DOI 10.31494/2412-9208-2021-1-2-203-210

EDUCATIONAL EXAMPLE OF INFORMATION MASKING IN THE ACOUSTIC SIGNAL

НАВЧАЛЬНИЙ ПРИКЛАД МАСКУВАННЯ ІНФОРМАЦІЇ В АКУСТИЧНОМУ СИГНАЛІ

Mykola HOLOVIN,
Candidate of Physical and Mathematical Sciences, Associate Professor

Микола ГОЛОВІН,
кандидат фізико-математичних наук, доцент

ninaholovina@gmail.com
https://orcid.org/0000-0003-4516-4677

Nina HOLOVINA,
Candidate of Physical and Mathematical Sciences, Associate Professor

Ніна ГОЛОВІНА,
кандидат фізико-математичних наук, доцент

ninaholovina@gmail.com
https://orcid.org/0000-0002-1152-1536

Volyn National University named after Lesya Ukrainka
✉ 13,Voli avenue., Lutsk, Volyn region

Волинський національний університет імені Лесі Українки
✉ Пр. Волі, 13, м. Луцьк, Волинської обл.

Original manuscript received: June 2, 2021
Revised manuscript accepted: September 15, 2021

ABSTRACT

The paper presents a steganographic method of hiding textual information in an audio file. Hiding is implemented by a program in Python. The introduction of individual letters of the text into the sound is carried out by the method of «the least significant bit». The program can be used for both educational and practical purposes. The commonly used wave library was used to work with sound files. It is not a library specialized for cryptographic and steganographic needs. Its use and the conciseness of the program code makes it possible to visualize the mechanism of hiding information in the classroom and demonstrate in the process of creating a program its debugging and testing. It is also important for educational purposes that working within the library allows you to see the state of an empty and filled audio container at the level of individual bits. To assess the practical value of the program, it was tested with texts of different lengths and with sound containers of different grades. In particular, the sound of a tuning fork, the sound of a guitar string, classical music, rap, jazz, and an audiobook were used. The experiment showed the correct reproduction of texts. It was found that if you listen carefully to the «pure sound» of the tuning fork, when the container is overloaded with information, suspicions of a text bookmark may arise. A text bookmark in the sound, in which the volume, tempo and frequency change quickly, does not reveal the suspicion of a possible bookmark. However, if the party who intercepted the masked message has guesses about how to bookmark the text, then the text is easily removed. Therefore, the use of the program for practical purposes requires additional manipulations in the code,

in particular related to the order of text input and the choice of location. Additional text encryption is also desirable. Analysis of sound and its manipulation at the level of individual bits also has educational value in the sense that it gives an idea of the noise level, the magnitude of the useful physical signal and the sensitivity of the human ear.

Key words: *Python language, steganography, hiding information, masking information in an audio file, educational example.*

Вступ. Глобалізація мережі робить інформацію легкою для передачі відкритими каналами в будь-яку точку Земної кулі. Однак, при цьому існує широкий спектр можливостей по перехопленню повідомлень та зламу шифрів. Тому існує необхідність передачі важливої інформації у прихованому, а краще і в зашифрованому вигляді. Актуальними є оригінальні способи приховування інформації як у візуальних (Головін, 2020), так і звукових контейнерах. Вивчення програмування як навчального предмета завжди має базуватись на актуальних тематиках. Такий підхід підвищує мотивацію студентів до навчання. Цікавими темами для створення різноманітних навчальних аплікацій при освоєнні роботи з масивами, файлами, строками є робота щодо кодування, шифрування та приховування інформації.

Праці (Грибунин, 2002; Конахович, 2006) є одними з перших системних видань у галузі стеганографії, у яких розглянуті відомі стеганографічні методи, що дозволяють приховувати конфіденційні дані у звукових та графічних комп'ютерних файлах. Тут системно викладені проблеми стійкості, пропускної здатності та надійності каналу прихованого обміну даними. Також тут представлені результати інформаційно-теоретичних досліджень стосовно проблем приховування інформації. Більш сучасні огляди проблеми приховування інформації представлені, зокрема, в роботах (Рябко, 2013; Конахович, 2018). *Метою статті* є реалізація засобами мови Python простого способу приховування текстової інформації у звукових файлах для використання цієї задачі в навчальних цілях.

Методи дослідження. Розглянемо реалізацію популярного алгоритму LSB (Least Significant Bit, найменший значущий біт) приховування «секретного» тексту у звуковому файлі (музика, пісня, аудіо книга). «Найменший значущий біт» – це біт на місці наймолодшого розряду у двійковому поданні числа. Суть цього стеганографічного методу полягає в заміні останніх значущих бітів байтів контейнера (зображення, аудіо чи відеозапису) на біти повідомлення, що приховується. Різниця між порожнім та заповненим контейнером не повинна бути відчутна для органів сприйняття людини. Наприклад, нехай десяткове число 130 показує миттєве значення рівня аудіосигналу. У двійковому коді це число має вигляд 10000010. Найменший значущий біт у цьому випадку дорівнює 0. Саме цей біт і змінюється на 1 або залишається 0, в залежності від поточного процесу впровадження інформаційної текстової закладки. Адже алгоритм LSB замінює молодший біт кожного байта звуку одним бітом із «секретного» повідомлення (Рис.1). Нехай значення аудіосигналу змінюються від 130

ICV 2019: 79.31
DOI 10.31494/2412-9208-2021-1-2

до 137, і саме на цьому фрагменті побітово впроваджується чергова буква з тексту закладки N, код якої $78_{10}= 01001110_2$.

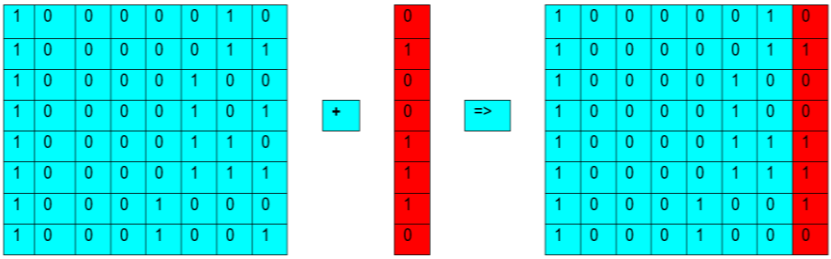

Рис. 1. Схема приховування даних в аудіо сигналі

Маніпуляції бітами в LSB, що показані на рисунку однокроковою дією «+», насправді досить прості, однак мають два етапи. На першому етапі між кожним байтом звукового контейнера і бітовою маскою 11111110 відбувається побітова логічна дія «AND», яка в мові Python позначається «&». Вона скидає на 0 всі найменші значущі біти байтів звукового контейнеру. На другому етапі між кожним модифікованим байтом контейнера і бітовою маскою 0000000[0/1] виконується логічна побітова операція «OR», де молодший біт байта контейнера може прийняти значення 0 або 1 із секретного повідомлення. Побітова логічна дія «OR» позначається в мові Python «|». Нижче представлений код програмного вбудовування текстового повідомлення в звук.

Програма та механізм її роботи. Програма розпочинається з під'єднання бібліотеки **wave** для роботи зі звуковими файлами. Далі завантажується звуковий файл (музика, аудіо книга, тощо) і текст, що має бути прихований у звуковому файлі. Надалі звуковий файл виконує роль контейнера для тексту.

```
import wave    # під'єднання бібліотеки роботи зі звуковими файлами wave
sound = wave.open("music.wav", mode='rb') # завантаження звукового файлу
file=open('secret_text.txt', 'r'); text=file.read(); print(text); file.close() #завантаження тексту
```

Після під'єднання бібліотеки і завантаження потрібних файлів у програмі відбуваються підготовчі дії, необхідні для успішного маскування тексту в звуковому файлі. Перетворенням піддається як звуковий файл, так і текстовий. Звуковий файл перетворюється в масив байтів sound_bytes. У кінець текстового файлу додається знак фінішу тексту '#' і далі текст text представляється у вигляді бітового масиву bits_text.

```
sound_bytes=bytearray(list(sound.readframes(sound.getnframes()))) # звуку в масив байтів
text=text+'#'                                  # додати в кінець тексту знак кінця тексту '#'
bits_text=list(map(int, ''.join([bin(ord(i)).lstrip('0b').rjust(8,'0') for i in text])))  # текст в масив бітів
```

Безпосереднє впровадження тексту як сукупності бітів у звуковий файл відбувається побітово по черзі в кожний окремий поточний звуковий байт. Зрозуміло, що алгоритмічно це організовано у вигляді циклу. Зупинка впровадження бітів тексту в звуковий файл після їх

закінчення реалізується відповідним розгалуженням. Описаний програмний фрагмент представлений нижче.

```
for i, bit in enumerate(bits_text):    # цикл впровадження бітів тексту в звуковий файл
    if i < len(text)*8 :               # зупинка впровадження бітів тексту в звуковий файл
        sound_bytes[i]=(sound_bytes[i]&254) | bit # ввод текстового біту в звуковий байт
```

Видно використання побітового оператора «&», що виконує роль логічної операції «AND», але в бітовій інформаційній розмірності. Аналогічно спрацьовує в бітовому просторі оператор «|». Його сенс – логічна дія «OR».

У виразі «sound_bytes[i]&254» число 254_{10} грає роль бітової маски 11111110_2, про яку йшлося вище. Адже $254_{10}= 11111110_2$.

Зрозуміло, що повне впровадження в звуковий байт одного текстового біту реалізується виразом «sound_bytes[i]=(sound_bytes[i]&254) | bit».

Можливість спостерігати процес приховування тексту побітово відкривається, якщо безпосередньо за розгалуженням вставити наступний рядок:

```
print(sound_bytes[i],sound_bytes[i]&254, bit, (sound_bytes[i]&254) | bit )
```

Ця можливість є досить цінною, якщо використовувати програму як аплікаційну на лекційному занятті при демонстрації її роботи вживу через проектор при візуалізації процесу приховування інформації в акустичному сигналі.

Таблиця 1

sound_bytes[i]	sound_bytes[i]&254	bit	(sound_bytes[i]&254) | bit	sound_bytes[i]	sound_bytes[i]&254	bit	(sound_bytes[i]&254) | bit	sound_bytes[i]	sound_bytes[i]&254	bit	(sound_bytes[i]&254) | bit
S				a				n			
99	98	0	98	88	88	0	88	131	130	0	130
105	104	1	105	113	112	1	113	16	16	1	17
197	196	0	196	184	184	1	185	65	64	1	65
118	118	1	119	96	96	0	96	241	240	0	240
185	184	0	184	12	12	0	12	234	234	1	235
124	124	0	124	74	74	0	74	210	210	1	211
224	224	1	225	190	190	0	190	101	100	1	101
122	122	1	123	46	46	1	47	183	182	0	182

ICV 2019: 79.31
DOI 10.31494/2412-9208-2021-1-2

У таблиці 1 представлений фрагмент роздруківки протоколу роботи програми із впровадження в цифрований звуковий сигнал тексту. «Santa Claus is coming to town». У таблиці представлено фрагмент з трьох букв «San». Запис модифікованих байтів, тобто байтів контейнера, що наповнений текстом, у новий звуковий файл «sound_txt.wave» реалізується наступним програмним фрагментом.

```
sound_modified = bytes(sound_bytes)
with wave.open('sound_txt.wav', 'wb') as fd:      # цикл запису модифікованих байтів файл
    fd.setparams(sound.getparams()); fd.writeframes(sound_modified)
sound.close()
```

Для вилучення тексту зі звуку необхідно запустити представлений нижче код. Як і попередня, ця програма починається із завантаження бібліотеки «wave» роботи зі звуком. Далі відбувається завантаження файлу звуку. Нагадаємо, що в цьому випадку звуковий контейнер заповнений прихованим у ньому текстом.

```
import wave
sound = wave.open("sound_txt.wav", mode='rb')              # завантажити звуковий файл
```

Після завантаження звукового файлу відбуваються підготовчі дії для подальшої його обробки. Звук у форматі «wave» трансформується в масив байтів.

```
sound_bytes = bytearray(list(sound.readframes(sound.getnframes())))
```

Далі проводиться безпосереднє вилучення всіх найменших значущих бітів зі звукових байтів з подальшою перспективою на об'єднання їх в коди букв.

```
extracted = [sound_bytes[i] & 1 for i in range(len(sound_bytes))]
```

Формування тексту реалізується рядком представленим нижче.

```
text = "".join(chr(int("".join(map(str,extracted[i:i+8])),2)) for i in range(0,len(extracted),8))
```

Далі відбувається вилучення з тексту знаку # - кінця тексту, роздруківка тексту на екрані. Це реалізується наступним програмним фрагментом.

```
decoded = text.split("#") [0]                                  # обрізати символ "#"
print("Успішно декодовано: "+decoded)                  #роздруківка вилученого тексту
sound.close()
```

Результати дослідження. Було проведено випробовування програм з текстами різної довжини та зі звуковими контейнерами різного ґатунку. Зокрема, використовувався звук камертону, звук гітарної струни, класична музика, реп, джаз, аудіокнига. Експеримент показав коректне відтворення текстів. Було виявлено, що при уважному прослуховуванні на «чистому звуку» камертону при заповненні найменших значущих бітів по всьому звуковому контейнеру можуть виникнути підозри про текстову закладку. Якщо ж заповнення неповне, фрагментарне і дуже проріджене, тобто не в кожний звуковий байт (а, наприклад, в кожний десятий байт), то закладка малочутлива. Текстова закладка у звук, в якому швидко змінюється гучність, темп і частота підозр про можливу закладку не виявляє.

Необхідно відмітити: якщо сторона, що перехватила замасковане повідомлення, має здогадки про факт закладки тексту в файл та спосіб закладки, то текст, що був прихований описаним вище способом, легко вилучається. Тому використання програми в практичних цілях вимагає додаткових маніпуляцій у коді, зокрема пов'язаних з порядком, щільністю впровадження тексту та з вибором місця впровадження. Бажаним є також додаткове шифруванням тексту хоча б простим методом. Таке шифрування можливо і окремою програмою.

Цінність представленої програми для навчальних цілей. При реалізації програми була застосована загальновживана бібліотека wave для роботи зі звуковими файлами. Це не є спеціалізована для криптографічних та стеганографічних потреб бібліотека. *Лаконічність коду програми* та використання бібліотеки wave дає можливість контрастно візуалізувати механізм приховування інформації на навчальних заняттях. Така демонстрація можлива на відповідному лекційному занятті або під час лабораторного заняття. Механізм приховування добре демонструвати наживу в процесі створення програми, її відлагодження та випробовування. Важливо для навчальних цілей і те, що робота в межах бібліотеки wave дозволяє на рівні окремих бітів побачити стан порожнього та заповненого звукового контейнеру. Аналіз звуку і маніпуляції з ним на рівні окремих бітів має також навчальну цінність у тому сенсі, що дає уявлення про рівень шумів та величину корисного фізичного сигналу й межі чутливості людського вуха.

Висновки. Реалізовано просту програму, що дозволяє приховувати текстову інформацію в звуковому файлі. Прослуховування заповненого і незаповненого звукового контейнера не виявляє відмінностей. Заповнений і порожній контейнери мають однаковий розмір. Програма цікава для навчальних цілей завдяки лаконічності і прозорості програмного коду. Механізми приховування інформації в акустичному сигналі легко візуалізуються. Програма може бути використана як аплікаційна на лекційному занятті при демонстрації її роботи вживу через проектор. Корисна ця задача й на лабораторному занятті. Аналіз звуку як фізичного сигналу на рівні окремих бітів, що відкривається при використанні програми, має також значну навчальну цінність. Маніпуляції зі звуком на такому низькому рівні дають уявлення про шуми сенсора при запису сигналу, амплітуду та частоту корисного фізичного сигналу, межі чутливості людського вуха та про ресурси приховування в сигналі сторонньої інформації. Використання програми в практичних цілях вимагає додаткових маніпуляцій у коді, зокрема, пов'язаних з порядком та з щільністю впровадження тексту, з вибором його місця розташування в файлі. Бажаним є також додаткове шифруванням тексту хоча б простим методом.

Література

Головін М. Б. Захист інформації стеганографічним способом мовою Python засобами графічної бібліотеки Pillow / М. Б. Головін, Н. А. Головіна, С. М. Яцюк, Ю. В. Сачук // Комп'ютерно-інтегровані технології: освіта, наука,

ICV 2019: 79.31
DOI 10.31494/2412-9208-2021-1-2

виробництво" Луцьк, 2020. Випуск № 40 с.110-115. URL: http://cit-journal.com.ua/index.php/cit/article/view/166 (дата звернення: 31.08.2021).

Грибунин В. Г., Оков И. Н., Туринцев И. В. Цифровая стеганография. – М. : Солон-Пресс, 2002. – 272 с.

Конахович Г. Ф., Пузыренко А. Ю. Компьютерная стеганография. Теория и практика. – К.: МК-Пресс, 2006. – 288 с.

Конахович Г. Ф., Прогонов Д. О., Пузиренко О. Ю. Комп'ютерна стеганографічна обробка й аналіз мультимедійних даних : [підручник]. – Київ : Центр навчальної літератури, 2018. – 558 с.

Рябко Б. Я., Фионов А. Н. Основы современной криптографии и стеганографии. – 2-е изд. – Москва : Горячая линия – Телеком, 2013. – 232 с.

References

Holovin, M.B., Holovina, N.A., Yaciuk, S.M., Sachuk, Yu.V. (2020). Zaxyst informaciyi steganografichnym sposobom movoyu Python zasobamy grafichnoyi biblioteky Pillow [Protection of information in a steganographic way in the Python language by means of the Pillow graphic library]. *Komp'yuterno-integrovani texnologiyi: osvita, nauka, vyrobnycztvo*. Luczk. Vypusk № 40. S. 110-115 [in Ukrainian].

Gribunin, V. G., Okov, I. N., Turincev, I. V. (2002). *Cyfrovaya steganografiya* [Digital steganography]. Moscow : Solon-Press. 272 s. [in Russian].

Konaxovich, G. F., Puzyrenko, A. Yu. (2006). *Komp`yuternaya steganografiya. Teoriya i praktika* [Computer steganography. Theory and practice]. Kyiv : MK-Press. 288 s. [in Russian].

Konaxovich, G. F., Progonov, D. O., Puzyrenko, O. Yu. (2018). *Komp'yuterna steganografichna obrobka i analiz mul`tymedijnyx danyx* [Computer steganographic processing and analysis of multimedia data] : [pidruchnyk]. Kyiv : «Centr navchal`noyi literatury». 558 s. [in Ukrainian].

Ryabko, B. Ya., Fionov, A. N. (2013). *Osnovy sovremennoj kriptografii i steganografii* [Fundamentals of modern cryptography and steganography]. 2-e izd. – Moscow : Goryachaya liniya – Telekom. 232 s. [in Russian].

АНОТАЦІЯ

У роботі представлений стеганографічний метод приховування текстової інформації в звуковому файлі. Приховування реалізовано програмою на мові Python. Упровадження окремих букв тексту в звук здійснено методом «найменшого значущого біта». Програма може бути використана як для навчальних, так і практичних цілей. Була застосована загальновживана бібліотека wave для роботи зі звуковими файлами. Це не є спеціалізована для криптографічних та стеганографічних потреб бібліотека. Використання її та лаконічність коду програми дає можливість візуалізувати механізм приховування інформації на навчальних заняттях і продемонструвати його в процесі створення програми, її відлагодження та випробовування. Важливо для навчальних цілей і те, що робота в межах бібліотеки дозволяє на рівні окремих бітів побачити стан порожнього та заповненого звукового контейнера. Для оцінки практичної цінності програми були проведені її випробовування з текстами різної довжини та зі звуковими контейнерами різного ґатунку. Зокрема, використали звук камертону, гітарної струни, класичної музики, репу, джазу, аудіокнигу. Експеримент показав коректне відтворення текстів. Було виявлено, що при уважному прослуховуванні на «чистому звуці» камертону при перевантаженні контейнера інформацією можуть виникнути підозри про текстову закладку. Текстова закладка в звук, у якому швидко змінюється

гучність, темп і частота, підозр про можливу закладку не виявляє. Однак, якщо сторона, що перехватила замасковане повідомлення, має здогадки про спосіб закладки тексту, то цей текст легко вилучається. Тому використання програми в практичних цілях вимагає додаткових маніпуляцій у коді, пов'язаних зокрема з порядком впровадження тексту та з вибором місця застосування. Бажаним є також додаткове шифруванням тексту. Аналіз звуку і маніпуляції з ним на рівні окремих бітів має також навчальну цінність у тому сенсі, що дає уявлення про рівень шумів, величину корисного фізичного сигналу та межі чутливості людського вуха.

Ключові слова: *мова Python, стеганографія, приховування інформації, маскування інформації в звуковому файлі, навчальний приклад.*

ICV 2019: 79.31
DOI 10.31494/2412-9208-2021-1-2