DOI: https://doi.org/10.53052/9788367652124.08

Michał HERCZYŃSKI[1]

Opiekun naukowy: Ruslana ZIUBINA[2]

PROJEKT I REALIZACJA PROGRAMU DO STEGANOGRAFICZNEGO PRZECHOWYWANIA DOKUMENTÓW O RÓŻNYCH FORMATACH W PLIKACH AUDIO

Streszczenie: Artykuł omawia znaczenie steganografii w komunikacji cyfrowej i dystrybucji treści cyfrowych. Autorzy prezentują oprogramowanie do ukrywania danych w plikach audio. Program oferuje prosty interfejs i pozwala na ukrycie oraz odczytanie danych. Testy wykazały dobrą pojemność kontenerów i jakość wbudowania danych. Oprogramowanie ma potencjał rozwoju.

Słowa kluczowe: Steganografia, Least Significant Bit, LSB, WAV, Audio, Python

DESIGN AND IMPLEMENTATION OF A PROGRAM FOR DIFFERENT FORMATS DOCUMENTS STEGANOGRAPHIC DATA HIDING IN AUDIO FILES

Summary: The article discusses the significance of steganography in digital communication and the distribution of digital content. The authors present software for hiding data in audio files. The program offers a simple interface and the ability to hide and retrieve data. Tests showed good container capacity and data embedding quality. The software has the potential for further development.

Keywords: Steganography, Least Significant Bit, LSB, WAV, Audio, Python

1. Wstęp

W obecnej rzeczywistości, coraz więcej komunikacji międzyludzkiej przenosi się do świata cyfrowego. W badaniach przeprowadzonych przez GUS w roku 2022, 69,3% ogółu Polaków w wieku 16-74 lat używa Internetu w celu korzystania z poczty elektronicznej, a nie wiele mniejsza część - 64,7% użytkuje przynajmniej jeden z komunikatorów internetowych. Są to czołowe pozycje wymienione w badaniu, a jedyną czynnością częściej deklarowaną przez społeczeństwo, jest wyszukiwanie informacji o towarach lub usługach.

[1] Uniwersytet Bielsko-Bialski, WBMiI, Informatyka specjalność: inżynieria oprogramowania, mherczynski1991@gmail.com
[2] dr inż., Uniwersytet Bielsko-Bialski, WBMiI, rziubina@ubb.edu.pl

Nie inaczej jest z rosnącą popularnością cyfrowej dystrybucji wytworów kultury takich jak muzyka, filmy, grafiki. W roku 2022 wyraźnie zwiększyło swą popularność oglądanie nagrań wideo z serwisów tworzonych przez użytkowników, jak i od komercyjnych usługodawców oraz słuchanie muzyki pobranej lub w formie streamingu. [1]
Wraz ze wzrostem tych trendów, naturalnie wzrasta również zagrożenie wymierzone w kierunku tych właśnie aktywności. Od prób podejrzenia, lub przechwycenia komunikacji pomiędzy rozmówcami, po dystrybucję nielegalnych kopii dóbr cyfrowych, co generuje ogromne straty finansowe.
Jedną z metod zabezpieczeń przed tymi zagrożeniami może być steganograficzne wbudowanie odpowiednich danych w plik docelowy, będący nośnikiem.
Steganografia jest nauką o prowadzeniu komunikacji w taki sposób, aby jej fakt pozostawał utajony przed stroną atakującą. Od kryptografii różni ją więc to, że w przypadku komunikacji szyfrowanej można zaobserwować przekazywanie szyfrogramów między osobami i jeśli nie dokonać złamania szyfru, to skutecznie taką komunikację zakłócać. W przypadku steganografii osoba lub organizacja atakująca, może mieć ogromny problem, żeby zidentyfikować alternatywne sposoby komunikacji i im przeciwdziałać.
Steganografia nie musi jednak dotyczyć jedynie komunikacji. Z jej pomocą można skutecznie znakować sprzedawane dobra cyfrowe m.in. muzykę, tworząc na nich swoisty znak wodny, który pozwoli na ewentualne określenie źródła nielegalnych dystrybucji w Internecie.
Aby to zrobić, można przed udostępnieniem do pobrania klientowi zakupionej kopii dobra cyfrowego, wbudować w nią indywidualnie wygenerowaną sygnaturę, pozwalającą na późniejszą identyfikację dla kogo ta konkretna kopia została wygenerowana. Sam klient nie powinien jej zauważyć i nie powinna ona wpływać na odbiór utworu.

2. Realizacja oprogramowania

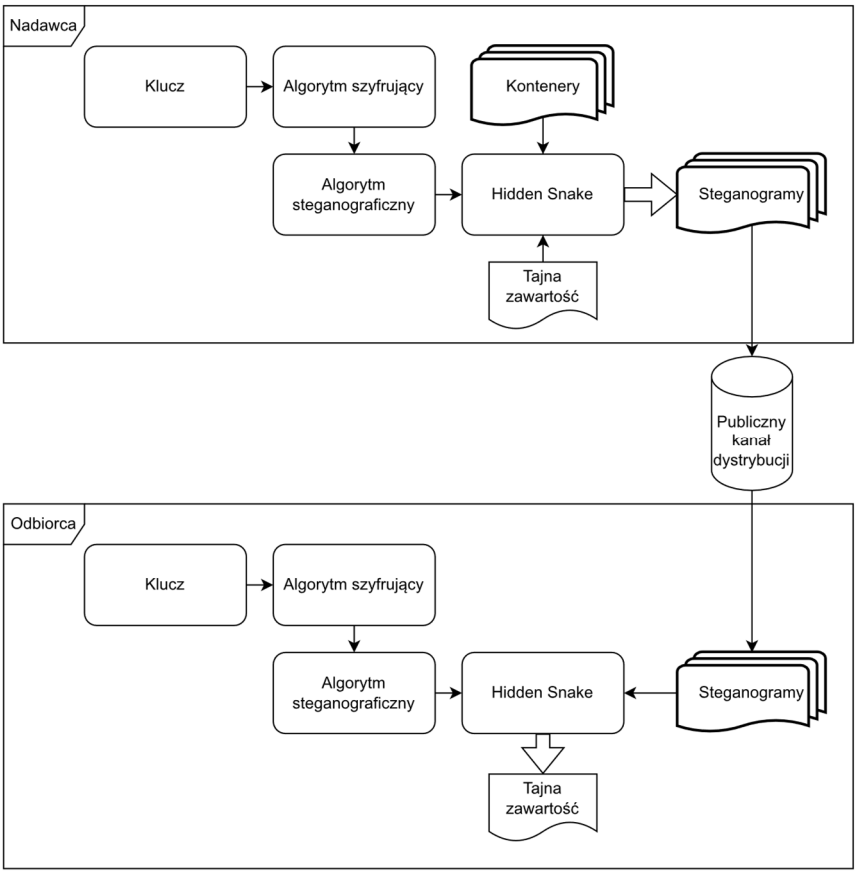
Znając kontekst oraz cele zadania stawianego przez tematykę pracy, utworzona została lista wymagań dla oprogramowania, które powinno ono spełniać:

- Program będzie obsługiwany przez prosty interfejs konsolowy, pozwalający zrealizować dotyczące go zadania pracy bez konieczności posiadania przez użytkownika umiejętności programistycznych,
- Program powinien mieć możliwość rozszerzania swojej funkcjonalności bez zbędnej ingerencji w utworzone komponenty,
- W swej bazowej wersji, program powinien pozwalać na wbudowanie ukrytej zawartości w dowolnie wskazany plik audio w formacie WAV,
- Treść wbudowanej wiadomości powinna być dowolna. Oznacza to, że możliwe powinno być ukrycie w kontenerze wiadomości tekstowej, jak i plików o dowolnym formacie,
- Ukrywane dane powinny być możliwe do podziału na dowolną ilość kontenerów, jeśli pojemność pojedynczego jest zbyt mała.
- Modyfikacja nośnika nie może spowodować uszkodzenia pliku, lub ryzyka kompromitacji faktu wbudowania w niego danych, na przykład poprzez wprowadzenie zbyt dużego szumu,
- Ukrywane dane powinny być szyfrowane.

Program realizujący zadanie będące tematem prac zrealizowany został w języku Python, wykorzystując podejście obiektowe. Do generowania raportów z testów oraz badań wykorzystano środowisko JupyterLab i pakiet Matplotlib.

2.1. Schemat ideowy realizowanego programu

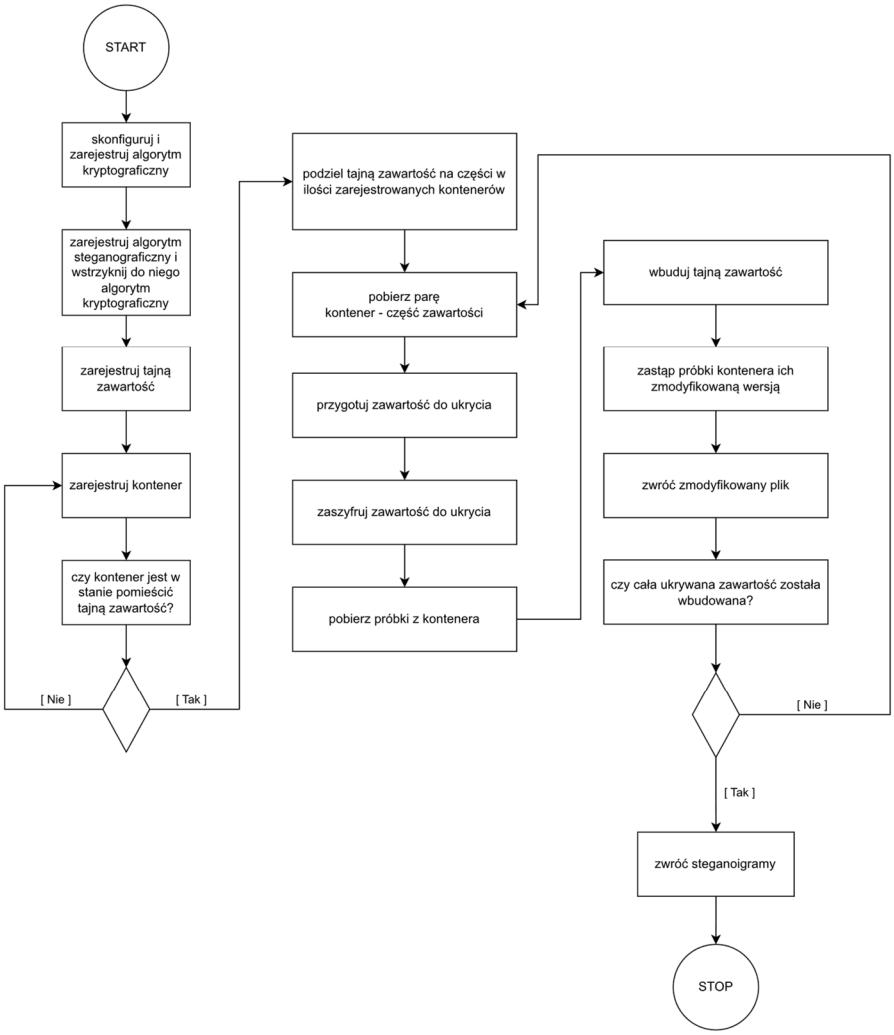
W przypadku komunikacji scenariusz może wyglądać następująco:
- Osoba A chce przesłać komunikat osobie B,
- Oboje ustalają znane tylko im hasło (lub przekazują sobie klucz publiczny) oraz kanał komunikacyjny – np. jakąś znaną platformę streamingową lub hosting danych,
- Osoba A tworzy steganogram w formie pliku audio zawierającym dowolną, jawną treść, w który wbudowano zaszyfrowany, tajny komunikat,
- Osoba A publikuje steganogram w ustalonym serwisie, dla niewtajemniczonych obserwatorów jest to nagranie, które nie wzbudza żadnych podejrzeń,
- Osoba B pobiera nagranie, wyodrębnia z niego komunikat i deszyfruje go, co umożliwi swobodny odczyt tajnej wiadomości.



Rysunek 1. Schemat przebiegu komunikacji z użyciem steganografii

W przypadku wykorzystania programu jako zabezpieczenia kopii dystrybuowanych dóbr cyfrowych, nadawcą jak i odbiorcą będzie dystrybutor, a danymi umieszczonymi w kontenerze będą dane klienta dla którego powstała kopia.

Dokonując wbudowania, nadawca wiadomości wybiera algorytm szyfrowania dokonując jego konfiguracji. Na cele projektu zaimplementowano algorytm Data Encryption Standard działający w trybie CBC [2][3]. Po wykonaniu tego kroku, użytkownik wybiera algorytm steganograficzny oraz wstrzykuje do niego gotowy obiekt kryptograficzny. Bazową implementacją w projekcie jest algorytm LSB [4][5]. Utworzony algorytm pozwala na modyfikację ilości zmienianych bitów w zakresie od 1 do 8. W drodze późniejszych testów wykazany zostanie wpływ modyfikacji tej liczby na wprowadzony do kontenera szum oraz jego pojemność. Zaznaczyć w tym miejscu należy, że system pozwala na dowolną wymianę i implementację tych komponentów. Warunkiem jest zachowanie interfejsu obiektu, którego definicja zawarta jest w odpowiednich klasach abstrakcyjnych. Dzięki temu można uzyskać konfiguracje programu wykorzystujące inne metody szyfrowania i steganografii. W dalszej kolejności konieczne jest zarejestrowanie tajnej zawartości w postaci bloku bajtów. Może być to zarówno wiadomość tekstowa jak i plik o dowolnym formacie. Użytkownik rejestruje listę kontenerów, do których zostanie wbudowana tajna wiadomość. Oprogramowanie na bieżąco sprawdza pojemność pakietu kontenerów i sprawdza, czy wbudowanie tajnej zawartości jest możliwe. Jeśli warunek ten zostanie spełniony, można rozpocząć proces ukrywania zarejestrowanej zawartości. Program wykorzystując zarejestrowane komponenty przeprowadza proces zwracając gotowe steganogramy.

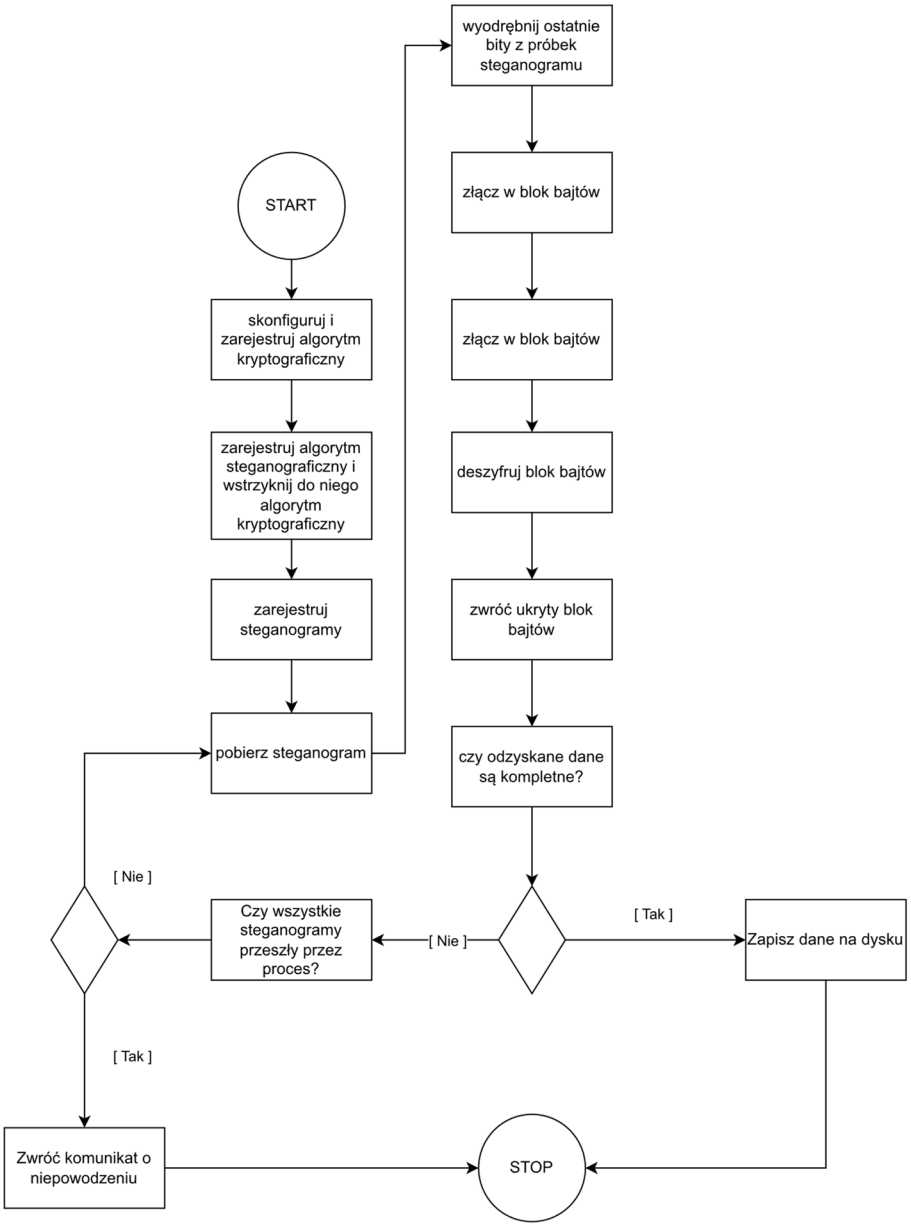


Rysunek 2. Proces wbudowania danych do kontenera

Utworzone w ten sposób pliki można potraktować jako poufne wiadomości przekazywane kanałem publicznym lub po prostu jako dobro cyfrowe przekazane do odbiorcy, który postanowił je nabyć.

Chcąc wyekstrahować i odczytać wiadomość, odbiorca wiadomości musi dokonać konfiguracji ustalonej z nadawcą wiadomości, to jest: wybrać ten sam algorytm kryptograficzny odpowiednio go konfigurując, a następnie wstrzyknąć go w odpowiedni algorytm steganograficzny. Bez poprawnej konfiguracji proces ekstrakcji danych nie powiedzie się. Tak samo niepowodzeniem zakończy się próba pozyskania ukrytej informacji w przypadku, gdy któryś z kontenerów zostanie pominięty lub uszkodzony. Po poprawnym ustawieniu programu, użytkownik

rejestruje w nim steganogramy. W tym momencie można rozpocząć odzyskiwanie danych ze steganogramu, zapisując ukrytą zawartość na komputerze odbiorcy.



Rysunek 3. Proces ekstrakcji danych z kontenera

2.2. Format plików WAV

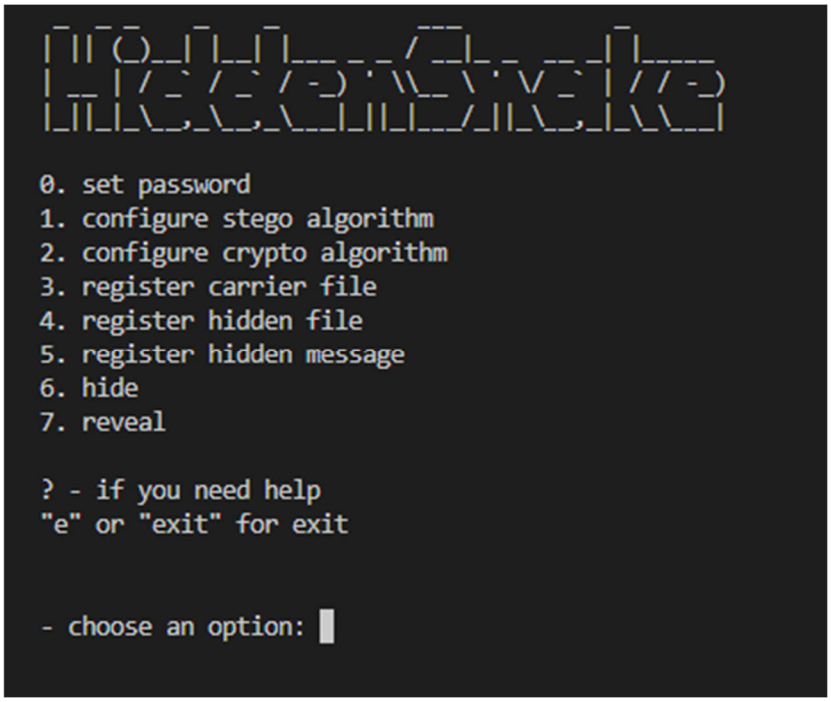
Bazowym formatem kontenera, możliwym do wykorzystania w utworzonej aplikacji jest format WAV. Jest to standard formatu audio opracowany przez IBM oraz Microsoft na potrzeby przechowywania dźwięków w formie cyfrowej na komputerach osobistych. Zawiera on przeważnie nieskompresowany zapis danych audio w postaci próbek formatu „linear pulse-code modulation" (LPCM) będącego również standardem kodowania informacji w standardowych Audio CD. Prosta budowa tego formatu, z klarownym podziałem na nagłówek zawierający metadane i dane będące treścią nagrania pozwala na łatwą edycję zawartości, tak więc również łatwo poddawać się będzie modyfikacjom wprowadzanym przez algorytm LSB.
Budowę standardowego pliku w tym formacie przedstawiono w poniższej tabeli [6]:

Tabela 1. Zawartość standardowego pliku w formacie wav

Przesunięcie [bajty]	**Rozmiar [bajty]**	**Nazwa**	**Opis**
0	4	ChunkID	Zawiera napis „RIFF" w kodowaniu ASCII
4	4	ChunkSize	Rozmiar całego pliku po odjęciu bloku „RIFF"
8	4	Format	Zawiera napis „WAVE"
12	4	Subchunk1ID	Standardowo „fmt ". Blok ten zawiera szczegółowe dane na temat kodowania dźwięku
16	4	Subchunk1Size	Rozmiar bloku „fmt ".
20	2	AudioFormat	Rodzaj kwantyzacji. Jeśli wartość jest ina niż 1, oznacza to, że dźwięk jest kompresowany
22	2	NumChannels	Liczba kanałów 1- mono, 2- stereo itd.
24	4	SampleRate	Liczba próbek w sekundzie nagrania
28	4	ByteRate	Liczba bajtów na próbkę dźwięku (SampleRate * NumChannels * BitsPerSample / 8)
32	2	BlockAlign	NumChannels * BitsPerSAmple / 8
34	2	BitsPerSample	Liczba bitów na próbkę 8, 16, 24 itd.
36	4	Subchunk2ID	Zawiera napis „data"
40	4	Subchunk2Size	Informacja o liczbie bajtów danych
44	*	data	Dane

3. Rezultaty wykonanych prac

Program posiada interfejs konsolowy pozwalający na konfigurację programu, utworzenie steganogramu oraz ekstrakcję z niego danych. Rejestrowana tajna wiadomość może być zarówno plikiem wskazanym ścieżką, jak i wiadomością tekstową. Program pozwala na zarejestrowanie jednej tajnej zawartości oraz dowolnej ilości kontenerów.



Rysunek 4. Interfejs konsolowy utworzonego programu

W celu weryfikacji rezultatów osiąganych przez finalny produkt będący tematem niniejszego artykułu, przygotowano zestaw testów w zakresie:

- Wydajności programu w kwestii wykorzystania pojemności kontenera,
- Jakości procesu wbudowania danych, oceniając ilość szumu wprowadzonego do kontenera,
- Przydatności programu do komunikacji steganograficznej oraz znakowania wodnego dóbr cyfrowych.

W celu oceny działania programu przygotowano trzy nagrania o następującej charakterystyce:

Tabela 2. Kluczowe charakterystyki plików wykorzystanych do testowania aplikacji

Nr nagrania	1	2	3
Liczba kanałów	2(stereo)	2(stereo)	2(stereo)
Liczba próbek na sekundę	48000	48000	48000
Rozdzielczość próbki	16 bit	16 bit	16 bit
Czas nagrania	7:03	3:51	5:00
Rozmiar	79.441 MiB	43.431 MiB	56.397 MiB
Opis	Muzyka gatunku psytrance, w większości utworzona w środowisku komputerowym	Muzyka rockowa, nagranie studyjne, gitary z dużą ilością efektów gitarowych przesterowujących dźwięk	Muzyka rockowa, remasterowane nagranie koncertu z roku 1993, duża ilość szumu

3.1. Pojemność kontenera

W trakcie testów sprawdzono wydajność algorytmu pod kątem wykorzystania pojemności kontenera. Jest ona zmienna zależnie od długości nagrania i ilości próbek na sekundę, co przekłada się na ilość próbek dźwięku możliwych do modyfikacji, oraz ilości bitów ulegających podmianie. Aby ją obliczyć należy oszacować ile próbek dźwięku potrzebnych jest do zapisania jednego bajtu ukrywanych danych, a następnie podzielić ilość próbek kontenera przez tę wartość. Obliczenia te można zawrzeć we wzorze:

$$C = \frac{S}{\lceil 8/nlsb \rceil} \tag{1}$$

gdzie :

C – pojemność kontenera
S – ilość próbek dźwiękowych w kontenerze
nlsb – ilość zmienianych bitów

Wyniki testów pojemności przedstawiono w poniższej tabeli:

Tabela 3. Pojemność testowanych kontenerów dla różnej ilości modyfikowanych bitów

Nr nagrania	1	2	3
1 LSB	4.85 MiB	2.65 MiB	3.44 MiB
2 LSB	9.7 MiB	5.3 MiB	6.88 MiB
3 LSB	12.93 MiB	7.07 MiB	9.18 MiB
4 LSB	19.39 MiB	10.6 MiB	13.77 MiB
5 LSB	19.39 MiB	10.6 MiB	13.77 MiB
6 LSB	19.39 MiB	10.6 MiB	13.77 MiB
7 LSB	19.39 MiB	10.6 MiB	13.77 MiB
8 LSB	38.79 MiB	21.21 MiB	27.54 MiB

3.2. Jakość procesu wbudowania danych

Przez jakość wbudowania rozumie się ilość szumu wprowadzonego do kontenera poprzez umieszczenie ukrywanych danych. W tym celu w każdym kontenerze wbudowano losowe bajty danych w ilości całkowicie wyczerpującej jego pojemność. Proces ten został przeprowadzony dla liczby modyfikowanych bitów w zakresie 1-8. Następnie przeprowadzono porównanie kolejnych steganogramów z plikiem bazowym, wyliczając szczytowy stosunek sygnału do szumu (PSNR) dla każdego z nich.

Aby obliczyć PSNR dla pliku audio w pierwszej kolejności należy wyliczyć współczynnik średniego błędu kwadratowego (MSE) dla porównywanych nagrań. MSE dla szeregów czasowych jakim są sygnały dźwiękowe, oblicza się przy użyciu wzoru:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})^2 \text{ oraz } \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2 \tag{2}$$

gdzie: $\bar{X} = (X_1 + \cdots + X_n)/n$

Następnie w celu obliczenia PSNR wyliczoną wartość MSE podstawia się do wzoru końcowego:

$$PSNR = 10 \cdot log_{10}\frac{[\max(f(i))]^2}{MSE} \tag{3}$$

gdzie: $\max(f(i))$ – wartość maksymalna danego sygnału; w przypadku plików audio o próbce 16bit wartość ta wynosi 32767.

W przypadku, gdy MSE zmierza do 0, wartość PSNR zwiększa się, dążąc do nieskończoności. Wraz ze wzrostem błędu wywołanego dodanym szumem, wartość PSNR maleje.

Wyniki przeprowadzonych obliczeń przedstawiono w poniższej tabeli:

Tabela 4. Szczytowy stosunek sygnału do szumu (PSNR) dla steganogramów o różnej ilości zmodyfikowanych bitów

Nr nagrania	1	2	3
1 LSB	93.319 dB	93.319 dB	93.319 dB
2 LSB	86.322 dB	86.315 dB	86.326 dB
3 LSB	80.091 dB	80.086 dB	80.094 dB
4 LSB	74.004 dB	73.992 dB	74.018 dB
5 LSB	67.241 dB	67.243 dB	67.243 dB
6 LSB	60.47 dB	60.477 dB	60.473 dB
7 LSB	54.247 dB	54.253 dB	54.249 dB
8 LSB	53.033 dB	53.032 dB	53.012 dB

Porównując je z wynikami zawartymi w opracowaniu [5] str. 106-108, można przyjąć, że zniekształcenia których wartość PSNR nie jest mniejsza niż 85dB, są porównywalne do obecnego już na rynku oprogramowania realizującego procesy

steganografii dla sygnału audio zawierającego muzykę. Utworzone oprogramowanie spełnia ten rygor dla głębokości zmian próbek równej 1 i 2 bity. Jednocześnie zwrócić należy uwagę na rozwojowy charakter prac, gdyż osiągane wyniki mają jeszcze potencjał wzrostowy przy zastosowaniu innych, bardziej złożonych metod ukrywania.

4. Podsumowanie

Artykuł przedstawia oprogramowanie do ukrywania danych w plikach audio poprzez steganografię, podkreślając jego znaczenie w dzisiejszym cyfrowym świecie. Przeprowadzone testy wykazały, że program oferuje dobrą pojemność kontenerów oraz wysoką jakość ukrywania danych. To istotne, gdyż pojemność kontenera ma kluczowe znaczenie dla sukcesu komunikacji steganograficznej.

Niezwykła elastyczność oprograowaniaa jest kolejnym aspektem, który zasługuje na uwagę. Program umożliwia ukrywanie zarówno danych tekstowych, jak i binarnych, co czyni go wszechstronnym narzędziem do ukrywania informacji. Otwarty dostęp do kodu źródłowego programu sprzyja jego dalszemu rozwojowi i doskonaleniu. Współpraca z innymi naukowcami i studentami może prowadzić do nowych i innowacyjnych metod ukrywania danych oraz zwiększania pojemności kontenerów.

LITERATURA

1. Główny Urząd Statystyczny – Jak korzystamy z internetu 2022: *https://stat.gov.pl/download/gfx/portalinformacyjny/pl/defaultaktualnosci/5497/5/13/1/jak_korzystamy_z_internetu_2022__00b.pdf*, 08.09.2023
2. The Legacy of DES – Schneider on Security, *www.schneider.com*, 25.10.2023
3. Federal Information Processing Standards Publication: Data Encryption Standard *(DES)* , 1999.
4. LOKESWARA REDDY V., SUBRAMANYAM A., CHENNA REDDY P.: Implementation of LSB Steganography and its Evaluation for Various File Formats, Int. J. Advanced Networking and Applications 868 Volume: 02, Issue: 05, 868-872 (2011).
5. KOZIEŁ G.: Zmodyfikowane metody cyfrowego przetwarzania sygnałów dźwiękowych w steganografii komputerowej, Lublin, 2010
6. Soundfile.sapp.org, *http://soundfile.sapp.org/doc/WaveFormat/*, 8.08.2023