Biuletyn WAT
Vol. LXII, Nr 4, 2013

Steganografia z wykorzystaniem cyklicznych kodów korekcji błędów

KAMIL KACZYŃSKI

Wojskowa Akademia Techniczna, Wydział Cybernetyki, Instytut Matematyki i Kryptologii,
00-908 Warszawa, ul. gen. S. Kaliskiego 2, kkaczynski@wat.edu.pl

Streszczenie. Kody cykliczne, będące podklasą kodów liniowych, znalazły największe zastosowanie praktyczne w korekcji błędów. Ich główne zalety to efektywność konstruowania kodów o wymaganych właściwościach, a także prosta realizacja koderów i dekoderów za pomocą LFSR.W niniejszej publikacji omówiono jeden z najważniejszych typów kodów cyklicznych — kody BCH. Opracowano i zaimplementowano stegosystem, będący modyfikacją algorytmu LSB i wykorzystujący właściwości syndromu kodu BCH. Dokonano porównania efektywności obydwóch algorytmów, a także wyników stegoanalizy algorytmem RS oraz metodami porównawczymi.
Słowa kluczowe: steganografia, kody cykliczne, LSB, BCH

1. Wstęp

Podstawowe algorytmy steganograficzne dla bitmap, takie jak LSB (z ang. najmniej znaczący bit) powodują wprowadzenie dużej liczby zmian do obrazu – nośnika. Alternatywą dla prostych algorytmów stają się algorytmy zmodyfikowane, wykorzystujące tzw. kodowanie syndromami. Pierwszym algorytmem, który wykorzystywał liniowy kod Hamminga, był algorytm F5 stworzony przez Westfelda w 2001 roku. Kod Hamminga jest kodem doskonałym, co oznacza, że każdy syndrom reprezentuje dokładnie jeden wektor błędu. W przypadku kodu Hamminga (7,4) możliwe stało się ukrycie trzech bitów wiadomości przy zmianie zaledwie jednego z siedmiu bitów nośnika. Rozwiązanie to jest bardzo wygodne w implementacji, niestety ukrycie pewnej wiadomości zawsze wprowadza taki sam wektor błędu do nośnika. Alternatywą dla kodów doskonałych stają się zatem kody cykliczne takie jak

BCH. Ich główną zaletą jest fakt, że każdy syndrom opisuje kilka wektorów błędu. Właściwość ta może zostać wykorzystana w procesie wbudowywania wiadomości tak, aby wprowadzić ją w tych obszarach obrazu, w których wykrycie w procesie stegoanalizy będzie najtrudniejsze. W poniższym artykule zaprezentowany zostanie stegosystem wykorzystujący kod BCH (15,7).

2. Kody BCH (Bose-Chaudhuri-Hocquenghem)

Kody BCH są podklasą cyklicznych kodów korekcji błędów zbudowanych nad ciałami skończonymi. Zostały wynalezione w 1959 r. przez francuskiego matematyka Hocquenghema i niezależnie w 1960 r. przez Bose'go i Ray-Chadhuriego. Skrót BCH powstał z połączenia pierwszych liter nazwisk twórców. Główną zaletą kodów BCH jest możliwość dokładnego określenia liczby korygowanych błędów w czasie projektowania kodu. Kolejną zaletą jest możliwość łatwej korekcji błędów przy wykorzystaniu syndromów.

Kody BCH można konstruować zarówno nad ciałami binarnymi, jak i ciałami rozszerzonymi, najczęściej stosowane są kody binarne. Dla każdej liczby całkowitej m i $t < 2^{m-1}$ istnieje kod BCH o długości $n = 2^m - 1$, który może korygować maksymalnie t błędów i ma nie więcej niż mt elementów kontrolnych. Długość wektora kodowego — $n = 2^m - 1$, liczba pozycji kontrolnych — $n - k \leq mt$, odległość minimalna — $d \geq 2t + 1$.

Praktyczne wykorzystanie kodu BCH wymaga wyznaczenia tzw. wielomianu generującego kod BCH. Metoda wyznaczenia wielomianu generującego została zaczerpnięta z [1]. Niech α będzie elementem pierwotnym ciała $GF(2^m)$. Zbiór $\{f(x)\}$ jest zbiorem ciągów kodowych kodu BCH, jeśli pierwiastkami dowolnie wybranego wielomianu $f(x)$ są elementy ciała $\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{2t}$. Każdy element ciała o parzystym wykładniku ma w tej sekwencji taką samą funkcję minimalną jak któryś z poprzedzających go elementów o wykładniku nieparzystym. Na przykład α^2 i α^4 są pierwiastkami $m_1(x)$, α^6 jest pierwiastkiem $m_3(x)$ itd. Uwzględniając ten fakt podczas wyznaczania wielomianu generującego kod BCH, wystarczy wziąć pod uwagę elementy ciała z wykładnikami nieparzystymi. Wielomian generujący kod BCH o zdolności korekcyjnej t jest najmniejszą wspólną wielokrotnością funkcji minimalnych $m_1(x), m_3(x), \ldots, m_{2t-1}(x)$.

$$g(x) = NWW(m_1(x), m_3(x), \ldots, m_{2t-1}(x)).$$

3. Kod BCH (15,7)

Na potrzeby opracowywanego stegosystemu wykorzystany został kod BCH (15,7), zbudowany nad ciałem $GF(2^4)$ o długości słowa kodowego 15 bitów, który może

korygować dwa błędy. Długość syndromu wynosi 8 bitów. Wielomian generujący dla powyższego kodu ma postać:

$$g(x) = (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1) = x^8 + x^7 + x^6 + x^4 + 1.$$

Efektywna realizacja stegosystemu wymaga wyznaczenia macierzy kontrolnej kodu BCH. Macierz ta jest wyznaczana na podstawie macierzy generującej kod cykliczny. Może ona zostać wyznaczona przy wykorzystaniu właściwości przesunięcia cyklicznego kodu. Dla powyższego kodu BCH (15,7) macierz ta ma 7 wierszy i 15 kolumn. Poniżej przedstawiona jest postać macierzy generującej:

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Słowa kodowe są uzyskiwane w wyniku mnożenia słów wiadomości przez macierz G. Słowo kodowe to wektor: $c = (c_0, c_1, \ldots, c_{n-1})$, słowo wiadomości to wektor $m = (m_0, m_1, \ldots, m_{k-1})$. Zależność pomiędzy wektorem c i m przedstawia poniższa formuła:

$$c = mG.$$

Macierz kontroli parzystości H jest macierzą o wymiarach 8 wierszy na 15 kolumn. Wiersze macierzy G są ortogonalne do wierszy macierzy H, co oznacza, że $G \cdot H^T = 0$, gdzie H^T jest transponowaną macierzą H.

Macierz kontroli parzystości można obliczyć na podstawie znajomości macierzy generującej G. Przestrzeń wektorowa generowana przez macierz G oraz przestrzeń wektorowa generowana przez macierz H są podprzestrzeniami przestrzeni wektorowej zawierającej wszystkie wektory 15-elementowe i dlatego stanowią one kody liniowe.

Dla kodu BCH (15,7) transponowana macierz kontroli parzystości H^T ma następującą postać:

$$H^T = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Macierz kontroli parzystości pozwala na obliczenie syndromu s. Syndrom informuje o położeniu błędu w przesłanym ciągu. Syndrom obliczany jest przy wykorzystaniu następującej formuły:

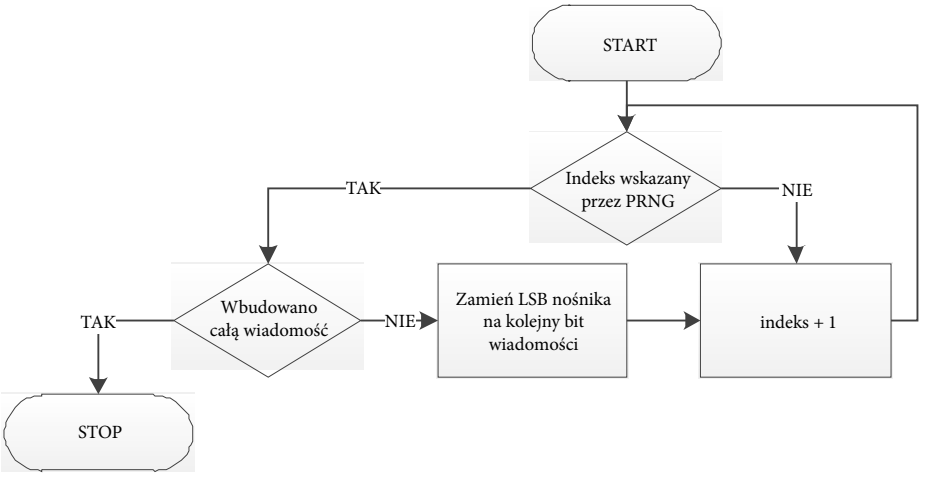
$$s = c \cdot H^T.$$

Wektor s zawiera informacje o dodanym w czasie transmisji wektorze błędu e. Zerowa wartość syndromu oznacza, że otrzymany wektor c jest wektorem kodowym. Niezerowa wartość syndromu sygnalizuje, że otrzymany wektor c nie jest wektorem kodowym i zostały wykryte błędy transmisyjne. Dla kodu BCH (15,7) syndrom ma długość 8 bitów, a wartości kolejnych bitów mogą zostać obliczone przy wykorzystaniu poniższego układu równań:

$$\begin{cases} s_0 = c_0 + c_4 + c_6 + c_7 \pmod 2 \\ s_1 = c_0 + c_1 + c_4 + c_5 + c_6 + c_8 \pmod 2 \\ s_2 = c_0 + c_1 + c_2 + c_4 + c_5 + c_9 \pmod 2 \\ s_3 = c_1 + c_2 + c_3 + c_5 + c_6 + c_{10} \pmod 2 \\ s_4 = c_0 + c_2 + c_3 + c_{11} \pmod 2 \\ s_5 = c_1 + c_3 + c_4 + c_{12} \pmod 2 \\ s_6 = c_2 + c_4 + c_5 + c_{13} \pmod 2 \\ s_7 = c_3 + c_5 + c_6 + c_{14} \pmod 2. \end{cases}$$

4. Algorytm LSB

Algorytm LSB (ang. Najmniej Znaczący Bit — *Least Significant Bit*) to jeden z najbardziej rozpowszechnionych i najlepiej zbadanych algorytmów steganograficznych. Zasada działania algorytmu jest wyjątkowo prosta — najmniej znaczący bit nośnika jest zamieniany na bit wiadomości. Najczęściej stosowaną wersją algorytmu LSB jest algorytm ulosowiony. Nadawca oraz odbiorca posiadają wspólny klucz, który jest używany jako ziarno generatora pseudolosowego. Generator tworzy sekwencję indeksów pikseli, w których ukrywana będzie wiadomość. Dwie główne cechy tej metody to dodatkowe zabezpieczenie przesyłanych danych przed nieuprawnionym dostępem, a także równomierne rozmieszczenie bitów wiadomości w nośniku. Poniższy rysunek przedstawia schemat algorytmu LSB.



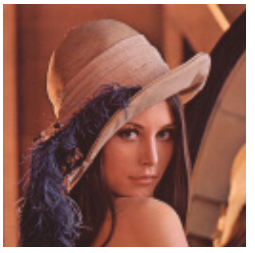
Rys. 1. Schemat blokowy algorytmu LSB

Główną zaletą algorytmu LSB jest bardzo duża pojemność nośnika. Dla zaproponowanego algorytmu pojemność c standardowej bitmapy o wymiarach h na w pikseli jest w przybliżeniu równa:

$$C \approx \frac{h * w}{2}.$$

Przykładowo, bitmapa o wymiarach 128 × 128 pikseli, przy modyfikacji tylko jednego bitu koloru niebieskiego i wykorzystaniu pseudolosowego wzorca, pozwala na ukrycie ok. 8192 bitów, co daje 1 kB danych — 1000 znaków.

Podstawową wadą algorytmu jest liczba zmian wprowadzanych do obrazu – nośnika. Średnia liczba zmienionych pikseli wynosi ¼ całości. W przypadku ogólnym ukrycie dwóch bitów wiadomości wymaga zmiany wartości jednego piksela nośnika. Im więcej zmian algorytm wprowadza do nośnika, tym łatwiej wykryć istnienie ukrytego przekazu w nośniku.



Rys. 2. Obraz nośnik

Rys. 3. Obraz z wbudowaną wiadomością

5. Zmodyfikowany algorytm LSB

Kodowanie syndromami kodu BCH (15,7) pozwala na ukrycie 8 bitów danych w bloku 15-bitowym przy zmianie zera, jednego, dwóch lub trzech bitów nośnika. Pojemność (w bitach) standardowej bitmapy o wymiarach h na w pikseli jest w przybliżeniu równa:

$$C = \frac{h * w * 8}{15}.$$

Przyjmijmy, że oryginalny blok danych nośnika to V = $\{v_0, v_1, \ldots, v_{14}\}$, natomiast blok danych po modyfikacji to R = $\{r_0, r_1, \ldots, r_{14}\}$. Ukrywana wiadomość $m = \{m_0, m_1, \ldots, m_7\}$ może być obliczona z poniższej zależności:

$$m = \mathrm{R} \cdot \mathrm{H}^{\mathrm{T}}.$$

Ukrycie wiadomości *m* wymaga zatem odnalezienia takiego R, że $R \cdot H^T = m$. Różnica $E = \{e_0, e_1, \ldots, e_{14}\}$ pomiędzy blokiem V i R wskazuje liczbę oraz położenie bitów, które powinny zostać zmienione.

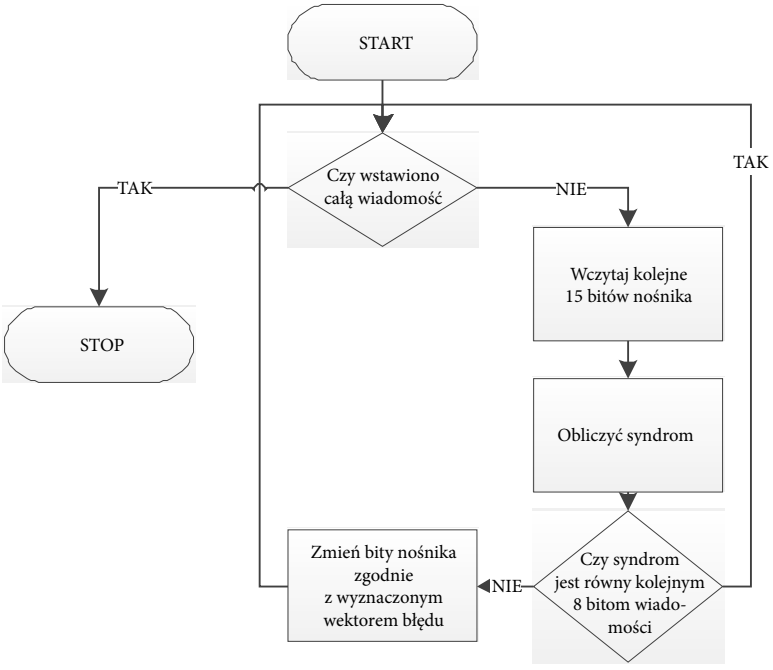
$$E = V - R.$$

Ostatnim krokiem ukrycia wiadomości *m* do nośnika jest dodanie wektora błędu E do bloku bitów nośnika R.

Przykładowo, gdy w słowie kodowym 110100011001011 chcemy ukryć bity 11011001:

a) obliczamy syndrom dla słowa kodowego: $c \cdot H^T = [0\ 1\ 0\ 0\ 1\ 0\ 1\ 0]$;
b) obliczamy różnicę pomiędzy obliczonym syndromem a ukrywaną wiadomością $s - m = [1\ 0\ 0\ 1\ 0\ 0\ 1\ 1]$;
c) wyznaczamy wektor błędu E, dla którego syndrom jest równy [1 0 0 1 0 0 1 1], E = [1 0 0 0 0 1 0 0 0 0 0 1 0 0 0];
d) dodajemy wektor błędu E do wektora V
R = E + V = [0 1 0 1 0 1 0 1 1 0 0 0 0 1 1].

Poniższy rysunek przedstawia schemat blokowy zmodyfikowany algorytmu LSB.

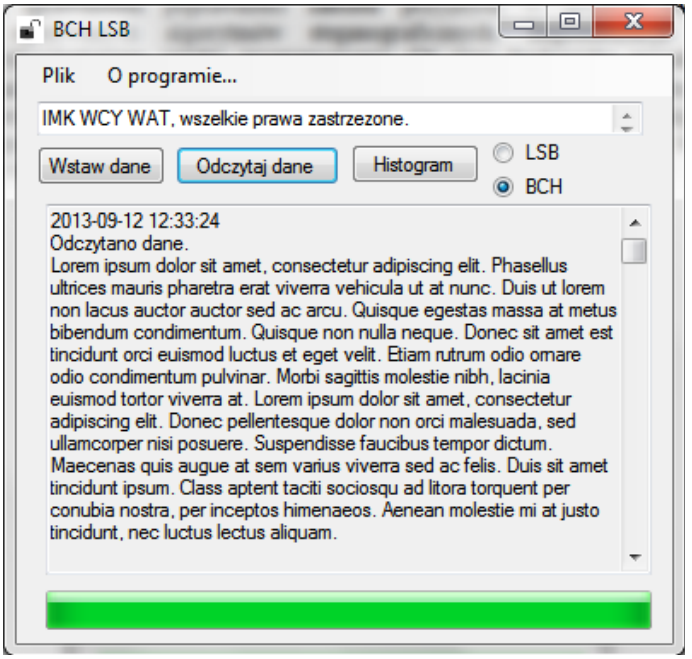


Rys. 4. Schemat blokowy zmodyfikowanego algorytmu LSB

Przykładowo, bitmapa o wymiarach 128 × 128 pikseli, przy modyfikacji tylko jednego bitu koloru niebieskiego i wykorzystaniu macierzy parzystości kodu BCH (15,7) pozwoliła na ukrycie 8736 bitów wiadomości, co daje 1092 znaki. W przypadku ogólnym, ukrycie 8 bitów wiadomości wymaga zmiany 2,47 bitu nośnika.

6. Implementacja

W celu sprawdzenia poprawności założeń przyjętych w artykule wykonano implementację opisywanych algorytmów steganograficznych. Implementacja została wykonana z wykorzystaniem języka programowania C# oraz środowiska projektowego Microsoft Visual Studio 2010. Program umożliwia wbudowanie oraz wyodrębnienie wiadomości przy wykorzystaniu podstawowego algorytmu LSB oraz algorytmu zmodyfikowanego. Na potrzeby analizy statystycznej zaimplementowano moduł tworzący histogram dla wybranego koloru bitmapy.



Rys. 5. Aplikacja BSH LSB

Praktyczna realizacja algorytmu zmodyfikowanego wymagała utworzenia tablicy wektorów błędu odpowiadających wszystkim syndromom. Ze względu na dużą złożoność czasową poszukiwania rozwiązań, utworzono pomocniczą

aplikację, której zadaniem było odnalezienie wektorów błędu o minimalnej wadze Hamminga. Czynność ta pozwoliła na znaczne przyspieszenie pracy algorytmu zmodyfikowanego.

Aplikacja pozwala na dodanie wiadomości do plików o rozszerzeniu bmp, dla których każda składowa koloru RGB składa się z 8 bitów. Wiadomość może być dowolnym ciągiem binarnym, jednak na potrzeby testowania aplikacji przyjęto, że będzie tekstem kodowanym zgodnie ze standardem UTF-8.

7. Stegoanaliza

W celu porównania algorytmu podstawowego oraz algorytmu zmodyfikowanego została przeprowadzana analiza utworzonych plików wynikowych, do której zostały wykorzystane najpopularniejsze algorytmy stegoanalityczne.

Stegoanaliza algorytmem RS wykorzystuje funkcje dyskryminacji oraz operację przerzucania do identyfikacji trzech grup pikseli — regularnych (*Regular* — R), pojedynczych (*Singular* — S) oraz niezmienionych (*Unchanged* — U). Przypisanie jest zależne od zmiany wartości funkcji dyskryminacji po wykonaniu operacji przerzucania. Wielkość grupy pikseli i odpowiadająca maska przerzucania M jest zakładana na wstępie. Przykładowo, gdy M = [0 1 0] odpowiada to testowi przeprowadzanemu na grupie 3 pikseli, w której tylko piksel środkowy został przerzucony. W typowych obrazach stosowanie maski przerzucania algorytmu LSB będzie częściej prowadziło do wzrostu funkcji dyskryminacji niż do jej spadku. Stąd całkowita liczba grup regularnych w obrazie będzie większa niż grup pojedynczych. Losowość, jaką wprowadza algorytm LSB, powoduje, że różnica ta będzie dążyła do zera wraz ze wzrostem długości wbudowanej wiadomości.

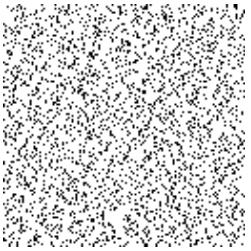
Wbudowywanie wiadomości o względnej długości p ($p = 1$ oznacza wykorzystanie wszystkich pikseli obrazu) do obrazu źródłowego wymusza przeciętne przerzucenie $p/2$ pikseli. Przerzucenie wszystkich pikseli w obrazie będzie skutkowało utworzeniem obrazu, w którym udział zmienionych pikseli będzie równy $1 - 1/2$. W procesie stegoanalizy obrazu zakładamy, że wartość p nie jest znana. Względna liczba grup R i S jest obliczana zarówno dla obrazu oryginalnego, jak i wersji obrazu, dla którego przerzucono wszystkie wartości LSB. Wynikiem są cztery punkty tzw. diagramu RS, który jest wykorzystywany do estymacji wartości p. Szczegółowy opis metody znajduje się w [2].

W ramach realizacji zadania dokonano analizy kolejno obrazu oryginalnego, obrazu z wiadomością wbudowaną klasycznym algorytmem LSB oraz obrazu z wiadomością wbudowaną przy wykorzystaniu algorytmu zmodyfikowanego. Do wykonania analizy użyte zostało oprogramowania Stegsecret oraz Virtual Steganographic Laboratory. Tabela 1 przedstawia wyniki analizy.

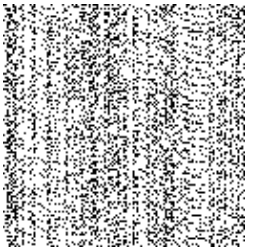
Tabela 1

Wyniki analizy algorytmem RS

	Obraz oryginalny	Algorytm LSB	Zmodyfikowany algorytm LSB
Procent zmienionych bitów (kolor niebieski)	9,13%	62,21%	38,65%
Procent zmienionych bitów (kolor czerwony)	13,6%	13,6%	13,6%
Procent zmienionych bitów (kolor zielony)	10,07%	10,07%	10,07%
Szacowana długość wiadomości [B]	672	1759	1276



Rys. 6. Różnica dla zmodyfikowanego algorytmu LSB



Rys. 7. Różnica dla algorytmu LSB

Kolejna analiza polegała na porównaniu zmian wprowadzonych w obrazie oryginalnym przez obydwa algorytmy. W wyniku porównania została wykonana operacja xor na najmniej znaczącym bicie koloru niebieskiego obrazu oryginalnego oraz obrazów zmodyfikowanych.

8. Podsumowanie

Na podstawie przeprowadzonych badań jasno wynika, że zmodyfikowany algorytm LSB wprowadza znacząco mniej zmian do nośnika niż algorytm podstawowy, przy jednoczesnym zwiększeniu pojemności nośnika. Zmniejszenie liczby zakłóceń wprowadzanych do obrazu pozwala na zwiększenie bezpieczeństwa stosowanego algorytmu.

Warto zaznaczyć, że publikacja zawiera opis wykorzystania jedynie macierzy kontroli parzystości kodu BCH (15,7), a zastosowany algorytm opierał się na stablicowanych wartościach wektora błędu. Teoria kodowania dysponuje jednak wieloma innymi kodami, które z powodzeniem mogą zostać wykorzystane do modyfikacji większości aktualnie wykorzystywanych algorytmów steganograficznych.

LITERATURA

[1] W. Mochnacki, *Kody korekcyjne i kryptografia*, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 2000.

[2] J. Fridrich, M. Goljan, D. Hogea, D. Soukal, *Quantitative steganalysis of digital images: estimating the secret message length*, Multimedia Systems, 9, 2003, 288-302.

[3] E. Cole, *Hiding in Plain Sight: Steganography and the Art of Covert Communication*, Wiley Publishing, Inc., 2003.

[4] S. Katzenbeisser, F.A.P. Petitcolas, *Information Hiding Techniques for Steganography and Digital Watermarking*, Artech House, 2000.

[5] I.J. Cox, M.L. Miller, J.A. Bloom, J. Fridrich, T. Kalker, *Digital Watermarking and Steganography*, second edition, Morgan Kaufmann, 2008.

[6] P. Forczmański, M. Węgrzyn, *Virtual Steganographic Laboratory for Digital Images, In Information Systems Architecture and Technology: Information Systems and Computer Communication Networks*, Wrocław, Polska, 2008, 163-174.

[7] P. Forczmański, M. Węgrzyn, *Open Virtual Steganographic Laboratory*, International Conference on Advanced Computer Systems, ACS-AISBIS, 2009.

[8] J. Gawinecki, N. Courtois, G. Song, *Contradiction immunity and guess-then-determine attacks on GOST*, Tatra Mt. Publ., 53, 65-79.

[9] J.A. Gawinecki, N.T. Courtois, D. Hulme, K. Hussain, *On Bad Randomness and Cloning of Contactless Payment and Building Smart Cards*, Security and Privacy Workshop, IEEE, 2013, 105-110.

K. KACZYŃSKI

Steganography and cyclic error-correcting codes

Abstract. Cyclic codes, which are subclass of linear codes, are of particularly widespread use in error correction. Their main advantages are: the effectiveness of constructing codes with required properties and simple implementation of decoders and encoders using LFSR. This paper discusses one of the most important cyclic codes — BCH code. Developed and implemented stegosystem is a modification of LSB algorithm, which uses the properties of BCH code syndromes. Comparison of simple and modified algorithm efficiency was made. Steganalysis of both algorithms were made and the results were compared.

Keywords: steganography, cyclic code, error correction codes, LSB, BCH