III. ALGORYTMY STENOGRAFICZNE ORAZ KRYPTOLOGICZNE I ICH BEZPIECZEŃSTWO

STEGODROID – APLIKACJA MOBILNA DO PROWADZENIA UKRYTEJ KOMUNIKACJI

Jerzy Gawinecki, Kamil Kaczyński

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

Streszczenie. Komputery i telefony komórkowe są narzędziami wykorzystywanymi w niemalże wszystkich dziedzinach życia codziennego. Ogólnoświatowe sieci komputerowe łączą zarówno komputery osobiste, wielkie centra danych, jak i urządzenia przenośne takie jak telefony komórkowe. Zapewnienie bezpieczeństwa systemów informatycznych wymaga wykorzystania technik umożlwiających przesyłanie danych do zdalnych lokalizacji w sposób gwarantujący ich poufność, integralność i dostępność. Do realizacji tych celów wykorzystuje się techniki kryptograficzne i steganograficzne. Kryptografia obrabia wiadomości w taki sposób, aby nie można było ich w łatwy sposób zrozumieć, zaś steganografia ukrywa wiadomości, tak aby ich istnienie było niemożliwe do odkrycia. Obydwie metody można ze sobą łączyć, dzięki czemu przesyłana wiadomość może być chroniona w dwojaki sposób. W tym przypadku, gdy steganografia zawiedzie i wiadomość zostanie wykryta, nadal nie będzie możliwe odczytanie jej treści przez wzgląd na zastosowane metody kryptograficzne. W ramach niniejszej pracy wykonano implementację autorskiego algorytmu steganograficznego, wykorzystującego liniowe kody korekcji błędów. Opracowana aplikacja jest przeznaczona dla urządzeń pracujących pod kontrolą systemu Android, tym samym istnieje szerokie grono odbiorców takiego oprogramowania.

Słowa kluczowe: steganografia, kryptografia, Android

1. Wstęp

Gwałtowny postęp technologiczny w zakresie budowy urządzeń mobilnych doprowadził do sytuacji, w której mogą one z powodzeniem zastąpić komputery osobiste. Urządzenia takie, jak smartfony wydają się być ideal-

Praca dofinansowana ze środków NCBiR na lata 2013-2018, w ramach projektu numer DOBR/0001/RON/ID1/2013/01.

nymi do prowadzenia komunikacji. Wątpliwym natomiast jest, czy komunikacja prowadzona w taki sposób jest w pełni bezpieczna. Istnieje wiele możliwości przechwytywania transmitowanych danych, a także ich odczytywania. Warto nadmienić, że metody kryptograficzne, które są powszechnie wykorzystywane, nie gwarantują pełnego bezpieczeństwa prowadzonej komunikacji. Przykładem może być tu m. in. skompromitowany szyfr strumieniowy A5/1 stosowany do przykrycia kryptograficznego komunikacji GSM. Ostatni atak na ten algorytm pochodzi z 2009 roku [5]. Pozwala on na złamanie kluczy A5/1 w czasie 3-5 minut przy wykorzystaniu tzw. „tęczowych tablic o rozmiarze 2 TB.

Producenci oprogramowania skupiają się przede wszystkim na stosowaniu kryptografii. Jest to słuszne podejście, jednakże stosowanie wyłącznie kryptografii niesie także pewne zagrożenia. Strony, które będą chroniły swoją komunikację z wykorzystaniem kryptografii niechybnie zwrócą na siebie uwagę odpowiednich służb. Powyższe może zostać ograniczone poprzez połączenie metod kryptograficznych ze steganografią. Zabieg taki pozwala nie tylko na zabezpieczenie przesyłanych informacji, ale także na ukrycie samego faktu prowadzenia komunikacji.

W niniejszej pracy przedstawiono oprogramowanie „Stegodroid" stworzone na potrzeby urządzeń pracujących pod kontrolą systemu operacyjnego Android. Oprogramowanie łączy autorski algorytm steganograficzny oparty na liniowych kodach korekcji błędów i klasyczny szyfr blokowy AES.

2. Algorytm steganograficzny

Na potrzeby aplikacji „Stegodroid zaadaptowano algorytm steganograficzny będący połączeniem algorytmu LSB i idei kodowania syndromami [10]. Algorytm ten wykorzystuje kod Hamminga, który należy do klasy liniowych kodów korekcji błędów. Kod korekcji błędów to nadmiarowa informacja dodana do ciągu binarnego, która pozwala na całkowitą lub częściową detekcję i korekcję błędów powstałych w wyniku zakłóceń lub częściowej modyfikacji sygnału źródłowego. Kodowanie korekcyjne jest stosowane wtedy, gdy ponowne przesłanie informacji jest kosztowne, kłopotliwe lub niemożliwe. Zarówno detekcja jak i korekcja błędów opiera się na założeniu, że jedynie pewne sekwencje odebranych bitów są poprawne.

Utworzenie n-bitowego słowa kodowego wymaga dodania do k-bitowego słowa wiadomości $n-k$ bitów parzystości. Kod taki jest nazywany kodem blokowym, ponieważ jednocześnie kodowane są bloki danych składające się z wielu bitów. Gdy bity wiadomości nie zostają zmodyfikowane, kod nazywany jest kodem systematycznym. Dekodowanie takiego kodu wymaga jedynie odrzucenia uprzednio dodanych bitów parzystości.

Liniowy kod blokowy (n, k) to k-wymiarowa podprzestrzeń przestrzeni wektorowej n-wymiarowej. Możliwe zatem jest znalezienie k liniowo niezależnych wektorów, które będą bazą przestrzeni kodowej. Kod liniowy (n, k) może być jednoznacznie określony przez dowolny zbiór k liniowo niezależnych wektorów, tworzących bazę przestrzeni kodowej, która stanowi podstawę do utworzenie macierzy generującej kod $G_{k\times n}$.
Słowa kodowe są uzyskiwane w wyniku mnożenia słów wiadomości przez macierz G. Słowo kodowe to wektor: $c = (c_0, c_1, \cdots, c_{n-1})$, słowo wiadomości to wektor $m = (m_0, m_1, \cdots, m_{k-1})$.
Zależność pomiędzy wektorem c i m przedstawiona jest poniżej:

$$c = mG$$

Liniowy kod Hamminga jest kodem korekcyjnym zbudowanym nad $GF(2)$. Macierz generująca G kodu Hamminga (7,4) może mieć następującą postać:

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

Aby zakodować wiadomość $m = 1010$ wykonujemy następującą operację:

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

Tak otrzymany wektor kodowy $c = 1011010$. Detekcji i korekcja błędów wymaga zastosowania tzw. macierzy kontroli parzystości H. Macierz kontroli parzystości dla kodu Hamminga (7,4) ma postać:

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

Co jest równoważne poniższemu układowi równań:

$$\begin{cases} x_1 + x_3 + x_5 + x_7 = 0 (\mathrm{mod}\, 2) \\ x_2 + x_3 + x_6 + x_7 = 0 (\mathrm{mod}\, 2) \\ x_4 + x_5 + x_6 + x_7 = 0 (\mathrm{mod}\, 2) \end{cases}$$

Macierz kontroli parzystości pozwala na obliczenie syndromu s. Syndrom zawiera informację o położeniu błędu w przesłanym słowie kodowym. Syndrom obliczany jest przy wykorzystaniu następującej formuły:

$$s = Hc^T$$

Przykładowo, gdy otrzymamy wektor $c = 0110101$, obliczenia wyglądają następująco:

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

Wektor s^T zawiera informacje o bicie, na którym nastąpiło przekłamanie. Informacja ta jest zapisana z wykorzystaniem kodowania little-endian. W powyższym przykładzie syndrom wskazuje na przekłamanie bitu nr 6

W steganografii stosuje się tzw. kodowanie syndromami. Zabieg ten pozwala na ukrycie większej ilości danych w obrazie, przy jednoczesnym zmniejszeniu ilości wprowadzanych zniekształceń. Przykładowo, wykorzystanie kodu Hamminga (7,4) pozwala na ukrycie 3 bitów wiadomości w 7 bitach nośnika. W tym celu zmieniana jest wartość tylko jednego bitu nośnika. Pojemność (w bitach) standardowej bitmapy o wymiarach h na w pikseli jest w przybliżeniu równa:

$$c = \frac{h \cdot w \cdot 3}{7}$$

Aby ukryć dane, należy zmodyfikować słowo kodowe w taki sposób, aby syndrom obliczony dla zmodyfikowanego słowa kodowego równał się ukrywanej wiadomości. Przykładowo, gdy w słowie kodowym 1001001 chcemy ukryć bity 101:

a. Obliczamy syndrom dla słowa kodowego:

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

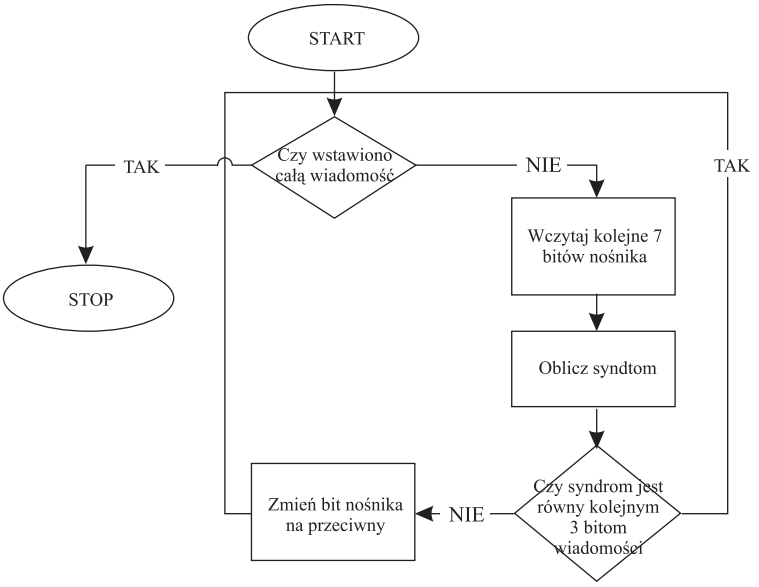
b. Obliczony syndrom $s = 010$ różni się od wiadomości na wszystkich trzech bitach.
c. Zgodnie z układem:

$$x_1 + x_3 + x_5 + x_7 = s_1$$
$$x_2 + x_3 + x_6 + x_7 = s_2$$
$$x_4 + x_5 + x_6 + x_7 = s_3$$

należy zmienić bit x_7 na przeciwny.
d. Nowe słowo kodowe ma postać 1001000.
e. Obliczamy ponownie syndrom:

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

Poniższy rysunek przedstawia schemat blokowy zmodyfikowanego algorytmu LSB:



Rysunek 1. Schemat blokowy zmodyfikowanego algorytmu LSB

Ze względu na sposób interpretacji bitmap przez system Android, w procesie ukrywania wiadomości wykorzystany został drugi najmniej znaczący bit koloru niebieskiego każdego z pikseli.

3. Algorytm kryptograficzny

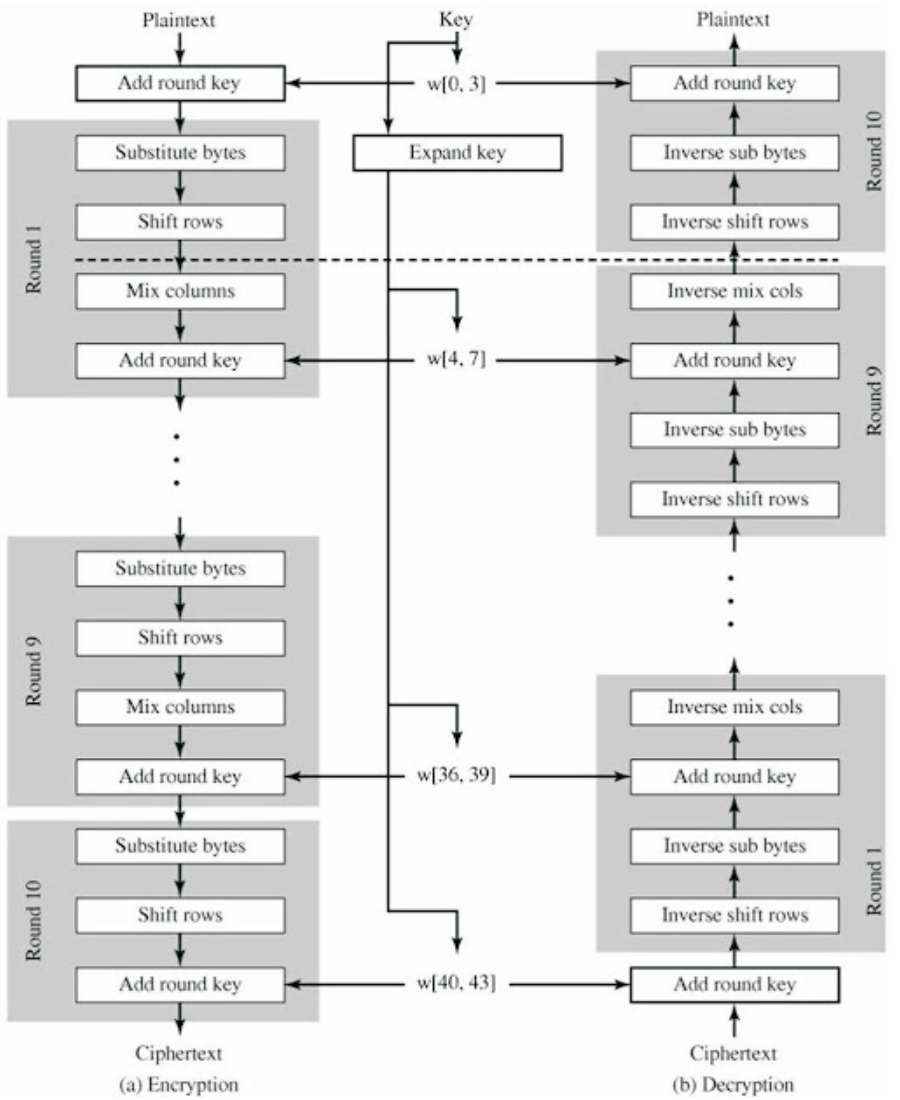
Opracowana aplikacja wykorzystuje symetryczny szyfr blokowy AES z kluczem o długości 128 bitów. AES wykonuje 10 rund podstawieniowo-przestawieniowych. Każda z nich składa się z podstawienia wstępnego, permutacji macierzowej (mieszanie wierszy i mieszanie kolumn) oraz modyfikacji za pomocą klucza. Trybem wybranym na potrzeby wykonanej aplikacji jest tryb wiązania bloków zaszyfrowanych (CBC). Tryb ten wykorzystuje sprzężenie zwrotne i cechuje się samosynchronizacją. W tym trybie blok tekstu jawnego jest sumowany modulo 2 z szyfrogramem poprzedzającego go bloku. Efektem powyższego jest zależność od poprzednich bloków. Pierwszy blok jest sumowany modulo 2 z losowo wygenerowaną wartością początkową IV.

Wybór algorytmu AES był podyktowany szeroką dostępnością implementacji, a także wysokim poziomem bezpieczeństwa potwierdzonym przez międzynarodową społeczność kryptologiczną.

4. Implementacja

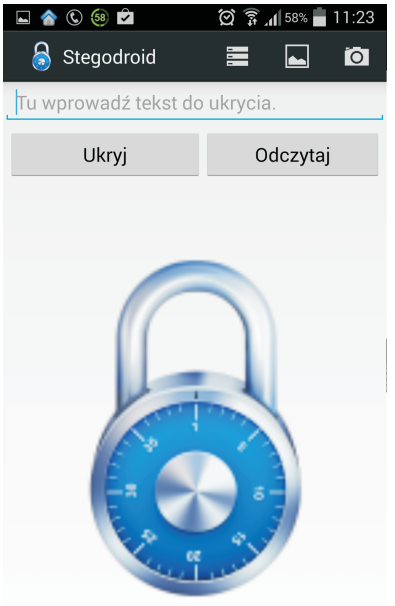
Aplikacja Stegodroid została zbudowana z wykorzystaniem pakietu Android Developer Tools (ADT) v22.3.0-887826. ADT to dodatek do środowiska Eclipse przeznaczony do projektowania aplikacji przeznaczonych dla systemu Android. ADT rozszerza funkcjonalność środowiska Eclipse poprzez łatwe tworzenie nowych projektów systemu Android, tworzenie interfejsu użytkownika, dodawania pakietów bazujących na oficjalnym API, a także łatwe testowanie stworzonych aplikacji. Aplikacje mogą być uruchamiane na wbudowanym emulatorze lub na fizycznym urządzeniu podłączonym do ADT.

Aplikacja Stegodroid może zostać uruchomiona na urządzeniu wyposażonym w system operacyjny Android w wersji 3.2 lub nowszy. Do poprawnej pracy wymagany jest dostęp do zapisu i odczytu pamięci zewnętrznej, a także dostęp do aparatu fotograficznego wbudowanego w urządzenie.
Przez wzgląd na ograniczenia Android API niezbędne było wykorzystanie dwóch bibliotek zewnętrznych. Pierwsza biblioteka aFileDialog jest wykorzystywana do wyświetlenia okna dialogowego, pozwalającego na dokonanie wyboru pliku, który ma przenosić wiadomość. Biblioteka pozwala także

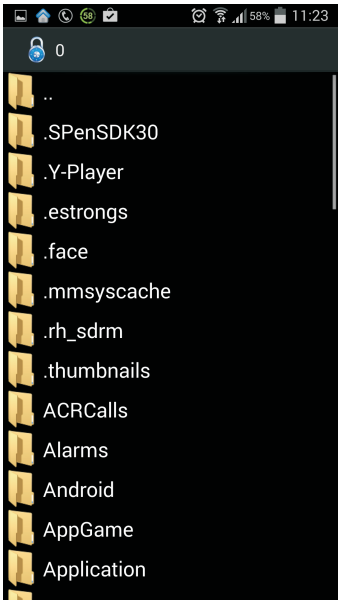


Rysunek 2. Schemat algorytmu AES

na wybór folderów, stosowanie filtrów wyboru przy wykorzystaniu wyrażeń regularnych, a także na tworzenie nowych plików i folderów. Drugą z zastosowanych bibliotek jest mCrypt. Biblioteka ta pozwala na wykorzystanie istniejącej implementacji algorytmu AES w trybie CBC, ze szczególnym uwzględnieniem szyfrowania strumieni tekstu. Biblioteka ta zapewnia pełną kompatybilność ze swoim odpowiednikiem stosowanym w PHP.



Rysunek 3. Ekran główny aplikacji Stegodroid



Rysunek 4. Okno dialogowe wyboru pliku

Aplikacja pozwala na dodanie wiadomości do:

- plików utworzonych wcześniej, dostępnych w pamięci urządzenia,
- obrazów dostępnych w galerii urządzenia,
- zdjęć wykonanych aparatem fotograficznym wbudowanym w urządzenie.

Plik wynikowy jest zapisywany w formacie kompresji bezstratnej PNG w pamięci urządzenia.

5. Uzyskane wyniki

Do sprawdzenia poprawności działania aplikacji Stegodroid wykorzystano telefon komórkowy Samsung GT-i9505. Jako nośnik został wybrany obraz kot.png [1], w którym ukryto 3597 bitów wiadomości. Obrazy poniżej prezentują ukrywanie wiadomości w wybranym obrazie:

Tak otrzymany obraz poddany został stegoanalizie wizualnej i RS. Stegoanaliza wizualna pozwala odkryć zniekształcenia wprowadzone do obrazu przez algorytmy takie jak np. LSB. Poniższe obrazy przedstawiają wyniki dla dwóch najmniej znaczących bitów obrazu.

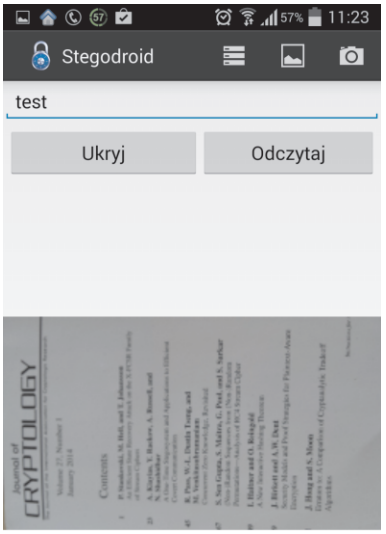
Analiza powyższych obrazów nie pozwala na wyciągnięcie żadnych wniosków na temat umieszczonej wiadomości. W związku z powyższym obraz został poddany także stegoanalizie algorytmem RS. W tym celu posłużono się oprogramowaniem StegSecret [2]. Tabela przedstawia wyniki przeprowadzonego testu.

Tabela 1

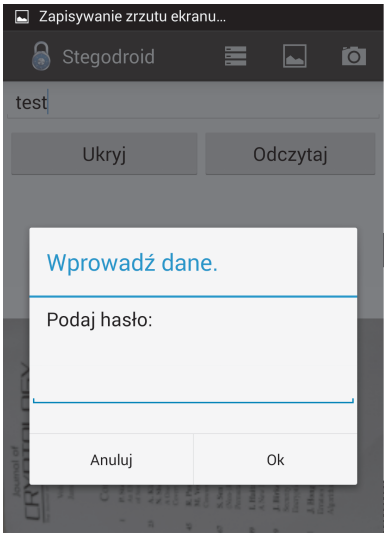
Wyniki analizy algorytmem RS

Obraz	Procent zmienionych bitów pikseli koloru czerwonego	Procent zmienionych bitów pikseli koloru zielonego	Procent zmienionych bitów pikseli koloru niebieskiego
Oryginalny	0,088	0,257	0,018
Stego	0,088	0,257	0,057

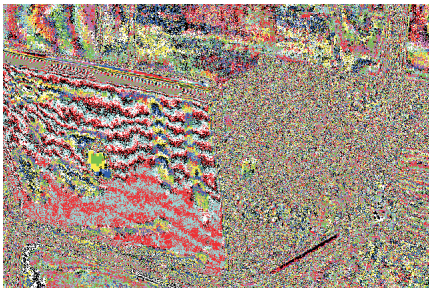
Jak wynika z założeń algorytmu RS [3], [4], można przyjąć, że obraz przenosi ukrytą treść, wtedy, gdy liczba zmienionych pikseli jest większa niż 5%. W przypadku zastosowanego algorytmu uzyskano wynik 0,057% dla koloru niebieskiego, co jest wartością 100-krotnie niższą. Dodatkowo, przeanalizowano obraz oryginalny jak i stegogram przy wykorzystaniu ataku Chi-kwadrat [5]. Na rysunkach 9 i 10 zaprezentowano prawdopodobieństwo umieszczenia wiadomości odpowiednio w obrazie oryginalnym jak i stego-



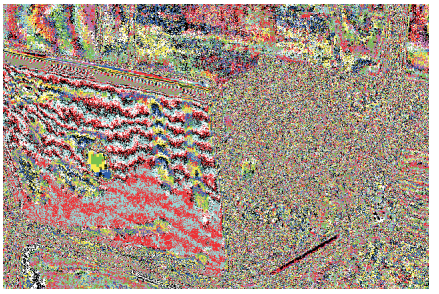
Rysunek 5. Ukrywanie wiadomości w nośniku



Rysunek 6. Ukrywanie wiadomości w nośniku

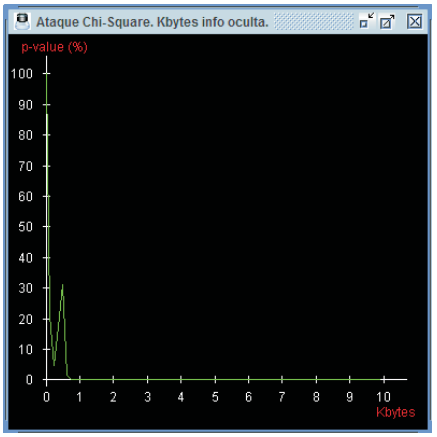


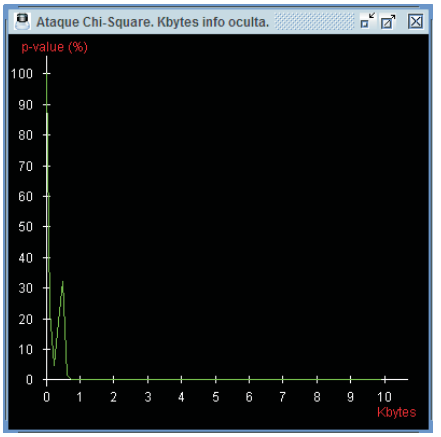
Rysunek 7. LSB obrazu z wbudowaną wiadomością



Rysunek 8. LSB obrazu oryginalnego

gramie. Jak można zauważyć, obydwa wykresy są niemal identyczne, a sam atak określa długość wprowadzonej wiadomości na ok. 100 B.
Ponownie można wyciągnąć wnioski, że tak spreparowany obraz nie przenosi żadnej ukrytej treści. Wymiana wiadomości o rozmiarze 3597 bitów pozostanie zatem niezauważona. Warto nadmienić, że odpowiada to wiadomości o długości 514 znaków.





Rysunek 9, 10. Wyniki ataku chi-kwadrat dla obrazu oryginalnego i stegogramu

6. Wnioski

Przedstawiona w pracy aplikacja Stegodroid pozwala na prowadzenie ukrytej komunikacji użytkownikom urządzeń przenośnych pracujących pod kontrolą systemu Android. Zastosowanie algorytmu steganograficznego o obniżonej liczbie wprowadzanych zmian [10], pozwoliło na znaczące zmniejszenie możliwości wykrycia faktu wprowadzenia danych do nośnika. Dodatkowym składnikiem zwiększającym bezpieczeństwo komunikacji jest zastosowany algorytm szyfrujący
AES-128, który nawet w przypadku odzyskania ukrytych danych nie pozwoli na odtworzenie przenoszonej w nich informacji.

W kolejnych etapach rozwoju aplikacji planuje się wprowadzenie algorytmu steganograficznego dla stratnych formatów kompresji plików graficznych, a także możliwość bezpośredniego przesyłania wytworzonych plików poprzez e-mail, bluetooth, NFC, a także serwisy społecznościowe.

Literatura

[1] http://meinkat.files.wordpress.com/2013/09/cat_using_computer.jpg dostęp 16.07.2014.
[2] http://stegsecret.sourceforge.net/dostęp 11.07.2014.
[3] J. Fridrich, M. Goljan, Rui Du Detecting LSB steganography in color, and grayscale images. IEEE Multimedia 8 pp 22-28, 2001.
[4] J. Fridrich, M. Goljan, D. Hogea, D. Soukal, Quantitative steganalysis of digital images: estimating the secret message length.

Multimedia Systems 9, pp 288-302, 2003.
[5] N. Karsten, Ch. Paget, "GSM: SRSLY?", Chaos Communication Congress, 21.01.2010.
[6] W. Mochnacki, Kody korekcyjne i kryptografia, Oficyna Wydawnicza Politechniki Wrocławskiej, 2000.
[7] J. Gawinecki, J. Szmidt, "Zastosowanie ciał skończonych i krzywych eliptycznych w kryptografii", Wydawnictwo Instytutu Matematyki i Badań Operacyjnych, WAT, 1999.
[8] I. J. Cox, M. L. Miller, J. A. Bloom, J. Fridrich, T. Kalker, "Digital Watermarking and Steganography" (Second Edition), Morgan Kaufmann, 2008.
[9] K. Kaczyński, Steganografia z wykorzystaniem cyklicznych kodów korekcji błędów. Biuletyn WAT, Vol. LXII, Nr 4, 2013, pp. 267–277.
[10] K. Kaczyński, Steganografia z wykorzystaniem optymalnych kodów liniowych, Nowe techniki badań kryminalistycznych a bezpieczeństwo informacji, Wyd. PWN 2014, ISBN 978-83-01-17890-1, str. 110–120.

STEGODROID – MOBILE APPLICATION TO CONDUCT COVERT COMMUNICATION

Abstract. Computers and mobile phones are tools used in almost all areas of daily life. The worldwide computer networks are connecting a personal computers as well as a large data centers and mobile devices such as a mobile phones. Ensuring the security of information systems requires the use of techniques to send data to remote sites in a way that ensures the confidentiality, integrity and availability. These goals are achieved by using an cryptographic and steganographic techniques. Cryptography process messages in such a way that they can not be easily understood, when steganography hides the message, so that their existence is impossible to discover. In some cases, transmission of encrypted messages can arouse suspicions, while messages sent using steganography will remain undetected. Both methods can be combined, so that the transmitted message can be protected in two ways. In this case, if the steganography fails and the message is detected, the content of the message will be still protected by cryptography. In this paper we described our implementation of proprietary steganographic algorithm using linear error correction codes. The developed application is designed for devices running under Android OS, thus there is a wide range of customers for such a software.

Keywords: steganography, cryptography, Android