STUDIA INFORMATICA 2001
Volume 22 Number 1 (43)

Katarzyna TRYBICKA-FRANCIK
Politechnika Śląska, Instytut Informatyki

WYKORZYSTANIE WIZUALIZACJI ALGORYTMÓW W PROCESIE OCENY JAKOŚCI SZYFRÓW

Streszczenie. W artykule zostały przedstawione dwa przykłady wykorzystania wizualizacji algorytmów w procesie oceny jakości szyfrów. Algorytm IDEA użyty do eksperymentu zaliczany jest do jednych z najbezpieczniejszych w swojej klasie. Uzyskane wyniki pozwalają twierdzić, że metody oceny jakości szyfrów oparte na wizualizacji algorytmów mają przed sobą przyszłość.

APPLICATION OF SOFTWARE VISUALIZATION IN ESTIMATING THE QUALITY OF CRYPTOGRAPHIC ALGORITHMS

Summary. The paper presents two sample applications of software visualization in the field of estimating the quality of cryptographic algorithms. The IDEA algorithm, proposed as an object of the research, is supposed to be one of the safest in its class. The results show, that the quality estimation methods that apply software visualization techniques are successful in cryptography.

1. Wstęp

Ocena jakości algorytmów kryptograficznych nie jest zadaniem banalnym. Wyrazem tego są wieloletnie, otwarte konkursy, mające wyłonić najlepsze szyfry dla określonych zastosowań. Jak dotąd nie stworzono metodyki badania algorytmów kryptograficznych, natomiast dość dobrze zdefiniowano wymagania, jakie powinny one spełniać. Wśród nich znajdują się następujące założenia [1]:

- Algorytm musi zapewniać wysoki stopień bezpieczeństwa; powinien być odporny na znane metody kryptoanalizy.
- Algorytm musi być kompletny i łatwy do zrozumienia.

- Bezpieczeństwo algorytmu musi opierać się na tajności klucza; bezpieczeństwo to nie powinno zależeć od utajnienia algorytmu.
- Algorytm musi zapewniać losowość wyjścia.

Pozostałe z wymagań odnoszących się do budowy algorytmów kryptograficznych dotyczą ich wydajności, wymagań pamięciowych, łatwości implementacji dla różnych platform programowych i sprzętowych, sposobu licencjonowania itp. Problematyka ta wykracza poza zakres tego artykułu.

W artykule omówione zostały wyniki prac nad wykorzystaniem wizualizacji algorytmów w procesie oceny jakości algorytmów kryptograficznych. Jest to kontynuacja wcześniej zasygnalizowanego kierunku badań [2, 3].

2. Wizualizacja

Terminy „wizualizacja algorytmów, programów bądź oprogramowania" są dość intuicyjne. Kojarzą się z próbą przedstawienia wybranych cech algorytmu lub oprogramowania przy zastosowaniu odpowiedniej reprezentacji graficznej. Jest to prawda, choć w istocie termin „wizualizacja" (w sensie *visualisation*) jest znacznie szerszy znaczeniowo. Za słownikiem Webstera [5], **wizualizacja** to „formowanie mentalnego obrazu (org. ang. *mental image*) czegoś aktualnie niewidocznego, abstrakcji". Oznacza to, że wizualizacja nie jest tylko ograniczona do reprezentacji graficznej, obraz mentalny może być formowany jako rezultat danych wejściowych dowolnego ze zmysłów.

Wizualizację oprogramowania można więc zdefiniować jako zastosowanie różnorodnych, zaawansowanych technik komputerowych przez człowieka [6, 7]. Techniki wykorzystywane w ramach wizualizacji to ogólnie multimedia.

Wizualizacja programów jest odmianą wizualizacji oprogramowania koncentrującą się na tych aspektach wizualizowanego oprogramowania, które łączą się z konkretną realizacją, to znaczy – programem komputerowym. W przeciwieństwie do tego **wizualizacja algorytmu** uwidacznia raczej cechy samego algorytmu, w oderwaniu od jego konkretnej realizacji.

3. Wizualizacja w procesie oceny jakości szyfrów

W niniejszym artykule został opisany proces oceny jakości algorytmu kryptograficznego na przykładzie IDEA (*International Data Encryption Standard*) [1]. Szyfr ten został stworzony przez panów Xuejia Lai i Jamesa Masseya i do roku 1995 był uważany za najlepszy i je-

den z najbardziej bezpiecznych algorytmów blokowych spośród udostępnionych szerokiej publiczności.

IDEA pracuje na 64-bitowych blokach danych. Klucz ma długość 128 bitów. Ten sam algorytm jest stosowany do szyfrowania i odszyfrowania.

3.1. Opis działania IDEA

Na początku blok danych o długości 64 bitów dzielony jest na cztery dwubajtowe podbloki X_1, X_2, X_3, X_4. Te cztery podbloki stanowią wejście do pierwszego cyklu algorytmu. Podobnych cykli w algorytmie jest osiem. By móc rozpocząć pracę konieczne jest też wygenerowanie ze 128-bitowego klucza dwubajtowych podkluczy. W tym celu klucz dzielony jest na osiem części, które są pierwszymi podkluczami. Sześć pierwszych stanowi wejście dla pierwszego cyklu algorytmu, dwa pozostałe będą wykorzystane w kolejnym cyklu jako dwa pierwsze podklucze. Kolejne brakujące podklucze wyznacza się podobnie jak pierwsze, przy czym klucz jest przesuwany cyklicznie o 25 bitów w lewo i ponownie dzielony na osiem podkluczy. Procedura ta jest powtarzana aż do końca szyfrowania. Mając podklucze i podbloki danych możemy rozpocząć proces szyfrowania.

Na jeden cykl szyfru składają się następujące kroki:

1 - mnożymy X_1 przez pierwszy podblok klucza,
2 - dodajemy X_2 do drugiego podbloku klucza,
3 - dodajemy X_3 do trzeciego podbloku klucza,
4 - mnożymy X_4 przez czwarty podblok klucza,
5 - dodajemy modulo 2 wyniki kroków (1) i (3),
6 - dodajemy modulo 2 wyniki kroków (2) i (4),
7 - mnożymy wynik kroku (5) przez piąty podblok klucza,
8 - dodajemy wyniki kroków (6) i (7),
9 - mnożymy wynik kroku (8) przez szósty podblok klucza,
10 - dodajemy wyniki kroków (7) i (9),
11 - dodajemy modulo 2 wyniki kroków (1) i (9),
12 - dodajemy modulo 2 wyniki kroków (3) i (9),
13 - dodajemy modulo 2 wyniki kroków (2) i (10),
14 - dodajemy modulo 2 wyniki kroków (4) i (10).

Na wyjściu cyklu otrzymujemy cztery podbloki będące wynikiem kroków (11), (12), (13) i (14). Po zamianie miejscami dwóch wewnętrznych podbloków (w ostatnim cyklu nie wykonuje się tej operacji) otrzymujemy blok wejściowy do następnego cyklu.

Po ósmym cyklu wykonywane są następujące kroki:

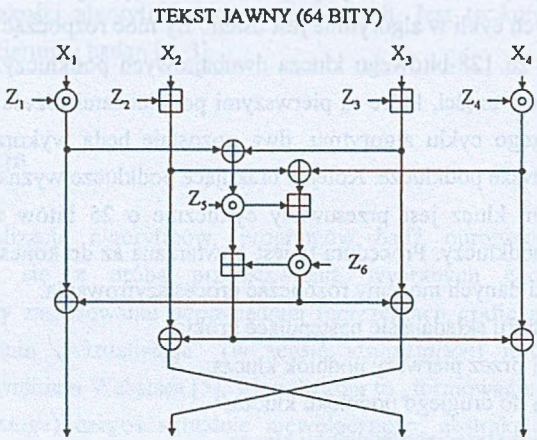
1 - mnożymy X_1 przez pierwszy podklucz,
2 - dodajemy X_2 do drugiego podklucza,

3 - dodajemy X_3 do trzeciego podklucza,

4 - mnożymy X_4 przez czwarty podklucz.

Ostatecznie, otrzymane w wyniku przedstawionych operacji cztery podbloki są łączone w jeden blok szyfrogramu.

Po przeczytaniu opisu działania algorytmu widać, że znacznie czytelniejszą formą przekazu dla ludzkiej percepcji jest obraz (rys. 1).



Rys. 1. Schemat jednego cyklu w algorytmie IDEA
Fig. 1. Schema of a single cycle of the IDEA algorithm

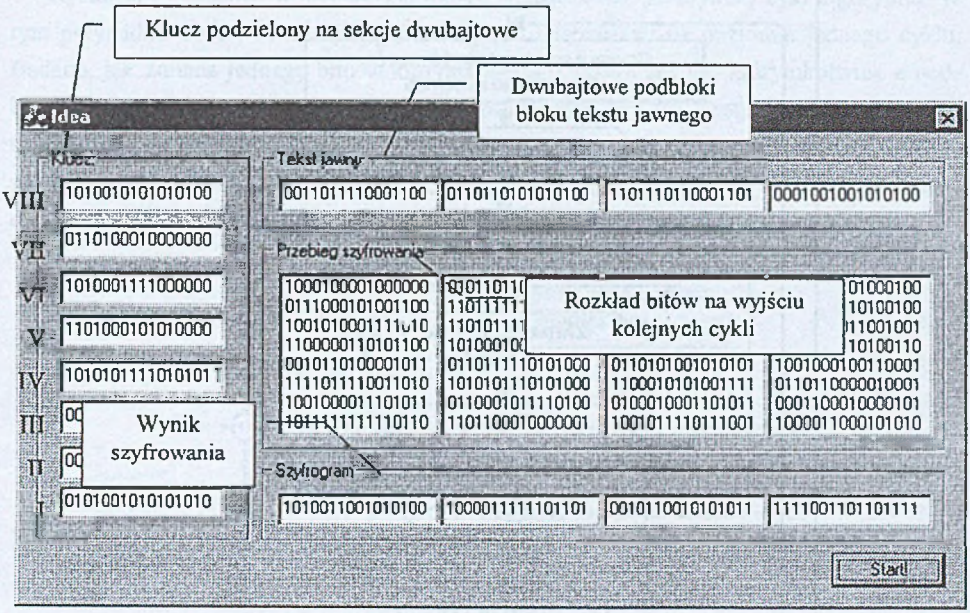
Rysunek statyczny daje ogólne pojęcie o działaniu algorytmu, nie pozwala jednak wychwycić niuansów funkcjonalnych.

3.2. Metoda badania algorytmów kryptograficznych

Jednym z podstawowych założeń, jakie musi spełniać algorytm kryptograficzny, jest odporność na znane metody kryptoanalizy. Techniki przełamywania szyfrów są dokładnie udokumentowane (przynajmniej te, o istnieniu których wiemy). Znamy zatem cechy, które sprawiają, że algorytm kryptograficzny jest podatny na daną metodę ataku. To oznacza również, że twórcy nowych szyfrów wiedzą, jakich błędów należy się wystrzegać. Zaproponowana metoda badania może być pomocna przy lokalizacji takich usterek, ale przede wszystkim pozwala na wykrywanie wad, które mogą składać się na słabość szyfru i to zarówno w procesie samej analizy algorytmu, jak i późniejszej jego implementacji.

Kolejnym istotnym czynnikiem wpływającym na jakość szyfratora jest tak zwany efekt lawinowy. Polega on na tym, że zmiana jednego bitu klucza, przy stałym ciągu wejściowym, powoduje liczne zmiany bitów w ciągu wynikowym. Podobny rezultat powinniśmy uzyskać zmieniając pojedynczy bit w ciągu wejściowym przy stałym kluczu. Ważne jest, by analiza tych zmian nie pozwoliła kryptoanalitykowi w rozsądnym czasie określić, jak wygląda klucz lub tekst jawny [4].

Algorytmy kryptograficzne konstruowane są tak, by na ich wyjściu, niezależnie od kształtu danych wejściowych (tekst jawny i klucz), rozkład ilościowy zer i jedynek był równomierny. Rysunek 2 przedstawia wizualizację algorytmu IDEA, która pozwala na obserwację efektu lawinowego.

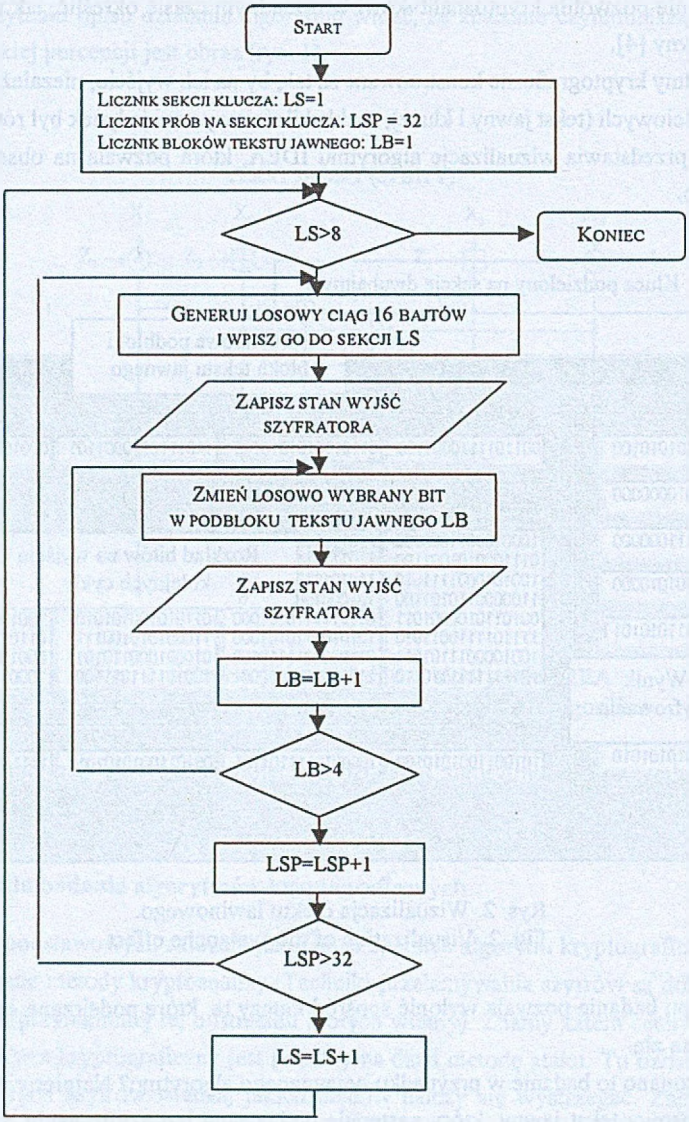


Rys. 2. Wizualizacja efektu lawinowego.
Fig. 2. Visualization of the avalanche effect

Tego typu badanie pozwala wyłonić spośród kluczy te, które podejrzane są o słabość lub ewidentnie są złe.

Jak wykonano to badanie w przypadku omawianego algorytmu? Najpierw wygenerowano w sposób losowy tekst jawny, który następnie poddawany był szyfrowaniu. By sprawować lepszą kontrolę nad kluczem podzielono go dla celów wizualizacji na osiem dwubajtowych sekcji (na rysunku 2 oznaczone od I do VIII). Następnie próbkowano każdą z sekcji, obserwując, jak zmiany bitów w kluczu wpływają na wyjście szyfratora. Ponieważ istotne znacze-

nie dla postaci szyfrogramu ma również kształt tekstu jawnego, sprawdzono, jaki wpływ na postać szyfrogramu ma zmiana jednego bitu w podblokach tekstu jawnego. By uzyskać pełniejszy obraz połączono oba eksperymenty w jeden. Schemat sposobu przeprowadzania badanie przedstawia rysunek 3.

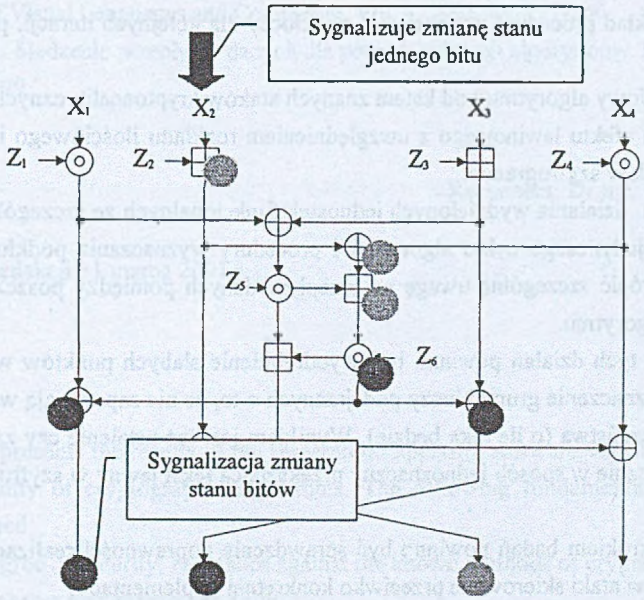


Rys. 3. Procedura testowania
Fig. 3. Testing procedure

Ta stosunkowo niewielka liczba testów (8 sekcji klucza × 32 kombinacje bitów × 4 podbloki tekstu jawnego = 1024) jest wystarczająca, by wyrobić sobie pogląd, jakie grupy bitów powinny być poddane dalszej obserwacji.

Dobry rozkład bitów na wyjściu szyfratora nie jest wystarczającą przesłanką, by stwierdzić, że algorytm spełnia stawiane przed nim wymogi bezpieczeństwa. Ważne są kroki pośrednie. W przypadku algorytmu IDEA można, jak już pisano wcześniej, wyróżnić osiem cykli. Zbadanie wzajemnych relacji między danymi wejściowymi i wyjściowymi w pojedynczym cyklu może ujawnić wady konstrukcyjne samego algorytmu lub też wskazać sytuacje szczególne, w których ogólnie zadowalająco działający algorytm okazuje się być wyjątkowo podatny na atak.

Rysunek 4 pokazuje, w jaki sposób można wizualizować pojedynczy cykl algorytmu. W tym przypadku obserwowano przepływ danych i lawinowość na poziomie jednego cyklu. Badano, jak zmiana jednego bitu w którymkolwiek z podkluczy lub którymkolwiek z podbloków tekstu jawnego wpływa na wyjście z pojedynczego cyklu. Testy przeprowadzone na tym poziomie były bardzo podobne do opisanych wyżej, jednak sposób reprezentacji graficznej był odmienny. Na tym etapie istotna była zmiana stanu a nie postać ciągu wynikowego, dlatego zastosowano sygnalizację barwną.



Rys. 4. Wizualizacja efektu lawinowego dla jednego cyklu
Fig. 4. Visualization of the avalanche effect in a single cycle

Na rysunku strzałki sygnalizują miejsca, gdzie została wymuszona zmiana. Barwne kółka odzwierciedlają następstwa wprowadzonej zmiany, a stopień wysycenia barwy odpowiada procentowej liczbie bitów, które uległy zmianie.

Przedstawione do tej pory testy odnosiły się bezpośrednio do idei działania algorytmu. Brak w nich odniesienia do faktycznej implementacji. Doświadczenia minionych lat jednoznacznie sugerują, że nawet najlepszy szyfr źle zaimplementowany może stać się jedynie bezużytecznym gadżetem. Problem ten wykracza poza ramy tego artykułu.

4. Wnioski i perspektywy

Zaproponowana technika badania algorytmów wymaga określenia przez obserwatora tego, co, dlaczego i w jaki sposób wizualizować. Działania te można dość dobrze usystematyzować. Proponowana metodyka:

1. Zapoznanie się z ideą działania algorytmu. Techniki wizualizacyjne są bardzo pomocne w procesie zrozumienia działania algorytmu. Pomagają odpowiedzieć na pytanie *jak*, ale również *dlaczego*.
2. Analiza funkcjonalna z wyodrębnieniem oddzielnych jednostek funkcjonalnych, takich, jak na przykład procedura wyznaczania podkluczy dla kolejnych iteracji, pojedyncza iteracja itp.
3. Analiza budowy algorytmu pod kątem znanych ataków kryptoanalitycznych.
4. Obserwacja efektu lawinowego z uwzględnieniem rozkładu ilościowego i jakościowego zer i jedynek w szyfrogramie.
5. Obserwacja działania wydzielonych jednostek funkcjonalnych ze szczególnym uwzględnieniem pojedynczego cyklu algorytmu i procedury wyznaczania podkluczy dla cykli. Należy zwrócić szczególną uwagę na przepływ danych pomiędzy poszczególnymi elementami algorytmu.

Rezultatem tych działań powinno być wyodrębnienie słabych punktów w samym algorytmie oraz wyznaczenie grupy kluczy podejrzanych o to, że nie zapewniają wystarczającego stopnia bezpieczeństwa (o ile taka będzie). Wynikiem jest też ustalenie czy zaprezentowany algorytm faktycznie w sposób jednoznaczny przekształca tekst jawny w szyfrogram przy stałym kluczu.

Kolejnym krokiem badań powinno być sprawdzenie poprawności realizacji algorytmu i jej odporności na ataki skierowane przeciwko konkretnej implementacji.

Dotychczasowe wizualizacyjne eksperymenty przeprowadzone na algorytmach kryptograficznych pozwalają postawić i obronić tezę, że techniki wizualizacji algorytmów są po-

mocne w procesie analizy jakościowej algorytmów kryptograficznych, ułatwiają detekcję ich słabych punktów i pozwalają wyeliminować algorytmy ewidentnie złe.

LITERATURA

1. Schneier B.: Kryptografia dla praktyków. Protokoły, algorytmy i programy źródłowe w języku C. Wydawnictwa Naukowo-Techniczne, Warszawa 1995.
2. Francik J., Trybicka-Francik K.: Zastosowanie wizualizacji algorytmów w kryptologii. VII Konferencja Sieci Komputerowych, Studia Informatica, seria Informatyka nr 1 (39), v. 21, Zakopane 2000.
3. Praca zbiorowa pod redakcją Grzywaka A.: Bezpieczeństwo systemów komputerowych. Wydawnictwo Pracowni Komputerowej Jacka Skalmierskiego, Gliwice 2000.
4. Denning D. E.: Kryptografia i ochrona danych. Wydawnictwa Naukowo-Techniczne, Warszawa 1992.
5. The Webster's New Word Dictionary® on PowerCD®. 3rd College Edition. Simon & Schster Inc. 1994, wydanie multimedialne na CD-ROM, wersja 2.1, Zane Publishing Inc.
6. Price A., Baecker R. M., Small I. S.: A Principled Taxonomy of software Visualization. Journal of Visual Languages and Computing, Vol. 4, 1993, str. 211-266.
7. Francik J.: Śledzenie przepływu danych dla potrzeb animacji algorytmów. Rozprawa doktorska, 1999.

Recenzent: Dr inż. Andrzej Białas

Wpłynęło do Redakcji 31 marca 2001 r.

Abstract

The article presents the results of the research on applying software visualization in estimating the quality of cryptographic algorithms. The following fundamental requirements have been defined:

- High degree of security: resistance against the known methods of cryptanalysis,
- Completeness and comprehensibility,
- Security based on the secret key, not secret algorithm,
- Randomized output generated.

In the following part of the paper the basic notions in the field of software visualization have been defined. In the main part the visualizations and testing techniques are described.

As an object of the research a popular, widely-known IDEA algorithm has been proposed (fig. 1). One of the visualizations, although quite naïve at first sight (fig. 2), made it possible to observe the avalanche effect. After merely 1032 experiments allowed to precisely define the group of the suspected keys – i.e. the keys that do not encrypt messages or produce ciphers that are not explicit. These results were the starting point for a more advanced visualization, in which the data flow and avalanche effect have been observed on the level of a single IDEA iteration (fig. 3). The color signalization techniques have been applied to show the changes of states as well as bit-level data flow.

The conclusion is that visualization techniques may be a powerful tool in cryptography: they make it possible to estimate cryptographic algorithms, to detect their weak points and to eliminate the algorithms that are not of high quality.