Realizacja szyfru „bezkluczowego” `c80k395` do kryptograficznej ochrony załączników poczty elektronicznej w środowisku Maple

Maple implementation of „Keyless” Cipher **`c80k395`** *for cryptographic protection of e-mail enclosures*

Czesław Kościelny[1]

Treść. Opisano przykład aplikacji typu workhsheet, uruchamianej w środowisku Maple i realizującej procedury „bezkluczowego” szyfrowania i deszyfrowana plików za pomocą symetrycznego szyfru plikowego **`c80k395`**. Aplikacja posiada prosty graficzny interfejs użytkownika i przeznaczona jest głównie do kryptograficznej ochrony załączników poczty elektronicznej.

Słowa kluczowe: kodowanie transportowe, funkcja Maple **`convert/base`**, szyfrowanie symetryczne plików.

Abstract. An example of a Maple worksheet application which performs the „keyless” file encryption or decryption by means of the symmetric cipher **`c80k395`** has been presented. The application has a simple graphical user interface and may be used mainly for cryptographic protection of e-mail enclosures.

Keywords: Base 64 Encoding Scheme, Maple **`convert/base`** built-in function, Symmetric File Encryption.

1. Wstęp

Autor zauważył, że nieliniowe przekształcenia stosowane w systemach kodowania transportowego, opisanego w dokumencie RFC 4648, można zastosować do konstrukcji nowej rodziny szyfrów, które nie są ani szyframi strumieniowymi, ani szyframi blokowymi, aktualnie stosowanymi w kryptografii klasycznej [1]. Aby taki nowy szyfr skonstruować wystarczy algorytmy szyfrowania i deszyfrow ania zastąpić odpowiednio zmodyfikowanymi algorytmami kodowania i dekodowania transportowego. Jak wiadomo, procedura realizująca algorytm kodowania transportowego według tymczasowej normy internetowej RFC 4648 [2] generuje plik zawierający ściśle określony zestaw znaków siedmiobitowego kodu ASCII a każdemu znakowi odpowiada ustalona sekwencja 4, 5 lub 6 bitów. Procedura dekodowania poprawnie dekoduje taki plik zakodowany, w którym zachowane są reguły kodowania transportowego. Ten określony i skonfigurowany zgodnie z normą RFC zestaw znaków nazywa się alfabetem. Wyposażając procedury kodowania i dekodowania transportowego w parametr formalny, zwany kluczem, umożliwiający zmianę stosowanego alfabetu, otrzymuje się wygodne narzędzie do szyfrowania i deszyfrowania plików. Tworzone w ten sposób szyfry można nazwać symetrycznymi szyframi plikowymi, ponieważ procedury realizujące algorytm szyfrujący i algorytm deszyfrujący posiadają dwa parametry formalne: nazwę pliku do zaszyfrowania lub zdeszyfrowania oraz klucz, który jak zwykle w przypadku szyfru symetrycznego jest taki sam dla procedury szyfrującej oraz dla procedury deszyfrującej. Zaproponowano też, aby szyfrom nadawać nazwę **`cxky`**, gdzie po literze **`c`**, oznaczającej znak (character), umieszczona jest liczba znaków **`x`**, występujących w pliku zaszyfrowanym, a litera **`k`** (key) poprzedza liczbę bitów **`y`**, charakteryzującą moc szyfru. W opisywanym programie szyfrująco-deszyfrującym zastosowano funkcję biblioteczną programu Maple o nazwie **`convert/base`**. Jest to narzędzie bardziej universalne niż opisywane w normie RFC 4848 kodowanie transportowe, pozwala bowiem stosować dowolne wartości bazy. Poza tym ustalono, że w zaszyfrowanym pliku będzie występować 10 cyfr oraz 52 duże i małe litery polskiego alfabetu, łącznie 80 znaków drukowalnych. W takiej sytuacji tajnym kluczem szyfrującym jest lista, występująca w kodzie źródłowym pod nazwą **`a`**, zawierająca 80 wartości bajtowych znaków pokazanych na rys. 1. Zatem przestrzeń klucza posiada 80! elementów, czyli w przybliżeniu

$$7.156945705\ 10^{118}.$$

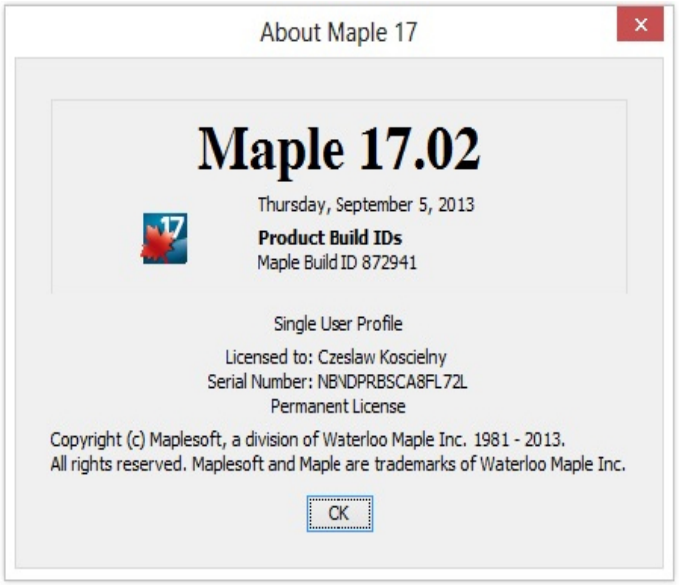
W zapisie binarnym jest to liczba 395-bitowa, można więc powiedzieć, że aplikacja stosuje tajny klucz o tej właśnie długości. Aplikację można zaprogramować albo w sposób tradycyjny, publikując jej kod źródłowy i stosując tajny klucz, albo klucz „wmontować" do aplikacji, którą wtedy należy stosownie chronić. W tym drugim przypadku aplikację można nazwać szyfrem „bezkluczowym", ponieważ to narzędzie realizuje procedury szyfrowania i deszyfrowania oraz pełni też rolę tajnego klucza. Ten „bezkluczowy” sposób szyfrowany jest wygodny dla wielu użytkowników i na ogół stosowany jest w wojsku. Aplikacja została napisana w języku Maple, a jej kod źródłowy oszczędnie

1Wydział Informatyki, Wrocławska Wyższa Szkoła Informatyki Stosowanej, ul. Wejherowska 28, 54-239 Wrocław, ckoscielny@horyzont.eu

zapisany zajmuje około 60% jednej strony formatu A4. Dlatego czytelnicy, którzy są zainteresowani szczegółami aplikacji mogą ściągnąć dostępny w internecie [3] plik **c80k395.mw.** W artykule zostanie jeszcze opisany skrótowo tylko kod źródłowy.

2. Kod źródłowy

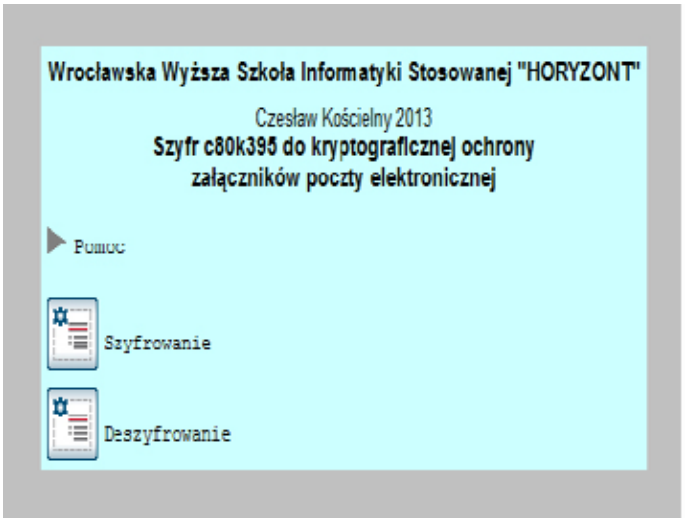
Zadanie szyfru „bezkluczowego" realizuje aplikacja w postaci pliku **c80k395.maplet**, zapisana w najnowszej wersji Maple (Rys. 1), ale aplikację można także uruchomić w wersjach starszych, np. Maple 14.



Rys. 1. Wersja Maple, w której zapisano program [3]
Fig. 1. The Maple version in which the program [3] is implemented

W programie zadeklarowano zmienne **a**, **self2ed**, **ef**, **df**, przy czym:
a − zmienna w postaci listy liczb typu byte, „montująca” tajny klucz w procedurach szyfrującej i deszyfrującej,
self2ed − procedura do wybierania plików, które mają być szyfrowane lub deszyfrowane,
ef − procedura szyfrująca,
df − procedura deszyfrująca.

Aby wydrukować lub obejrzeć kod źródłowy należy plik **c80k395.mw** otworzyć w sesji Maple. Program jest nieskomplikowany i wyjątkowo prosty w obsłudze: użytkownik nie musi znać języka Maple lecz powinien umieć posługiwać się myszą i jej klawiszami. Jak wspomniano, aplikację zaprogramowano w postaci pliku **c80k395.mw.** Wystarczy ten plik dwukrotnie kliknąć w komputerze z zainstalowanym programem Maple i wtedy zostanie wygenerowane okno interfejsu graficznego, pokazane na rys. 2. Po kliknięciu trójkąta obok napisu `Pomoc` ukaże się krótki opis aplikacji i sposobu jej użycia. Aplikacja musi posiadać uprawnienia do modyfikowania, usuwania, odczytu i zapisu przetwarzanych plików.



Rys. 2. Interfejs graficzny użytkownika aplikacji
Fig. 2. The graphical user interface of the application

Po uruchomieniu aplikacji użytkownik wybiera plik do zaszyfrowania lub zdeszyfrowania, a po zakończeniu pracy programu pojawi się szczegółowy opis wykonanych operacji.
Jeśli Czytelnik nie posiada zainstalowanego programu Maple i ma zamiar wypróbować jak działa opisana aplikacja, powinien zainstalować okrojoną, bezpłatną wersję programu Maple, czyli program Maple Player (http://www.maplesoft.com/products/maple/mapleplayer/). W tym środowisku działa aplikacja **80k395mp.mw**, nieco mniej wygodna dla użytkownika.

3. Podsumowanie i wnioski

Przedstawiono zaawansowaną aplikację zrealizowaną w środowisku Maple, która powinna być stosowana przede wszystkim do kryptograficznej ochrony załączników poczty elektronicznej (w postaci plików o dowolnym formacie) przed nieupoważnionym dostępem, ponieważ wygenerowane za pomocą aplikacji kryptogramy plików zawierają wyłącznie 80 znaków drukowalnych 7-bitowego kodu ASCII a takie załączniki są akceptowane przez wszystkie systemy internetowej poczty. Aplikacja używa szyfru z 395-bitowym kluczem, zaś algorytmy kryptograficzne oparto na transformacjach używanych w funkcji bibliotecznej **convert/base** programu Maple. Dzięki temu szyfr jest w wysokim stopniu randomizowany i jedynym znanym aktualnie autorowi sposobem łamania szyfru jest wypróbowywanie po kolei wszystkich kluczy kryptograficznych. Aplikacja jest wersją demonstracyjną, dużo wolniejszą od rozwiązań zrealizowanych w językach kompilowanych , mimo to nie stwarza istotnych problemów, jeśli rozmiar szyfrowanych plików nie przekracza 10 kB. Warto zauważyć, że rozmiar wygenerowanych plików zaszyfrowanych za pomocą aplikacji jest około 26,5% większy od rozmiaru pliku niezaszyfrowanego, podczas gdy aplikacja [1] generuje pliki kryptogramów z kluczem o 99 bitów krótszym i o 33% większe od plików niezaszy-

frowanych. Jest rzeczą oczywistą, że podobne aplikacje można realizować dla dowolnych wartości bazy, akceptowanych przez funkcję biblioteczną **`convert/base`** za pomocą których można także szyfrować pliki, zapamiętywane w dyskowych bazach danych. W tym przypadku warto zapisane w języku Maple algorytmy zrealizować w języku kompilowanym.

Podziękowanie

Autor pragnie podziękować anonimowemu recenzentowi za cenne uwagi, które pomogły usunąć usterki programu.

Literatura (References)

[1] C. Kościelny, Base 64 "Keyless" File Encryption, http://www.maplesoft.com/applications/view.aspx?SID=145918, 04.2013

[2] S. Josefsson, The Base16, Base32, and Base64 Data Encodings, http://www.h-online.com/nettools/rfc/rfcs/rfc4648.shtml , 10.2006

[3] C. Kościelny, Aplikacje **`c80k395.mw`** i **`c80k395mp.mw`**, link na portalu WWSIS: www.wydawnictwo.horyzont.eu/podstrony/publikacje.html