BIULETYN INSTYTUTU AUTOMATYKI I ROBOTYKI
NR 33, 2012

Metoda bezpiecznej wymiany danych z wykorzystaniem nośników Flash RAM[1]

Jan CHUDZIKIEWICZ, Janusz FURTAK

Instytut Teleinformatyki i Automatyki WAT,
ul. Gen. S. Kaliskiego 2, 00-908 Warszawa
j.chudzikiewicz@ita.wat.edu.pl, j.furtak@ita.wat.edu.pl

STRESZCZENIE: W artykule zaprezentowano metodę pozwalającą na przekazywanie, w sposób bezpieczny, plików z użyciem pamięci Flash RAM poprzez kanał niezabezpieczony, np. kuriera albo tradycyjny system pocztowy. Metoda bazuje na sterowniku typu *filter-driver* i wykorzystuje zarówno symetryczne jak i asymetryczne szyfrowanie. Zaprezentowane rozwiązanie pozwala nadawcy na określenie odbiorcy pliku, jak również jednoznaczną identyfikację nadawcy przez odbiorcę pliku.

SŁÓWA KLUCZOWE: sterownik urządzenia, algorytm szyfrujący, system plików, sterownik filtrujący.

1. Wprowadzenie

Coraz większa popularność pamięci Flash RAM, jako nośników danych, wymusza konieczność stosowania mechanizmów gwarantujących odpowiedni poziom zabezpieczenia danych na nich przechowywanych. Jest to szczególnie istotne w przypadku tzw. danych wrażliwych, mających istotny wpływ na bezpieczeństwo instytucji.

Do tego celu najczęściej używane jest oprogramowanie (np. *USB Flash Security*, *Secure Traveler*, ***Rohos Mini Drive* itp.)**, które należy zainstalować na nośniku przed jego użyciem [4], [5]. W trakcie instalacji takiego oprogramowania w pamięci Flash RAM jest tworzony szyfrowany wolumin, do którego uzyskuje się dostęp, po podaniu wcześniej zdefiniowanego hasła. Moc

[1] Materiały zawarte w artykule zostały zaprezentowane na Konferencji Systemy Czasu Rzeczywistego 2012.

zabezpieczenia nośnika, z wykorzystaniem tego typu oprogramowania, zależy od stosowanego symetrycznego algorytmu szyfrowania i długości klucza. Ten typ zabezpieczenia jest wystarczający w przypadku na przykład zagubienia lub kradzieży nośnika. Wykorzystanie takiego rozwiązania, do przekazywania danych pomiędzy różnymi podmiotami używającymi takiego nośnika danych, stwarza problemy wynikające głównie z konieczności przekazania wraz z zabezpieczonym nośnikiem klucza szyfrowania, za pomocą którego nośnik został zaszyfrowany. Ponadto nadawca danych nie ma pewności, że dane będą dostępne tylko dla adresata, a adresat nie ma pewności, że otrzymał dane od spodziewanego nadawcy.

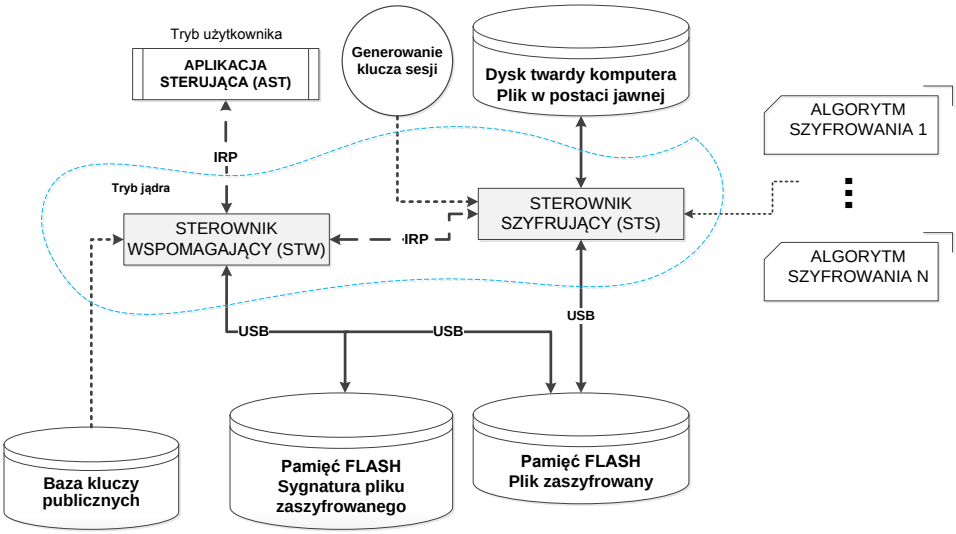
W artykule przedstawiono rozwiązanie pozwalające na takie przygotowanie danych zapisanych na nośnik pamięci Flash RAM, aby ten nośnik mógł być użyty do bezpiecznego przekazywania plików danych, w trakcie którego nadawca danych (tzn. twórca zabezpieczonej zawartości nośnika) ma pewność, że dane będą dostępne tylko dla wskazanego odbiorcy, a odbiorca ma pewność, że otrzymał dane od spodziewanego nadawcy. Opisywany mechanizm został zaimplementowany dla systemu Windows i wykorzystuje algorytmy szyfrowania symetrycznego i asymetrycznego. Zaprezentowane rozwiązanie opiera swoje działanie na sterowniku typu *filter-driver* [1], [2], [7], [9], który był prezentowany na XVII Konferencji Systemów Czasu Rzeczywistego. W rozwiązaniu zakłada się, że z punktu widzenia systemu operacyjnego, dane mogą być przetwarzane dwukierunkowo: z postaci jawnej, zapisanej na dysku twardym komputera na postać zabezpieczoną na nośniku wymiennym (np. w pamięci Flash RAM lub dysku twardym) podłączonym do systemu przez magistralę USB (ang. *Universal Serial Bus*) i odwrotnie – z zabezpieczonego nośnika wymiennego na postać jawną na dysku twardym. Nie dopuszcza się natomiast możliwości wykorzystania oprogramowania do bezpośredniej wymiany plików danych pomiędzy nośnikami wymiennymi (np. pamięciami typu Flash RAM) przyłączonymi do systemu poprzez magistralę USB.

Proces zabezpieczenia wymiany danych powinien spełniać następujące wymagania:

- powinien być realizowany w sposób przeźroczysty dla użytkownika;
- nie powinien powodować zauważalnego dla użytkownika obciążenia systemu operacyjnego;
- nie powinien mieć istotnego wpływu na szybkość odczytu i zapisu danych na nośnikach danych;
- powinien umożliwiać wykorzystanie różnych algorytmów szyfrowania danych zapewniających wymagany poziom poufności danych;
- powinien tworzyć na nośniku wymiennym zabezpieczony plik i sygnaturę dla tego pliku;

– powinien umożliwiać wykonywanie wszelkich operacji dopuszczalnych dla nośników danych, takich jak sprawdzanie powierzchni woluminu pod kątem błędów, czy defragmentację danych na dyskach.

Spełnienie tych wymagań wymusza wykorzystanie do realizacji procesu zabezpieczenia danych wydzielonych modułów (sterowników) systemu operacyjnego działających na poziomie jądra systemu [3], [4], [7], [9], [11]. Schemat poglądowy opracowanego rozwiązania przedstawiono na rys. 1.



Rys. 1. Schemat poglądowy systemu zabezpieczenia danych zapisywanych na nośniku wymiennym

Dla użytkownika, opisywane rozwiązanie jest dostępne za pośrednictwem ***Aplikacji Sterującej* (*AST*)**. Głównymi elementami zbudowanego systemu są współpracujące ze sobą sterowniki: szyfrujący [1] oraz wspomagający, który jest sterownikiem zgodnym z modelem WDM (ang. *Windows Driver Model*) [10], [11]. Obydwa elementy pracują w trybie jądra systemu operacyjnego i komunikują się między sobą z wykorzystaniem mechanizmów wewnętrznych systemu operacyjnego oznaczonych na rysunku jako IRP (ang. *Input-Output Request Packed*) [3], [7], [10]. Zadaniem ***Sterownika Szyfrującego* (*STS*)** jest realizacja procesu szyfrowania/deszyfrowania danych oraz wyznaczanie wartości skrótu dla tych danych. Do zadań ***Sterownika Wspomagającego* (*STW*)** należy między innymi wyznaczanie sygnatury dla zabezpieczanych danych oraz pośredniczenie w przesyłaniu komunikatów/poleceń pomiędzy ***STS*** a ***AST***.

Pozostałymi komponentami systemu są: biblioteka .DLL, udostępniająca funkcje z zaimplementowanymi algorytmami szyfrującymi, moduł generacji kluczy sesji oraz baza danych kluczy publicznych użytkowników.

Produktem procesu zabezpieczenia danych są dwa pliki: ***plik z danymi zaszyfrowanymi*** i ***plik z sygnaturą dla zaszyfrowanych danych***. Oba pliki mogą być zapisane na jednym nośniku Flash RAM albo każdy plik na innym nośniku. Wybór miejsca przechowywania pliku z sygnaturą jest określany przez użytkownika za pośrednictwem ***AST***. Należy zwrócić uwagę na to, że zapisywanie zaszyfrowanego pliku i jego sygnatury na oddzielnych nośnikach zwiększa bezpieczeństwo zapisanych danych, ale jest kłopotliwe w użytkowaniu.

2. Proces tworzenia i odczytu zabezpieczonego pliku

W procesie tworzenia zabezpieczonego pliku w wymiennej pamięci Flash RAM (tzn. tworzenia zaszyfrowanego pliku i jego sygnatury) oraz odczytu (deszyfrowania) pliku z wymiennej pamięci Flash RAM niezbędne są atrybuty użytkownika, który tworzy zabezpieczony plik (dalej ten użytkownik będzie nazywany nadawcą), i użytkownika, dla którego jest przeznaczony plik (odbiorca). Przy tworzeniu zabezpieczonego pliku rolę nadawcy pełni zalogowany w systemie użytkownik i on określa odbiorcę pliku za pomocą ***AST***. W przypadku odczytywania zabezpieczonego pliku z wykorzystaniem ***AST*** zalogowany użytkownik pełni rolę odbiorcy, a atrybuty nadawcy są odczytywane po skutecznym odszyfrowaniu sygnatury tego pliku za pomocą klucza prywatnego zalogowanego użytkownika. Dopuszczalną jest sytuacja, w której zalogowany użytkownik jest jednocześnie nadawcą i odbiorcą danych.

Proces tworzenia zabezpieczonego pliku obejmuje etap szyfrowania, a następnie utworzenia sygnatury dla tego pliku. Natomiast proces odczytu zabezpieczonego pliku wymaga w pierwszym kroku pozyskania z sygnatury parametrów niezbędnych do odszyfrowania tego pliku, a w drugim kroku odszyfrowanie tego pliku.

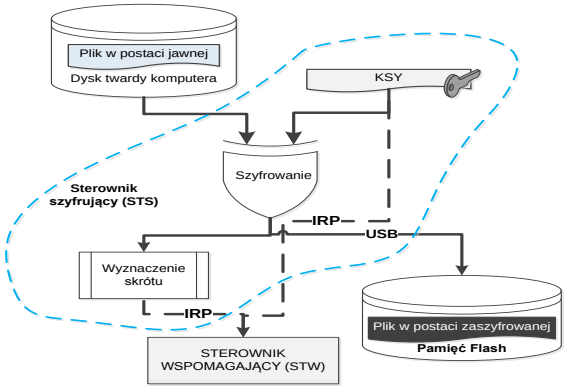
2.1. Tworzenie zabezpieczonego pliku

Proces zapisu pliku obejmuje szyfrowanie pliku z wykorzystaniem klucza KSY. Proces ten realizowany jest przez ***STS***. Działanie sterownika szyfrującego zostało przedstawione w [1]. Schemat opisujący przebieg procesu zapisu pliku przedstawiono na rys. 2. Linią przerywaną zaznaczono operacje wykonywane przez sterownik szyfrujący.

W trakcie realizacji procesu szyfrowania pliku wyznaczana jest dla niego

wartość skrótu zapewniająca integralność tego pliku.

Wyznaczona wartość skrótu oraz klucz sesji KSY po zakończeniu procesu zapisu przekazywane są do ***STW*** celem wygenerowania dla zapisanych danych sygnatury. Proces przesyłania skrótu oraz klucza sesji realizowany jest z wykorzystaniem mechanizmów wewnątrzsystemowych oznaczonych na rys. 2, jako IRP.



Rys. 2. Proces zapisu danych do pamięci Flash RAM

2.2. Wyznaczanie sygnatury

Dla każdego z zabezpieczanych plików generowana jest sygnatura, która zawiera dane niezbędne do jego odczytania. Sygnatura pliku obejmuje następujące pola:

KSY – losowy klucz do szyfrowania/deszyfrowania zabezpieczanego pliku;

SKR – wartość skrótu wyznaczona na postawie zawartości zabezpieczanego pliku po jego zaszyfrowaniu;

ID_SZY – identyfikator algorytmu użytego do szyfrowania;

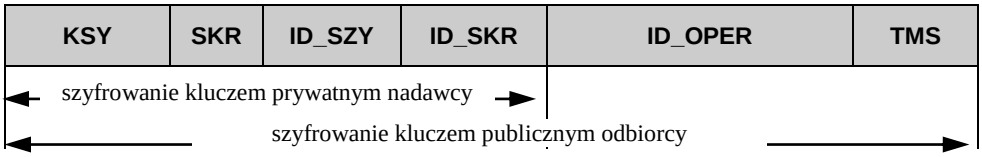
ID_SKR – identyfikator algorytmu użytego do wygenerowania skrótu;

ID_OPER – identyfikator zalogowanego w systemie użytkownika (nadawcy), który zainicjował operację zapisu danych - identyfikator ten jest niezbędny do określenia klucza publicznego nadawcy przy odczytywaniu pliku;

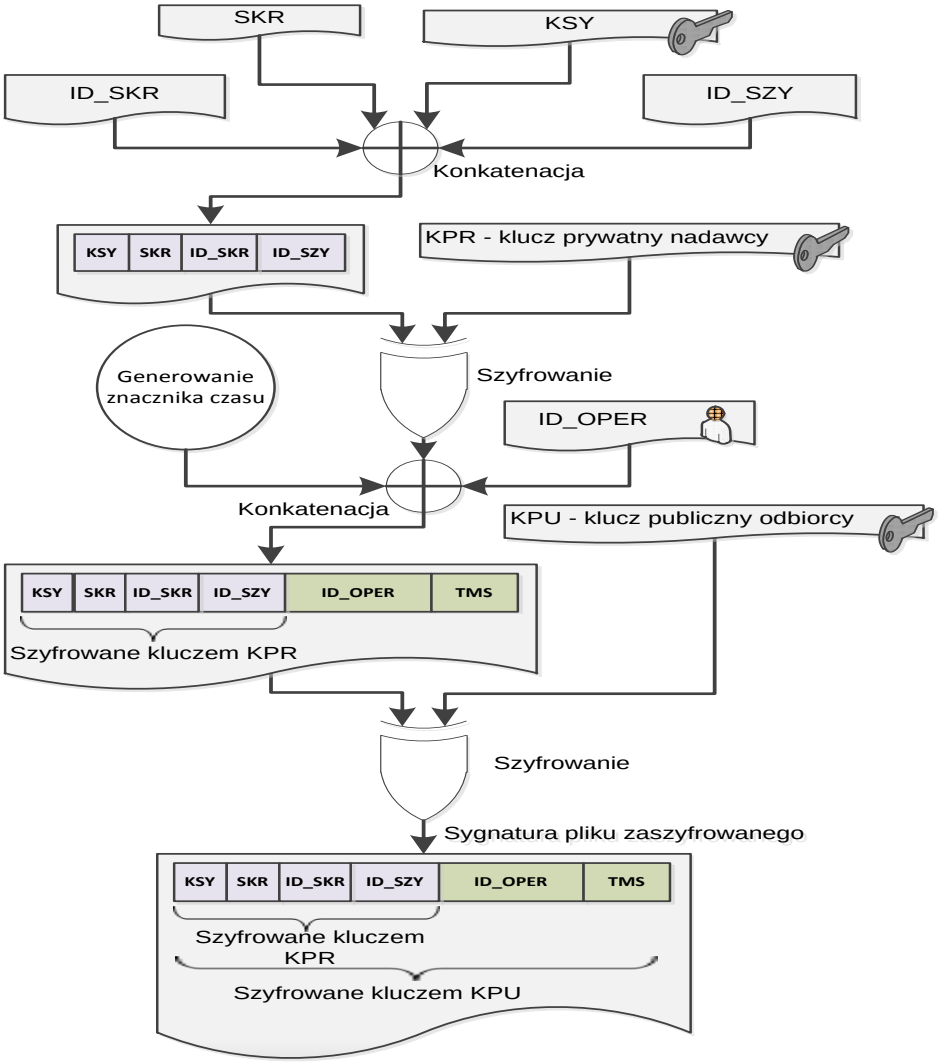
TMS – znacznik czasowy utworzenia pliku - wartość tego pola odpowiada systemowej dacie utworzenia pliku.

Postać sygnatury przedstawiono na rys. 3, natomiast proces tworzenia

sygnatury przebiega według schematu przedstawionego na rys. 4.



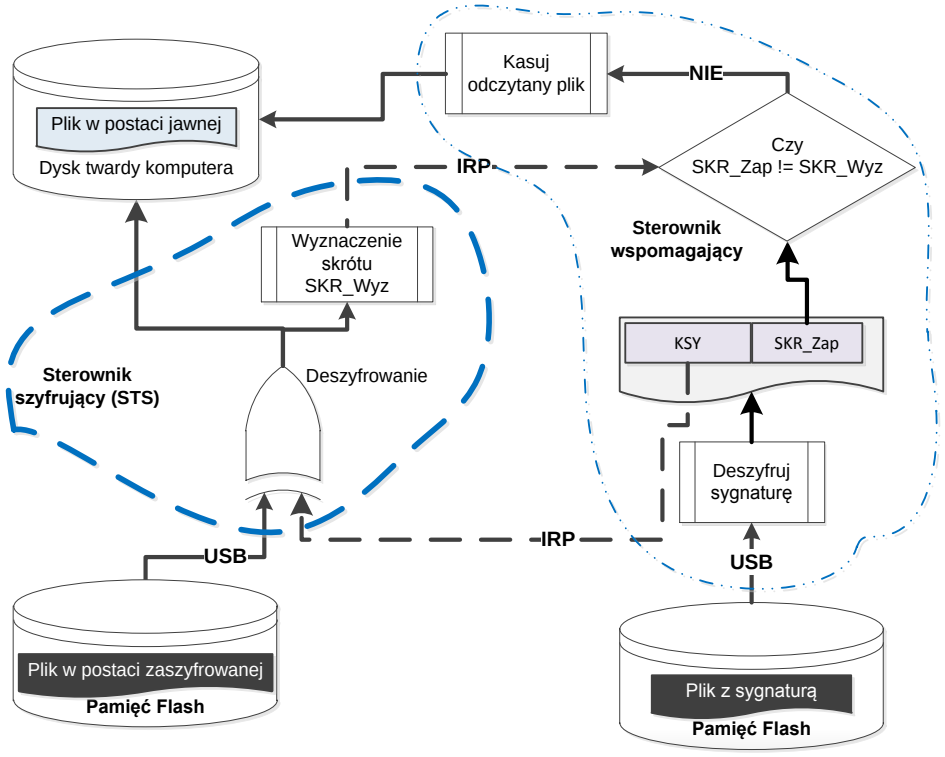
Rys. 3. Struktura sygnatury zabezpieczonego pliku



Rys. 4. Algorytm generowania sygnatury dla zabezpieczanego pliku

2.3. Odczyt pliku

Proces odczytywania pliku wymaga wcześniejszego odczytania oraz odszyfrowania sygnatury. Czynności te są wykonywane przez zalogowanego użytkownika (odbiorcę pliku) z wykorzystaniem ***AST***. Proces rozpoczyna się od odszyfrowania sygnatury przy użyciu klucza prywatnego zalogowanego użytkownika, odczytania znacznika czasu oraz identyfikatora użytkownika (**ID_OPER**), który pełnił rolę nadawcy przy tworzeniu zabezpieczonego pliku. Znacznik czasu zabezpiecza zaszyfrowany plik przed przenoszeniem go na nośnik inny niż ten, na który został pierwotnie zapisany. Niezgodność daty i czasu zapisanych w znaczniku czasu oraz daty i czasu utworzenia pliku powoduje wyświetlenie komunikatu informującego o wykrytej niezgodności i zakończenie procedury odczytu pliku. Przy zgodności wspomnianych wartości parametrów następuje deszyfrowanie kolejnego fragmentu sygnatury przy użyciu klucza publicznego użytkownika, którego identyfikator (**ID_OPER**) został wcześniej odczytany. Etapy deszyfracji pliku zostały schematycznie przedstawione na rys. 5.



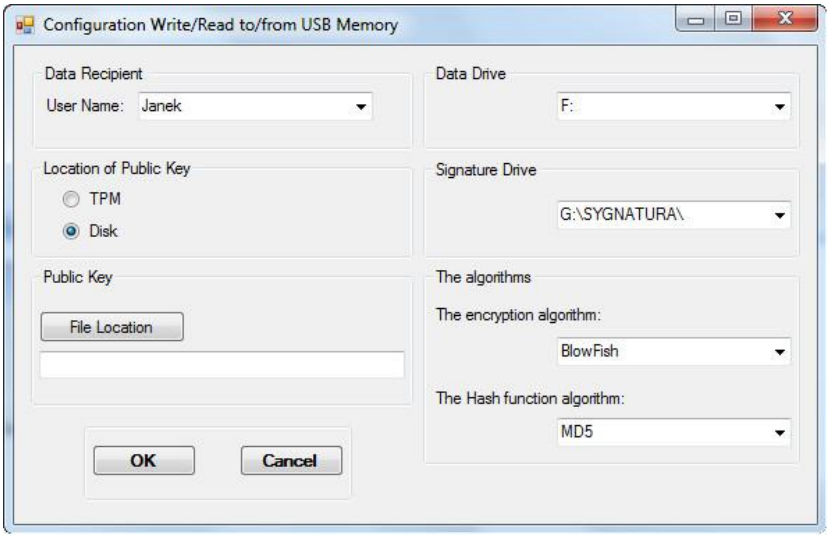
Rys. 5. Proces odczytu danych z pliku zewnętrznego

Na rysunku grubszą linią przerywaną zaznaczono operacje wykonywane przez sterownik szyfrujący, natomiast cieńszą linią (dwie kropki kreska) - operacje wykonywane przez sterownik wspomagający.

W trakcie odczytu danych wyznaczana jest wartość skrótu (SKR_Wyz). Jeżeli wartość SKR_Wyz jest różna od wartości odczytanej z sygnatury (SKR_Zap) wyświetlany jest odpowiedni komunikat, a zdeszyfrowany plik zapisany już na dysku komputera jest kasowany.

2.4. Obsługa tworzenia i odczytu zabezpieczonego pliku

Zalogowany użytkownik (nadawca) konfiguruje parametry procesu tworzenia i odczytu zabezpieczonego pliku za pomocą ***AST***, której okno przedstawiono na rys. 6.



Rys. 6. Okno aplikacji sterującej

Proces tworzenia zabezpieczonego pliku wymaga w pierwszej kolejności podłączenia do komputera, poprzez interfejs USB, jednej lub dwóch (zależnie od tego gdzie będzie przechowywany pliku z sygnaturą) pamięci Flash RAM. Urządzenia są automatycznie rozpoznawane przez ***STS***, który informacje o nich przekazuje za pośrednictwem ***STW*** do ***AST***. Następnie użytkownik zalogowany powinien określić parametry wymagane do zaszyfrowania pliku oraz wygenerowania sygnatury. Realizuje to wybierając:

- napęd, w którym będzie przechowywany plik zaszyfrowany (na rys. 6 pole „Data Drive”);

- napęd i ścieżkę dostępu do katalogu, w którym będzie przechowywany plik z sygnaturą (na rys. 6 pole „Signature Drive”);
- identyfikator algorytmu wykorzystywanego do szyfrowania (na rys. 6 pole „The encryption algorithm”);
- identyfikator algorytmu wykorzystywanego do generacji skrótu pliku zaszyfrowanego (na rys. 6 pole „The Hash function algorithm”);
- identyfikator użytkownika - odbiorcę danych zaszyfrowanych (na rys. 6 pole „Data Recipient”);
- lokalizację klucza publicznego odbiorcy danych (na rys. 6 pole „Location of Public Key”).

Identyfikator (**ID_OPER**) oraz klucz prywatny nadawcy (elementy wymagane do wygenerowania sygnatury) pobierane są automatycznie z systemu. Po określeniu danych konfiguracyjnych użytkownik zalogowany może rozpocząć proces kopiowania wykorzystując do tego, np. Explorator Windows. Nazwa pliku przechowującego sygnaturę będzie konkatenacją nazwy zaszyfrowanego pliku i ciągu „***.SIG***”. Proces utworzenia pliku z sygnaturą inicjowany jest po zakończeniu procesu szyfrowania i jest, podobnie jak sam proces szyfrowania, niewidoczny dla użytkownika. Przy szyfrowaniu kolejnego pliku dla tego samego odbiorcy nie ma potrzeby zmiany danych konfiguracyjnych, chyba że pozostałe parametry (identyfikator algorytmu szyfrującego lub identyfikator algorytmu generacji skrótu) mają być inne. Dla każdego kolejnego pliku zostanie automatycznie wygenerowany nowy klucz sesji.

Proces odczytu zabezpieczonego pliku wymaga w pierwszej kolejności podłączenia do komputera, poprzez interfejs USB, jednej lub dwóch (zależnie od tego gdzie przechowywany jest plik z sygnaturą) pamięci Flash RAM. Urządzenia są automatycznie rozpoznawana przez ***STS***, który informacje o nich przekazuje za pośrednictwem ***STW*** do ***AST***. Użytkownik zalogowany (odbiorca danych) za pomocą ***AST*** musi określić dysk, na którym przechowywany jest zaszyfrowany plik oraz wskazać plik z sygnaturą odpowiadającą zaszyfrowanemu plikowi. Realizuje to poprzez określenie:

- napędu, który przechowuje zaszyfrowany plik (na rys. 6 pole „Data Drive”);
- napędu i ścieżki dostępu do katalogu przechowującego plik z sygnaturą (na rys. 6 pole „Signature Drive”).

Po określeniu tych danych użytkownik może rozpocząć proces kopiowania pliku wykorzystując do tego np. Explorator Windows. Pozostałe parametry wymagane do deszyfracji pliku określane są na podstawie sygnatury. Po zainicjowaniu przez użytkownika procesu kopiowania pliku ***STS*** przesyła do ***STW*** nazwę kopiowanego pliku i wstrzymuje proces kopiowania do chwili

otrzymania danych wymaganych do deszyfracji pliku (identyfikatora algorytmu użytego do zaszyfrowania, klucza sesji oraz identyfikator algorytmu użytego do wygenerowania skrótu). Na podstawie przekazanej przez ***STS*** nazwy pliku zaszyfrowanego ***STW*** identyfikuje plik zawierający sygnaturę i przeprowadza proces jej deszyfrowania odczytując dane konfiguracyjne. Następnie realizuje proces weryfikacji odczytanego ***TMS*** z datą i czasem utworzenia pliku zaszyfrowanego. W przypadku niezgodności tych wartości wyświetlany jest komunikat i proces odczytu pliku zostaje przerwany. W przypadku zgodności ***TMS*** i daty oraz czasu utworzenia pliku odczytane z sygnatury pozostałe dane konfiguracyjne są przekazywane do ***STS***, który wznawia proces deszyfrowania. Jednocześnie z deszyfrowaniem pliku ***STS*** wyznaczana jest wartość skrótu. Po zakończeniu procesu kopiowania ***STS*** przekazuje do ***STW*** wyznaczoną wartość skrótu w celu jej weryfikacji. Jeżeli wyznaczona wartość skrótu nie będzie zgodna z odczytaną z sygnatury, to wyświetlany jest komunikat i ***STW*** usuwa odczytany plik. W przypadku zgodności obu wartości skrótów nie są już podejmowane żadne działania, a odszyfrowany plik staje się dostępny dla użytkownika.

3. Podsumowanie

Aktualnie szeroko dostępne zabezpieczania zawartości wymiennych nośników Flash RAM zwykle wykorzystują szyfrowanie symetryczne plików przy ich zapisywaniu. W tych rozwiązaniach zakłada się, że hasło niezbędne do szyfrowania/deszyfrowania jest wyznaczane przez użytkownika zapisującego zabezpieczony plik, a przy odczytywaniu pliku hasło dla użytkownika jest znane. W sytuacji gdy jeden użytkownik zapisuje zabezpieczony plik na nośniku wymiennym, a inny użytkownik z tego nośnika odczytuje ten plik, nie rozważa się problemów związanych z przekazywaniem klucza pomiędzy tymi użytkownikami, co jest istotnym niedostatkiem takich rozwiązań z punktu widzenia bezpiecznego przekazywania danych na nośniku Flash RAM.

Przedstawione, w artykule, rozwiązanie jest bardziej skomplikowane od powszechnie stosowanych i unikalne. Użytkownicy wykorzystujący je nie mają problemu z przekazywaniem klucza, bo korzystają z zalet szyfrowania asymetrycznego dającego rękojmię bezpiecznego przekazania klucza szyfrowania pliku stronom uczestniczącym w wymianie danych. Opracowany system wymaga, od użytkownika tworzącego zabezpieczony plik, określenia odbiorcy pliku i parametrów szyfrowania tego pliku. Proces zabezpieczania pliku jest ściśle związany z mechanizmami systemowymi obsługi nośników wymiennych Flash RAM i jest niezauważalny dla użytkownika. Przy odczytywaniu zabezpieczonego pliku użytkownik nie jest obarczony żadnymi dodatkowymi czynnościami. Ponadto zabezpieczenia są tak skonstruowane, że

odczyt pliku jest możliwy tylko z nośnika, na którym plik był pierwotnie zapisany. Próby kopiowania zabezpieczonego pliku na inny nośnik uniemożliwiają odczytanie takiego pliku. Działanie systemu zostało przetestowane w środowisku systemu Windows.

Opisany sposób zabezpieczenia danych na wymiennych nośnikach typu Flash RAM chroni dane przed niepowołanym dostępem na przykład w przypadku zagubienia lub kradzieży nośnika, ale również daje możliwość bezpiecznego przekazywania danych niezabezpieczonym kanałem na przykład z wykorzystaniem kuriera. Takie rozwiązanie jest niezbędne w systemach przetwarzających dane należące do różnych domen bezpieczeństwa (o różnych klauzulach tajności), w których przepływ danych musi być ściśle kontrolowany. Uzyskany poziom ochrony danych dla opisanego rozwiązania, poza zapewnieniem bezpiecznego sposobu przekazywania klucza, zależy od użytych algorytmów oraz, jak w większości przypadków [4], [5], mechanizmów zabezpieczeń systemu operacyjnego pod kontrolą którego pracuje.

Literatura

[1] CHUDZIKIEWICZ J., *Zabezpieczenie danych przechowywanych na dyskach zewnętrznych*, Metody wytwarzania i zastosowania systemów czasu rzeczywistego, Wydawnictwo Komunikacji i Łączności, Rozdział XVIII, Warszawa, 2010, str. 211-221.

[2] CHUDZIKIEWICZ J., *Programowe zabezpieczenie plików przechowywanych na dyskach zewnętrznych*, Wojskowa Akademia Techniczna, Biuletyn Instytutu Automatyki i Robotyki, nr 28, Warszawa, 2010, str. 61-72.

[3] *Microsoft Windows Driver Kit (WDK)* (dokumentacja techniczna), Microsoft Corporation, Redmond, 2009.

[4] *Rohos Mini Drive* (dokumentacja techniczna) http://www.rohos.com/products/rohos-mini-drive/.

[5] *USB Flash security* (dokumentacja techniczna) http://kashu-sd.co.jp/en/outline.html.

[6] http://www.microsoft.com/poland/technet/bazawiedzy/centrumrozwiazan/cr028_01.mspx.

[7] *MSDN Library, Glossary: Windows DDK* (October 2003 Release), Microsoft Corporation, Redmond, 2003.

[8] NAGAR R., *Filter Manager*, Microsoft, Redmond 2003 http://download.microsoft.com/download/f/0/5/f05a42ce-575b-4c60-82d6-208d3754b2d6/Filter_Manager.ppt.

[9] NAGAR R., *OSR's Classic Reprints: Windows NT File System Internals*, OSR Press, 2006.

[10] ONEY W., *Programming the Microsoft® Windows® Driver Model*, Microsoft Press, Redmond, 2003.

[11] RUSSINOVICH M. E., SOLOMON D. A., *Microsoft® Windows® Internals, Fourth Edition: Microsoft Windows Server™ 2003, Windows XP, and Windows 2000*, Microsoft Press, Redmond, 2005.

The method of secure data exchange using Flash RAM media

ABSTRACT: In this paper a method for secure transfer of files stored in a Flash RAM through unsecured transport channel (e.g.: courier, traditional postal system) between users is described. The presented method is based on a Microsoft Windows driver called "filter driver" and uses symmetric as well as asymmetric encryption. The solution allows a sender to determine a file recipient and the recipient to unambiguously identify the sender of the file.

KEYWORDS: device drivers, encryption algorithms, file system, filter driver.

Praca wpłynęła do redakcji:30.10.2012

Praca realizowana w ramach projektu NCBiR nr OR00014011.