ISBN 978-83-60810-51-4

The method of secure data exchange using Flash RAM media

Jan Chudzikiewicz
Military University of Technology
ul. Kaliskiego 2,
00-908 Warszawa, Poland
Email: jchudzikiewicz@wat.edu.pl

Janusz Furtak
Military University of Technology
ul. Kaliskiego 2,
00-908 Warszawa, Poland
Email: jfurtak@wat.edu.pl

*Abstract—***This document describes the method for secure transfer of files stored in Flash RAM through unsecured transport channel (e.g.: courier) between users. In this method the sender of the file specifies the recipient and the recipient knows who is the sender of the file. The idea of a solution that uses symmetric and asymmetric encryption is described. The following procedures are presented: creating protected file (encryption), generating signatures for that file and reading (decryption) the file.**

I. Introduction

THE increasing popularity of Flash RAM used as storage media forces the need for mechanisms to ensure an adequate level of protection of data stored on them. This is particularly important in the case of sensitive data which has significant impact on the safety of the institution.

For this purpose, the most commonly used is the software (e.g. USB Flash Security, Secure Traveler, Rohos Mini Drive, etc.) that must be installed on the media prior to its use. During the installation of such software in Flash RAM it is created an encrypted volume that is accessed using a password defined. The power of safeguard of the medium using this type of software depends on the used symmetric encryption algorithm and key length. This type of security is sufficient in the event of such loss or theft of the media. Use this solution to transfer data between the various entities that use such media poses problems arising mainly from the need to provide the transfer the medium and an encryption key with help which the media has been encrypted. In addition, the data sender is not certain that the data will be available only for the recipient, and the recipient is not certain that it has received data from the expected sender.

The article presents a solution enabling to such preparation of data stored in Flash RAM, so that the recording medium can be used for secure transfer of data files, during which the sender of data (i.e., the creator of the protected media content) is assured that data will be available only for designated recipient and the recipient is assured that the data received come from the expected sender. The described mechanism uses both symmetric encryption algorithms and asymmetric. The presented solution uses a filter driver [1][3][4]. In this solution, it is assumed that in terms of operating system data can be processed in two directions: from plain text form stored on your hard disk to secure form on removable media (e.g. Flash RAM, hard drive) connected to the system via the USB bus and, conversely, from secure form on removable media to plain text form on the hard drive. Do not allow the possibility of using the software for the direct exchange of data files between removable media (e.g. flash memory) connected to the system via the USB bus.

The process of securing the exchange of data should satisfy the following functional requirements:

– should be implemented in a manner transparent to the user;
– should not cause any noticeable to the user loads of the operating system;
– should not have a significant impact on the speed of read and write data onto data media;
– should allow the use of various encryption algorithms to ensure the required level of confidentiality;
– should build on removable media, protected file and the signature for this file;
– should allow to perform any operation allowed for data storage media, such as volume, surface checking for errors, and defragment the disk-based data.

These requirements force the use of the process of securing the data separate modules (drivers) operating at the kernel-level of operating system [2][3][6]. Schematic representation of a solution being developed is shown in Figure 1.

Described solution is available for a user through the control application (AST). The main elements of the built system are interacting drivers: encryption driver [1] and driver supporting, which are compatible to the Windows Driver Model [5]. Both elements work in kernel mode operating system and communicate with each other using the internal mechanisms of the operating system (in the figure these mechanisms are labeled as IRP - Input-Output Request Packet) [2][5].

This work was supported by The National Center for Research and Development, Project OR00014011

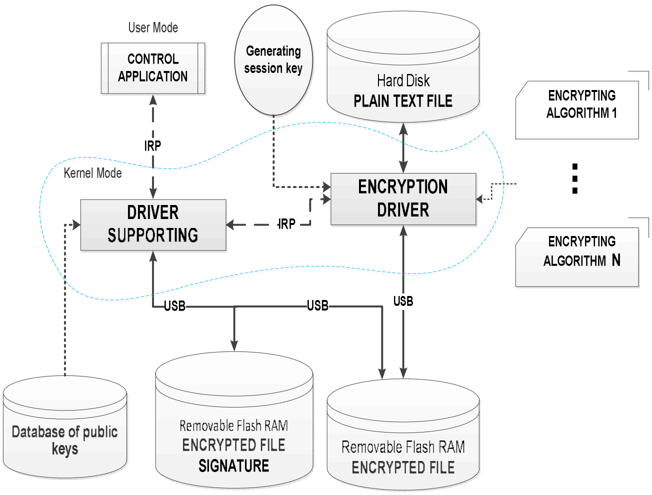


Fig. 1 Schematic diagram of securing data stored on removable media

The purpose of the encryption driver (STS) is the realization of the process of encryption / decryption of data and determination of its hash value for these data. The driver supporting (STW) sets the signature for protected data (according to the algorithm presented in the next section) and mediates the transfer of messages / commands between STS and AST. The other components of the system are: the .DLL library that provides the functionality of the implemented encryption algorithms, module of generation of session key, and the database of public keys of users.

The product of the process to secure the original file are two files: a file with encrypted data and the file containing the signature for the encrypted file. Both files can be stored on one medium or each file on a different medium. Choosing a storage location of the signature file is defined by the user through AST. It should be noted that saving the encrypted file and the signature on separate media increases the security of stored data, but it is cumbersome to use.

II. The Process of Creating and Reading a Protected File

The process of creating a protected file on removable flash memory (that is, to create an encrypted file and the signature of this file) and reading (decryption) file from the removable flash memory needs attributes of a user which creates a protected file (file sender) and a user for which a secured file will be created (file recipient). When a protected file is created the user logged into the system is the sender, and he specifies the file recipient using the AST. When reading a protected file with using the AST logged user is the file recipient, a sender's attributes are read after successful decryption of signature that file with using a private key of the logged user. The situation in which at the same time the logged user is the sender and the recipient of the data is acceptable.

The process of creating a protected file includes the step of encryption, and then creating a signature for that file. However the process of reading a protected file in a first step obtains from file signature the attributes needed for decrypting that file and in the second step the protected file is decrypted.

A. *Creating a protected file*

The process of writing the file, including file encryption and hash generation, is performed by the STS. Operation of STS has been presented in [1]. The diagram describing the process of writing the file is shown in Figure 2. Dashed line in Figure 2 indicates operations implemented by the STS. During the process of file encryption is determined the value of the hash function to ensure the integrity of the file.

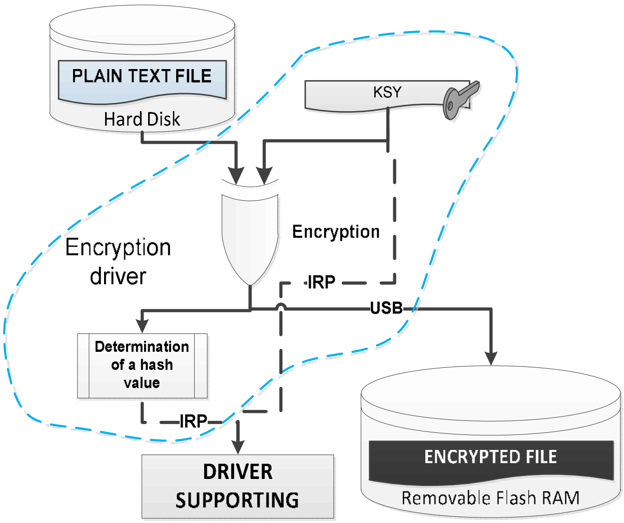


Fig. 2 The process of writing data to removable flash memory

Determined value of the hash function, and the generated session key after completion of record are transferred to the STW in order to generate a signature for the stored data. The process transferring the hash function value and the session key transferring is implemented using the system mechanisms marked in Figure 2, as the IRP.

B. *Determining the signature*

For each of the protected file the signature is generated which containing the information needed to read of this file. Signature of the file contains the following fields:

KSY - random key to encrypt / decrypt the secure file;
SK - value of hash function which is determined based on the content of protected file after encrypting this file;
ID_SZY - identifier of the algorithm used to encrypting;
ID_SKR - identifier of the algorithm used to generate the hash;
ID_OPER - identifier of the logged user (the sender) who initiated the operation of data write - this identifier is required to determine the public key of sender when the file is read;
TMS - time stamp of file creation - this value corresponds to the date of file creation.

The structure of signature is shown in Figure 3, and process of signature creation proceeds according to the diagram is shown in Figure 4.

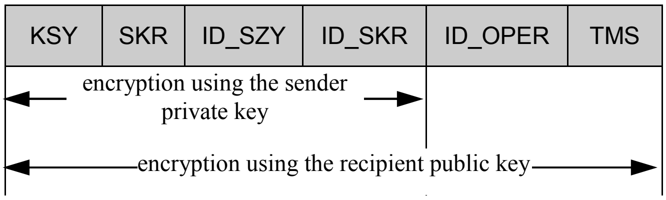

KSY
SKR
ID_SZY
ID_SKR
ID_OPER
TMS
encryption using the sender private key
encryption using the recipient public key

Fig. 3 The structure of signature secure file

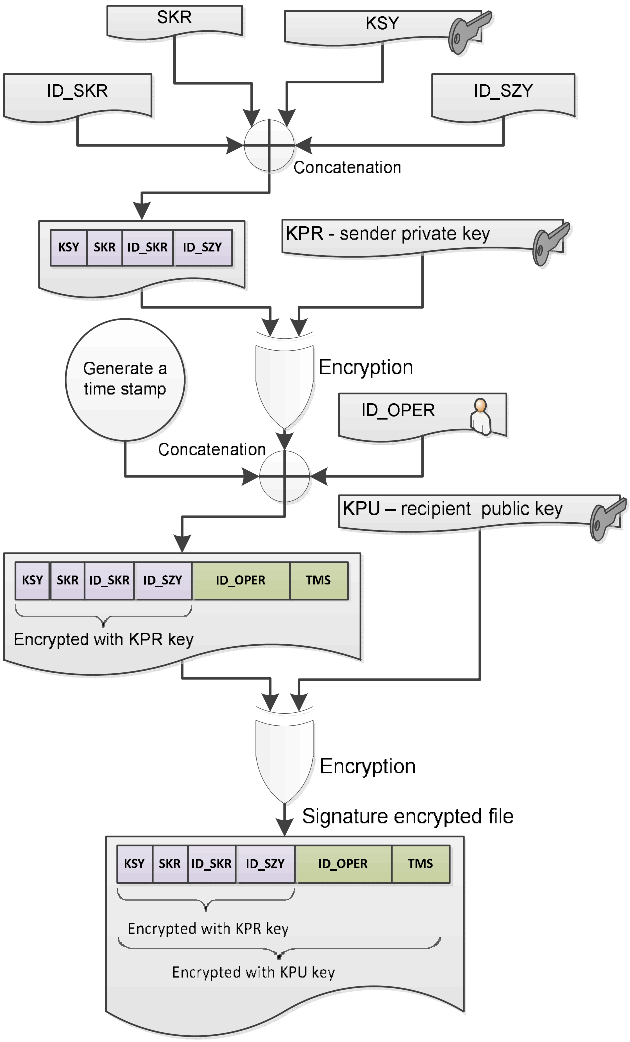


Fig. 4 The algorithm of signature generation for protected file

C. *Reading a protected file*

The process of reading the file requires that the signature was read before and then decrypted. These activities are performed by the logged user (recipient of file) using AST. The process starts with decrypting the signature file using the private key of the logged user, then reading time stamp and user identifier (ID_OPER) which assumed the role the sender creating a protected file. The time stamp protects the an encrypted file before moving it to another medium than that on which was originally written. Incompatibility of date and time stored in the time stamp and date and time the file was created displays the message and terminates the procedure of file reading. With compatibility of the parameters the next part of the signature is decrypted using the user public key which identifier (ID_OPER) has been read. The next steps of file decoding are schematically shown in Figure 5.

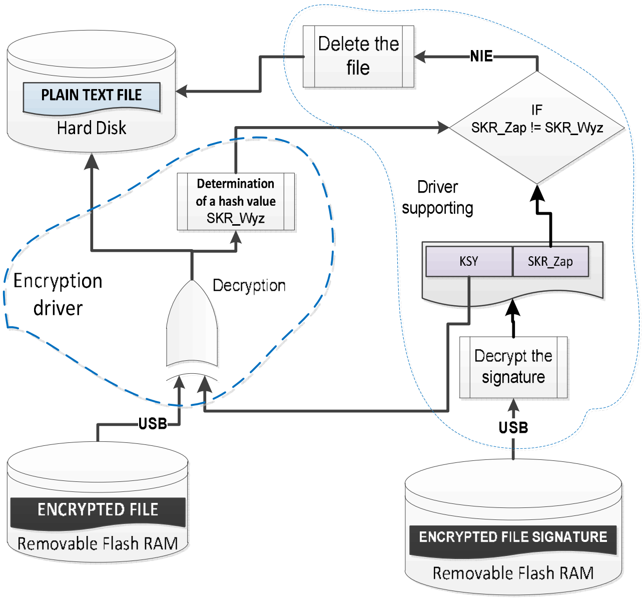


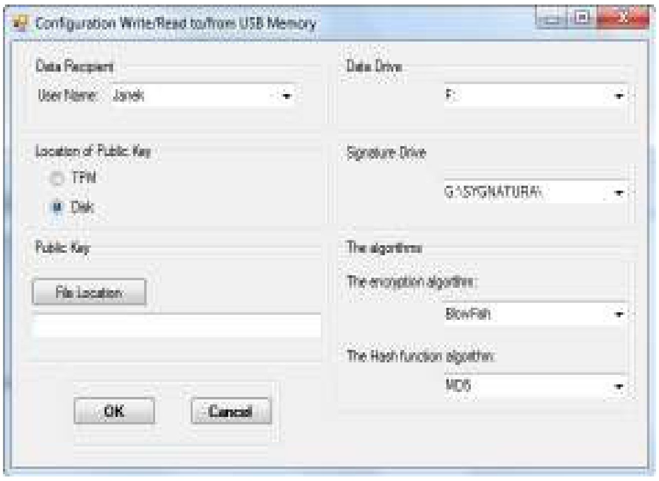
Fig. 5 The process of data reading from an external file

On the Figure 5 operations performed by the STS are marked using thick dashed line, and the operations performed by the STW are marked using the thinner line (two dots dash).

During the read data the value of hash function (SKR_Wyz) is determined. If the value SKR_Wyz is different from the values obtained from the signature (SKR_Zap) a message is displayed and the decrypted file, which was saved on hard disk, is being deleted.

III. Handling for Creating and Reading a Secure File

Logged user (file sender) configures the parameters of the process of creating and reading a protected file using AST whose window is shown in Figure 6.


Configuration Write/Read to/from USB Memory

Data Recipient
User Name: Janek

Data Drive
F:

Location of Public Key
○ TPM
● Disk

Signature Drive
G:\SYGNATURA\

Public Key

The algorithms
The encryption algorithm: BlowFish
The Hash function algorithm: MD5

Fig. 6 The window of control application AST

The process of creating protected file requires first of all connection one or two (depending on where the file with the signature will be stored) removable Flash RAM memories to a computer through USB interface. The devices are automatically detected by STS, which transmits information about them via the STW to AST. The logged user should determine the parameters required for encrypting the file and generating a signature. He does this by selecting the:

- drive in which will be stored the protected file (field "Data Drive" in Figure 6);
- drive and path to the directory in which will be stored the file with the signature (field "Signature Drive" in Figure 6);
- identifier for the algorithm used to encrypt (field "The encryption algorithm" in Figure 6);
- identifier for the algorithm used to generate Hash value for the protected file (field "The Hash function algorithm" in Figure 6);
- identifier for user (recipient) encrypted data (field "Data Recipient" in Figure 6);
- location of public key data recipient file (field "Location of Public Key" in Figure 6).

Identifier (ID_OPER) and the private key of the sender (the elements required to generate the signature) are automatically retrieved from the system. After determining the data configuration logged user can begin the process of copy the file using , e.g. Windows Explorer. The name of file which stores the signature will be concatenation of the name of protected file and string "SIG". The process of creating a file with the signature is started after the encryption process is finished and is, just as the encryption process, invisible to the user. When next file for the same recipient is being encrypted it does not need to change the configuration data unless the other parameters (that is the identifier of encryption algorithm or identifier of algorithm generating of hash value) will be changed. Always for the next file a new session key will be automatically generated.

The process of reading protected file requires connection to a computer through USB interface one or two (depending on where is stored the file with the signature) removable Flash RAM memories. The devices are automatically detected by STS, which transmit information about them via the STW to AST. The logged user (recipient of the data) using AST has to specify the drive on which is stored encrypted file and indicate the file with the signature corresponding to the encrypted file. He does this by selecting the:

- drive on which will be stored the protected file (field "Data Drive" in Figure 6);
- drive and path to the directory on which will be stored the file with the signature (field "Signature Drive" in Figure 6).

Other parameters required to decrypt the file are determined based on the signature. After initializing by the logged user the process of copying a file STS sends to the STW the name of the copied file and pauses the copy process to the moment when receives data required to decrypt the file (that is the identifier of encryption algorithm, session key and identifier of algorithm generating of hash value). Based on submitted by the STS the name of encrypted file, STW identifies a file containing the signature and performs the process of signature decryption and reading the configuration data. Then performs the verification process read out TMS with the date and time of the creation of an encrypted file. In the case of inequality of these values message is displayed and the file reading process is interrupted. In the case of equality of those values other configuration data read from the signature are passed to STS, which resumes the process of decryption. During the proces of decrypting the file the STS determines the value of a hash function for that file. After completion the process of copy STS transmit to STW determined value of the hash function for verification. If the designated hash value is not equal to the value read from the signature a message is displayed and the STW deletes the file.

IV. Conclusion

Currently widely available tools to secure the contents of removable Flash RAM uses symmetric encryption when writing files. In these solutions, it is assumed that the key needed for encryption / decryption is determined by the user who creates protected file, and during reading the protected file this key is known for the user. When a user saves a protected file on removable medium, and another user of this medium reads this file, the problems associated with the transmission of the key between the users are not taken into consideration, which is a significant lack of such solutions in terms securing of transfer of saved data on media Flash RAM.

The solution presented in this paper is unique and more complicated than the commonly used. Users using this solution do not see the problem with the transmission of the key, because they are using the advantages of asymmetric encryption which gives assurance secured transfer of encryption key between parties involved in the exchange of data. The developed system requires the user who is creating a protected file, to determine file recipient and the parameters to encrypt the file. The process of protecting file is closely linked with the mechanisms of systemic support for removable media Flash RAM, and is transparent to the user. When the protected file is read, user is not burdened with any additional activities. In addition, security is so constructed that the reading of the file is only possible with the medium on which the file was originally saved. Attempting to copy a protected file to a different medium locks the ability to read the file. This scheme has been tested in a Windows environment.

The described method for protecting data on removable Flash RAM protects data from unauthorized access, for example in case of loss or theft of the medium, but also makes it possible to secure transfer of that data through an unsecured transmission channel, for example using a courier. This approach is necessary in systems that process data belonging to different security domains (with different classification levels) in which the flow channel of data must be strictly controlled

REFERENCES

[1] J. Chudzikiewicz, "Zabezpieczenie danych przechowywanych nadyskach zewnętrznych" in *Metody wytwarzania i zastosowania systemów czasu rzeczywistego,* 2nd ed. vol. 3, J. Peters, Ed. Warszawa: Wydawnictwo Komunikacji i Łączności, 2010, pp. 211–221.

[2] *Microsoft Windows Driver Kit (WDK),* Technical Documentation, Redmond, Microsoft Corporation, 2009.

[3] R. Nagar, *Filter Manager*. Redmond, Microsoft Corporation, 2003.

[4] R. Nagar, *OSR's Classic Reprints: Windows NT File System Internals*. Redmond, OSR Press, 2006.

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

[6] M. E. Russinovich, D. A. Solomon, *Microsoft® Windows® Internals,* Fourth Edition: Microsoft Windows ServerTM 2003, Windows XP, andWindows 2000, Redmond, Microsoft Press, 2005.