Proceedings of the 2013 Federated Conference on
Computer Science and Information Systems pp. 811–818

How to use the TPM in the method of secure data exchange using Flash RAM media

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

Tomasz Pałys
Military University of Technology
ul. Kaliskiego 2,
00-908 Warszawa, Poland
Email: tpalys@wat.edu.pl

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

***Abstract*—This document describes how to use the Trusted Platform Module (TPM) in the method of secure transmission of data stored on the Flash RAM through insecure transport channel. 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 TPM is used to safely generate symmetric and asymmetric keys, and theirs the safe collection, storage and management in order to protect files during transfer. The way of organizing data in a cryptographic keystore for users authorized to use the system for the secure transmission of files stored in Flash RAM is described.**

I. Introduction

IN THE era of widespread use of the Internet moving data from one location to another is a natural phenomenon. Data may be transferred with the use, for example, of HTTP protocol (files describing the presentation of Websites in HTML) FTP protocol or others. Transferring a non-confidential data over the Internet is not born of trouble. The problem occurs when the transferred data are sensitive and can be made available only for selected recipients. Then securing the data transferred on the route between the sender and the recipient is necessary. For this purpose, in Internet TLS protocol (Transport Layer Security) at the application layer or IPSec protocol (Internet Protocol Security) at the network layer are commonly used. The applied solutions in this area use symmetric and asymmetric cryptography. In both approaches from the point of view of security, the most important task is authentication both sides communicating against each other. This can be achieved by using one of the following methods:

- password known for both sides (shared secret);
- manually exchange of public keys of both sides;
- using public key certificate (X.509).

Normally is used the last solution, which is most comfortable, but requires the access to the Certification Authority.

Transfer data in the data processing systems of different classification levels is much more complicated problem [4]. In such systems, it is important not just who is the data recipient, but also whether the data recipient has adequate security clearance certification. Due to the legislation regarding of classified data processing very often it is not possible using a computer network to data transfer and building a mechanism for user authentication [12]. Then the transfer of sensitive data over insecure transport channels becomes a necessity. In such systems, the only way to safe data transfers is secured a removable media (i.e. Flash RAM or removable disk) connected to the system by USB interface [3].

This paper presents a solution to such preparation the data stored in Flash RAM so that storage medium can be used for secure transmission of the data [3]. In this solution a sender of the data (ie, the creator of the protected media content) can be sure that the data will only be available for a designated recipient, and the recipient can be sure that the received data originates from the expected sender. The described mechanism uses both symmetric encryption algorithms and asymmetric. The presented solution uses a *filter driver* [1][5][6][9]. In this solution, it is assumed that in terms of operating system the data can be processed in two directions: from plain text form stored on your hard disk, to protected form on removable media (e.g. Flash RAM, hard drive) connected to the system via the USB bus and, conversely, from protected form on removable media to plain text form on the hard drive.

The process of securing exchange of data is transparent for the user except for moment at which the system will recognize new storage media Flash RAM connected to the station. Then, the user is forced to specify his preference of data encryption algorithms, origin of cryptographic keys and location of protected files. For this reason to implement the process securing data it is necessary using separate operating system modules (drivers) running on the system kernel mode [5][9][10]. Schematic diagram of developed solution is shown in Fig 1.

User have access to the described solution through the Control Applications (***CApp***). The basic elements of constructed system are cooperating with each other drivers: encryption driver [1][2][3] and driver supporting, which are compatible to the Windows Driver Model [5][9]. Both

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

978-1-4673-4471-5/$25.00

elements work in operating system kernel mode 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) [5][7][9][10]. The purpose of the encryption driver **(EnD)** is realization of the process of encryption / decryption data and determination a value of hash function for these data. The tasks of the Driver Supporting **(DSu)** are among other things, creation of signatures for protected data and intermediation in transmitting messages / commands between **DSu** and **CApp**. Both drivers cooperate with the Trusted Platform Module (TPM). The purpose of the TPM are a secure cryptographic key generation, storing them and sharing, and supporting cryptographic procedures. Management Console for Cryptographic Module (MCCM) is used to management of TPM and cryptographic keystore management. An important element of the system is a DLL library that provides the encryption functions.

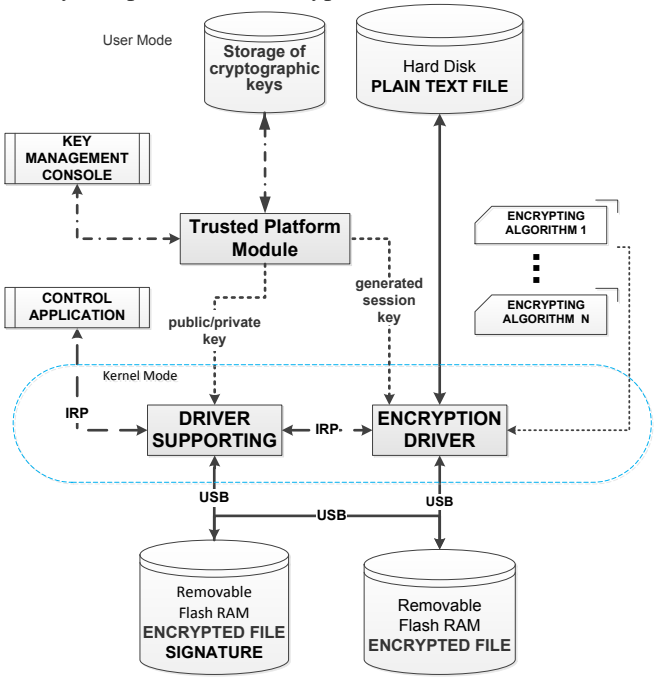

Fig. 1 Schematic diagram of securing data stored on removable media

The products of the process of securing data are two files: a file with encrypted data and signature of that file describing the file containing the encrypted data. The both files can be saved on one medium, or each file on different medium. Choosing a storage location of the signature file is defined by the user through **CApp**. It should be noted that saving the encrypted file and the signature file on separate media increases the security of stored data, but it is cumbersome to use. In the next sections of the paper is presented the process of creating and reading a protected file and the concept of using the capabilities of TPM to management of the cryptographic keys for procedures of protecting the file on a removable Flash RAM.

II. THE PROCESS OF CREATING AND READING A PROTECTED FILE

In the process of creating a protected file on removable flash memory (that is a creating an encrypted file and the signature of this file) and reading (decrypting) the file from the removable flash memory are necessary attributes of the user who created the protected file (this user will be called the sender) and user for whom the protected file was created (this user will be called the recipient). When a protected file is created the user logged into the system is the sender, and he specifies the file recipient using the **CApp**. When reading a protected file with using the **CApp** logged user plays the recipient role, a sender's attributes are read after successful decryption of signature of this file, using a private key of the logged on user. Permissible is a situation in which the logged user is simultaneously sender and the recipient of data.

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

A. Creating a protected file

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

Determined value of the hash function, and the generated session key after completion of record are transferred to the **DSu** 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 Fig. 2, as the IRP.

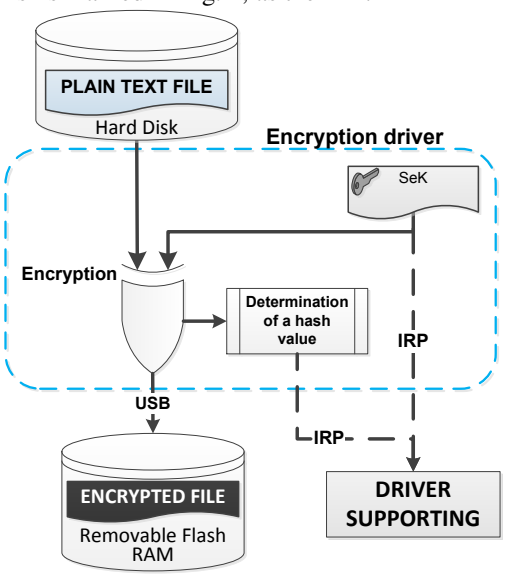

Fig. 2 The process of writing data to removable flash memory

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:

SeK - random key to encrypt / decrypt the secure file;
HASH - value of hash function which is determined based on the content of protected file after encrypting this file;
H_ID - identifier of the algorithm used to generate the hash;
En_ID - identifier of the algorithm used to encrypting;
O_ID - 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 Fig. 4, and process of signature creation proceeds according to the diagram is shown in Fig. 3.

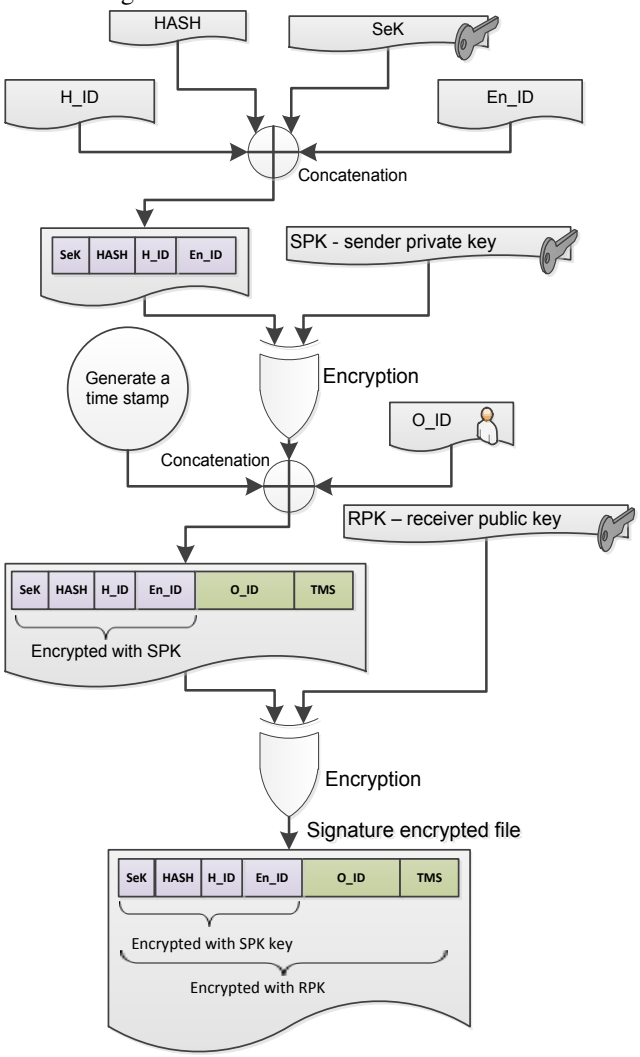

Fig. 3 The algorithm of signature generation for protected file

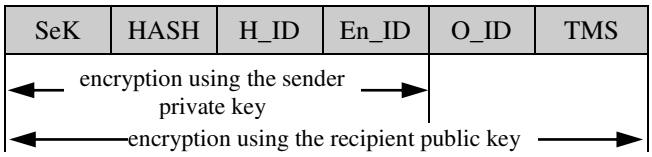

Fig. 4 The structure of signature secure file

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 **CApp**. The process starts with decrypting the signature file using the private key of the logged user, then reading time stamp and user identifier (O_ID) 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, when the file was created, causes displaying the message and terminating 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 (O_ID) has been read. The next steps of file decoding are schematically shown in Fig. 5.

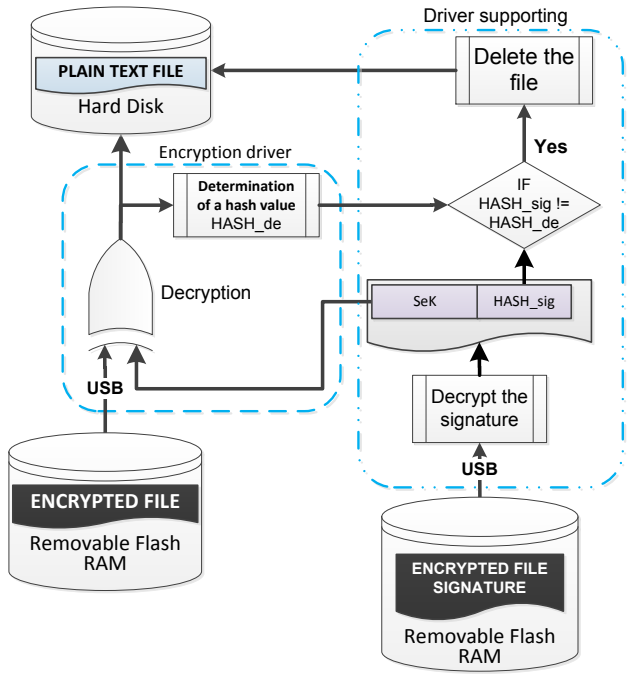

Fig. 5 The process of data reading from an external file

On the Fig. 5 operations performed by the **EnD** are marked using thick dashed line, and the operations performed by the **DSu** are marked using the thinner line (two dots dash).

During the read data the value of hash function (HASH_de) is determined. If the value HASH_de is different from the values obtained from the signature (HASH_sig) a message is displayed and the decrypted file, which was saved on hard disk, is being deleted [7].

III. THE CONCEPT OF USING TPM

Trusted Platform Module (TPM) is an implementation of a standard developed by the Trusted Computing Group [11]. This module is designed to support the cryptographic procedures and protocols that can be used for securing data [12]. Trusted Paltform Module, software TrouSerSwin consistent with TSS[1] and OpenSSL[2] library is a full-featured encryption system that provides the following functions:

- generating an asymmetric key pair;
- secure storage of keys;
- generating an electronic signatures;
- encryption and decryption;
- implementation of an operation defined by the standard PKCS # 11 cryptographic.

The TPM uses the mechanism of Root of Trust (RT) of secure data transmission in a computer system and is the basis to perform trusted cryptographic operations. This mechanism consists of the following components [8]:

- Root of Trust for Measurement (RTM) – uses Platform Configuration Registers (PCR) to record the state of a system;
- Root of Trust for Reporting (RTR) – uses PCR and RSA signatures to report the platform state to external parties in an unforgettable way;
- Root of Trust for Storage (RTS) – Uses PCR and RSA encryption to protect data and ensure that data can only be accessed if platform is in a known state.

The following algorithms are typically implemented in TPM: RSA, SHA-1, HMAC and AES[3]. In addition, each TPM chip stores a unique serial number and your RSA private key that is never available to read. TPM components are shown in Fig. 6.

The most important element of TPM is the cryptographic coprocessor. It performs the following actions:

- generating keys for asymmetric and symmetric cryptography using a Random Numbers Generator;
- encryption/decryption of data using the RSA algorithm;
- supporting of the integrity protection (SHA-1 Engine) and authentication (HMAC Engine).

The non-volatile memory of the module stores the asymmetric keys[4]: Endorsement Key (EK), Storage Root Key (SRK) and symmetric key NV_KEY. The EK key is usually generated during the production of the TPM and is used to decrypt the certificates for the other keys generated by the TPM. The SRK key is generated during the initialization of the TPM and is used as a master key to secure store of users' keys. In the key store are stored a keys of users who can create protected files (file sender) and users who will be recipients of protected files. The RK key is used to import the public key of recipients protected files. The NV_KEY key is designed to encrypt a data resource of users.

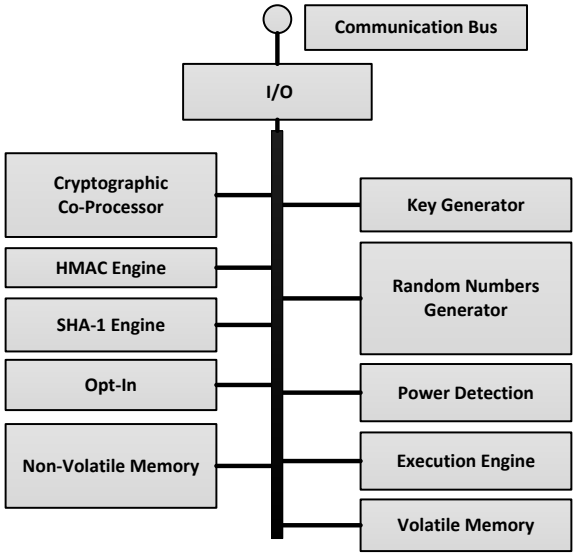

Fig. 6 TPM Component Architecture (based on [11])

For secure storage of user data involved in the exchange of protected files, especially cryptographic keys for that users, in the system created two protected resources: cryptographic keystore and TPM users' table.

A. Cryptographic keystore

The cryptographic keystore is designed to securely store, manage and share keys of all users involved in the exchange of protected files. These users include:

- local users who are on the station will be acted as the sender of uploaded files or recipient of received files;
- external users who are adequately recipient uploaded files and sender received files.

Keys for local users is generated by the TPM. The private key and public key of local users are stored in the cryptographic keystore. For each external user is kept only the public part of the key. The keys of external users are imported from other stations. Each key in the cryptographic keystore is identified by Universally Unique Identifier (UUID)[5].

Cryptographic keystore has a hierarchical structure (Fig. 7). At the top of the hierarchy is placed Storage Root Key (SRK). That key is generated by the TPM during the initialization of the module and is always stored in the non-volatile memory of TPM. Public part of the SRK key is used to encrypt the keys present in the hierarchy below the SRK key. The SRK key is the predecessor of the following keys:

[1] TSS – Trusted Computing Group Software Stack – specification of the software that allows the implementation of applications based on the TPM. For Windows OS, it is a project "TrouSerSwin", and in the case of Linux systems, it is possible to use the project "TrouSerS".

[2] OpenSSL – OpenSource cross-platform library that contains the implementation of protocols and general-purpose cryptographic algorithms.

[3] TPM uses a symmetric algorithm AES to protect the confidentiality of the session in which it participates following the recommendations of the TCG. However, symmetric encryption functions are not normally accessible outside the TPM.

[4] The Endorsement Key (EK) and the Storage Root Key (SRK) are mandatory for TPM while the Roaming Key (RK) and NV_KEY are necessary for the implementation of security procedures for files on Flash RAM.

[5] In the present solution as the UUID is used Globally Unique Identifier (GUID) of objects in Windows which is assigned to individual users.

- Roaming Key (RK) - used in the procedure to import the public keys of external users;
- Storage Key (SK) of local or external user - is used to encrypt the encryption keys used by individual users.

The safest place for storing cryptographic keys is non-volatile memory of the TPM, but due to the small size of the memory only the most important keys are stored there. Other keys are placed on your hard disk. The arrangement of keys is show in Fig. 7 to cryptographic keystore management is used the TrouSerSwin library with service TCSD[6] which works according to recommendations of the Trusted Computing Group.

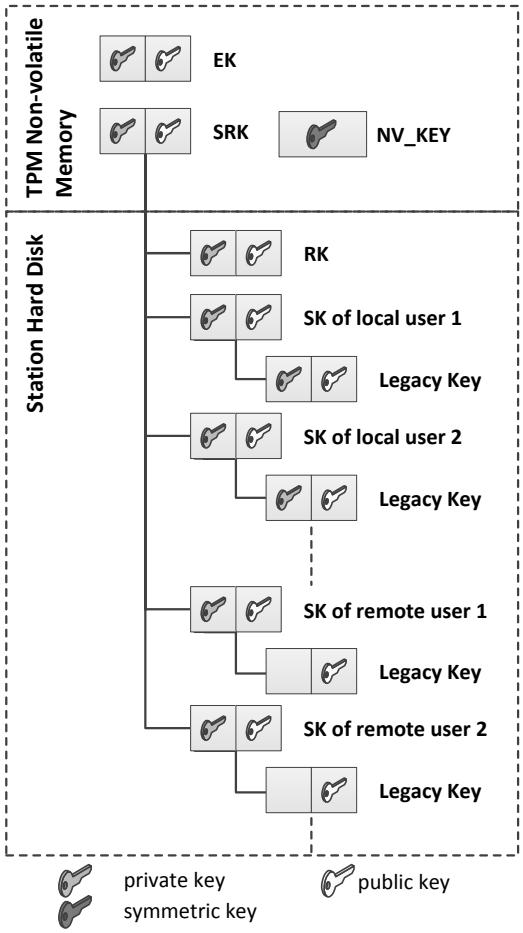

Fig. 7 Hierarchy of cryptographic keys supported by TPM

B. TPM users' table

The TPM users' table stores users data involved in the exchange of protected files. These refers to local and external users. A user with an access account to resources of the station may be considered as the local user. Otherwise an external user is the user with an access account to another station and whose data were imported from that station. The following record is combined with each TPM user:

- name of access account;
- display name of user;
- user Security Identifier (SID) on the local station - identifier used in Windows systems to determine access rights - USER_SID;
- Globally Unique Identifier of User - is used to identify the SK key of user – USER_GUID;
- identifier of user Legacy Key - this key is used to encrypt data intended for that user – USER_LEG_GUID.

The data of TPM users are encrypted with a symmetric key NV_KEY stored in non-volatile memory of TPM.

C. Cryptographic module

The cryptographic module is used to manage TPM users. This module consists of the following components:

- TPM;
- cryptographic keystore and a table of TPM users;
- TrouSerSwin library with service TCSD and Open SSL library;
- Management Console for Cryptographic Module (MCCM).

Architecture of cryptographic module is shown in Fig. 8.

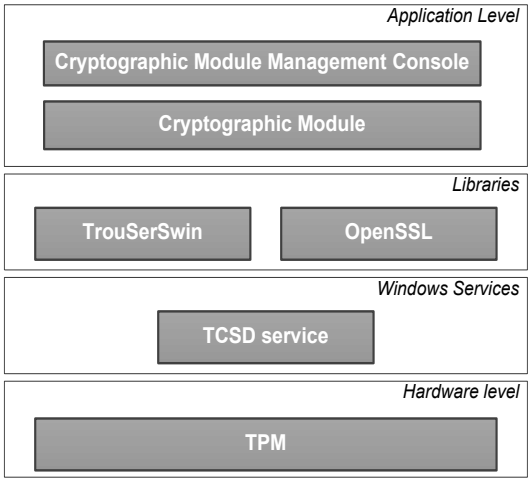

Fig. 8 Architecture of cryptographic module

The management console can be operated in a cryptographic module configuration mode and in TPM user management mode. The following functions are available in cryptographic module configuration mode:

- **login/logout** – opening / closing session of the cryptographic module configuration (entering the password for the owner of the TPM is necessary to opening the session);
- **reset TPM** – cleaning of TPM, i.e. delete all data from the cryptographic keystore and from TPM users' table;
- **initiate TPM** – creation of data structures for cryptographic keystore and for TPM users' table, and in particular the following keys: SRK, RK and NV_KEY.

A screenshot of the management console in configuration mode is shown on Fig. 9.

The following functions are available in user management mode:

- **Add user** – add a local user;
- **Remove user** – remove the user together with his keys stored in the cryptographic keystore;

[6] TCSD – software for management of TPM resources and support of local and remote requests. The software is part of TrouSerSwin software or TrouSerS software.

- **Generate keys** – generating the cryptographic keys for a local user;
- **Remove keys** – removal of previously generated cryptographic keys for the user;
- **Initiate import** – preparation of removable media to import external users data;
- **Import users** – import external users data;
- **Export users** – save the data of local user on previously prepared USB stick to transfer the data to another station.

A screenshot of the management console in user management mode is shown on Fig. 10.

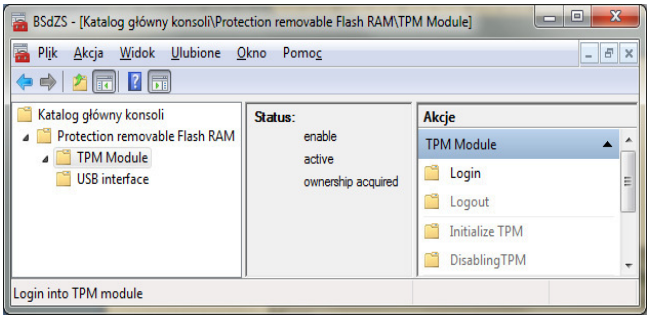

Fig. 9 A screenshot of the management console in Configuration Mode

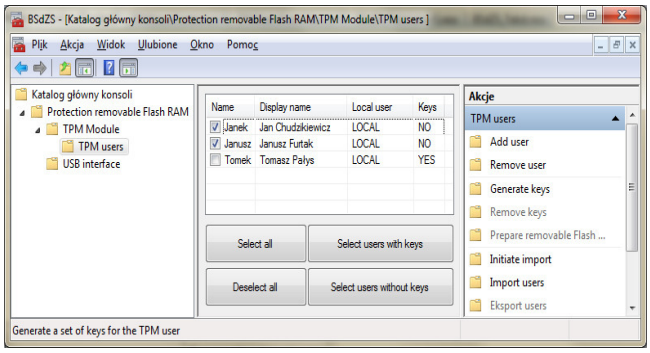

Fig. 10 A screenshot of the management console in User Management Mode

The "Add user" is designed to select from among the users defined in the system users who will be able to transfer protected files. For each selected user the GUID is generated. Data for these users are placed in TPM users' table.

The "Generate keys" is designed to generate the SK key and Legacy Key for local TPM user and put the keys in the keystore.

D. Export / import of user data

The procedure for preparing a protected file requires public key of the external user (file recipient) The source from which the keys are obtained, are stations where target user is a local user. The procedure for transfer of selected data about the local users of station ST2 to station ST1 includes the following activities (Fig. 11):

- transfer the public part of the RK key of ST1 station (e.g. via Flash RAM) - preparing a Flash RAM using the action "initiate import";
- providing media to the station ST2;
- preparation of a protected file that contains data about the local users of station ST2 using the action "Export users";
- providing the file and signature to the station ST1;
- decrypt user data, adding data these users (as an external users) to TPM users' table, and keys of that users to the keystore using the action "Import users".

Transfered protected file contains data of only selected local users of station ST2. The data of each user include:

- display name of user;
- Globally Unique Identifier of User – USER_GUID;
- identifier of the user Legacy Key - USER_LEG_GUID;
- public part of the user Legacy Key.

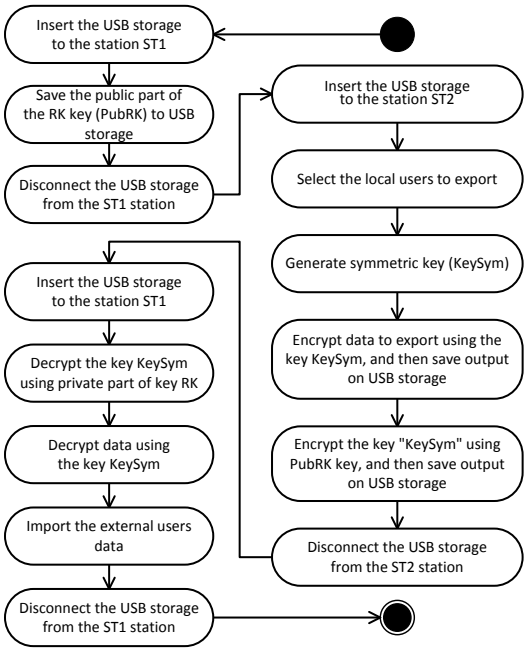

Fig. 11 Import data of users from station ST2 to station ST1

The file containing the user data is encrypted using a symmetric key generated at the station ST2. For that file is prepared other file (signature) containing symmetric key used to encrypt that file and value of SHA1 hash function for that file. The file (signature) is encrypted using the previously supplied public part of the key RK ST1 station.

At the station ST1 the received file containing signature is decrypted using the private key RK of the station ST1. In the next step the integrity of the file is checked up, then the file is decrypted using the symmetric key contained in the signature file.

The new records are created in TPM users' table basing on contents of received file. For each new external user in cryptographic keystore creates a hierarchy of keys containing the key SK and Legacy Key. The key SK is generated and will be identified by the sent user identifier USER_GUID. Lower in the hierarchy will be placed sent the Legacy Key.

IV. HANDLING FOR CREATING AND READING A SECURE FILE

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 **EnD**, which transmits information about them via the **DSu** to **CApp**.

Logged user (file sender) configures the parameters of the process of creating a protected file using **CApp**. These parameters are as the following:

- location of public key data recipient file ("Location of Public Key" field) – TPM is default (see Fig. 12), but it is possible provide the recipient public key from a disk (see Fig. 13);
- identifier for user (receiver) encrypted data ("Data Recipient" field) – one of the users whose keys are stored in the assets managed by the TPM;
- drive in which will be stored the protected file ("Data Drive" field);
- drive and path to the directory in which will be stored the file with the signature ("Signature Drive" field);
- identifier for the algorithm used to encrypt ("The encryption algorithm" field);
- identifier for the algorithm used to generate hash value for the protected file ("The Hash function algorithm" field).

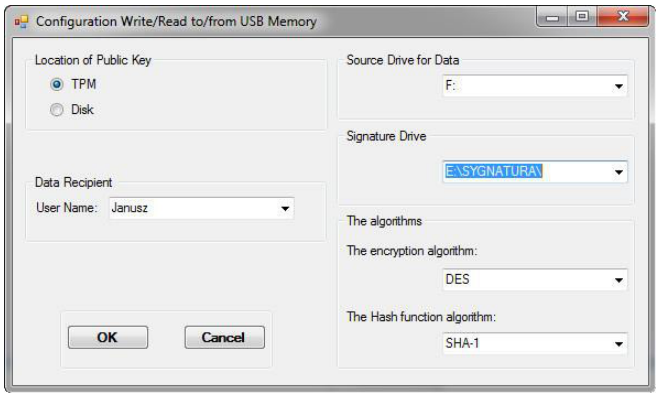

Fig. 12 The window of control application CApp for write data - recipient's public key is derived from the TPM

Identifier (O_ID) and the private key of the sender (the elements required to generate the signature) are automatically retrieved from the cryptographic keys storage managed by TPM. 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.

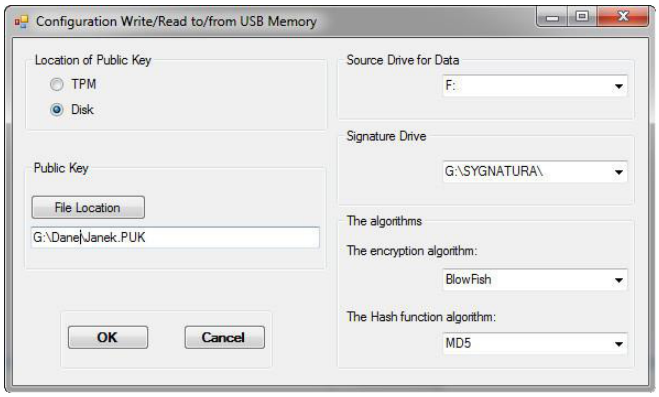

Fig. 13 The window of control application CApp for write data - recipient's public key is derived from disk

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 **EnD**, which transmit information about them via the **DSu** to **CApp**. The logged user (recipient of the data) using **CApp** has to specify the drive on which is stored encrypted file and indicate the file with the signature corresponding to the encrypted file. He accomplishes this by selecting (see the Fig. 14):

- drive on which is stored the protected file ("Data Drive" field);
- drive and path to the directory on which is stored the file with the signature ("Signature Drive" field).

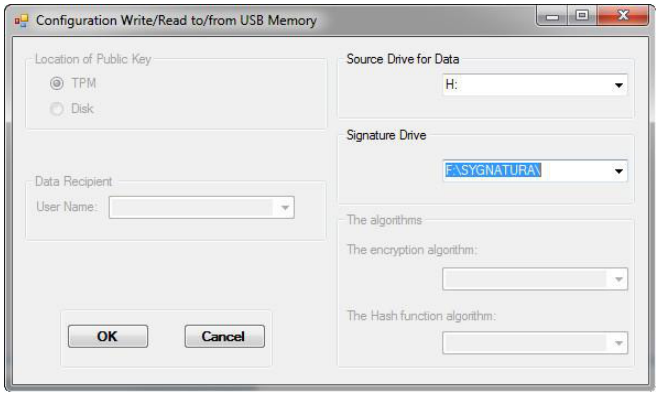

Fig. 14 The window of control application CApp for read data

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 **EnD** sends to the **DSu** 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 **EnD** the name of encrypted file, **DSu** 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 ***EnD***, which resumes the process of decryption. During the process of decrypting the file the ***EnD*** determines the value of a hash function for that file. After completion the process of copy ***EnD*** transmit to ***DSu*** 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 ***DSu*** deletes the file which has just been decrypted.

II. Conclusion

This paper presents a method for securing data on removable Flash RAM used to transfer sensitive data over insecure channels of transmission, such as a courier, traditional mail system and so on.

The developed method ensures the integrity and confidentiality of protected file. File sender (i.e., the creator of the protected content) is to ensure that the data will be available only to the specified recipient. The recipient can be sure that the resulting data are derived from the appropriate sender. Protecting the original file consists in preparing two files. The first one contains the encrypted contents of the original file (symmetric algorithm is here used). The second file is the signature for the encrypted file and contains the encryption key and the hash function value for encrypted file. The second file is encrypted using asymmetric cryptography. This approach ensures the safe transfer of the encryption key between stakeholders.

Both parties involved in transferring such protected data are forced to safely generate asymmetric keys and the safe theirs collection, storage and management. Particularly important is the proper procedure to generate symmetric key. For this purpose, the TPM has been used. This module supports (by using the hardware) many activities related to securing data on removable media and provides secure management of cryptographic keys.

From the safety point of view the most sensitive point of presented system for protecting the files transferred using removable Flash RAM is the procedure for export / import keys of external users. Particularly sensitive operation is "manual" transfer of a public key of station to which the data of users should be imported. The presented method is intended for use in an environment where it is not possible to use a computer network for data transfer. Such restrictions may apply in case of data transferring between systems, which belong to different security domains (i.e. with different levels of classification)[2][4] [12]. For this reason, it is not possible to use public key infrastructure (PKI) and Certificate Authority (CA). On the other hand organizational requirements for how to process data in such systems and how to use the removable media give a guarantee for safe transfer of the public key by the described procedure of export/import data. It should be noted that in cases where there are no obstacles to the use of computer networks in the presented procedure of export / import data is possible to use PKI and CA.

References

[1] J. Chudzikiewicz, "Zabezpieczenie danych przechowywanych na dyskach 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] J. Chudzikiewicz, J. Furtak, *Cryptographic protection of removable media with a USB interface for Secure Workstation for Special Applications,* Journal Of Telecommunications And Information Technology, no. 3, 2012, pp. 22-31.

[3] J. Chudzikiewicz, J. Furtak, *The method of secure data exchange using Flash RAM media*, Proceedings of the Federated Conference on Computer Science and Information Systems, 2012, pp. 621-626.

[4] A. Kozakiewicz, A. Felkner, J. Furtak, Z. Zieliński, M. Brudka, and Małowidzki M., *Secure Workstation for Special Applications***,** Lecture Notes in Computer Science, Vol. 187, pp.174-181, Springer 2011.

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

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

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

[8] R. Ng, Trusted *Platform Module. TPM Fundamental,* APTISS, August 2008 (http://www.cs.unh.edu/~it666/reading_list/Hardware /tpm_fundamentals.pdf)

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

[10] M. E. Russinovich, D. A. Solomon, A. Ionescu *Windows Internals Part 2*, Edition VI, Redmond, Washington, Microsoft Press, 2012.

[11] *TPM Main Part 1 Design Principles. Specification Version 1.2. Revision 116*, Trusted Computing Group, Incorporated, 2011

[12] *TCG Software Stack (TSS) Specification Version 1.2 Part1: Commands and Structures* (http://www.trustedcomputinggroup.org /files/resource_files/6479CD77-1D09-3519-AD89EAD1BC8C97F0 /TSS_1_2_Errata_A-final.pdf).

[13] Z. Zieliński, at all, *Trusted Workstation for Processing of Multi-level Security Data*, Journal of Telecommunications and Information Technology, 2012, pp. 5-12.