FACULTY OF ELECTRONICS,
TELECOMMUNICATIONS
AND INFORMATICS

Bezpieczeństwo Systemów Komputerowych – Projekt
Security of Computer Systems – Project

Tool for Emulating the Qualified Electronic Signature

Version 1.00, Gdańsk, 02.2024

1. The goal of project classes and the rules

The main goal of the project is to realize a software tool for emulating the qualified electronic signature, e.g. signing documents, and moreover basic ciphering operations. The goal is to fully emulate the process, including the hardware toll needed for person identification.

The project is marked regarding the rules (40 points in total):

- Correct realization of the project before the deadline, presentation during submission – 20 points.
- Technical report – 15 points.
- Presentation the project/concept (in a form of report) of the application and a part of application (during control term) – 5 points.

2. Project tasks

The main task of the project is to design and program an application to make a qualified electronic signature according to XAdES (XML Advanced Electronic Signatures) standard concept and to encrypt/decrypt small files (e.g. the documents, *.cpp files). In general, the application must take a form of a *set of tools for realization the qualified electronic signature*. The general concept is pointed in the Fig. 1.

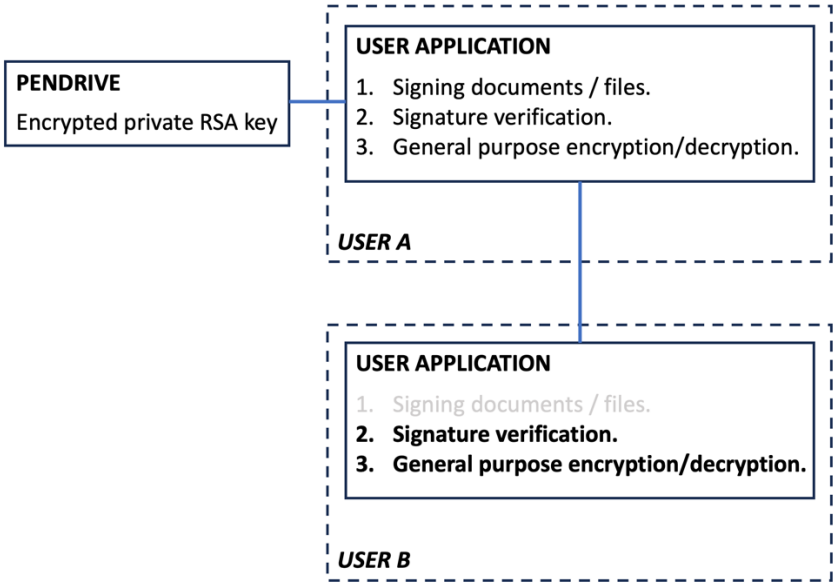

Fig. 1 – Block diagram of the project concept.

The user **A** has a hardware toll (a pendrive) with a private RSA key. The key is encrypted by AES algorithm using a user's PIN number. The application identifies the hardware tool and reads the private key to sign the document and generate the signing

report. Before signing the document, user **A** must enter the PIN number to decrypt the private RSA key stored on the pendrive. Regarding the XAdES standard, the electronic signature takes a form of the XML file containing details about the signature and signed document. The XML signature file must contain:

- General identification of the document (size, extension, date of modification).
- Information about the signing user.
- Encrypted by RSA private key hash of the document.
- Timestamp of the signature (local time of the user A).

The second user **B** must have a possibility to verify the signature (using the same application), by having the public key of user **A** and the document, XML signature file. During the verification user **B** generates the hash from the document and verifies it with the hash generated by user **A** (after RSA decryption).

Before using the application, the RSA keys must be generated for the user **A** and the private one must be encrypted by the AES algorithm and user **A** PIN number. It is obligatory to create second, simple application only with this functionality, it will emulate the trusted third party (TTP).

Another functionality is the basic encryption and decryption using the RSA keys. This functionality is partially used during signing the documents, but it must also be provided separately. This process can be tested on small files as the RSA algorithm is not designed to encrypt large files.

Requirements:

- The GUI interface must allow to select the document that will be signed (minimum two like **.pdf*, **.cpp*). Also, these types of files must be supported by the "Encryption / Decryption" functionality.
- The signature must use the RSA algorithm with a 4096-length key.
- The private key stored on the pendrive must be encrypted by the AES algorithm, where the key is the PIN number (and exactly hash from the PIN) known only to user A.
- Usage of the pendrive for storing the key is obligatory.

- It is obligatory to implement status/message icons to present the status of the application (recognition of hardware tool, status of signature, status of encryption).
- A pseudorandom generator must be used to generate the RSA keys.
- The public key can be stored on the hard disk of the computer or transfer to another physical computer to verify the signature.
- It is assumed that only user A can sign the documents, there is no need to create keys for two users.
- It is allowed to use the available implementations of the AES, RSA, SHA algorithms.
- Any language can be used to develop the application.
- In the report the results of performed tests must be included (e.g. signing scenarios, signature verification, encryption/decryption of other files).
- In the report the code of the application must be included in a form of listings, pointing the main functions of the application (also as a *.zip archive on eNauczanie platform).
- The operation of the application must be presented during project submission.

Notes:

- During realization of the project, it is obligatory to generate the following keys:
    - private and public RSA keys of the user A (used for signing and encrypting/decrypting),
    - local key for the AES algorithm (hash of PIN number) used for securing the RSA private key.
- It must be remembered that the parameters of the cipher (algorithm type, key size, block size, cipher mode, initial vector) must be set as constants.

3. Project submission

Control date submission:

- Partial report (recommended in the template).
- Code in a *.zip archive.
- Presentation during the classes.

Final date submission:

- Final report (recommended in the template).
- Code in a *.zip archive.
- Presentation during the classes, according to the "*Evaluation Card*".

The project report must contain a description of the applications' GUI interface, description of communication protocol, description of the secure key distribution method and results of performed tests (e.g. example transfers of different files - with different sizes, time of data transfer, usage scenarios).

Only one submission date is planned. In a case of obtaining insufficient number of points to obtain a positive mark from the project classes, a 2nd date for submission the project is proposed, but in that case 60% of points in total can be obtained.