**Security of Computer Systems**

**Project Report**

Authors:

Michał Warmbier 188889

Version: 1.0

**Versions**

|  |  |  |
| --- | --- | --- |
| Version | Date | Description of changes |
| 1.0 | 22.04.2024 | Creation of the document |
| 1.1 | … | … |

1. **Project – control term**
   1. ***Description***

The project is implemented in Python, leveraging two primary libraries: PyQt6 for the Graphical User Interface (GUI) and Crypto for essential cryptography functionalities. It's structured into two distinct applications:

* Application A: Responsible for generating and hashing RSA keys.
* Application B: Handles core functionalities like encryption, decryption, signing, and document verification.

The project repository can be accessed at:

<https://github.com/turboblaster44/BSK>

* 1. ***Results***

**Application A:**

A screenshot of a computer

Description automatically generated

Upon clicking "Generate RSA," two keys will populate the text boxes below. Following that, users can input a PIN number and select "Encrypt" to encrypt the private key using an AES algorithm. The encryption key will be the hashed PIN value, and the resulting encrypted private key will appear in the designated text box.

Subsequently, selecting either "Save encrypted private key" or "Save public key" will prompt a file explorer window for users to save the respective keys.

**Application B:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer error

Description automatically generated

The application revolves around four core functionalities:

* Encryption: Users can encrypt small documents (less than 512 bytes) using an RSA private key.
* Decryption: Encrypted documents can be decrypted using an RSA public key.
* Signature Generation: Files can be signed according to the XAdES standard, producing an XML file with:
  + Signature Verification: Signatures can be verified using the RSA public key by comparing the specified file hash with the signature hash.
  + General document identification (size, extension, date of modification).
  + User information.
  + RSA private key hash of the document.
  + Timestamp of the signature (local time of user A).
* Signature Verification: Signatures can be verified using the RSA public key by comparing the specified file hash with the signature hash

All the necessary files needed to be for the mentioned utilities need to be specified by the “*Choose file*” button

In addition to these key functionalities, the application provides features for:

* Loading RSA keys with PIN: Users can load private and public keys, with the private key requiring decryption using a PIN.
* Status Messages: The application offers informative messages, including error notifications or success confirmations, to aid users during their interactions.
  1. ***Summary***

The apps allow for:

* Generating RSA keys, with proper encryption and decryption mechanisms
* Encrypting and decrypting small files using the RSA keys.
* Generating an XML signature of a file according to the XAdES standard and verifying it.

All the necessary functionality has been implemented aside from the automatic key reading from the pendrive.

1. **Project – Final term**
   1. ***Description***

Content

* 1. ***Code Description***

Content

|  |
| --- |
| /\*!  \* A list of events:  \* <ul>  \* <li> mouse events  \* <ol>  \* <li>mouse move event  \* <li>mouse click event<br>  \* More info about the click event.  \* <li>mouse double click event  \* </ol>  \* <li> keyboard events  \* <ol>  \* <li>key down event  \* <li>key up event  \* </ol>  \* </ul>  \* More text here.  \*/ |

*List. 1 – Code listing [2].*

Final Content.

* 1. ***Description***

Content

* 1. ***Results***

Content

* 1. ***Summary***

Content

1. **Literature**

[1] Article.

[2] Online Doxygen documentation, <https://www.doxygen.nl/manual/lists.html>, (accessed on 18.02.2024).

[3] Book.