National College of Ireland

RAE - Configuration Manual

MSc Research Project
MSc in Cybersecurity

Ansh Ashwini Jain
Student ID: 23308320

School of Computing
National College of Ireland

Supervisor: Mr. Jawad Salahuddin

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Ansh Ashwini Jain
Student ID:	23308320
Programme:	MSc in Cybersecurity **Year:** 2025
Module:	Practicum - II
Lecturer:	Mr. Jawad Salahuddin
Submission Due Date:	11th August 2025
Project Title:	RAE – Configuration Manual

Word Count: 1259 **Page Count:** 7

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Ansh Ashwini Jain

Date: 11th August 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

RAE - Configuration Manual

Ansh Ashwini Jain
Student ID: 23308320

Contents

1 Introduction

The aim with this research is to showcase an algorithm combined with multiple techniques in a hybrid mix to enhance the effectiveness with which data can be encrypted to promote security. The main objectives are as follows:

- Implementation happens by taking a plaintext and first encrypting it with AES with Cipher Block Chaining Method and padding PKCS#7 with a key and Initialization vector and then using the ciphertext obtained to encrypt it using RSA with high value prime keys. Now this AES-RSA key obtained is encrypted further using ECC over the 'brainpoolP256r1' curve (*Profile SecurityPolicy – ECC-brainpoolP256r1*, no date), obtaining the AES-RSA-ECC encryption key.

- Now the decryption occurs in following the reverse of the above order with ECC first, then through RSA which gives the AES ciphertext obtained during encryption, (used for verification). Now this ciphertext is used for further decryption by passing through AES, giving us our original obtained plaintext.

By implementing this whole method, this research aims to create a robust encryption algorithm which can be used to evolve and used for various data encryption and decryption scenarios which can be used to show an effect on security.

2 System Configuration

The above technique as explained combines Rivest-Shamir-Adleman (RSA), Advanced Encryption Standard (AES) and Elliptic Curve Cryptography (ECC) (*Encryption Standards: AES, RSA, ECC, SHA and Other Protocols*, 2024) to create the said hybrid method which makes a new pick in the encryption and decryption world. Now to make use of and run the techniques wisely, there were multiple certain tools used to run and test the technique. A

hardware device computer with Software like Cryptool2 and a code editor platform to run and edit Python programs was used. The device must use a reliable and secure storage system to ensure the protection of the encrypted information. Overall, the hybrid approach can provide an effective and secure encryption solution across different applications, as its efficiency relies on the device's computational power, available memory, and storage capabilities.

2.1 Hardware Configuration

To efficiently process large datasets, a device with sufficient Random Access Memory (RAM) is recommended, as the method relies on memory resources. AES and RSA are implemented using CrypTool2 (*About CrypTool 2 – CrypTool*, no date), while ECC is handled through Python libraries, functions, and scripts—which may require additional storage space. Adequate computational capacity is essential for executing the complex mathematical operations involved in RSA, AES, and ECC. Moreover, a modern multi-core CPU can significantly enhance the speed of both encryption and decryption processes.

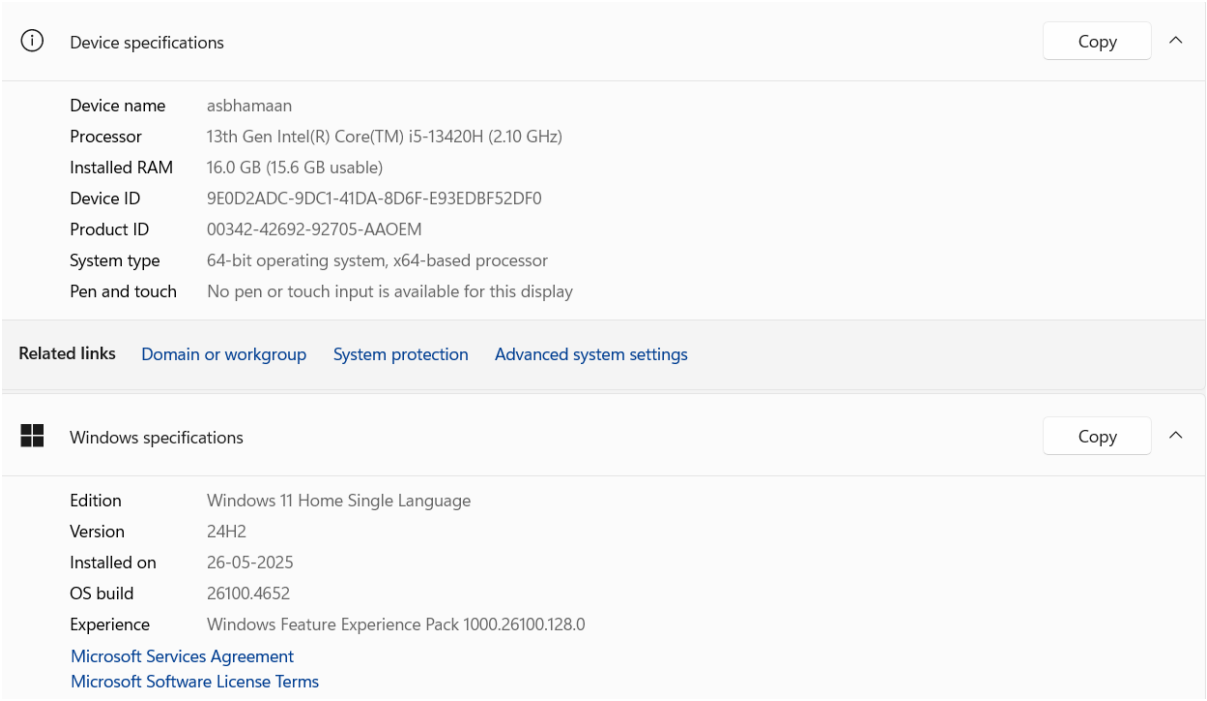

Fig 1. Hardware Specifications

2.2 Software Configuration

CrypTool2: CrypTool2 is used as a core platform to implement and visualize the RSA and AES components of the hybrid encryption method, offering an interactive environment to simulate and understand their cryptographic operations. Its modular workflow design makes it easy to integrate different encryption steps, allowing clear representation of key generation, encryption, and decryption processes. This combination allows for educational clarity, practical implementation, and modular experimentation within a single framework.

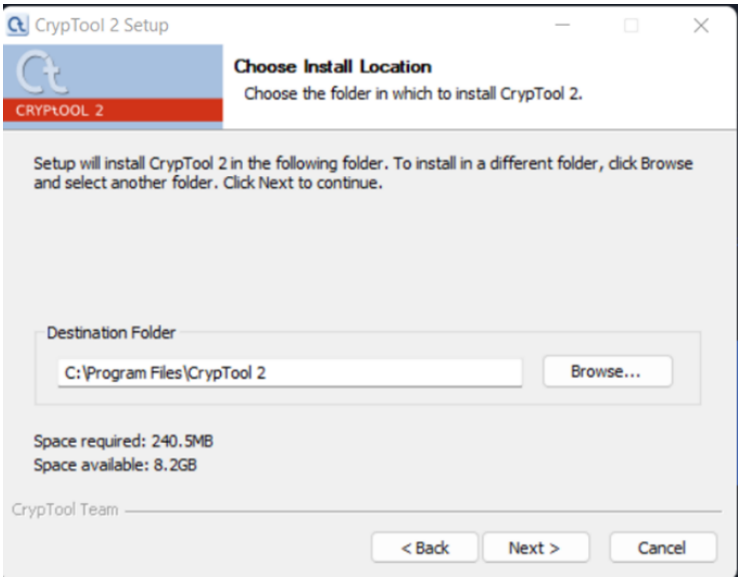

Fig 2. CrypTool2 Setup

Visual Studio Code for Python: Visual Studio Code, combined with Python, (*Visual Studio Python IDE - Python Development Tools for Windows*, no date) serves as an effective development environment for implementing the Elliptic Curve Cryptography (ECC) component in the RSA+AES+ECC hybrid encryption model. Python's rich set of cryptographic libraries, such as cryptography, pyca, and ecdsa, enables the generation of ECC key pairs, point multiplication, and secure key exchange with high precision and flexibility. Visual Studio Code enhances this process through features like code linting, real-time debugging, and seamless integration with version control systems. This setup allows for efficient development, testing, and integration of ECC with the RSA and AES modules, particularly when CrypTool2 handles RSA and AES operations separately.

AIDA64 Extreme: AIDA64 Extreme (*AIDA64 Extreme | AIDA64*, no date) is utilized as a benchmarking and diagnostic tool to test the hardware performance during the implementation of the RSA+AES+ECC hybrid encryption model. By monitoring system parameters such as CPU load, memory usage, and thermal stability, AIDA64 helps ensure that the encryption and decryption processes do not overburden the system. It provides real-time insights into how efficiently the system handles cryptographic computations, particularly during ECC key generation or AES block encryption. This allows for optimization of the implementation by identifying hardware bottlenecks and validating that the system meets the necessary performance requirements for secure and reliable encryption.

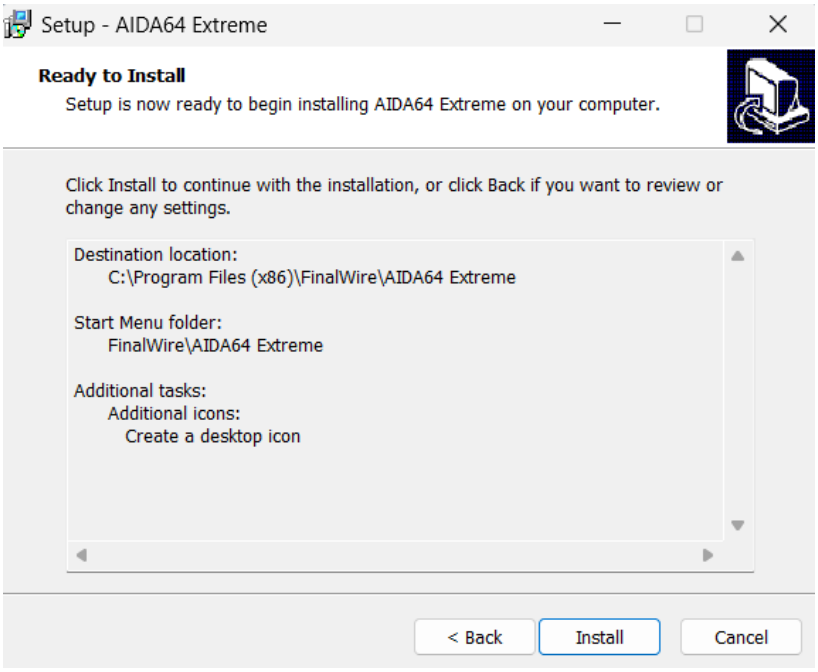

Fig 3. AIDA64 Setup

3 Implementation

Step 1: Open the ‘input.txt’ file and enter the plaintext you want to encrypt. And save the file.

Fig 4. Plaintext Input Screenshot

Step 2: Open the CrypTool 2.1 Software and the ‘AES-RSA.cwm’ file which will encrypt and provide the AES-RSA encrypted hexadecimal string. AES encrypted Base-64 string can be used later on to verify exact message is recovered during decryption, AES-RSA encrypted output string is used further in the process.

Step 3: Paste the AES-RSA encrypted output string in Visual Studio Code for the ECC encryption and decryption outputs. ‘Ciphertext.txt’ file is generated to save the ECC, ciphertext output. The decrypted AES-RSA output string is saved as ‘input2.txt’ file.

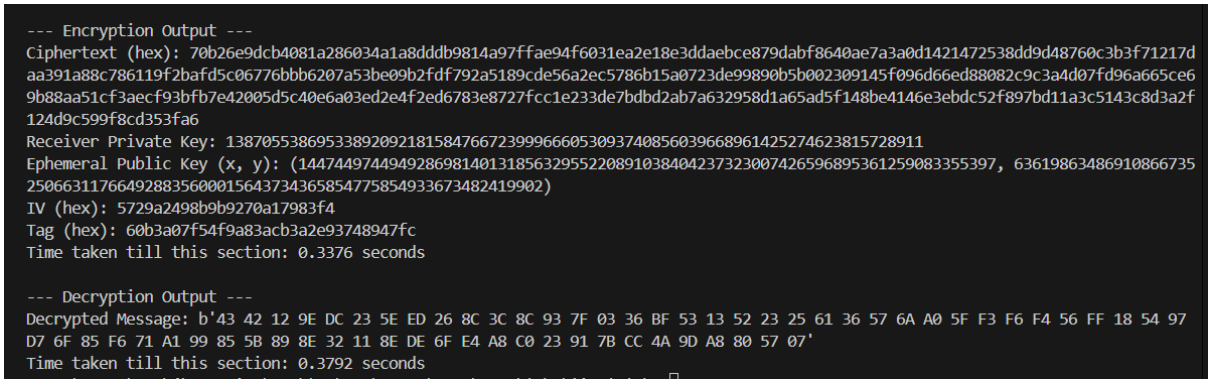

Fig 5. Python ECC Output

Step 4: Open ‘decrpyt RSA_AES.cwm’ file in cryptool2.1 and run the program.

Step 5: Open the ‘output.txt’ file and verify the decrypted plaintext.

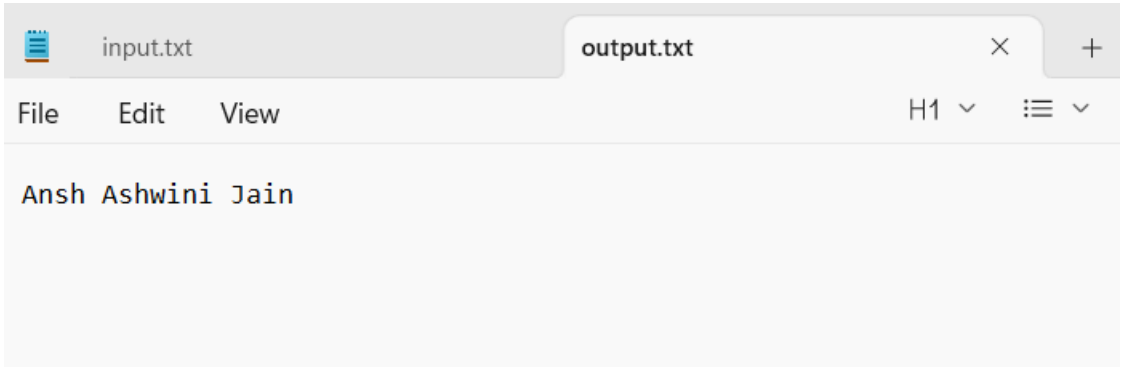

Fig 6. Decrypted output plaintext

References

[1]. *About CrypTool 2 – CrypTool* (no date). Available at: https://www.cryptool.org/en/ct2/ (Accessed: 7 August 2025).

[2]. *AIDA64 Extreme | AIDA64* (no date). Available at: https://www.aida64.com/products/aida64-extreme (Accessed: 7 August 2025).

[3]. *Encryption Standards: AES, RSA, ECC, SHA and Other Protocols* (2024) *DEV Community*. Available at: https://dev.to/hardy_mervana/encryption-standards-aes-rsa-ecc-sha-and-other-protocols-460c (Accessed: 7 August 2025).

[4]. *Profile SecurityPolicy – ECC-brainpoolP256r1* (no date). Available at: https://profiles.opcfoundation.org/profile/2066 (Accessed: 7 August 2025).

[5]. *Visual Studio Python IDE - Python Development Tools for Windows* (no date) *Visual Studio*. Available at: https://visualstudio.microsoft.com/vs/features/python/ (Accessed: 7 August 2025).