Project Progress Report

Harsh Manishbhai Siddhapura

Vaibhavi Nitin Honagekar

Arunava Lahiri

Thembelihle Shongwe

Sai Shashank Nagavaram

Anandha Krishnan Senthooraan

Group: Class96958 4

Ira A. Fulton School of Engineering, Arizona State University

IFT 520: Advanced Information Systems Security

Prof. Upakar Bhatta

October 27, 2023

“An asymmetric encryption application that automatically encrypts outgoing emails with a public key or decrypts incoming emails with a private key based on a database that houses the selected email addresses and asymmetric data”

Progress

What have you done till now?

As of now, we have achieved several critical milestones in the development of our Python application for email encryption and decryption. The primary accomplishment is the successful implementation of the RSA encryption algorithm, which serves as the foundation for securing email communication. Our application utilizes this algorithm to demonstrate how public and private keys are used to encrypt outgoing emails and decrypt incoming ones, ensuring the privacy and integrity of the messages.

To provide a realistic testing environment, we have simulated a database containing email addresses along with their corresponding public and private keys. This simulated database mirrors the structure of a real-world system, allowing us to validate the core functionality of our application. Through this simulation, we can emulate the process of sending encrypted messages to specific recipients and subsequently decrypting them with the recipient's private key. The Python code driving this simulation is well-organized and comprehensible, setting a solid groundwork for future development.

The encryption process within our application relies on the recipient's public key, a fundamental aspect of asymmetric encryption. This approach guarantees that only the intended recipient, possessing the corresponding private key, can decrypt and access the message's content. Conversely, the decryption process involves using the recipient's private key to reveal the original message. These functionalities have been successfully implemented and thoroughly tested. While the application is currently a proof of concept, it offers valuable

insights into the intricate workings of asymmetric encryption, key management, and secure email communication, setting the stage for further enhancements and real-world integration.

It could be summarized as below:

- Developed Python code for an asymmetric encryption application that simulates the encryption and decryption of outgoing and incoming emails using RSA encryption.
- Created a simulated database that stores email addresses and their associated public and private keys.
- Implemented functions to generate public and private keys for email addresses.
- Wrote functions for encrypting and decrypting messages based on the recipient's public key and the sender's private key, respectively.

```
Comment Code
def encrypt_message(recipient_email, message):
    # Simulate sending an email
    if recipient_email in database:
        recipient_public_key = database[recipient_email]["public_key"]
        if recipient_public_key:
            # Encrypt the message with the recipient's public key
            ciphertext = recipient_public_key.encrypt(
                message.encode(),
                padding.OAEP(
                    mgf=padding.MGF1(algorithm=hashes.SHA256()),
                    algorithm=hashes.SHA256(),
                    label=None,
                ),
            )
            return ciphertext
        else:
            return "Recipient's public key not found."
    else:
        return "Recipient email not found in the database."
```

```
# Generate public and private keys for each email address
for email, keys in database.items():
    private_key = rsa.generate_private_key(
        public_exponent=65537, key_size=2048
    )
    keys["private_key"] = private_key
    keys["public_key"] = private_key.public_key()
```

Comment Code

```
def decrypt_message(email, ciphertext):
    # Simulate receiving an email
    if email in database:
        private_key = database[email]["private_key"]
        if private_key:
            # Decrypt the message with the recipient's private key
            plaintext = private_key.decrypt(
                ciphertext,
                padding.OAEP(
                    mgf=padding.MGF1(algorithm=hashes.SHA256()),
                    algorithm=hashes.SHA256(),
                    label=None,
                ),
            )
            return plaintext.decode()
        else:
            return "Private key not found."
    else:
        return "Email not found in the database."
```

Future Plan

What is your plan for the future?

Our future plans for the application are focused on two core areas: enhancing the database management and improving the user interface (UI). These strategic upgrades will significantly contribute to the application's functionality and usability.

Firstly, we plan to transition from the current simulated database, implemented as a Python dictionary, to a real and secure database management system. This upgrade is paramount for ensuring data security, scalability, and data persistence in a real-world context. We will carefully assess and select an appropriate database solution that aligns with the performance requirements of the application while adhering to data protection and privacy regulations. The database will serve as the backbone for storing email addresses, public keys, and encrypted messages. By making this transition, we will enhance the application's reliability, security, and overall practicality.

Secondly, we intend to focus on the improvement of the user interface (UI). Designing an intuitive and user-friendly UI is critical for facilitating user interactions and ensuring a positive user experience. Our plan is to create a visually appealing and intuitive interface that accommodates both novice and experienced users. The upgraded UI will simplify the process for end-users, enabling them to manage email addresses, send encrypted messages, and decrypt incoming emails seamlessly. By investing in UI enhancements, we aim to make the application more accessible and user-centric, encouraging broader adoption and utilization.

Additionally, as part of our future plan, we will continue to focus on maintaining the security and privacy of the application. Security remains a top priority, particularly as the application deals with sensitive information, including private keys and encrypted messages. We will ensure that the database upgrade adheres to the highest standards of data security and encryption. Regular security audits and updates will be incorporated into our

development process to mitigate potential vulnerabilities. By maintaining a strong security posture, we aim to provide users with confidence in the application's ability to protect their sensitive communications effectively.

By concentrating on these two key areas, we aim to refine the application to a state where it is both practical and user-friendly. These upgrades will pave the way for a more secure, efficient, and user-centric email encryption solution.

Challenges

Explain any challenges and issues that you have encountered.

Throughout the development of the email encryption application, we confronted several challenges and issues that have guided our decision-making process and emphasized the importance of certain aspects in our future plans. One of the initial challenges was the simulation of a database using a Python dictionary. While this approach served the purpose of creating a proof of concept, it becomes clear that transitioning to a real and secure database management system is essential. Ensuring data security, scalability, and data persistence in a real-world application is significantly more complex. We need to carefully evaluate and select a suitable database solution that not only meets the performance requirements of the application but also complies with data protection and privacy regulations. The choice of the right database system will be a pivotal step in the application's evolution.

Designing a user-friendly graphical user interface (GUI) presented another challenge. The GUI is a critical component that will directly impact the application's usability and accessibility. It requires in-depth knowledge of user experience (UX) and user interface (UI) design principles. Balancing the needs of both novice and experienced users while maintaining a visually appealing and intuitive interface is a multifaceted task.

Achieving an effective GUI design necessitates conducting user testing and feedback gathering to refine the interface continually.

Security remains a paramount concern in the development of the application. Dealing with sensitive information such as private keys and encrypted data demands rigorous security measures. Securely storing private keys and encrypted messages is of utmost importance. Our future plans will prioritize robust security implementations in the application, including encryption algorithms, secure storage, and key management practices. We will adhere to established best practices in cryptography and data security to ensure the highest level of protection for user data.

Additionally, we must consider security measures for email attachments, which may pose additional challenges due to the variety of file types and sizes that could be encountered in real-world usage. Addressing these security challenges will be integral to the application's reliability and trustworthiness.

Time Table

The projected timetable for the "Asymmetric Encryption Application" project encapsulates the comprehensive journey toward developing an innovative and secure email encryption solution. This timeline provides a structured framework that outlines the project's distinct phases, each bearing its significance in the application's development. By adhering to this timeline, the project team aims to seamlessly transition from research and technology selection to code development, testing, and eventual deployment. Over an estimated duration of 10 weeks, the team will meticulously traverse through the diverse facets of this project, ensuring that the final product aligns with the initial objectives of creating an email encryption application that automates the encryption of outgoing emails and the decryption of incoming emails based on a secure database. The typical timeline of the project is as follows:

- **Topic Selection (Done):** The project's primary goal is to create an asymmetric encryption application for enhancing email communication security. The application will automatically encrypt outgoing emails using the recipient's public key and decrypt incoming emails with the recipient's private key, making use of a database to store email addresses and associated asymmetric key pairs. This initial phase involves outlining the project's scope and objectives, including defining the core functionalities of the application and identifying the technologies required.

- **Background Research (Done):** To ensure the project's success, a thorough background research phase is crucial. This step involves studying existing email encryption technologies, understanding the principles of asymmetric encryption, and exploring database management systems suitable for storing email addresses and keys securely. The research will also cover best practices in email security, encryption algorithms, and key management.

- **Technology Selection (Done):** The project will then proceed to select the appropriate technologies and tools for development. This includes choosing the programming language (e.g., Python), libraries, and frameworks that best suit the application's requirements. Additionally, we will select a database management system for secure data storage and explore options for creating the user interface.

- **Basic/Skeleton Code (Done):** In this phase, the project will begin coding the fundamental components of the application. The code will establish the core encryption and decryption functionalities using asymmetric encryption algorithms such as RSA. The primary focus will be on creating a working prototype that can encrypt and decrypt emails using simulated data.

- **Database Integration (In Progress):** The next step involves integrating the application with a real database system. Transitioning from the simulated dictionary database to a secure and scalable database management system is critical for the application's practicality and data security.

- **UI Development (In Progress):** A user-friendly interface is essential for widespread adoption. The UI development phase will focus on designing an intuitive and accessible graphical user interface that simplifies the encryption and decryption process for end-users. The interface will allow users to manage email addresses, send encrypted messages, and decrypt incoming emails seamlessly.

- **Test the Application for Security and Functionality:** Before deployment, the application will undergo rigorous testing to ensure both security and functionality. Security testing will include vulnerability assessments and penetration testing to identify and rectify potential security weaknesses. Functional testing will assess the application's usability, performance, and overall user experience.

- **Perform Additional Testing and Prepare for Potential Deployment:** The final phase includes additional rounds of testing to address any issues identified in the initial testing phase. Once all aspects of the application meet the desired standards, the project will prepare for potential deployment. This involves creating installation packages, user guides, and any necessary documentation for end-users. Additionally, the team will plan for ongoing maintenance and updates to ensure the application's long-term viability and security.

The estimated total project duration is approximately 10 weeks. However, it's important to note that timelines may be subject to adjustments as the project progresses, and real-time challenges and opportunities arise. Regular project monitoring and communication will be essential to ensure that the project stays on course and delivers the intended results. Throughout the project, regular communication and collaboration among team members will be essential to monitor progress, address challenges, and ensure that the project remains on track.