Information Systems Security CS - 446

Project Report

for

Encrypted QR Code

Submited By

Mehdi Raza Rajani - 16k3904

Tahir Raza Hemani – 16k3905

Junaid Mazhar - 16k3874

(Students of Section - F)

Course Instructor

Dr. Sufian Hameed

January 1, 2020

1 Introduction

1.1 Background

QR Codes are two-dimensional bar codes that are used to transmit information across all over the world. QR Code is used in daily life for the purpose of advertising, mobile payments, access control, and navigation. A person can easily encode the information in a QR Code and the recipient can decode it from a simple QR Code Scanner by scanning the QR Code. QR Code has made the transmission of information easy.

1.2 Problem Statement

With the widespread use of QR code comes its own unique challenge that is how to protect information from preying eyes or hands?. In the past years, a number of events have been witnessed in which QR Code was falsely modified for the purpose of phishing and stealing sensitive personal information. Therefore, this project aims to make secure encrypted QR code.

1.3 Scope

The scope of this project is:

- Encryption and Decryption of the Message
  - Symmetric Encryption Techniques
  - Asymmetric Encryption Techniques
  - Authenticated Encryption Techniques
- Generate the QR Code
- Read QR Code

2 Methodology & Architecture

The following diagrams illustrates the flow of encryption and decryption.

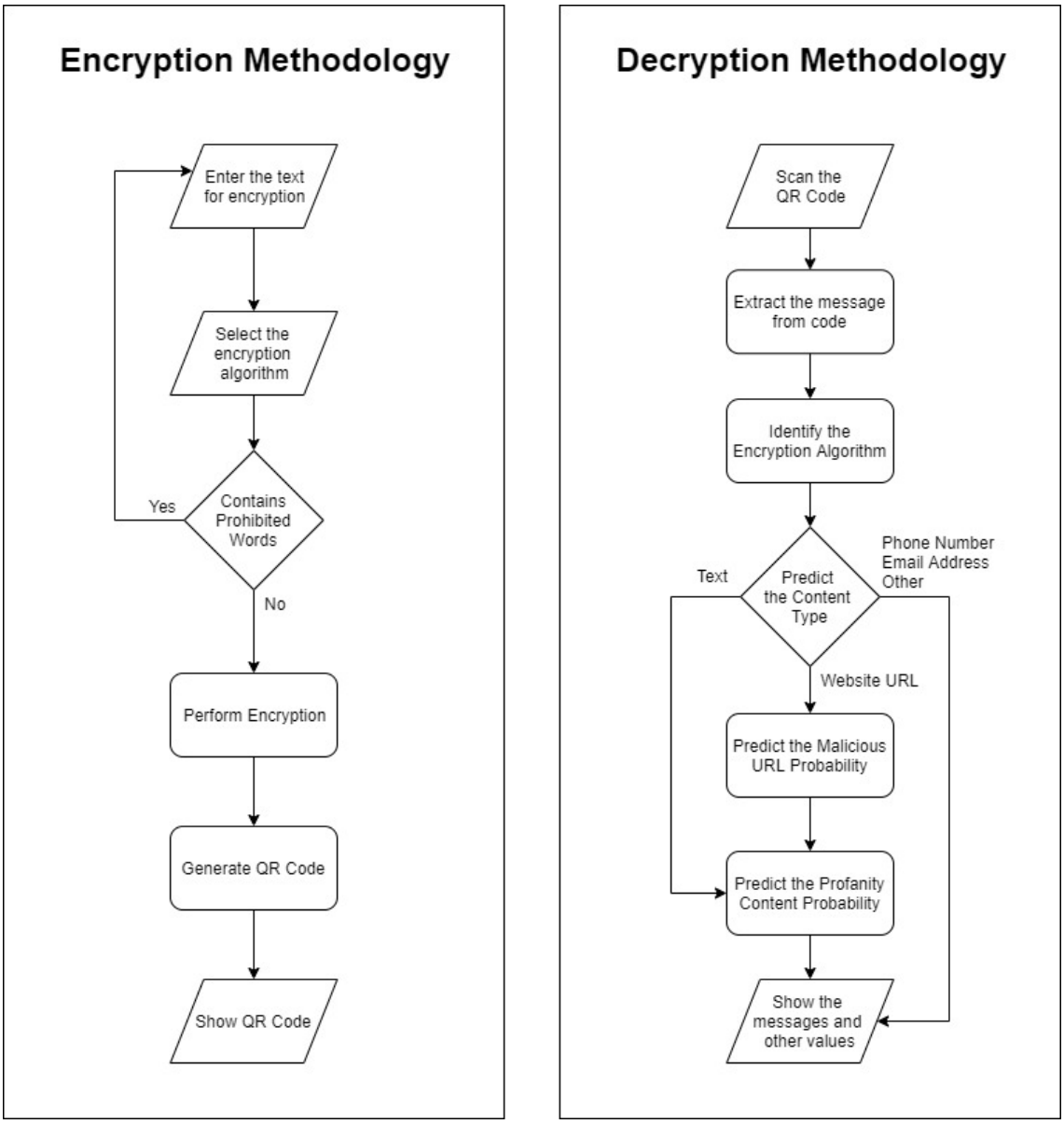


Figure 1: Process of System

2.1 Key Generation

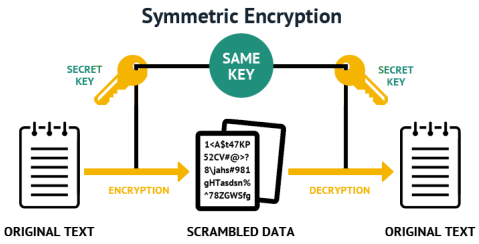
The public private key are being generated using JAVA Security RSA Generate Function. It returns a public private key pair. For symmetric algorithm we keep a hardcoded key in the system.

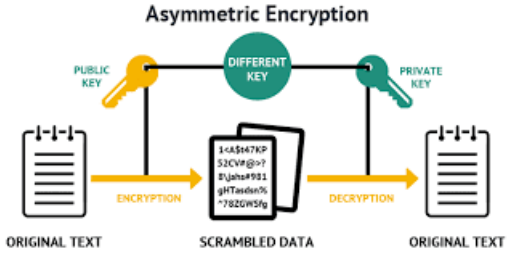
2.2 Algorithm that are being used

1. **Sym_AES_GCM_NoPadding:**
   Galois/Counter Mode (GCM) is a block cipher mode of operation that uses universal hashing over a binary Galois field to provide authenticated encryption. It provides high integrity and security. There was no padding used in this.

2. **Sym_AES_CBC_PKCS5Padding:**
   This is also a block cipher, it gives an output of 128bit and it also has padding in it.

3. **Assym_RSA_ECB_PKCS1Padding:**
   This is a block cipher with RSA, and there is padding as well.





2.3 QR Code Generation

Zxhing library is being used to generate QR code, as well as to read QR code.

2.4 Identifying Algorithm

For ease we have added 1_,2_,3_ appended to the string at the time of encoding. 1_ is for the three algorithm mentioned above. At time of decryption we remove the 1_,2_ or 3_ appended string of hexadecimals.
Incase users tries to read a QR Code which is not been generated from our application he can do that also. If the decoded string does not starts with [1,2,3]_ this means that the QR Code is non encrypted.

2.5 Blocking Offensive Words

Using google list of the words that it block in Google Search, we created an API. Now at the of encryption we call API to check it contains prohibited words or not. This is to block offensive words.

2.6 Predicting Content Types

Regex were used to predict content type that is being input.

```
text_regex = r"^[a-zA-Z]+(([a-zA-Z ])?[a-zA-Z]*)*$"
email_address_regex = r"^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$"
phone_number_regex = r"^[+]*[(]{0,1}[0-9]{1,4}[)]{0,1}[-\s\./0-9]*$"
website_regex = r"^(https?:\/\/)?(www\.)?([a-zA-Z0-9]+(-?[a-zA-Z0-9])*\.)+[\w]{2,}(\/\S*)?$"
```

2.7 Predicting Malicious URL

We found a dataset of about 2500 malicious URL and 2500 non malicious url. 3 models were trained then:

- Logistic Regression
- Light Gradient Boosting Tree
- Random Forest CHI Square

Mean of these values is used to predict whether URL is malicious or not.

2.8 Profanity Content Check

Python library (profanity_check) is being used that checks profanity. This can tell if url that is being added is profane or not, furthermore if a QR code that is not generated from our application and it has a profane url, we will tell the user using this check that see as per risk.

3 Conclusion

The application works as per expectation. Assymetric and Symmetric both types of encoding are working and they provide good results. Along with that we have checked for content type, malicious URL, and profanity checks as well.

4 References

- Research Papers
    1. https://www.ieee-security.org/TC/SP2019/posters/hotcrp_sp19posters-final33.pdf
    2. https://publications.sba-research.org/publications/llncs.pdf
    3. https://iopscience.iop.org/article/10.1088/1742-6596/1019/1/012008/pdf
- Python Profanity Check Library
  https://pypi.org/project/profanity-check/
- Data for Malicious URL
  https://www.kaggle.com/softline/machine-learning-model-to-identify-malicious-url
- List of Restricted Words
  https://github.com/RobertJGabriel/Google-profanity-words
- JAVA Docs
    1. https://docs.oracle.com/javase/7/docs/api/javax/crypto/Cipher.html
    2. https://www.mkyong.com/java/java-asymmetric-cryptography-example/
- GitHub Repository Link
  https://github.com/mehdirazarajani/EncryptedQRCodes