Faculty of Engineering
Cairo University

Cryptography project Report

Supervised by: Dr.Samir Shaheen

Name	Sec	Bench number
Ahmed Ayman	1	1
Omar Ahmed Mohamed	2	3

Table of contents

Project Idea

Our project is a web application-based chat system with rooms to chat and consists of two parts: front-end, back-end (node.js, express, and socket.io), and database MongoDB to store the chat. This chat system uses the RSA encryption algorithm to encrypt messages for secure chatting.

How it works

At the beginning of the connection, both sides send each other the public keys used in the encryption system.
If anyone wants to send any message the system will encrypt it using RSA, then send it via socket.io which the receiver will decrypt using his private key, and vice versa.

RSA encryption algorithm modules

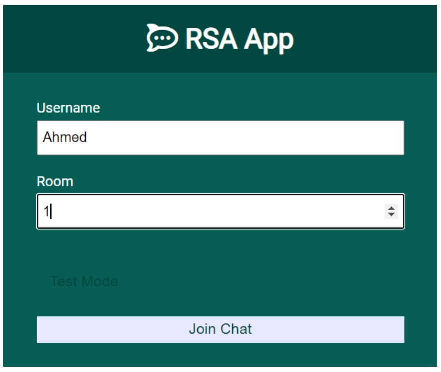
I- Key generation: this function takes the size of the key and generates {P: first prime, Q: second prime, E: private key}. It simply selects a random number and checks if it is prime or not in the case of {P, Q}, but for {E} it checks if the random number is less than phin and coprime with it.

II- Encrypt message: it does three main tasks. The first task split sent the message to sub-messages to decrease message length based on this condition ($m < n$). The second task is to encrypt the sub-message based on this equation ($c = pow(m, e) \bmod n$). Finally, it converts the encrypted message to string.

III- Decrypt message: the first task is to convert ciphertext from string to integer. decrypt each sub-message with the private key based on this equation ($m = pow(c, d) \bmod n$), private key is calculated based on this equation ($ed = 1 \bmod(phin)$). Finally, it joins sub-messages to construct the original message.

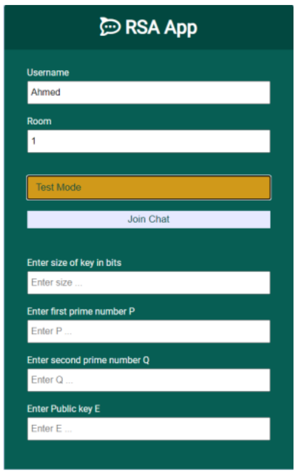
Project modes

I- Normal mode where the user enters his (username, roomId) to start chatting.
II- Test mode where user control of RSA encryption algorithm parameters to create custom (private, public), where the user must enter {N: key size} or {P: first prime, Q: second prime, E: public key}.



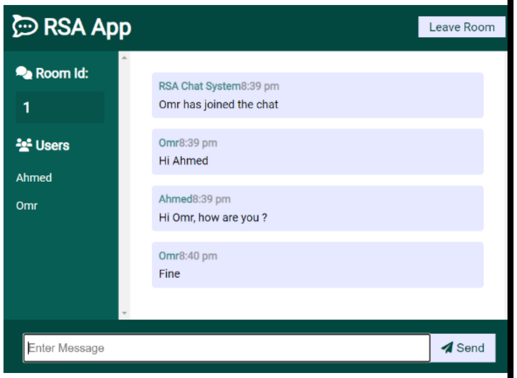
Username: Ahmed
Room: 1

Figure-1: normal mode



Username: Ahmed
Room: 1

Figure-2: test mode



Room Id: 1
Users: Ahmed, Omr

RSA Chat System8:39 pm
Omr has joined the chat

Omr8:39 pm
Hi Ahmed

Ahmed8:39 pm
Hi Omr, how are you ?

Omr8:40 pm
Fine

Figure-3: chat sample

Project Constraints

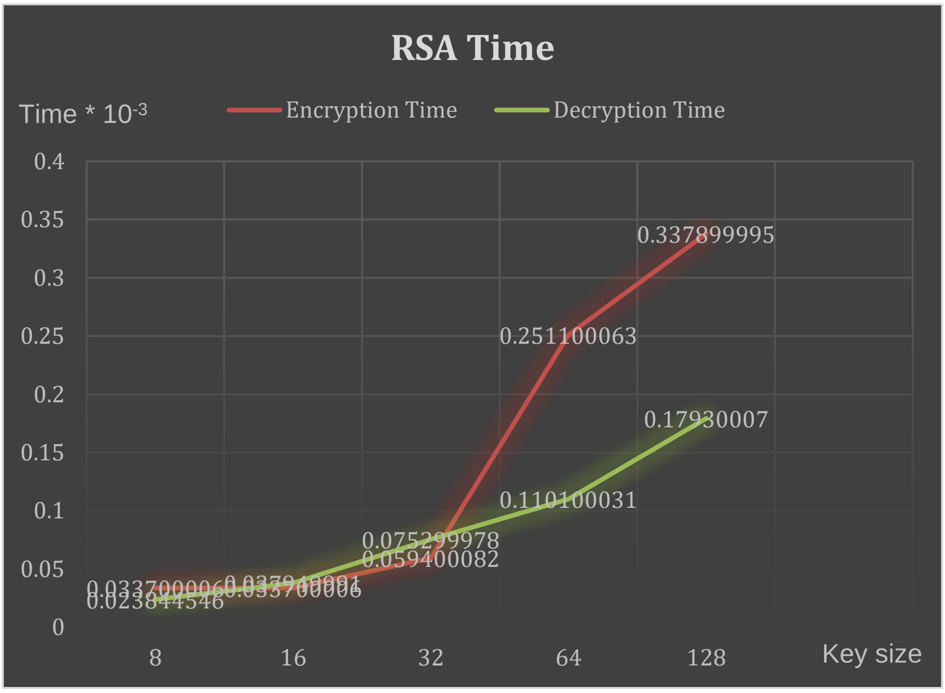
I- The Maximum limit of users in one room is two members.

II- Two members of the room must be online to start chatting, No one can send messages inside the room while the other member is not connected.

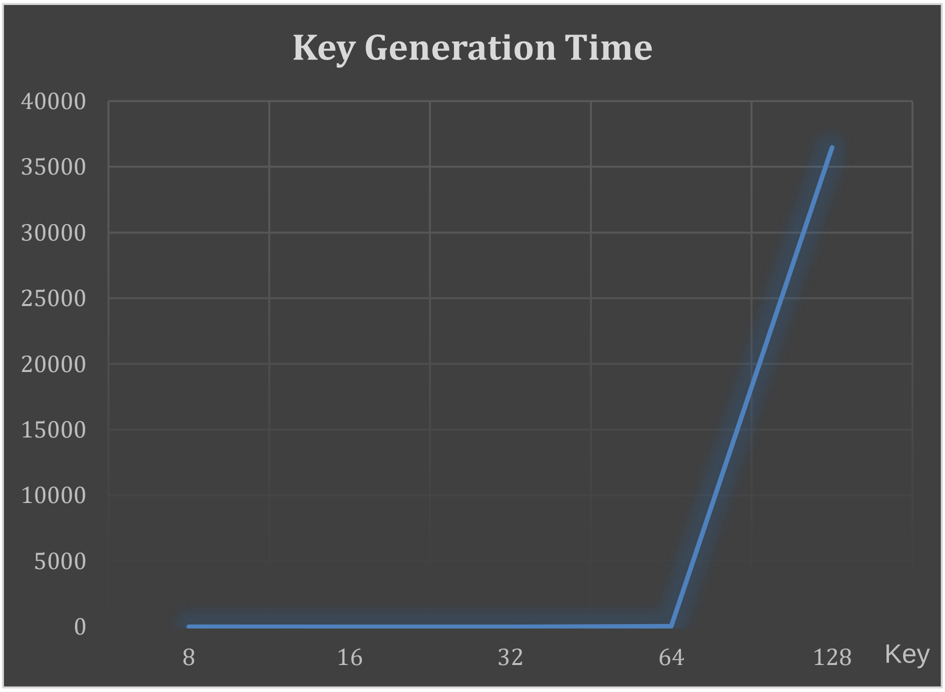
III- In test mode users must enter key size with two conditions (key size > 4 && key size % 4 == 0). So, the first valid key size is 8 bits. Actually, this constraint on key size is important for faster primes generation.

RSA encryption analysis

I- In this part we create keys with n size {8, 16, 32, 64, 128} and calculate the time of {key generation, encryption, decryption} for same message "attack".
II- consolation: For key generation its very obvious that time increase exponentially with size of key. About encryption and decryption, time increase linear with key size.



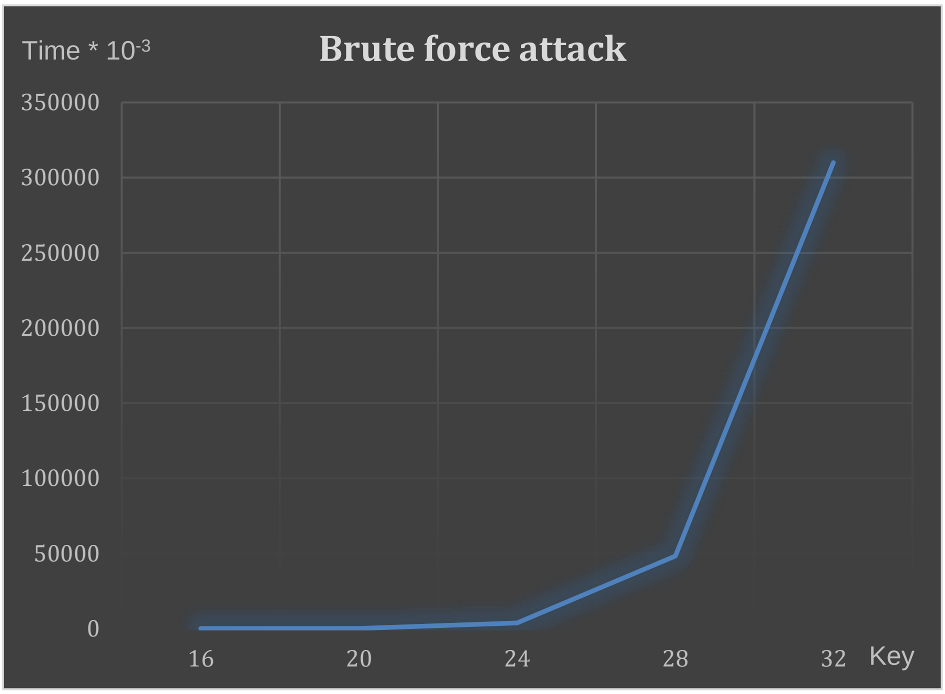
Graph-1: encryptoin and decrypton time



Graph-2: key generaton time

Brute force attack

I- In this part we create keys with n size {8, 16, 32, 64, 128} and try to guess private key. This operation stops when the (original, decrypted) message be the same. The used message was "attack"
II- consolation: it's very obvious that time increase exponentially with size of key. We stopped at key size (32 bits), we try to beak (36 bits) key but it takes more than one hour without find the private key.



Graph-3: brute force time

Chosen chipper text attack

The public key systems are usually susceptible to the chosen-ciphertext attack, as from the beginning the encryption system follows a standard encryption schema in the RSA cryptosystem like a checksum for the message, inject standard numbers, and padding for the message. so on the other hand on the receiver side, it holds all this information to check if the message is correct or not and gives a flag, and this happens if the server is vulnerable. And these attack to be done considers that the attacker has an access to the decryption system and see the flags or see the wrong recording into the databases, so simply I choose a ciphertext I need to decrypt and multiply it by random number r which is I knew, then send it to the server to decrypt it and it will give it back as the flag is raised and the message is understandable, finally, the attacker knows and get the decrypted message back, and simply divide it by r the random number he's already chosen and get the message.

$$C = m^e mod\, n$$
$$C' = C * r^e mod\, n$$
$$(C')^d = (C * r^e) \quad ^d\, mod\, n$$
$$(C')^d = m^{ed} * r^{ed} mod\, n$$
$$m * r = m^1 * r^1 mod\, n$$