Security Project Report

Team :

Omar Yasser 34-14741

Ebrahiem Hassan 34-12851

Mohamed Ahmed 34-14375

Hatem Ramadan 34-14629

Mina Amir 34-15582

– Summary of your project, motivation and how you implemented its features.
Our project is a chat application similar to the Whatsapp application, It supports private chat between users, it support ,making group chat between different users and also you can send messages in these chats a text messages. You can also send your current location as a message. In our app the security was a main concern. We decided to choose a whatsapp application because nowadays, everybody is using these apps a lot everyday, so it is important to understand how these applications is implemented and implement them ourselves. We used an available open source project (the link for it is in the references) that supports just the signing up, logging in and private chats between the users without any other features. This project was implemented using Node.js and Angular.js. We understood it and its flow, and based on this understanding we added the extra features we wanted which are group chats, sending location, sending files and most importantly the encryption as security is our main concern.

– Your design choices of cryptographic algorithms and protocols for each functionality. Explain why you used a certain methodology in particular and how you made such security techniques fit your project requirements.
For hashing the password, we used the passport module available for Node.js. We wanted to hash the password at the server side so to add more security we added some salt to the password then apply the hashing function before storing the password in the database.
For the Authentication: during the login, the hashed password is get from the server and then we check whether this is equal to the hash of the password the user entered and if the password is correct, the user is given a token which he can use to send other messages and requests afterwards.
For the Access Control, every time anyone tries to access a chat he does not have access to it, or tries to send a message to some chat he is not authorized to send messages to, the user is blocked and is redirected to the global chat room. This is done

by checking whether this person is involved in this chat before giving him access to do what he wants.
For the Tokens: after the user is authenticated he is given a token that he will send in all upcoming requests. We used JWT library that is called (jsonwebtoken) to create the tokens.
For the encryption of the messages, we used the AES encryption, the messages are encrypted before they are being sent to the server, and are decrypted at the receiver to get the original message.

– A detailed comparison between the security methods, protocols and algorithms you researched and why you think your approach is best.

While searching we found the DES but we know that DES have smaller key size which make it more easy to be hacked and the key can be known easily, and it has been already broken before. The AES has an advantage of larger chunks of code up to 265 bits per one chunk where DES chunk was just 64 bits. In addition, AES is faster than the DES
We also found asymmetric key algorithms but after some research, we found that the AES is faster than the asymmetric key algorithms (like RSA) especially that our application needs real time messaging so speed here is a big concern.

– Attack scenarios that you thought about (and researched) and how your system is secure against them. Examples and screenshots would be much appreciated.

1- If someone hacked the database, this will expose the passwords of all users or even If someone internally at the company has the access to query the passwords table, he can know these passwords and use them to access the users’ accounts (even in another websites as they may use the same password in different websites) so, we hashed the passwords at the server side before saving them in the database. For this we used the passport module.

2- If someone hacked the database (or even if someone who has access the database can access these tables) , this will expose all the messages in the database between the user chats, so these messages should be encrypted even inside the database.

3- If someone knows the link of the chat between two other people or a group chat he is not in, he may use it to see the contents of the messages and send new messages, we handled this by checking that the user is involved in this chat before responding by the chat messages or posting a new message.

4- If someone tried to make an XSS attack by writing some javascript code it may result in some bad behaviour, so we used angular which uses sanitization which escapes untrusted values.

5- If somebody tried to make an SQL Injection attack, he will not be able to do that as we use mongo (NO SQL) database.

– Explanation of each computer security technique you used along with its advantages/disadvantages trade-offs.

<u>AES algorithm:</u>

Advantages:

- It uses higher length key sizes such as 128, 192 and 256 bits for encryption.
- The longer the key the more robust against hacking.
- Faster than asymmetric key algorithms.
- It is most commonly used security protocol for many applications such as wireless communications and financial transactions.

- For 128 bits, about 2^128 attempts are needed to break

Disadvantages:

- It uses too simple algebraic structure.
- Every block is always encrypted in the same way.

Passport:

Advantages:

- Very simple to use.
- Easy for integration with frameworks like express.

– Explanation of libraries/frameworks you used. In particular, what security features you utilized and how did the library offer you such features?

Angular.js: We use angular as a front-end framework.

Node.js: Our backend is base on javascript and its (express) module.

Socket.io: a JavaScript library for realtime web applications.

Mongodb : Our database is a NOSQL one.

Passport Module (JavaScript) : we used it in hashing the passwords

Jsonwebtoken : a JWT library that generates JWT tokens for the user after logging in.

Agm : a library to view the locations sent in messages on Google Maps.

– A comprehensive list of all references, links and other aids that helped you during the project implementation, security assessment and research.

https://angular.io/

https://socket.io/

https://nodejs.org/

https://www.mongodb.com/

https://mongoosejs.com/

https://www.npmjs.com/package/@agm/core

http://www.passportjs.org/

https://www.youtube.com/watch?v=lApggVS0icc

https://www.npmjs.com/package/crypto-js