Report on 12 week internship/training undertaken at EPFL | École polytechnique fédérale de Lausanne

Proposed Scheme:-

The project aimed to develop and implement schemes so that the anonymous social network could be turned into really anonymous social network! The scheme that we implemented and successfully tested for feasibility was for trusted users where users trust each other. Here the term anonymous social network refers to social networks where users have the privilege to post anonymously (for example- whisper). The term ‘really anonymously’ refers to the fact that the anonymity is ensured at client side as well as on the server side. Under this scheme, the following set of measures were undertaken to ensure anonymity at server side-

1. Users in a particular group (particularly the set of users which trust each other) decide on a particular group key.
2. The server generates a pair of public and private keys.
3. The public key is available to the client while the private key is maintained at the server side.
4. The users encrypt their message first using the public key and then using group key. The server receives this encrypted message. Instead of broadcasting the message, it waits till it collects n number of messages.
5. The server then broadcasts these messages to all the clients. They (clients) decrypt the message using the group key and then randomly shuffle these messages.
6. The server waits till it receives a particular set of messages (Here it takes into account the first message set received). Then it decrypts these messages using the private key.
7. Then the messages (as usual) are broadcasted to the group.

This ensured anonymity to users both on client as well as server side.

Implementation:-

This scheme was implemented as a broadcast server-client application in java. The server application could be started on a particular system. Now different clients could connect to it once they knew the name of the server running (all
system connected via LAN). For this purpose, NIO library in java were used. RSA Encryption was used for public and private key cryptography and for group key encryption PBEWithMD5AndDES was used. Since the generation of public and private key takes place on server side, access to the public key has to be provided to the clients. In this case, it was done using a static constant embedded in the code. The message from the client was encrypted using a padded cipher using the group key first and then encrypted using the public key. For encoding purpose, ISO-8859-1 was used since it facilities one to one transformation from bytes to string and vice versa without any loss of data.

The message encrypted was sent to the server. The server waited till it collected certain number of messages (in this case 3 for testing purpose).

After the due numbers of messages were received, the messages were sent to all the clients in the group. The clients received the messages and shuffled them and then the messages were sent to the server. The server takes the first set of received messages and then decrypts using the private key and then they were sent back to all the clients. Since the messages were shuffled, it is not possible on the server side to know which message came from whom.

Feasibility:-

Encouraging results were received. The model worked perfectly for the given scheme. In order to achieve better results in terms of accuracy and to avoid information loss, more efficient server could be used. Here the term efficient refers to the speed of computation.

Real-time implementation:-

We collaborated with a research group at EPFL which had prepared a solution for anonymous social networking. After a lot of deliberation, it was decided to move on to an open source solution. Spika-app is the open source messenger that was finally settled on.

Next phase of the project:-

After testing the feasibility of the scheme and choosing a suitable open source solution for the same, the next phase would be implementation of the above
scheme over the existing framework. Spika-app, (http://spikaapp.com/), an open source messenger app by Clover studio.

Changes made to the existing framework of Spika-app:

1. A suitable public and private key is embedded on the server source code which could be used for encrypting and decrypting the messages.
2. In Sign in Activity.java, methods for generation of public and private key are added, when the user first signs in, if there isn’t a public and private key present, a pair of public and private keys are generated.
3. Whenever a group is created, if there isn’t a group key for that particular group, generation of group key takes place (in creategroupactivity.java).
4. As soon as particular user subscribers to a group, the public key as a message is sent to the group admin.