Sending e-mail securely using DES, key transmision by means elicptic DH and signature El-Gamal

Group 22 - Aviv Eitam, Anton Grudsky, Ori Burshtein

The process of the project study

We have started the process by going through the lectures of the relevant subjects, to remember and understand the general concept of each algorithm and to familiarize with the specific details of each one. Through this process, we created a flow of the email encryption in order to visualize and have a better understanding of the email sending action. After completing this step, we realized the right sequence of the methodologies in our project and we could proceed to the implementation stage. Our project is composed with three main algorithms that synergically encrypting the message. As a first step, we decided to implement our project using Python. Since we are not well experienced with the language, we needed to explore more thoroughly the Python tools before we were able use them. During the exploration, we came across several unfamiliar notions:

1. **Elliptic curve** – for better security, the key values of DH should be very large. Therefore, we could not create our own elliptic curve. Hence, we used an existing Python library called tinyec that contains large scaled elliptic curves we could use.

2. **Hash function** – as part of the sequence of the El-Gamal digital signature algorithm, we needed to hash the message before we sign it, so we used Python’s hash library and applied one of the existing functions on the message.

3. **User communication** – we applied user communication using socket which resembles the email sending task. The communication using socket is asynchronous, meaning that if a task is not fully executed, it may jeopardize the synchronous flow of Python and block the message requests. To overcome this threat, we inserted a short sleep between consecutive transmissions to prevent data corruption.

During the implementation, we came across a hurdle in the shape of data type conversions. For example, the message is received as a string and then, in the DES encryption process, it’s converted to bits and then to bytes for the message transmission, and back to bits in the decryption process and finally back to string. In addition, sending the public key using DH (which formatted as a point since we are using an elliptic curve) has to be performed in two steps, each for every point’s coordinate, and on the receiver’s side they are combined back to a point.

The experience of the implementation helped us a lot with understanding how the message transmission actually works and the different algorithms it contains. Also, at that stage we saw the similarities and differences between the various tasks and how the integration between them occurs.

The project’s flow

The main stages in the project are:

1. Elliptic curve DH key generation – after a connection is established, we proceed to generate ECDH keys. Each side, sender and receiver, create their own pair of keys (private and public), where the public key is the product of the private key and the base point of the same elliptic curve.
2. El-Gamal key signing – in the next step, after we generate the signature variables, we can proceed to signing each side’s public key before the key exchange. In this process we hash the key, calculate the signature, and send it to the recipient.
3. Signature verification and calculating the shared key – Each side receives the sender’s signature variables and the signed public key and verifies it. If the verification was successful, they proceed to calculate the product of their private key and the sender’s public key to create the shared key that will be used in the DES encryption.
4. DES encryption with CBC- at first, we generate a random initial vector (IV) to be used in the CBC. Next, the message is encrypted with DES algorithm using the shared key and the IV if needed (for long messages encryption using CBC mode). The last step before sending the email is to sign the message with El-Gamal.
5. Decryption – after receiving the encoded message, we verify the sender’s authenticity and integrity. And finally, we decrypt it with the shared key.

The obtained results

Our project contains several layers of security: we generate the key using ECDH, then we sign it with El-Gamal signature to verify the authenticity and maintain integrity, and finally, we use DES algorithm to encrypt the message. Our model resulted in secured Email transmission along with integrity authentication and receiving of complete message in a reasonable period of time.

Conclusions

The combination of the three algorithms provides an efficient yet secured communication. It allows its users to send or receive messages whom their authenticity and originality is promised. Although we believe this project provide a secure email sending, we think that there are some improvements that can be added such as encrypting the variables of the El-Gamal signature and the public keys of ECDH. Also, there is a room for minimizing duration of the communication procedure (find an alternative way to the short sleeps between consecutive transmissions).