Encryption Algorithms for Secure Communication over the Internet

BSP S3

BICS winter semester 2018/2019

Summary of report in English

BICS student: Desislava MARINOVA
PATs: PhD Student Asya MITSEVA, Prof. Dr. Thomas ENGEL
Faculty of Science, Technology and Communication
University of Luxembourg

Email:
desislava.marinova.002@student.uni.lu
asya.mitseva@uni.lu
thomas.engel@uni.lu

The main domains of this Bachelor semester projects are Cryptography and Network security. Nowadays, we depend on online connectivity to execute various errands among which online banking, accessing medical and governmental records as well as transfer/storage of sensitive information. Hence, it is of vital necessity to protect our data against lurking intruders and potential attacks.

We build a Python chat application that encrypts user's messages before transmitting them over the Internet. Basic programming skills in Python are needed along with clear comprehension of basic cryptographic primitives. The encryption algorithms reviewed in our project are Caesar cipher, ROT13, One-time pad (OTP), Vigenere cipher, Rivest cipher 4 (RC4), Data Encryption Standard (DES), 3DES, and Advanced Encryption Standard (AES). Our chat application makes use of four of them: Caesar cipher, ROT13, DES and AES. The latter two have been implemented using the support of PyCryptodome library.

In cryptography, the original message is known as *plaintext* or *cleartext*. The process of concealing message's content is *encryption*. Thus, an encrypted message is *ciphertext*. The process of turning ciphertext back into plaintext is *decryption*. A *cipher* is an algorithm for executing a series of steps during encryption or decryption operations. A *cryptosystem* or *cipher system* is an implementation of cryptographic techniques to provide information security services.

There are three types of cryptography: symmetric key, asymmetric key and hash functions. For this project, we focus mainly on symmetric-key cryptography, also called conventional. It comprises a single key for both encryption and decryption operations. To use symmetric encryption securely one needs a strong encryption algorithm as well as a secret key known only to sender/receiver. Symmetric-key systems are simpler and faster, however, their main disadvantage is key management, since both parties must exchange the key in a secure way and preserve its security. Otherwise, the key could be intercepted and compromised by a third party leading to decryption of the secret information. Hence, asymmetric or public-key cryptography came into existence. It requires two separate keys, one public and one private, for encryption and decryption, respectively. Notwithstanding, many modern ciphers are hybrid cryptographic systems since they use both symmetric-key and asymmetric-key encryption.

The types of operations used in classical symmetric cryptography are substitution and transposition along with two ways in which the plaintext is processed: block and stream.

Substitution ciphers replace letters of plaintext with other letters, numbers of symbols. The plaintext is viewed as a sequence of bits where substitution operation replaces plaintext bit patters with ciphertext bit patterns. The earliest known substitution cipher is Caesar cipher which shifts a letter 3 places forward. ROT13 is a special case of Caesar cipher. It encodes a letter 13 shifts left or right, making encryption and decryption identical. Hence, it is its own inverse and is an example of a weak cipher. The One-time pad (OTP) XORs a random sequence of 0's and 1's to plaintext and uses a random key, which is never repeated and is of the same length as the message. It completely obscures the original message and for this is considered unbreakable. Transposition ciphers rearrange the letter order without altering the actual letters used. Nevertheless, their drawback is that the ciphertext has the same frequency distribution as the plaintext. A representative is the rail fence cipher that writes out message letters diagonally over a number of rows.

Block ciphers, the most widely used cryptographic algorithms for encrypting communication over the Internet, process messages into entire blocks at a time, each of which is then encrypted/decrypted. Block ciphers work similarly as substation ciphers but on 64-bits (e.g., DES) or more characters (e.g., AES' 128-bit block length). Stream ciphers, on the other hand, process messages a bit or byte at a time, individually, when encrypting/decrypting. However, they sometimes require fewer resources for implementation than Block ciphers which makes them suitable for constrained environments (e.g. cell phones).

Data Encryption Standard (DES) is a modern block cipher, standardized in 1979. It has a 64 – bit key comprised of 8-bit parity, leaving a length of 56-bits. DES has a 64-bit input and produces a 64-bit output. The 56-bit key generates keys throughout the sixteen 48-bits rounds. Permutation "frames" the algorithm as it is used once before the rounds and one final time before the 64-bit output. DES, along with other block ciphers, supports several modes of operations (ECB, CBC, OFB, CFB, CTR) used when we want to encrypt larger than 16-bytes plaintext. Nevertheless, DES is nowadays considered insecure but acceptable for non-sensitive applications. It has been replaced by Triple DES (3DES) which encrypts data three times in contrast to DES. In particular, the one 56-bit key becomes three individual keys, rendering a 168-bit key. Nowadays, one of the most widely used ciphers is the Advanced Encryption Standard. AES comprises three block ciphers: AES-128, AES-192 and AES-256. Each cipher has a 128-bit block length, with key sizes of 128, 192 and 256 bits, respectively.

We create a client/server architecture for our chat application. We have chosen TCP networking as a communication endpoint. Once the server and client establish a connection, the client can send encrypted messages to the server. The server decrypts and sends them back in their original format. Thus, the client.py script stores all encryption functions, whereas the server.py is in possession of all decryption operations.