CPA-secure encryption scheme

Sarthak Mahajan, 2018111005

March 7, 2022

1 CPA Attacks:

CPA(Chosen Plain Text attack) is an attack in which the adversary has access to the encryption server and can send chosen plain texts to it and get their encryption.

2 Definition of CPA Security:

Assume that the Adversary has access to the encryption server, and it sends two messages m_0 and m_1 (of same length) to the encryption algorithm and the encryption algorithm sends it an encryption of message m_b back(chosen randomly), where b = 0 or 1. Let b_{guess} be adversary's guess of which message is sent. Then the Cryptosystem is CPA-secure if for all PPT(probabilistic polynomial time) adversaries A:

$$P\left[b_{\text{guess}} = b\right] <= 1/2 + \text{ negl}(\ n\ )$$

3 CPA security:

4 Idea:

We apply the pseudorandom function on some random number, rather than the message, in this way even if the adversary has access to the encryption server(CPA attack), he won't be able to decrypt as he doesn't have the random number that the pseudorandom function is being applied on. Since the PRF applied on this random number is XOR'ed with the message, and the adversary can't access this random number and the encryption of the message varies depending on this number(non-deterministic), the adversary won't be able to decrypt.

4.1 Provable CPA security using PRF:

Let F be a pseudorandom function. Define a private-key encryption scheme for messages of length n as follows:

- **Gen:** on input 1^n, choose $k \leftarrow \{0,1\}^n$ uniformly at random and output it as the key.
- **Enc:** on input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^n$, choose $r \leftarrow \{0,1\}^n$ uniformly at random and output the ciphertext

$$c := \langle r, F_k(r) \oplus' m \rangle .$$

- **Dec:** on input a key $k \in \{0,1\}^n$ and a ciphertext $c = \langle r, s \rangle$, output the plaintext message

$$m := F_k(r) \oplus s$$

5 Modes of Operations for CPA Security

The above construction was for when length of the key k is equal to the length of the message. If the length of the message is greater than k's length, then, there are multiple secure modes of operation in which PRF's can be used for CPA security like:

- CBC(Cipher Block Chaining)
- OFB(Output Feed back Mode)
- RCM(Randomized Counter Mode)

The mode of operation used in this code is OFB

5.1 OFB:

5.1.1 Encryption:

Let the length of the key $k = n$, assume that the length of the message $m = n_2$ and $n_2 >= n$, We start breaking m into chunks of size n, starting from the MSB, in case the last chunk cannot be of size n, we pad it with zeros on the left to make it of size n. Let m_i donate the i^{th} message chunk. Let IV be an initialization vector, which is just a random n bit number. Let each "block" of encryption in the OFB scheme denote a PRF $F_k(.)$, with key k. The input to the i^{th} block would be:

$$in_i = (F_k)^{i-1}(IV)$$

, where $(F_k)^i(IV)$ denotes the function $F_k(.)$ applied on itself i times with IV being the innermost/first input to the first $F_k(.)$ and output of i^{th} block is:

$$c_i = (F_k)^i(IV) \oplus m_i$$

The final cipher text would be a concatenation of IV with all the c_i's with IV being leftmost, all the rest c_i's follow in order.

5.1.2 Decryption:

We break the ciphertext into chunks of size n starting from the left, The IV for the OFB in Decryption mode would be $c_1, the input to i^{th}$ block is:

$$in_i = (F_k)^{i-1}(IV)$$

, where $(F_k)^i(IV)$ denotes the function $F_k(.)$ applied on itself i times with IV being the innermost/first input to the first $F_k(.)$ and output of i^{th} block is:

$$m_i = (F_k)^i(IV) \oplus c_{i+1}$$

The final message is the concatenation of all m_i's, the earlier padded zeros during encryption(if at all) on the rightmost side taken out.

5.1.3 Explanation:

OFB is symmetric in nature, i.e, each block for encryption and decryption is the same function, it works because i^{th} cipher chunk during encryption and i^{th} message chunk during decryption are made in a symmetric manner, by XORing with the same quantity: $(F_k)^i(IV)$, since XOR of two same quantities is 0, we retain the original message chunk after XORing twice with $(F_k)^i(IV)$, during both encryption and decryption