CAS CS 357

In-class Note 9

1. Background on XOR
    a. a XOR 0 = a
    b. a XOR 1 = not a
    c. a XOR a = 0
    d. a XOR b→ addition of a and b, and mod it by 2
    e. XOR associative → (a XOR b) XOR c == a XOR (b XOR c)
2. Some intuition - encrypting a single plaintext bit via xor a randomly generated key bit

    Plaintext bit

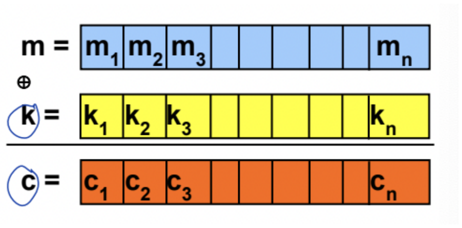
    m = 0 is known   m = 1 is known

    Key bit:

    m XOR k → 0 or 1 with the probability of one half (both cases where m=0 or m=1)

    - If m = 0, m XOR k is 0 with a probability of ½ and 1 with a probability of ½
    - If m = 1, m XOR k is 0 with a probability of ½ and 1 with a probability of ½

    a. If the adversary knows m and m XOR k (ciphertext = c), the adversary can guess the key using XOR.
    b. If we do m XOR c → m XOR m XOR k → k (we get back the key)

   c. If the adversary just knows the ciphertext without m, it makes it difficult to guess the key
   d. Similarly, if the adversary just knows the plaintext (m) without the ciphertext, it makes it difficult to guess the key
3. Kerckhoff's principle
   a. How do we know we are using XOR with the key?
   b. Kerckhoff's principle: assume that adversary knows everything about the system except the secret key (knows what scheme is being used, knows Enc() and Dec())
4. One-time pad
   a. 

$m =$ m_1 m_2 m_3 m_n
$\oplus$
$k =$ k_1 k_2 k_3 k_n
$c =$ c_1 c_2 c_3 c_n

   b. Key length = message length = n (in a typical one-time pad)
   c. The Key is a random n-bit string
   d. With XOR, Enc(m) = c = m XOR k and Dec(c) = c XOR k
   e. Correctness: Dec(Enc(m)) = m XOR k XOR k = m
   f. Roughly speaking, the one-time pad is secure because the XOR of a random bit with a known bit produces a bit that is random
   g. It makes sense only if you use the key once and change the key so that it is not reused, since if we reuse the same key, the key may be found by the adversary or the adversary can learn few information regarding the key

h. One time pad is the only scheme that we know that satisfies perfect secrecy → we do not make any computational assumptions on the adversary (and the probability of getting the message right is ½ always)

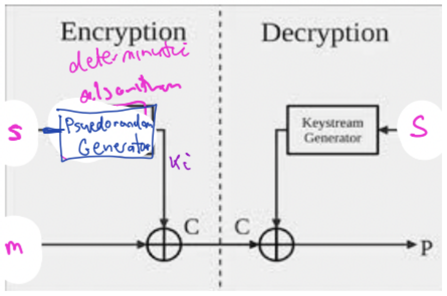
i. However, one-time pad is not practical in real life because of few conditions:

   - Key must be the same size as the message being sent (m must equal k in size)
   - Keys must be truly random
   - Keys are not reused

5. Stream Ciphers

   a. Problem with one-time pad: key length must equal plaintext length

   b. Stream cipher uses a cryptographic key-generation function to produce a pseudorandom keystream so that this keystream is then XOR'd with the plaintext

   c. 

   d. When a person sends a message that is bigger than the size of the key, we can create a pseudo-random cipher digit stream, known as keystream.

   e. For example, if the key is 4 bits and the message is 10 bits, we cannot use the one-time pad. However, we can place a random seed value to the key so that the pseudo-random keystream has the size of the plaintext.

6. Block ciphers

a. Block cipher is a deterministic algorithm operating on fixed-length groups of bits, called blocks of pre-determined size
b. For example, if you are encrypting a plain text with bits: 11011001, we have to break these digits into say 4 digits (these digits vary), producing 1101 1001. These blocks are then entered into the key block to produce the ciphertext
c. If there isn't enough data to form a complete block (having 10 digits when the block size is 4), you can add padding (set number of 0s) to the end of the block to ensure that the block is that fixed size