- **Problems**
  - **CIA Triad**
    - Confidentiality - information or resources can only be accessed and viewed by authorized parties
    - Integrity - information or resources can be modified only by authorized parties
    - Availability - information or resources accessible to authorized parties at appropriate times
  - **Security Model -** combination of a trust and threat model that addresses the set of perceived risks
- **Tools**
  - **Secret Key Cryptography -**
    - symmetric keys, a single key is used for both encryption and decryption algorithms.
    - Management of keys determines who has access to encrypted data
  - **Public Key Cryptography**
  - **Hash Function**
- **Application**
  - **CIA Triad**
    - **Confidentiality**
    - **Integrity**
    - **Availability**
    - **Examples**
  - **Security Model**
    - **Thread model -** a collection of threats (set of circumstances that has the potential to cause loss or harm to a system) that is deemed important for a particular system.
    - **Trust model -** describes for a particular environment who is trusted (degree to which an entity is expected to behave) to do what
  - **Cryptosystem -** 5 tuple
    - **E -** encryption algorithm - algorithm used to make messages unreadable by all but the intended receivers - E(plaintext, key) = ciphertext
    - **D -** decryption algorithm - the reverse of the encryption algorithm - D(ciphertext, key) = plaintext
    - **M -** set of plaintext - message in its original form
    - **K -** set of keys - input to a cryptographic algorithm
    - **C -** set of ciphertext - encrypted message
  - **Secret Key Cryptography**
    - **Definition**
      - An encryption/decryption key is known only to the parties that exchange secret messages. This shared key would be used to encrypt and decrypt messages.
    - **Transposition cipher**

        - Permute the symbols in plaintext to produce ciphertext
- **Substitution cipher**
    - Substitutes one symbol for another
- **Caesar cipher**
    - Each letter in the alphabet is replaced with the letter k slots to the right. Index the letters in the alphabet from 0 to 25. Each ciphertext letter c = E(m, k) = (m+k) mod 26, where k takes the value in [1, 25].
- **One-time pad**
    - Using a random bit string (a pad) as the key, which is the same length as the plaintext. Encrypt the plaintext by XORing it with the random bit string. Decrypt the ciphertext by XORing it with the random bit string. C = E(m,k) = m XOR k; m = D(c,k) = c XOR k
    - Pad is used to encrypt and decrypt only one message, then it is discarded. Perfectly secure, assume the value of each bit in k is equally likely, then you have no info to work with.
- **DES**
    - Block cipher, map a 64-bit input block to a 64-bit output block by using a 64-bit key (56 bits + 8 parity bits)
    - Final permutation is the inverse of the initial permutation
    - Per-round key generation: 16 48-bit per-round keys. Per- round encryption
        - Per round key generation: left rotation: rounds 1, 2, 9, 16 circular shift of 2 bits to the left
    - Mangler Function
        - Expand R_n from 23-bit to 48-bit. Break R_n into 8 4-bit chunks. Expand each chunk to 6 bits by concatenating adjacent bits to it.
        - Break 48-bit K_i into 8 6-bit chunks
        - XOR the 8 chunks of R_n and K_i and feed the results into 8 substitution boxes to generate 32-bit output
        - Final permutation
        - Substitution box = S box. Used to obscure the relationship between the plaintext and the ciphertext. Are carefully chosen to resist cryptanalysis.
- **ECB**
    - Encryption greater than 64 bits. Broken into 64 bits (padding last block if needed). Each block independently encrypted using DES with the same key.
    - Error in encryption of one block does not affect other blocks.
    - Can create recognizable pattern. Ciphertext blocks can easily be rearranged.

        - To fix, XOR each block with a 64-bit random number before encrypting it with DES. (Drawback, volume of data transmitted doubles)
    - **CBC**
        - Uses the previous ciphertext block as random number and XOR it with the next plaintext block. Select a random number (IV) that is XORed with the first plaintext block.
        - Identical plaintext blocks will not cause repeats in the ciphertext.
        - IV needs to be shared between sender and receiver.
        - Error in one ciphertext block will affect the decryption of the next ciphertext block.
    - **Brute-force attack**
        - Attacker tries every possible key on a ciphertext until an intelligible translation into plaintext is obtained.
        - Worst case: attacker tries all possible keys
        - Average case: half of all possible keys must be tried
- **Hash function**
    - Compression of data into a hash value. Algorithms are generally useful in systems (speed/space optimization)
    - Birthday paradox - the probability that two or more people in a group of 23 share the same birthday is larger than 50%. Given function *f* with *n* possible outputs that are uniformly distributed on *k* inputs $\{m_1, m_2, \ldots, m_k\}$ if $k > 1.2n^{1/2}$, $Pr[f(m_i)] > 0.5$ for some *i* and *j, i* != *j*
    - **One-way**
        - Computationally hard to invert h()
    - **collision resistant**
        - Computationally hard to find two messages x_1 and x_2 such that $h(x_1) = h(x_2)$
    - **Message authentication code**
        - Given hash function *h()*, key *k*, and message *m*, MAC(k, m) = h(m|k)
        - Send message and authentication code to receiver
        - The receiver computes *h(m|k)* using the received message and compares the result with the received *MAC(k,m)*.
- **Public key cryptography**
    - **Definition**
        - Each individual has two keys: a public key k^+ known to everyone, and a private key k^- kept secret to the owner.
        - Everyone can use the receiver's public key to encrypt a message, and only the receiver can use his private key to decrypt it
    - **RSA**

- Key length is variable, plaintext block must be smaller than the key length, ciphertext block is the same as the key length.
- Key generation:
  - Pick two large primes *p* and *q* (512 bits)
  - N = pq
  - Choose e such that it is relatively prime to f(n) = (p-1)(q-1)
  - Find d that is the multiplicative inverse of e mod f(n)

■ **Digital Signatures**
- Create association between private key and document.
- Provide data integrity and non-repudiation
- $E(m, k^+) = c$ and $D(c, k^+) = m$

■ **Diffie-Hellman Protocol**
- Neither encrypts or gives a digital signature. Used to negotiate a shared secret key over an insecure media.
- Is computationally hard to calculate discrete logarithm
  - Key Agreement:
    - Two participants A and B agree on a prime number p and a base g(<p)
    - A picks a private Value SA(<p - 1), B picks a private value SB(<p - 1)
    - A generates $T_A = g^{SA}$ mod p, B generates $T_B = g^{SB}$ mod p. A and B exchange T_A and T_B.
    - A computes T_B^{SA} mod p, B computes T_A^{SB} mod p. The shared secret key $k = T_B^{SA}$ mod p = T_a^{SB} mod p

■ **Man - in - the - middle attack**
- Don't know anything about who you have exchanged keys with
- A and B think they are talking directly to each other, but the attacker is actually performing two separate exchanges
  - Solution: authentication

○ **Data Confidentiality**
■ **SKC solution -** DES, CBC
■ **PKC solution -** RSA

○ **Authentication**
■ **Identities -** give you access, which is largely determined by context
■ **Credentials -** evidence used to prove identity
■ **Pros and cons of authentication mechanisms -** use random challenges to resist eavesdropping. Cryptographic functions can effectively resist online and offline guessing attacks.
■ **Reflection attack**
- Principle 1
  - Do not have sender and receiver do the exact same thing.

    - Different keys: have the key used to authenticate the sender be different from the key used to authenticate the receiver.
    - Different challenges: the challenge from the sender is different from the challenge from the receiver
- Principle 2
    - The initiator should be the first to prove its identity