Requirements

- Two requirements for secure use of symmetric encryption:
  - a strong encryption algorithm
  - a secret key known only to sender / receiver

    $Y = E_K(X)$ plaintext

    $X = D_K(Y)$
- assume encryption algorithm is known
- the security of symmetric encryption depends on the secrecy of the key
- implies a secure channel to distribute key

A strong encryption algorithm

Data sets

(x_1, y_1)

(x_2, y_2)

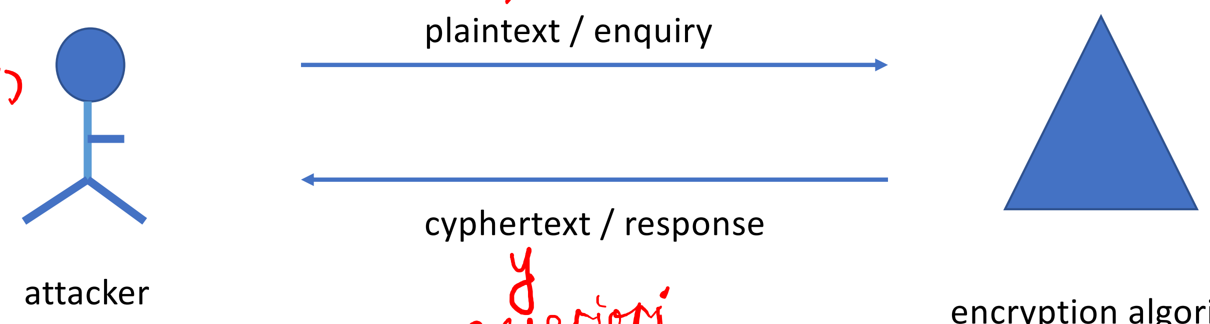
$\vdots$

(x_n, y_n)

ML $\longrightarrow k$

MAP

(x, y)



x (plaintext) · y (cyphertext)

2^k ?

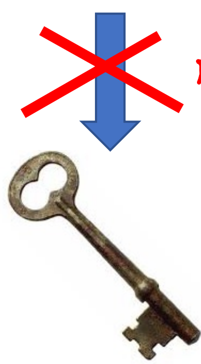
plaintext

ciphertext

ciphtext

$K \pm \Delta K$ → confident interval

encryption algorithm → public



posteriori

$$\max P(K \mid x, y, F) = \frac{P(x, y, F \mid K) \cdot P(K)}{P(x, y, F)} = \max \frac{P(x, y, F \mid K) \cdot P(K)}{\sum_{i=1}^{n} P(x, y, F \mid K_i) \cdot P(K_i)}$$

MAP

MAximum a Posteriori Estimation

$$= \max P(x, y, F \mid K) \cdot P(K)$$

Secure Encryption Scheme

- **Unconditional security**
  - no matter how much computer power is available, the cipher cannot be broken since the ciphertext provides insufficient information to uniquely determine the corresponding plaintext without K
- **Computational security**
  - the cost of breaking the cipher exceeds the value of the encrypted information;
  - or the time required to break the cipher exceeds the useful lifetime of the information

trade off resource, time, security, ~~evaluation~~ value

$k \to f(k, m)$

$k+\Delta\delta \to f(k+\Delta\delta, m)$, differential privacy

$k_1 \to y_1$,

$k_2 \to y_2$, $|y_2 - y_1|$ no pattern very

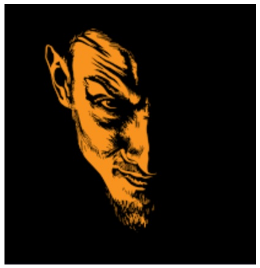
$k+\Delta\delta$

Desired characteristics

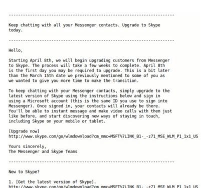
- Cipher needs to completely obscure statistical properties of original message
  → capacity channel,
- more practically Shannon suggested combining elements to obtain:
  - Confusion – how does changing a bit of the key affect the ciphertext?
  - Diffusion – how does changing one bit of the plaintext affect the ciphertext?

confusion

ciphertext ~~ciphertext~~



plaintext



diffusion

Ways to achieve

$x \longrightarrow y$

- Symmetric Encryption: → bit
  - substitution / transposition / hybrid
- Asymmetric Encryption: → NP problems
  - Mathematical hardness - problems that are efficient to compute in one direction, but inefficient to reverse by the attacker
    - Examples: Modular arithmetic, factoring, discrete logarithm problem, Elliptic Logs over Elliptic Curves

numbers

Big number

RSA → certificate

→ & PK

Two basic types

64, 128 bit

Cipher ← Key

64, 128 bit

AES 128, 192 and 256 bits

- Block Ciphers
  - Typically 64, 128 bit blocks
  - A k-bit plaintext block maps to a k-bit ciphertext block
  - Usually employ Feistel structure
- Stream Ciphers
  - A key is used to generate a stream of pseudo-random bits – key stream
  - Just XOR plaintext bits with the key stream for encryption
  - For decryption generate the key stream and XOR with the ciphertext!