IERG 4130 – 2026 Fall

香港中文大學
CUHK

Block Cipher

Yajin ZHOU

https://yajin.org

Credits: the slides are from CS161 of UC Berkeley,

Permutation

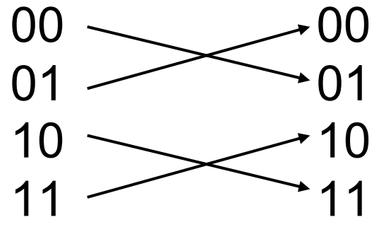


- In mathematics, a **permutation** is a bijective function of a finite set onto itself.
  - demo

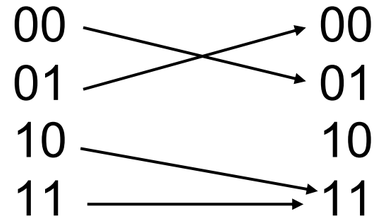
```
000   ----------------------------------------->   101
001   ----------------------------------------->   110
010   ----------------------------------------->   000
011   ----------------------------------------->   011
100   ----------------------------------------->   111
101   ----------------------------------------->   001
110   ----------------------------------------->   010
111   ----------------------------------------->   100
```

Block Ciphers: Correctness

- **$E_K(M)$ must be a permutation (bijective function) on n-bit strings**
  - Each input must correspond to exactly one unique output
- **Intuition**
  - Suppose $E_K(M)$ is not bijective
  - Then two inputs might correspond to the same output: $E(K, x_1) = E(K, x_2) = y$
  - Given ciphertext y, you can't uniquely decode. $D(K, y) = x_1$? $D(K, y) = x_2$?



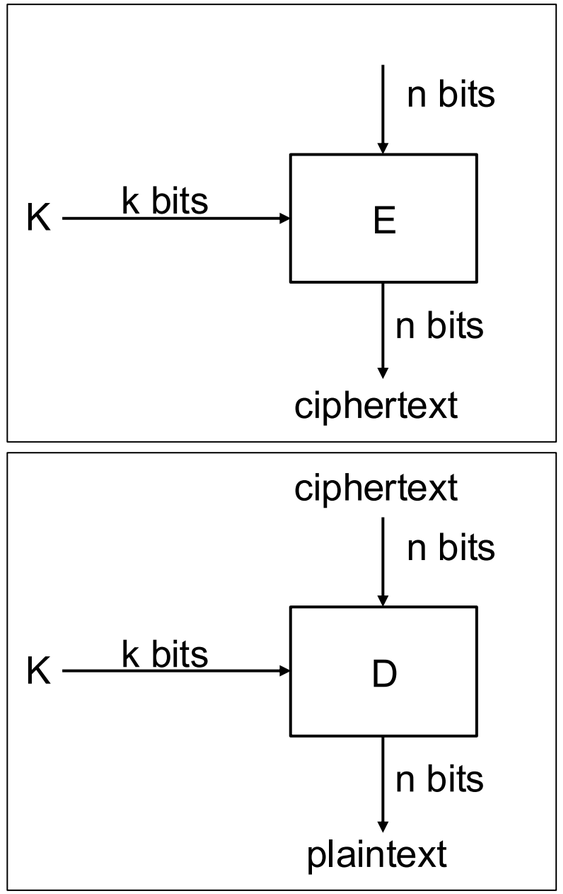
Bijective: Each input maps to exactly one unique output



Not bijective: Two inputs encode to the same output

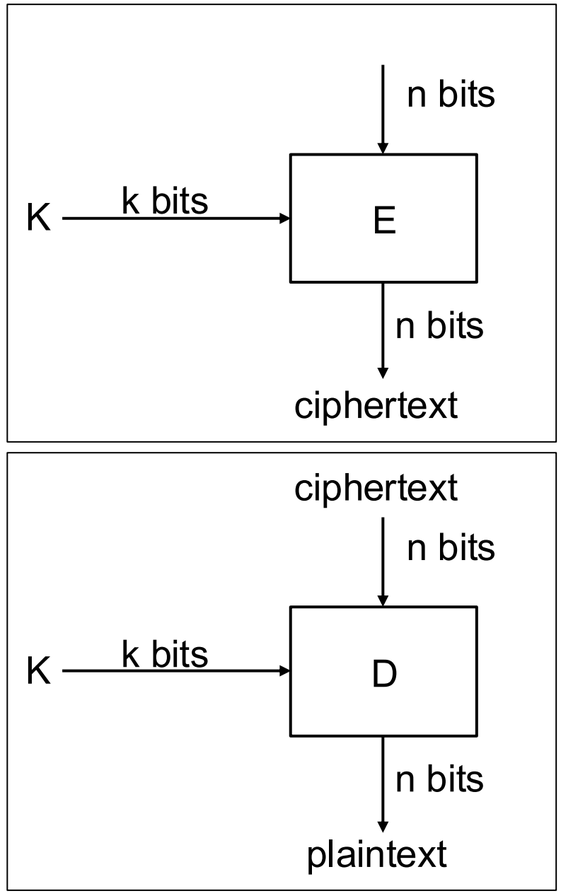
Block Ciphers: Definition

- **Block cipher**: A cryptographic scheme consisting of encode/decode algorithms for a fixed-sized block of bits:
- ***E*K(*M*) → *C*: Encode**
  - Inputs: *k*-bit key *K* and an *n*-bit plaintext *M*
  - Output: An *n*-bit ciphertext *C*
  - Sometimes written as: $\{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$



Block Ciphers: Definition

- **$D_K(C) \rightarrow M$: Decode**
  - Inputs: a k-bit key, and an n-bit ciphertext C
  - Output: An n-bit plaintext
  - Sometimes written as: $\{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$
  - The inverse of the encryption function
- **Properties**
  - **Correctness**: E_K is a permutation, D_K is its inverse
  - **Efficiency**: Encode/decode should be fast
  - **Security**: E behaves like a random permutation

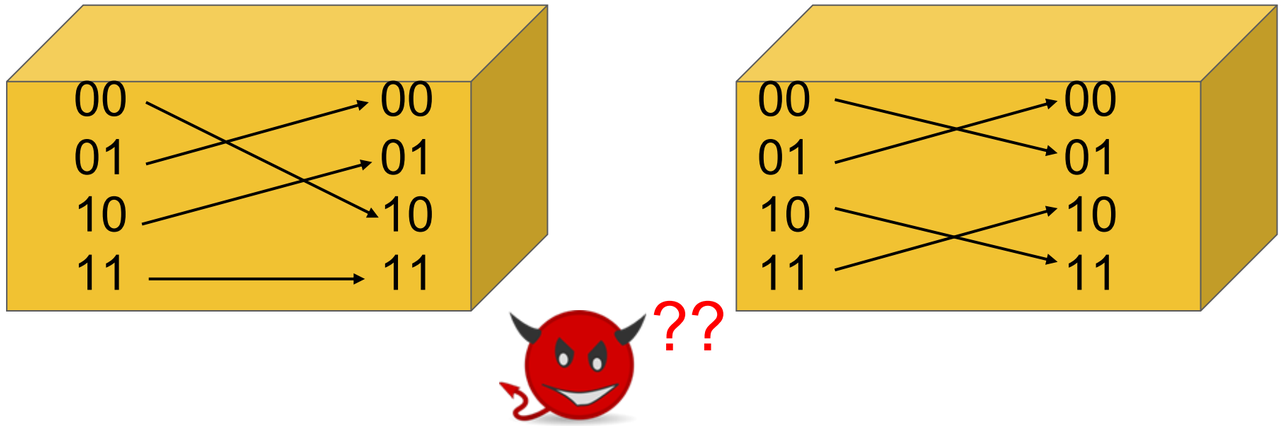


Block Ciphers: Security



- A secure block cipher **behaves like a permutation** (reversible function) **randomly chosen from the set of all permutations** on ***n*-bit strings**
  - A random permutation: Each *n*-bit input is mapped to one randomly-chosen *n*-bit output
- **Defined by a distinguishing game**
  - Eve gets two boxes: One is a randomly chosen permutation, and one is E_K with a randomly chosen key *K*
  - Eve should not be able to tell which is which with probability > ½+negl

Block Ciphers: Security



One of these is EK with a randomly chosen K, and the other one is a randomly chosen permutation. Eve can’t distinguish them.

Block ciphers: Brute-force attacks?

- **How hard is it to run a brute-force attack on a 128-bit key?**
  - We have to try 2^{128} possibilities. How big is 2^{128}?
- **Handy approximation: $2^{10} \approx 10^3$**
  - $2^{128} = 2^{10*12.8} \approx (10^3)^{12.8} \approx (10^3)^{13} = 10^{39}$
- Suppose we have massive hardware that can try 10^9 (1 billion) keys in 1 nanosecond (a billionth of a second). That's 10^{18} keys per second
  - We'll need $10^{39} / 10^{18} = 10^{21}$ seconds. How long is that?
  - One year $\approx 3\times10^7$ seconds
  - 10^{21} seconds / $3\times10^7 \approx 3\times10^{13}$ years ≈ 30 trillion years
- **Takeaway: Brute-forcing a 128-bit key takes astronomically long. Don't even try.**

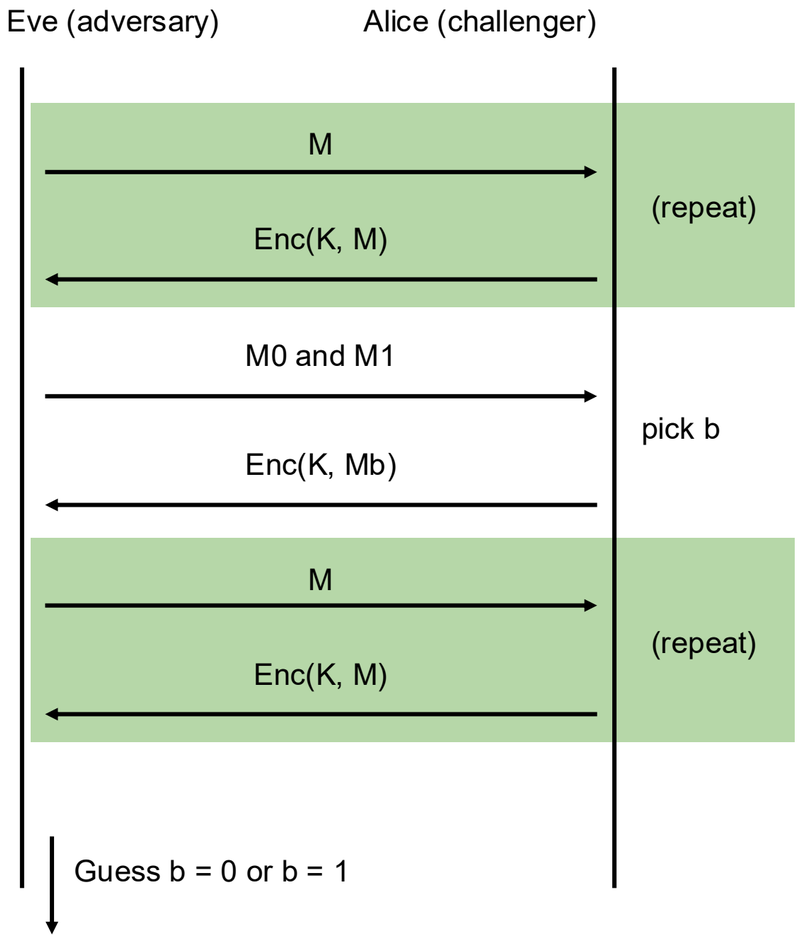
Block Ciphers: Efficiency



- **Encryption and decryption should be computable in microseconds**
- Block cipher algorithms typically use operations like XOR, bit-shifting, and small table lookups, which are very fast on modern processors
- **Modern CPUs provide dedicated hardware support for block ciphers**

Are Block Ciphers IND-CPA Secure?

- **Consider the following adversary:**
  - Eve sends two different messages M_0 and M_1
  - Eve receives either $E_K(M_0)$ or $E_K(M_1)$
  - Eve requests the encryption of M_0 again
  - Strategy: If the encryption of M_0 matches what she received, guess b = 0. Else, guess b = 1.
- **Eve can win the IND-CPA game with probability 1!**
  - Block ciphers are not IND-CPA secure because they are deterministic



AES (Advanced Encryption Standard)



- **Key size 128, 192, or 256 bits (k = 128, 192, or 256)**
  - Use key size 256 these days
- **Block size 128 bits (n = 128)**
  - Note: The block size is still always 128 bits, regardless of key size

Issues with Block Ciphers



- **Block ciphers are not IND-CPA secure, because they're deterministic**
  - A scheme is **deterministic** if the same input always produces the same output
  - No deterministic scheme can be IND-CPA secure because the adversary can always tell if the same message was encrypted twice
- **Block ciphers can only encrypt messages of a fixed size**
  - For example, AES can only encrypt-decrypt 128-bit messages
  - What if we want to encrypt something longer than 128 bits?
- To address these problems, we'll add **modes of operation** that use block ciphers as a building block!

Summary



- **Encryption: input a *k*-bit key and *n*-bit plaintext, receive *n*-bit ciphertext**
- **Decryption: input a *k*-bit key and *n*-bit ciphertext, receive *n*-bit plaintext**
- **Correctness: when the key is fixed, $E_K(M)$ should be bijective**
- **Security**
  - Without the key, E_K(m) is computationally indistinguishable from a random permutation
  - Brute-force attacks take astronomically long and are not possible
- **Efficiency: algorithms use XORs and bit-shifting (very fast)**
- **Implementation: AES is the modern standard**
- **Issues**
  - Not IND-CPA secure because they're deterministic
  - Can only encrypt *n*-bit messages