CSE 484/M584: Computer Security (and Privacy)

Spring 2025

David Kohlbrenner
dkohlbre@cs

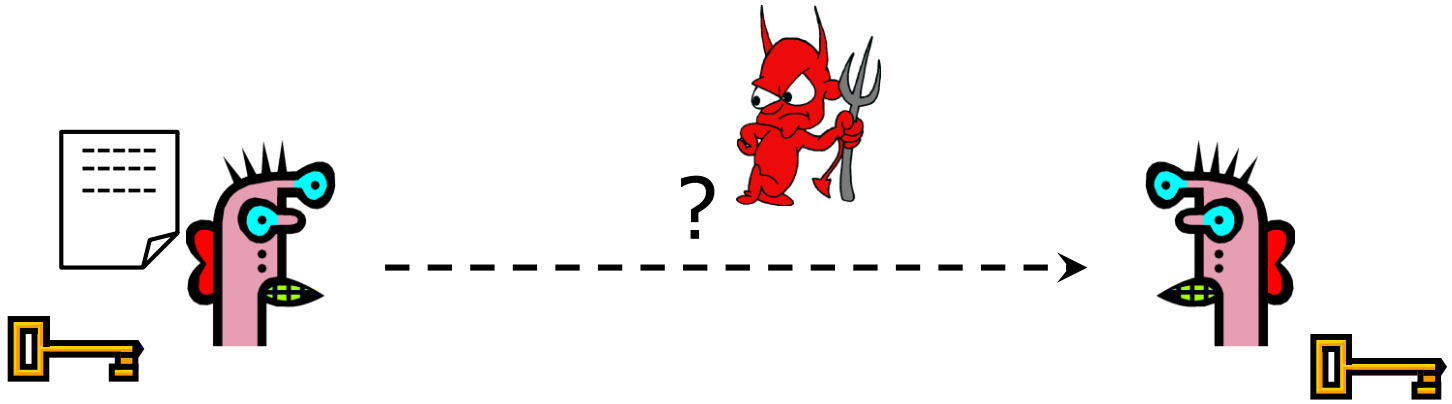
UW Instruction Team: David Kohlbrenner, Yoshi Kohno, Franziska Roesner, Nirvan Tyagi. Thanks to Dan Boneh, Dieter Gollmann, Dan Halperin, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials

Admin

- Lab 1b due tonight.
- Lab1a Writeup grades are out!
  - Regrade requests close on May 7th (hard deadline)
  - Exploit grades will be fixed later today.
- HW1 due next Wednesday
- Lab 2 (Cryptolab) goes out today!
  - We won't have covered everything, but worth reading through ASAP.

Symmetric Encryption

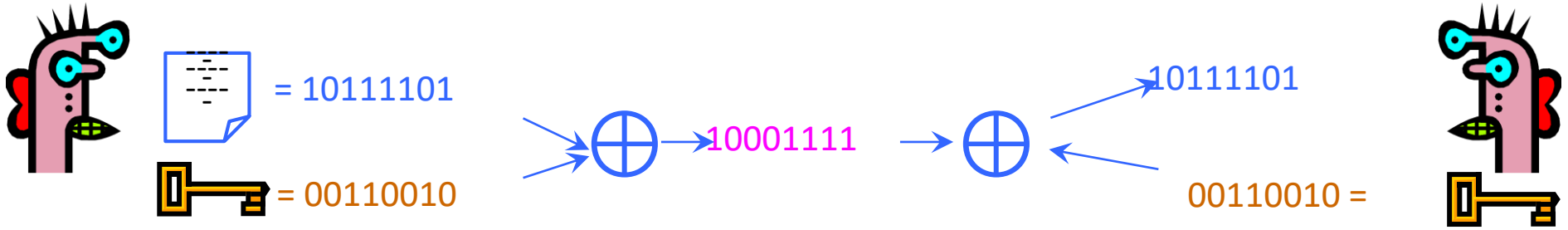
Goal: Confidentiality



Given: Both parties know a shared random key

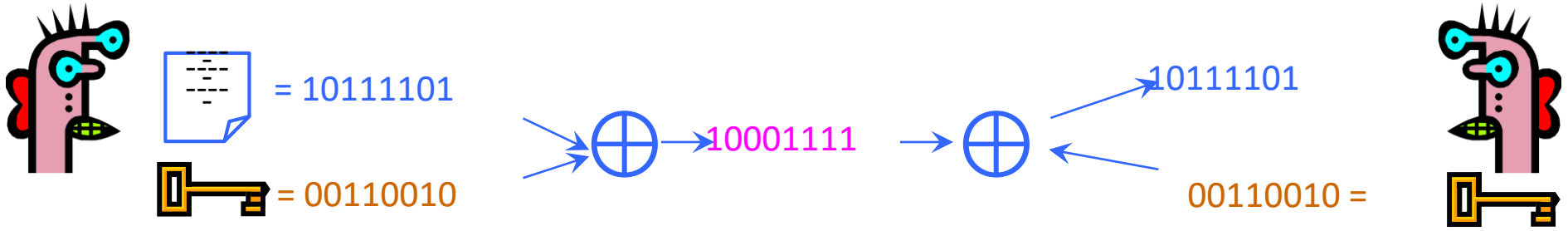
Goal: Send a message without revealing information about the message

One-Time Pad (Simple XOR Encryption)



Cipher achieves perfect secrecy if key is chosen uniformly at random (Claude Shannon, 1949)

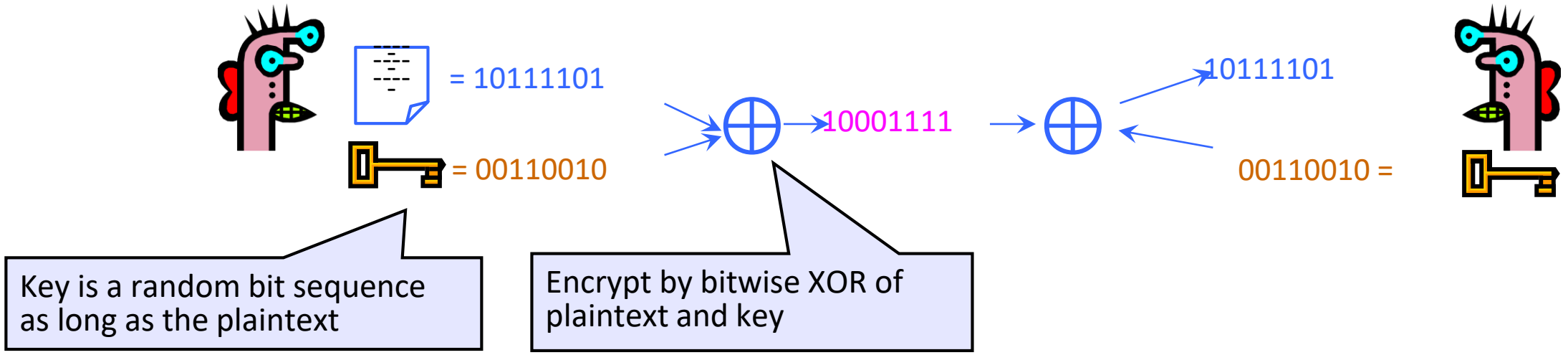
One-Time Pad (Simple XOR Encryption)



Key is a random bit sequence as long as the plaintext

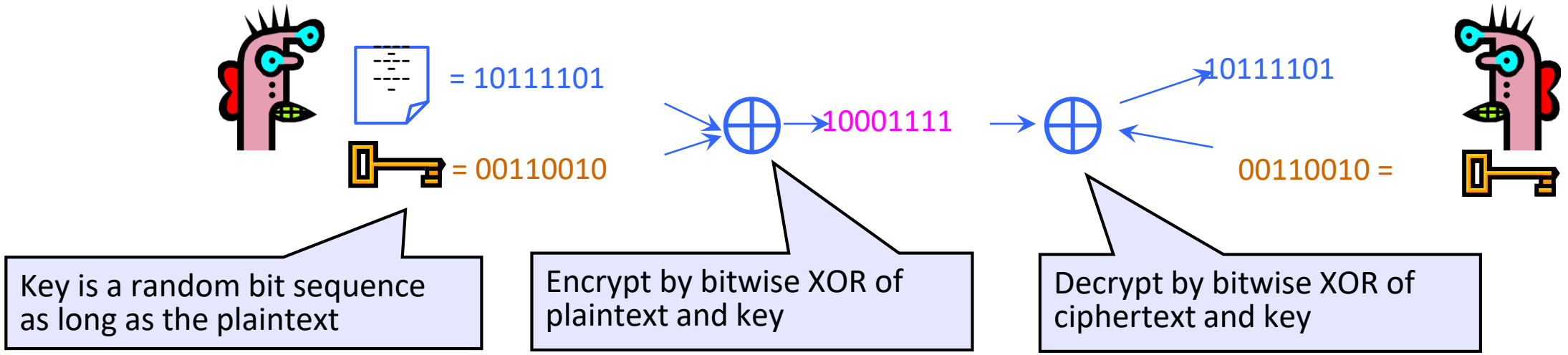
Cipher achieves perfect secrecy if key is chosen uniformly at random (Claude Shannon, 1949)

One-Time Pad (Simple XOR Encryption)



Cipher achieves perfect secrecy if key is chosen uniformly at random (Claude Shannon, 1949)

One-Time Pad (Simple XOR Encryption)



Cipher achieves perfect secrecy if key is chosen uniformly at random (Claude Shannon, 1949)

Advantages of One-Time Pad

- Easy to compute
  - Encryption and decryption are the same operation
  - Bitwise XOR is very cheap to compute

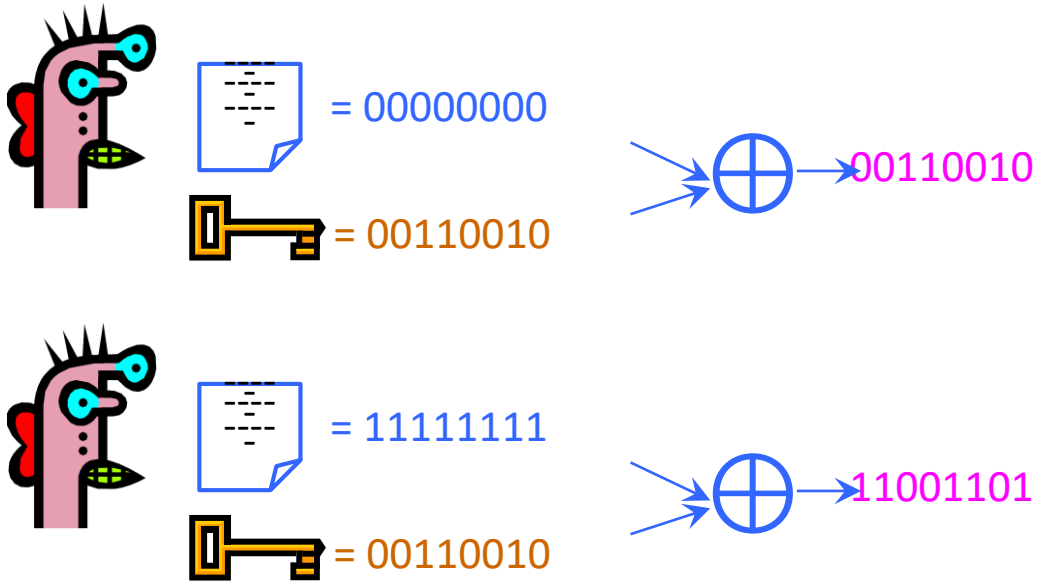
- Perfect secrecy
  - Given a ciphertext, all plaintexts are equally likely, regardless of attacker's computational resources
  - …as long as the key sequence is truly random
    - True randomness is expensive to obtain in large quantities
  - …as long as each key is same length as plaintext
    - But how does sender communicate the key to receiver?

Problems with the One-Time Pad?

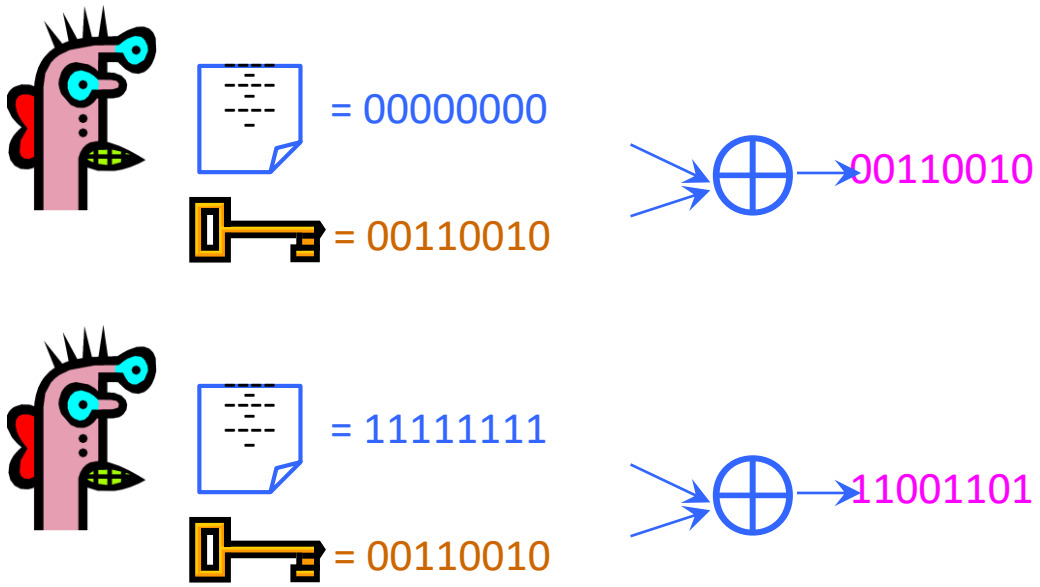
Gradescope

- What potential security problems do you see with the one-time pad?
- Q1: Confidentiality: What if key is reused?
- Q2: Integrity: Any protection against corruption?

Dangers of Reuse



Dangers of Reuse



Learn relationship between plaintexts

C1⊕C2 = (P1⊕K)⊕(P2⊕K) =

(P1⊕P2)⊕(K⊕K) = P1⊕P2

Problems with One-Time Pad

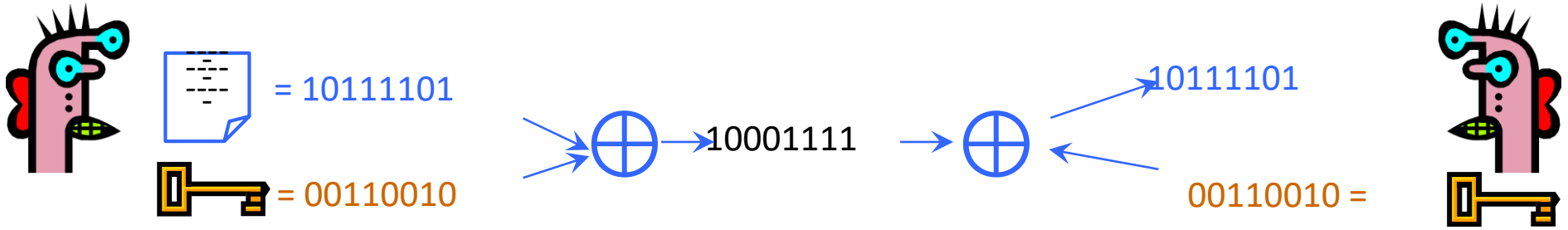
- (1) Key must be as long as the plaintext
  - Impractical in most realistic scenarios

Problems with One-Time Pad

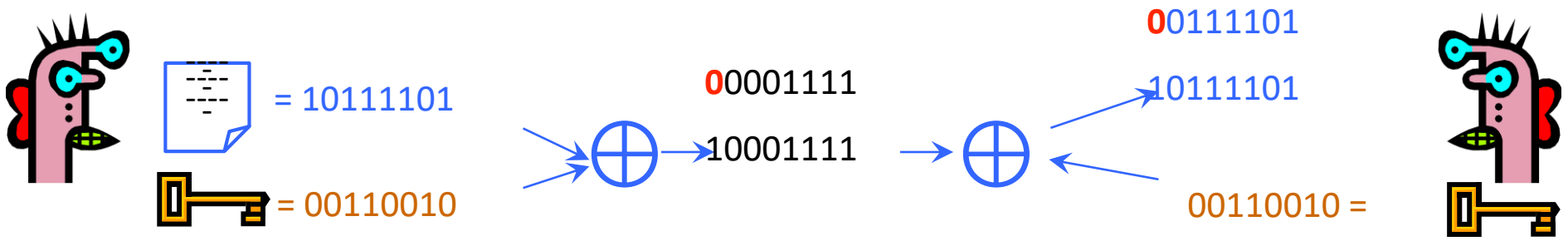
- (1) Key must be as long as the plaintext
  - Impractical in most realistic scenarios

- **(2) Insecure if keys are reused**
  - **Attacker can obtain XOR of plaintexts**

One-Time Pad Integrity?



One-Time Pad Integrity?



Problems with One-Time Pad

- (1) Key must be as long as the plaintext
  - Impractical in most realistic scenarios
- (2) Insecure if keys are reused
  - Attacker can obtain XOR of plaintexts
- **(3) Does not guarantee integrity**
  - **One-time pad only guarantees confidentiality**
  - **Attacker cannot recover plaintext, but can easily change it to something else**

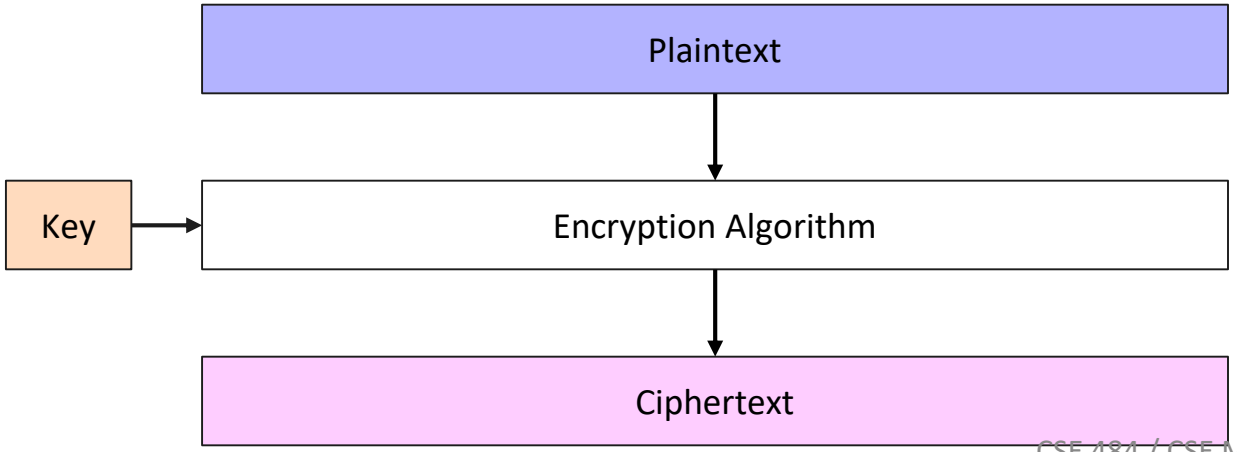
Reducing Key Size

- What to do when it is infeasible to pre-share huge random keys?
  - When one-time pad is unrealistic…

- Use special cryptographic primitives: block ciphers
  - Single key can be re-used (with some restrictions)
  - Not as theoretically secure as one-time pad

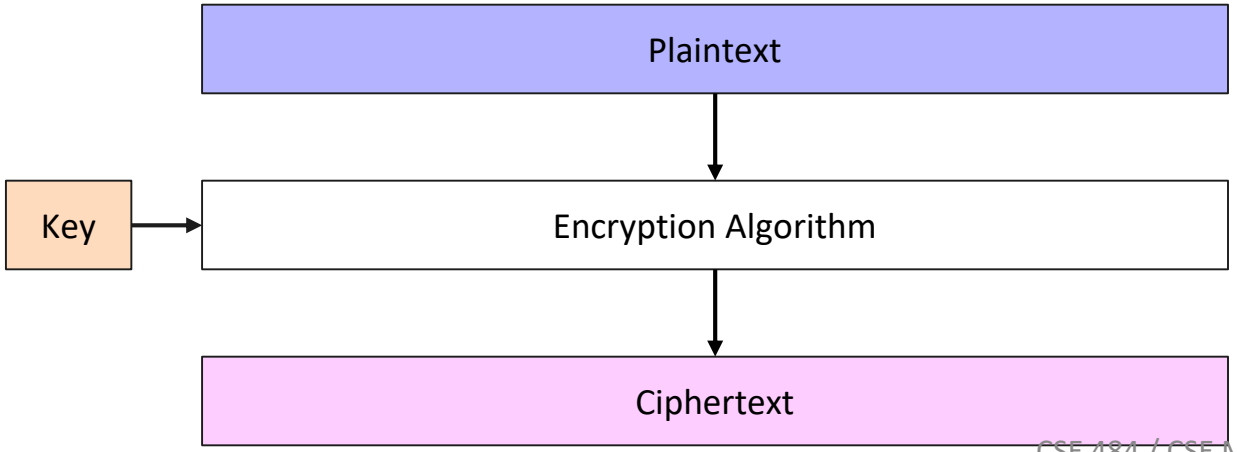
Symmetric Encryption Goal

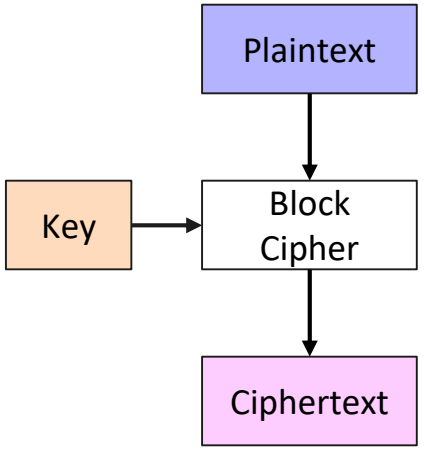
- Operates on an arbitrary size plaintext
  - Each key defines a random and invertible mapping from plaintext space to ciphertext space



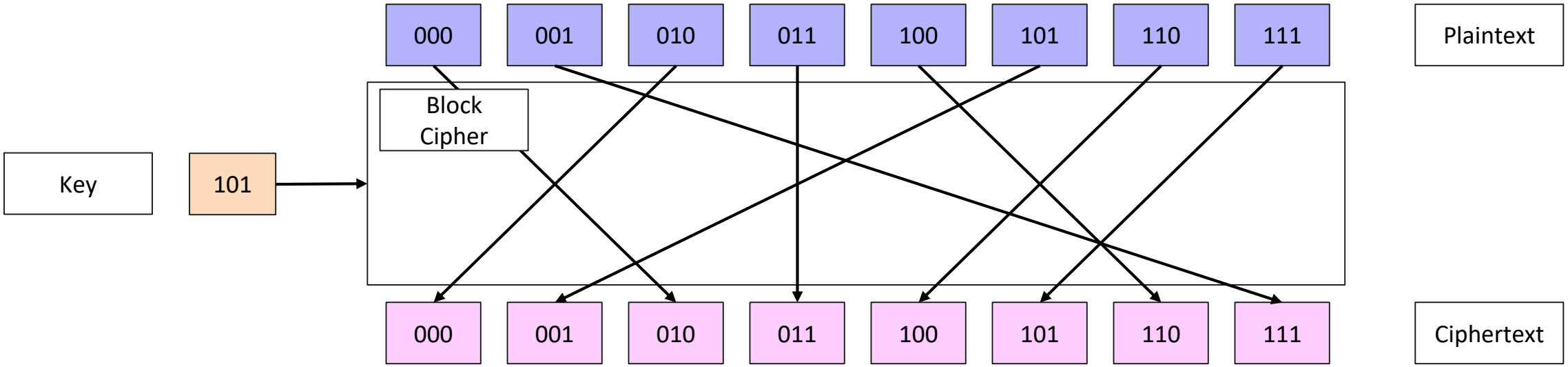
Symmetric Encryption Goal

- Operates on an arbitrary size plaintext
  - Each key defines a random and invertible mapping from plaintext space to ciphertext space
- Block ciphers operate on a single chunk ("block") of plaintext
  - Block sizes on the order of 64 bits, 128 bits, 256 bits
  - Each key defines a "random" invertible **permutation** of inputs to possible outputs
  - Plan: Build encryption algorithm from block cipher building block

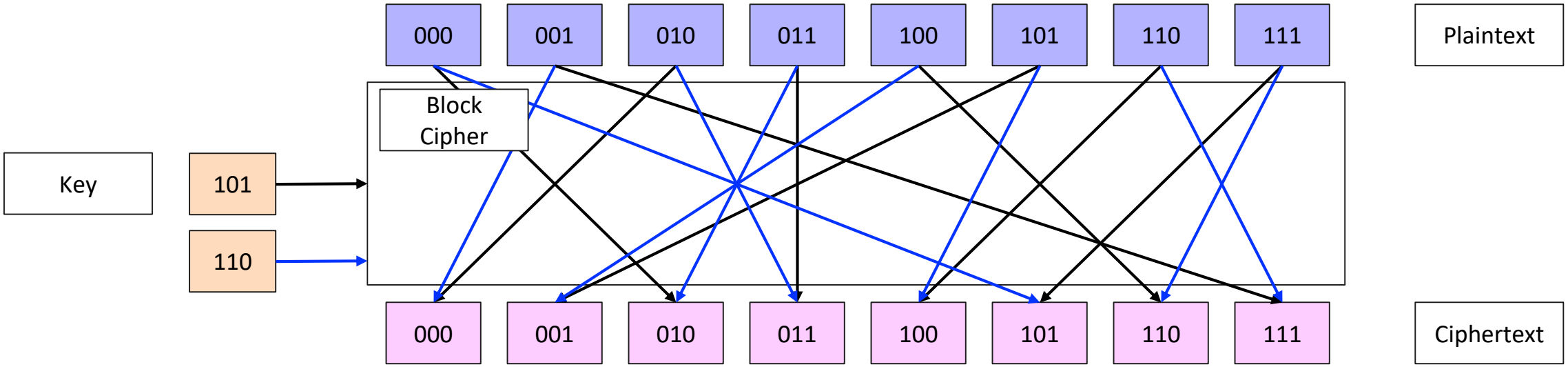




Keyed Permutation

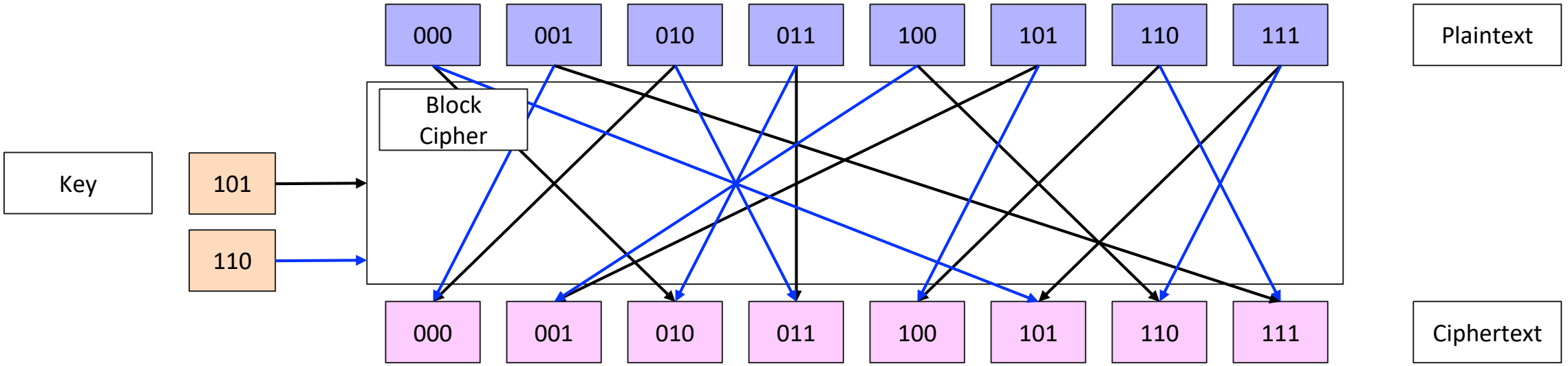


Keyed Permutation



- Each key defines a permutation from the plaintext space to the ciphertext space
- Ideal: Permutation looks "random" if you don't know the key
  - Is that possible?
  - How many permutations exist?
- Next best: Computationally indistinguishable from random!

Keyed Permutation



- Each key defines a permutation from the plaintext space to the ciphertext space
- Ideal: Permutation looks "random" if you don't know the key
  - Is that possible?
  - How many permutations exist?
- Next best: Computationally indistinguishable from random!

For N-bit input:
$2^N!$ possible permutations

For K-bit key:
2^K possible keys

Block Cipher Security

- Security: For a randomly selected key: Computationally hard to distinguish outputs of the block cipher from outputs of a truly random permutation

- Popular block ciphers (e.g., AES) do not have proofs of security!
  - Design is open and standard is created through public competition
  - Current best attacks against AES-128 take 2^{126} time
  - Block ciphers with proofs of security exist but are less efficient

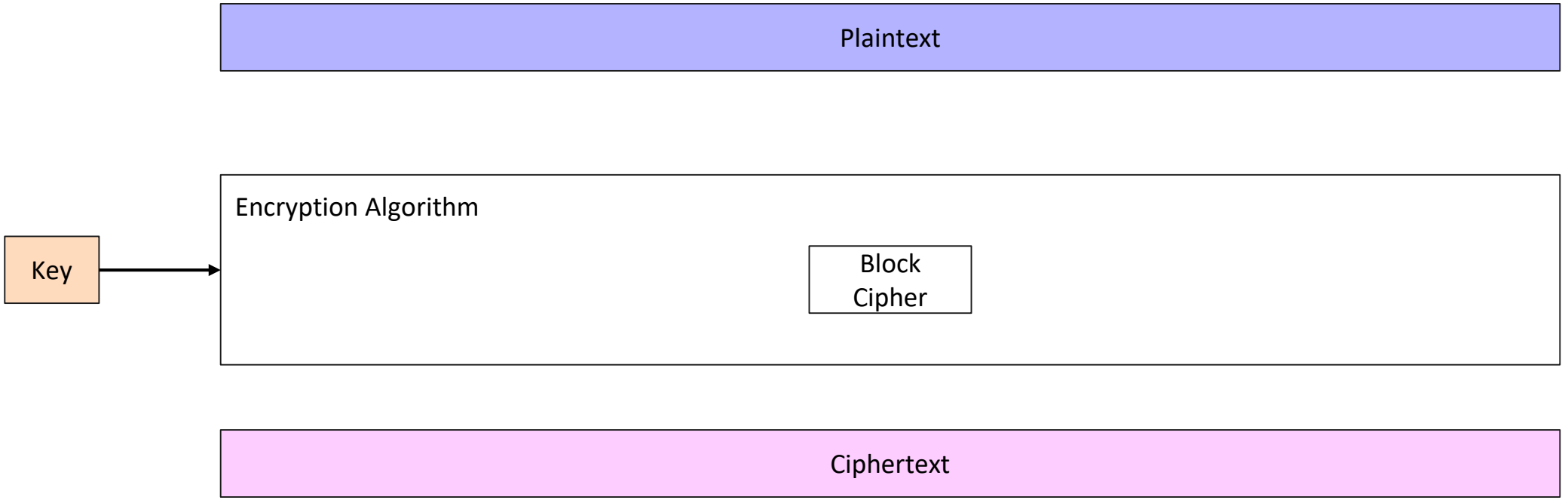
Standard Block Ciphers

- **DES: Data Encryption Standard**
  - Feistel structure: builds invertible function using non-invertible ones
  - Invented by IBM, issued as federal standard in 1977
  - 64-bit blocks, 56-bit key + 8 bits for parity

- **AES: Advanced Encryption Standard**
  - New federal standard as of 2001
    - NIST: National Institute of Standards & Technology
  - Based on the Rijndael algorithm
    - Selected via an open process
  - 128-bit blocks, keys can be 128, 192 or 256 bits

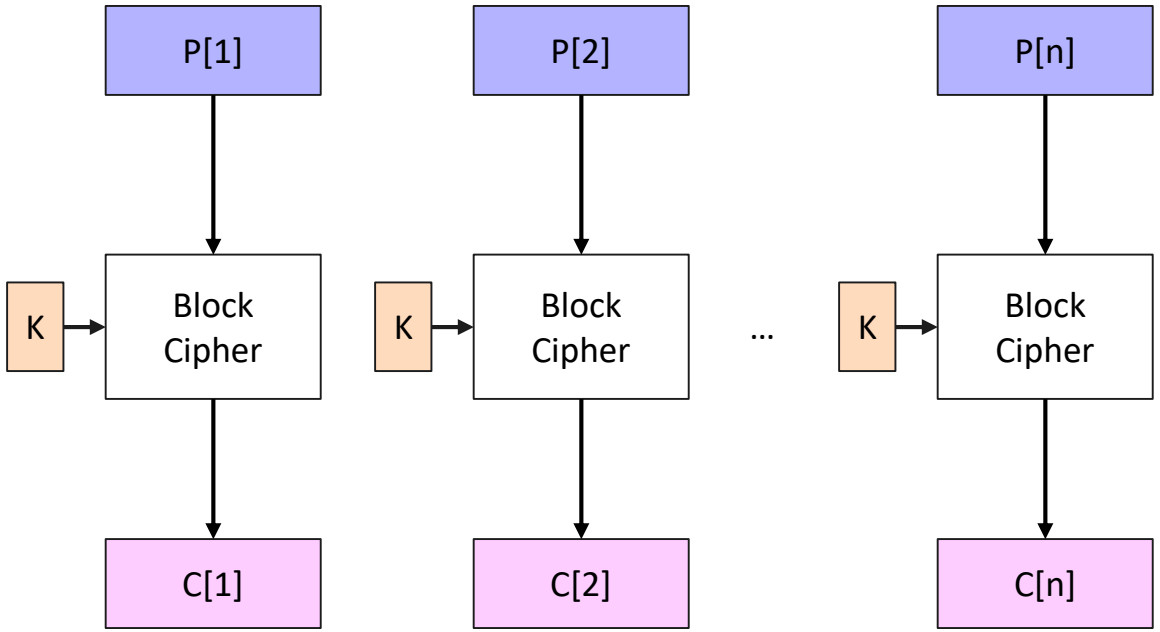
Encrypting a Large Message

- How do we use block ciphers to encrypt larger messages?
- Plan: Build an encryption “mode” from a block cipher



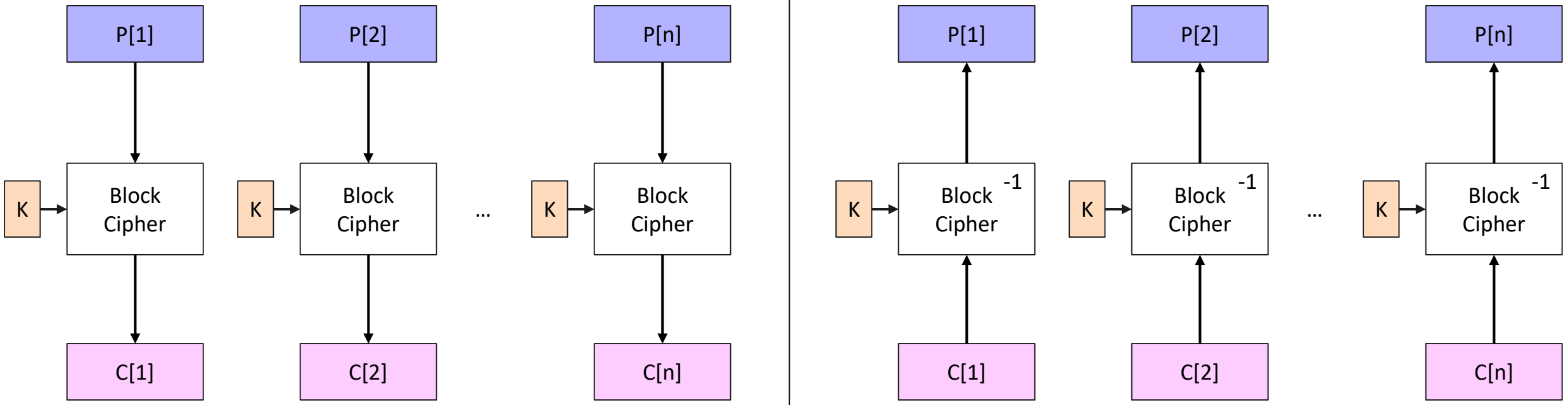
Electronic Code Book (ECB) Mode

- Idea: Simply split up the plaintext into block-sized chunks!



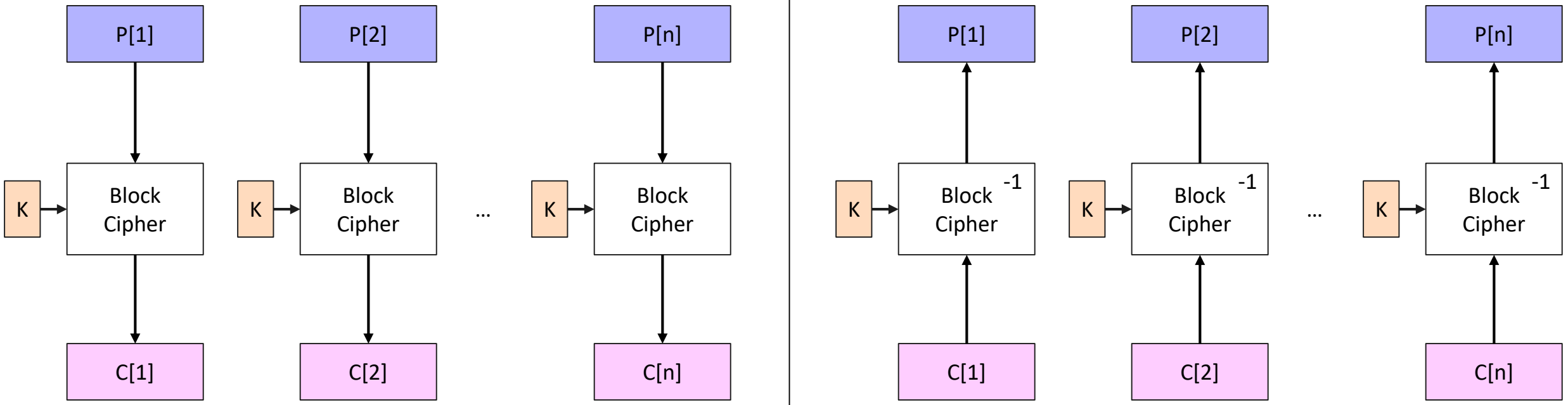
Electronic Code Book (ECB) Mode

- Idea: Simply split up the plaintext into block-sized chunks!



Electronic Code Book (ECB) Mode

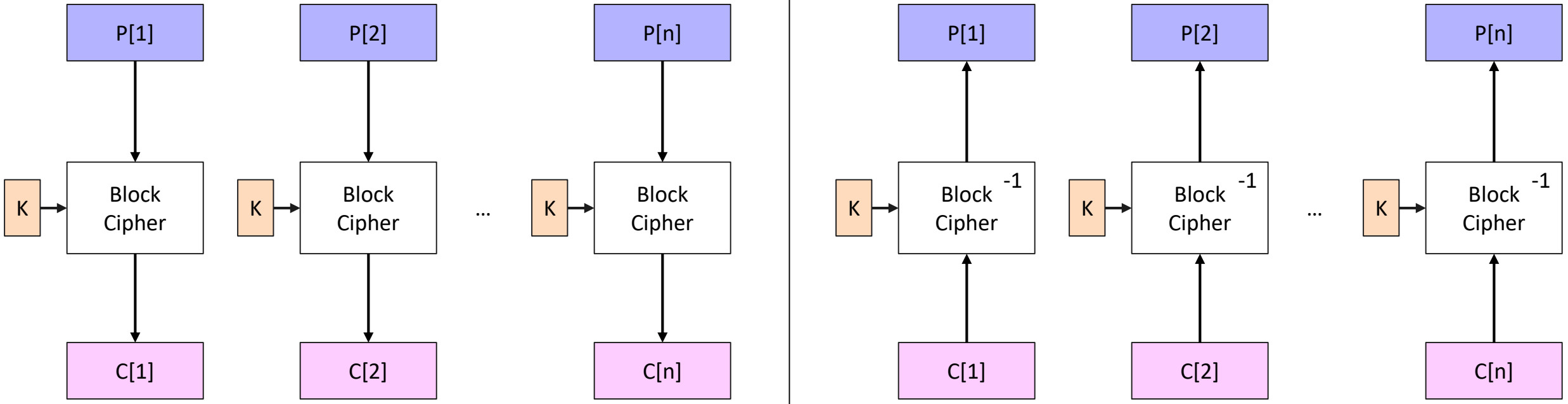
- Idea: Simply split up the plaintext into block-sized chunks!



In-class Activity 1/29: Good idea?

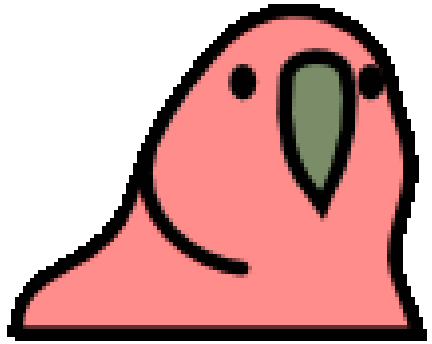
Electronic Code Book (ECB) Mode

- Idea: Simply split up the plaintext into block-sized chunks!



- Identical blocks of plaintext produce identical blocks of ciphertext
- No integrity checks: can mix and match blocks

Information Leakage in ECB Mode



Encrypt in ECB mode