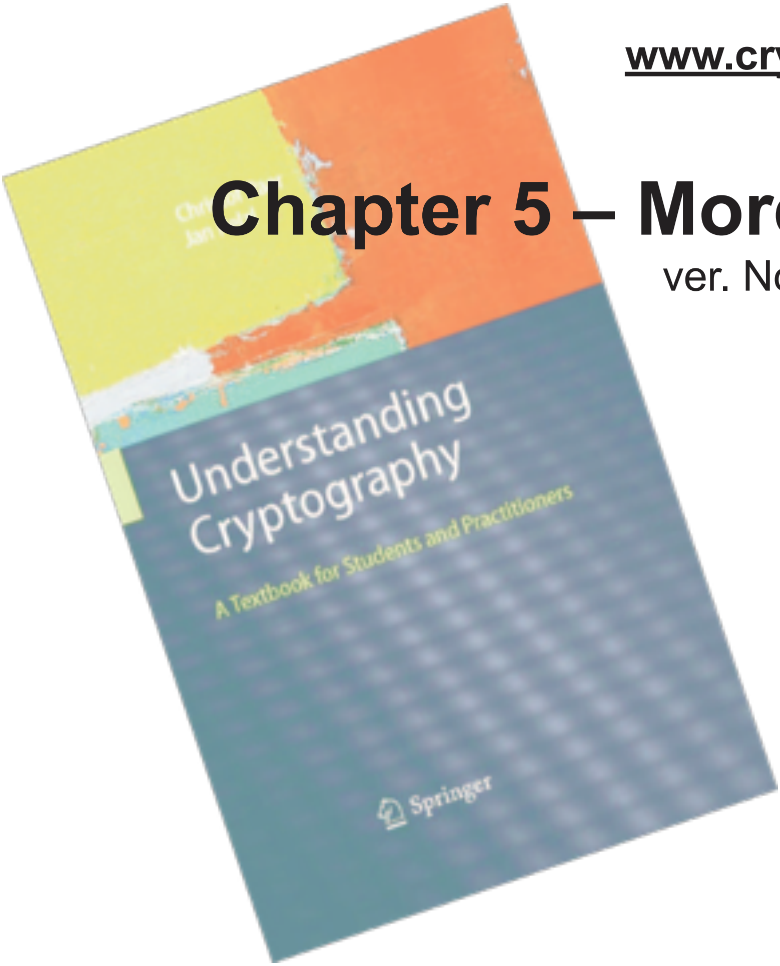
Understanding Cryptography – A Textbook for Students and Practitioners

by Christof Paar and Jan Pelzl

www.crypto-textbook.com



Chapter 5 – More About Block Ciphers

ver. November 26, 2010

Last modified 10-2-17

These slides were prepared by Amir Moradi, Christof Paar and Jan Pelzl
And modified by Sam Bowne

Some legal stuff (sorry): Terms of Use

- The slides can used free of charge. All copyrights for the slides remain with Christof Paar and Jan Pelzl.
- The title of the accompanying book “Understanding Cryptography” by Springer and the author’s names must remain on each slide.
- If the slides are modified, appropriate credits to the book authors and the book title must remain within the slides.
- It is not permitted to reproduce parts or all of the slides in printed form whatsoever without written consent by the authors.

Contents

Modular Arithmetic: Multiplication and Multiplicative Inverses

In this chapter you will learn

- the most important modes of operation for block ciphers in practice
- security pitfalls when using modes of operations
- the principles of key whitening
- why double encryption is not a good idea, and the meet-in-the-middle attack
- triple encryption

Block Ciphers

- A block cipher is much more than just an encryption algorithm, it can be used ...
  - to build different types of block-based encryption schemes
  - to realize stream ciphers
  - to construct hash functions
  - to make message authentication codes
  - to build key establishment protocols
  - to make a pseudo-random number generator
  - ...
- The security of block ciphers also can be increased by
  - key whitening
  - multiple encryption

5.1 Encryption with Block Ciphers: Modes of Operation

Encryption with Block Ciphers

- There are several ways of encrypting long plaintexts, e.g., an e-mail or a computer file, with a block cipher (“modes of operation”)
  - Electronic Code Book mode (ECB)
  - Cipher Block Chaining mode (CBC)
  - Output Feedback mode (OFB)
  - Cipher Feedback mode (CFB)
  - Counter mode (CTR)
  - Galois Counter Mode (GCM)
- All of the 6 modes provide **confidentiality**
  - They may also provide **authenticity** and **integrity**:
    - Is the message really coming from the original sender? (authenticity)
    - Was the ciphertext altered during transmission? (integrity)

Block Size

- ECB and CBC require plaintext that's an exact multiple of the block size
  - Otherwise, plaintext must be padded
- CFB, OFB and CTR modes use a block cipher to create a stream cipher
  - *Error on page 124: CFB -> CBC (Link Ch 5a)*

Block Size

- ECB and CBC require plaintext that's an exact multiple of the block size

- **CBC in Python**

```
>>> from Crypto.Cipher import AES
>>> key = "Sixteen byte key"
>>> plain1 = "X"
>>> plain16 = "0123456789abcdef"
>>> iv = "0123456789abcdef"
>>> cipher = AES.new(key, AES.MODE_CBC, iv)
>>> cipher.encrypt(plain16)
'\xcch\x08A\x93;\xa9:\xa9*\n\xeaA]\x13\xec'
>>> cipher.encrypt(plain1)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "build/bdist.macosx-10.11-x86_64/egg/Crypto/Cipher/blockalgo.py", l
ine 244, in encrypt
ValueError: Input strings must be a multiple of 16 in length
```

Block Size

- CFB, OFB and CTR modes use a block cipher to create a stream cipher
- Works for **CFB** and **CTR** but not **OFB**

```
>>> cipher = AES.new(key, AES.MODE_CFB, iv)
>>> cipher.encrypt(plain1)
'\x19'
```

```
>>> counter = "0123456789abcdef"
>>> cipher = AES.new(key, AES.MODE_CTR, counter=lambda: counter)
>>> cipher.encrypt(plain1)
'\x19'
```

```
>>> cipher = AES.new(key, AES.MODE_OFB, iv)
>>> cipher.encrypt(plain1)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "build/bdist.macosx-10.11-x86_64/egg/Crypto/Cipher/blockalgo.py", l
ine 244, in encrypt
ValueError: Input strings must be a multiple of 16 in length
```

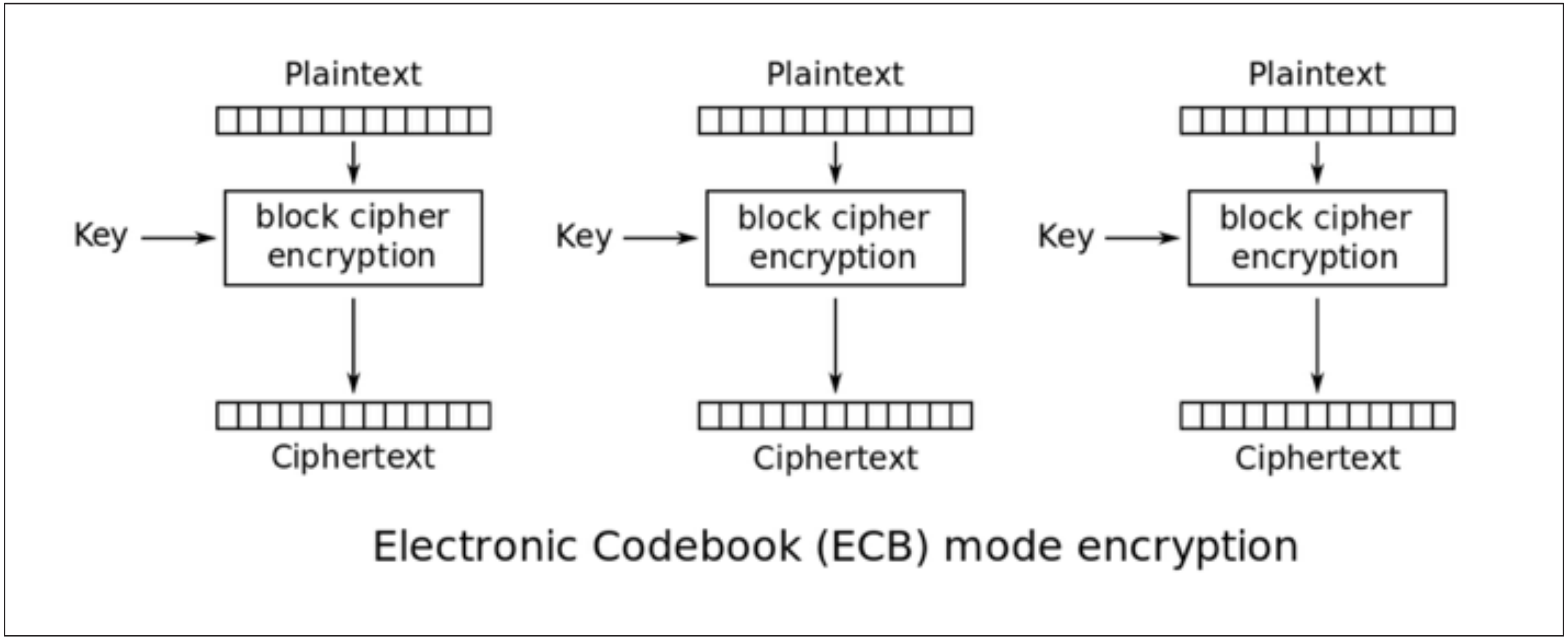
A Bug in Python

- Link Ch 4d

5.1.1 Electronic Codebook Mode (ECB)

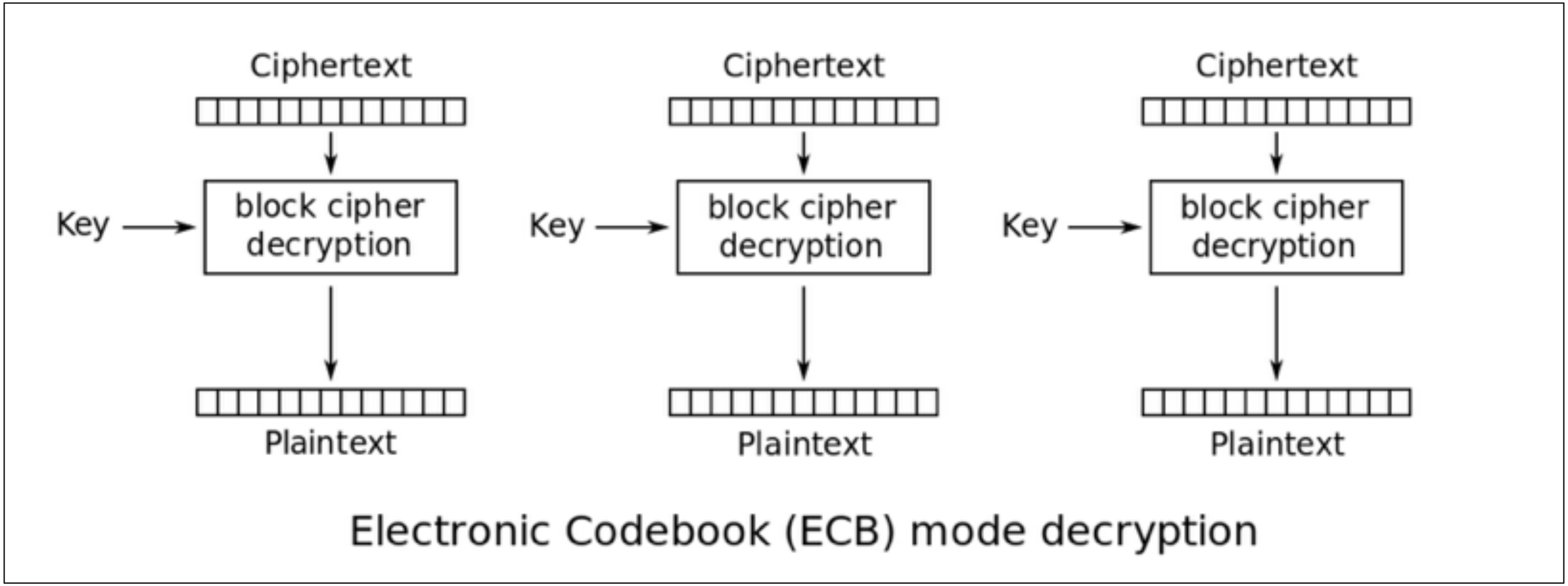
Electronic Code Book mode (ECB)

- Messages which exceed b bits are partitioned into b-bit blocks
- **Each Block is encrypted separately**
  - Image from Wikipedia (Link Ch 5a)



Electronic Code Book mode (ECB)

- Image from Wikipedia (Link Ch 5a)



ECB Advantages

- No block synchronization between sender and receiver is required
  - OK if some blocks are lost in transit
- Bit errors caused by noisy channels only affect the corresponding block but not succeeding blocks
- Block cipher operating can be parallelized
  - Advantage for high-speed implementations

ECB Disadvantages

- ECB encrypts highly deterministically
  - Identical plaintexts result in identical ciphertexts
  - An attacker recognizes if the same message has been sent twice
    - Simply by looking at the ciphertext: ***traffic analysis***
  - Plaintext blocks are encrypted independently of previous blocks
    - An attacker may reorder ciphertext blocks which results in valid plaintext

Substitution Attack on ECB

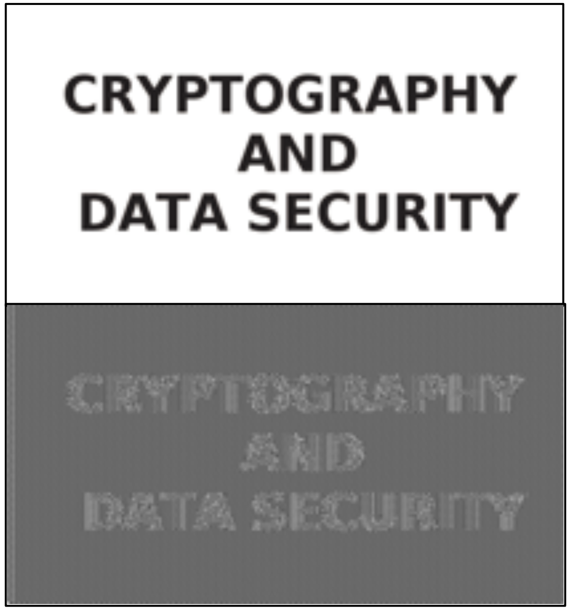
- Once a particular plaintext to ciphertext block mapping $x_i \rightarrow y_i$ is known, a sequence of ciphertext blocks can easily be manipulated
- Consider an *electronic bank transfer*

Block #	1	2	3	4	5
	Sending Bank A	Sending Account #	Receiving Bank B	Receiving Account #	Amount $

- the encryption key between the two banks does not change too frequently
- The attacker sends $1.00 transfers from his account at bank A to his account at bank B repeatedly
  - He can check for ciphertext blocks that repeat, and he stores blocks 1,3 and 4 of these transfers
- He now simply replaces block 4 of other transfers with the block 4 that he stored before
  - *all transfers* from some account of bank A to some account of bank B are redirected to go into the attacker's B account!

Example of encrypting bitmaps in ECB mode

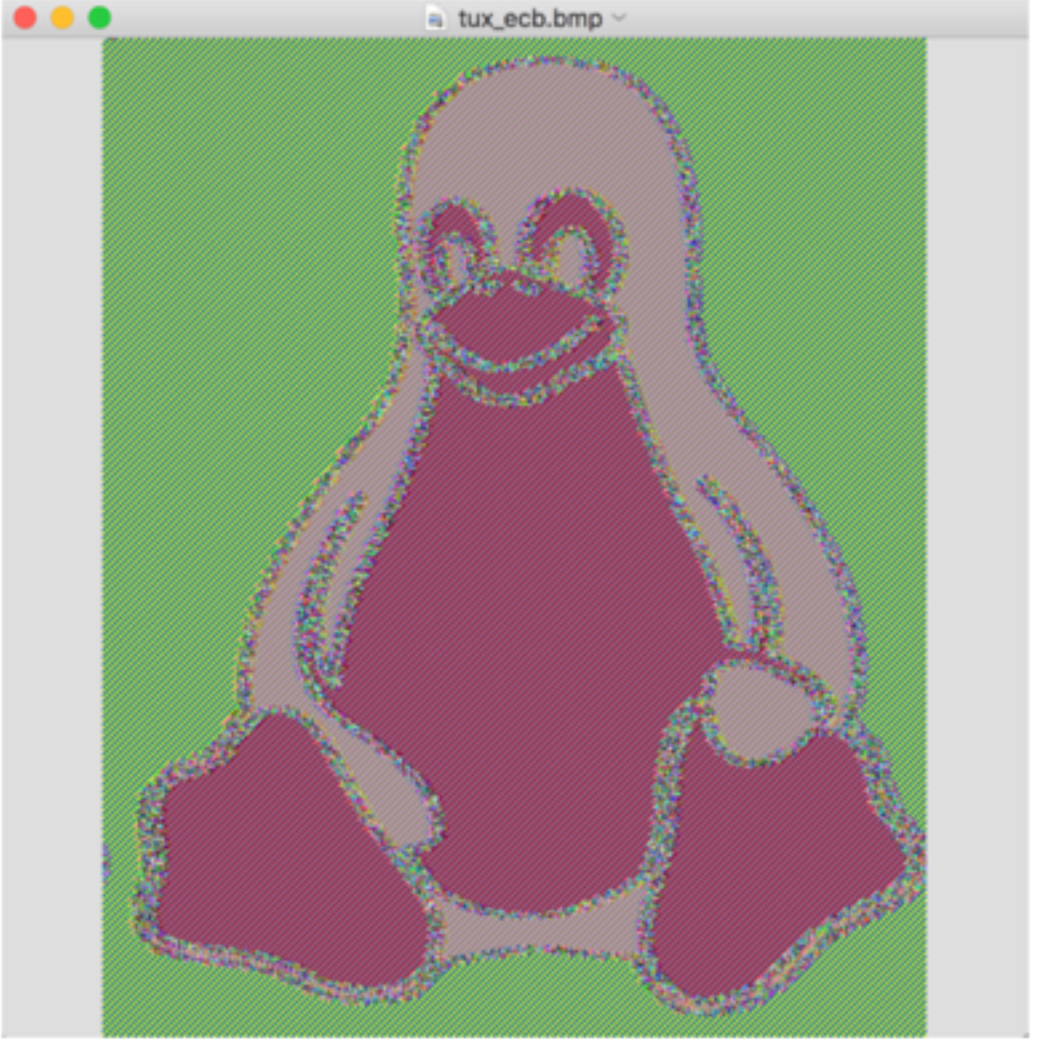
- Identical plaintexts are mapped to identical ciphertexts



- Statistical properties in the plaintext are preserved in the ciphertext

Example of encrypting bitmaps in ECB mode

Project 8



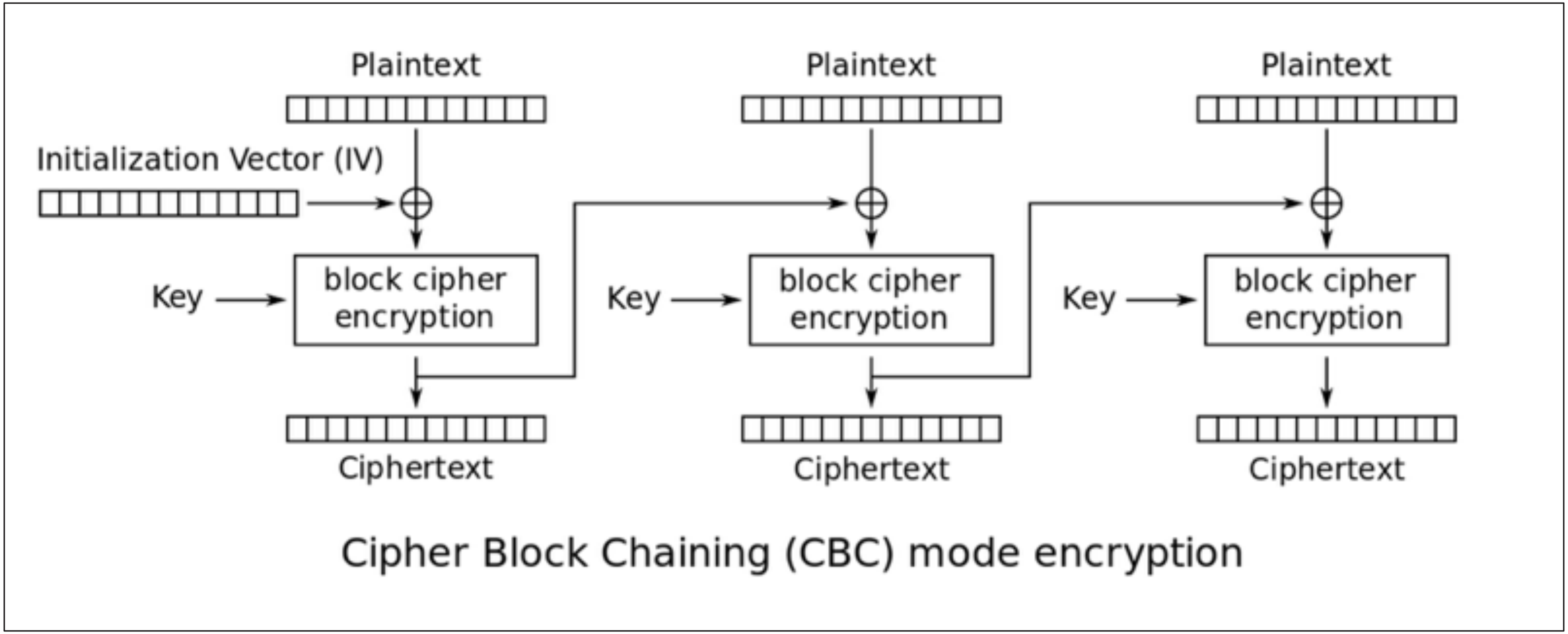
5.1.2 Cipher Block Chaining Mode (CBC)

Cipher Block Chaining mode (CBC)

- There are two main ideas behind the CBC mode:
  - The encryption of all blocks are “**chained** together”
    - ciphertext y_i depends not only on block x_i but on all previous plaintext blocks as well
  - The encryption is randomized by using an **initialization vector (IV)**

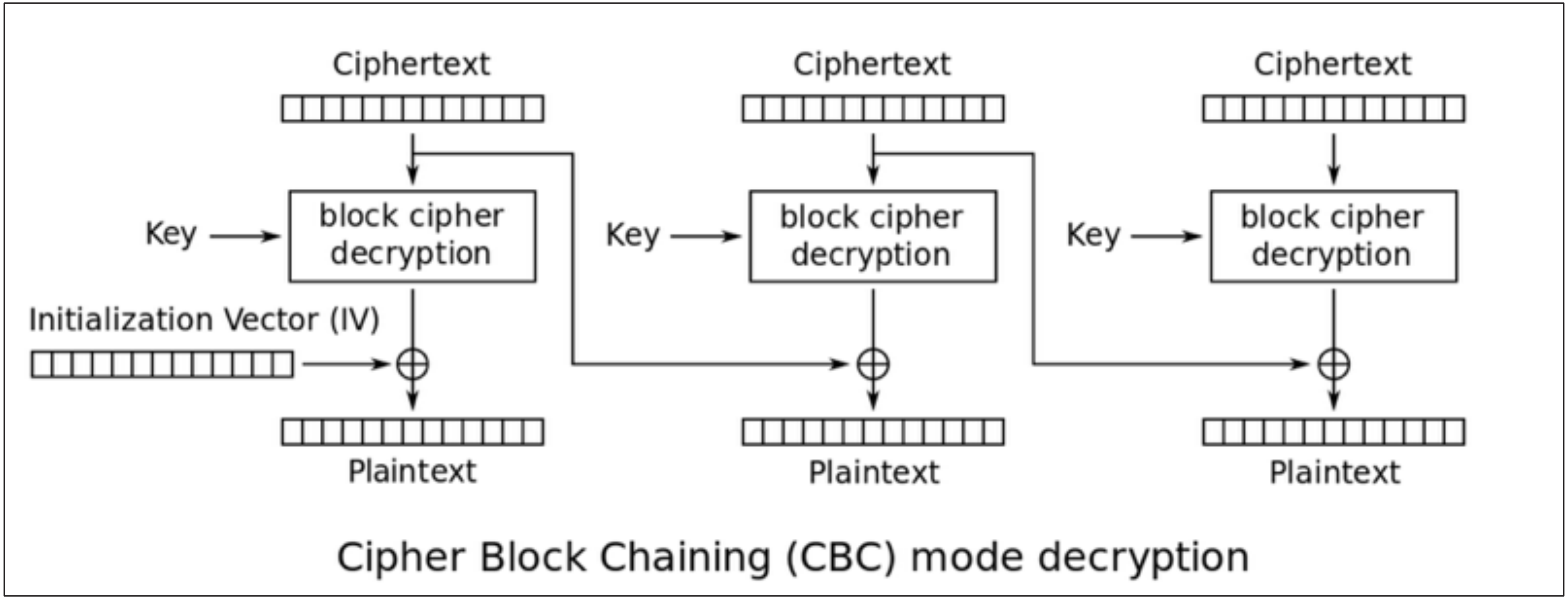
Cipher Block Chaining mode (CBC)

- Image from Wikipedia (Link Ch 5a)



Cipher Block Chaining mode (CBC)

- Image from Wikipedia (Link Ch 5a)



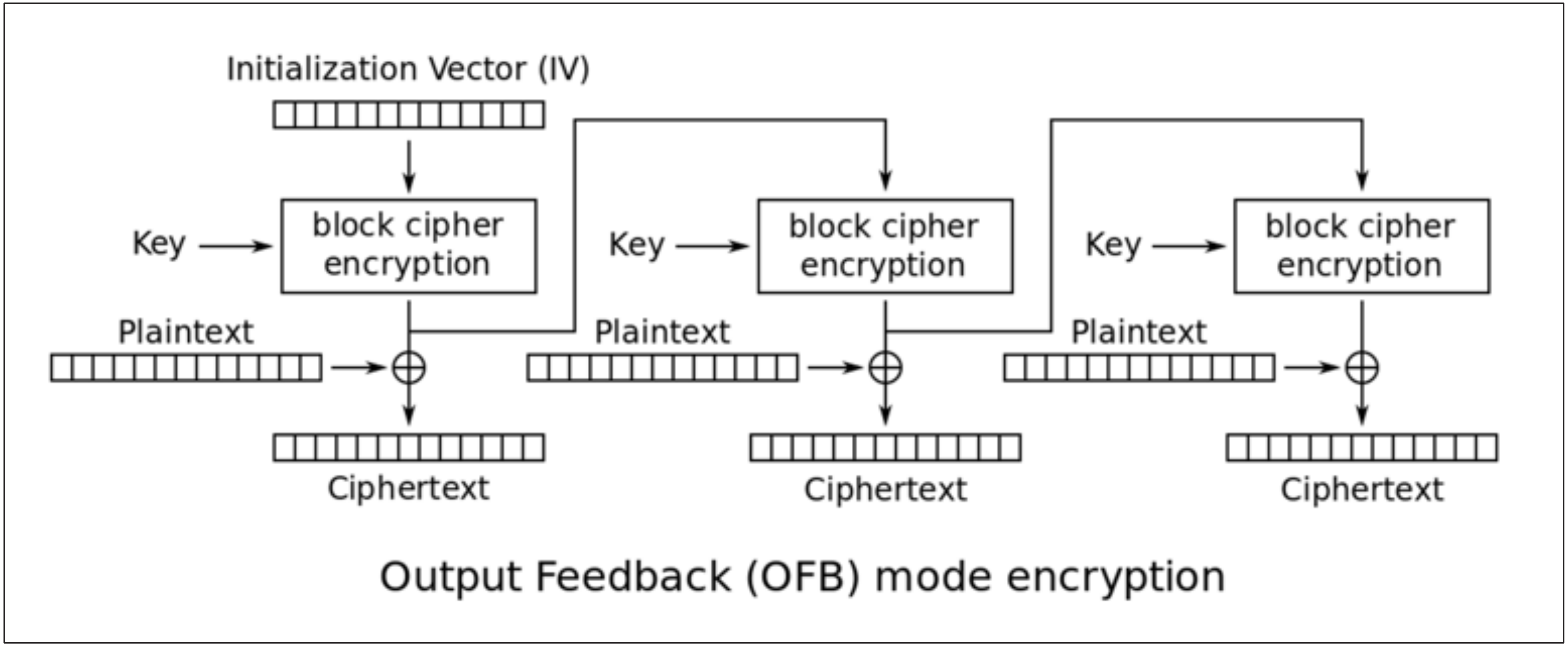
Substitution Attack on CBC

- Consider the last example (*electronic bank transfer*)
- If the IV is properly chosen for every wire transfer, the attack will not work at all
- If the IV is kept the same for several transfers, the attacker would recognize the transfers from his account at bank A to back B
- If we choose a new IV every time we encrypt, the CBC mode becomes a probabilistic encryption scheme, i.e., two encryptions of the same plaintext look entirely different
- It is not needed to keep the IV *secret*! It can be sent in plaintext.
- But it should be *unpredictable*

5.1.3 Outbook Feedback Mode (OFB)

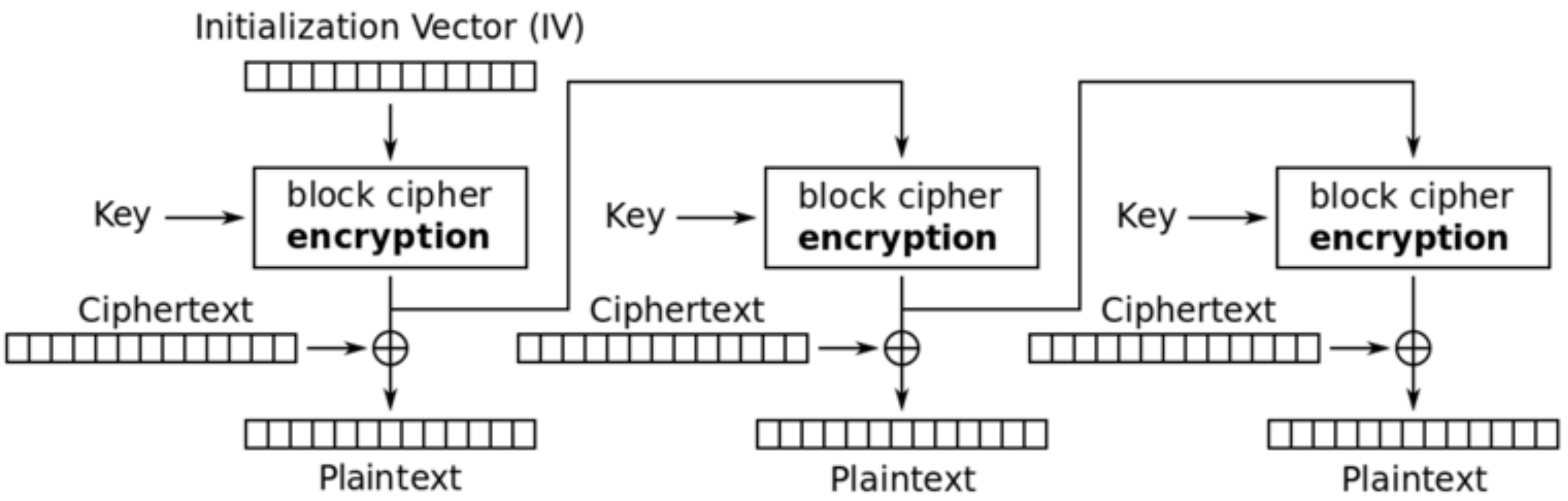
Output Feedback mode (OFB)

- It is used to build a *synchronous* **stream cipher** from a block cipher
- The key stream is not generated bitwise but instead in a blockwise fashion
- The output of the cipher gives us key stream bits S_i with which we can encrypt plaintext bits using the XOR operation
    - Image from Wikipedia (Link Ch 5a)



Output Feedback mode (OFB)

- Image from Wikipedia (Link Ch 5a)



Output Feedback (OFB) mode decryption

5.1.4 Cipher Feedback Mode (CFB)

Cipher Feedback mode (CFB)

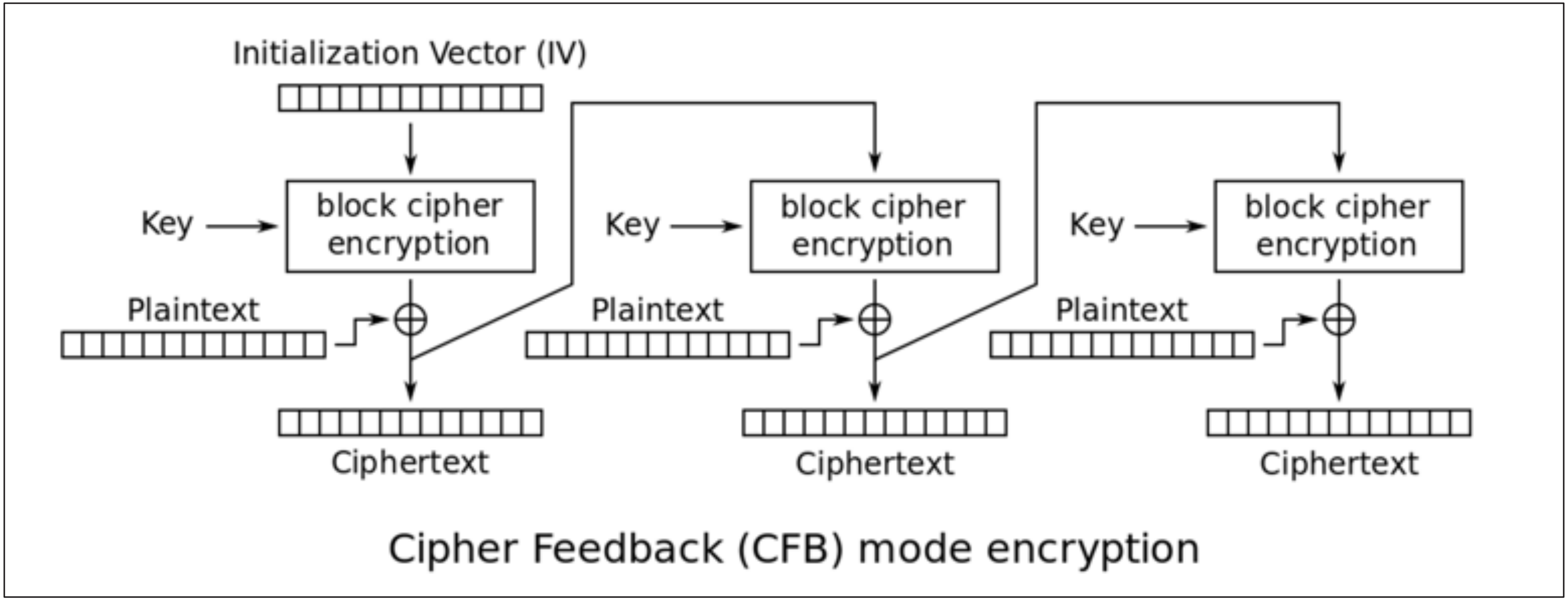
- It uses a block cipher as a building block for an asynchronous **stream cipher**

  similar to the OFB mode

- The key stream S_i is generated in a blockwise fashion and is also a function of the ciphertext
- As a result of the use of an IV, the CFB encryption is also nondeterministic
- It can be used in situations where short plaintext blocks are to be encrypted

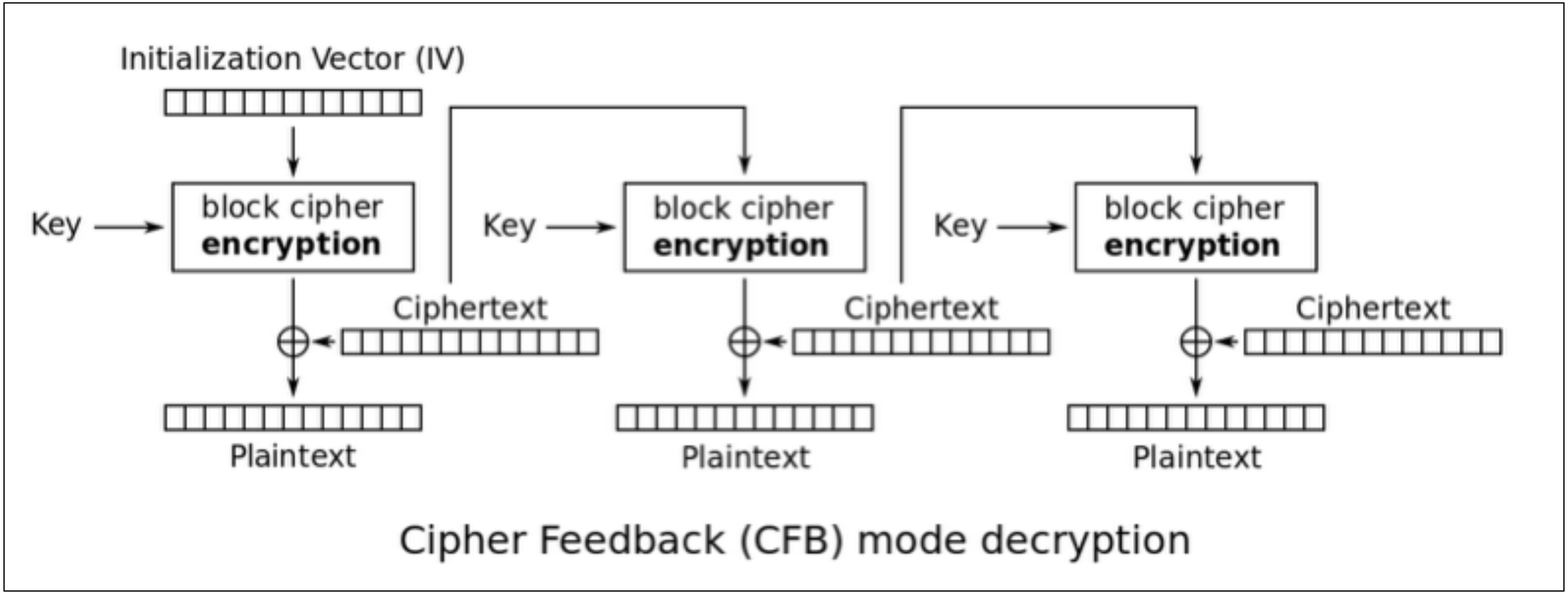
Cipher Feedback mode (CFB)

- Image from Wikipedia (Link Ch 5a)



Cipher Feedback mode (CFB)

- Image from Wikipedia (Link Ch 5a)



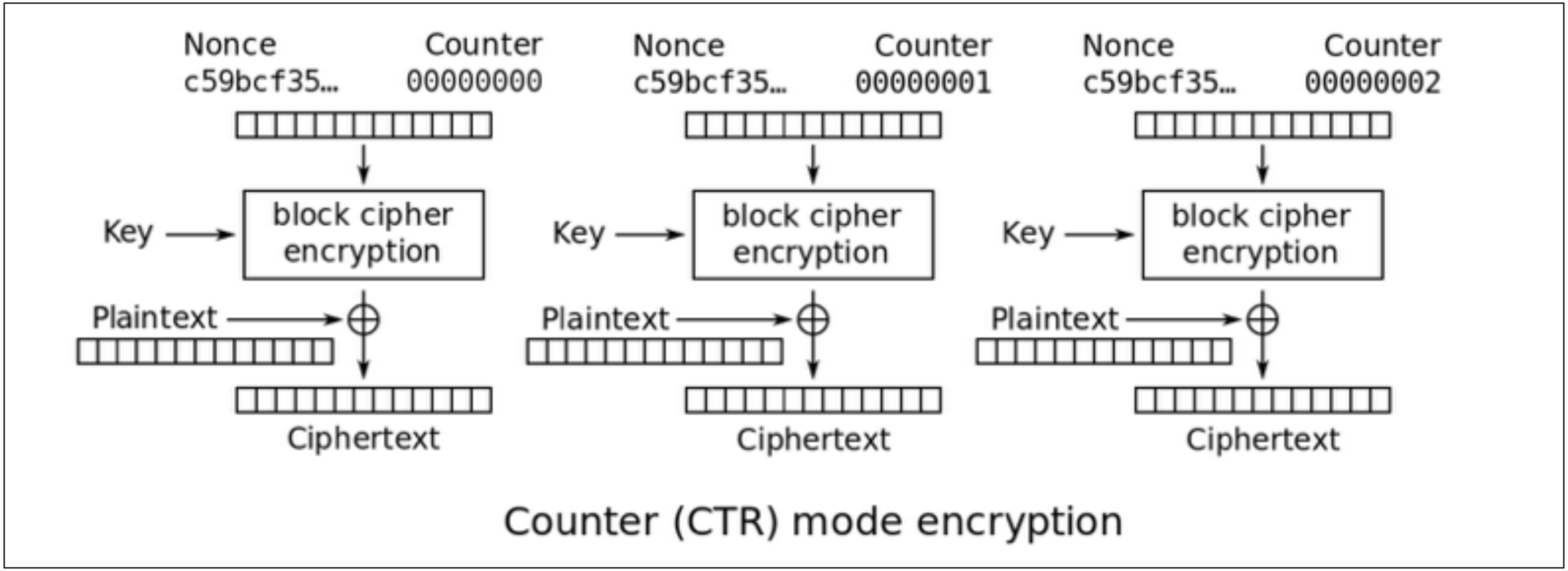
5.1.5 Counter Mode (CTR)

Counter mode (CTR)

- It uses a block cipher as a **stream cipher** (like the OFB and CFB modes)
- The key stream is computed in a blockwise fashion
- The input to the block cipher is a counter which assumes a different value every time the block cipher computes a new key stream block
- Unlike CFB and OFB modes, the CTR mode can be parallelized since the 2nd encryption can begin before the 1st one has finished
    - Desirable for high-speed implementations, e.g., in network routers

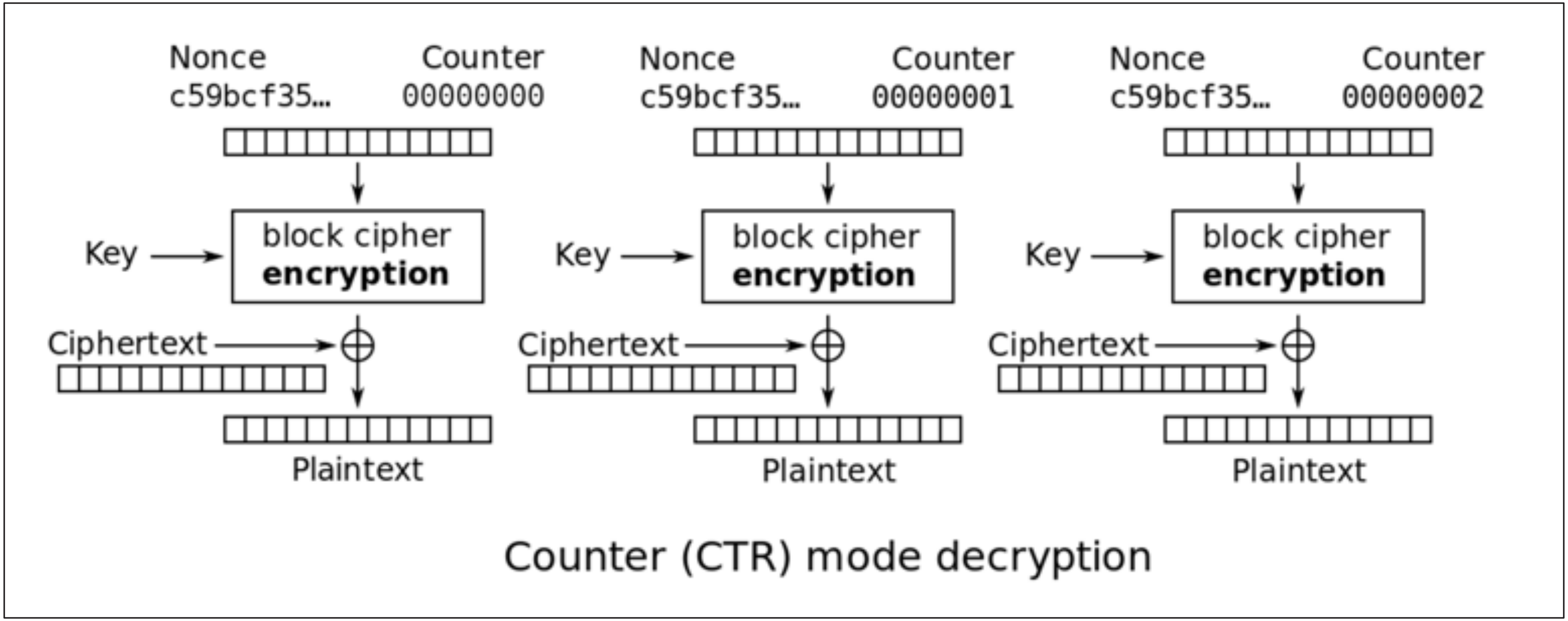
Counter mode (CTR)

- Image from Wikipedia (Link Ch 5a)



Counter mode (CTR)

- Image from Wikipedia (Link Ch 5a)



Kahoot!

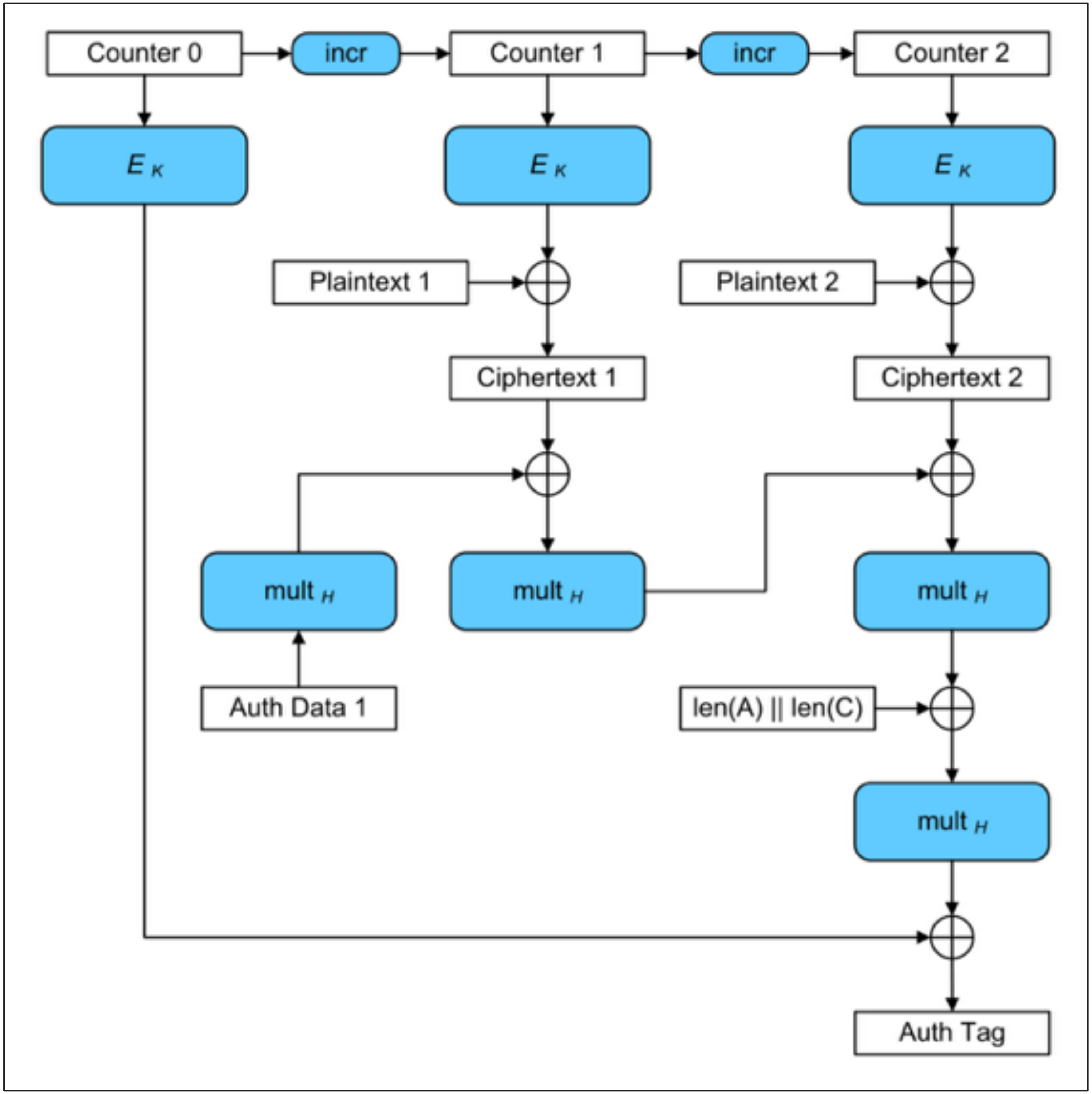
5.1.6 Galois Counter Mode (GCM)

Galois Counter Mode (GCM)

- Encrypts data in CTR mode, but also computes a **Message Authentication Code (MAC)**
- By making use of GCM, two additional services are provided:
  - Message Authentication
    - the receiver can make sure that the message was really created by the original sender
  - Message Integrity
    - the receiver can make sure that nobody tampered with the ciphertext during transmission

Galois Counter Mode (GCM)

- Image from Wikipedia (Link Ch 5b)



Chapter 5 of *Understanding Cryptography* by Christof Paar and Jan Pelzl

5.2 Exhaustive Key Search Revisited

Exhaustive Key Search Revisited

- For DES, a 56-bit key encrypts a 64-bit block
  - Only one key can decrypt a block
- In AES, a 128-bit or longer key encrypts a 128-bit block
  - Only one key can decrypt a block
- If a cipher has a longer block size than key size, there's more than one key that deciphers that block
- So several blocks must be tested to find the correct key

5.3 Increasing the Security of Block Ciphers

Increasing the Security of Block Ciphers

- In some situations we wish to increase the security of block ciphers
  - e.g., if a cipher such as DES is available in hardware or software for legacy reasons in a given application
- For AES, there are already three security levels
  - 128, 192, or 256-bit keys
  - No realistic attacks known for any of those levels
  - No reason to increase the security with these methods

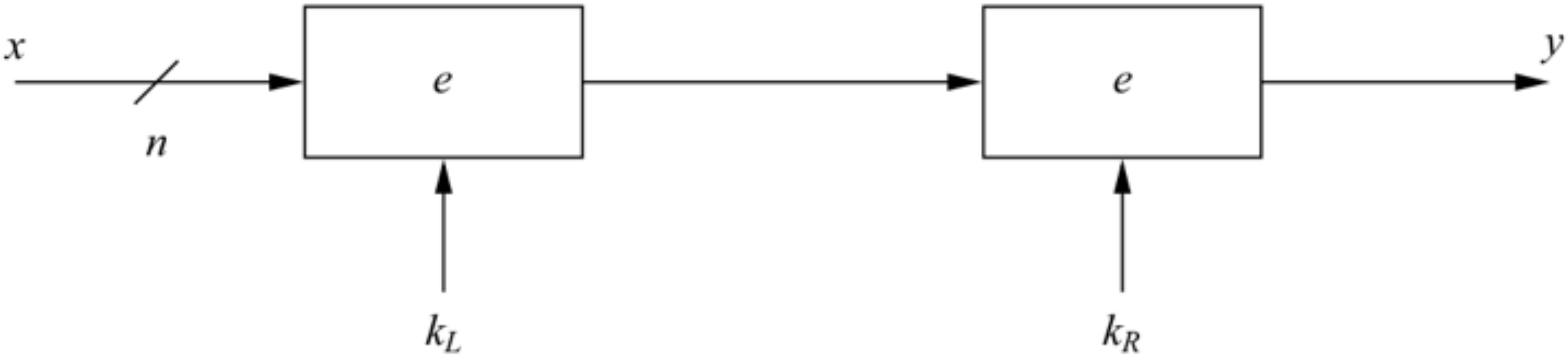
Increasing the Security of Block Ciphers

- Two approaches are possible
  - Multiple encryption
    - theoretically much more secure, but **sometimes** in practice increases the security very little
  - Key whitening
    - Adding two additional keys

5.3.1 Double Encryption and Meet-in-the-Middle Attack

Double Encryption

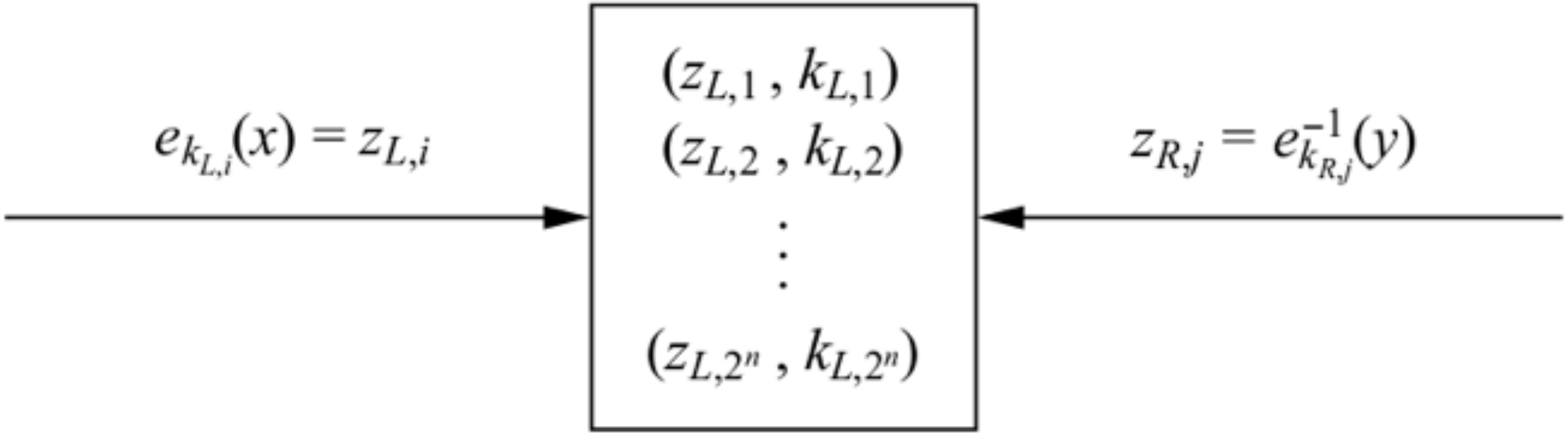
- A plaintext x is first encrypted with a key k_L
- and the resulting ciphertext is encrypted again using a second key k_R



- Assuming a key length of k bits, an exhaustive key search would require $2^k \cdot 2^k = 2^{2k}$ encryptions or decryptions

Meet-in-the-Middle Attack

- A Meet-in-the-Middle attack requires only $2^k + 2^k = 2^{k+1}$ operations!
- It also requires 2^k records of data storage for a look-up table



- **Double encryption is not much more secure then single encryption!**

Meet-in-the-Middle Attack

- **Phase I**
  - Brute-force the left half
  - Save a table of middle values for each k_L
- **Phase II**
  - Brute-force the right half
  - Find the k_R value that matchs one of the middle values; that determines k_L

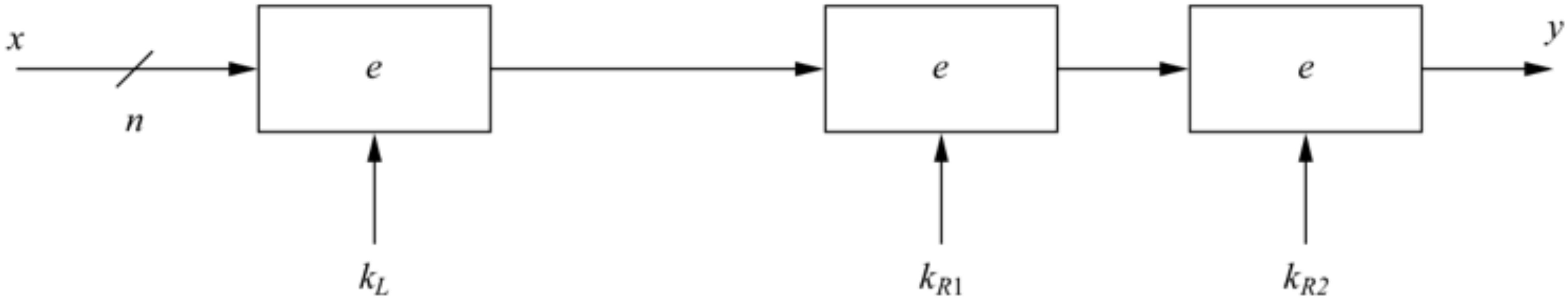
number of encryptions and decryptions = $2^k + 2^k = 2^{k+1}$
number of storage locations = 2^k

- **Double encryption is not much more secure than single encryption!**

5.3.2 Triple Encryption

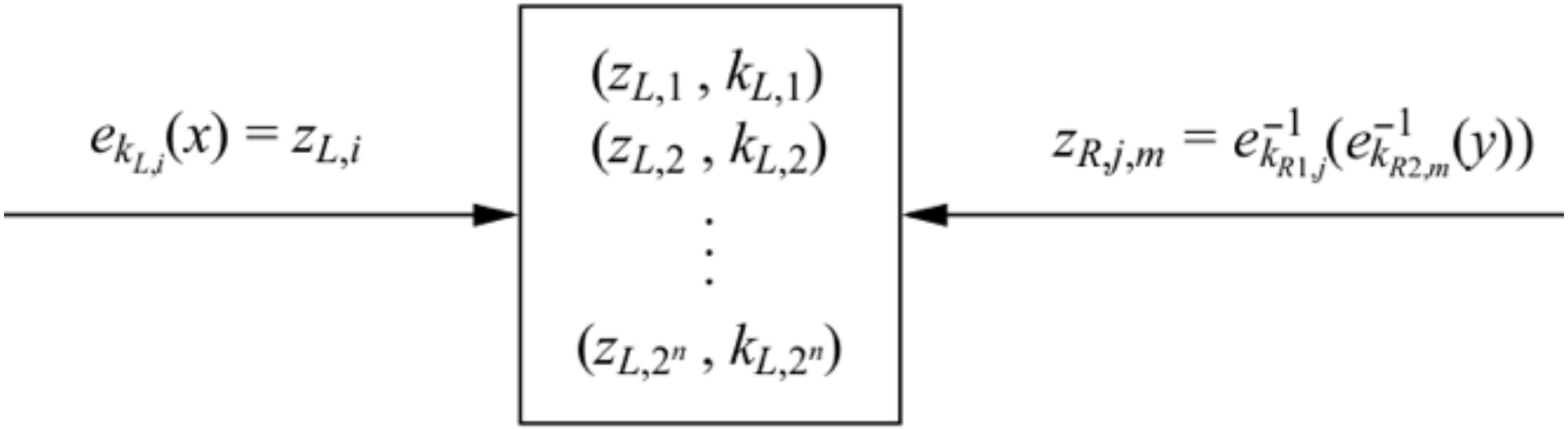
Triple Encryption

- Encrypt a block three times with three different keys



Triple Encryption

- Meet-in-the-middle attack has one side with k_L
- The other side has k_{R1} and k_{R2}

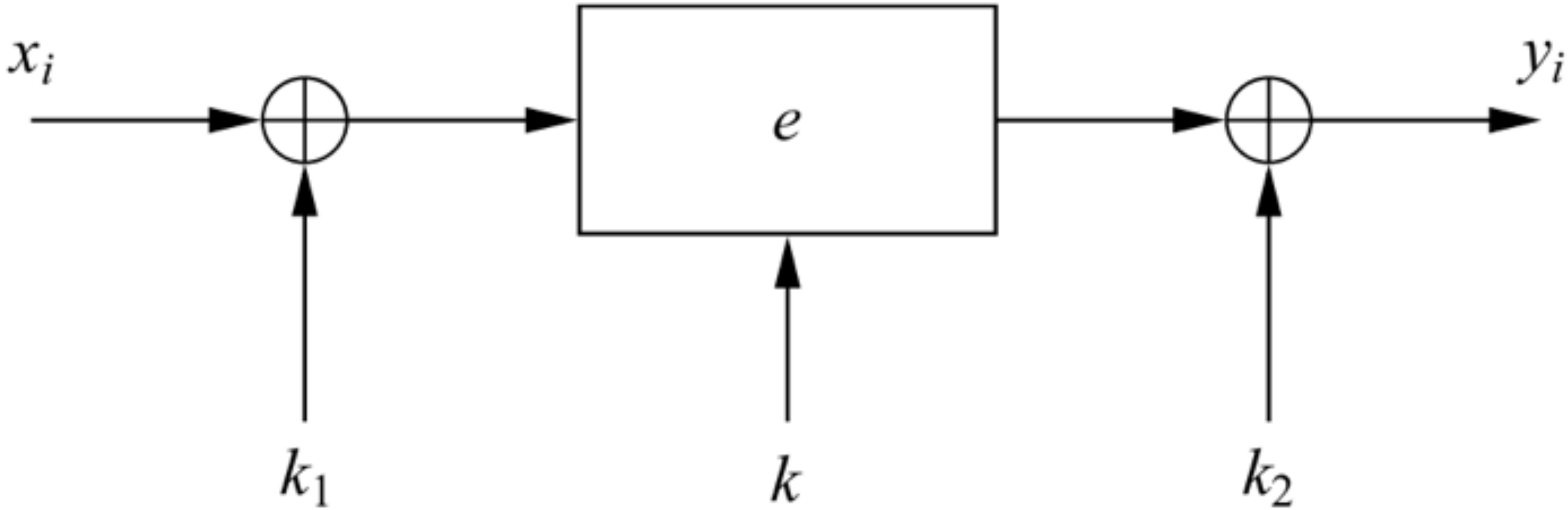


Triple encryption effectively doubles the key length

5.3.3 Key Whitening

Key Whitening

- Makes block ciphers such as DES much more resistant against brute-force attacks
- In addition to the regular cipher key k, two whitening keys k_1 and k_2 are used to XOR-mask the plaintext and ciphertext



Chapter 5 of *Understanding Cryptography* by Christof Paar and Jan Pelzl

Key Whitening

- DESX uses key whitening to make DES stronger
  - In addition to the regular cipher key k, adds a whitening key k_1
  - k_2 is calculated from key k and k_1
  - Even advanced attacks still take 2^{88} calculations
- AES already includes key whitening
  - Using a subkey before the first round and after the last round

Quantum Computers

- Can crack a 128-bit key with only 2^{64} calculations (Grover's algorithm)
  - This is why AES has 192-bit and 256-bit modes
  - They should still be unbreakable even when quantum computers become available
  - **AES will remain secure**
- Factoring a number becomes MUCH faster
  - Exponential time changes to polynomial time (Schor's algorithm, link Ch 5c)
  - Algorithms like **RSA may become insecure**, even for long keys

Lessons Learned

- There are many different ways to encrypt with a block cipher. Each mode of operation has some advantages and disadvantages
- Several modes turn a block cipher into a stream cipher
- There are modes that perform encryption together together with authentication, i.e., a cryptographic checksum protects against message manipulation
- The straightforward ECB mode has security weaknesses, independent of the underlying block cipher
- The counter mode allows parallelization of encryption and is thus suited for high speed implementations
- Double encryption with a given block cipher only marginally improves the resistance against brute-force attacks
- Triple encryption with a given block cipher roughly *doubles* the key length
- Triple DES (3DES) has an effective key length of 112 bits
- Key whitening enlarges the DES key length without much computational overhead.

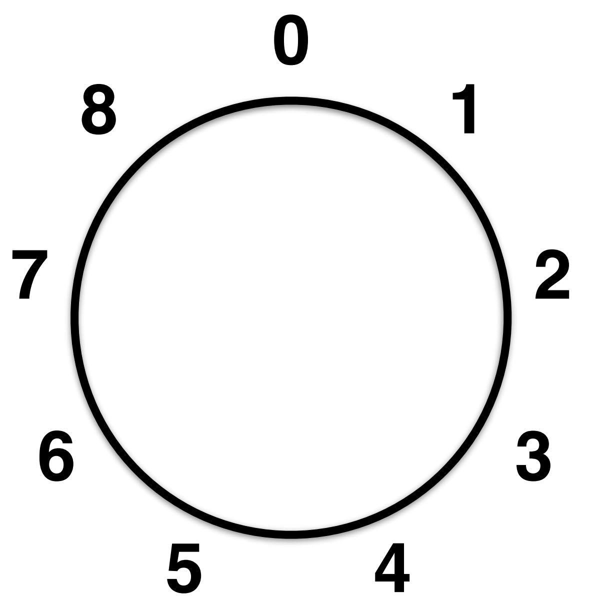
Kahoot!

Modular Arithmetic

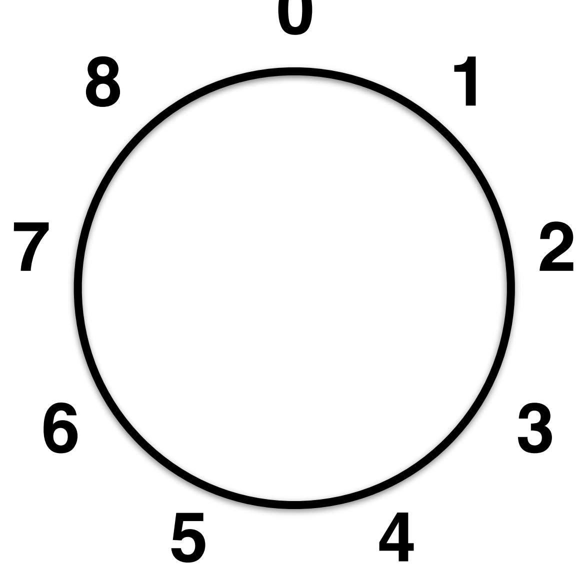
Multiplication and Multiplicative Inverses

Modulus 9

0 mod 9 = 0	9 mod 9 = 0
1 mod 9 = 1	10 mod 9 = 1
2 mod 9 = 2	11 mod 9 = 2
3 mod 9 = 3	12 mod 9 = 3
4 mod 9 = 4	13 mod 9 = 4
5 mod 9 = 5	14 mod 9 = 5
6 mod 9 = 6	15 mod 9 = 6
7 mod 9 = 7	16 mod 9 = 7
8 mod 9 = 8	17 mod 9 = 8



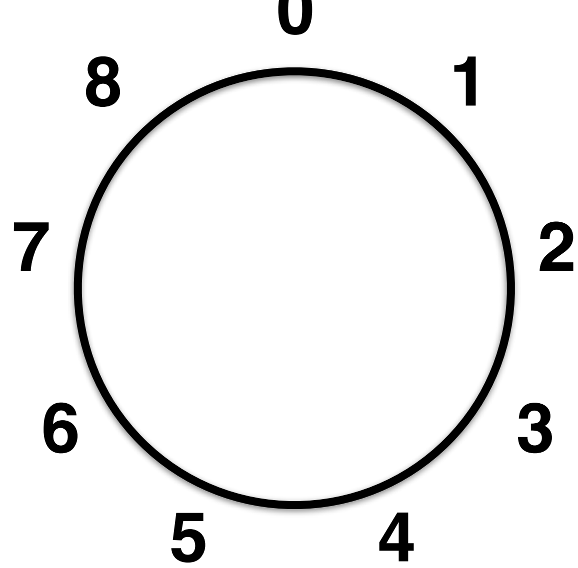
Modulus 9



2 * 9 mod 9 = (2 mod 9) * (0 mod 9) = 2 * 0 = 0

2 * 10 mod 9 = (2 mod 9) * (10 mod 9) = 2 * 1 = 2

Modulus 9



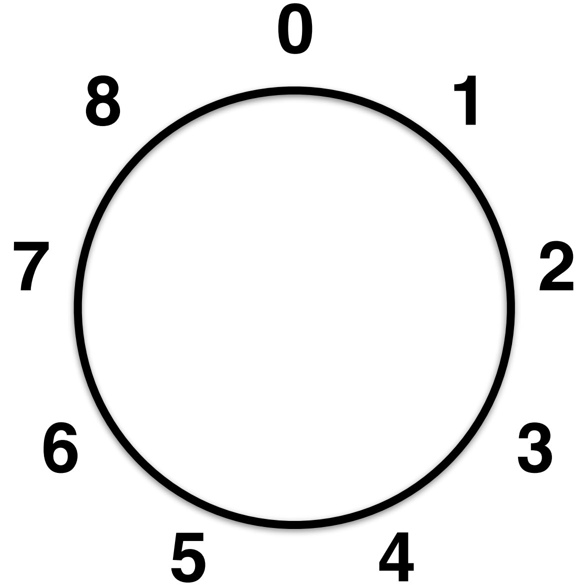
2 * 9 mod 9 = (2 mod 9) * (0 mod 9) = 2 * 0 = 0

2 * 10 mod 9 = (2 mod 9) * (10 mod 9) = 2 * 1 = 2

2 * 5 mod 9 = (2 mod 9) * (5 mod 9) = 2 * 5 = 10 = 1

5 is the *multiplicative inverse* of 2, mod 9

Modulus 9



6 * 1 mod 9 = 6

6 * 2 mod 9 = 3

6 * 3 mod 9 = 0

6 * 4 mod 9 = 6

6 * 5 mod 9 = 3

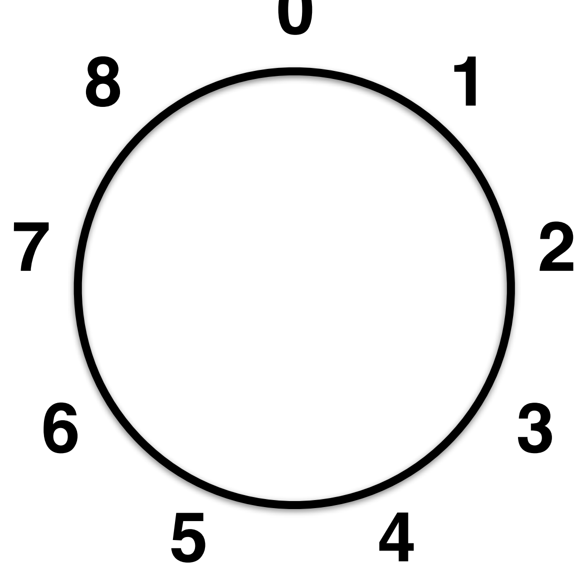
6 * 6 mod 9 = 0

6 * 7 mod 9 = 6

6 * 8 mod 9 = 3

6 has no inverse mod 9

Modulus 9



7 * 1 mod 9 = 7

7 * 2 mod 9 = 5

7 * 3 mod 9 = 3

7 * 4 mod 9 = 1

4 is the multiplicative inverse of 7 mod 9

Modulus 4

0 mod 4 = 0

1 mod 4 = 1

2 mod 4 = 2

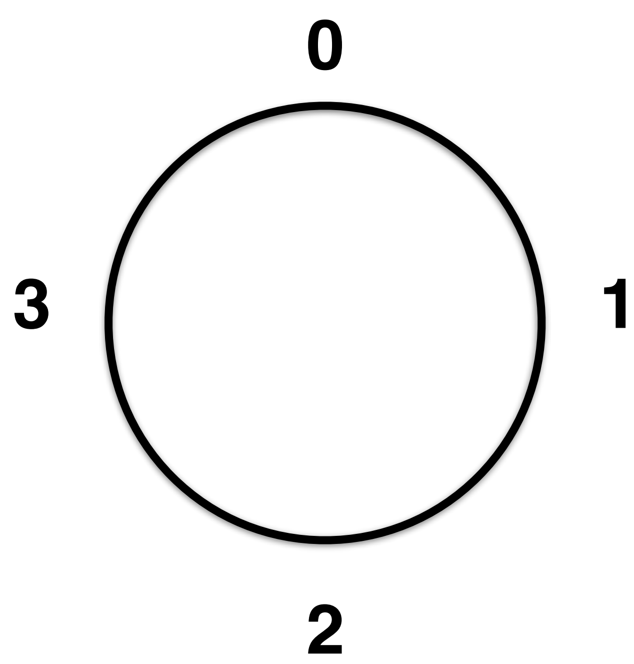
3 mod 4 = 3

4 mod 4 = 0

5 mod 4 = 1

12 mod 4 = 0

13 mod 4 = 1



Kahoot!