Computer Security and Privacy (COM-301)

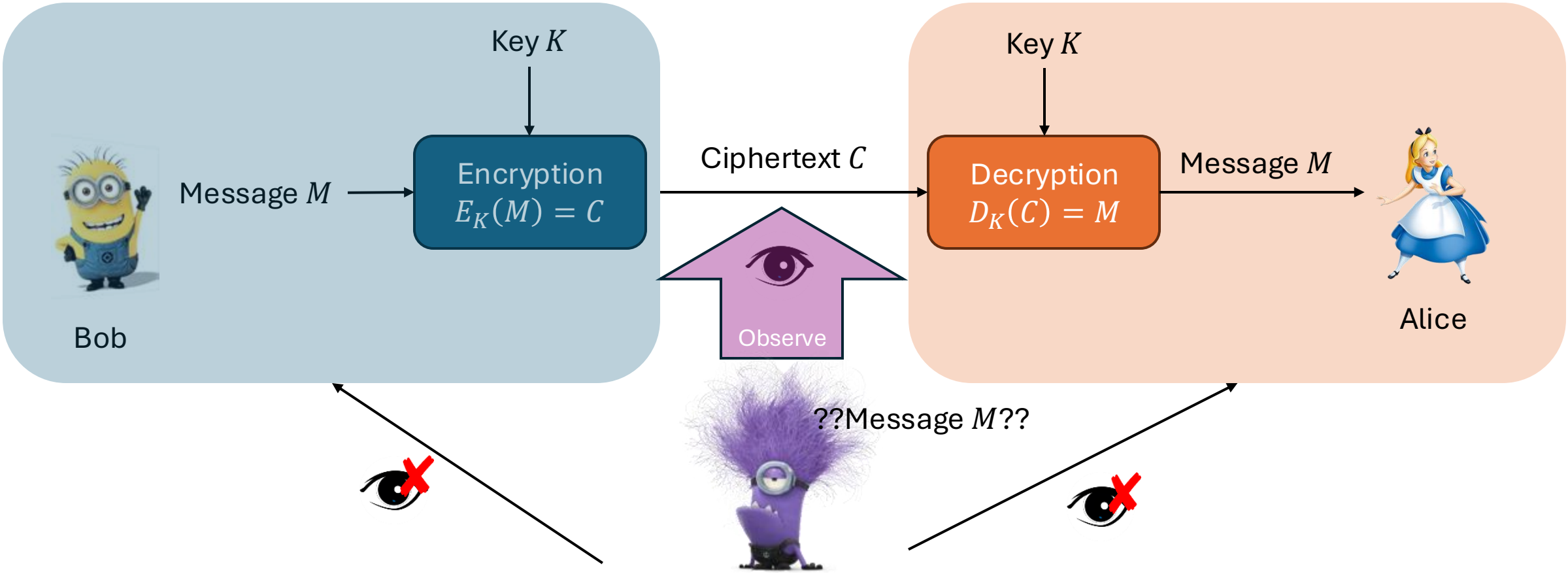
Week 6: Applied Cryptography Continued

Theresa Stadler

Some slides/concepts adapted from: Carmela Troncoso, George Danezis

Recap: Last week

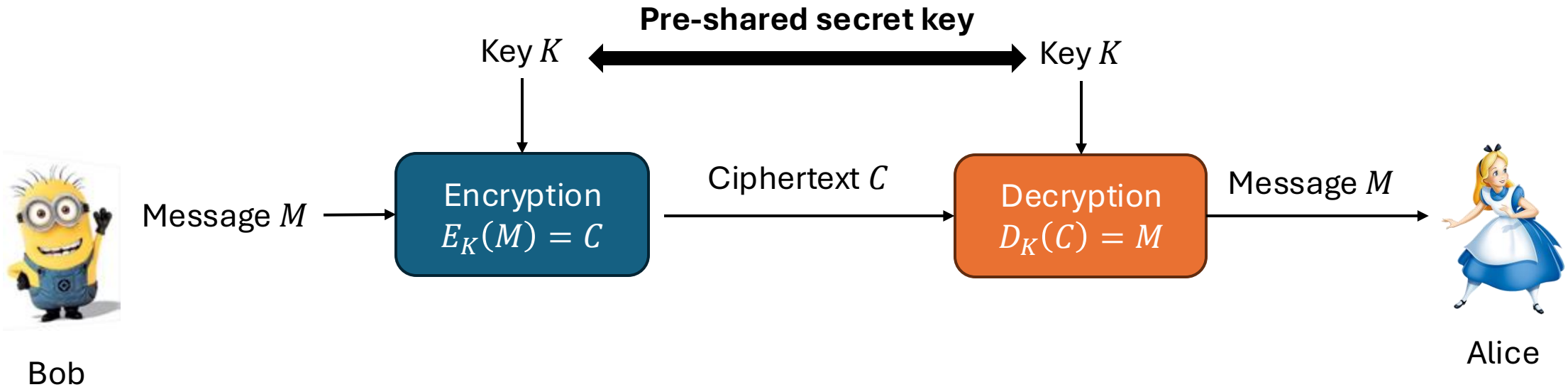
Confidentiality: information cannot be accessed by unauthorized parties



Assumption: Adversary Eve cannot see inside the boxes

Recap: Last week

Symmetric encryption schemes: Encryption and decryption use the same key



Recap: Last Week

Diffie-Hellman key exchange

Shared **public** parameters p , g

Because of the discrete logarithm hardness, an adversary observing these values cannot recover a and b

Bob's secret: b

Bob's public value: $B = g^b (mod\ p)$

Alice's public value: $A = g^a (mod\ p)$

Alice's secret: a

Recap: Last Week

Diffie-Hellman key exchange

Shared **public** parameters p , g

Because of the discrete logarithm hardness, an adversary observing these values cannot recover a and b

Shared key K:
$A^b = \boldsymbol{g^{ab}} \ (\boldsymbol{mod\ p})$

Bob's secret: b

Bob's public value: $B = g^b (mod\ p)$

Alice's public value: $A = g^a (mod\ p)$

Shared key K:
$B^a = \boldsymbol{g^{ab}} \ (\boldsymbol{mod\ p})$

Alice's secret: a

Recap: Last week

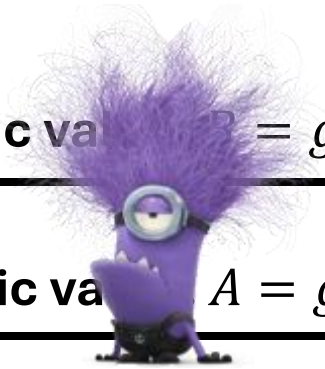
How do Alice and Bob know they are talking to the right person?

I want to talk to

d key K:
$\boldsymbol{mod\ p})$

Bob's public va $= g^b(mod\ p)$

Alice's public va $A = g^a(mod\ p)$



M(inion)ITM attacks

S
$B^a = \boldsymbol{g}$

I want to make sure I am talking to

Bob's secret: b

Alice's secret: a

Problem I: Authenticity

How do we verify the identity of the person to whom we are talking?

Bob's secret: b

...while avoiding Eve in the middle

Alice's secret: a

Public key cryptography

Public key infrastructure

Public key PK_{Bob}

Public key $\mathrm{PK}_{\text{Alice}}$

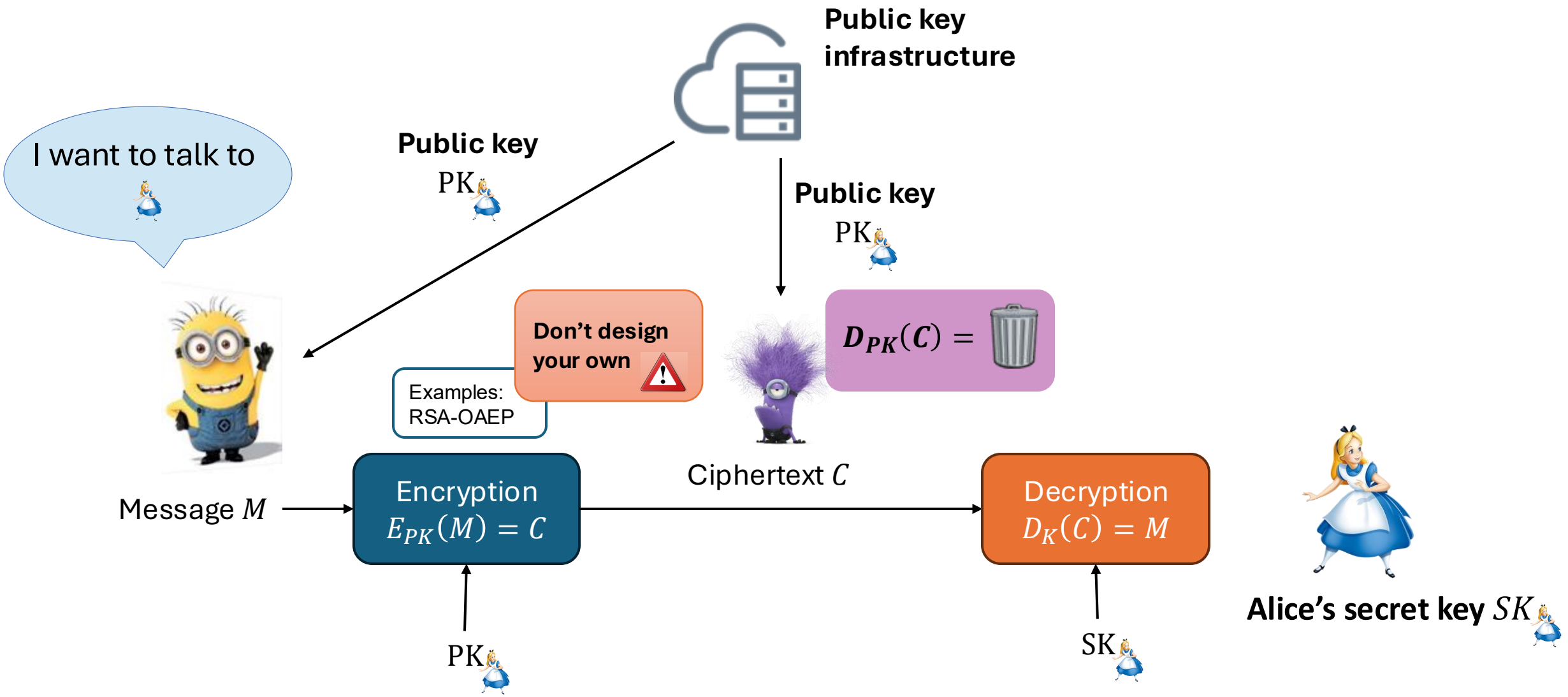
Bob's secret key $\mathrm{SK}_{\text{Bob}} : b$

Bob's public key $\mathrm{PK}_{\text{Bob}} : B = \mathrm{g}^{\mathrm{b}}$

Alice's secret key $SK_{\text{Alice}} : a$

Alice's public key $\mathrm{PK}_{\text{Alice}} : \mathrm{A} = \mathrm{g}^{\mathrm{a}}$

Confidentiality: Encryption & Decryption



Digital signatures: Signing & Verification

Public key infrastructure

Cannot forge a signature S on M that verifies without knowing SK

$$V_{PK}(M, Sign_{PK}(M)) = \times$$

Public key
PK

I want to make sure I am talking to

Message M

Signing
$Sign_{SK}(M) = S$

Signature S

Message M

Verification
$V_{PK}(M, S) =$ ✓

Bob's secret key SK : b

SK

Examples:
NIST DSA

Don't design your own

PK

Digital signatures

Computationally costly compared with most symmetric key algorithms of equivalent security

Signing and encrypting **is slow**

→ Sign **hash** of messages

Public key infrastructure

Public key

PK

Signing

$Sign_{SK}(M) = S$

Signature S

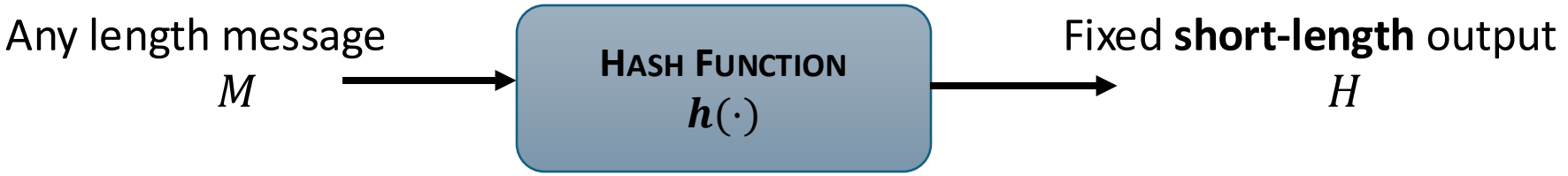
Verification

$V_{PK}(M, S) =$ ✅

SK

PK

Hash functions



THREE SECURITY PROPERTIES

PRE-IMAGE RESISTANCE

Given $\mathbf{H} = \mathbf{h}(\boldsymbol{M})$, difficult **to find** $\boldsymbol{M}$

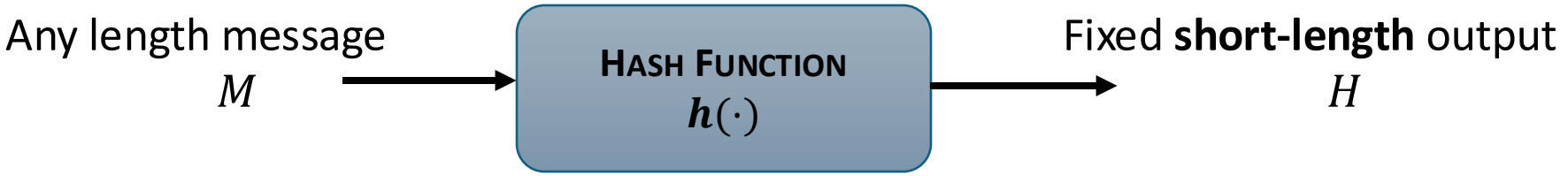
SECOND PRE-IMAGE RESISTANCE

Given $\boldsymbol{M}$, difficult **to find an** $\boldsymbol{M'} != M$ such that $\mathrm{h}(M') = h(M)$

COLLISION RESISTANCE

Difficult to find any M, M' such that $\mathrm{h}(M) = h(M')$

Hash functions



MD5 (1991): 128 bit hash – insecure

SHA0, SHA1: 160 bits – insecure

SHA-2 (224/256 /384/512) – OK but slow

SHA-3 (224/256 /384/512)

Don't design your own

THREE SECURITY PROPERTIES

PRE-IMAGE RESISTANCE

Given $\mathbf{H} = \mathbf{h}(\boldsymbol{M})$, difficult **to find** $\boldsymbol{M}$

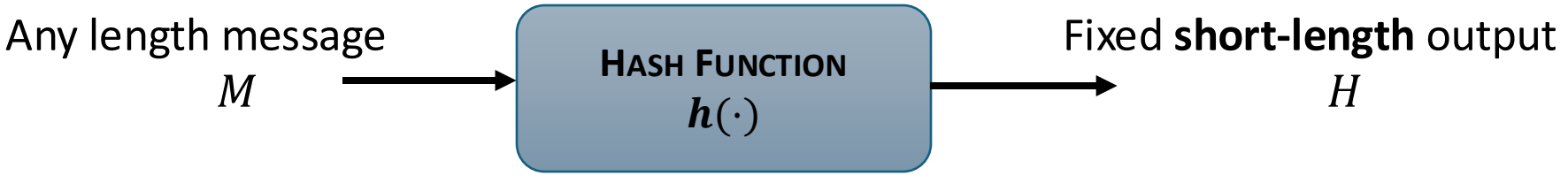
SECOND PRE-IMAGE RESISTANCE

Given $\boldsymbol{M}$, difficult **to find an** $\boldsymbol{M'} != M$ such that $\mathrm{h}(M') = h(M)$

COLLISION RESISTANCE

Difficult to find any M, M' such that $\mathrm{h}(M) = h(M')$

Hash functions



THREE SECURITY PROPERTIES

PRE-IMAGE RESISTANCE

Given $\mathbf{H} = \mathbf{h}(\boldsymbol{M})$, difficult **to find** $\boldsymbol{M}$

SECOND PRE-IMAGE RESISTANCE

Given $\boldsymbol{M}$, difficult **to find an** $\boldsymbol{M'} != M$ such that $\mathrm{h}(M') = h(M)$

COLLISION RESISTANCE

Difficult to find any M, M' such that $\mathrm{h}(M) = h(M')$

USES

Support digital signatures, build HMAC, password storage, file integrity, secure commitments, secure logging, blockchains, …

Digital signatures on hash functions

Public key infrastructure

Public key
PK

Hash of message
$H = h(M)$

Signing
$Sign_{SK}(H) = S$

SK

Signature S

Message M

Hash of message
$H = h(M)$

Verification
$V_{PK}(H, S) =$ ✅

PK

Digital signatures on hash functions

~~**Pre-image resistance**~~

~~**Given $H = h(M)$**, difficult **to find M**~~

Second pre-image resistance

Given M, difficult **to find an $M' != M$** such that $h(M') = h(M)$

Collision resistance

Difficult to find any M, M' such that $h(M) = h(M')$

Public key infrastructure

Public key

PK

Hash of message

$H = h(M)$

Signing

$Sign_{SK}(H) = S$

SK

Signature S

Message M

Hash of message

$H = h(M)$

Verification

$V_{PK}(H, S) =$ ✅

PK

And all together now…

Message confidentiality an authenticity for asymmetric cryptography

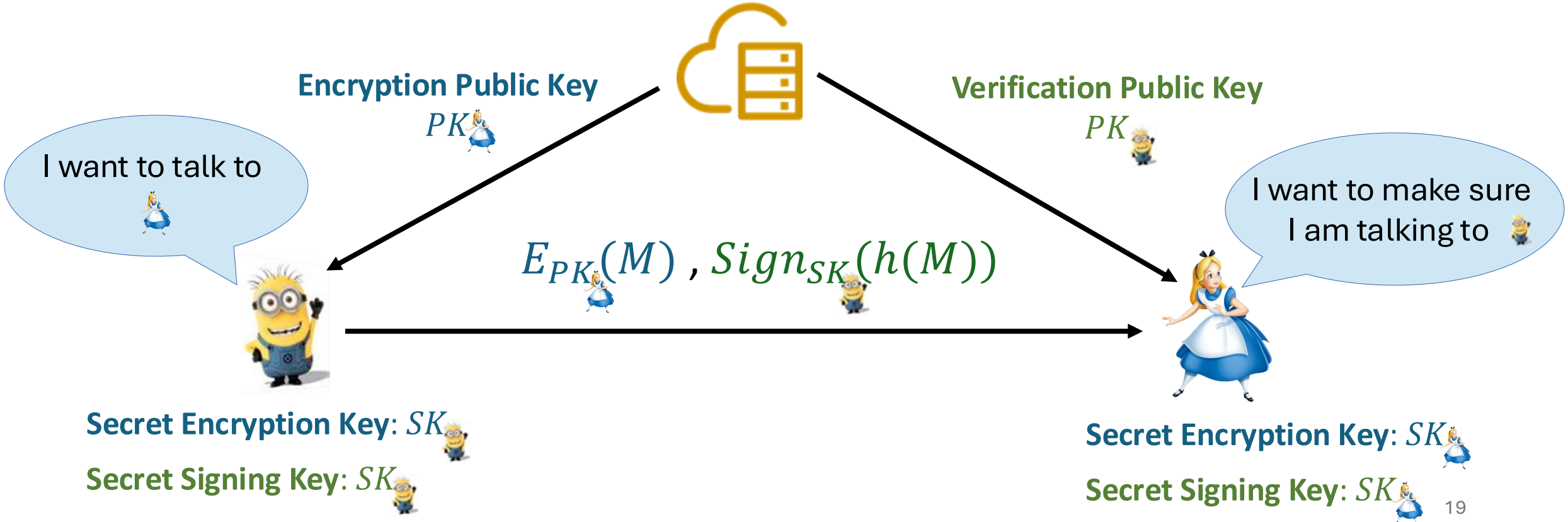
All together

ASYMMETRIC CRYPTOGRAPHY

Users have **TWO PAIRS** of keys (2x secret key, 2 x public key)

Confidentiality $D_{SK}(C) = M$ **with** $C = E_{PK}(M)$

Authentication $V_{PK}(M, S) =$ ✅ **with** $S = Sign_{SK}(M)$



Problem II: Integrity

Integrity

Integrity: information cannot be **modified** by unauthorized parties

Bob

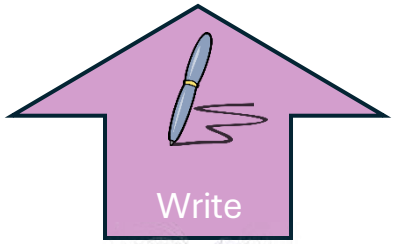
Message M

"Banana"

Key K

Encryption
$E_K(M) = C$

Ciphertext C



Key K

Decryption
$D_K(C) = M$

Message M

"Apple" "

Alice

Message Integrity through Digital Signatures

Digital signatures provide

- **Sender authenticity**:
- **Message integrity**
- **Non-repudiation**

Public key infrastructure

Public key
PK

I want to make sure Bob's message has not been modified

✅ Signature S only verifies if message M has not been modified

Signing
$Sign_{SK}(M) = S$

Signature S

Verification
$V_{PK}(M, S) =$ ✅

SK 

PK 

And for symmetric cryptography?

Bob

Message M

"Banana"

Pre-shared secret key

Key K

Key K

Encryption
$E_K(M) = C$

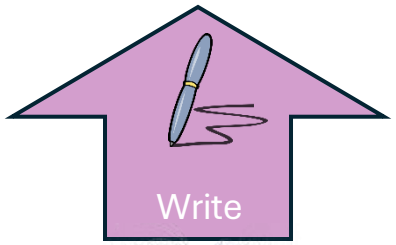
Ciphertext C

Decryption
$D_K(C) = M$

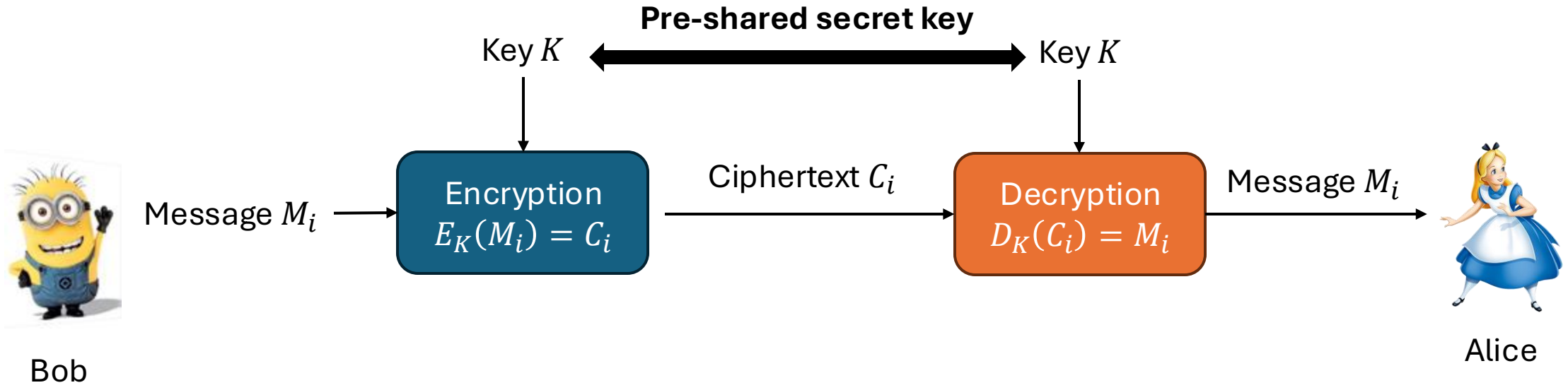
Message M

"Apple" "

Alice



Block cipher



The algorithms work on blocks the size of the key

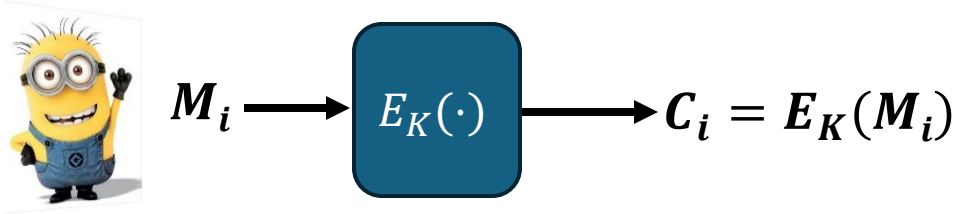
→Typically 128/256 bits

Problem: Messages are longer than a block!

→Requires iteration → Block ciphers have a "**Mode of Operation**"

Mode 1: Electronic Code Book (ECB)

Straightforward scheme: encrypt & decrypt single blocks

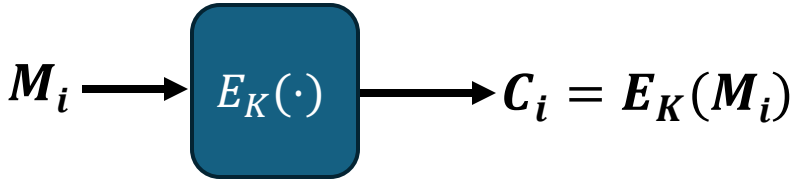


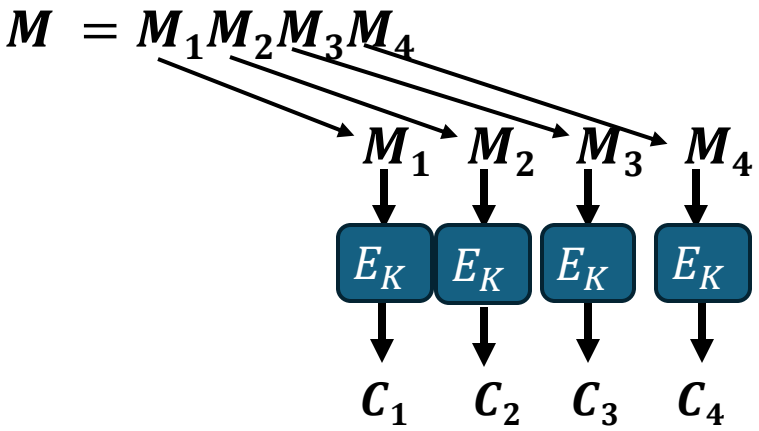
$$M = M_1 M_2 M_3 M_4$$

Each block M_i is the same size as the key

Mode 1: Electronic Code Book (ECB)

Straightforward scheme: encrypt & decrypt single blocks



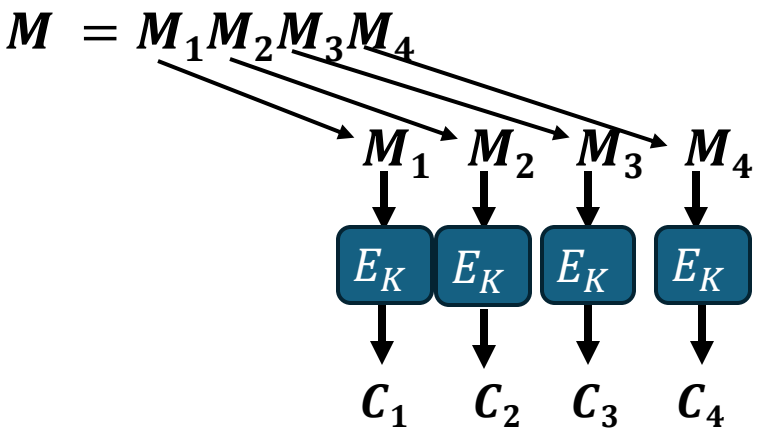


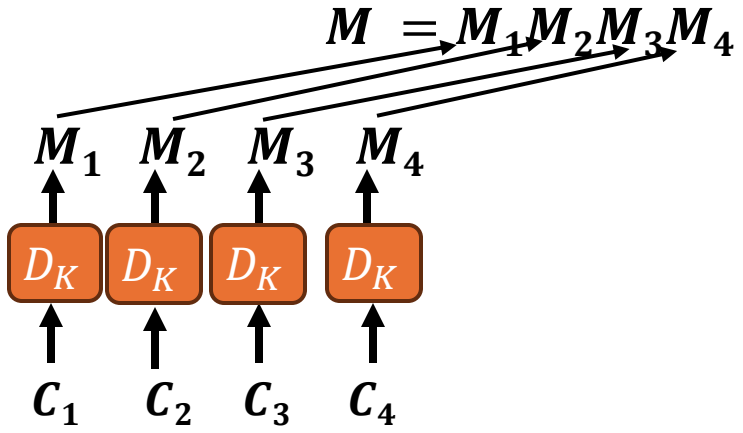
Mode 1: Electronic Code Book (ECB)

Straightforward scheme: encrypt & decrypt single blocks

$M_i \rightarrow E_K(\cdot) \rightarrow C_i = E_K(M_i)$

$C_i \rightarrow D_K(\cdot) \rightarrow M_i = D_K(C_i)$





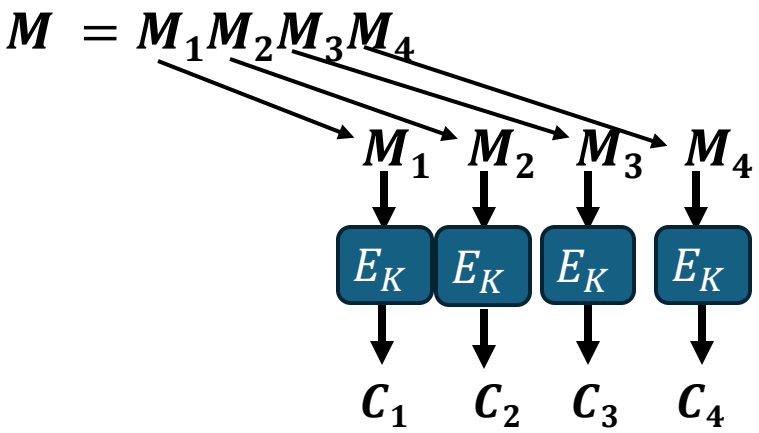
Can someone spot a problem?

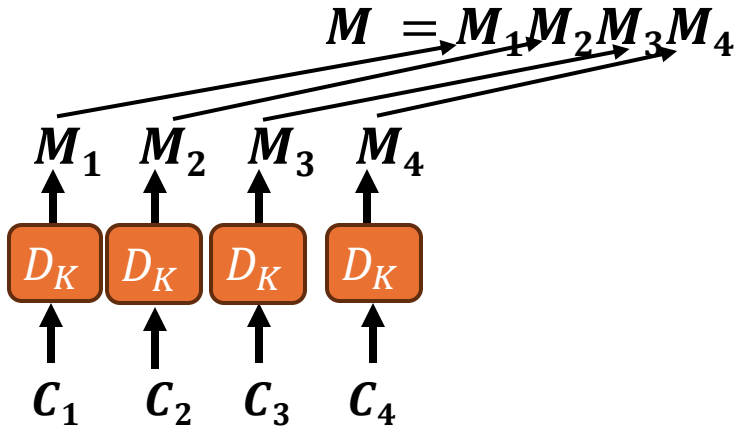
Mode 1: Electronic Code Book (ECB)

Straightforward scheme: encrypt & decrypt single blocks

$M_i \rightarrow E_K(\cdot) \rightarrow C_i = E_K(M_i)$

$C_i \rightarrow D_K(\cdot) \rightarrow M_i = D_K(C_i)$



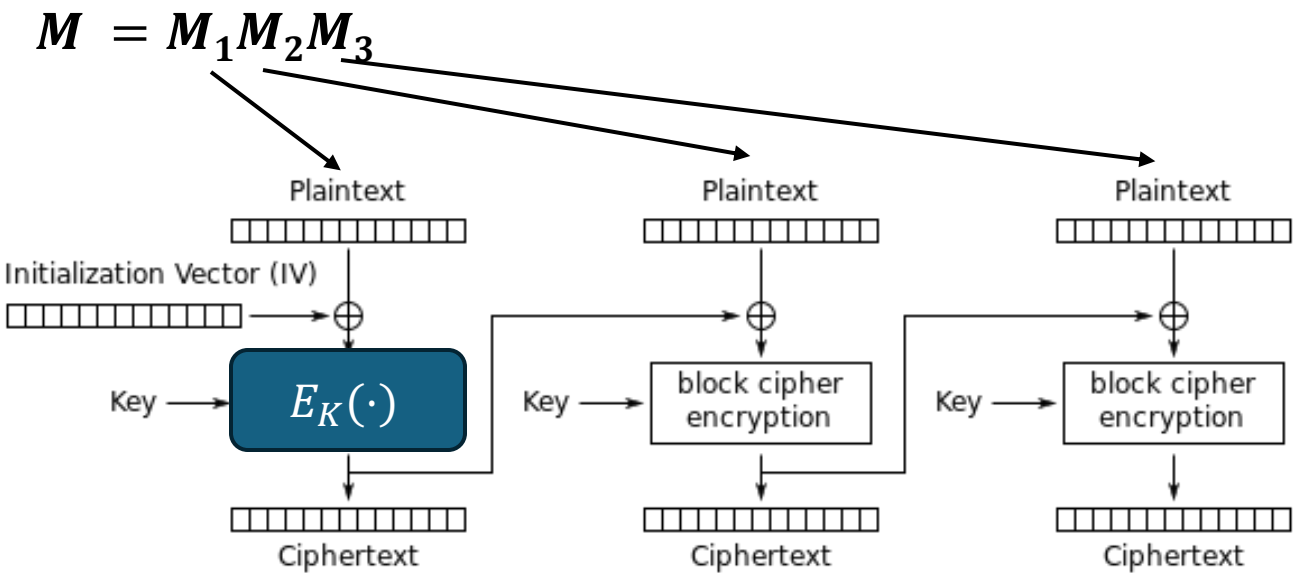


If $M_i = M_j$ then $C_i = C_j$

Mode 2: CIPHER BLOCK CHAINING (CBC)

Add IV and propagate information across blocks to introduce randomness

$M_i \rightarrow E_K(\cdot) \rightarrow C = E_K(M_i)$



CBC Encryption $E_K(\cdot)$

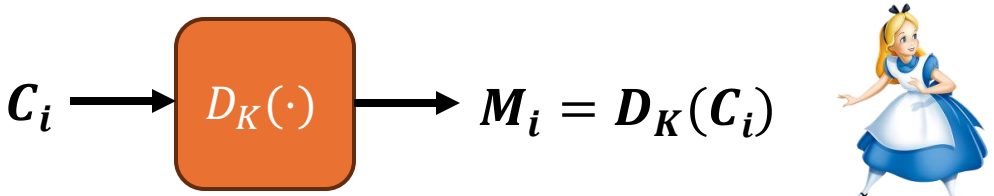
$$C_0 = IV$$

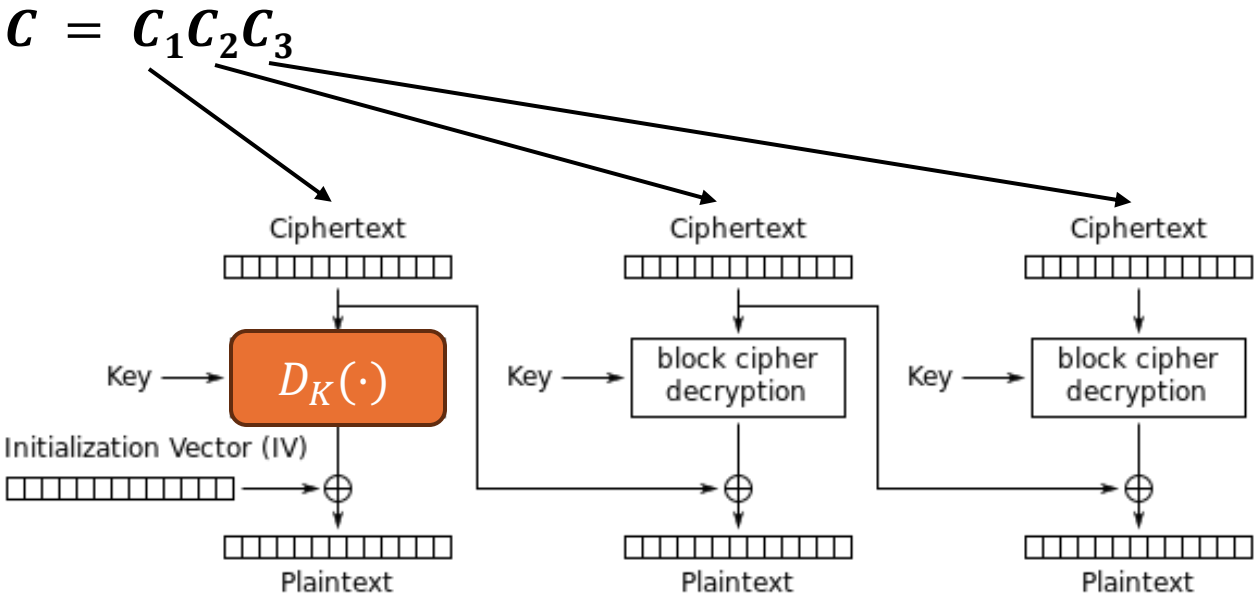
$$C_i = E_K(M_i \oplus C_{i-1})$$

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#/media/File:CBC_encryption.svg

Mode 2: Cipher Block Chaining (CBC)

Add IV and propagate information across blocks to introduce randomness





CBC Decryption $D_K(\cdot)$

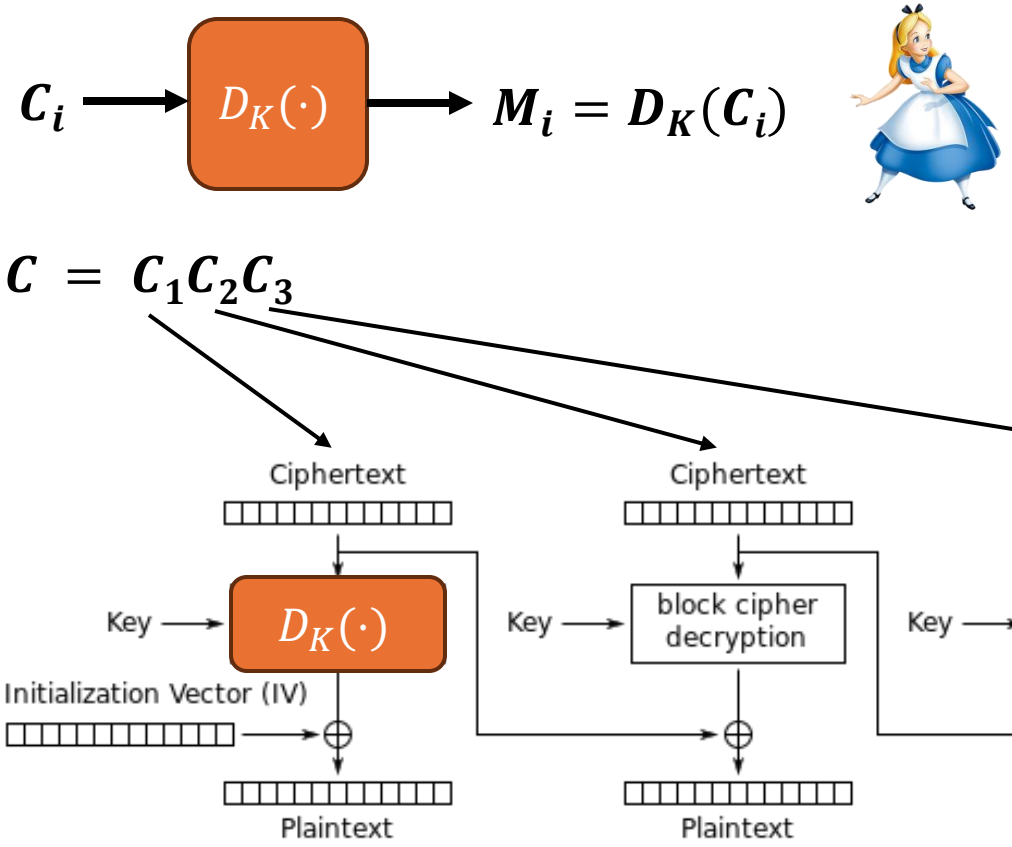
$$C_0 = IV$$

$$M_i = D_K(C_i) \oplus C_{i-1}$$

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#/media/File:CBC_encryption.svg

Mode 2: Cipher Block Chaining (CBC)

Add IV and propagate information across blocks to introduce randomness



CBC Decryption $D_K(\cdot)$

$$C_0 = IV$$

$$M_i = D_K(C_i) \oplus C_{i-1}$$

What if IV is incorrect? Is the full decryption wrong?

Can you decrypt a block alone? What do you need?

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#/media/File:CBC_encryption.svg

Mode 3: Counter Mode (CTR)

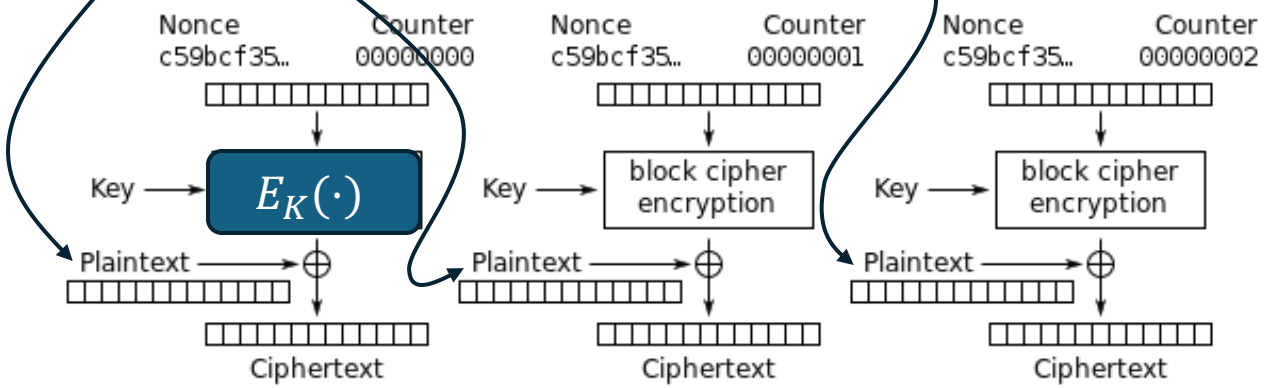
Use increasing nonce to add randomness without dependencies between blocks

$M_i \rightarrow E_K(\cdot) \rightarrow C = E_K(M_i)$

CTR Encryption $E_K(\cdot)$

$$C_i = E_K(Nonce + i) \oplus M_i$$

$M = M_1 M_2 M_3$



Counter

Mode 3: Counter Mode (CTR)

Use increasing nonce to add randomness without dependencies between blocks

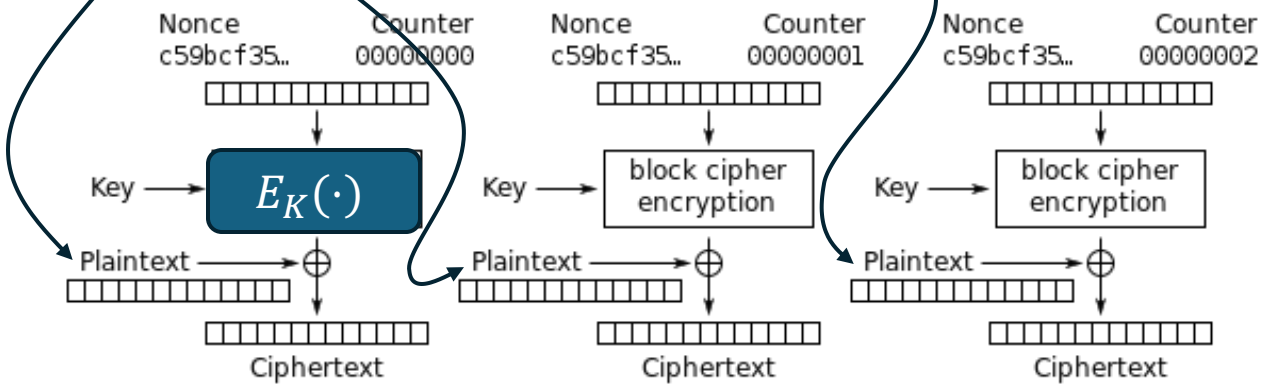
$M_i \rightarrow E_K(\cdot) \rightarrow C = E_K(M_i)$

Nonce: Random number used only **once**

CTR Encryption $E_K(\cdot)$

$$C_i = E_K(Nonce + i) \oplus M_i$$

$M = M_1 M_2 M_3$

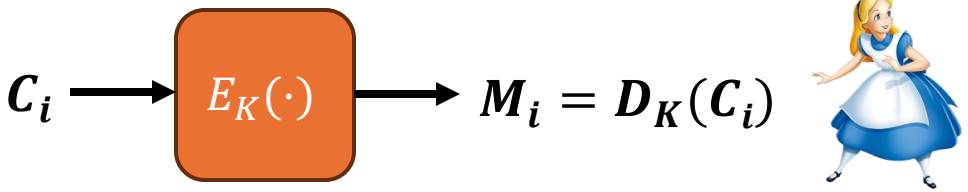


Counter

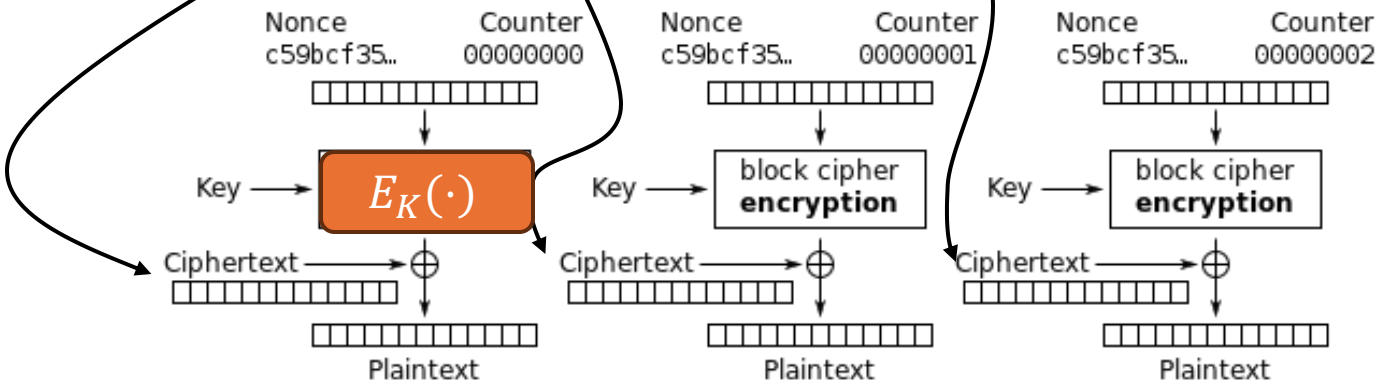
Do we need a decryption algorithm?

Mode 3: Counter Mode (CTR)

Use increasing nonce to add randomness without dependencies between blocks



$C = C_1 C_2 C_3$



CTR Decryption $E_K(\cdot)$

$$M_i = E_K(Nonce + i) \oplus C_i$$

Block ciphers

STRENGTHS

High diffusion: information from one plaintext symbol is diffused into several ciphertext symbols

Immunity to tampering: difficult to insert symbols without detection

WEAKNESSES

Slow: an entire block must be accumulated before encryption / decryption can begin

Error propagation: in some modes of operation errors affect several bits/blocks

*Individual modes may not fall in these categories and may offer a different trade-off

Modes of operation: summary

Electronic Code Book (**ECB**)

- Directly encrypt and decrypt single blocks
- Large information leakage due to lack of randomness across blocks' encryption

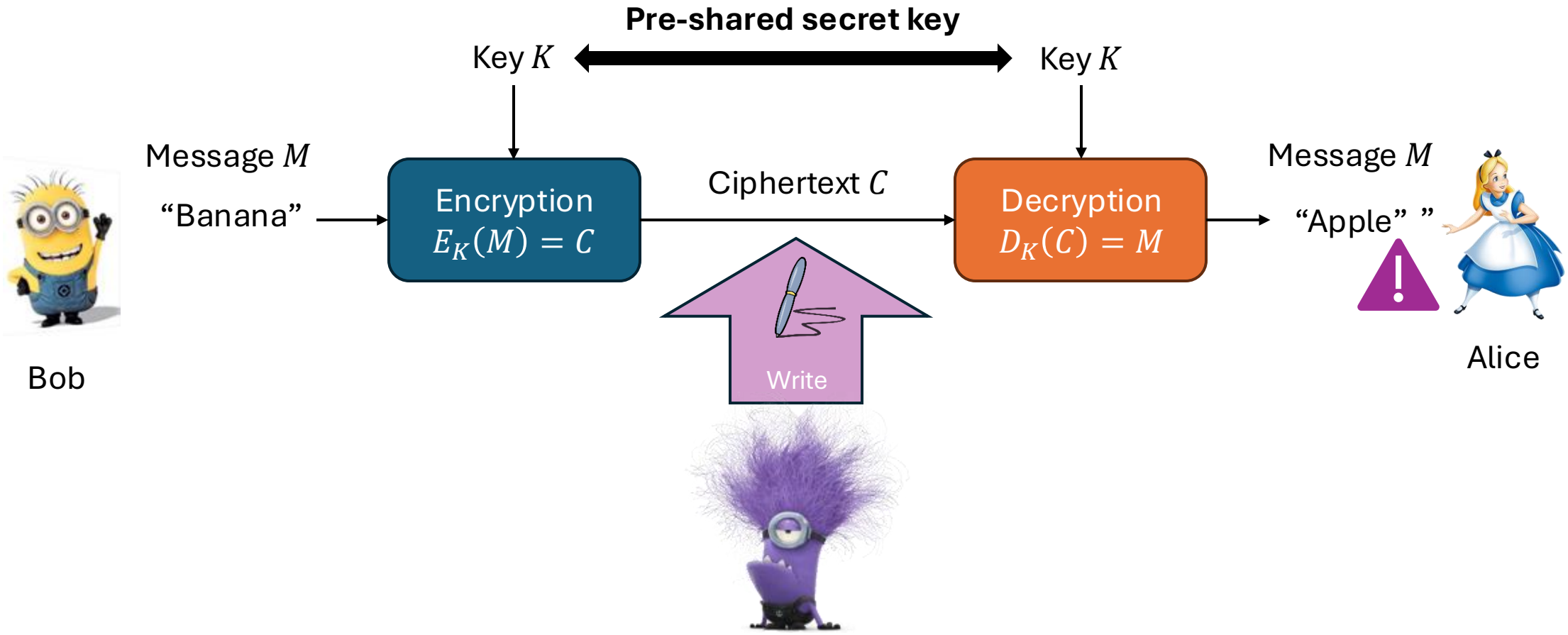
Cipher Block Chaining (**CBC**)

- Avoids ECB problems by using previous blocks to add randomness to every encryption
- Propagates errors and prevents single-block decryption

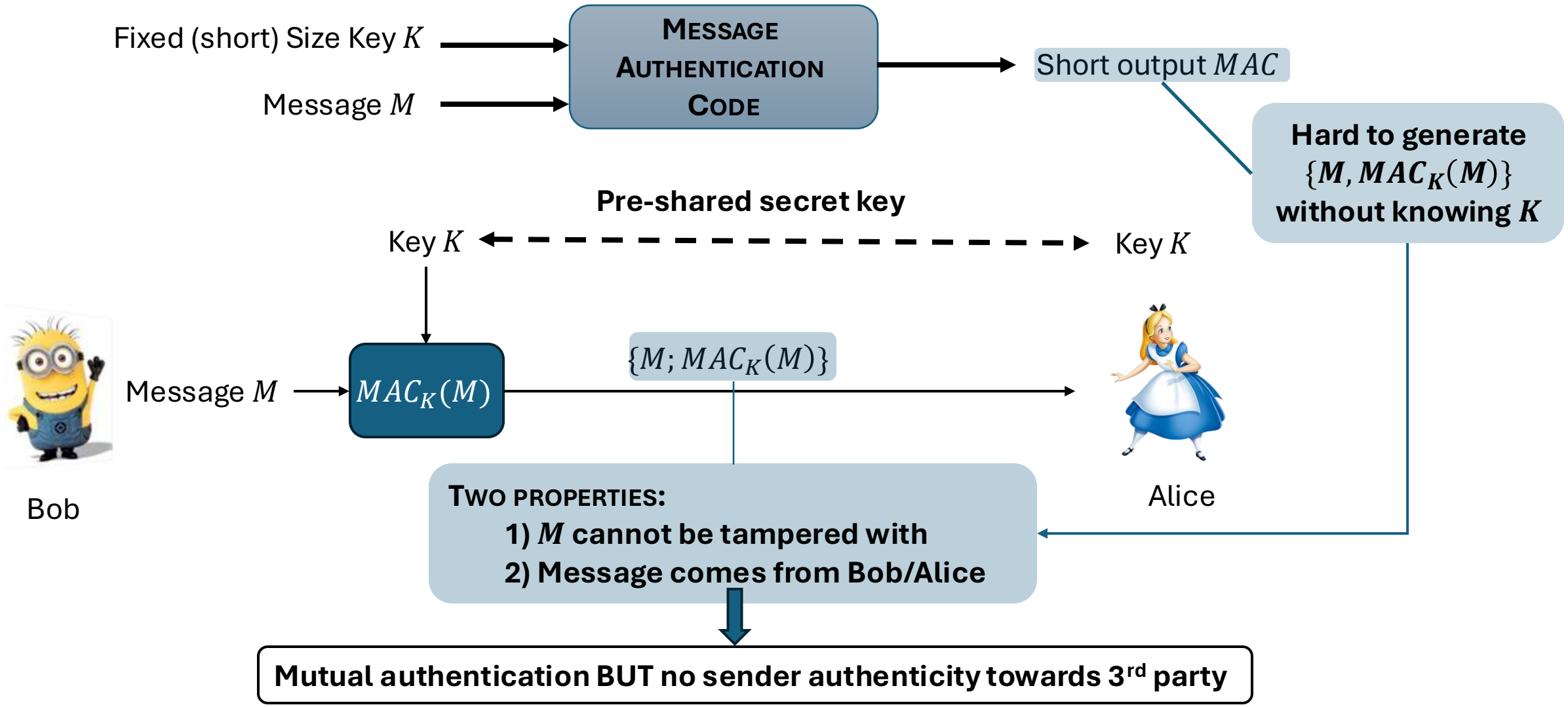
Counter mode (**CTR)**

- Uses an increasing "nonce" to introduce randomness across block's encryptions
- Enables parallel encryption and decryption

So what about integrity?



Integrity from symmetric encryption: Message Authentication Codes (MAC)



Repudiation

Charlie – 3rd Party

"I never wrote that message. It was Alice."

$\{M = \text{“}I\ don't\ like\ Charlie\text{”}\ ; MAC_K(M)\}$

"I never wrote that message. It was Bob."

Bob
Key K

Alice
Key K

Pre-shared secret key

Mutual authentication BUT no sender authenticity towards 3rd party

Example MAC: **CBC-MAC**

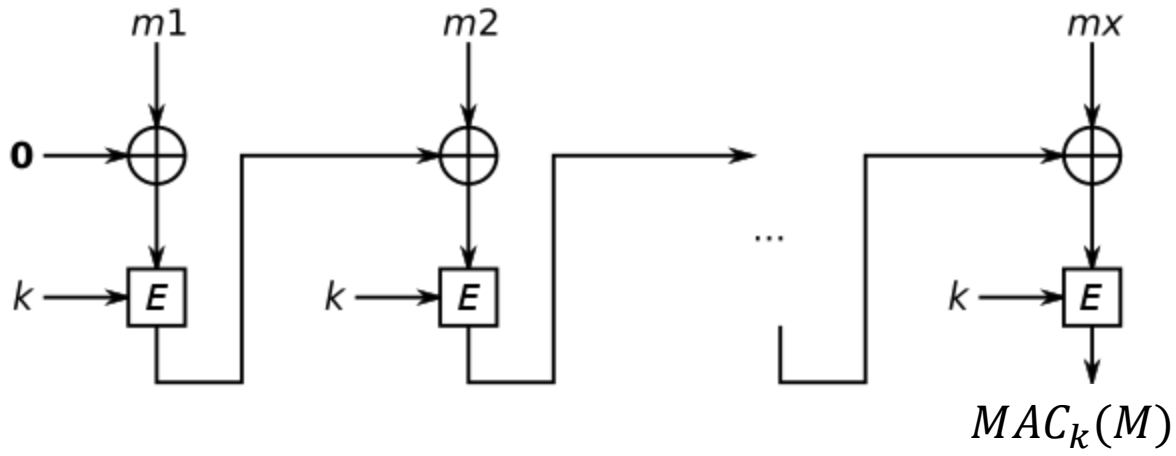
Turning a block cipher into a MAC

CBC MAC

$C_0 = 0$ [any fixed IV]

$C_i = E_K(M_i \oplus C_{i-1})$

$MAC_K(M_1 \dots M_x) = C_n$



Differences with respect to CBC Encryption:

- CBC-MAC is deterministic
- Only output of CBC-MAC is the last block

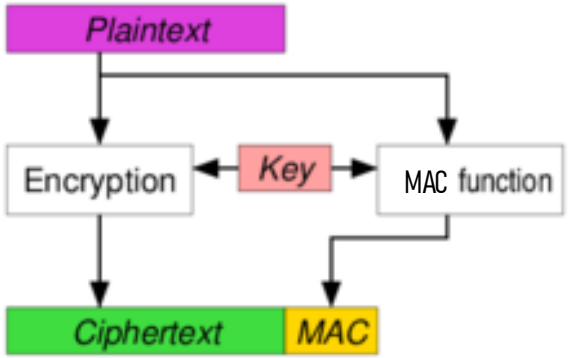
Limitation: Only secure if length of message M is known

And all together now…

Message confidentiality and integrity for symmetric encryption schemes

How to obtain confidentiality and integrity?

ENCRYPT-AND-MAC



✗ No integrity on the ciphertext → Cipher can be attacked need to decrypt to know if valid

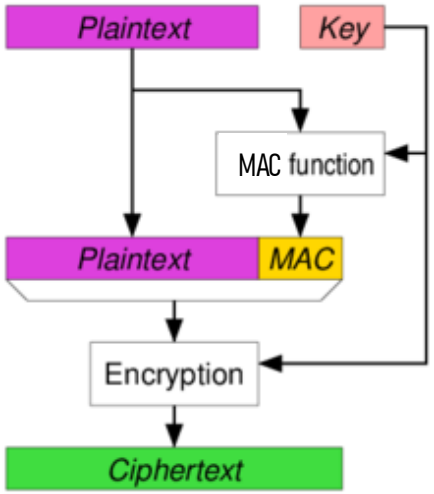
✓ Integrity of the plaintext can be verified

✗ May reveal information about the plaintext → repeated msg, recall the IV of the MAC is fixed (can be solved with a counter)

Bellare, M., & Namprempre, C. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *International Conference on the Theory and Application of Cryptology and Information Security*, 2000.
Bellare, M., Kohno, T., & Namprempre, C. Breaking and provably repairing the SSH authenticated encryption scheme: A case study of the Encode-then-Encrypt-and-MAC paradigm. *ACM Transactions on Information and System Security*, 2004.
https://en.wikipedia.org/wiki/Authenticated_encryption

How to obtain confidentiality and integrity?

MAC-THEN-ENCRYPT



✗ No integrity of ciphertext

(in theory) possible to change ciphertext and have a valid MAC

need to decrypt to know if valid

✓ Integrity of the plaintext can be verified

✓ No information on the plaintext either, since it is encrypted

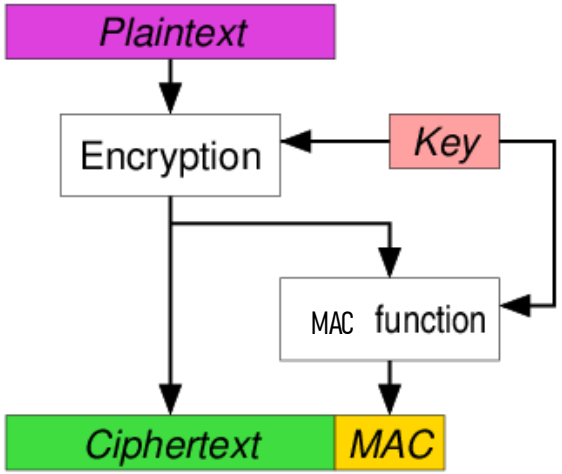
Bellare, M., & Namprempre, C. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *International Conference on the Theory and Application of Cryptology and Information Security*, 2000.

Bellare, M., Kohno, T., & Namprempre, C. Breaking and provably repairing the SSH authenticated encryption scheme: A case study of the Encode-then-Encrypt-and-MAC paradigm. *ACM Transactions on Information and System Security*, 2004.

https://en.wikipedia.org/wiki/Authenticated_encryption

How to obtain confidentiality and integrity?

ENCRYPT-THEN-MAC



- ✓ Integrity of ciphertext → ensures you only read valid messages! Cipher cannot be attacked!
- ✓ Integrity of the plaintext can be verified
- ✓ No information on the plaintext either, since it is encrypted

Bellare, M., & Namprempre, C. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *International Conference on the Theory and Application of Cryptology and Information Security*, 2000.
Bellare, M., Kohno, T., & Namprempre, C. Breaking and provably repairing the SSH authenticated encryption scheme: A case study of the Encode-then-Encrypt-and-MAC paradigm. *ACM Transactions on Information and System Security*, 2004.
https://en.wikipedia.org/wiki/Authenticated_encryption

Conclusions

What we have learned about applied cryptography

Symmetric cryptography

- Confidentiality: Stream ciphers, Block ciphers (modes of operation!)
- Integrity / Authentication: Message Authentication Codes (MACs)

Asymmetric cryptography

- Confidentiality: Encryption
- Integrity / Authentication: Digital signatures

Hash functions

- Three security properties
- Support Digital Signatures + other functions