CSE 484 / CSE M 584
Computer Security: Cryptography

TA: Adrian Sham
adrsham@cs

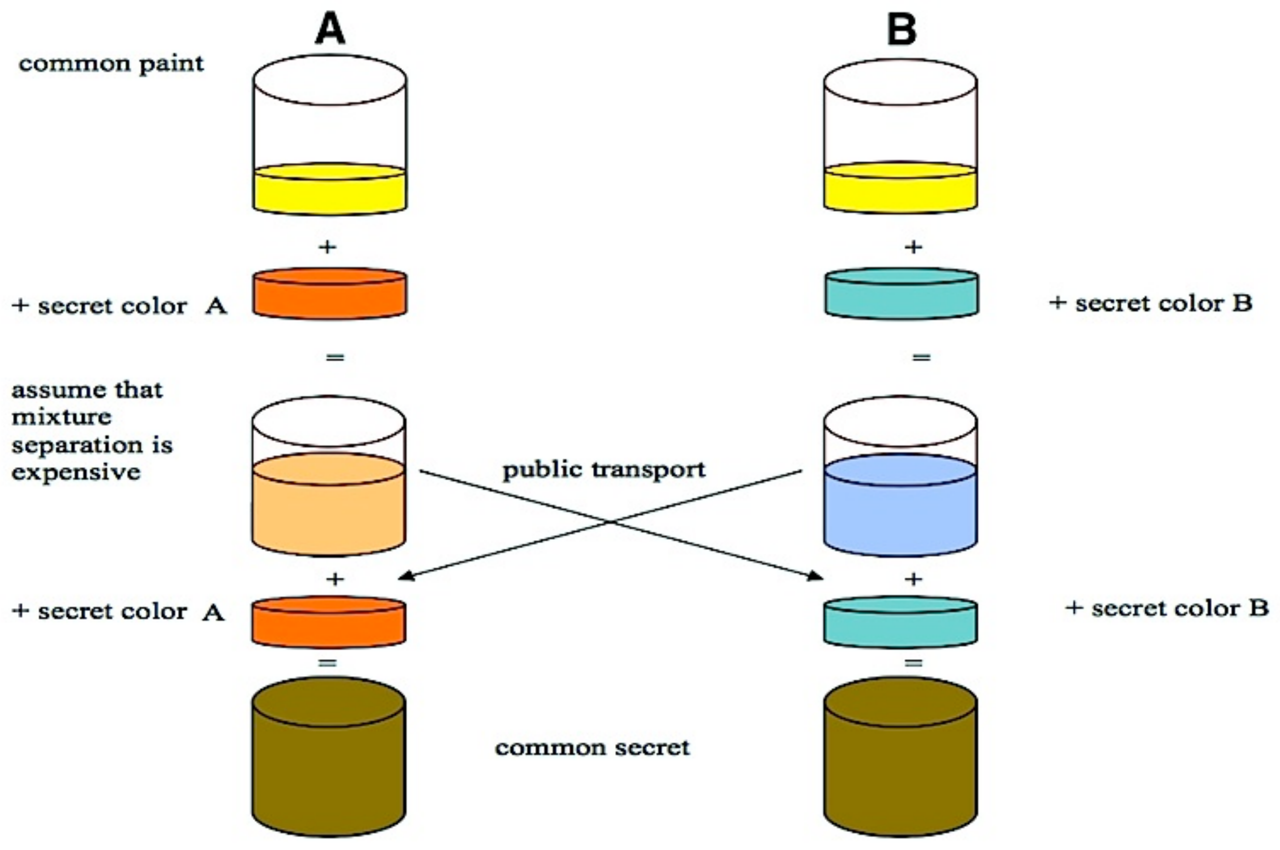
Original slides by Franzi

[Examples/Images thanks to Wikipedia.]

Lab 1 Deadline Reminders

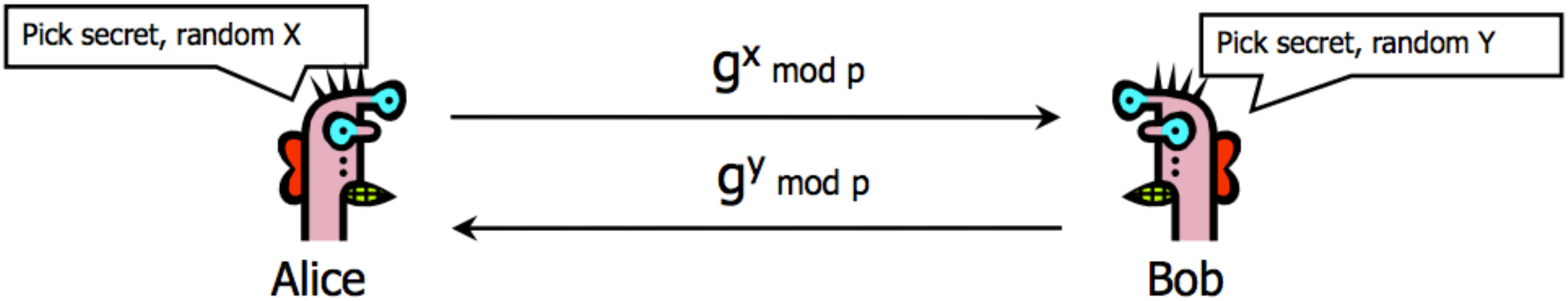
- Lab 1 Final due next week (5/1, 5pm).
- Upcoming office hours:
  - Tomorrow (Friday) 9:30 am – Michael & Adrian
  - Monday 9:30 am – Franzi
  - Wednesday 3:30 pm – Adrian & Peter
  - Thursday 12:30 pm – Peter & Michael

Illustration of DH as paint mixing



Wikipedia image

DH Summary



Compute $k=(g^y)^x=g^{xy} \bmod p$

Compute $k=(g^x)^y=g^{xy} \bmod p$

- Public info: p (large prime) and g (generator of Z_p^*)

$Z_p^*=\{1, 2 \ldots p-1\}$; $\forall a \in Z_p^* \; \exists i$ such that $a=g^i \bmod p$

RSA Summary

- Key generation
  - Generate large primes p, q
    - Say, 1024 bits each (need primality testing, too)
  - Compute $n = pq$ and $\varphi(n) = (p-1)(q-1)$
  - Choose small e, relatively prime to $\varphi(n)$
  - Compute unique d such that $ed = 1 \bmod \varphi(n)$
  - Public key = (e,n); private key = (d,n)
- Encryption of m: $c = m^e \bmod n$
  - Modular exponentiation by repeated squaring
- Decryption of c: $c^d \bmod n = (m^e)^d \bmod n = m$

Sample RSA Decryption

- 26 2 15 13 7 14 13 13 1 28 14 15 13 14 20 9 6 31 25 26 14 16 23 15 26 2 6 13 1
- p=3, q=11, n=33, e=7, d=3

- A-1 B-2 C-3 D-4 E-5 F-6 G-7 H-8 I-9 J-10 K-11 L-12 M-13 N-14 O-15 P-16 Q-17 R-18 S-19 T-20 U-21 V-22 W-23 X-24 Y-25 Z-26

Sample RSA Decryption

- How to compute d?
  - Recall: ed = 1 mod φ(n) (where φ(n) = (p-1)(q-1))
  - So d is inverse of e mod φ(n).
  - How to compute modular inverse?
    - Use extended Euclidean algorithm
    - … or Wolfram Alpha ☺
    - Note that this is hard if you don't know φ(n) (i.e., can't factor n).

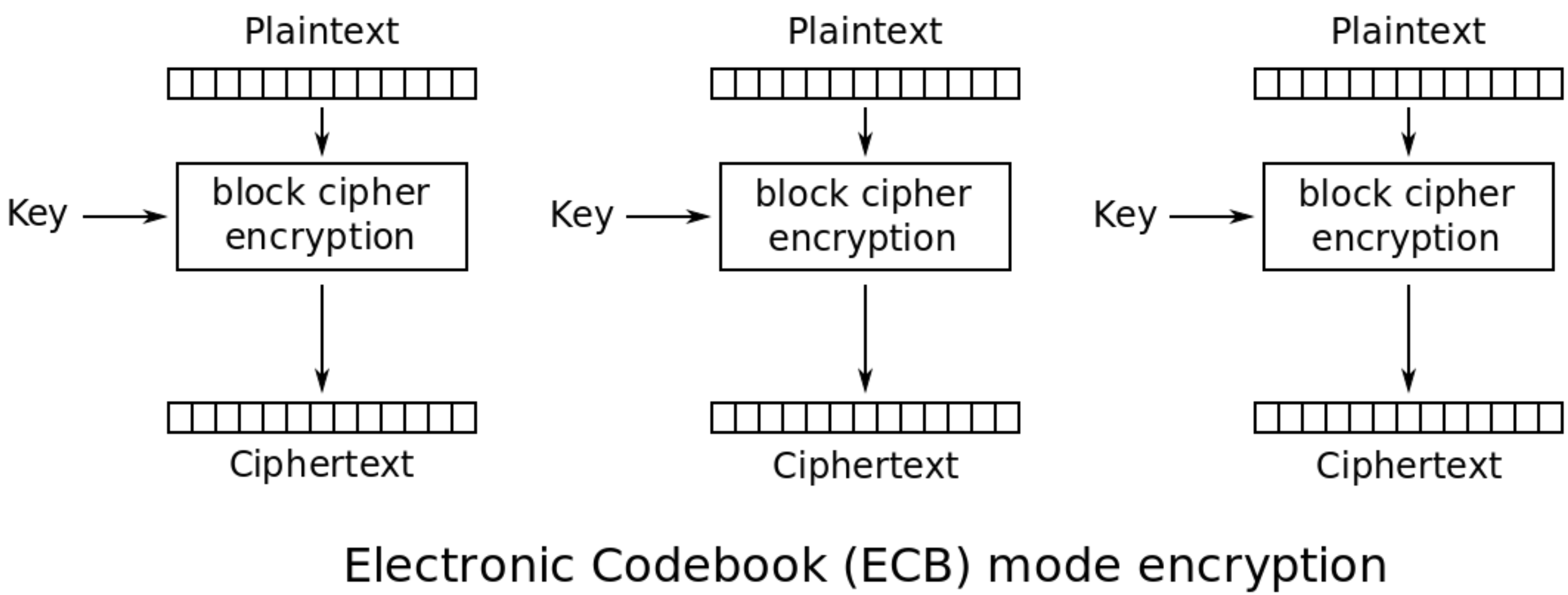
Public Key Crypto Summary

- Diffie-Hellman: Why is it secure?
  - Discrete log; computational DH problem; decisional DH problem are hard.
- RSA: Why is it secure?
  - Taking e^{th} root is hard; Factoring is hard.

Cryptography Summary

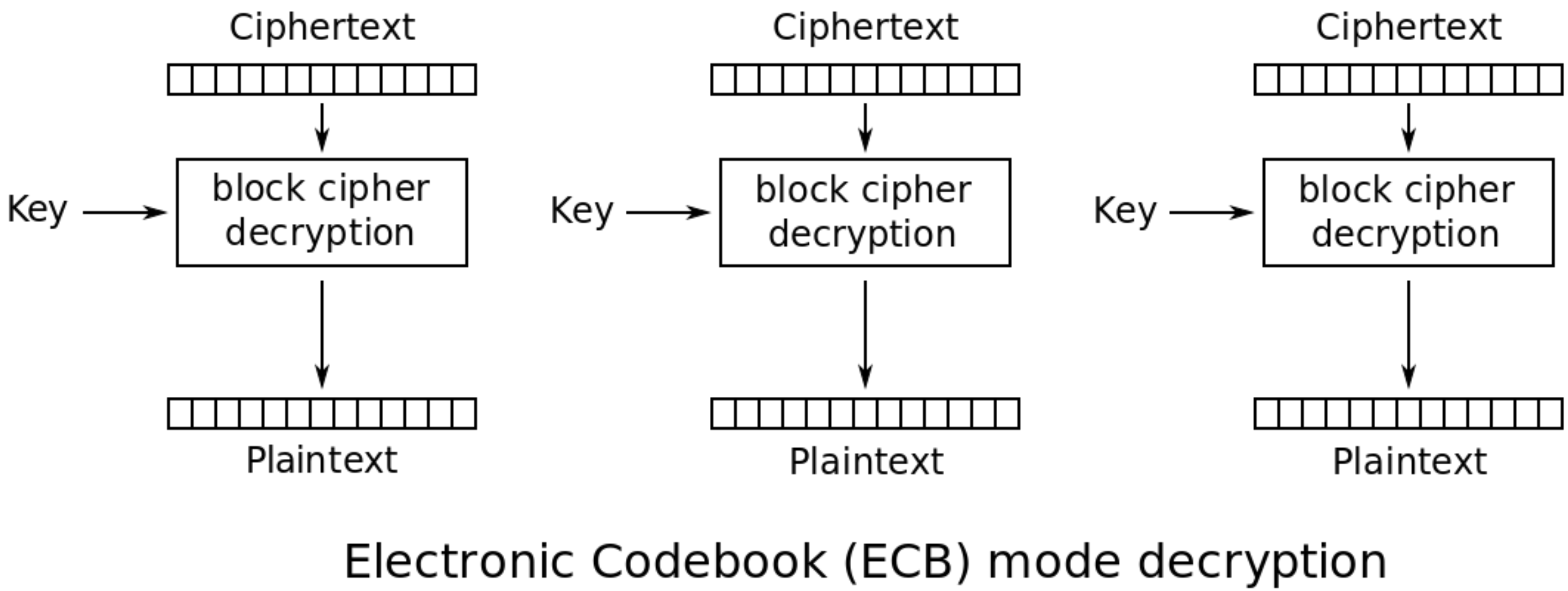
- Goal: Privacy
  - One-time pad
  - Block ciphers w/ symmetric keys (e.g., DES, AES)
    - Modes: EBC, CBC, CTR
  - Public key crypto (e.g., Diffie-Hellman, RSA)
- Goal: Integrity
  - MACs, often using hash functions (e.g, MD5, SHA-256)
- Goal: Privacy and Integrity
  - Encrypt-then-MAC (why?)
- Goal: Authenticity (and Integrity)
  - Digital signatures (e.g., RSA, DSS)

Block Cipher Mode: ECB



[Image from Wikipedia]

Block Cipher Mode: ECB

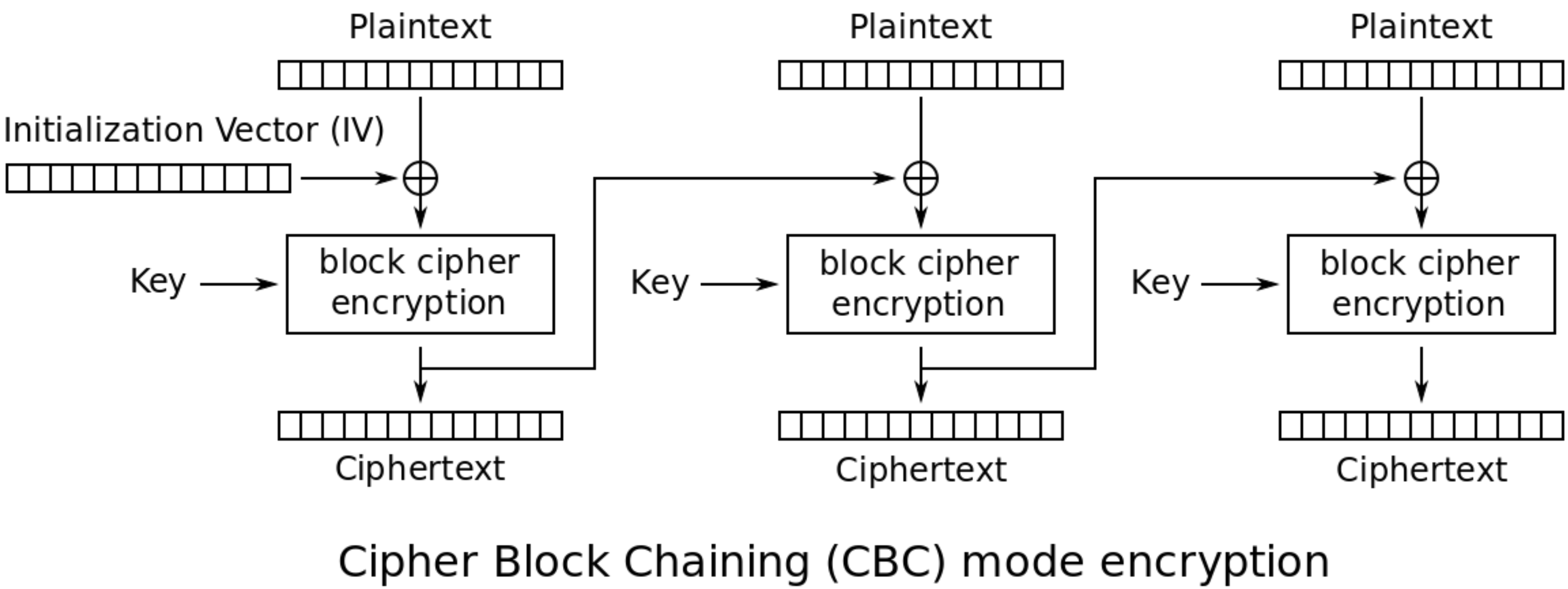


[Image from Wikipedia]

ECB Pros and cons

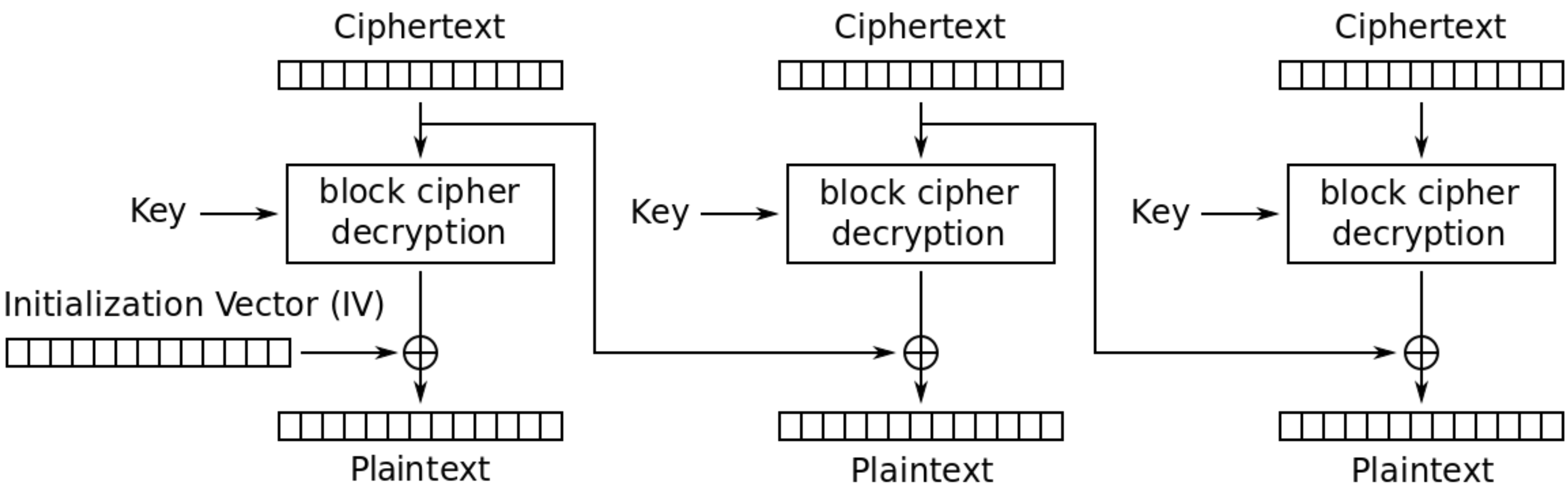
- Encryption and decryption parallelizable
- Does not hide data patterns well, not recommended

Block Cipher Mode: CBC



[Image from Wikipedia]

Block Cipher Mode: CBC



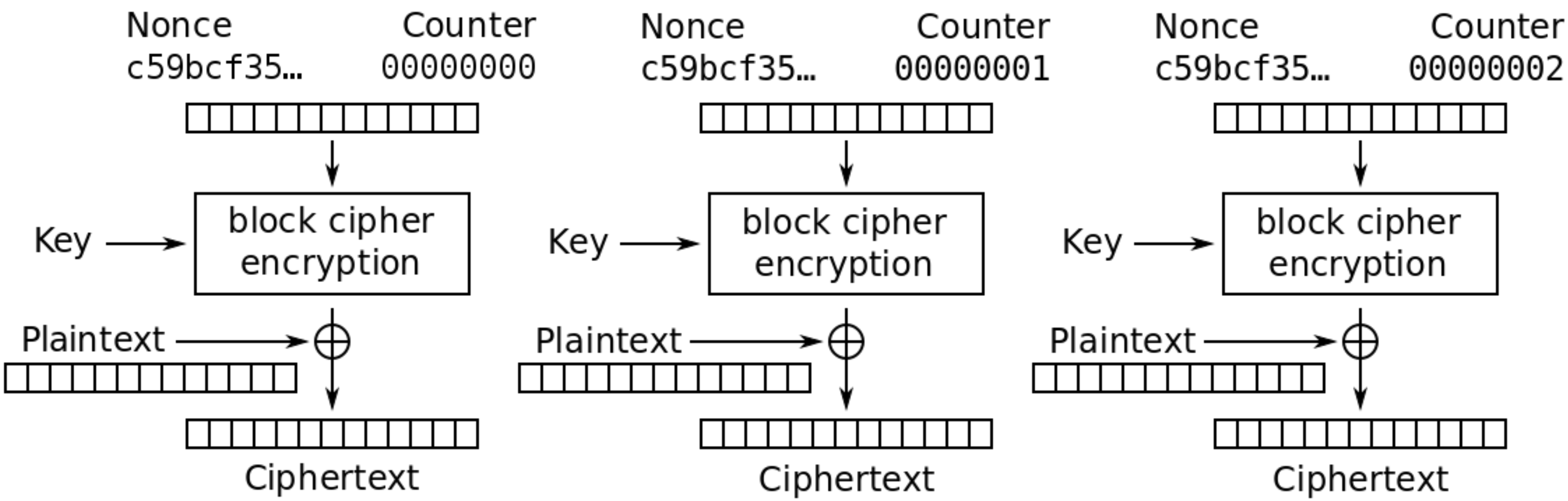
Cipher Block Chaining (CBC) mode decryption

[Image from Wikipedia]

CBC Pros and cons

- Encryption not parallelizable
- Decryption is parallelizable

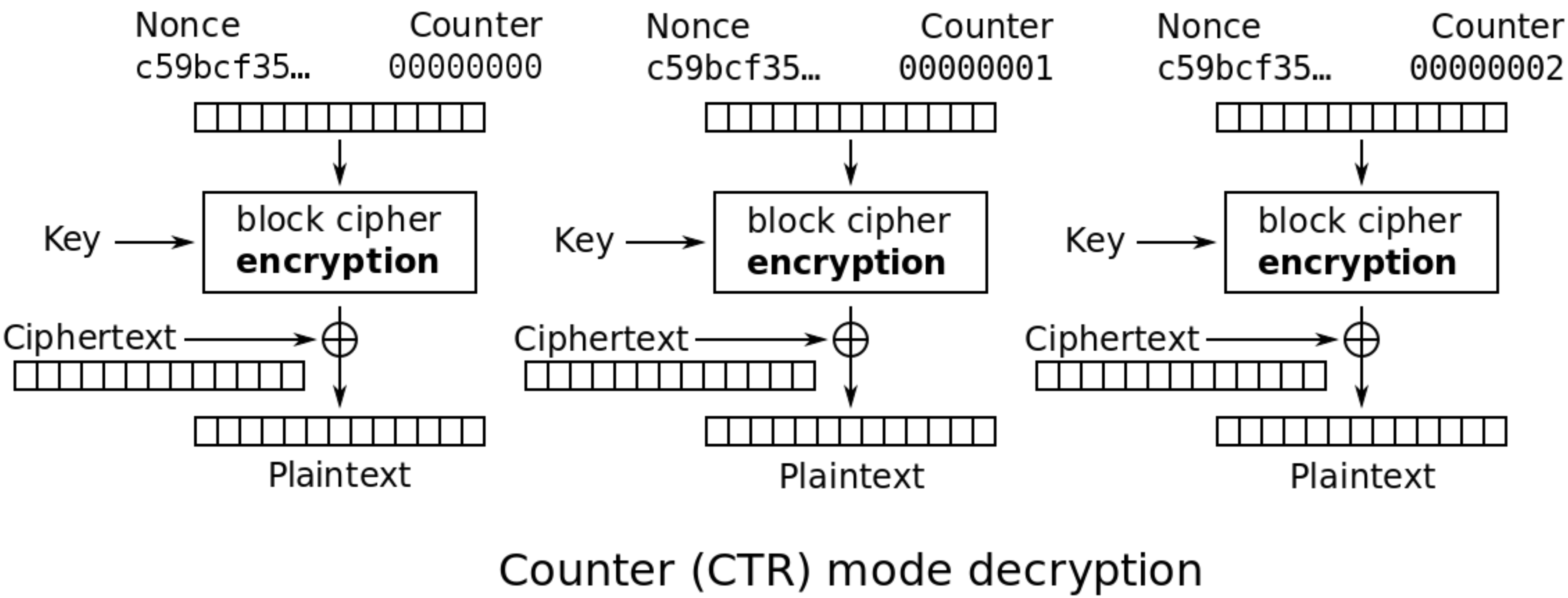
Block Cipher Mode: CTR



Counter (CTR) mode encryption

[Image from Wikipedia]

Block Cipher Mode: CTR

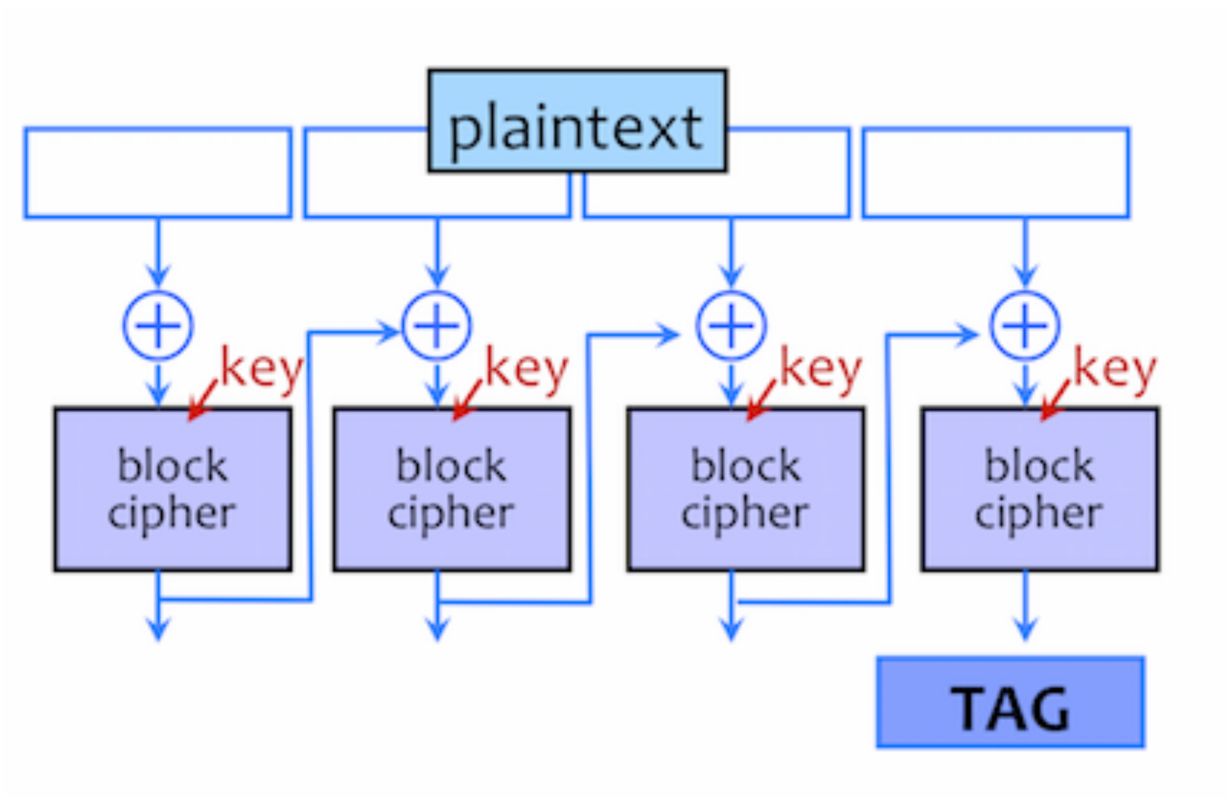


[Image from Wikipedia]

Pros and cons

- Encryption and decryption parallelizable
- CBC and CTR usage recommended by Yoshi, Niels and Bruce Schneier! (Cryptography Engineering, 2010)

CBC-MAC question

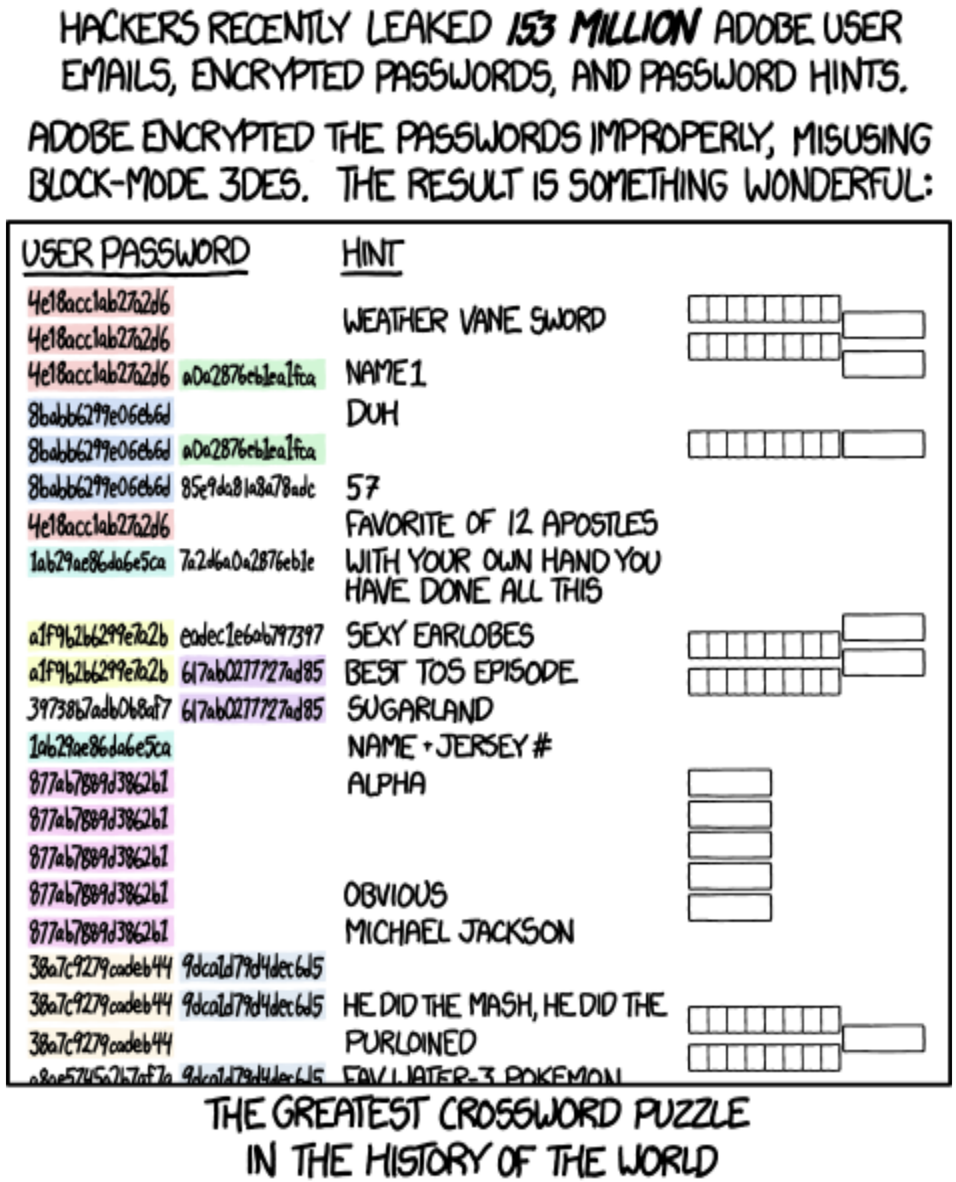


Given a message M with tag T (aka CBC-MAC(M)=T), can you construct a message M' (not necessarily the same length as M) for which the tag is *also* T, aka CBC-MAC(M')=T?

Password Salting

- Servers shouldn't store passwords, but password hashes. (Why?)
- Threat: rainbow tables (pre-computed password hashes)
- Solution: salt
  - Each password is hashed/stored with a random value. Now a pre-computed table is useless.
  - Other benefits?

Real world example, by xkcd



HTTP://XKCD.COM/1286/

Additional Resources

- Stanford online crypto class: https://class.coursera.org/crypto-preview/class
- Books:
  - "The Codebreakers" by David Kahn
  - "The Code Book" by Simon Singh