CSE 484 (Winter 2010)

Symmetric Cryptography

Tadayoshi Kohno

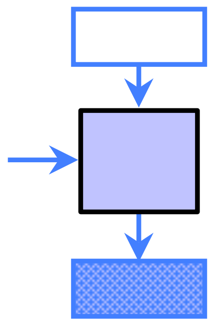
Thanks to Dan Boneh, Dieter Gollmann, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials ...

Goals for Today

- Under the hood: Symmetric encryption

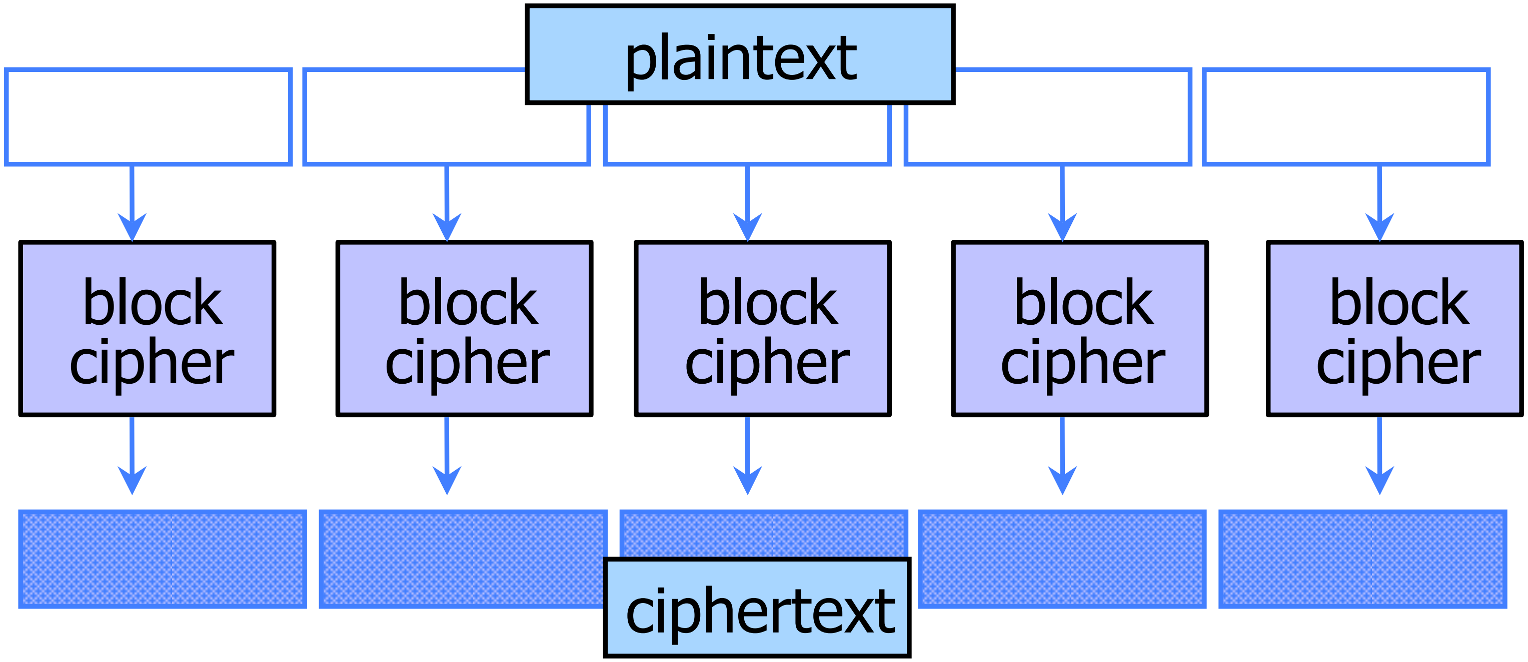
Encrypting a Large Message

- So, we've got a good block cipher, but our plaintext is larger than 128-bit block size
- Electronic Code Book (ECB) mode
  - Split plaintext into blocks, encrypt each one separately using the block cipher



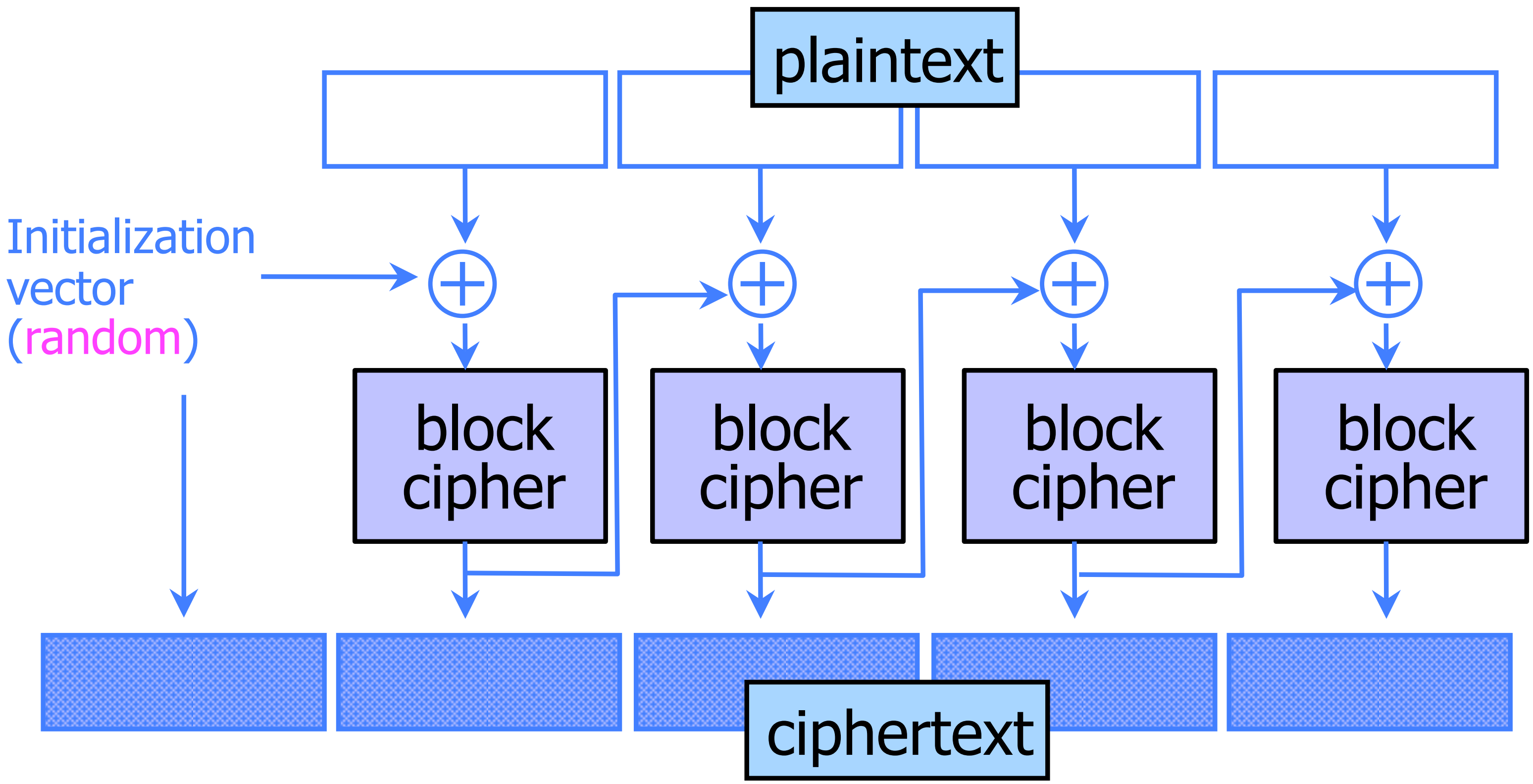
- Cipher Block Chaining (CBC) mode
  - Split plaintext into blocks, XOR each block with the result of encrypting previous blocks
- Counter (CTR) mode
  - Use block cipher to generate keystream, like a stream cipher
- …

ECB Mode



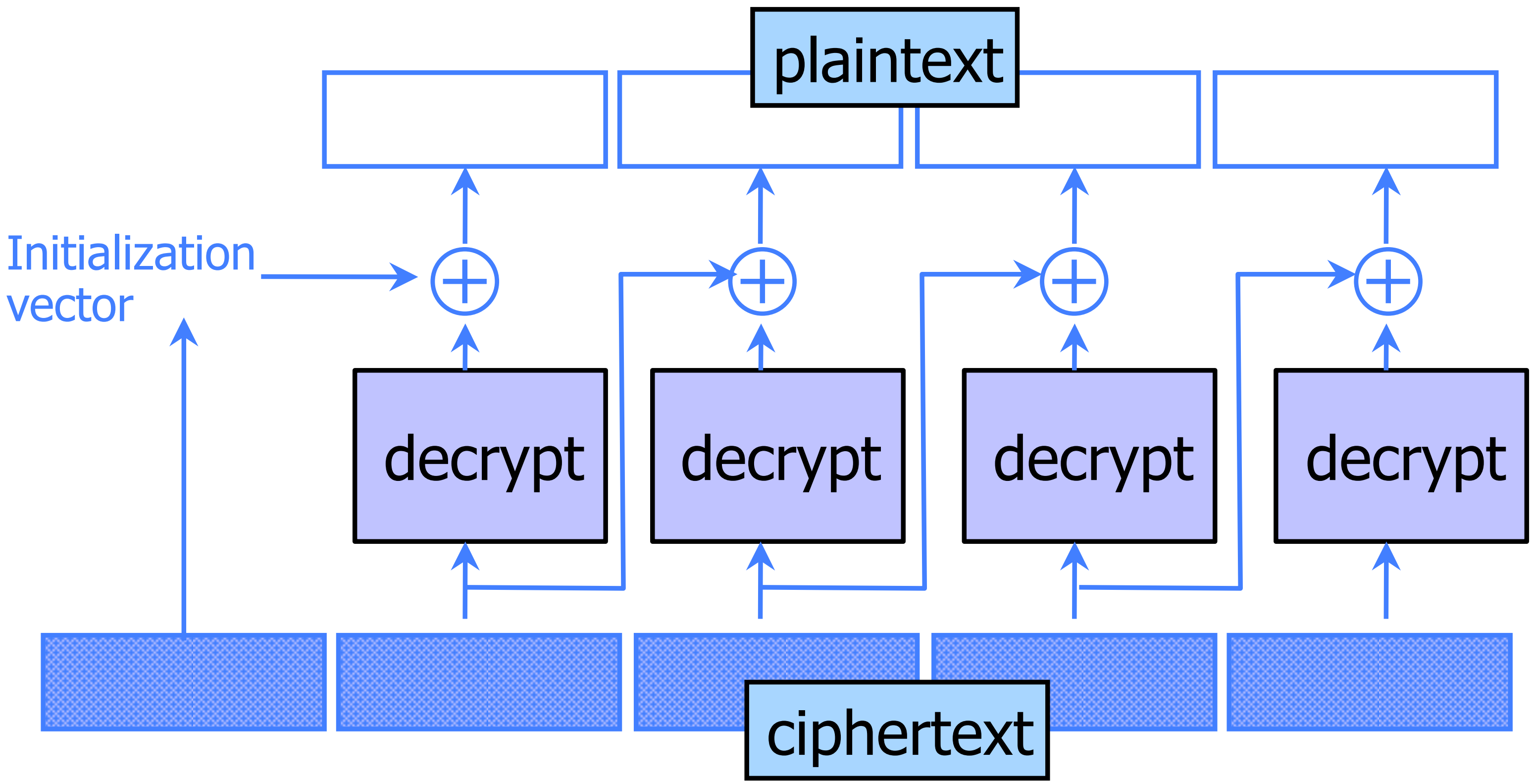
- Identical blocks of plaintext produce identical blocks of ciphertext
- No integrity checks: can mix and match blocks

CBC Mode: Encryption



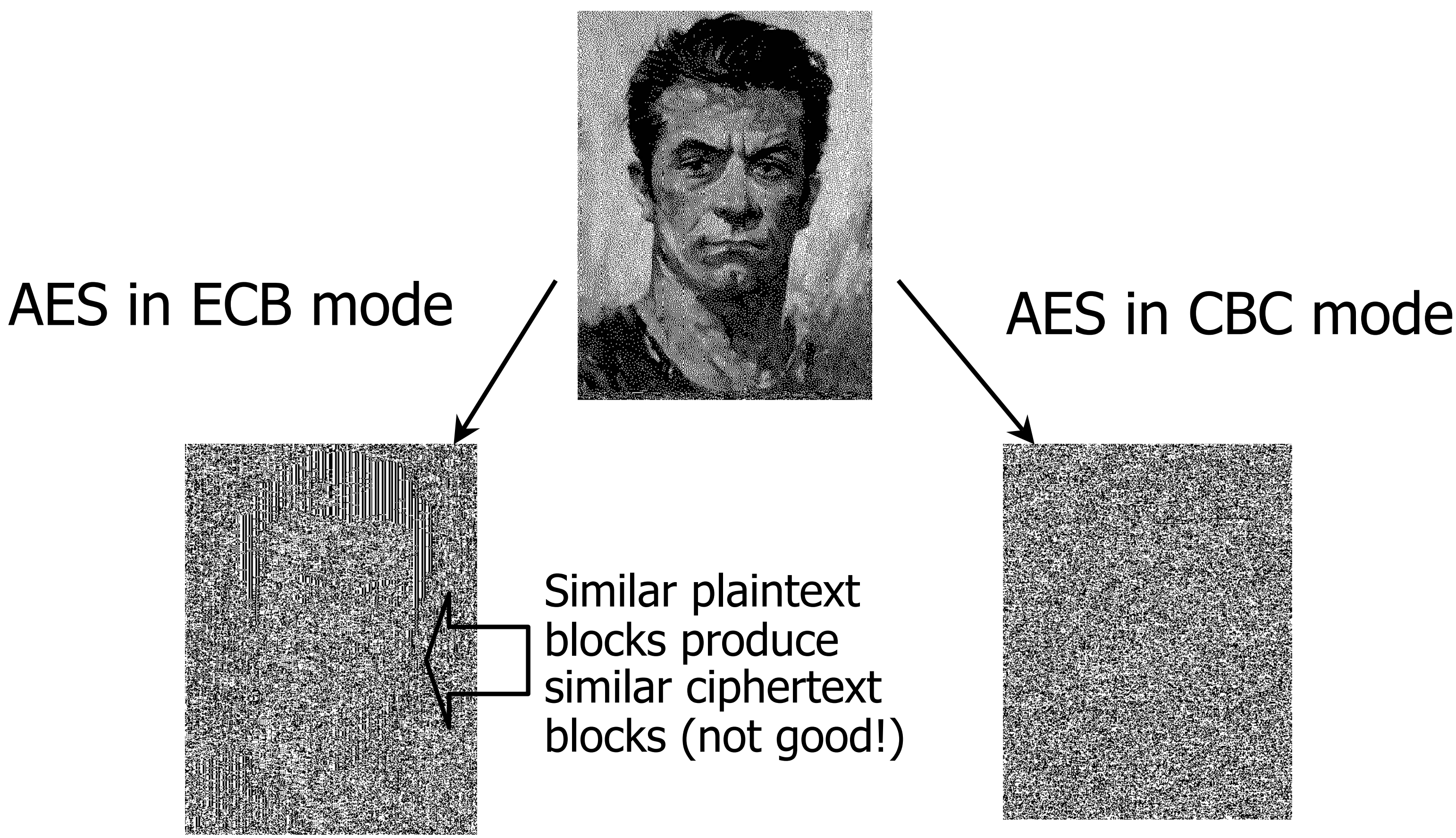
- Identical blocks of plaintext encrypted differently
- Last cipherblock depends on entire plaintext
  - Still does not guarantee integrity

CBC Mode: Decryption



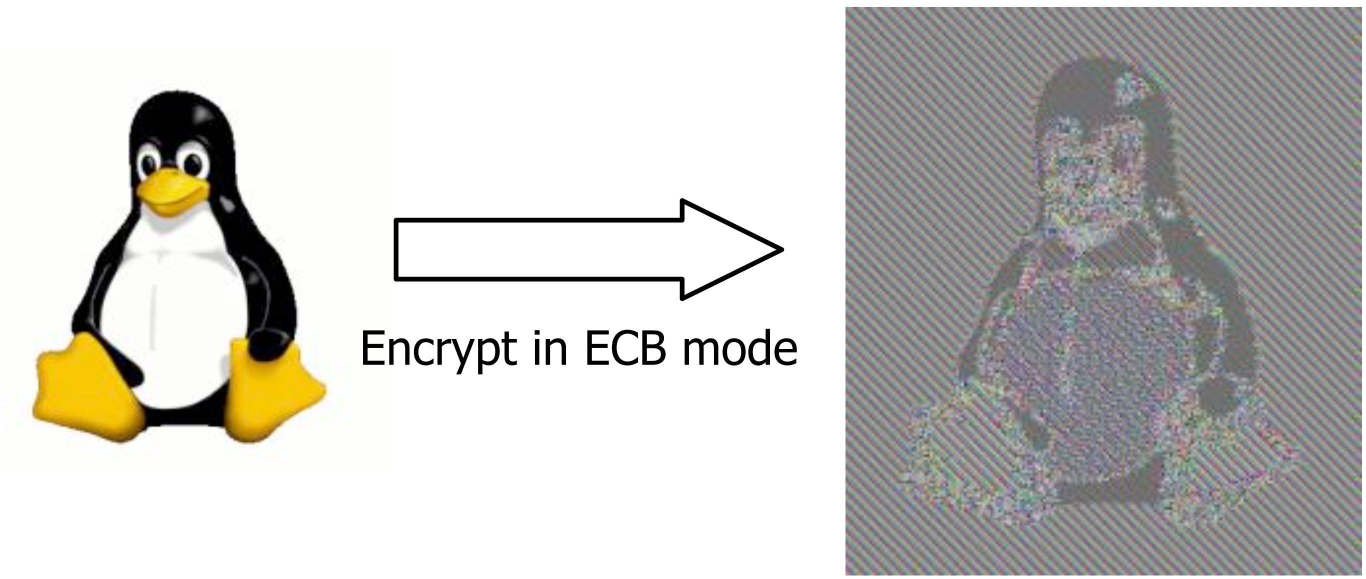
ECB vs. CBC

[Picture due to Bart Preneel]

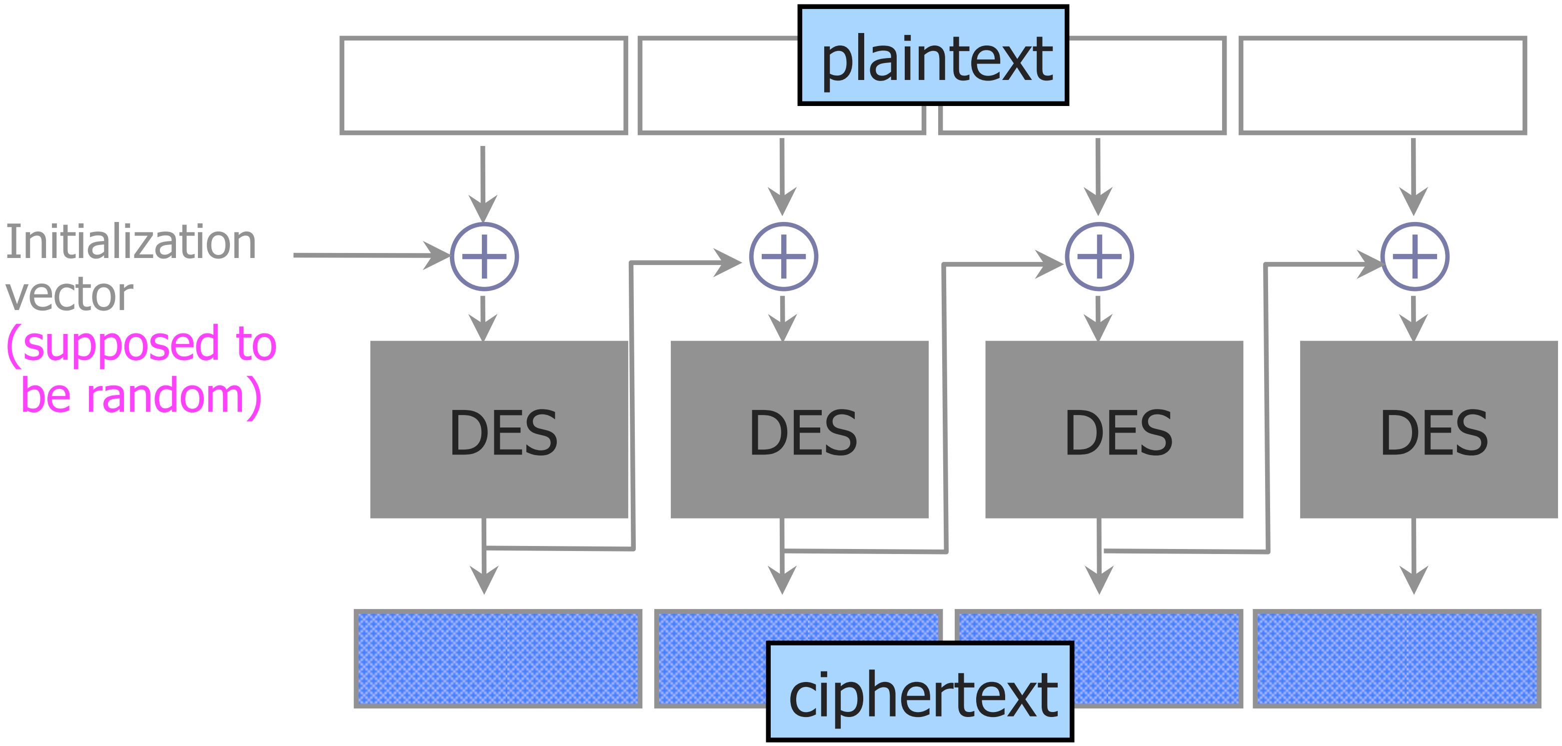


Information Leakage in ECB Mode

[Wikipedia]



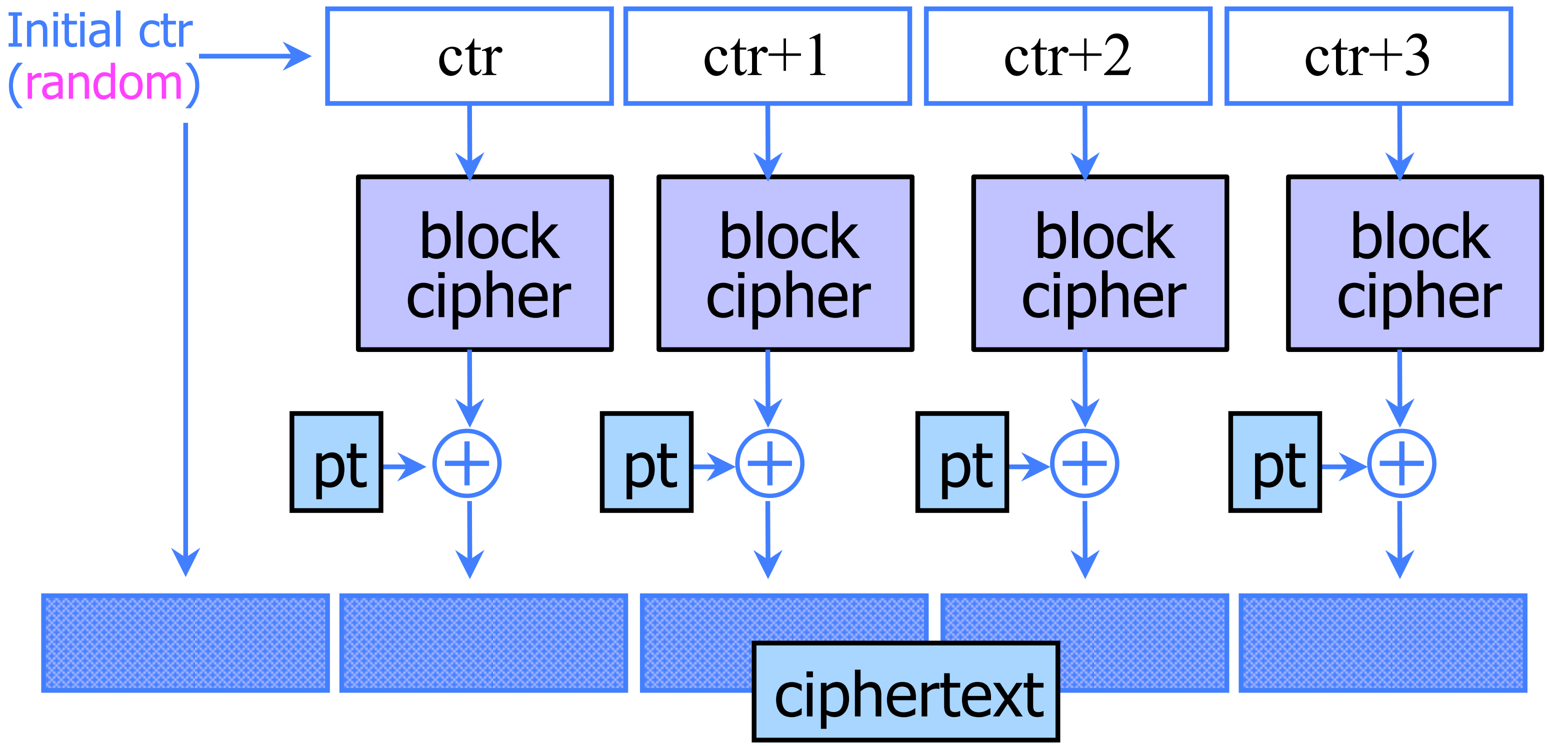
CBC and Electronic Voting



Found in the source code for Diebold voting machines:

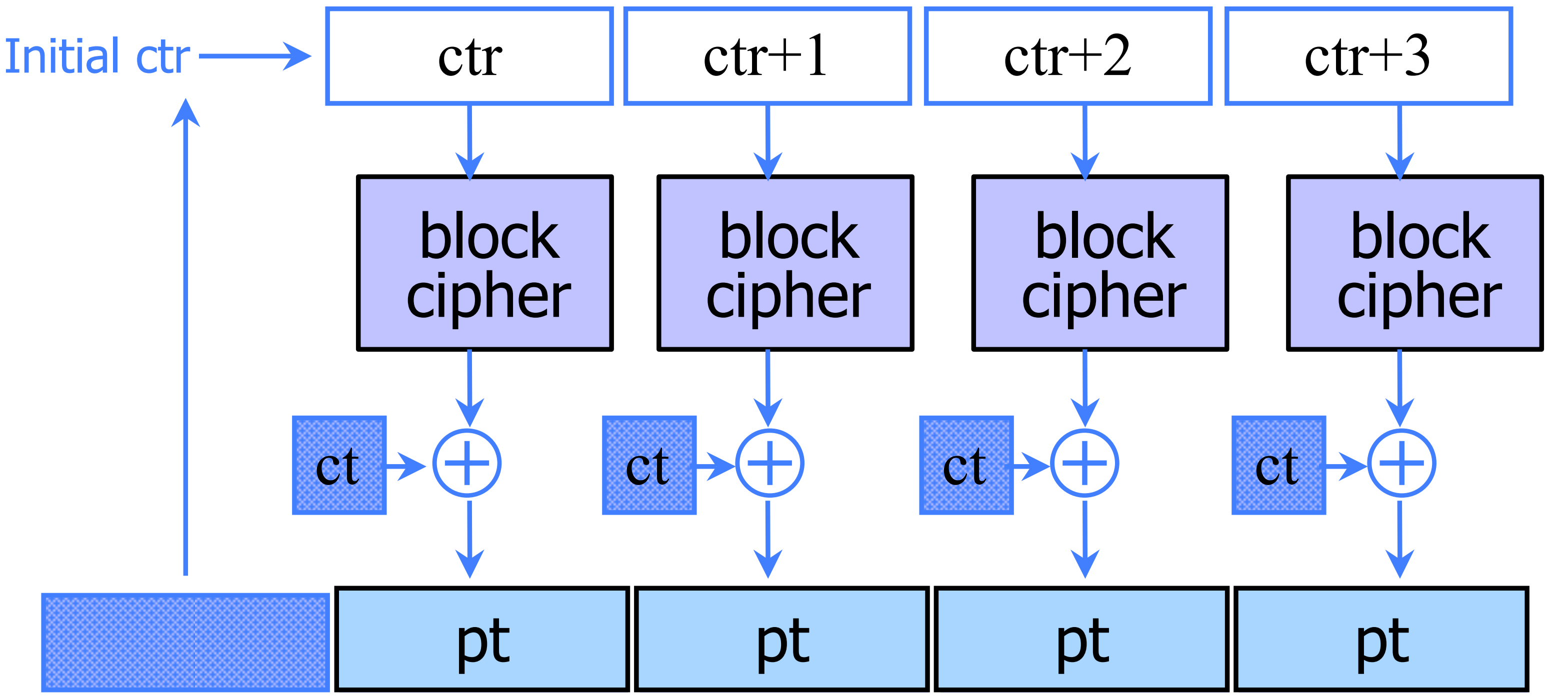
```
DesCBCEncrypt((des_c_block*)tmp, (des_c_block*)record.m_Data,
        totalSize, DESKEY, NULL, DES_ENCRYPT)
```

CTR Mode: Encryption



- Identical blocks of plaintext encrypted differently
- Still does not guarantee integrity
- Fragile if ctr repeats

CTR Mode: Decryption



When Is a Cipher "Secure"?

- Hard to recover the key?
  - What if attacker can learn plaintext without learning the key?
- Hard to recover plaintext from ciphertext?
  - What if attacker learns some bits or some function of bits?
- Fixed mapping from plaintexts to ciphertexts?
  - What if attacker sees two identical ciphertexts and infers that the corresponding plaintexts are identical?
  - Implication: encryption must be randomized or stateful

How Can a Cipher Be Attacked?

- Assume that the attacker knows the encryption algorithm and wants to decrypt some ciphertext
- Main question: what else does attacker know?
  - Depends on the application in which cipher is used!
- Ciphertext-only attack
- Known-plaintext attack (stronger)
  - Knows some plaintext-ciphertext pairs
- Chosen-plaintext attack (even stronger)
  - Can obtain ciphertext for any plaintext of his choice
- Chosen-ciphertext attack (very strong)
  - Can decrypt any ciphertext <u>except</u> the target
  - Sometimes very realistic model

Defining Security (Not Required)

- Attacker does not know the key
- He chooses as many plaintexts as he wants, and learns the corresponding ciphertexts
- When ready, he picks two plaintexts M_0 and M_1
  - He is even allowed to pick plaintexts for which he previously learned ciphertexts!
- He receives either a ciphertext of M_0, or a ciphertext of M_1
- He wins if he guesses correctly which one it is

Defining Security (Not Required)

- Idea: attacker should not be able to learn even a single bit of the encrypted plaintext
- Define Enc(M_0,M_1,b) to be a function that returns encrypted M_b

  0 or 1

  - Given two plaintexts, Enc returns a ciphertext of one or the other depending on the value of bit b
  - Think of Enc as a magic box that computes ciphertexts on attacker's demand. He can obtain a ciphertext of any plaintext M by submitting M_0=M_1=M, or he can try to learn even more by submitting $M_0 \neq M_1$.
- Attacker's goal is to learn just one bit b

Chosen-Plaintext Security (Not Required)

- Consider two experiments (A is the attacker)

Experiment 0	Experiment 1
A interacts with Enc(-,-,0) and outputs bit d	A interacts with Enc(-,-,1) and outputs bit d

  - Identical except for the value of the secret bit
  - d is attacker's guess of the secret bit
- Attacker's advantage is defined as

If A "knows" secret bit, he should be able to make his output depend on it

| Prob(A outputs 1 in Exp0) - Prob(A outputs 1 in Exp1)) |

- Encryption scheme is chosen-plaintext secure if this advantage is negligible for any efficient A

"Simple" Example (Not Required)

- Any deterministic, stateless symmetric encryption scheme is insecure
  - Attacker can easily distinguish encryptions of different plaintexts from encryptions of identical plaintexts
  - This includes ECB mode of common block ciphers!

Attacker A interacts with Enc(-,-,b)

Let X,Y be any two different plaintexts

$C_1 \leftarrow$ Enc(X,Y,b); $C_2 \leftarrow$ Enc(Y,Y,b);

If $C_1=C_2$ then b=1 else say b=0

- The advantage of this attacker A is 1

Prob(A outputs 1 if b=0)=0 Prob(A outputs 1 if b=1)=1

Why Hide Everything?

- Leaking even a little bit of information about the plaintext can be disastrous
- Electronic voting
  - 2 candidates on the ballot (1 bit to encode the vote)
  - If ciphertext leaks the parity bit of the encrypted plaintext, eavesdropper learns the entire vote
- D-Day: Pas-de-Calais or Normandy?
  - Allies convinced Germans that invasion will take place at Pas-de-Calais
    - Dummy landing craft, feed information to double spies
  - Goal: hide a 1-bit secret
- Also, want a strong definition, that implies others

Birthday attacks

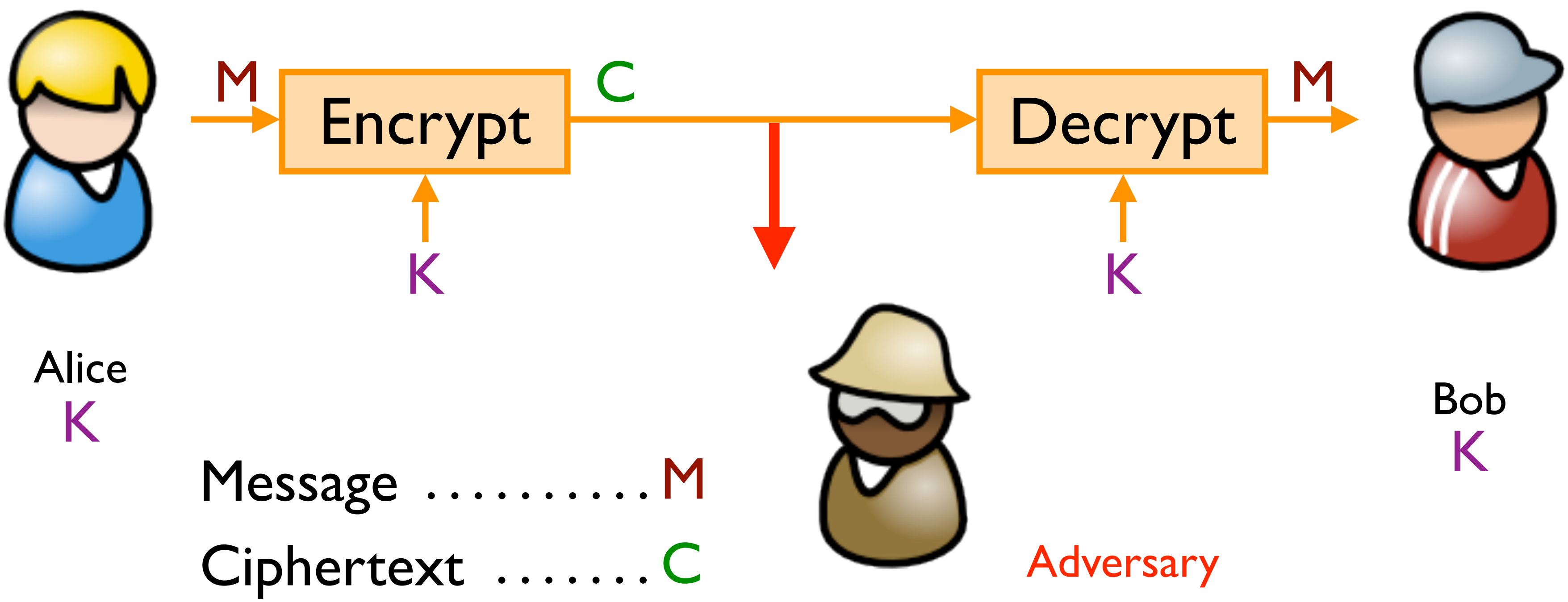
- Are there two people in the first 1/3 of this classroom that have the same birthday?
  - Yes?
  - No?
  - Experiment

Birthday attacks

- Why is this important for cryptography?
  - 365 days in a year (366 some years)
    - Pick one person. To find another person with same birthday would take on the order of 365/2 = 182.5 people
    - Expect "collision" -- two people with same birthday -- with a room of only 23 people
    - For simplicity, approximate when we expect a collision as the square root of 365.
  - 2^{128} different 128-bit keys
    - Pick one key at random. To exhaustively search for this key requires trying on average 2^{127} keys.
    - Expect a "collision" after selecting approximately 2^{64} random keys.
    - 64 bits of security against collision attacks, not 128 bits.

Achieving Privacy (Symmetric)

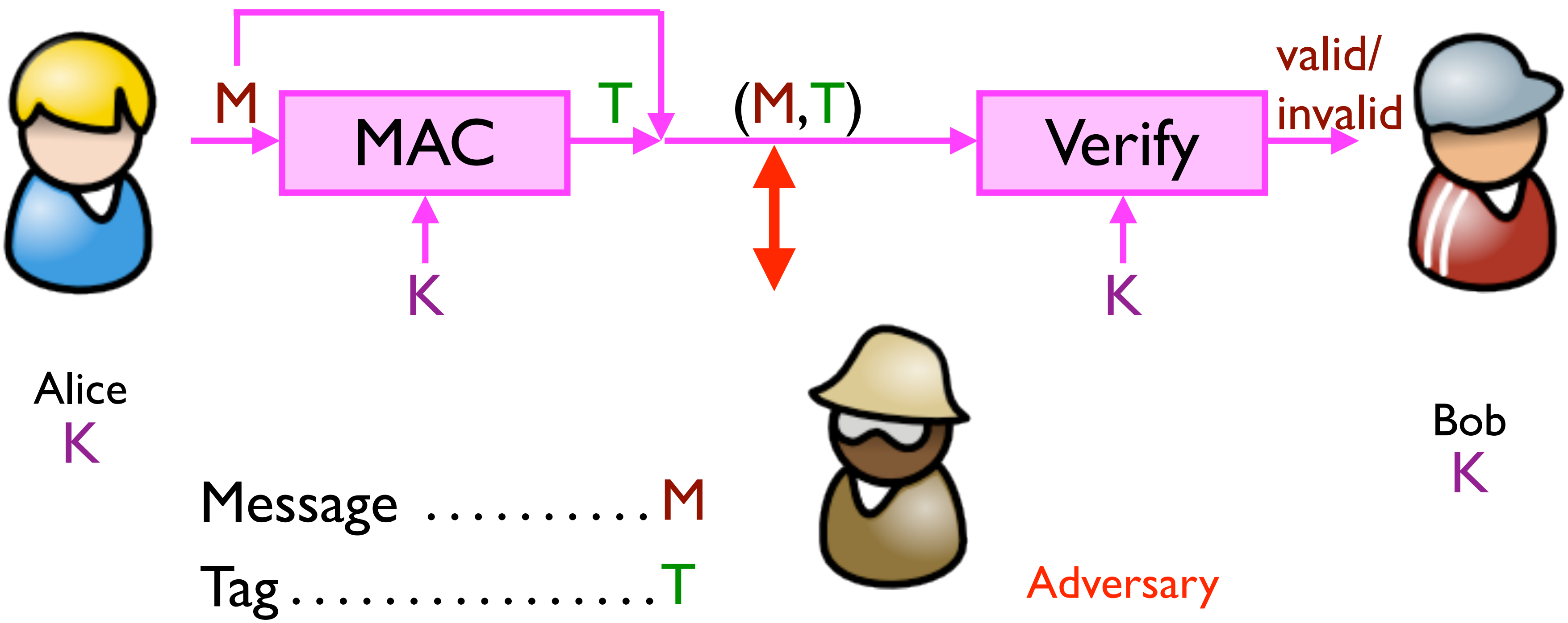
Encryption schemes: A tool for protecting privacy.



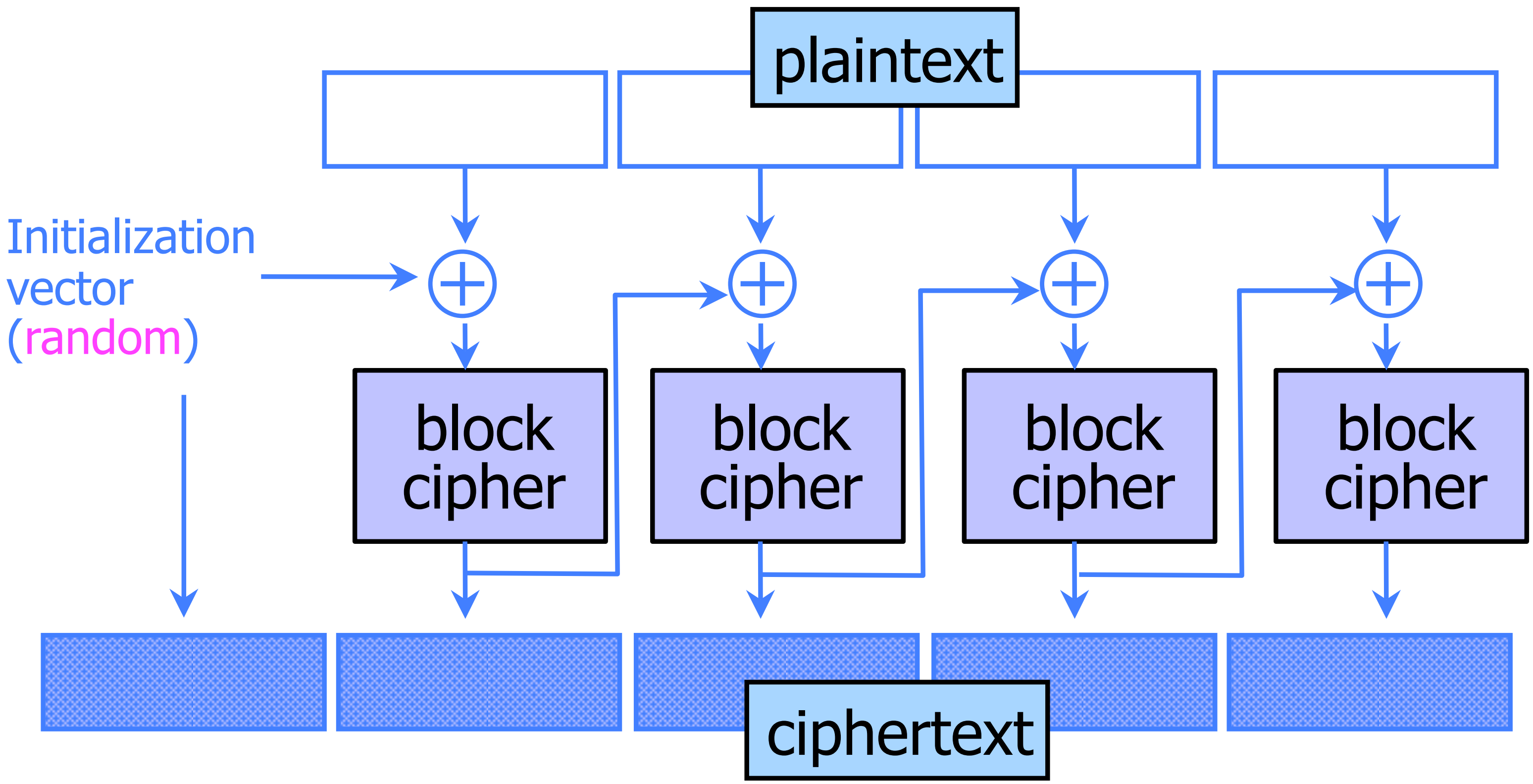
Achieving Integrity (Symmetric)

Message authentication schemes: A tool for protecting integrity.

(Also called message authentication codes or MACs.)

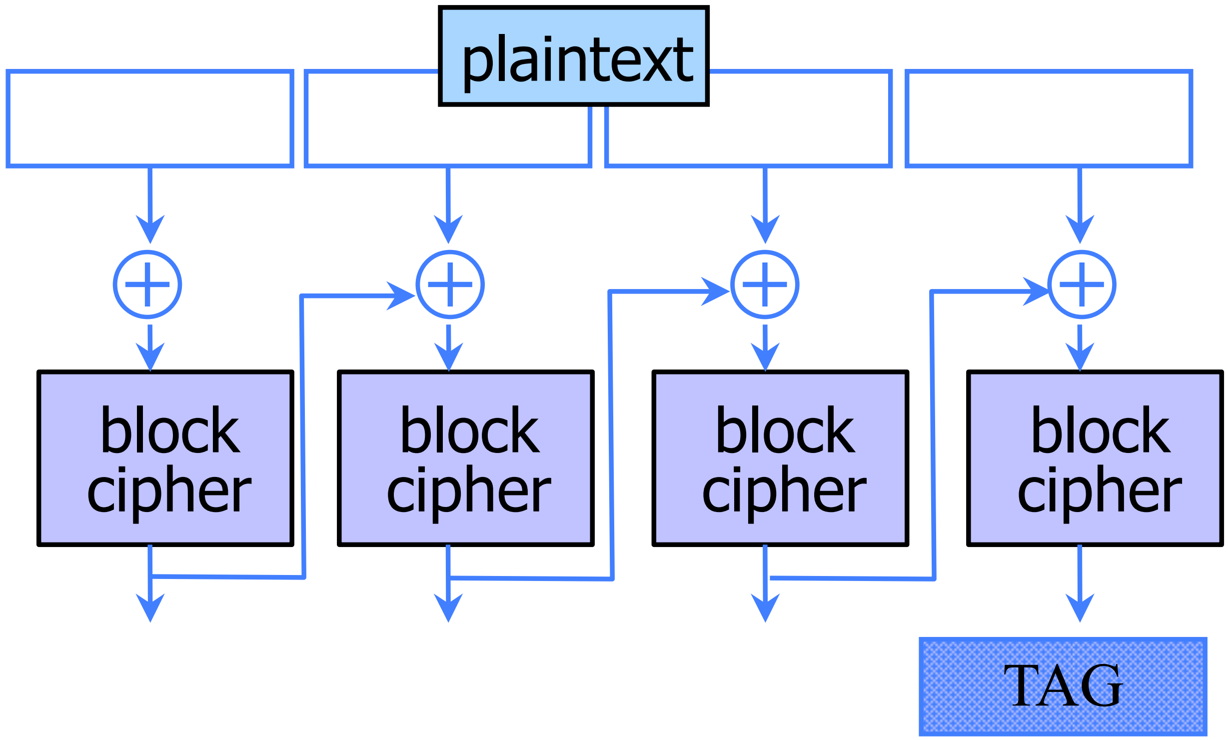


CBC Mode: Encryption



- Identical blocks of plaintext encrypted differently
- Last cipherblock depends on entire plaintext
  - Still does not guarantee integrity

CBC-MAC



- Not secure when system may MAC messages of different lengths.
  - Encode length at beginning: Whiteboard example
  - Use a derivative called CMAC
- Internal collisions and birthday attacks: Whiteboard example