CSE 484 / CSE M 584: Computer Security and Privacy

Cryptography: Symmetric Encryption

Fall 2016

Adam (Ada) Lerner
lerner@cs.washington.edu

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

Alice and Bob

- Archetypical characters

Eve
(eavesdrops)

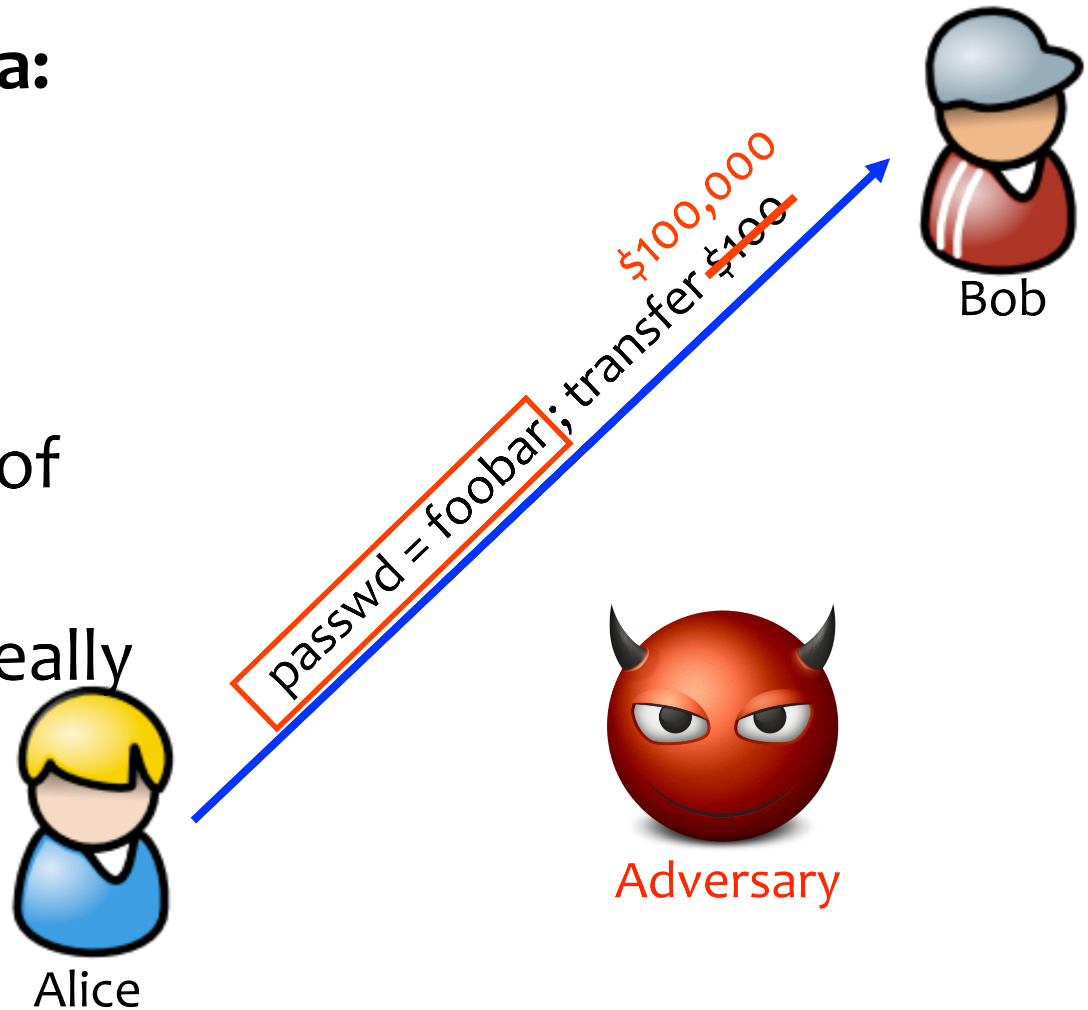
Mallory
(is malicious)

Common Communication Security Goals

Confidentiality of data:
Prevent exposure of information

Integrity of data:
Prevent modification of information

Authenticity : Is this really Bob I'm talking to?



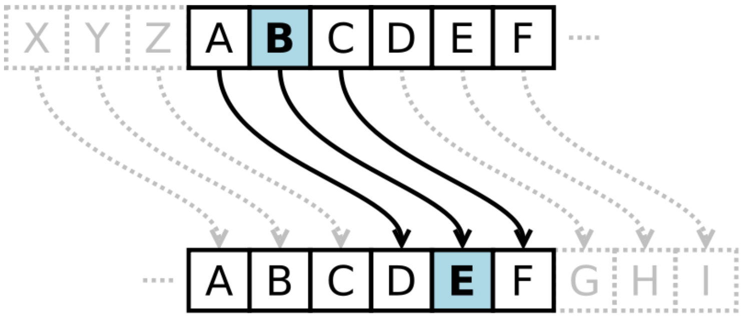
History

- Substitution Ciphers
  - Caesar Cipher
- Transposition Ciphers
- Codebooks
- Machines

- Recommended Reading: **The Codebreakers** by David Kahn and **The Code Book** by Simon Singh.

History: Caesar Cipher (Shift Cipher)

- Plaintext letters are replaced with letters a fixed shift away in the alphabet.



- Example:
  - Plaintext: `The quick brown fox jumps over the lazy dog`
  - Key: Shift 3

    `ABCDEFGHIJKLMNOPQRSTUVWXYZ`

    `DEFGHIJKLMNOPQRSTUVWXYZABC`

  - Ciphertext: `WKHTX LFNEU RZQIR AMXPS VRYHU WKHOD CBGRJ`

History: Caesar Cipher (Shift Cipher)

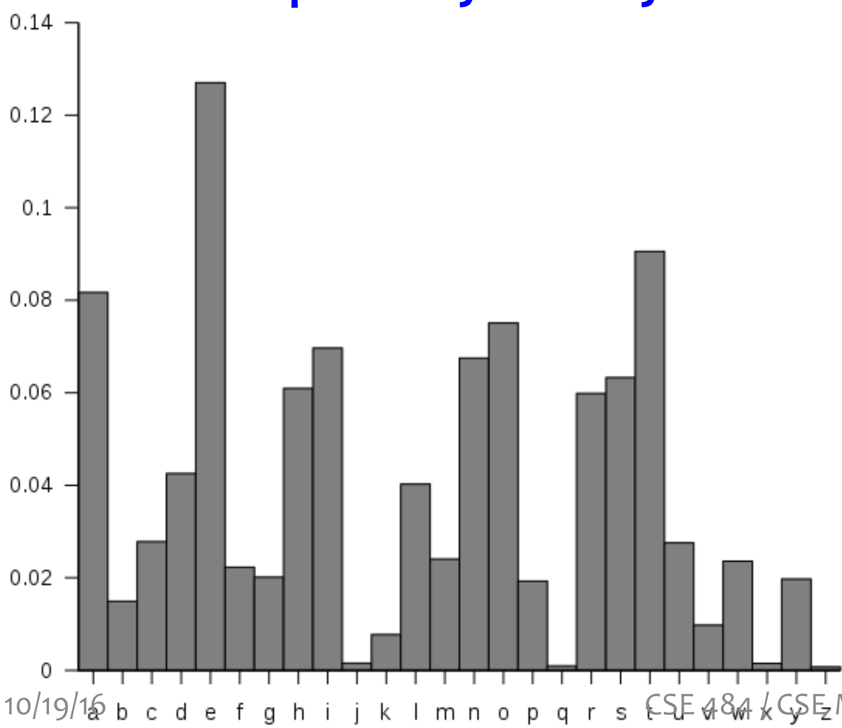
- ROT13: shift 13 (encryption and decryption: same operation)
- What is the key space?
  - 26 possible shifts.
- How to attack shift ciphers?
  - Brute force.

History: Substitution Cipher

- Superset of shift ciphers: each letter is substituted for another one.
- Add a secret key
- Example:
  - Plaintext: `ABCDEFGHIJKLMNOPQRSTUVWXYZ`
  - Cipher: `ZEBRASCDFGHIJKLMNOPQTUVWXY`
- "State of the art" for thousands of years

History: Substitution Cipher

- What is the key space? 26! ~= 2^88
- How to attack?
  - Frequency analysis.



Bigrams:

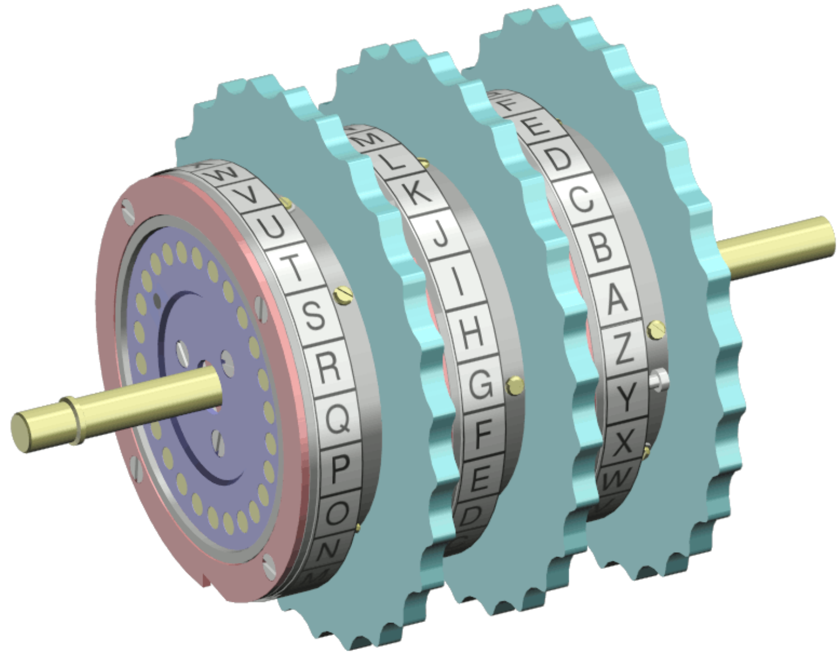
th 1.52%	en 0.55%	ng 0.18%
he 1.28%	ed 0.53%	of 0.16%
in 0.94%	to 0.52%	al 0.09%
er 0.94%	it 0.50%	de 0.09%
an 0.82%	ou 0.50%	se 0.08%
re 0.68%	ea 0.47%	le 0.08%
nd 0.63%	hi 0.46%	sa 0.06%
at 0.59%	is 0.46%	si 0.05%
on 0.57%	or 0.43%	ar 0.04%
nt 0.56%	ti 0.34%	ve 0.04%
ha 0.56%	as 0.33%	ra 0.04%
es 0.56%	te 0.27%	ld 0.02%
st 0.55%	et 0.19%	ur 0.02%

Trigrams:

1. the	6. ion	11. nce
2. and	7. tio	12. edt
3. tha	8. for	13. tis
4. ent	9. nde	14. oft
5. ing	10.has	15. sth

History: Enigma Machine

Uses rotors (substitution cipher) that change position after each key.



Key = initial setting of rotors

Key space?

26^n for n rotors

Kerckhoff’s Principle

- Security of a cryptographic object should depend only on the secrecy of the secret (private) key.
- Security should not depend on the secrecy of the algorithm itself (“security by obscurity”).

How Cryptosystems Work Today

- Public algorithms (Kerckhoff's Principle)
- Security proofs based on assumptions (not this course)

- Don't roll your own!

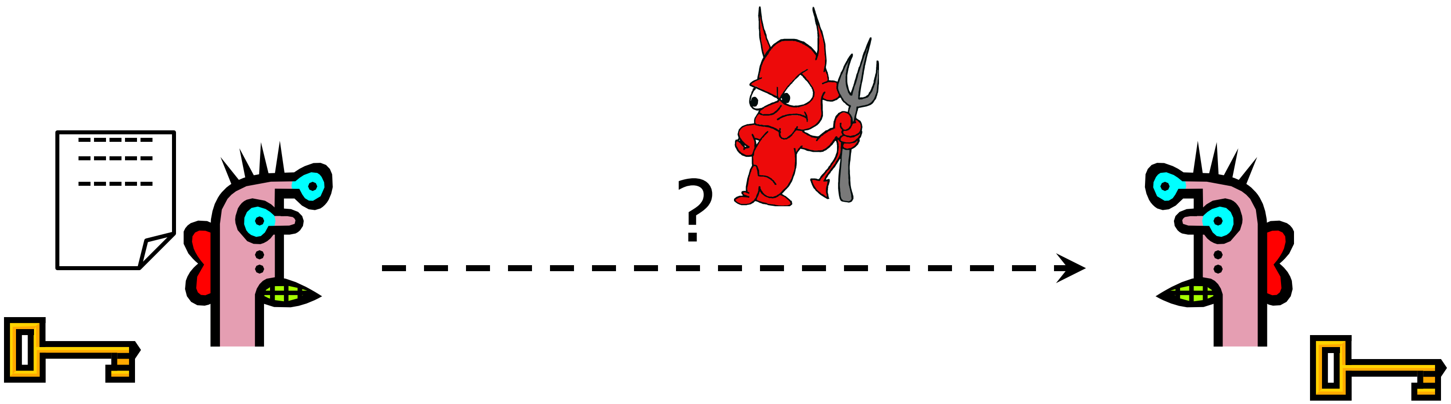
How Cryptosystems Work Today

- Layered approach:
  - Cryptographic primitives, like block ciphers, stream ciphers, hash functions, and one-way trapdoor permutations
  - Cryptographic protocols, like CBC mode encryption, CTR mode encryption, HMAC message authentication

Flavors of Cryptography

- Symmetric cryptography
  - Both communicating parties have access to a shared random string K, called the key.
- Asymmetric cryptography
  - Each party creates a public key pk and a secret key sk.

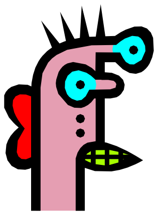
Confidentiality: Basic Problem

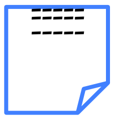


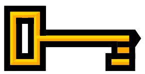
Goal: send a message confidentially.

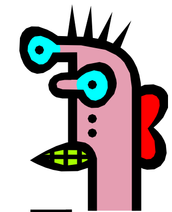
Given: both parties already know the same secret.

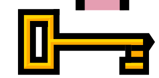
One-Time Pad



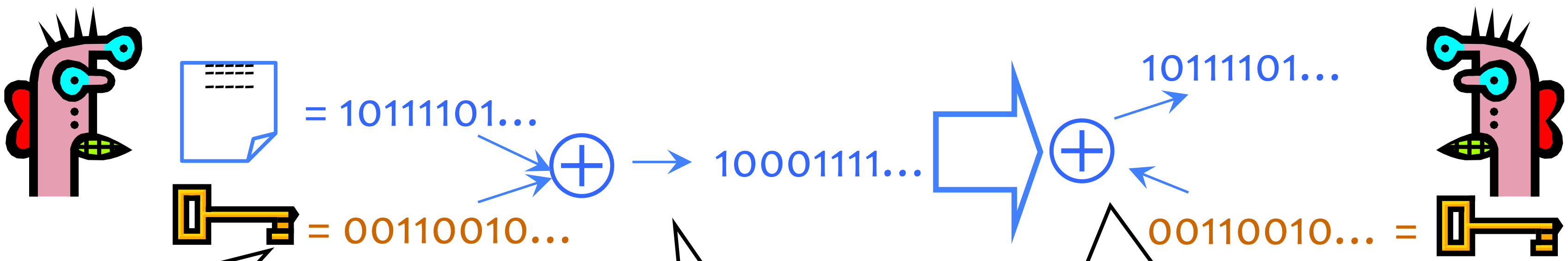
 = 10111101…

 = 00110010…



00110010… = 

One-Time Pad



Key is a random bit sequence as long as the plaintext

Encrypt by bitwise XOR of plaintext and key:
ciphertext = plaintext ⊕ key

Decrypt by bitwise XOR of ciphertext and key:
ciphertext ⊕ key =
(plaintext ⊕ key) ⊕ key =
plaintext ⊕ (key ⊕ key) =
plaintext

Cipher achieves perfect secrecy if and only if there are as many possible keys as possible plaintexts, and every key is equally likely (Claude Shannon, 1949)

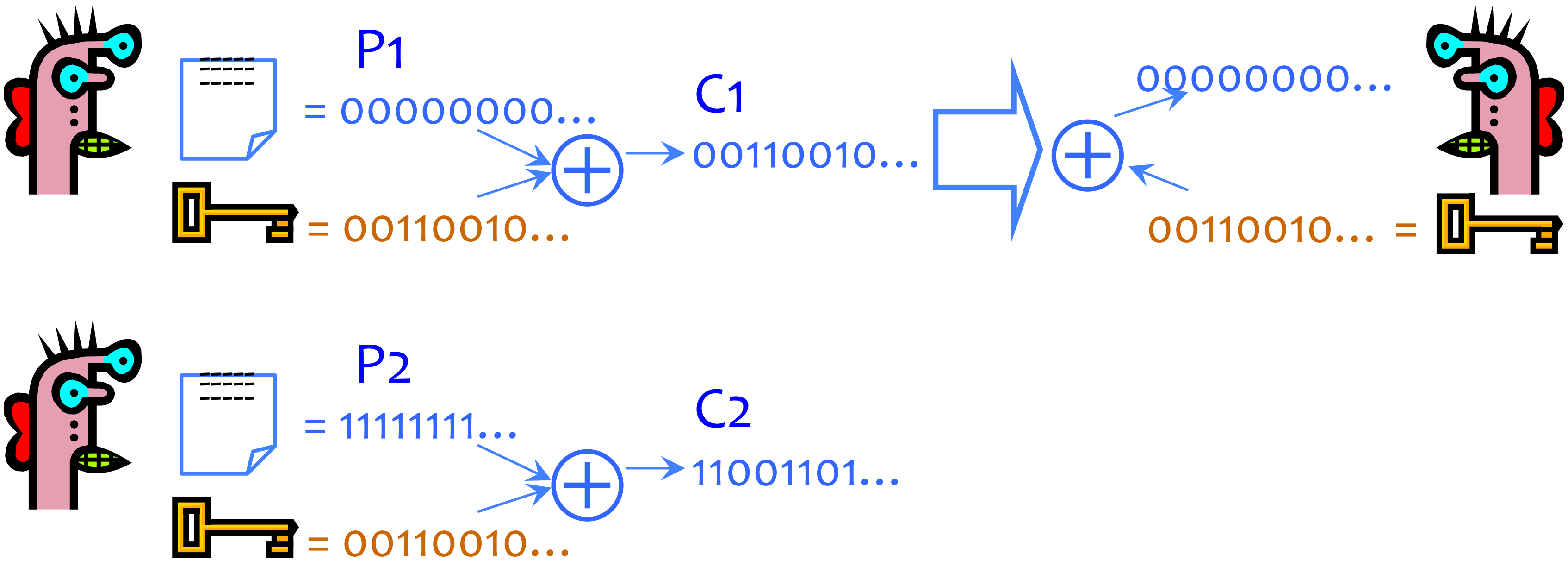
Advantages of One-Time Pad

- Easy to compute
  - Encryption and decryption are the same operation
  - Bitwise XOR is very cheap to compute
- As secure as theoretically possible
  - 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 One-Time Pad

- Key must be as long as the plaintext
  - Impractical in most realistic scenarios
  - Still used for diplomatic and intelligence traffic
- Insecure if keys are reused
  - Attacker can obtain XOR of plaintexts
- Does not guarantee integrity
  - One-time pad only guarantees confidentiality
  - Attacker cannot recover plaintext, but can easily change it to something else

Dangers of Reuse

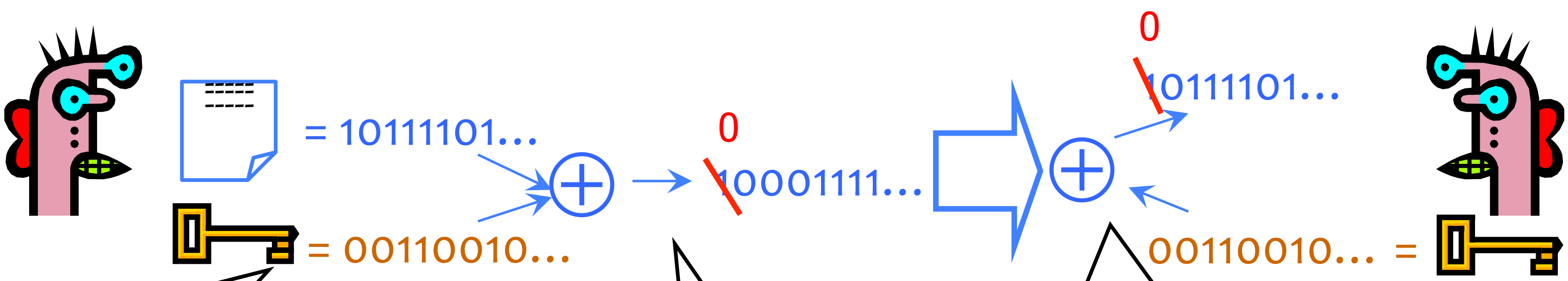


Learn relationship between plaintexts

$C1 \oplus C2 = (P1 \oplus K) \oplus (P2 \oplus K) =$

$(P1 \oplus P2) \oplus (K \oplus K) = P1 \oplus P2$

No Integrity



Key is a random bit sequence as long as the plaintext

Encrypt by bitwise XOR of plaintext and key:
ciphertext = plaintext ⊕ key

Decrypt by bitwise XOR of ciphertext and key:
ciphertext ⊕ key =
(plaintext ⊕ key) ⊕ key =
plaintext ⊕ (key ⊕ key) =
plaintext

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, stream ciphers
  - Single key can be re-used (with some restrictions)
  - Use them in ways that provide integrity

Stream Ciphers

- One-time pad:
  Ciphertext(Key,Message)=Message⊕Key
  – Key must be a random bit sequence as long as message

- Idea: replace "random" with "pseudo-random"

Stream Ciphers

- One-time pad:
  Ciphertext(Key,Message)=Message $\oplus$ Key

- Stream cipher:
  Ciphertext(Key,Message)=
  Message $\oplus$ PRNG(Key)

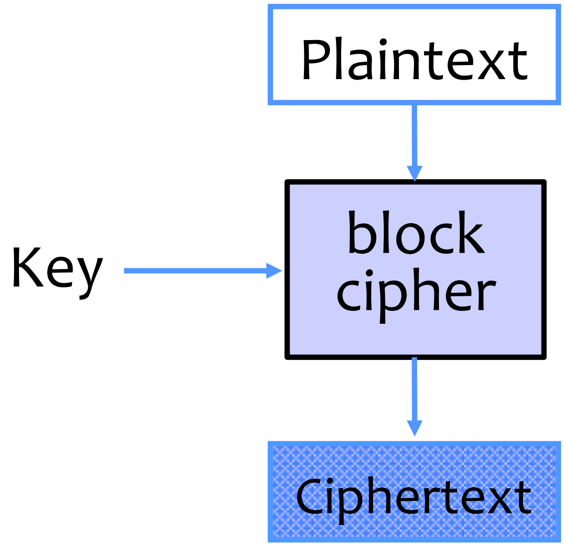
Stream Ciphers

- One time pad, replace "random" with "pseudo-random"
  - Use a pseudo-random number generator (PRNG)
  - PRNG takes a short, truly random secret seed and expands it into a long "random-looking" sequence
    - E.g., 128-bit seed into a 10^6-bit pseudo-random sequence

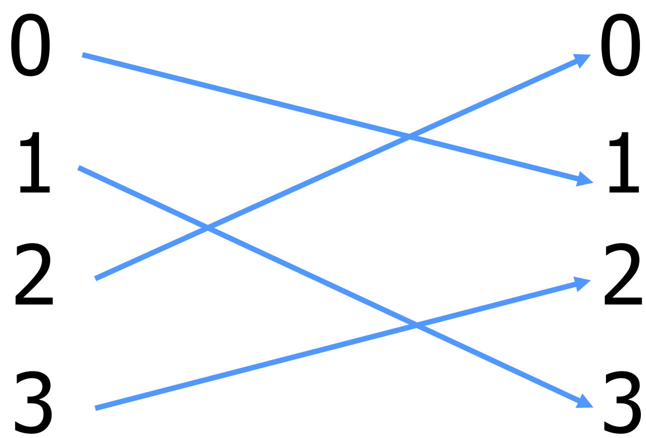
No efficient algorithm can tell this sequence from truly random

Block Ciphers

- Operates on a single chunk (“block”) of plaintext
  - For example, 64 bits for DES, 128 bits for AES
  - Each key defines a different permutation
  - Same key is reused for each block (can use short keys)



Permutations



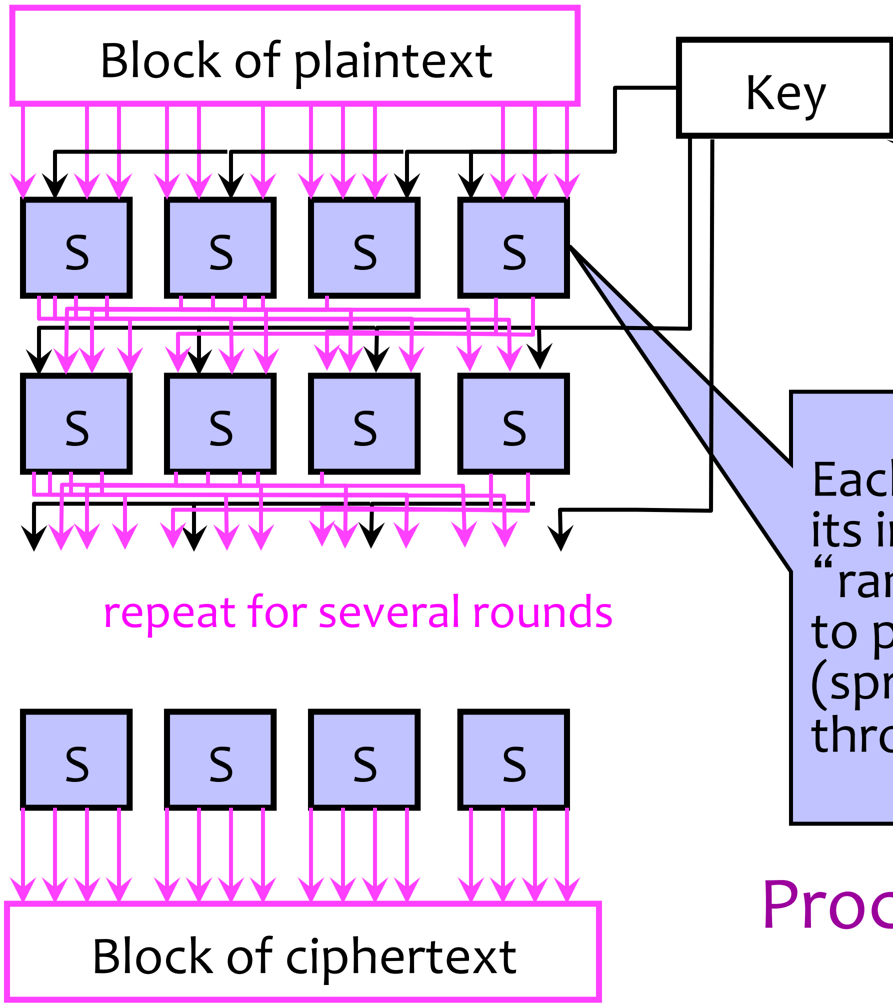
- For N-bit input, $2^N!$ possible permutations
- Idea for how to use a keyed permutation: split plaintext into blocks; for each block use secret key to pick a permutation
  - Without the key, permutation should "look random"

Block Cipher Security

- Result should look like a random permutation on the inputs
  - Recall: not just shuffling bits. N-bit block cipher permutes over 2^N inputs.

- Only computational guarantee of secrecy
  - Not impossible to break, just very expensive
    - If there is no efficient algorithm (unproven assumption!), then can only break by brute-force, try-every-possible-key search
  - Time and cost of breaking the cipher exceed the value and/or useful lifetime of protected information

Block Cipher Operation (Simplified)



Add some secret key bits to provide confusion

Each S-box transforms its input bits in a "random-looking" way to provide diffusion (spread plaintext bits throughout ciphertext)

Procedure must be reversible (for decryption)

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

DES and 56 bit keys

- 56 bit keys are quite short
- 1999: EFF DES Crack + distributed machines
  - < 24 hours to find DES key
- DES ---> 3DES
  - 3DES: DES + inverse DES + DES (with 2 or 3 diff keys)

DES and 56 bit keys

- 56 bit keys are quite short

Key Size (bits)	Number of Alternative Keys	Time required at 1 encryption/μs	Time required at 10^6 encryptions/μs
32	$2^{32} = 4.3 \times 10^9$	$2^{31}\ \mu s = 35.8$ minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	$2^{55}\ \mu s = 1142$ years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	$2^{127}\ \mu s = 5.4 \times 10^{24}$ years	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	$2^{167}\ \mu s = 5.9 \times 10^{36}$ years	5.9×10^{30} years
26 characters (permutation)	$26! = 4 \times 10^{26}$	$2 \times 10^{26}\ \mu s = 6.4 \times 10^{12}$ years	6.4×10^6 years

- 1999: EFF DES Crack + distributed machines
  - < 24 hours to find DES key
- DES ---> 3DES
  - 3DES: DES + inverse DES + DES (with 2 or 3 diff keys)

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

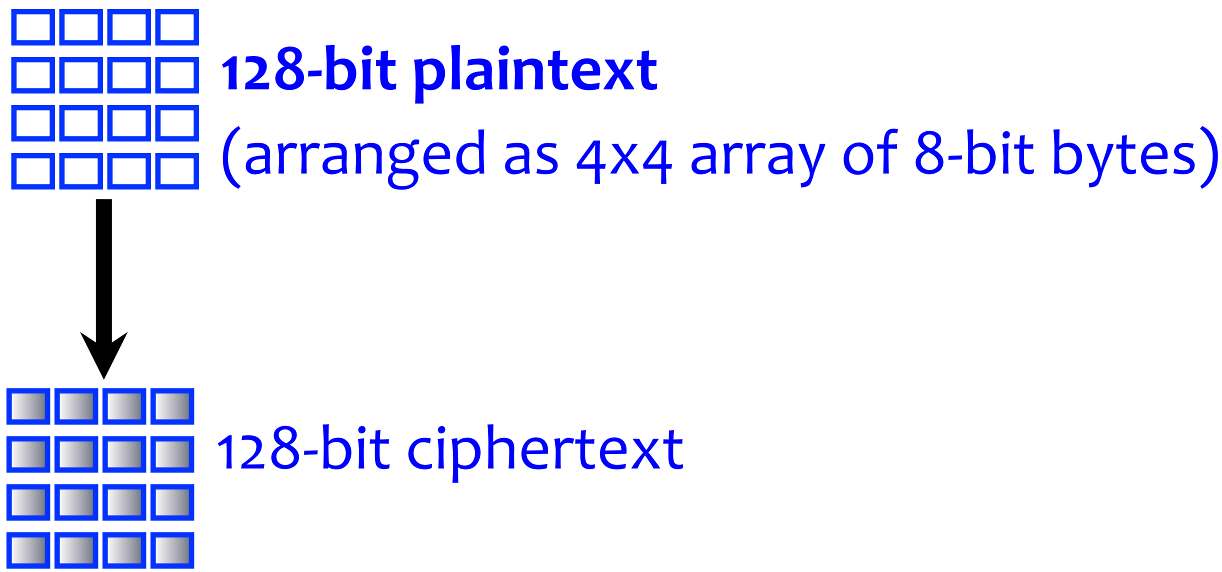
- **AES: Advanced Encryption Standard**
  - New federal standard as of 2001
    - NIST: National Institute of Standards & Technology
  - Based on the Rijndael algorithm
    - Selected via an open process
  - 128-bit blocks, keys can be 128, 192 or 256 bits

Block Ciphers Work on Fixed Length Blocks of Message

- How do you encrypt a short message?

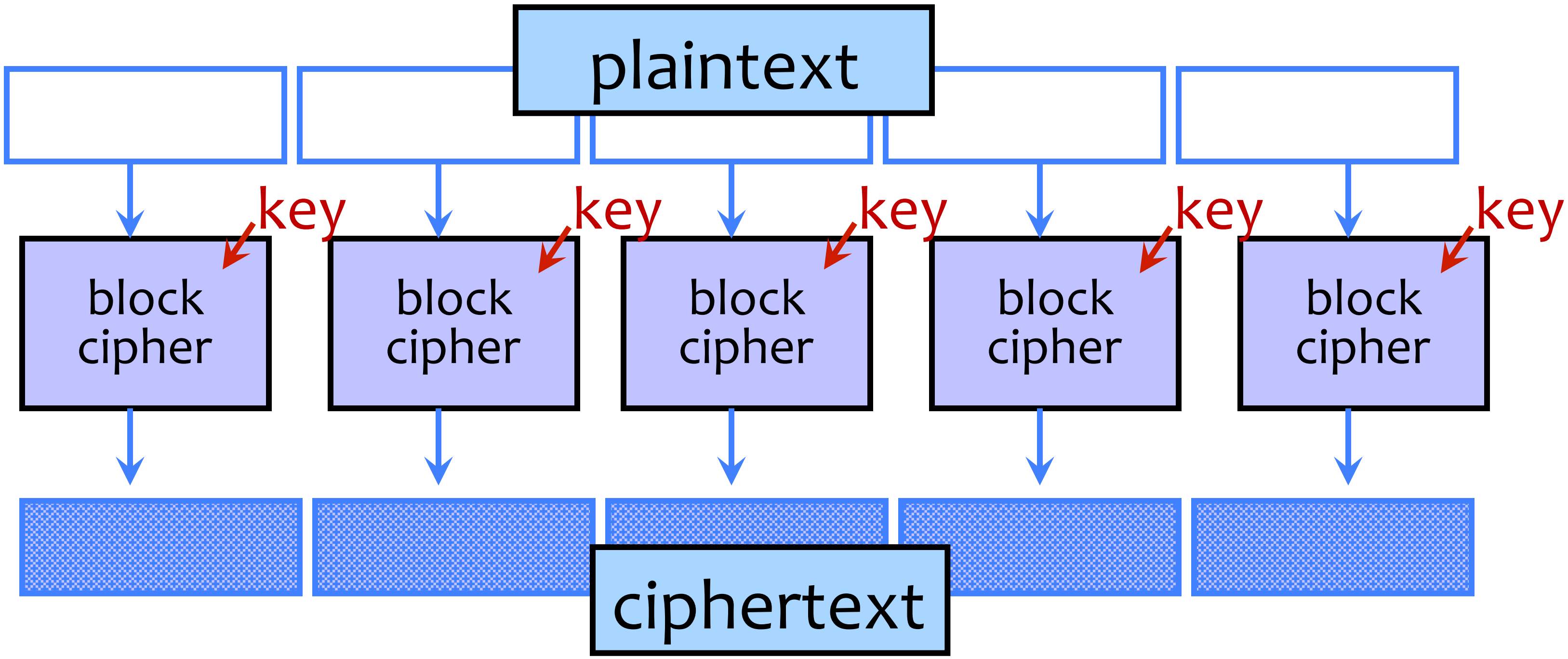
Encrypting a Large Message

- So, we've got a good block cipher, but our plaintext is larger than 128-bit block size



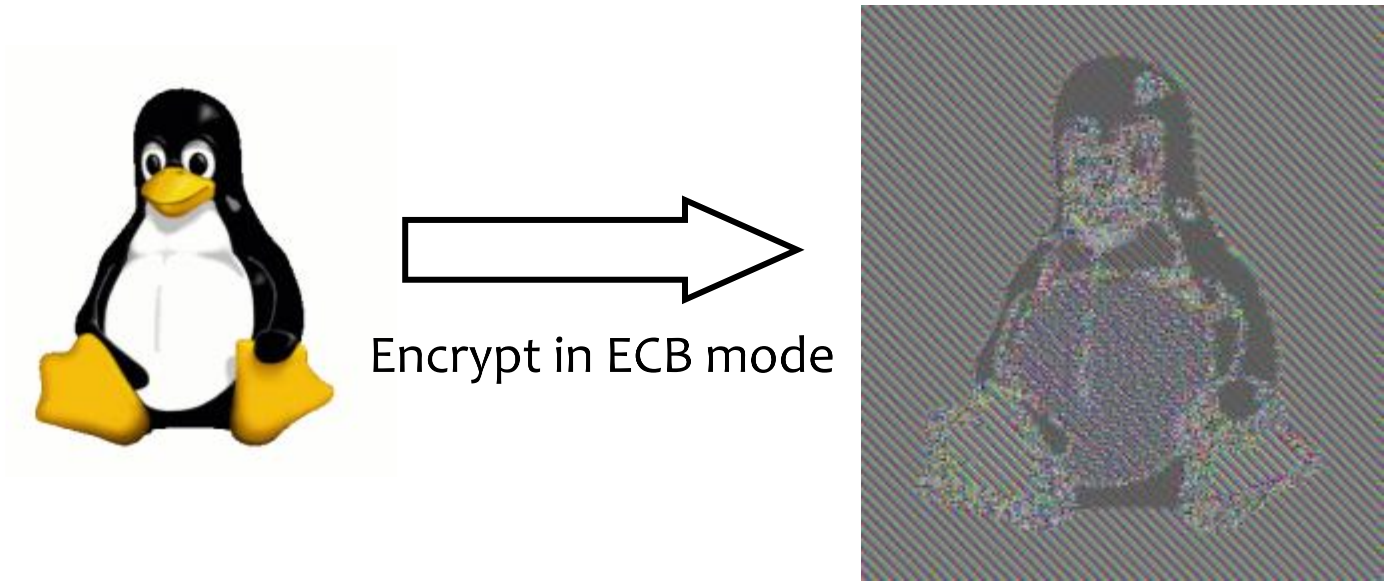
- What should we do?

Electronic Code Book (ECB) Mode



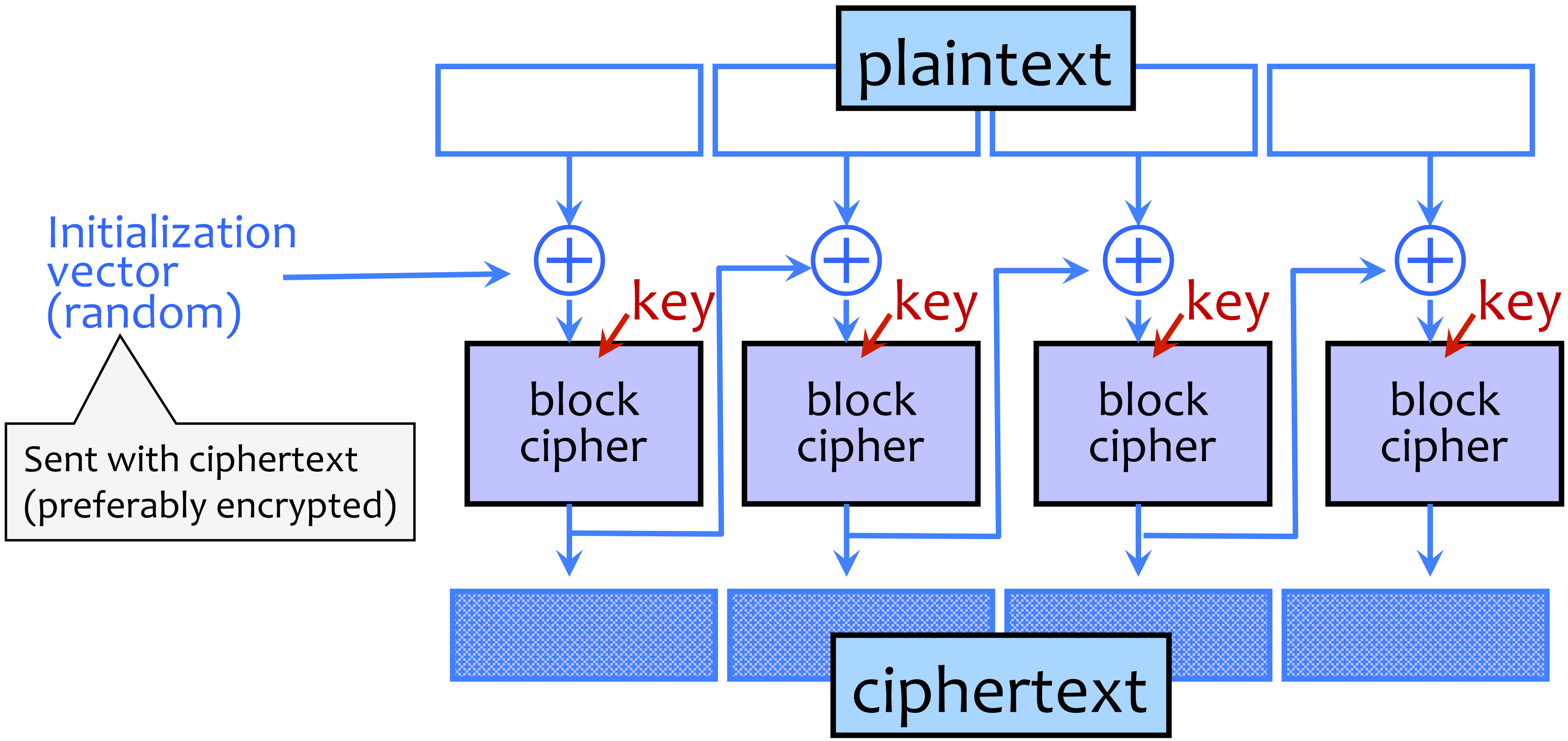
- Identical blocks of plaintext produce identical blocks of ciphertext
- No integrity checks: can mix and match blocks

Information Leakage in ECB Mode



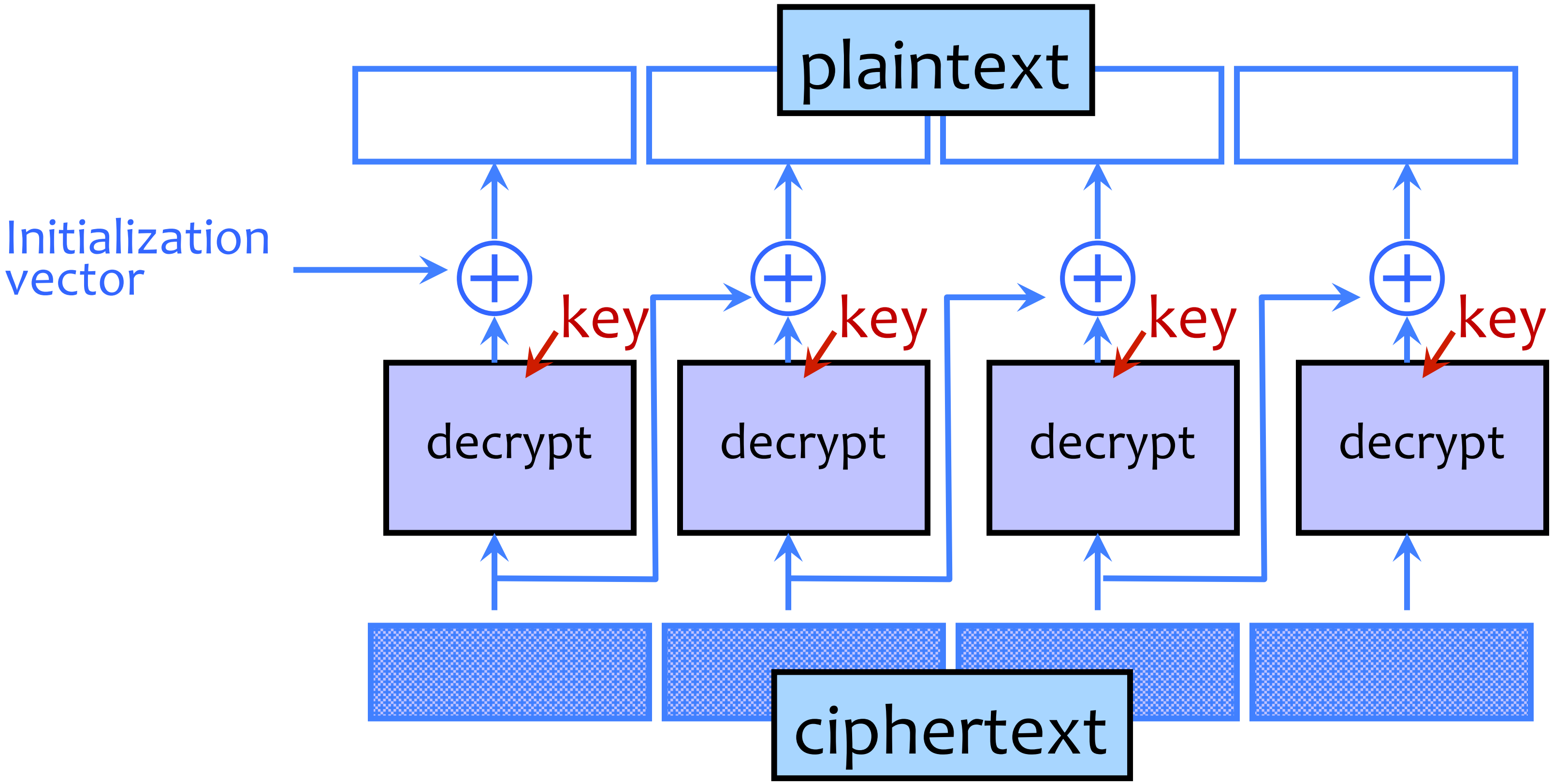
[Wikipedia]

Cipher Block Chaining (CBC) Mode: Encryption

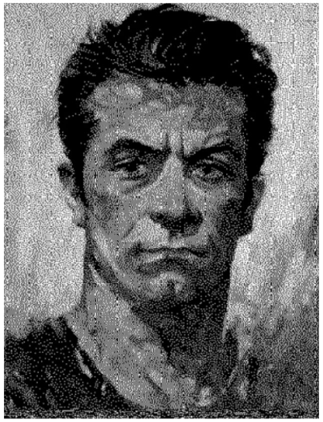


- Identical blocks of plaintext encrypted differently
- Last cipherblock depends on entire plaintext
  - Still does not guarantee integrity

CBC Mode: Decryption

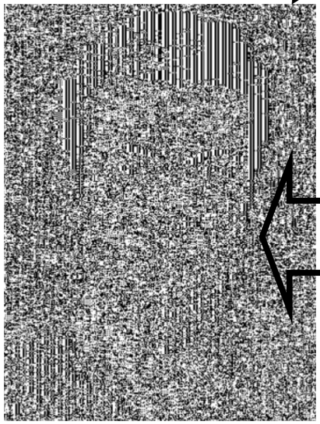


ECB vs. CBC

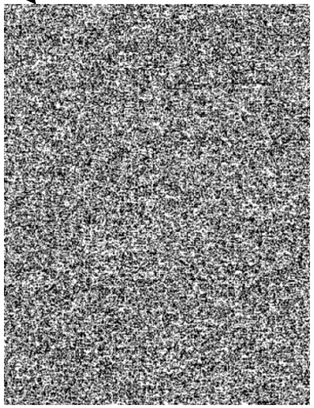


AES in ECB mode

AES in CBC mode

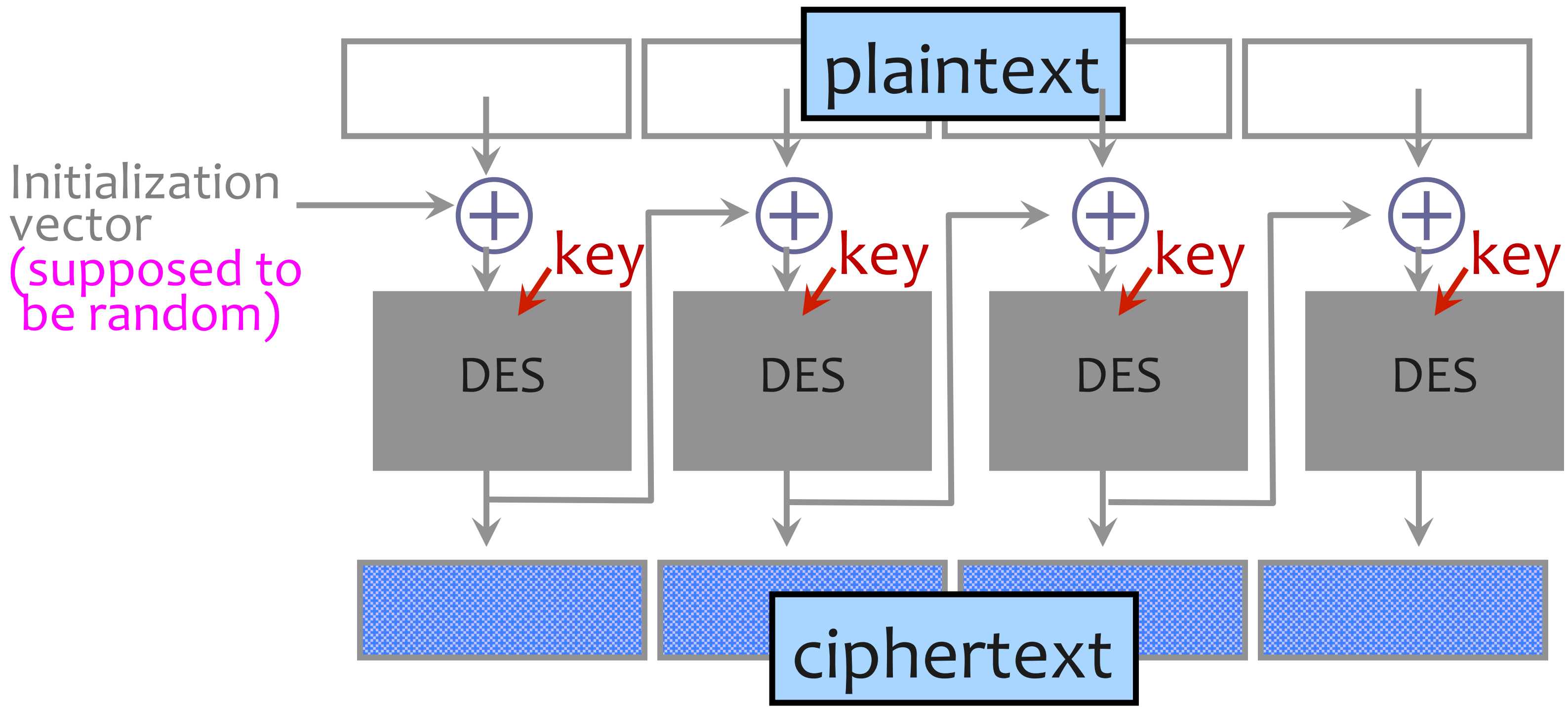


Similar plaintext blocks produce similar ciphertext blocks (not good!)



[Picture due to Bart Preneel]

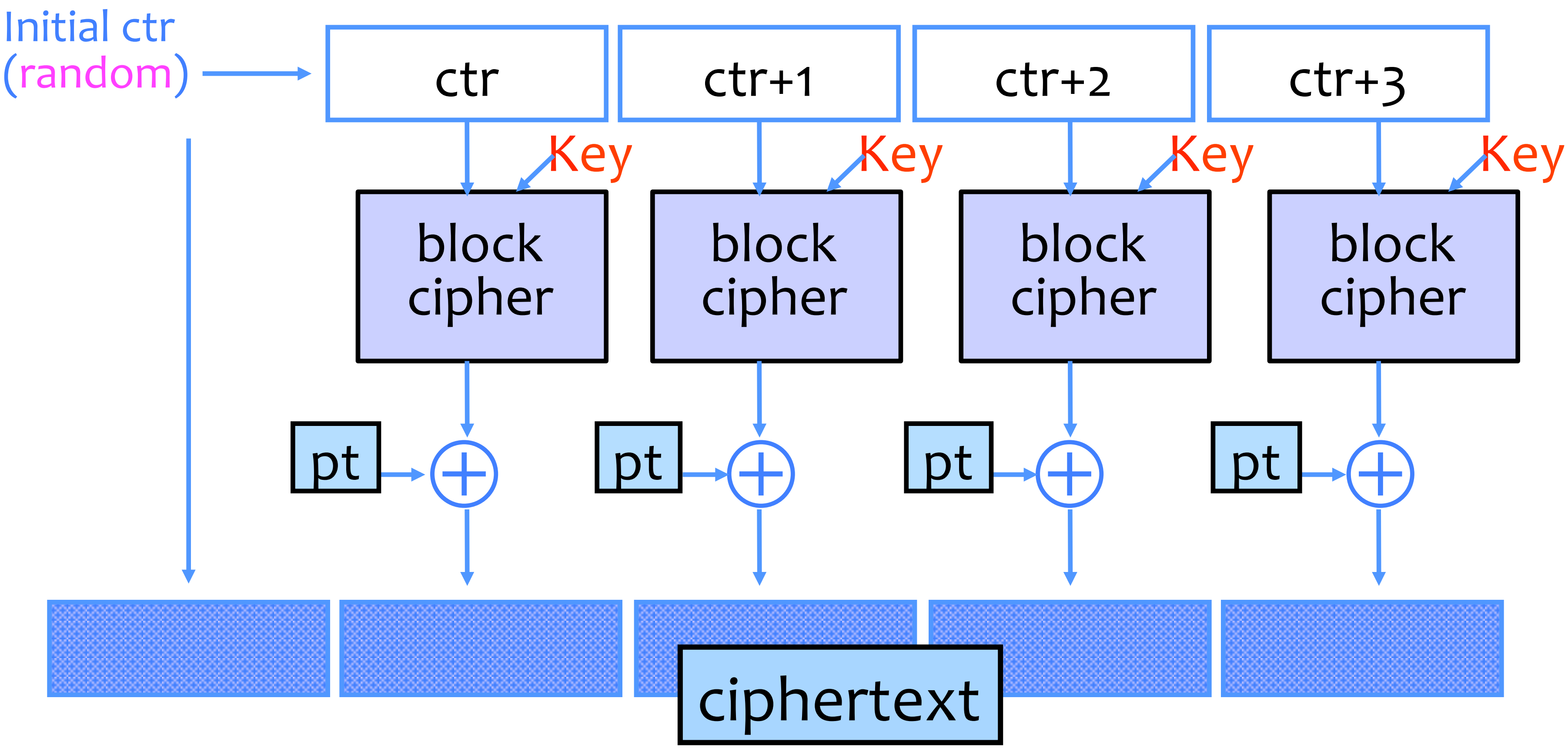
CBC and Electronic Voting



Found in the source code for Diebold voting machines:

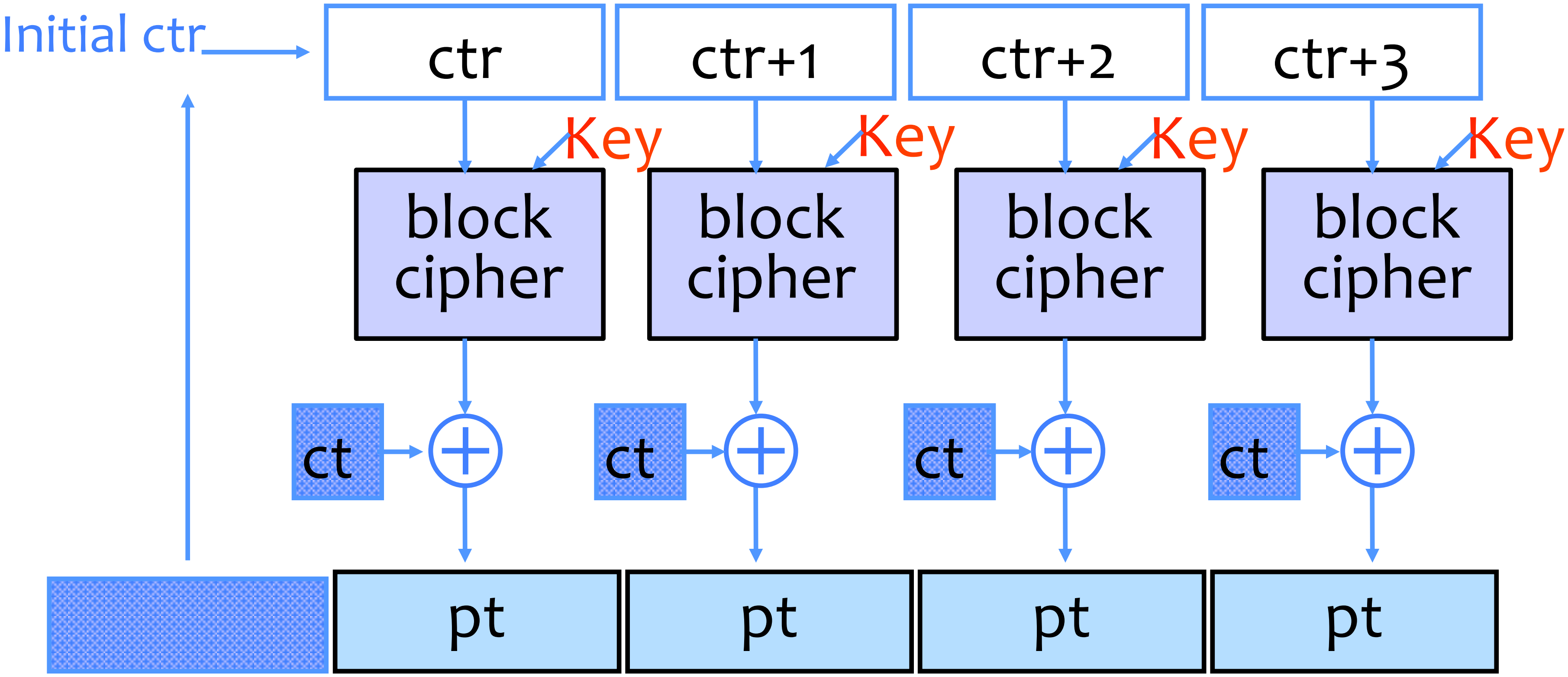
```
DesCBCEncrypt((des_c_block*)tmp, (des_c_block*)record.m_Data,
        totalSize, DESKEY, NULL, DES_ENCRYPT)
```

Counter Mode (CTR): Encryption



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

Counter Mode (CTR): Decryption



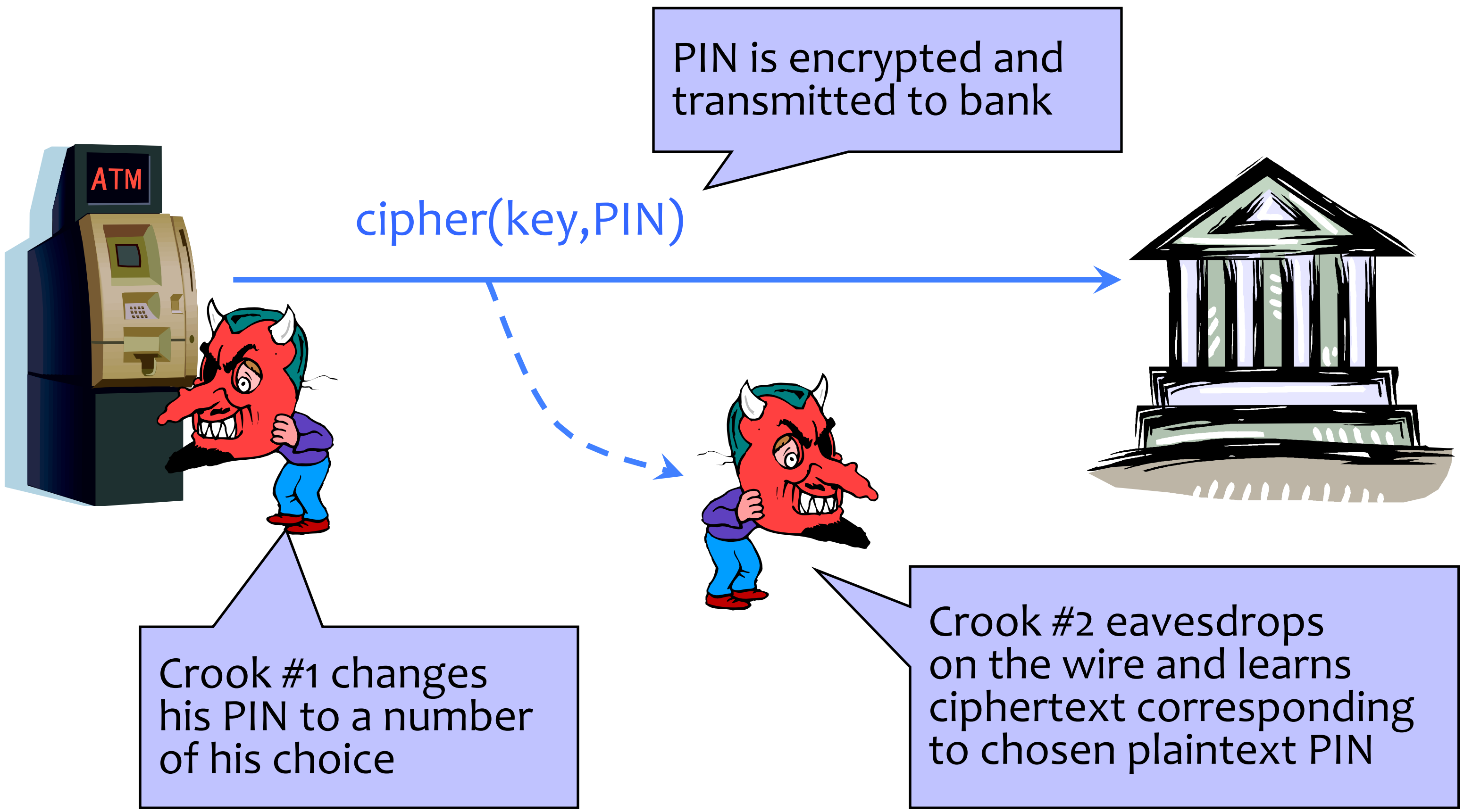
When is an Encryption Scheme "Secure"?

- 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?

- Attackers knows ciphertext and encryption algthm
  - What else does the attacker know? Depends on the application in which the cipher is used!
- Ciphertext-only attack
- KPA: Known-plaintext attack (stronger)
  - Knows some plaintext-ciphertext pairs
- CPA: Chosen-plaintext attack (even stronger)
  - Can obtain ciphertext for any plaintext of his choice
- CCA: Chosen-ciphertext attack (very strong)
  - Can decrypt any ciphertext except the target

Chosen Plaintext Attack



… repeat for any PIN value

Very Informal Intuition

Minimum security requirement for a modern encryption scheme

- Security against chosen-plaintext attack (CPA)
  - Ciphertext leaks no information about the plaintext
  - Even if the attacker correctly guesses the plaintext, he cannot verify his guess
  - Every ciphertext is unique, encrypting same message twice produces completely different ciphertexts
- Security against chosen-ciphertext attack (CCA)
  - Integrity protection – it is not possible to change the plaintext by modifying the ciphertext

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
- Also, want a strong definition, that implies other definitions (like not being able to obtain key)