

Symmetric-Key Encryption

CS 161: Computer Security

Prof. Raluca Ada Popa

Feb 7, 2019

Announcements

- Will try to post lecture slides without answers before class and complete slides after class.
- Proj 1 due Feb 12
- Hmw 2 (crypto) due Feb 17
- Midterm 1: Feb 21 7-9pm, will cover memory safety and all of crypto

Recall: Block cipher

A function $E : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$. Once we fix the key K , we get

$E_K : \{0, 1\}^n \rightarrow \{0, 1\}^n$ defined by $E_K(M) = E(K, M)$.

Three properties:

- Correctness:
 - $E_K(M)$ is a permutation (bijective/ one-to-one function)
- Efficiency
- Security

Block cipher security

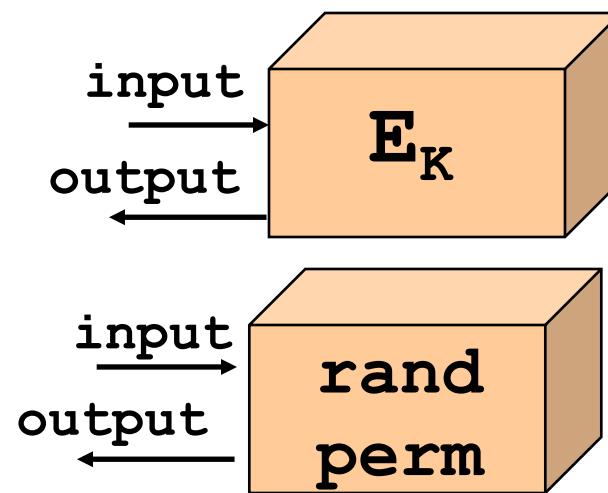
For an unknown key K , E_K “behaves” like a random permutation

For all polynomial-time attackers, for a randomly chosen key K , the attacker **cannot distinguish** E_K from a random permutation

Block cipher: security game

- Attacker is given two boxes, one for E_K and one for a random permutation (also called “oracles”)
- Attacker does not know which is which (they were shuffled randomly)
- Attacker can give inputs to each box, look at the output, as many times as he/she desires
- Attacker must guess which is E_K

??? Which is E_K ???



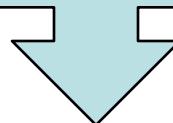
Security game

For all polynomial-time attackers,

$$\Pr[\text{attacker wins game}] \leq \frac{1}{2} + \text{negl}$$

Security

For an unknown key K , E_K “behaves” like a random permutation



Q: If the attacker receives $E_K(x)$ and nothing else about x , can he determine x ?

A: No. If he could, he could distinguish the block cipher from a random permutation

Similarly, if the attacker receives only $E_K(x_1), E_K(x_2), \dots, E_K(x_n)$. The only information he sees is if any $x_i = x_j$ but not their values

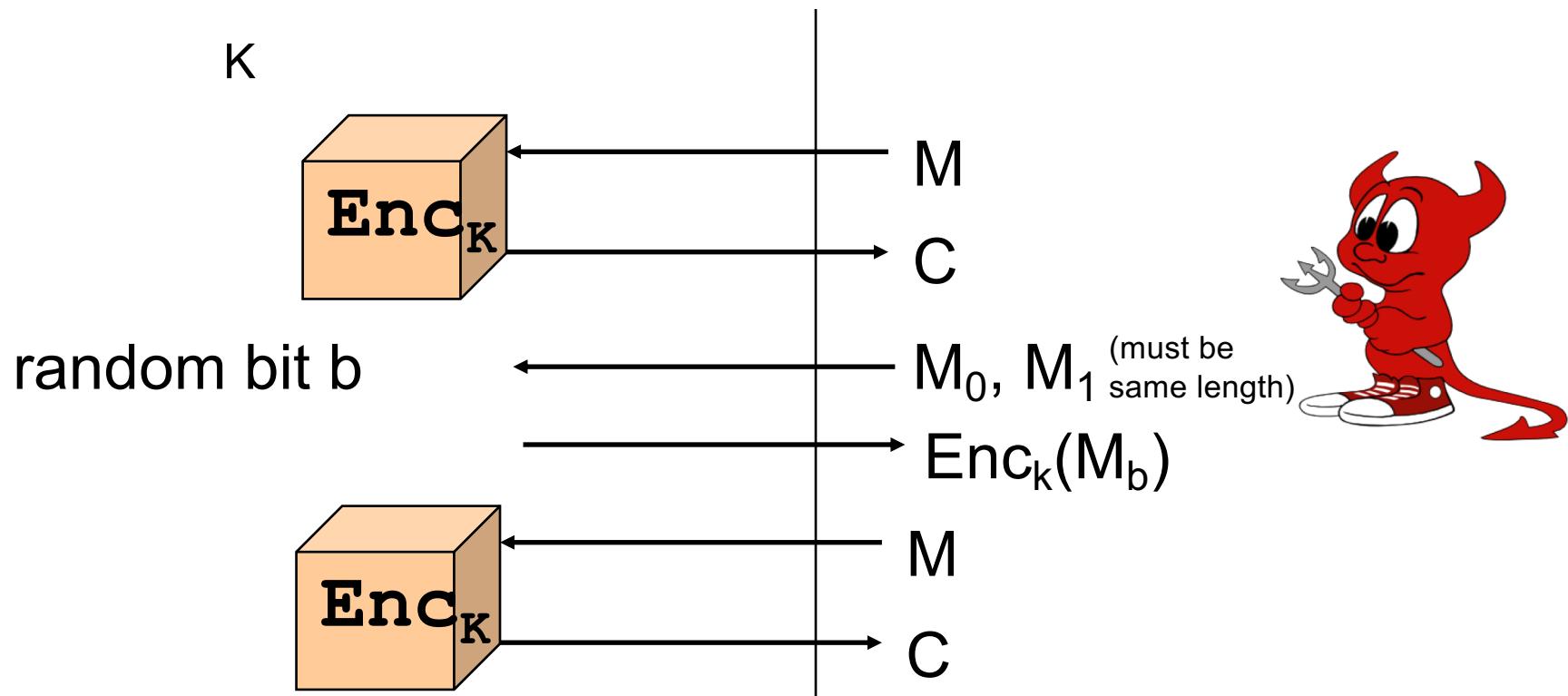
So block ciphers provide some confidentiality, but not enough for IND-CPA (because they have this deterministic leakage)

Desired security: Indistinguishability under chosen plaintext attack (IND-CPA)

- Strong security definition
- Nothing leaks about the encrypted value other than its length

IND-CPA (Indistinguishability under chosen plaintext attack)

Challenger



Here is my guess: b'

IND-CPA

An encryption scheme is IND-CPA if
for all polynomial-time adversaries

$$\Pr[\text{Adv wins game}] \leq \frac{1}{2} + \text{negligible}$$

Note that IND-CPA requires that the encryption scheme is randomized

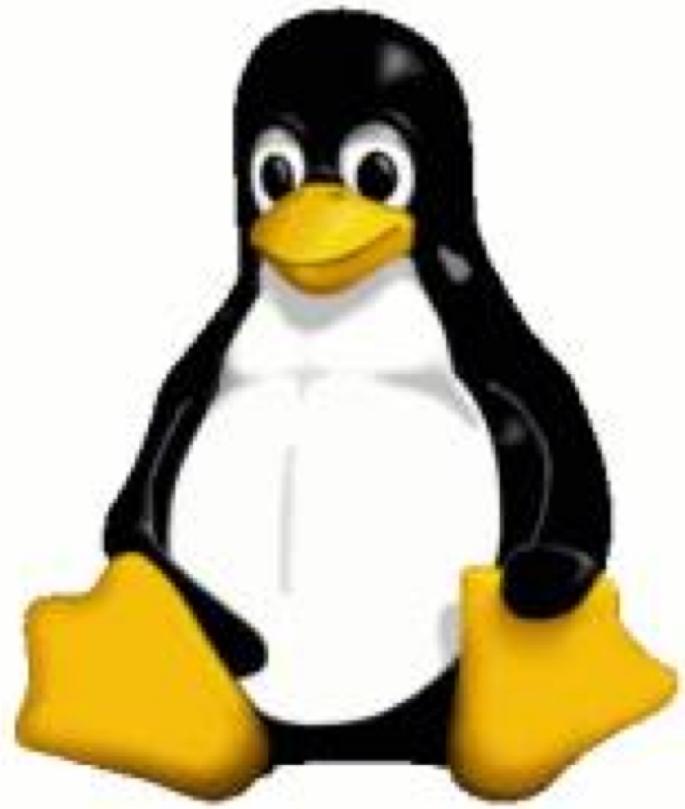
(An encryption scheme is deterministic if it outputs the same ciphertext when encrypting the same plaintext; a randomized scheme does not have this property)

Are block ciphers IND-CPA?

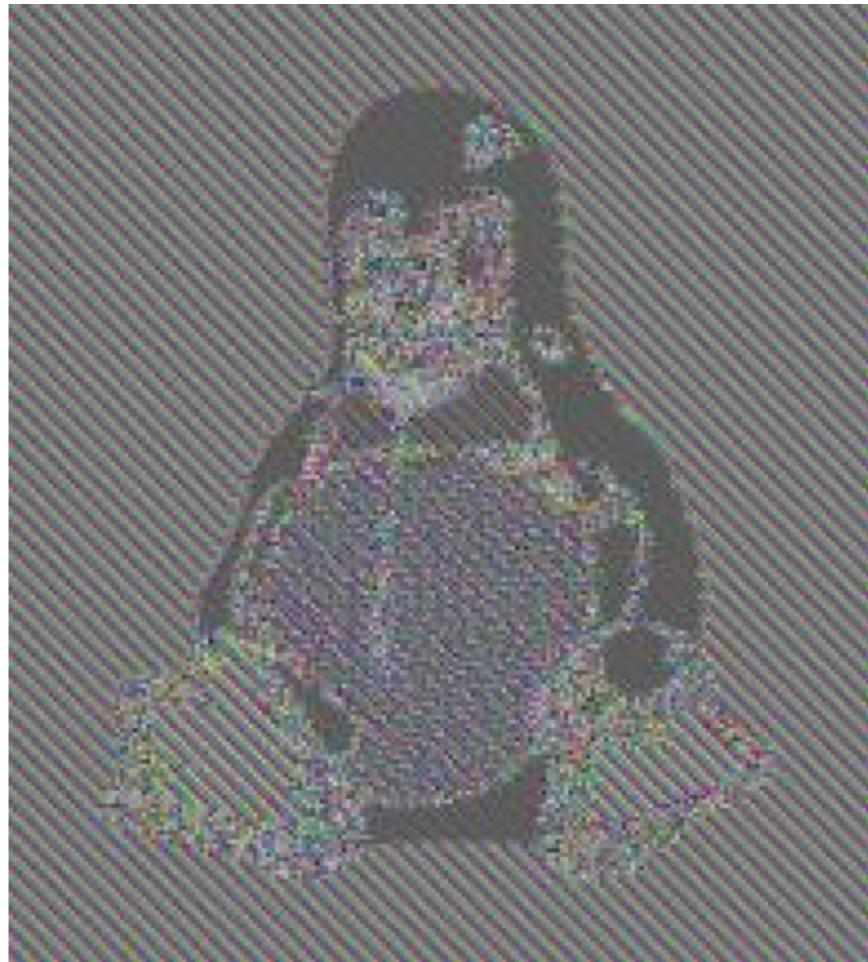
Recall: $E_K : \{0,1\}^n \rightarrow \{0,1\}^n$ is a permutation (bijective)

Q: Are block ciphers IND-CPA?

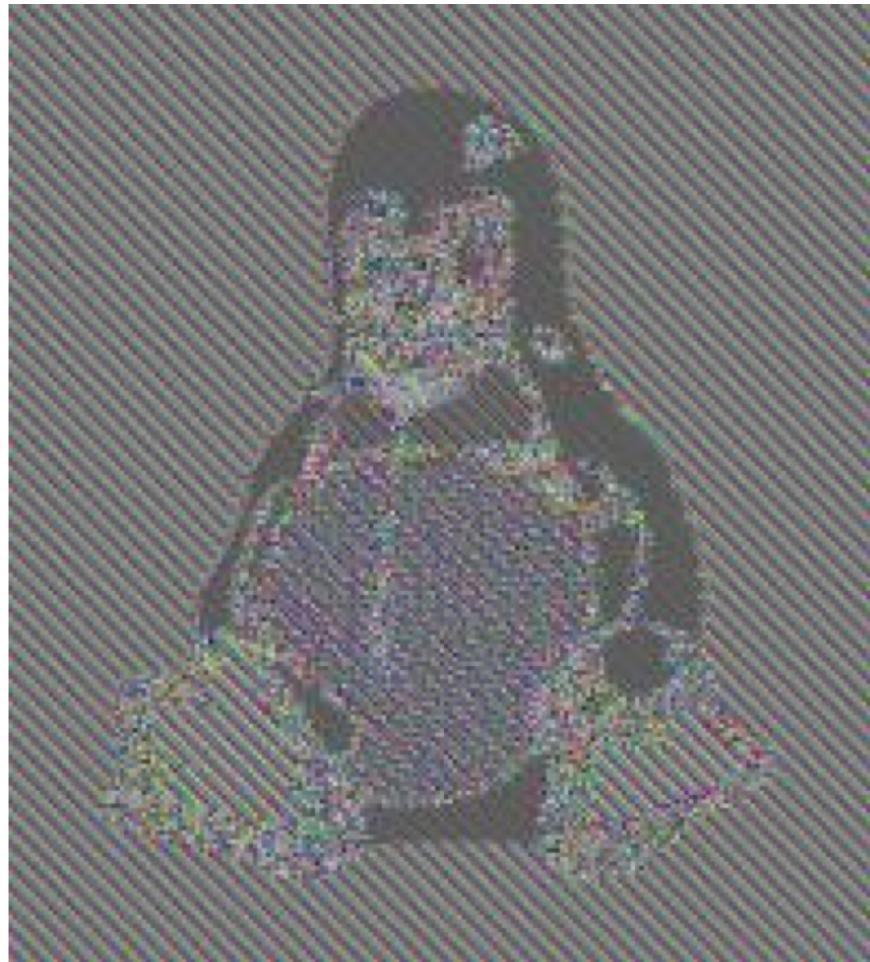
- A: No, because they are deterministic
- Here is an attacker that wins the IND-CPA game:
 - Adv asks for encryptions of “bread”, receives C_{br}
 - Then, Adv provides ($M_0 = \text{bread}$, $M_1 = \text{honey}$)
 - Adv receives C
 - If $C=C_{br}$, Adv says bit was 0 (for “bread”), else Adv says says bit was 1 (for “honey”)
 - Chance of winning is 1



Original image



Each block encrypted with a block cipher



Later (identical) message again encrypted

Why block ciphers not enough for encryption by themselves?

- Can only encipher messages of a certain size
- Not IND-CPA (If message is encrypted twice, attacker knows it is the same message)

Use block ciphers to construct symmetric-key encryption

- Want two properties:
 - IND-CPA security even when reusing the same key to encrypt many messages (unlike OTP)
 - Can encrypt messages of any length
- Build symmetric key encryption on block ciphers:
 - Can be used to encrypt long messages
 - Wants to hide that same block is encrypted twice
 - Uses block ciphers in certain modes of operation
 - There are any block ciphers besides AES

Modes of operation

Chain block ciphers **in certain modes of operation**

- Invoke block cipher multiple times on inputs related to other blocks

Need some **initial randomness IV** (initialization vector)

Q: Why?

A: To prevent the encryption scheme from being deterministic

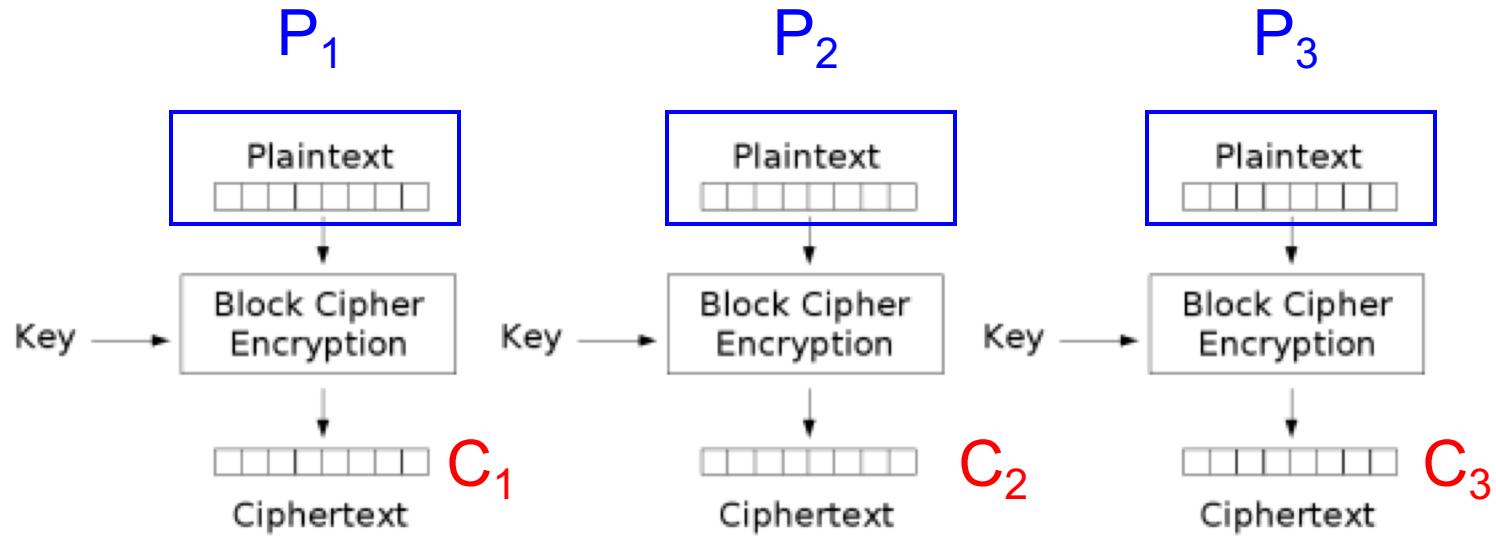
Electronic Code Book (ECB)

- Split message M in blocks P_1, P_2, \dots where each plaintext block is as large as n , the block cipher input size
 - For now assume that M is a multiple of n , but we will see how to pad if that is not the case
- Each block is a value which is substituted, like a codebook
- Each block is encoded independently of the other blocks

$$C_i = E_K(Pi)$$

ECB: Encryption

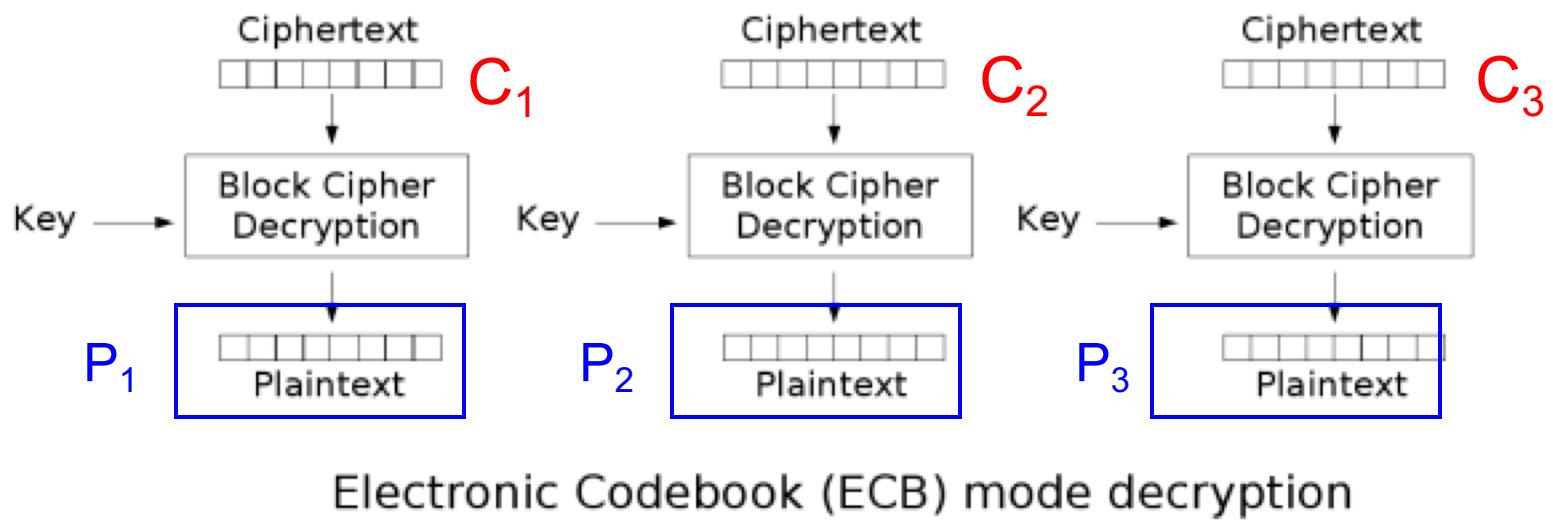
break message M into $P_1|P_2|\dots|P_m$ each of n bits (block cipher input size)



Electronic Codebook (ECB) mode encryption

$$\text{Enc}(K, P_1 | P_2 | \dots | P_m) = (C_1, C_2, \dots, C_m)$$

ECB: Decryption

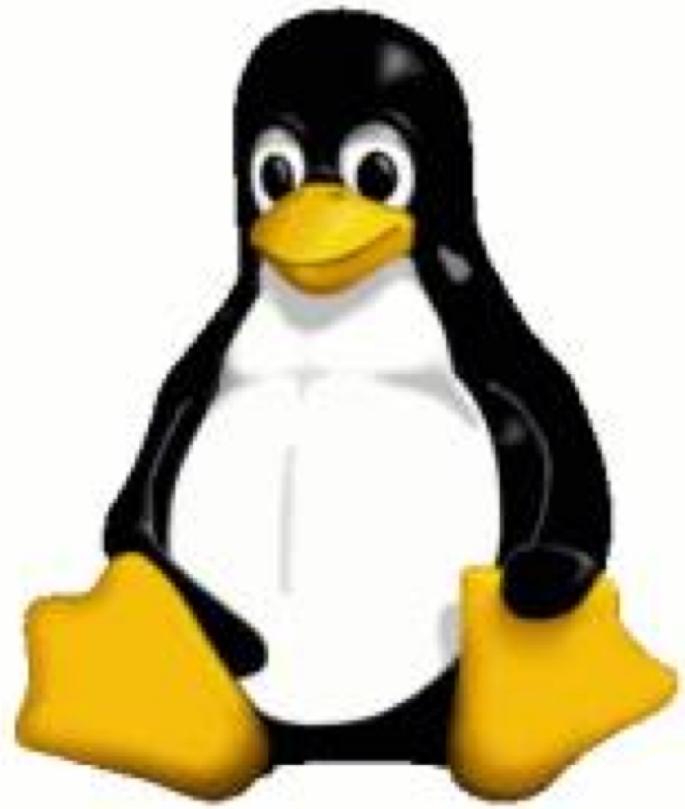


$$\text{Dec}(K, (C_1, C_2, \dots, C_n)) = (P_1, P_2, \dots, P_m)$$

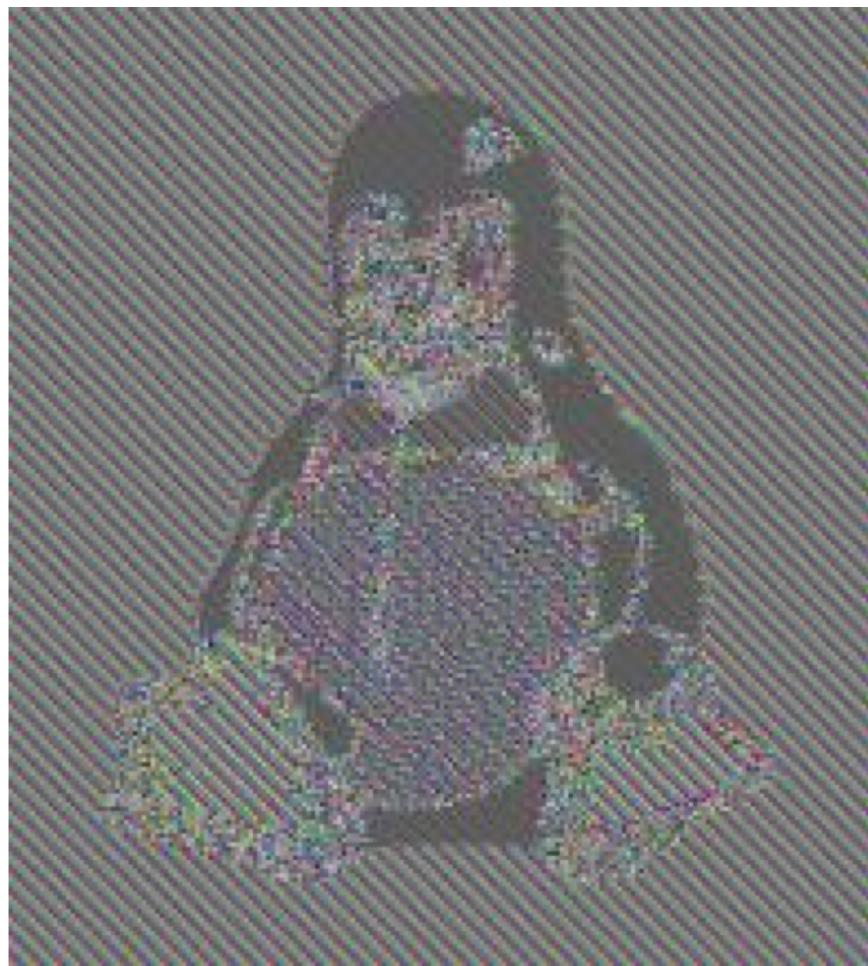
What is the problem with ECB?

Q: Does this achieve IND-CPA?

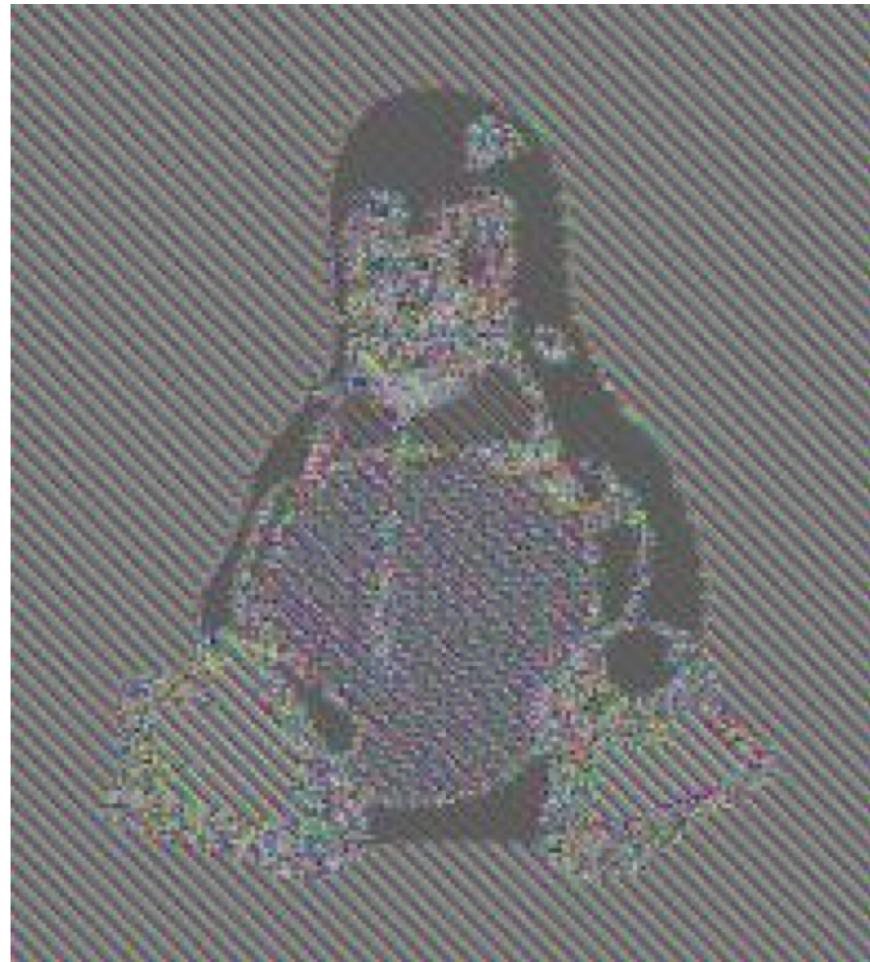
A: No, attacker can tell if $P_i = P_j$



Original image



Encrypted with ECB

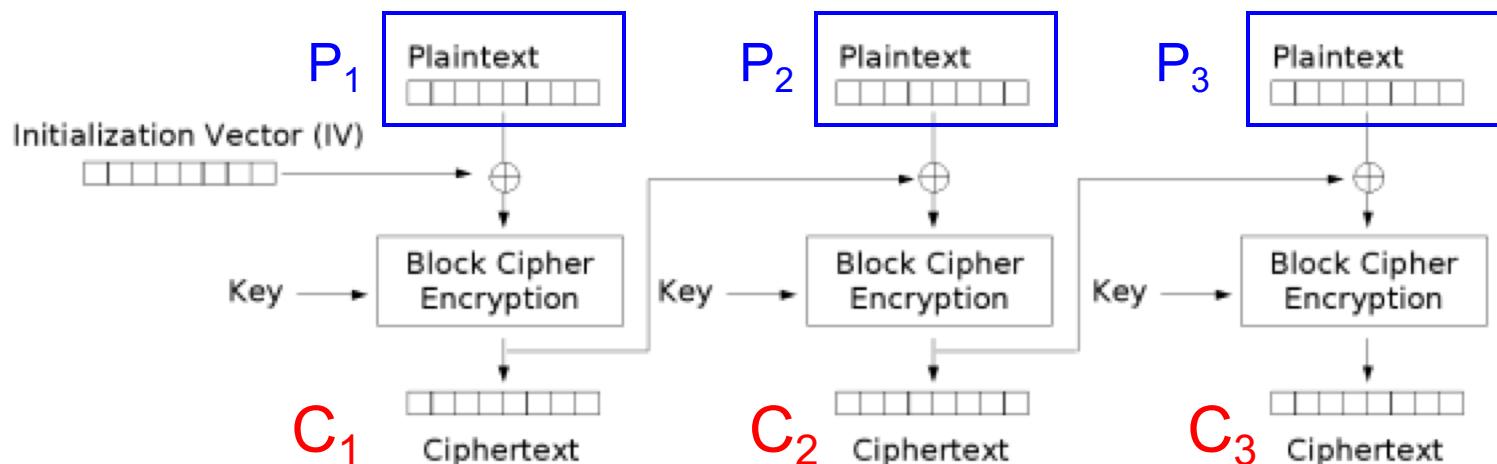


Later (identical) message again encrypted with ECB

CBC: Encryption

Break message M into $P_1|P_2|\dots|P_m$

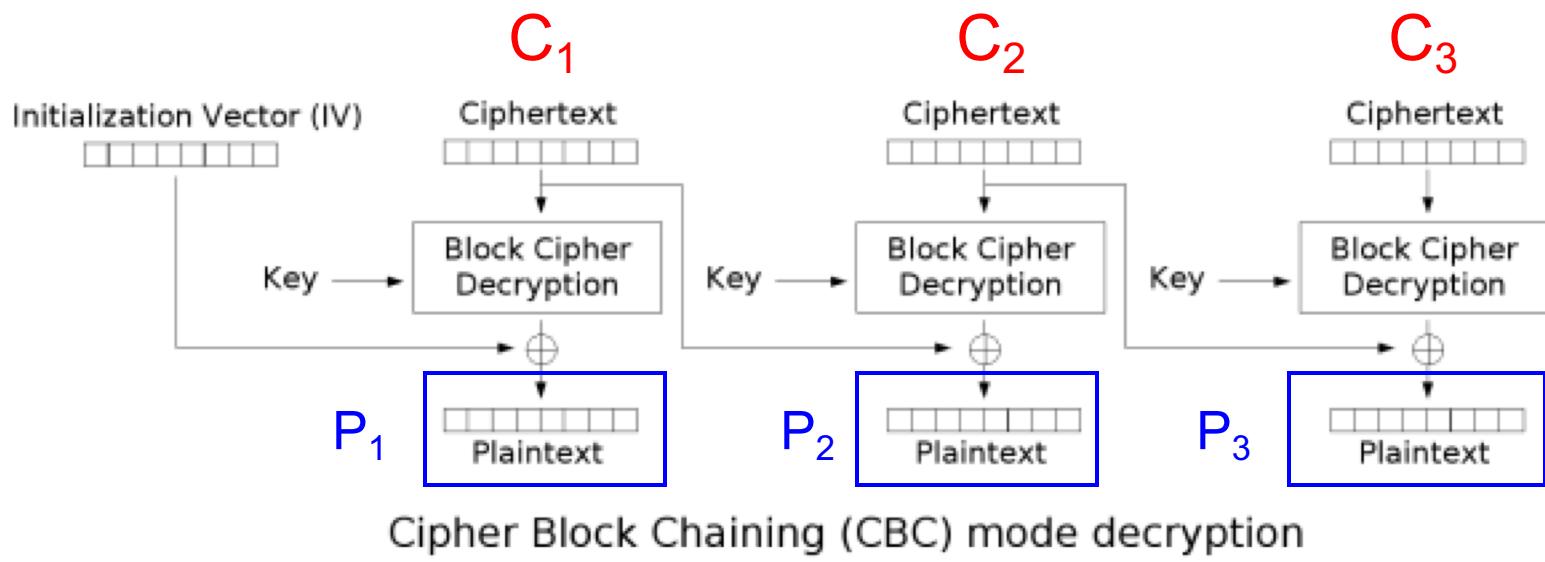
Choose a random IV (it may not repeat for messages with same P_1 , it is not secret and is included in the ciphertext)



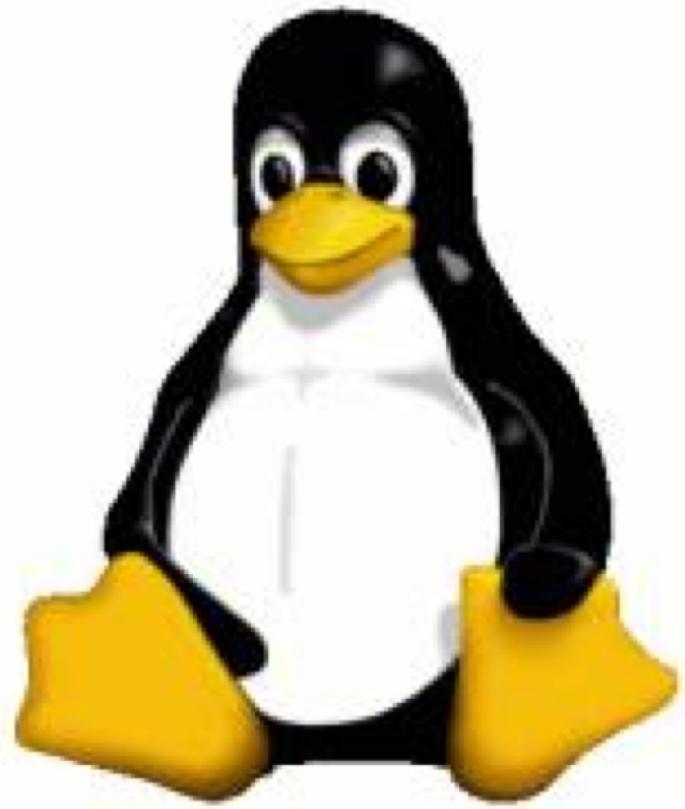
Cipher Block Chaining (CBC) mode encryption

$$\text{Enc}(K, P_1 | P_2 | \dots | P_m) = (\text{IV}, C_1, C_2, \dots, C_m)$$

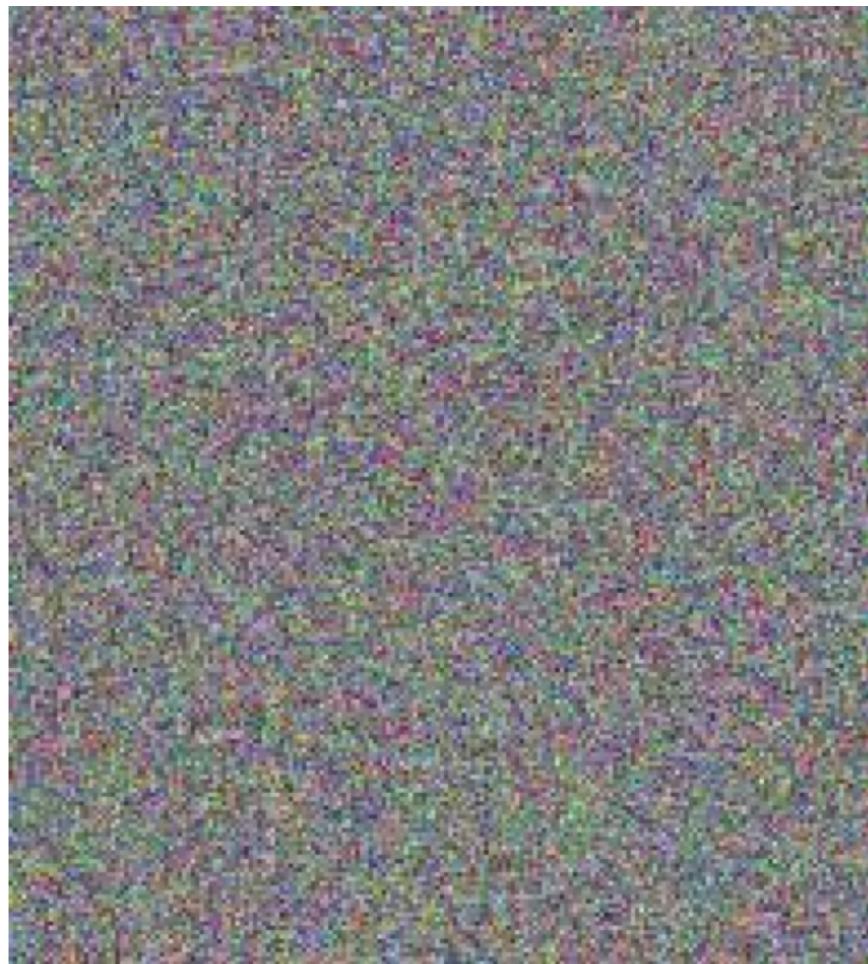
CBC: Decryption



$$\text{Dec}(K, (IV, C_1, C_2, \dots, C_m)) = (P_1, P_2, \dots, P_m)$$



Original image



Encrypted with CBC

CBC

Popular, still widely used
Achieves IND-CPA

Slight caveat: sequential encryption, hard to parallelize

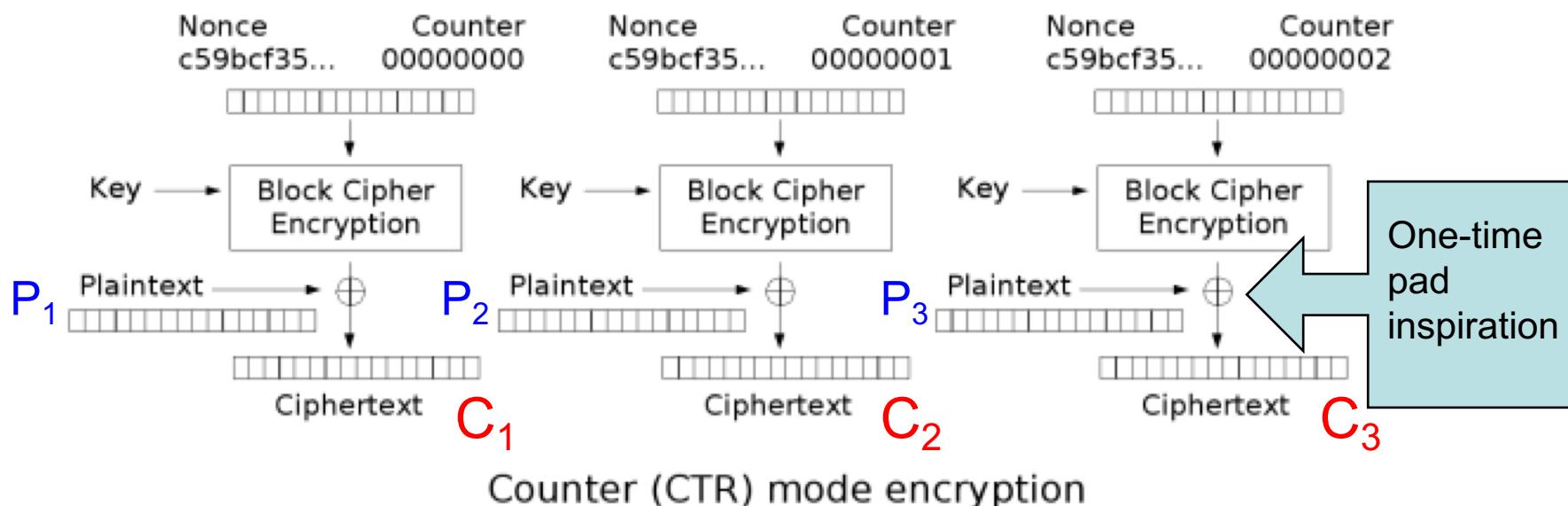
CTR mode gaining popularity

Counter mode (CTR)

CTR: Encryption

Enc(K , plaintext):

- If n is the block size of the block cipher, split the plaintext in blocks of size n : P_1, P_2, P_3, \dots
- Choose a random nonce (Nonce = Same as IV)
Important that nonce does not repeat across different encryptions (choose it at random from large space)
- Now compute:

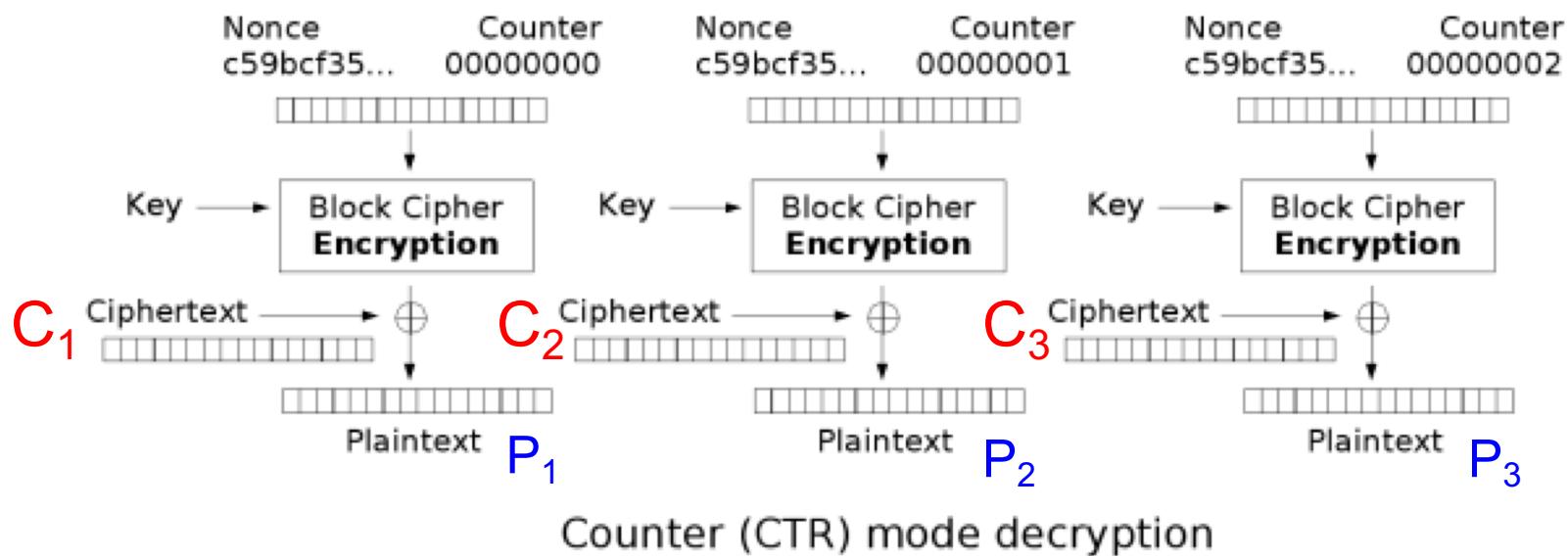


- The final ciphertext is (nonce, C_1, C_2, C_3)

CTR: Decryption

Dec(K , ciphertext=[nonce, C_1 , C_2 , C_3 , ...]):

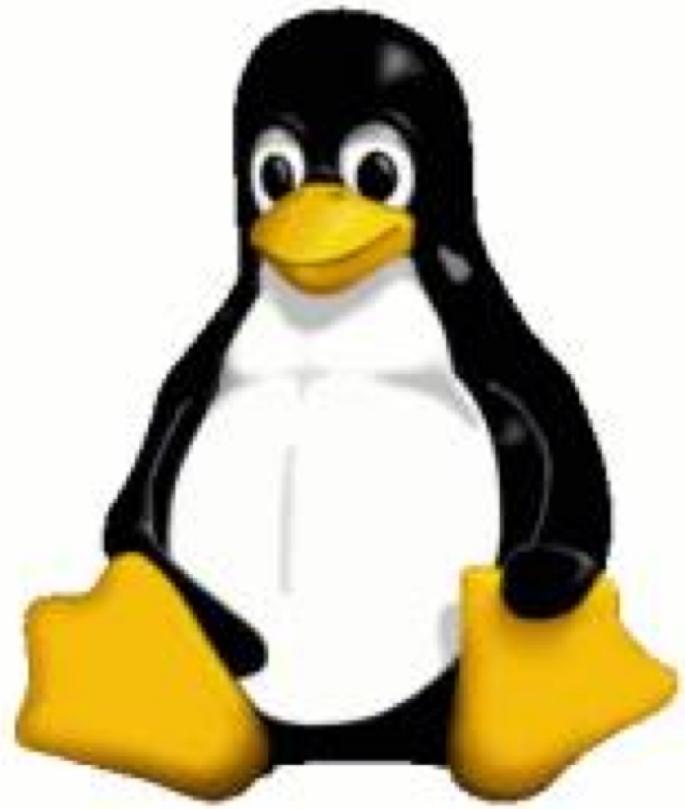
- Take nonce out of the ciphertext
- If n is the block size of the block cipher, split the ciphertext in blocks of size n : C_1 , C_2 , C_3 , ...
- Now compute this:



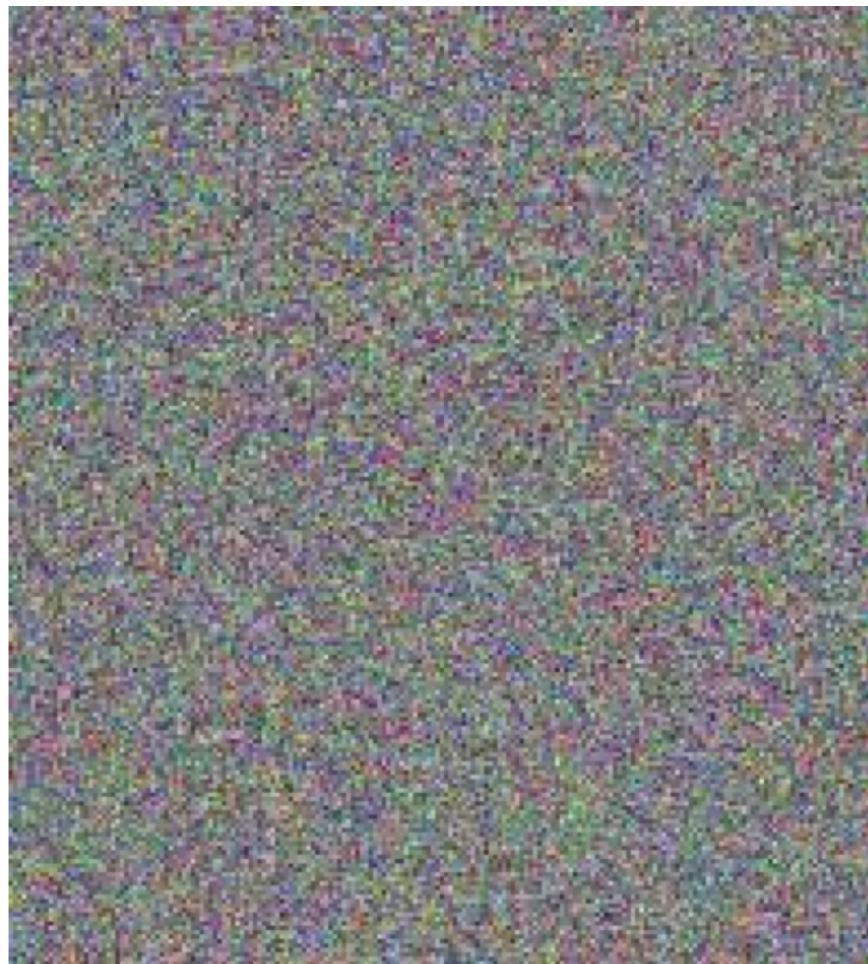
- Output the plaintext as the concatenation of P_1 , P_2 , P_3 , ...

Note, CTR decryption uses block cipher's *encryption*, not decryption

Would you like me to explain
CTR one more time?



Original image



Encrypted with CBC

CBC vs CTR

Security: If no reuse of nonce/IV, **both are IND-CPA.**

Speed: Both modes require the same amount of computation, but CTR is parallelizable for encryption as well (CBC was parallelizable for decryption but not for encryption)

Padding

If messages might not be multiple of n , the block cipher length, we pad the message before encryption and unpad after decryption.

Bad padding:

message 0000000000

n bits

Q: Why bad?

A: When unpading, it is not clear which 0s belong to the padding vs the message

Good padding:

message 1000000000

n bits

If the message is exactly n bits long, still pad by adding another n bits.

Random fact about ... John DeNero



If you attend his office hours, you will meet his assistant, who is almost always there...



Samosa



2 min break

Pseudorandom generator (PRG)

Pseudorandom Generator (PRG)

- Given a seed, it outputs a sequence of random bits
$$\text{PRG}(\text{seed}) \rightarrow \text{random bits}$$
- It can output arbitrarily many random bits

PRG security

- Can $\text{PRG}(K)$ be truly random?

No. Consider key length $|K|=k$. Have 2^k possible initial states of PRG. Deterministic from then on. There are more random states.

- A secure PRG suffices to “look” random (“pseudo”) to an attacker (no attacker can distinguish it from a random sequence)

Example of PRG: using block cipher in CTR mode

If you want m random bits, and a block cipher with E_k has n bits, apply the block cipher m/n times and concatenate the result:

$$\text{PRG}(K \mid IV) = E_k(IV|1) \mid E_k(IV|2) \mid E_k(IV|3) \dots E_k(IV|\text{ceil}(m/n)), \text{ where } \mid \text{ is concatenation}$$

Application of PRG: Stream ciphers

- Another way to construct encryption schemes
- Similar in spirit to one-time pad: it XORs the plaintext with some random bits
- But random bits are not the key (as in one-time pad) but are output of a pseudorandom generator PRG

Application of PRG: Stream cipher

Enc(K, M):

- Choose a random value IV
- $C = \text{PRG}(K | IV) \text{ XOR } M$
- Output (IV, C)

Q: How decrypt?

A: Compute $\text{PRG}(K | IV)$ and XOR with ciphertext C

Q: What is advantage over OTP?

A: Can encrypt any message length because PRG can produce any number of random bits, and multiple times because IV is chosen at random in Enc

Summary

- Desirable security: IND-CPA
- Block ciphers have weaker security than IND-CPA
- Block ciphers can be used to build IND-CPA secure encryption schemes by chaining in careful ways
- Stream ciphers provide another way to encrypt, inspired from one-time pads