Block ciphers, stream ciphers

(start on:) Asymmetric cryptography

CS 161: Computer Security

Prof. Raluca Ada Popa

Jan 31, 2018

Announcements

- Project 1 is out, due Feb 14 midnight

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

Security

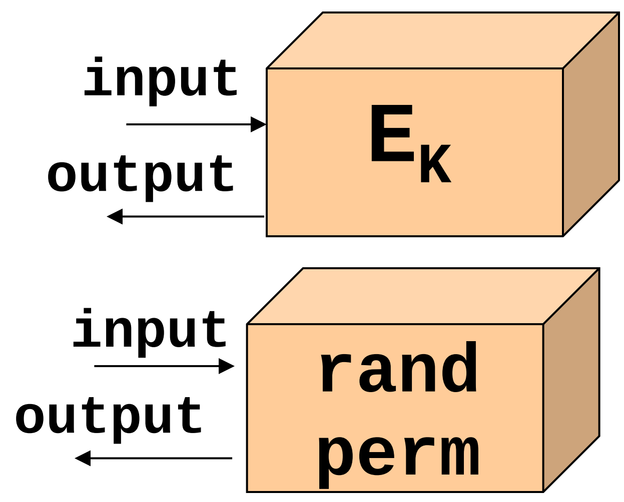
For an unknown key K, E_K “behaves” like a random permutation

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

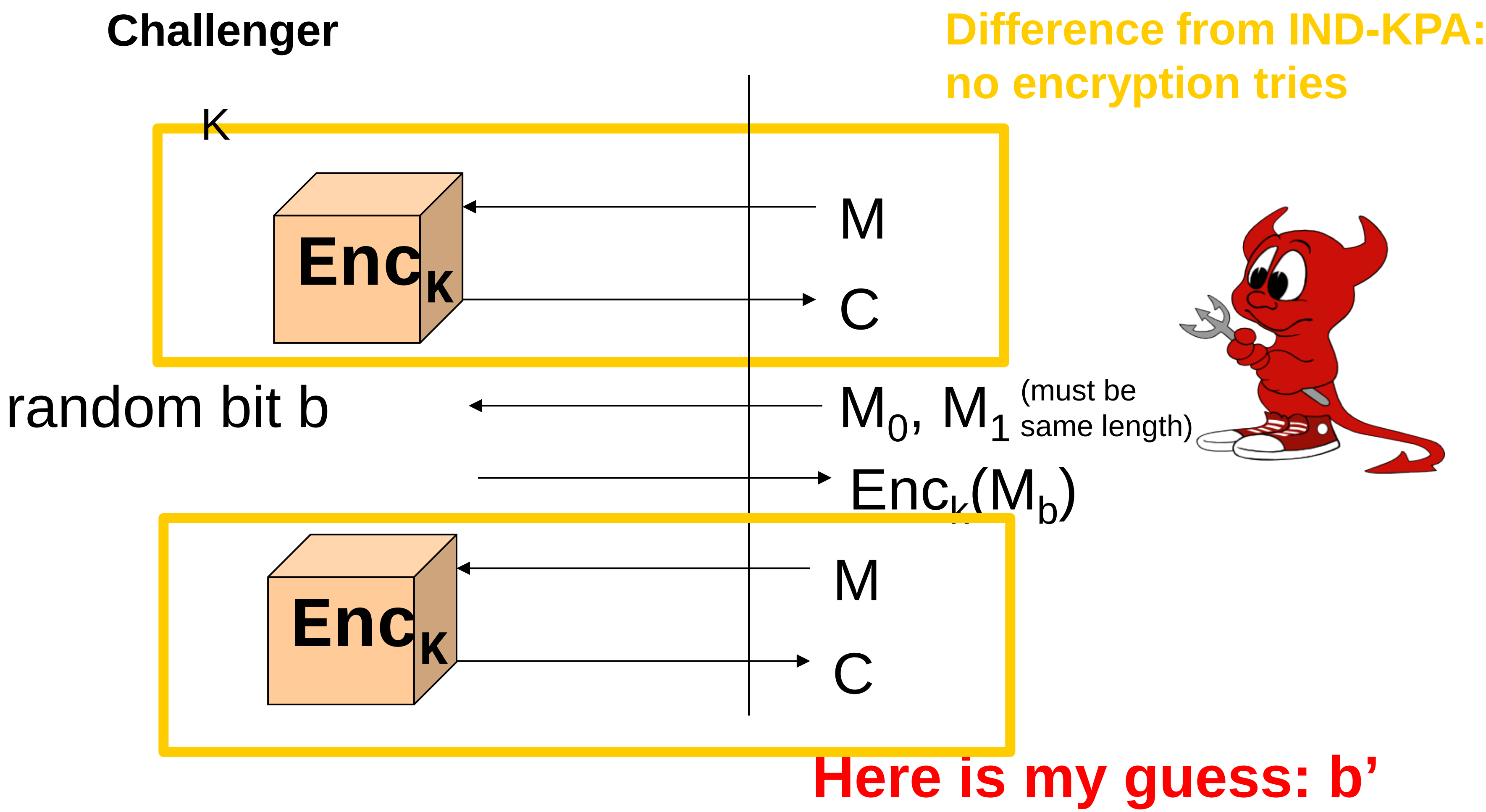
Use block ciphers to construct symmetric-key encryption

- Want two properties:
  - IND-CPA security even when reusing the same key to encrypt many messages
  - Can encrypt messages of any length

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)



IND-CPA

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

Pr[Adv wins game] <= ½ + 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)

Difference from known-plaintext attack from last time

- The extra queries to Enc_K
- Q: Why is IND-CPA a stronger security?
  - A: The attacker is given more capabilities so the IND-CPA scheme resists a more powerful attacker

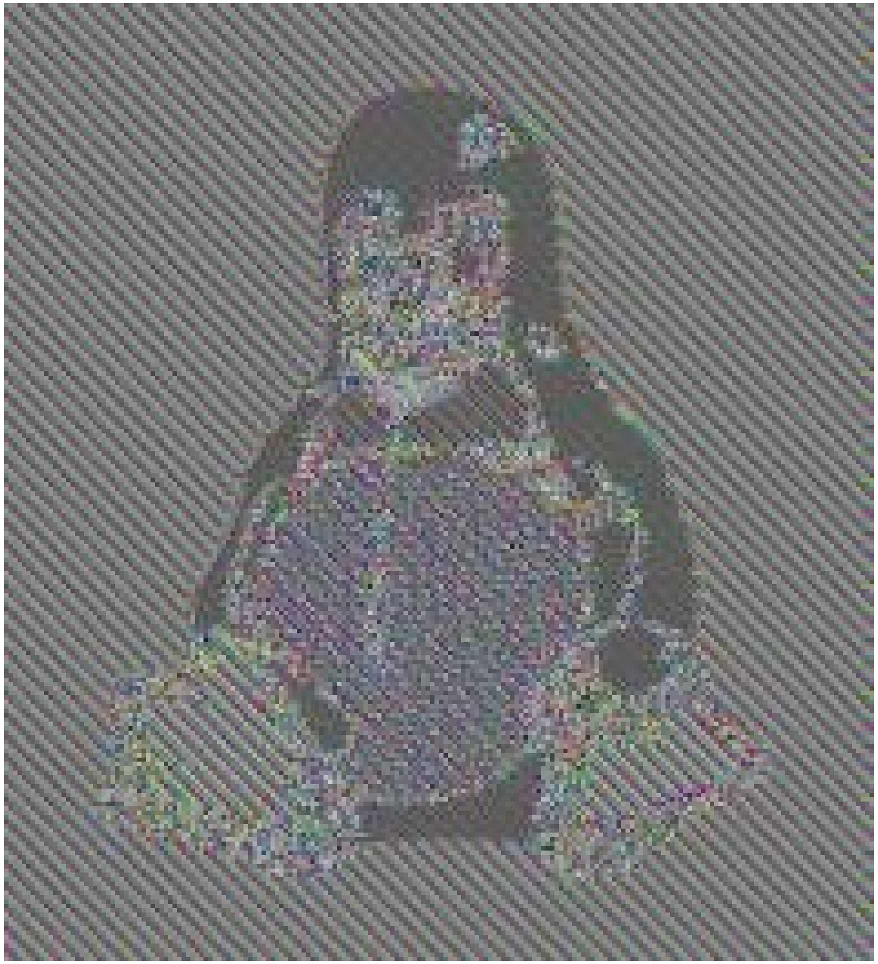
Are block ciphers IND-CPA?

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

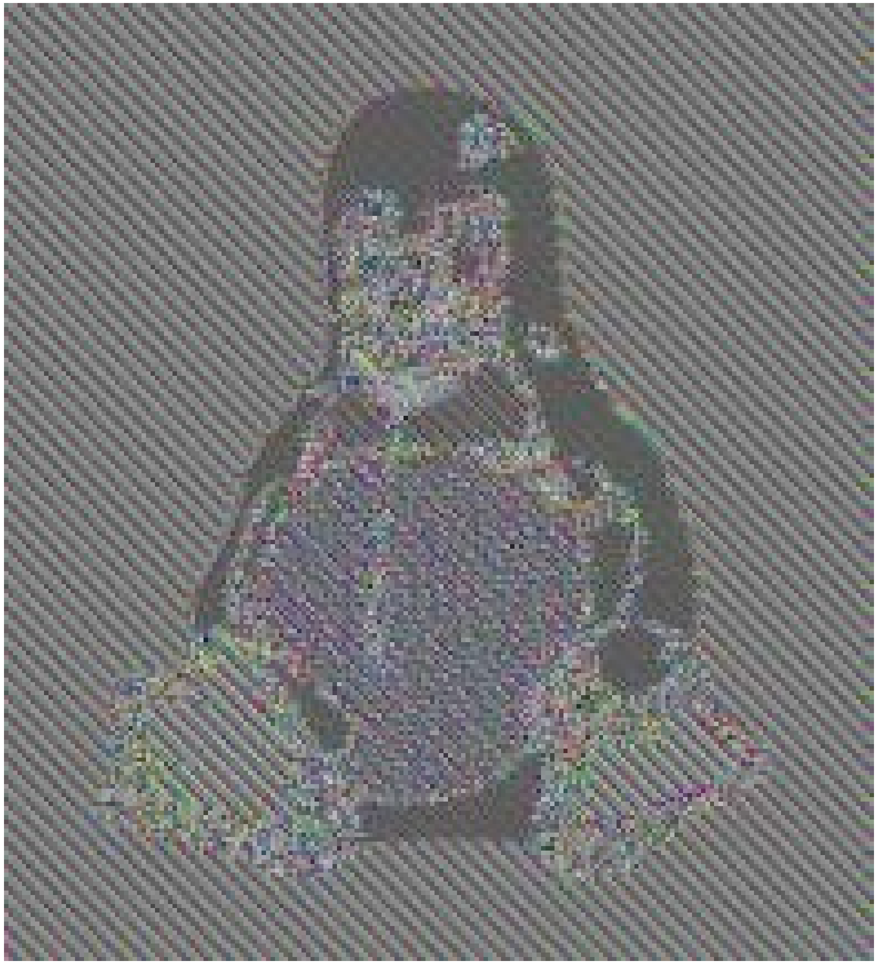
Are block ciphers IND-CPA?

- 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 = bread, M_1 = 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



Eack block encrypted with a block cipher



Later (identical) message again encrypted

Modes of operation

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

- Certain output from one block feeds into next block

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

Why? To prevent the encryption scheme from being deterministic

Counter mode (CTR)

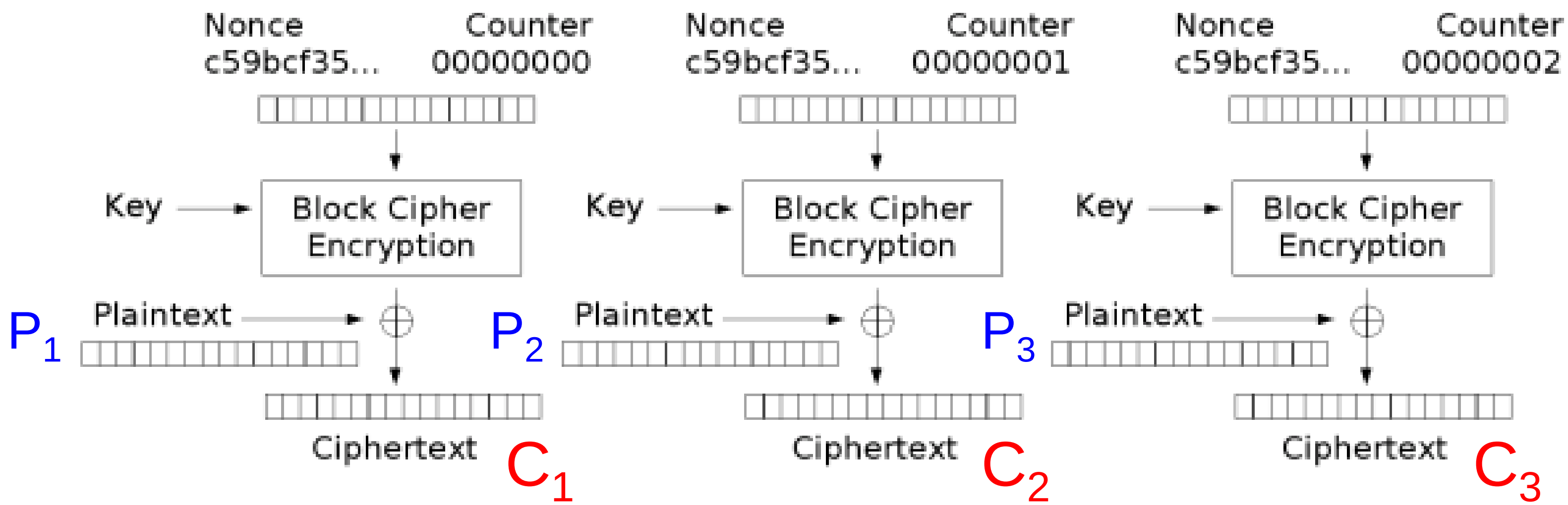
Last time: ECB, CBC

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,..
- Choose a random nonce (Nonce = Same as IV)
- Now compute:

Important that nonce does not repeat across different encryptions (choose it at random from large space)



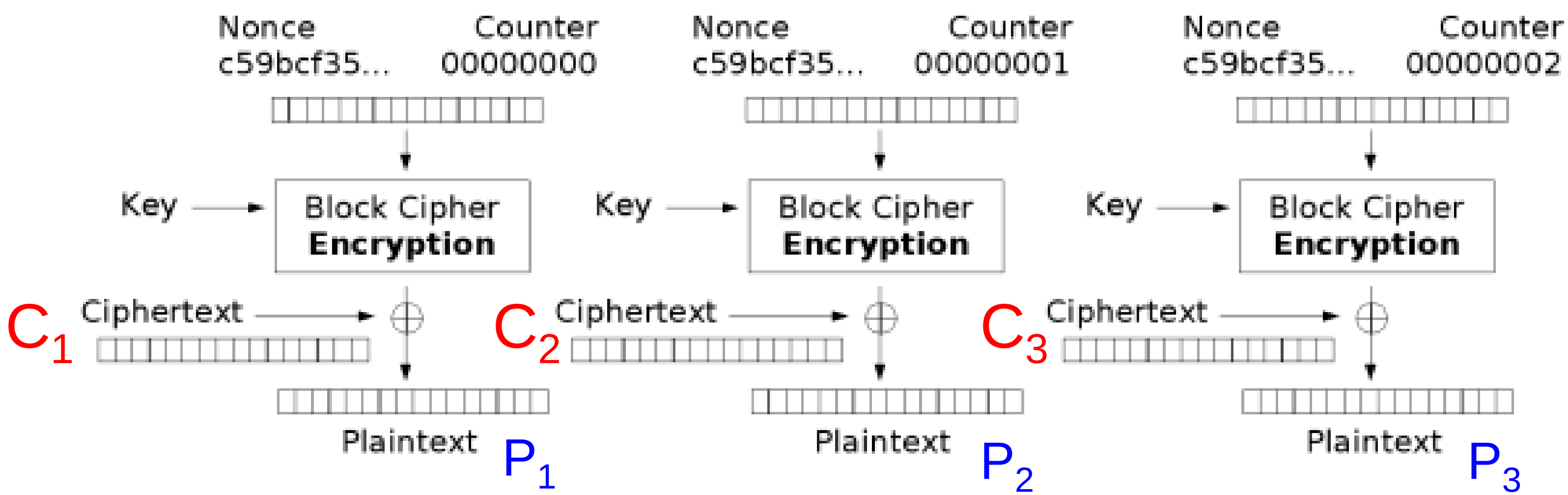
Counter (CTR) mode encryption

- 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:



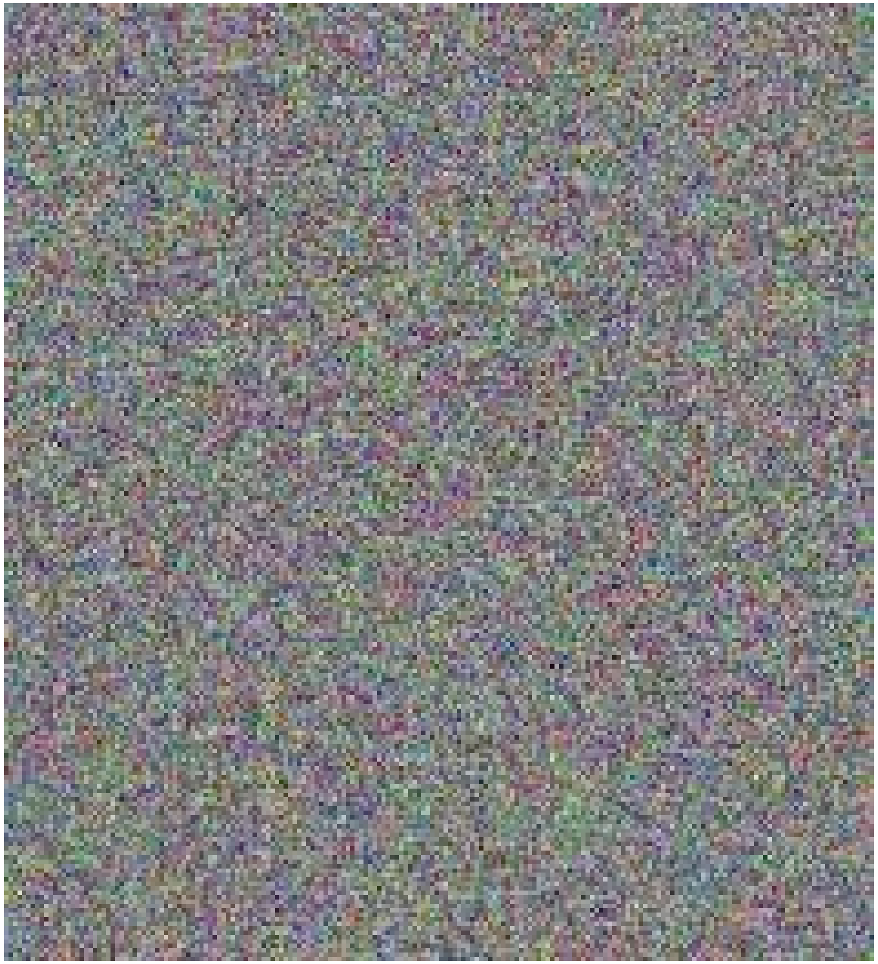
Counter (CTR) mode decryption

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

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

Want to see CTR explained slowly on “whiteboard”?

Original image



Encrypted with CBC

CBC vs CTR

Security: If no reuse of nonce**, 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)

Pseudorandom generator (PRG)

Pseudorandom Generator (PRG)

- Given a seed, it outputs a sequence of random bits

  PRG(seed) -> random bits

- It can output arbitrarily many random bits

PRG security

- Can PRG(K) be truly random?

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

- 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:

$PRG(K, IV) = E_k(IV|1), E_k(IV| 2), E_k(IV|3) \ldots E_k(IV| ceil(m/n))$, where | 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 = PRG(K, IV) XOR M
- Output (IV, C)

Q: How decrypt?

A: Compute PRG(K, IV) and XOR with ciphertext C

Q: What is advantage of OTP?

A: Can encrypt any message length because PRG can produce any number of random bits

Block ciphers 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

Start asymmetric cryptography on board