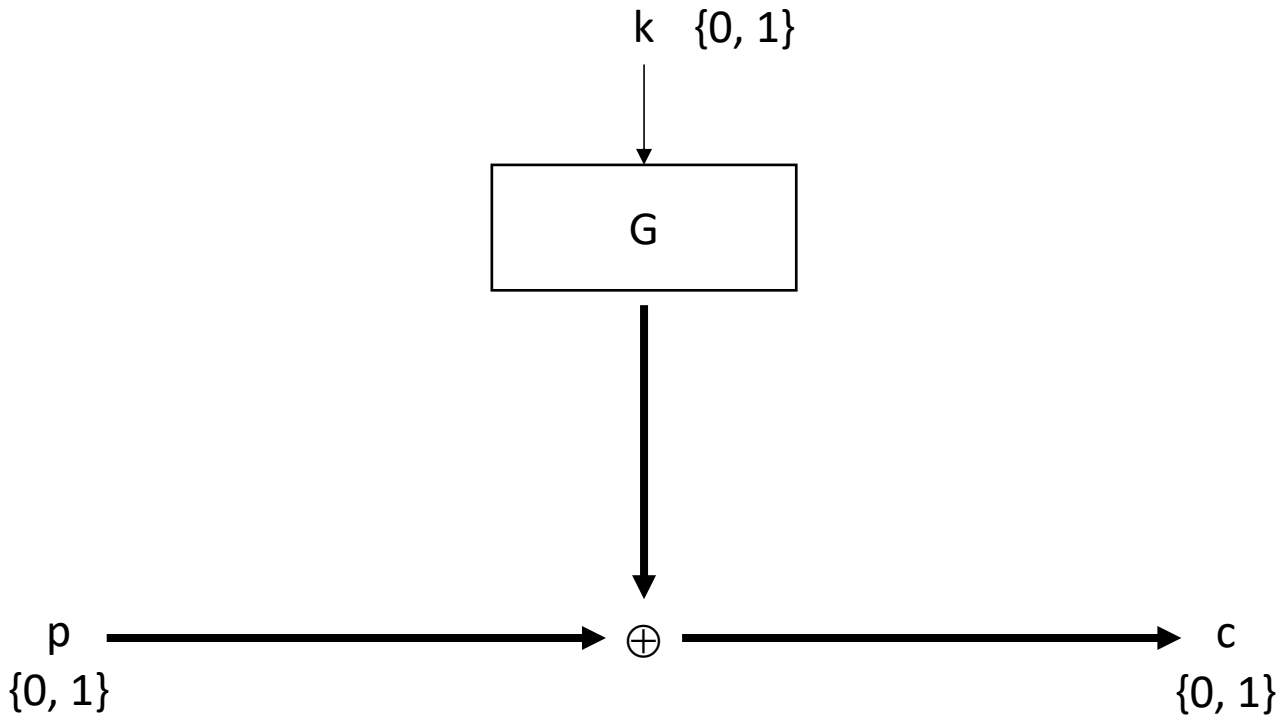
Security in Informatics and in the Organizations (2018/2019)

Practical Class (#3), #4 (, #5):

Applied Cryptography

Vitor Cunha (vitorcunha@ua.pt)

Symmetric Cryptography



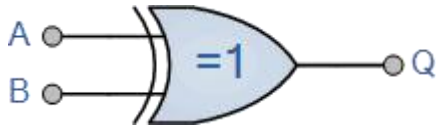
G	p	c
0	0	0
0	1	1
1	0	1
1	1	0

If G is not truly random:

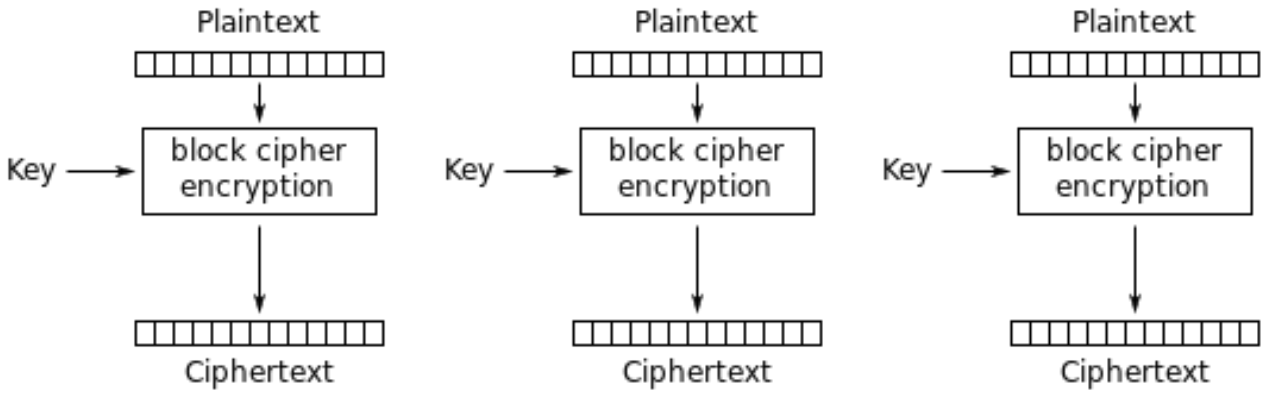
$P_c(0) = ½ + \varepsilon$

$P_c(1) = ½ + \varepsilon$

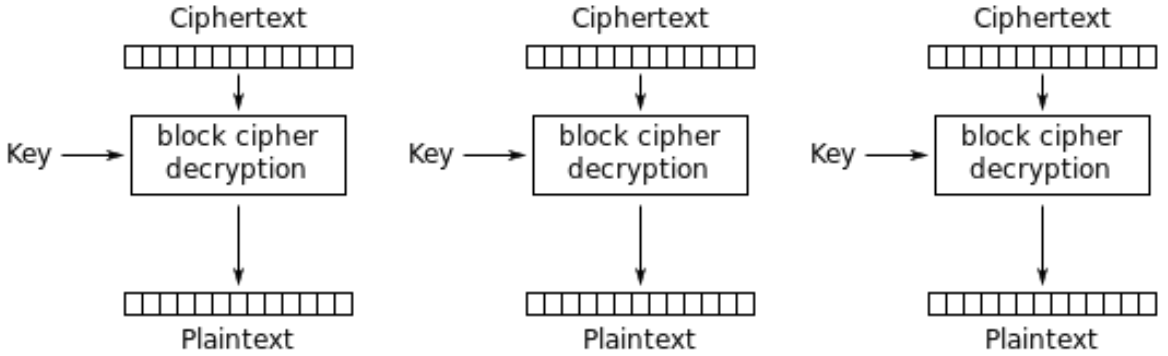
Secure if $\varepsilon < E$



Symmetric Cryptography (cipher modes)

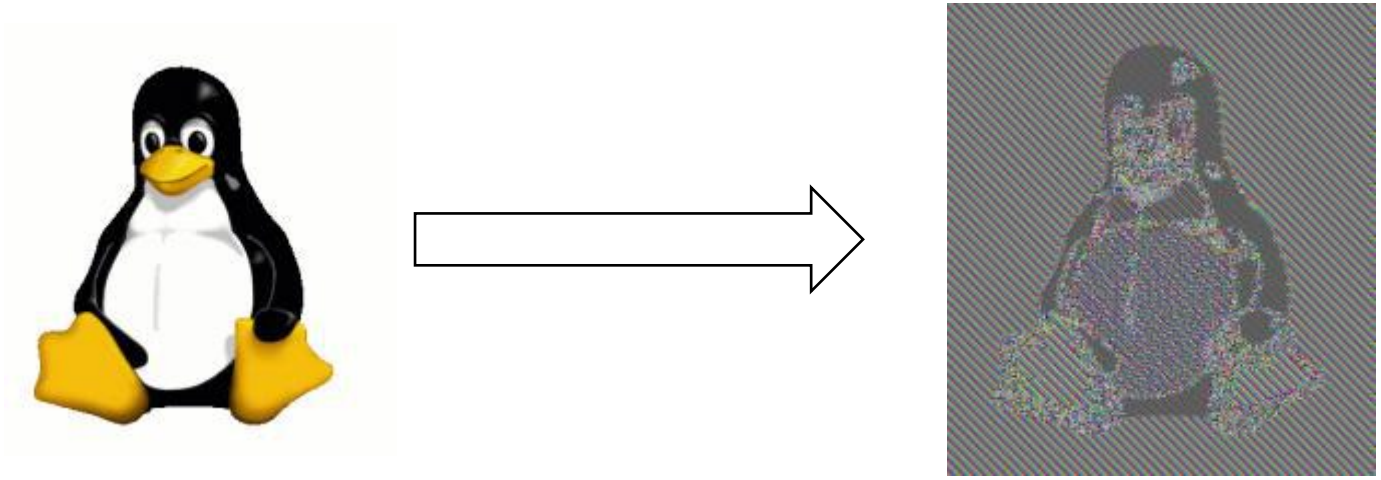


Electronic Codebook (ECB) mode encryption

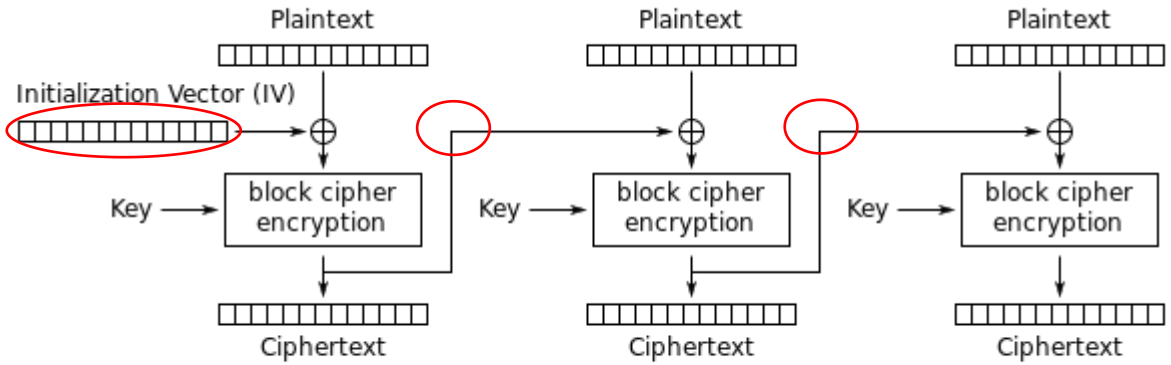


Electronic Codebook (ECB) mode decryption

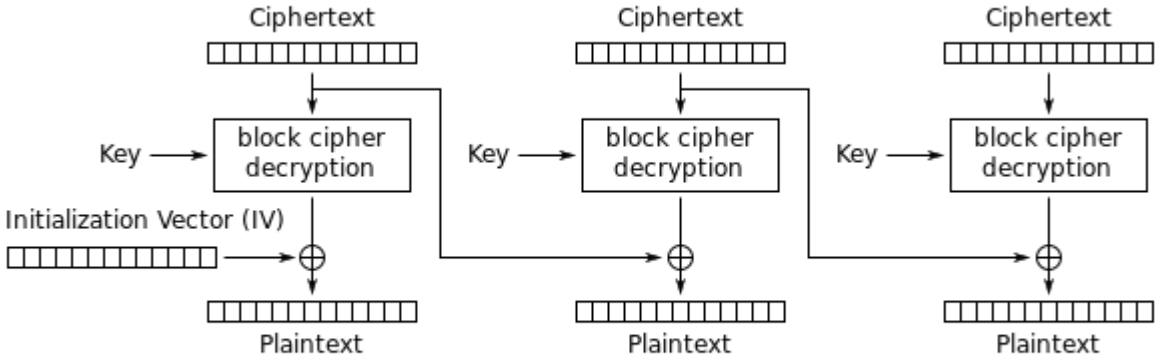
Symmetric Cryptography (ECB)



Symmetric Cryptography (CBC)

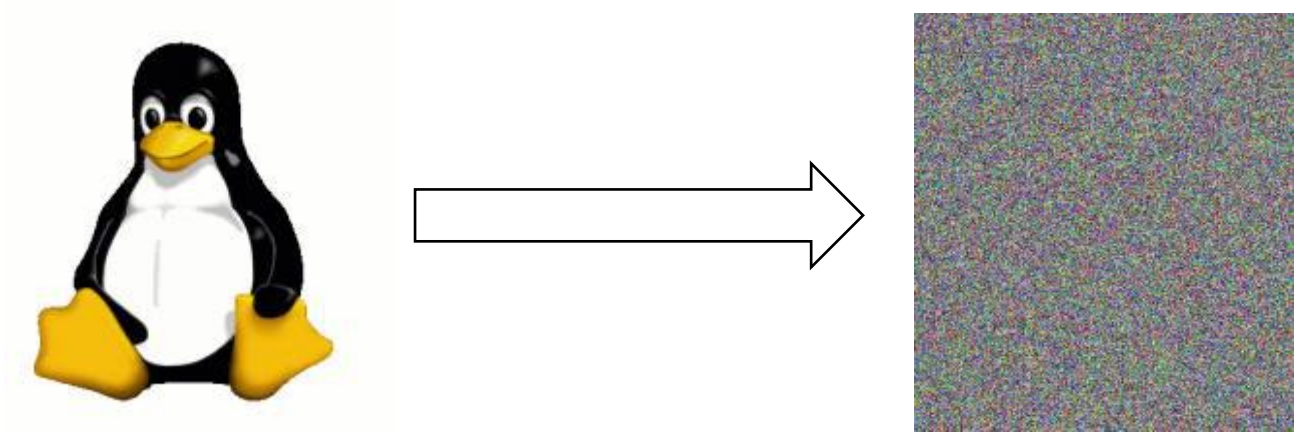


Cipher Block Chaining (CBC) mode encryption



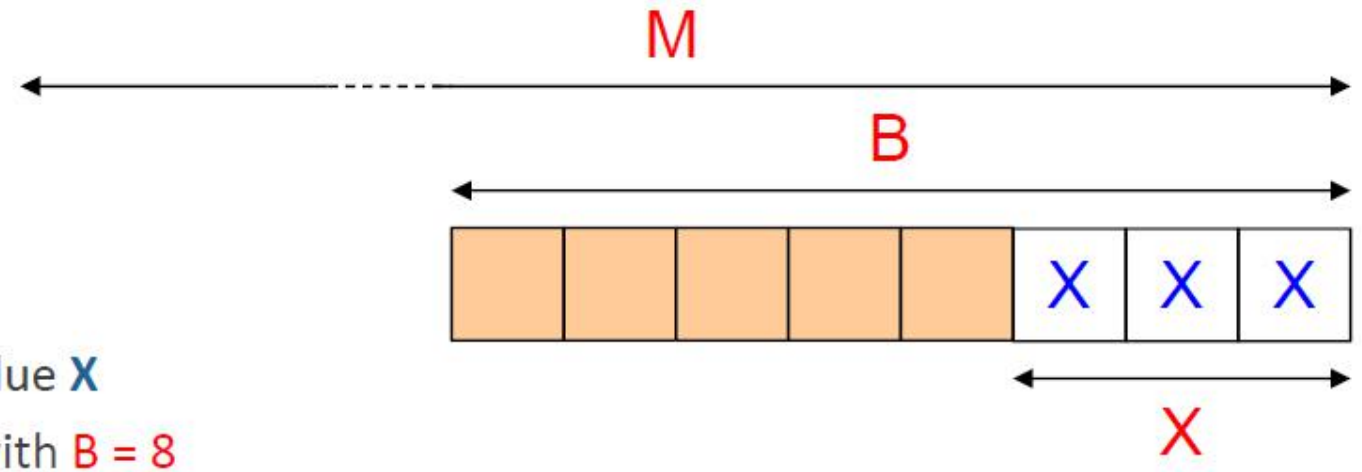
Cipher Block Chaining (CBC) mode decryption

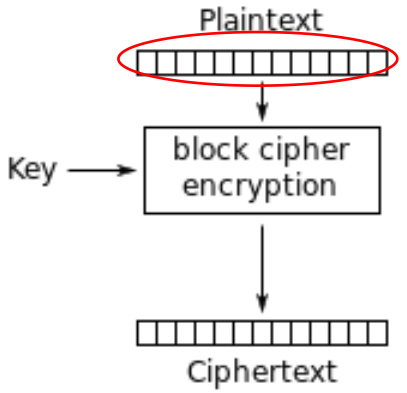
Symmetric Cryptography (CBC)



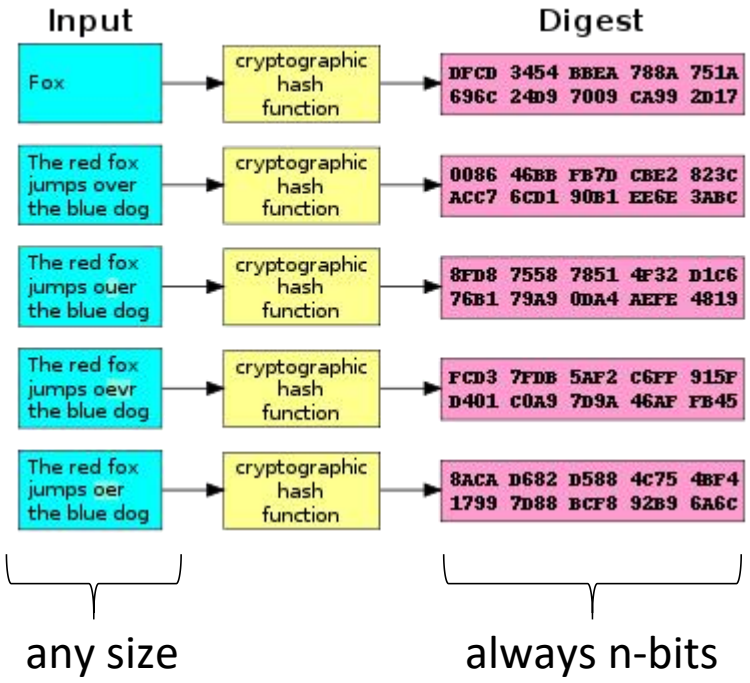
Symmetric Cryptography (Padding)

- Padding
  - Of last block, identifiable
  - PKCS #7
    - **X = B – (M mod B)**
    - **X** extra bytes, with the value **X**
  - PKCS #5: Equal to PKCS #7 with B = 8
- Different processing for the last block
  - Adds complexity





Digest Function (aka “Hash”)



1 - Pre-image resistance

Given a hash value h it should be difficult to find any message m such that h = hash(m). This concept is related to that of a one-way function. Functions that lack this property are vulnerable to preimage attacks.

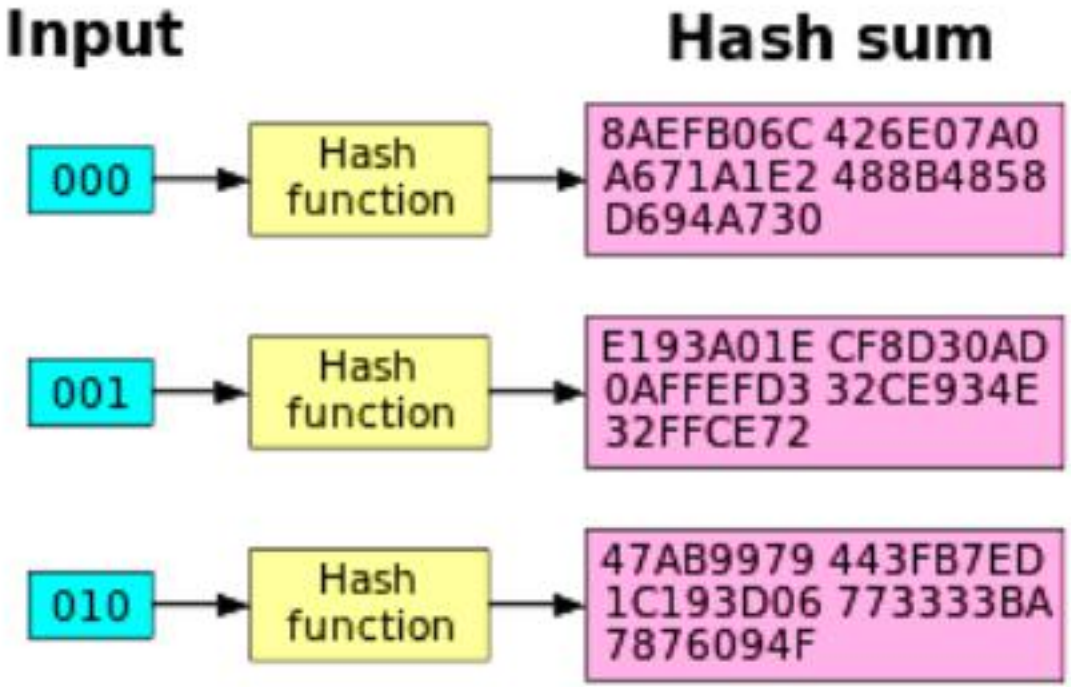
2 - Second pre-image resistance

Given an input m1, it should be difficult to find a different input m2 such that hash(m1) = hash(m2). Functions that lack this property are vulnerable to second-preimage attacks.

3 - Collision resistance

It should be difficult to find two different messages m1 and m2 such that hash(m1) = hash(m2). Such a pair is called a cryptographic hash collision. This property is sometimes referred to as strong collision resistance. It requires a hash value at least twice as long as that required for pre-image resistance; otherwise collisions may be found by a birthday attack.

Avalanche Effect



Asymmetric ciphers *(aka Public-Key Cryptography)*

Symmetric cryptography uses the same key both for encryption and decryption

- Key distribution may be an issue (N x N-1) $\cong$ N^2

- Any of the N parties can generate the ciphertext
  - Authenticity is 1 of N
  - No non-repudiation!

- Unsuitable for most massive communications
  - eg. all citizens sending their IRS form to the Finances Ministry

*We need separate keys, each party as one keypair -- a **Public** key and a matching **Private** key*

Asymmetric ciphers *(RSA n-bits)*

"Plain" RSA encryption

N, e

c

$(N, e, d) \leftarrow RSAGen(1^n)$

$pk = (N, e)$

$sk = d$

$m = [c^d \bmod N]$

$c = [m^e \bmod N]$

Source: *J. Katz, UMD – in Coursera*

Practical Use

- Confidentiality (eg. File Encryption)
  - Encrypt data with ***public key***
  - Decrypt with ***private key***
  - Source is not authenticated!

- Authenticity (eg. Digital Signatures, Authentication)
  - Encrypt challenge/identifier with ***private key***
  - Decrypt with ***public key***
  - Source is now authenticated!

Caveat

- Asymmetric encryption is very computationally intensive.

  Common strategies to work-around this:

  - Use symmetric encryption for bulk data encryption, then use the PKI to cipher the key

  - Use a digest to shorten the challenge / data that must be authenticated

Caveat Emptor

The previously described "Plain" RSA method is not secure!

Random padding is always added to make it secure.

Any Questions?