

# Block Ciphers

Gianluca Dini  
Dept. of Ingegneria dell'Informazione  
University of Pisa  
[gianluca.dini@unipi.it](mailto:gianluca.dini@unipi.it)  
Version: 2023-04-03



1

Block Ciphers

# GENERAL CONCEPTS


Apr-23

Block Ciphers

2

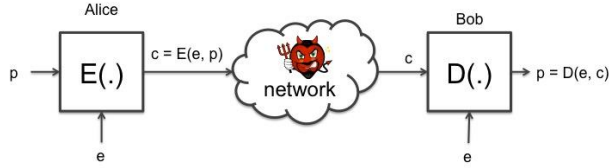
2

# Block cipher



UNIVERSITÀ DI PISA

- Block ciphers break up the plaintext in blocks of fixed length  $n$  bits and encrypt one block at time




- $E_k: \{0,1\}^n \rightarrow \{0,1\}^n$        $D_k: \{0,1\}^n \rightarrow \{0,1\}^n$
- $E$  is a keyed permutation:  $E(k, m) = E_k(m)$
- $E_k(\cdot)$  is a permutation

Apr-23      Block Ciphers      3

3

# Permutation




UNIVERSITÀ DI PISA

- $E_k$  is a permutation
  - $E_k$  is efficiently computable
  - $E_k$  is bijective
    - Surjective (or onto)
    - Injective (or one-to-one)
  - $E_k^{-1}$  is efficiently computable

Apr-23      Block Ciphers      4

4

## Examples



- Block ciphers
  - DES       $n = 64$  bits,       $k = 56$  bits
  - 3DES     $n = 64$  bits,       $k = 168$  bits
  - AES       $n = 128$  bits       $k = 128, 192, 256$  bits
- Performance (AMD Opteron, 2.2 GHz)
  - RC4              126 MB/s
  - Salsa20/12      643 MB/s
  - Sosemanuk      727 MB/s
  - 3DES             13 MB/s
  - AES-128        109 MB/s


Apr-23

Block Ciphers

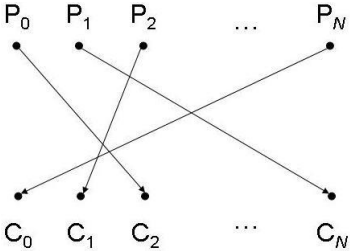
5

5

## Random permutations



$N = 2^n - 1$



- Let  $\text{Perm}_n$  be the set of all permutations  $\pi: \{0,1\}^n \rightarrow \{0,1\}^n$
- $|\text{Perm}_n| = 2^n!$
- A true random cipher
  - implements all the permutations in  $\text{Perm}_n$
  - uniformly selects a permutation  $\pi \in \text{Perm}_n$  at random

A possible random permutation  $\pi$

Apr-23

Block Ciphers

6

6

## True Random Cipher



UNIVERSITÀ DI PISA

- A True random cipher is perfect
- A true random cipher implements all possible Random permutations ( $2^n!$ )
  - Need a uniform random key for each permutation (naming)
    - key size  $:= \log_2 (2^n!) \approx (n - 1.44) 2^n$ 
      - Exponential in the block size!
      - The block size cannot be small in order to avoid a dictionary attack
- A true random cipher cannot be implemented

Apr-23

Block Ciphers

7

7

## Pseudorandom permutations



UNIVERSITÀ DI PISA

- Consider a *family of permutations* parametrized by  $\kappa$   
 $\in K = \{0, 1\}^k$ ,  $E_\kappa: \{0, 1\}^n \rightarrow \{0, 1\}^n$
- A  $E_\kappa$  is a *pseudorandom permutation* (PRP) if it is indistinguishable from a uniform random permutation by a limited adversary
- $|\{E_\kappa\}| = 2^k \ll |\text{Perm}_n|$ , with  $|\kappa| = k$
- A block cipher is a practical instantiation of a PRP

Apr-23


Block Ciphers

8

8

## Practical block cipher

- In practice, the encryption function corresponding to a randomly chosen key should appear as a randomly chosen permutation to a limited adversary




$x_i$

$\pi(x_i)$  or  $E(k, x_i)?$

$\pi \leftarrow \text{Perms}(\{0, 1\}^n)$

$\kappa \leftarrow K, E_k$

- Oracle access
  - adversary cannot look into the box



UNIVERSITÀ DI PISA

Apr-23


Block Ciphers

9

9

## Exhaustive key search

- The attack
  - Given a pair (pt, ct), check whether  $ct == E_{k_i}(pt)$ ,  $i = 0, 1, \dots, 2^k - 1$ 
    - Known-plaintext attack
    - Time complexity:  $O(2^k)$
  - False positives
    - Do you expect that just one key  $k$  maps  $pt$  into  $ct$ ?
    - How many keys (false positives) do we expect to map  $pt$  into  $ct$ ?
    - How do you discriminate the good one?



UNIVERSITÀ DI PISA


Apr-23

Block Ciphers

10

10

# Exhaustive key search



UNIVERSITÀ DI PISA

- False positives
  - Do you expect that just one key  $k$  maps  $pt$  into  $ct$ ?
  - How many keys (false positives) do we expect to map  $pt$  into  $ct$ ?
  - How do you discriminate the good one?


Apr-23

Block Ciphers

11

11

# False positives



UNIVERSITÀ DI PISA

- Problem: Given  $(ct, pt)$  s.t.  $ct = E_{k^*}(pt)$  for a given  $k^*$ , determine the number of keys that map  $pt$  into  $ct$
- Solution.
  - Given a certain key  $k$ ,  $P(k) = \Pr[E_k(pt) == ct] = 1/2^n$
  - The *expected* number of keys that map  $pt$  into  $ct$  is  $2^k \times 1/2^n = 2^{k-n}$


Apr-23

Block Ciphers

12

12

## False positives



UNIVERSITÀ DI PISA

- Example 1 – DES with  $n = 64$  and  $k = 56$ 
  - On average  $2^{-8}$  keys map pt into ct
  - One pair (pt, ct) is sufficient for an exhaustive key search
- Example 2 – Skipjack with  $n = 64$  and  $k = 80$ 
  - On average  $2^{16}$  keys map pt into ct
  - Two or more plaintext-ciphertext pairs are necessary for an exhaustive key search


Apr-23

Block Ciphers

13

13

## False positives



UNIVERSITÀ DI PISA

- Consider now  $t$  pairs  $(pt_i, ct_i)$ ,  $i = 1, 2, \dots, t$ 
  - Given  $k^*$ ,  $\Pr[E_{k^*}(pt_i) = ct_i, \text{ for all } i = 1, 2, \dots, t] = 1/2^{tn}$
  - Expected number of keys that map  $pt_i$  into  $ct_i$ , for all  $i = 1, 2, \dots, t$ , is  $2^k/2^{tn} = 2^{k-tn}$
- Example 3 – Skypjack with  $k = 80$ ,  $n = 64$ ,  $t = 2$ 
  - The expected number of keys is  $= 2^{80-2 \times 64} = 2^{-48}$
  - Two pairs are sufficient for an exhaustive key search


Apr-23

Block Ciphers

14

14

# False positives



UNIVERSITÀ DI PISA

- THEOREM
  - Given a block cipher with a key length of  $k$  bits and a block size of  $n$  bits, as well as  $t$  plaintext-ciphertext pairs,  $(pt_1, ct_1), \dots, (pt_t, ct_t)$ , the expected number of false keys which encrypt all plaintexts to the corresponding ciphertexts is  $2^k - t^n$
- FACT
  - Two input-output pairs are generally enough for exhaustive key search

Apr-23

Block Ciphers

15

15

Block ciphers

# EXERCISES

Apr-23

Block Ciphers


16

16



## Exercise 1 - Exhaustive key search

- Exhaustive key search is a known-plaintext attack
- However, the adversary can mount a ciphertext-only attack if (s)he has some knowledge on PT



UNIVERSITÀ DI PISA

Apr-23


Block Ciphers

17

17

## Exercise 1 – exhaustive key search

- Assume DES is used to encrypt 64-bit blocks of 8 ASCII chars, with one bit per char serving as parity bit
- How many CT blocks the adversary needs to remove false positives with a probability smaller than  $\epsilon$ ?



UNIVERSITÀ DI PISA


Apr-23

Block Ciphers

18

18

## Exercise 2 - dictionary attack



UNIVERSITÀ DI PISA

- Consider  $E$  with  $k$  and  $n$ .
- The adversary has collected  $D$  pairs  $(pt_i, ct_i)$ ,  $i = 1, \dots, D$ , with  $D \ll 2^n$
- Now the adversary reads  $C$  newly produced cyphertexts  $ct^*_j$ ,  $j = 1, \dots, C$ .
- Determine the value of  $C$  s.t. the  $\Pr[\text{Exists } j, j = 1, 2, \dots, C, \text{ s.t. } c^*_j \text{ is in the dictionary}] = P$


Apr-23

Block Ciphers

19

19

## Exercise 3 - Rekeying



UNIVERSITÀ DI PISA

- An adversary can successfully perform an exhaustive key search in a month.
- Our security policy requires that keys are changed every hour.
- What is the probability  $P$  that, in a month, the adversary is able to find any key before it is changed?
  - For simplicity assume that every month is composed of 30 days.
- What if we refresh key every minute?

Apr-23

Block Ciphers

20

20

Symmetric Encryption

MULTIPLE ENCRYPTION AND KEY WHITENING


Apr-23

Block Ciphers

21

21

Increasing the Security of Block Ciphers

  
UNIVERSITÀ DI PISA

- DES is a secure cipher
  - No efficient cryptanalysis is known
- DES does not define a group
- DES key has become too short
- Can we improve the security of DES?
- Yes, by means of two techniques
  - Multiple encryption
  - Key whitening


Apr-23

Block Ciphers

22

22

## DES does not define a group




UNIVERSITÀ DI PISA

- If DES were a group then  $\forall k_1, k_2 \in \mathcal{K}, \exists k_3 \in \mathcal{K}$  s.t.  
 $\forall x \in \mathcal{M}, E_{k_2}(E_{k_1}(x)) = E_{k_3}(x)$
- So, double encryption would be useless
- Furthermore, DES would be vulnerable to Meet-in-the-Middle attack that runs in  $2^{28}$

Apr-23      Block Ciphers      23

23

## Two-times Encryption (2E)



UNIVERSITÀ DI PISA

- $y = 2E((e_L, e_R), m) = E(e_R, E(e_L, x))$ 
  - key size is  $2k$  bits
  - Brute force attack requires  $2^{2k}$  steps
  - 2E is two times slower than E
- Is it really secure?
- Meet-in-the-middle attack

$x \longrightarrow$

$E(e_L, \cdot)$

$\longrightarrow$


$E(e_R, \cdot)$

$\longrightarrow y$

Apr-23      Block Ciphers      24

24

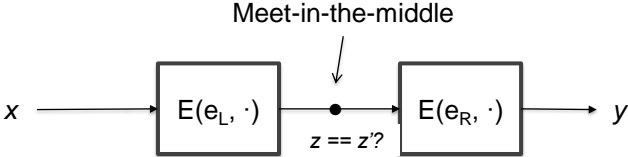
# Meet-in-the-middle attack



UNIVERSITÀ DI PISA

- Attack Sketch
  - Build a table  $T$  containing  $z = E(e_L, x)$  for all possible keys  $e_L$ . Keep  $T$  sorted according to  $z$ .
  - Check whether  $z' = D(e_R, y)$  is contained in the table  $T$ , for all possible key  $e_R$ .
    - If  $z'$  is contained in  $T$  then  $(e_L, e_R)$  maps  $x$  into  $y$  with  $e_L$  s.t.  $T[e_L] = z'$ .

Meet-in-the-middle



$x \rightarrow E(e_L, \cdot) \rightarrow \bullet \rightarrow E(e_R, \cdot) \rightarrow y$   
 $z == z'?$


Apr-23

Block Ciphers

25

25

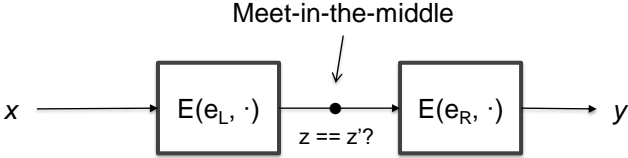
# Meet-in-the-middle attack



UNIVERSITÀ DI PISA

- Attack complexity
  - Storage complexity
    - Storage necessary for table  $T \approx O(2^k)$
  - Time complexity
    - Time complexity for step 1 + Time complexity for step 2 = Time for building and sorting the table + Time for searching in a sorted table =  $k \cdot 2^k + k \cdot 2^k \approx O(2^k)$

Meet-in-the-middle



$x \rightarrow E(e_L, \cdot) \rightarrow \bullet \rightarrow E(e_R, \cdot) \rightarrow y$   
 $z == z'?$


Apr-23

Block Ciphers

26

26

# Two-times DES



UNIVERSITÀ DI PISA

- 2DES
  - Time complexity:  $2^{56}$  (doable nowadays!)
  - Space complexity:  $2^{56}$  (lot of space!)
  - 2DES brings no advantage


Apr-23

Block Ciphers

27

27

# Triple DES (3DES)



UNIVERSITÀ DI PISA

- EDE scheme
  - Standard ANSI X9.17 and ISO 8732
  - $Y = 3E((e_1, e_2, e_3), x) = E(e_1, D(e_2, E(e_3, x)))$ 
    - If  $e_1 = e_2 = e_3$ , 3DES becomes DES
      - backward compatibility
  - Key size = 168-bits
  - 3 times slower than DES
  - Simple attack  $\approx 2^{118}$


Apr-23

Block Ciphers

28

28

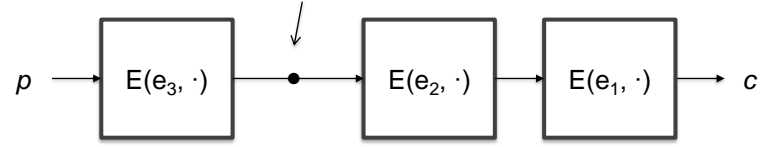
### 3DES – meet-in-the-middle attack



UNIVERSITÀ DI PISA

- Time =  $2^{112}$  (undoable!)
- Space =  $2^{56}$  (lot of space!)


*Meet-in-the-middle*



Apr-23 Block Ciphers 29

29

### False positives for multiple encryption



UNIVERSITÀ DI PISA


- THEOREM
  - Given there are  $r$  subsequent encryptions with a block cipher with a key length of  $k$  bits and a block size of  $n$  bits, as well as  $t$  plaintext-ciphertext pairs,  $(pt_1, ct_1), \dots, (pt_t, ct_t)$ , the expected number of false keys which encrypt all plaintext to the corresponding ciphertext is  $2^{rk - tn}$

Apr-23 Block Ciphers 30

30

## Limitations of 3DES

- 3DES resists brute force but
  - It is not efficient regarding software implementation
  - It has a short block size (64 bit)
    - A drawback if you want to make a hash function from 3DES, for example
  - Key lengths of 256+ are necessary to resist quantum computing attack



UNIVERSITÀ DI PISA

Apr-23


Block Ciphers

31

31

## Key whitening

- Considerations
  - KW is not a “cure” for weak ciphers
- Applications
  - DESX: a variant of DES
  - AES: uses KW internally
- Performance
  - Negligible overhead w.r.t. E (Just two XOR’s!)



UNIVERSITÀ DI PISA

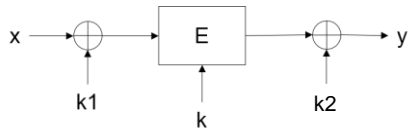


Diagram illustrating Key Whitening for block ciphers:

Input  $x$  is XORed with key  $k_1$ , then the result is encrypted by block cipher  $E$  using key  $k$ , and finally the output is XORed with key  $k_2$  to produce  $y$ .

Definition 5.3.1 Key whitening for block ciphers

Encryption:  $y = e_{k,k_1,k_2}(x) = e_k(x \oplus k_1) \oplus k_2$

Decryption:  $x = e_{k,k_1,k_2}^{-1}(y) = e_k^{-1}(y \oplus k_2) \oplus k_1$

Apr-23


Block Ciphers

32

32



# Key whitening



UNIVERSITÀ DI PISA

- Attacks
  - Brute-force attack
    - Time complexity:  $2^{k+2n}$  encryption ops
  - Meet-in-the-middle:
    - Time complexity  $2^{k+n}$
    - Storage complexity:  $2^n$  data sets
  - The most efficient attack
    - If the adversary can collect  $2^m$  pt-ct pairs, then time complexity becomes  $2^{k+n-m}$ 
      - The adversary cannot control  $m$  (rekeying)
    - Example: DES ( $m = 32$ )
      - Time complexity  $2^{88}$  encryptions (nowadays, out of reach)
      - Storage complexity  $2^{32}$  pairs = 64 GBytes of data (!!!)

Apr-23

Block Ciphers

33

33

# Symmetric Encryption

## ENCRYPTION MODES


Apr-23

Block Ciphers

34

34

# Encryption Modes



UNIVERSITÀ DI PISA

- A block cipher encrypts PT in fixed-size  $n$ -bit blocks
- When the PT len exceeds  $n$  bits, there are several modes to use the block cipher
  - Electronic Codebook (ECB)
  - Cipher-block Chaining (CBC)


Apr-23

Block Ciphers

35

35

# Other encryption modes



UNIVERSITÀ DI PISA

- Other encryption modes
  - To build a stream cipher out of a block cipher
    - Cipher Feedback mode (CFB)
    - Output Feedback mode (OFB)
    - Counter mode (CTR)
  - Authenticated encryption
    - Galois Counter mode (GCM, CCM, ...)
  - and many others (e.g., CTS, ...)
- Block ciphers are very versatile components

Apr-23

Block Ciphers

36

36

# Electronic codebook

*plaintext*

*ciphertext*

**PT blocks are encrypted separately**

$\forall 1 \leq i \leq t, c_i \leftarrow E(e, p_i)$

$\forall 1 \leq i \leq t, p_i \leftarrow D(e, c_i)$

Apr-23

Block Ciphers

37

37

# ECB - properties

- **PROS**
  - No block synchronization is required
  - No error propagation
    - One or more bits in a single CT block affects decryption of that block only
  - Can be parallelized
- **CONS (it is insecure)**
  - Identical PT results in identical CT
    - ECB doesn't hide data pattern
    - ECB allows traffic analysis
  - Blocks are encrypted separately
    - ECB allows block re-ordering and substitution


Apr-23

Block Ciphers

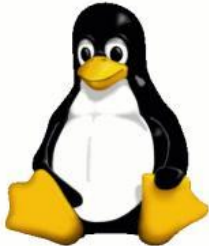
38

38

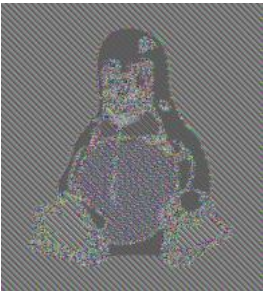
# ECB doesn't hide data patterns



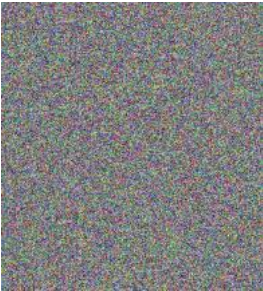
UNIVERSITÀ DI PISA



Plaintext



ECB encrypted




Non-ECB encrypted

Apr-23

Block Ciphers

39

# ECB – block attack



UNIVERSITÀ DI PISA


- Bank transaction that transfers a customer C's amount of money D from bank B1 to bank B2
  - Bank B1 debits D to C
  - Bank B1 sends the "credit D to C" message to bank B2
  - Upon receiving the message, Bank B2 credits D to C
- Credit message format
  - Src bank: M (12 byte)
  - Rcv bank: R (12 byte)
  - Customer: C (48 byte)
  - Bank account number: N (16 byte)
  - Amount of money: D (8 byte)
- Cipher:  $n = 64$  bit; ECB mode

Apr-23

Block Ciphers

40

## ECB – block attack




UNIVERSITÀ DI PISA

- Mr. Lou Cipher is a client of the banks and wants to make a fraud
- Attack aim
  - To replay Bank B1’s message "credit 100\$ to Lou Cipher" many times
- Attack strategy
  - Lou Cipher activates multiple transfers of 100\$ so that multiple messages "credit 100\$ to Lou Cipher" are sent from B1 to B2
  - The adversary identifies at least one of these messages
  - The adversary replies the message several times

Apr-23      Block Ciphers      41

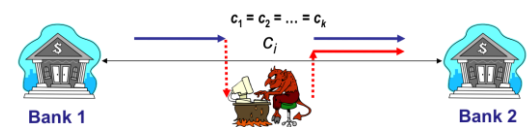
41

## ECB – block attack



UNIVERSITÀ DI PISA


- The fraud
  1. Mr. Lou Cipher performs  $k$  equal transfers
    - credit 100\$ to Lou Cipher  $\rightarrow c_1$
    - credit 100\$ to Lou Cipher  $\rightarrow c_2$
    - ...
    - credit 100\$ to Lou Cipher  $\rightarrow c_k$
  2. Then, he searches for “his own” CTs, namely  $k$  equal CTs!
  3. Finally he replies one of these cryptograms (many times)



Apr-23      Block Ciphers      42

42

## ECB – block attack



UNIVERSITÀ DI PISA

- The message lacks any notion of time so it can be easily replied
- An 8-byte timestamp field T (block #1) is added to the message to prevent replay attacks
- A replied message can now be discarded

block no.	1	2	3	4	5	6	7	8	9	10	11	12	13
	T	M	R	C						N		D	


Apr-23

Block Ciphers

43

43

## ECB – block attack



UNIVERSITÀ DI PISA

- However, Mr Lou Cipher can still perform the attack
  1. Identify “his own” CTs by inspecting blocks #2-#13
  2. Select any his-own-CT
  3. Substitute block #1 of his-own-CT with block #1 of any intercepted “fresh” block
  4. Replay the resulting CT


Apr-23

Block Ciphers

44

44

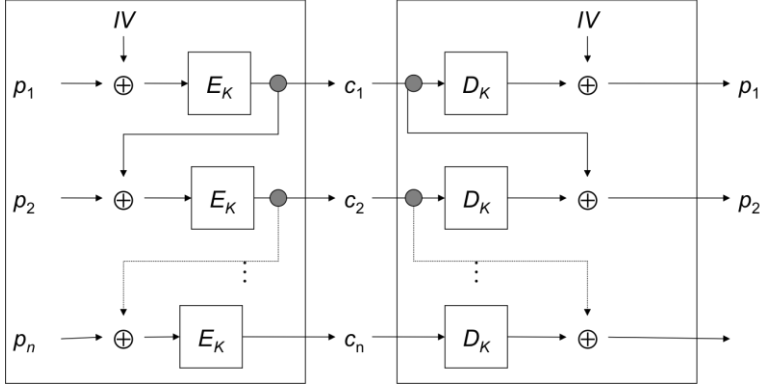
# Cipher block chaining (CBC)



UNIVERSITÀ DI PISA

Encryption:  $c_0 \leftarrow IV. \forall 1 \leq i \leq t, c_i \leftarrow E_k(p_i \oplus c_{i-1})$

Decryption:  $c_0 \leftarrow IV. \forall 1 \leq i \leq t, p_i \leftarrow c_{i-1} \oplus D_k(c_i)$




Apr-23

Block Ciphers

45

45

# CBC – properties (→)



UNIVERSITÀ DI PISA

- CBC mode is CPA-secure
- Chaining dependencies:  $c_i$  depends on  $p_i$  and the preceding PT blocks
- Cyphertext expansion is just one block
- CBC-Enc is *randomized* by using IV (nonce)
  - Identical ciphertext results from the same PT under the same key and IV
- CT-block reordering affects decryption

Apr-23

Block Ciphers

46

46

Foundations of Cybersecurity

23

## CBC – properties



UNIVERSITÀ DI PISA

- IV can be sent in the clear but its integrity must be guaranteed
- CBC suffers from Error propagation
  - Bit errors in  $c_i$  affect  $p_i$  and  $p_{i+1}$  (*error propagation*)
  - CBC is self-synchronizing (*error recovery*)
  - CBC does not tolerate “lost” bits (*framing errors*)
- CBC-dec can be parallelized

Apr-23

Block Ciphers

47

47

## CBC – block attack



UNIVERSITÀ DI PISA

- If Bank A chooses a random IV for each wire transfer the attack will not work
- However, if Lou Cipher substitutes blocks #5–10 and #13, bank B would decrypt *account number* and *deposit amount* to random numbers => this is highly undesirable!
- Encryption itself is not sufficient, we need additional mechanisms (MDC, MAC, digsig) to protect integrity

Apr-23


Block Ciphers

48

48



## Chosen-Plaintext Attack (Informal)



UNIVERSITÀ DI PISA

- CPA Attack
  - Attacker *makes* the sender to encrypt  $x_1, \dots, x_t$ 
    - The attacker may influence or control encryption
  - The sender encrypts and transmits  $y_1 = E_k(m_1), \dots, y_t = E_k(m_t)$
  - Later on, the sender encrypts  $x$  and transmits  $y = E_k(m)$
- CPA-security guarantees that the adversary cannot learn anything about  $x$
- The encryption scheme must be randomized


Apr-23

Block Ciphers

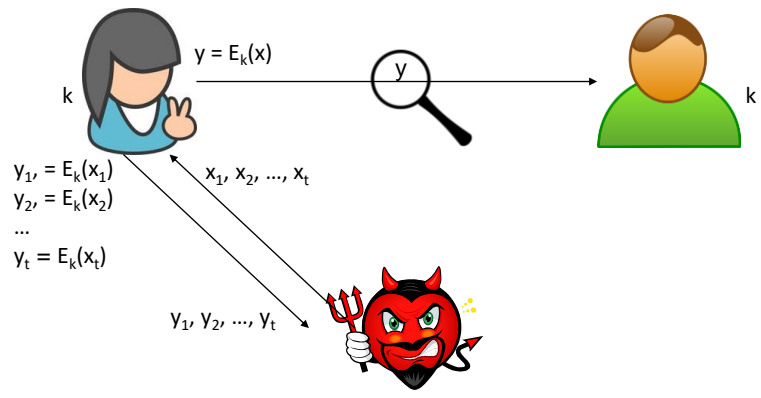
49

49

## CPA model



UNIVERSITÀ DI PISA



The diagram illustrates the Chosen-Plaintext Attack (CPA) model. It shows three main components: a sender (represented by a person icon), a receiver (represented by a person icon), and an attacker (represented by a devil icon). The sender has a key  $k$  and a message  $x$ . They compute the ciphertext  $y = E_k(x)$  and send it to the receiver. The receiver also has the key  $k$  and receives the ciphertext  $y$ . The attacker intercepts the ciphertext  $y$  (indicated by a magnifying glass). The attacker can also request encryptions of chosen plaintexts  $x_1, x_2, \dots, x_t$  from the sender. The sender provides the corresponding ciphertexts  $y_1 = E_k(x_1), y_2 = E_k(x_2), \dots, y_t = E_k(x_t)$  to the attacker. The attacker then uses this information to learn something about the original message  $x$ .

Apr-23

Block Ciphers

50

50

Block Ciphers: How to transform a block cipher into a stream cipher

MORE ENCRYPTION MODES: OFB,  
CFB, CTR


Apr-23

Block Ciphers

51

51

Block cipher vs stream cipher

  
UNIVERSITÀ DI PISA

- Padding is not necessary
- Stream ciphers can operate in real-time

Apr-23

Block Ciphers

52

52

# Output Feedback Mode (OFB)

The diagram illustrates the OFB mode for both encryption and decryption. In encryption (left), an Initialization Vector (IV) is input to a block cipher function E along with a key k. The output s<sub>1</sub> is XORed with the first plaintext block x<sub>1</sub> to produce ciphertext y<sub>1</sub>. Subsequent blocks s<sub>i</sub> are generated by feeding the previous ciphertext y<sub>i-1</sub> into E. In decryption (right), the same IV and key k are used. The ciphertext blocks y<sub>i</sub> are XORed with the keystream blocks s<sub>i</sub> to recover the plaintext x<sub>i</sub>. The keystream blocks s<sub>i</sub> are generated by feeding the previous ciphertext y<sub>i-1</sub> into E.

Let  $e()$  be a block cipher of block size  $b$ ; let  $x_i$ ,  $y_i$  and  $s_i$  be bit strings of length  $b$ ; and  $IV$  be a nonce of length  $b$ .

**Encryption (first block):**  $s_1 = e_k(IV)$  and  $y_1 = s_1 \oplus x_1$

**Encryption (general block):**  $s_i = e_k(s_{i-1})$  and  $y_i = s_i \oplus x_i$ ,  $i \geq 2$

**Decryption (first block):**  $s_1 = e_k(IV)$  and  $x_1 = s_1 \oplus y_1$

**Decryption (general block):**  $s_i = e_k(s_{i-1})$  and  $x_i = s_i \oplus y_i$ ,  $i \geq 2$

Apr-23

Block Ciphers

53

53

# Output Feedback Mode (OFB)

- OFB builds a stream cipher out of a block cipher
- The key stream is generated block-wise
- OFB is a *synchronous* stream cipher, i.e., key stream is a function of K and IV, only
  - ➔ precomputation of key stream is possible
- The receiver does not use decryption
- If |last pt block| < block, keystream bits are discarded


Apr-23

Block Ciphers

54

54

# Output Feedback Mode (OFB)



- IV should be a nonce → OFB non-deterministic
- No error propagation
- OFB suffers from malleability

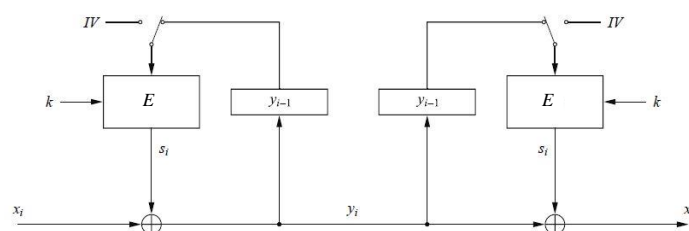

Apr-23

Block Ciphers

55

55

# Cipher Feedback Mode (CFB)



**Definition 5.1.4** Cipher feedback mode (CFB)

Let  $e()$  be a block cipher of block size  $b$ ; let  $x_i$  and  $y_i$  be bit strings of length  $b$ ; and  $IV$  be a nonce of length  $b$ .

**Encryption (first block):**  $y_1 = e_k(IV) \oplus x_1$

**Encryption (general block):**  $y_i = e_k(y_{i-1}) \oplus x_i, \quad i \geq 2$

**Decryption (first block):**  $x_1 = e_k(IV) \oplus y_1$

**Decryption (general block):**  $x_i = e_k(y_{i-1}) \oplus y_i, \quad i \geq 2$

Apr-23


Block Ciphers

56

56

## Cipher Feedback Mode (CFB)

- OFB builds a stream cipher out of a block cipher
- CFB is an *asynchronous* stream cipher as the key stream is also a function of the CT
- Key stream is generated block-wise
- IV is a nonce and makes CFB nondeterministic
- Enc is sequential, Dec may be parallelized
- CFB may operate on pt/ct smaller than a block
  - $\text{Sizeof}(\text{pt/ct}) = s \leq n$  (cipher block size)



UNIVERSITÀ DI PISA

Apr-23

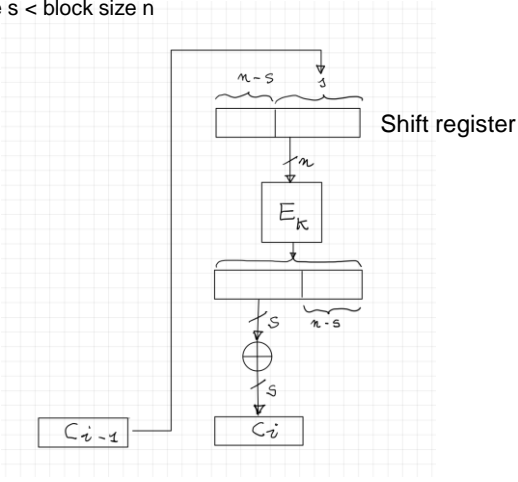
Block Ciphers

57

57

## Cipher Feedback Mode (CFB)

Pt/ct character size  $s <$  block size  $n$




Shift register

$E_k$

$C_{i-1}$

$C_i$



UNIVERSITÀ DI PISA

Apr-23

Block Ciphers

58

58

# Counter Mode (CTR)

**Definition 5.1.5** Counter mode (CTR)  
Let  $e()$  be a block cipher of block size  $b$ , and let  $x_i$  and  $y_i$  be bit strings of length  $b$ . The concatenation of the initialization value  $IV$  and the counter  $CTR_i$  is denoted by  $(IV||CTR_i)$  and is a bit string of length  $b$ .  
**Encryption:**  $y_i = e_k(IV||CTR_i) \oplus x_i, \quad i \geq 1$   
**Decryption:**  $x_i = e_k(IV||CTR_i) \oplus y_i, \quad i \geq 1$

Apr-23

Block Ciphers

59

59

# Counter Mode (CTR)

- CTR prevents two-time pad (keystream reuse)
- CTR can be parallelized
- Counter can be a regular counter or a more complex functions, e.g., LFSR
- Ciphertext expansion is just one block
  - Output  $y_0, y_1, \dots, y_t$  with  $y_0 = (IV||ctr_0)$  being the *expansion block*
  - $IV||ctr_0$  does not have to be kept secret
  - Can be transmitted together with ct  $y_i$


Apr-23

Block Ciphers

60

60

# CTR is CPA-secure



UNIVERSITÀ DI PISA

- A block cipher is a good approximation of a PRP (PRF), so the sequence  $E_k(iv \parallel ctr_0+1), \dots, E_k(iv \parallel ctr_0+t)$  is pseudorandom
  - Two-time pad may occur when  $(iv \parallel ctr_0+i)$  wraps around → limit to the maximum number of messages you can encrypt
  - Two-time pad may occur when  $(iv \parallel ctr_0+i) = (iv' \parallel ctr_0'+j)$  but the probability of this event is exponentially small

Apr-23

Block Ciphers

61

61

Block Ciphers: How to avoid ciphertext expansion

# MORE ENCRYPTION MODES: CTS


Apr-23

Block Ciphers

62

62

## Ciphertext Stealing (CTS) mode



UNIVERSITÀ DI PISA

- CTS allows encrypting PT that is not evenly divisible into blocks without resulting in any ciphertext expansion
- $\text{sizeof}(\text{ciphertext}) = \text{sizeof}(\text{plaintext})$
- CTS operates on the last two blocks
  - **Intuition:** a portion of the 2nd-last CT block *is stolen* to pad the last PT block


Apr-23

Block Ciphers

63

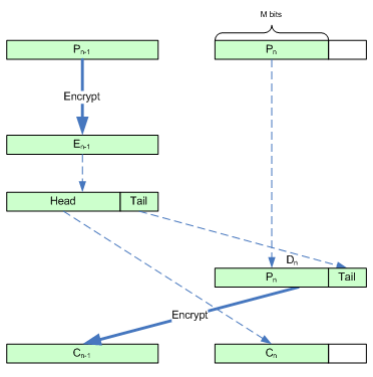
63

## Ciphertext stealing (CTS)



UNIVERSITÀ DI PISA

- CTS + ECB mode



[Picture taken from Wikipedia](#)

Apr-23

Block Ciphers

64

64



# Ciphertext stealing (CTS)

Diagram illustrating Ciphertext Stealing (CTS) in CBC mode. The process shows three plaintext blocks being encrypted sequentially using a Key and an Initialization vector (IV). The first block is a full block, the second is a full block, and the third is a partial block followed by zeros. The output shows the first ciphertext block, the second ciphertext block (partially encrypted twice), and the last ciphertext block.

Picture taken from Wikipedia

Apr-23

Block Ciphers

65

65

# Block Ciphers

## PADDING


Apr-23

Block Ciphers

66

66

## The need for a padding scheme



**Naïve (wrong) solution:** Pad the message with zeroes to the right, without ambiguous boundaries

				0x00	0x00	0x00	0x00
--	--	--	--	------	------	------	------

**Problem:** What if the message was a NULL-terminated string?


							0x00
--	--	--	--	--	--	--	------

At the receiving side: Was it a NULL-terminated string or a 7-bytes pt?

Apr-23Block Ciphers67

67

## The PKCS #5 padding scheme



- Padding is necessary when PT len is not an integer multiple of the block

**If PT len is NOT a block multiple**

- We need b padding bytes
- Fill each padding byte by b

Example: b = 3 then append 0x030303

Block							
H	E	L	L	O	3	3	3

**If PT is a block multiple**

Padding = block  
Fill each padding block by 8

8	8	8	8	8	8	8	8
---	---	---	---	---	---	---	---

Padding causes *ciphertext expansion*

Apr-23Block Ciphers68

68

## PKCS #5: encryption



UNIVERSITÀ DI PISA

- Let  $L$  be the block length (in bytes) of the cipher
- Let  $b$  be the # of blocks that need to be appended to the plaintext to get its length a multiple of  $L$ 
  - $1 \leq b \leq L$
- Before encryption
  - Append  $b$  (encoded in 1 byte),  $b$  times
    - i.e., if  $b = 3$ , append `0x030303`

Apr-23

Block Ciphers

69

69

## PKCS #5: decryption



UNIVERSITÀ DI PISA

- After decryption, say the final byte has value  $b$ 
  - If  $b = 0$  or  $b > L$ , return “error”
  - If the trailing  $b$  bytes are not all equal to  $b$ , return “error”
  - Strip off the trailing  $b$  bytes and output the left as the message


Apr-23

Block Ciphers

70

70

# PKCS #7



UNIVERSITÀ DI PISA

- Difference between PKCS#5 and PKCS#7
  - PKCS#5: padding is defined for 8-byte block sizes (RFC 2898)
  - PKCS#7: padding is defined for block of any size ranging from 1 to 255 bytes (RFC 2315)

Apr-23

Block Ciphers

71

71

Block Ciphers

# PADDING ORACLE ATTACK

Apr-23

Block Ciphers

72

72

## Padding Oracle Attack



UNIVERSITÀ DI PISA

- The attacker
  - intercepts  $y$  and wants to obtain  $x$  (*ciphertext-only attack*)
  - modifies  $y$  into  $y'$  and submits to the receiver
- The receiver (the padding oracle)
  - Receiver decrypts  $y'$  and returns “error”, if  $x'$  is not properly formatted
- On padding oracles
  - Frequently present in web applications
  - Error, receiver timing, receiver behaviour,...

Apr-23

Block Ciphers

73

73

## Main idea of the attack



UNIVERSITÀ DI PISA

- For simplicity, let the ciphertext be a two-block ciphertext  $(IV, y)$ , with  $y = E_k(x)$ 
  - So, at the receiving site,  $x = D_k(y) \oplus IV$
- Message  $x$  is well formatted (padding)
- Main intuition
  - If the attacker changes the  $i$ -th byte of  $IV$ , this causes a predictable change (only) to the  $i$ -th byte of  $x$


Apr-23

Block Ciphers

74

74

The attack – step 1 – padding lenght

  
UNIVERSITÀ DI PISA

$D_k(y)$

yy	yy	yy	yy	yy	yy	yy	yy
----	----	----	----	----	----	----	----

$\oplus$

IV

AB	01	4F	21	00	7C	02	9E
----	----	----	----	----	----	----	----

$=$

$x$

XX	XX	XX	XX	XX	XX	XX	XX
----	----	----	----	----	----	----	----


Apr-23

Block Ciphers

75

75

The attack – step 2 – determine pt

  
UNIVERSITÀ DI PISA

$D_k(y)$

yy	yy	yy	yy	yy	yy	yy	yy
----	----	----	----	----	----	----	----

$\oplus$

IV

AB	01	4F	21	00	7C	02	9E
----	----	----	----	----	----	----	----

$\leftarrow 9F = 9E \oplus 06 \oplus 07$

$=$

$x$

XX	XX	06	06	06	06	06	06
----	----	----	----	----	----	----	----

$\leftarrow 07$


Apr-23

Block Ciphers

76

76

The attack – step 2 – determine pt

  
UNIVERSITÀ DI PISA

$D_k(y)$

yy	yy	yy	yy	yy	yy	yy	yy
----	----	----	----	----	----	----	----

$\oplus$

IV

AB	01	4E	20	01	7D	03	9F
----	----	----	----	----	----	----	----

$=$

x

XX	XX	07	07	07	07	07	07
----	----	----	----	----	----	----	----


Apr-23

Block Ciphers

77

77

The attack – step 2 – determine pt

  
UNIVERSITÀ DI PISA

$D_k(y)$

yy	yy	yy	yy	yy	yy	yy	yy
----	----	----	----	----	----	----	----

$\oplus$

IV

AB	41	4E	20	01	7D	03	9F
----	----	----	----	----	----	----	----

$=$

x

XX	07	07	07	07	07	07	07
----	----	----	----	----	----	----	----

Apr-23


Block Ciphers

78

78

# Attack complexity

- At most  $L$  tries to learn the # of padding bytes
- At most  $2^8 = 256$  tries to learn each plaintext byte



UNIVERSITÀ DI PISA


Apr-23

Block Ciphers

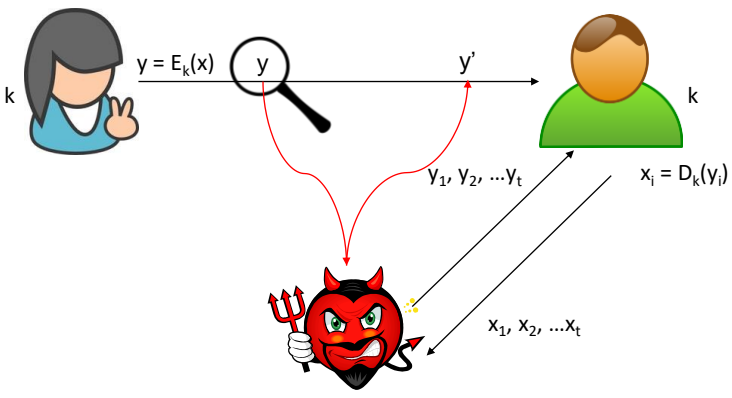
79

79

# CCA model



UNIVERSITÀ DI PISA



The diagram illustrates the Chosen Plaintext Attack (CCA) model. A sender (blue person) encrypts a message  $x$  using key  $k$  to produce ciphertext  $y = E_k(x)$ . An attacker (red devil) intercepts  $y$  and chooses a new ciphertext  $y'$ . The receiver (green person) decrypts  $y'$  using key  $k$  to produce plaintext  $x_i = D_k(y_i)$ . The attacker also receives a sequence of ciphertexts  $y_1, y_2, \dots, y_t$  and a corresponding sequence of plaintexts  $x_1, x_2, \dots, x_t$ .

Apr-23

Block Ciphers

80

80



## Chosen-ciphertext attack



UNIVERSITÀ DI PISA

- Now the attacker becomes active
- The CCA
  - The attacker intercepts  $y = E_k(x)$  and modifies it into  $y'$
  - The receiver decrypts  $y'$  and returns (the attacker) either  $x'$  or some information about  $x'$
  - The adversary can derive either  $x$  or some information about  $x$
- CCA and malleability
  - CCA-security implies non-malleability

Apr-23

Block Ciphers

81

81

## CCA-security



UNIVERSITÀ DI PISA

- Chosen-ciphertext attacks represent a significant, real-world threat
- Modern encryption schemes are designed to be CCA-secure

Apr-23

Block Ciphers


82

82

# XTS-AES MODE

83

# XTS-AES Mode for Block-Oriented Storage Devices



UNIVERSITÀ DI PISA

- IEEE Std 1619-2007
  - Standard describes an encryption mode for data stored in sector-based devices where the threat model includes possible access to stored data by the adversary
  - Has received widespread industry support
- Approved as an additional block cipher mode of operation by NIST in 2010

Apr-23

Block Ciphers

84




UNIVERSITÀ DI PISA

- IEEE Std 1619-2007
  - Standard describes an encryption mode for data stored in sector-based devices where the threat model includes possible access to stored data by the adversary
  - Has received widespread industry support
- Approved as an additional block cipher mode of operation by NIST in 2010

84

# Assumptions



UNIVERSITÀ DI PISA

- Hard disk organized in tracks and sectors
- A sector is the read/write unit
- Sector size is typically 512 byte
- A sector may be divided up in blocks
- Encryption
  - Use all the space
  - Depends only on a) Cleartext, b) Encryption key, c) Sector number and block number


Apr-23

Block Ciphers

85

85

# XTS-AES – Requirements (→)



UNIVERSITÀ DI PISA

- The requirements for encrypting stored data, also referred to as “data at rest”, differ somewhat from those for transmitted data
- The ciphertext is freely available for an attacker
- The data layout is not changed on the storage medium and in transit
- Data are accessed in fixed sized blocks, independently from each other


Apr-23

Block Ciphers

86

86

## XTS-AES – Requirements (→)



UNIVERSITÀ DI PISA

- Encryption is performed in 16-byte blocks, independently from each other
- There are no other metadata used, except the location of the data blocks within the whole data set
- The same plaintext is encrypted to different ciphertexts at different locations, but always to the same ciphertext when written to the same location again


Apr-23

Block Ciphers

87

87

## XTS-AES – Requirements (↓)



UNIVERSITÀ DI PISA

- A standard conformant device can be constructed for decryption of data encrypted by another standard conformant device

Apr-23


Block Ciphers

88

88

## CTR and CB are inadequate

- CTR is malleable
- CBC (with IV = location)
  - Only the first block depends on location
  - Malleable



UNIVERSITÀ DI PISA

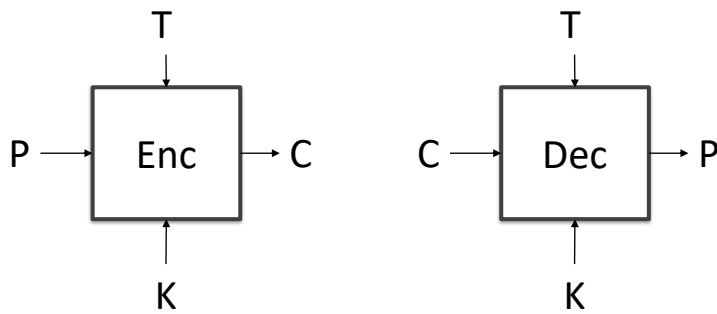
Apr-23

Block Ciphers


89

89

## Tweakable Block Ciphers



The diagram illustrates the structure of tweakable block ciphers. On the left, an encryption process 'Enc' takes a plaintext 'P' as input and produces a ciphertext 'C'. It is also influenced by a public tweak 'T' (indicated by a downward arrow) and a secret key 'K' (indicated by an upward arrow). On the right, a decryption process 'Dec' takes a ciphertext 'C' as input and produces the plaintext 'P'. It is similarly influenced by the same public tweak 'T' (downward arrow) and secret key 'K' (upward arrow).



UNIVERSITÀ DI PISA

- T is public
- K provides security while T provides variability

Apr-23

Block Ciphers

90

90

## Tweakable Block Ciphers

The diagram illustrates the structure of tweakable block ciphers. On the left, the encryption process takes a plaintext  $P$  and a tweak  $T$  as inputs. The tweak  $T$  is processed by a function  $H$ . The output of  $H$  is XORed ( $\oplus$ ) with the plaintext  $P$ . The result then passes through an encryption block  $Enc_K$ . The output of  $Enc_K$  is XORed ( $\oplus$ ) with the output of  $H$  again to produce the ciphertext  $C$ . On the right, the decryption process takes a ciphertext  $C$  and the same tweak  $T$ . The tweak  $T$  is again processed by  $H$ . The output of  $H$  is XORed ( $\oplus$ ) with the ciphertext  $C$ . The result then passes through a decryption block  $Dec_K$ . The output of  $Dec_K$  is XORed ( $\oplus$ ) with the output of  $H$  again to produce the plaintext  $P$ .

UNIVERSITÀ DI PISA

Apr-23      Block Ciphers      91

91

## XTS-AES: block encryption

The diagram shows the XTS-AES block encryption process. A sector index  $i$  is input to an encryption block  $Enc$  along with a key  $K_2$ . The output of this block is multiplied ( $\otimes$ ) by a constant  $\alpha^j$ , where  $j$  is the block index within the sector. This result is then XORed ( $\oplus$ ) with the plaintext  $P$ . The result then passes through a second encryption block  $Enc$ , which uses key  $K_1$ . The output of this second block is XORed ( $\oplus$ ) with the result from the first XOR operation to produce the final ciphertext  $C$ .

- $Enc/Dec = AES$
- Tweak =  $i, j$
- $i$ : sector
- $j$ : block in the sector
- $\otimes$  multiplication in  $GF(2^{128}) \bmod x^{128} + x^7 + x^2 + x + 1$
- $\alpha$  = primitive polynomial  $x (000\dots010)_2$  of  $GF(2^{128})$

UNIVERSITÀ DI PISA

Apr-23      Block Ciphers      92

92

# XTS-AES: block decryption

- Enc/Dec = AES
- Tweak =  $i, j$
- $i$ : sector
- $j$ : block in the sector
- $\otimes$  multiplication in  $GF(2^{128})$   
mod  $x^{128} + x^7 + x^2 + x + 1$
- $\alpha$  = primitive polynomial  $x (000...010)_2$  of  $GF(2^{128})$

Apr-23

Block Ciphers

93

93

# XTS-AES: sector enc/dec

(a) Encryption

(b) Decryption


Apr-23

Block Ciphers

94

94

# Implementations



- Software
  - BestCrypt, dm-crypt, FreeOTFE, TrueCrypt, DiskCryptor, FreeBSD e OpenBSD+
  - Nativo in Mac OS X Lion (nel FileVault)
  - BitLocker di Windows 10
- Hardware
  - SPYRUS Hydra PC Digital Attaché
  - Kingston DataTraveler 5000

Apr-23

Block Ciphers

95

95

Apr-23

Block Ciphers

96

96