Network Security Elements of Applied Cryptography

Symmetric encryption

- Block ciphers and operation modes
- Perfect ciphers
- One-time Pad
- Practical security and unconditional security

Symmetric Encryption Scheme

$\mathcal{M}$: message space

$\mathcal{C}$: ciphertext space

$\mathcal{K}$: keyspace

$E : \mathcal{P} \times \mathcal{K} \rightarrow \mathcal{C}$ encryption transformation

$D : \mathcal{C} \times \mathcal{K} \rightarrow \mathcal{P}$ decryption transformation

Two properties

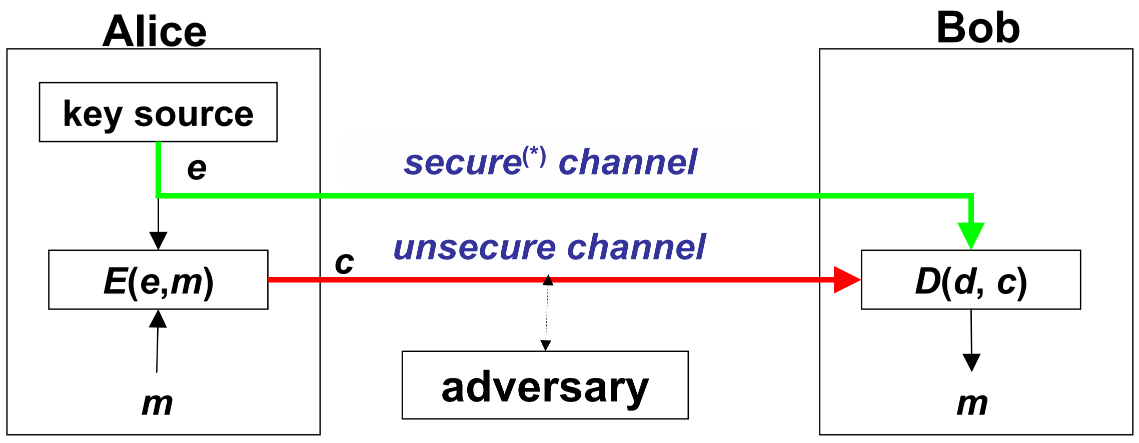
- $\forall m \in \mathcal{M}, \forall e \in \mathcal{K}, \exists d \in \mathcal{K} : m = D(d, E(e, m))$
- It is *computationally* "*easy*" to compute *d* knowing *e*, and viceversa

In most practical symmetric encryption scheme *e* = *d*

Security of a symmetric cipher (intuition)

- Let ***c*** **=** ***E*****(*e*, *m*)** and ***m*** **= D(e, c)**
- **The symmetric cipher (E, D) is secure iff**
  - Given ***c*** is **difficult** to determine ***m*** without knowing **e**, and viceversa
  - Given ***c*** and m is **difficult** to determine ***e***, unless it is used just once

2-party comm with symmetric encryption



- Alice and Bob know ***E*** and ***D***
- Alice and Bob trust each other
- key ***e*** is a shared secret between Alice and Bob

(*) the channel is not ***physically*** accessible to the adversary and ensures both confidentiality and integrity

Discussion

- How can Bob be sure that ***m* = *D*(*k*,*c*)** is good?
  - Bob knows ***m*** in advance
  - Bob knows a part of ***m*** in advance (e.g., email)
  - Bob knows that ***m*** has certain structural redundancies (e.g., ASCII)

Discussion

EXAMPLE (DES-CBC)

- Bob receives

  c = `f3 9e 8a 73 fc 76 2d 0f`
  `59 43 bd 85 c3 c9 89 d2`
  `bf 96 b6 4f 34 b8 51 dd`

- Bob deciphers ***c*** with

  k = `0x3dd04b6d14a437a9`

- Bob obtains
  - ***m*** = "`Ci vediamo alle 20!`"

Discussion

What is the effect of a "small" change in the ciphertext?

- Single bit change
  - ▶ $c[0]_7 = \sim c[0]_7$ (`73 9e 8a 73 fc ...`)
  - ▶ ***m′***="`e8¢biö=}o alle 20:00!`"
- Single byte change
  - ▶ c[c.lenght() – 1] = 0x00 (`... 34 b8 51 00`)
  - ▶ ***m′***="`Ci vediamo alle "}2gÀlõ`"

Discussion

- Upon *seeing* ***m***, Bob *believes* that:
  - only Alice saw message ***m*** (privacy)
  - message ***m*** comes from Alice (?provenience?)
  - message ***m*** has not been modified (?integrity?)

On trust

What does "Alice and Bob trust each other" mean?

- Alice (Bob) believes that Bob (Alice) does not reveal ***m***
- Alice (Bob) believes that Bob (Alice) keeps key ***e*** secret, i.e.,
  - Alice (Bob) believes that Bob (Alice) is competent to do key management
  - Alice (Bob) believes that Bob (Alice) does not reveal the key

Symmetric ciphers

- **Block ciphers** are encryption schemes which break up the plaintext in blocks of fixed lenght *t* bits and encrypt one block at time
- **Stream ciphers** are simple block ciphers in which *t* = 1 and the encryption function can change for each bit

Block cipher

$$\begin{tikzcd}
P \arrow[r] & \boxed{E} \arrow[r] & C \arrow[rr, dashed] & & C \arrow[r] & \boxed{D} \arrow[r] & P \\
& K \arrow[u] \arrow[rr, dash] & & \text{random} & & K \arrow[u] \arrow[ll, dash] &
\end{tikzcd}$$

|*P*| = |*C*| = ***n*** bits (block lenght)

|*K*| = ***k*** bits (key lenght)

$K \in \mathcal{K} \subseteq V_k$

$P \in \mathcal{P} \subseteq V_n$

$C \in \mathcal{C} \subseteq V_n$

V_i set of *i*-bits vectors

For any *K*,

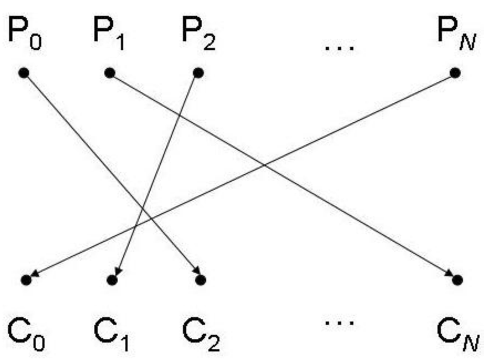
- *E*(*K*, *P*) must be an *invertible* mapping from V_n to V_n and
- *D(K, P)* is the *inverse function*

- *E*(*K*, *P*) will be often denoted by $E_K(P)$

True random cipher

For any key *K*, E_K defines a particular substitution (permutation)

$N = 2^n$



- All the possible substitutions are $2^n!$
- Therefore, in a true random cipher, the key lenght $k = \lg(2^n!) \approx (n - 1.44)\, 2^n$ i.e., the **key lenght is 2^n times the block lenght**
- This key lenght is impractical

In practice, the encryption function corresponding to a randomly chosen key ***should appear*** a randomly chosen invertible function

Standard assumptions

- The objective of the adversary is
  to recover the plaintext from the ciphertext (partial break) or even the key (total break)
- **Standard assumptions**. An adversary
  1. has access to all data transmitted over the ciphertext channel;
  2. (Kerckhoff's assumption) knows all details of the encryption function except the secret key

Classification of attacks

- Attacks are classified according to what information an adversary has access to
  - **ciphertext-only attack**
  - **known-plaintext attack**
  - **chosen-plaintext attack**

↓ **stronger**

- A cipher secure against chosen-plaintext attacks is also secure against ciphertext-only and known-plaintext attack
- It is customary to use ciphers resistant to a chosen-plaintext attack even when mounting that attack is not practically feasible

Computational (practical) security

- A cipher is **computationally** (**practically**) **secure** if the perceived level of computation required to defeat it, ***using the best attack known***, exceeds, by a comfortable margin, the ***computation resources of the hypothesized adversary***
- **The adversary is assumed to have a limited computation power**

Attack complexity

- **Attack complexity** is the dominant of:
  - **data complexity** — expected number of input data units required
    - Ex.: exhaustive data analysis is O(2^n)
  - **storage complexity** — expected number of storage units required
  - **processing complexity** — expected number of operations required to processing input data and/or fill storage with data
    - Ex.: exhaustive key search is O(2^k)

Attack complexity

- A block cipher is **computationally secure** if
  - ***n*** is sufficiently large to preclude exhaustive data analysis
  - ***k*** is sufficiently large to preclude exhaustive key search
  - ***no known attack*** has data complexity and processing complexity significantly less than, respectively, 2^n and 2^k

Exhaustive key search

- Number of processors necessary to break a key
- Every processor performs 10^6 encryption/second

Key size (bit)	1 Year	1 Month	1 Week	1 Day
56	2,300	28,000	120,000	830,000
64	590,000	7,100,000	3.1×10^7	2.1×10^8
128	$1,1\times10^{25}$	$1,3\times10^{26}$	$5,6\times10^{26}$	$3,9\times10^{27}$

Exhaustive key search

- Cost of a year-2005 hardware cracker

1 Year	1 Month	1 Week	1 Day
56 bit			
\$2000	\$24,000	\$100,000	\$730,000
64 bit			
\$510,000	\$6.2M	\$27M	\$190M
128 bit			
$\$9.4\times10^{24}$	$\$1.2\times10^{26}$	$\$4.9\times10^{26}$	3.3×10^{27}

Exhaustive key search

- Exhaustive key search is a known-plaintext attack
- Exhaustive key search may be a ciphertext-only attack if the plaintext has known redundancy
- Exhaustive key search has widespread applicability since cipher operations (including decryption) are generally designed to be computationally efficient
- Given $\lceil (k+4)/n \rceil$ pairs of plaintext-ciphertext, a key can be recovered by exhaustive key search in an expected time $O(2^{k-1})$
  - Exhaustive DES key search requires 2^{55} decryptions and one plaintext-ciphertext pair

Exhaustive data analysis

- A dictionary attack is a known-plaintext attack
- A dictionary attack requires to assemble plaintext-ciphertext pairs for a fixed key
- A complete dictionary requires at most 2^n pairs

Cryptoanalysis: an historical example

Monoalphabetic substitution

Cleartext alphabet	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Key	J	U	L	I	S	C	A	E	R	T	V	W	X	Y	Z	B	D	F	G	H	K	M	N	O	P	Q

- **The key is a permutation of the alphabet**
- **Encryption algorithm**: every cleartext character having position *p* in the alphabet is *substituted* by the character having the same position *p* in the key
- **Decryption algorithm**: every ciphertext character having position *p* in the key is *substituted* by the character having the same position *p* in the cleartext
- **Number of keys** = $26! - 1 \approx 4 \times 10^{26}$ (≥ number of seconds since universe birth)

Cryptoanalysis: an historical example

P = "**TWO HOUSEHOLDS, BOTH ALIKE IN DIGNITY,**
IN FAIR VERONA, WHERE WE LAY OUR SCENE"

("Romeo and Juliet", Shakespeare)

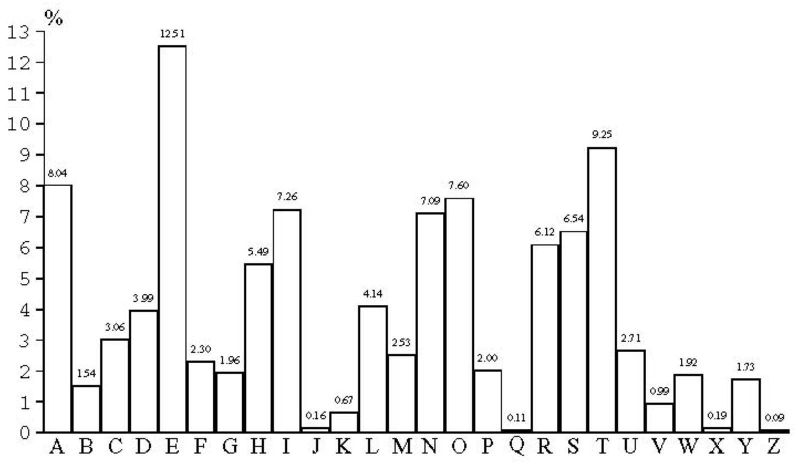
P′ = "**TWOHO USEHO LDSBO THALI KEIND IGNIT**
YINFA IRVER ONAWH EREWE LAYOU RSCEN E"

C = "**HNZEZ KGSEZ WIGUZ HEJWR VSRYI RAYRH**
PRYCJ RFMSF ZYJNE SFSNS WJPZK FGLSY S"

Cryptoanalysis: an historical example

- The monoalphabetic-substitution cipher maintains the redundancy that is present in the cleartext
- It can be "easily" cryptoanalized with a ciphertext-only attack based on language statistics

Frequency of single characters in English text



Linear/differential cryptoanalysis

- **Linear cryptonalysis**
  - è una tecnica di crittoanalisi per cifrari a blocchi ed a caratteri
  - Attribuita a Mitsuru Matsui (1992)
- **Differential cryptoanalysis**
  - è una tecnica di crittoanalisi principalmente concepita per cifrari a blocchi ma che può essere applicata anche ai cifrari a caratteri
  - Attribuita a to Eli Biham and Adi Shamir verso la fine degli anni `80

Security of DES

attack method	data complexity		storage complexity	processing complexity
	known	chosen		
exhaustive precomputation	—	1	2^{56}	1*
exhaustive search	1	—	negligible	2^{55}
linear cryptanalysis	2^{43} (85%)	—	for texts	2^{43}
	2^{38} (10%)	—	for texts	2^{50}
differential cryptanalysis	—	2^{47}	for texts	2^{47}
	2^{55}	—	for texts	2^{55}

* Table lookup
%: probability of success

- Linear cryptanalysis is a known-plaintext attack
- Differential cryptanalysis is primarily a chosen-plaintext attack

Cryptoanalysis of DES

- **Linear cryptonalysis**
  - A known-plaintext attack has O(2^{43}) data complexity and O(2^{43}) computation complexity.
    - With a chosen-plaintext attack, data complexity can be reduced by a factor of 4.
- **Differential cryptoanalysis**
  - Known-plaintext attack has O(2^{55}) data complexity and O(2^{55}) computation complexity
  - Chosen-plaintext attack has O(2^{47}) data complexity and O(2^{47}) computation complexity
  - DES is "surprisingly" resilient to DC.
- LC is the "best" analytical attack but is considered unpractical

Perfect cipher

- Shannon's theory

Cifrario perfetto

- Con un cifrario perfetto, il crittoanalista, esaminando un crittogramma *c* non acquisisce sul messaggio *m* alcuna conoscenza di cui non disponesse già prima
- Shannon (1949) formalizzò questa proprietà in termini di processi stocastici come segue.
- Sia *M* una variabile aleatoria che assume valori nello spazio $\mathcal{M}$ dei messaggi
- Sia *C* una cariabile aleatoria che assume valori nello spazio $\mathcal{C}$ dei crittogrammi
- **DEFINIZIONE**. Un cifrario è perfetto se per ogni $m \in \mathcal{M}$ e per ogni $c \in \mathcal{C}$, vale la relazione $\mathcal{P}(\mathbf{M = m} \mid \mathbf{C = c}) = \mathcal{P}(\mathbf{M = m})$

Cifrario perfetto

- **TEOREMA**. In un cifrario perfetto, il numero delle chiavi deve essere maggiore o uguale al numero dei messaggi possibili
- **Dimostrazione (per assurdo)**. Sia N_m il numero dei messaggi e N_k il numero delle chiavi.

1. $|\mathcal{M}| = N_m \le |\mathcal{C}|$ altrimenti il cifrario non è invertibile
2. Supponiamo per assurdo che $N_k < N_m$. Quindi $N_k < |\mathcal{C}|$
3. Sia m, $\mathcal{P}(M = m) \neq 0$. Da (2) segue che esiste $c' \in \mathcal{C}$ che non è immagine di m. Perció

   $\mathcal{P}(M = m \mid C = c') = 0 \neq \mathcal{P}(M = m)$

   contro l'ipotesi che il cifrario sia perfetto

Encryption modes and Multiple encryption

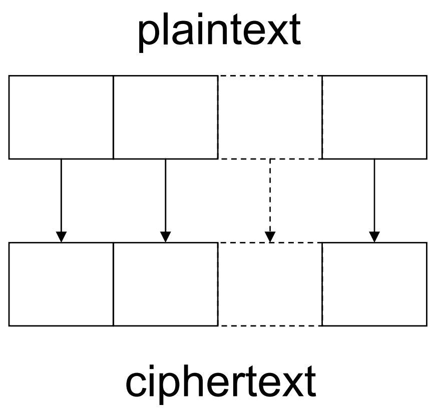
- Electronic CodeBook
- Cipher Block Chaining
- 3DES (EDE, EEE)

Encryption modes

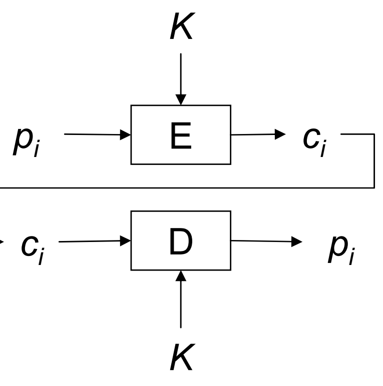
- A block cipher encrypts plaintext in fixed-size *n*-bit blocks
- When the plaintext exceeds *n* bit, there exist several methods to use a block
  - Electronic codebook (ECB)
  - Cipher-block Chaining (CBC)
  - Cipher-feedback (CFB)
  - Output feedback (OFB)

Encryption modes: ECB

- **Electronic Codebook (ECB)**



$$\forall 1 \le i \le t, c_i \leftarrow E_k(p_i)$$
$$\forall 1 \le i \le t, p_i \leftarrow D_k(c_i)$$



plaintext blocks are encrypted separately

Encryption modes: ECB

Properties

- Identical plaintext results in identical ciphertext
  - ▶ ECB doesn’t hide data patterns
- No chaining dependencies: blocks are enciphered independently of other blocks
  - ▶ ECB allows block reordering and substitution

▶ Error propagation: one or more bit errors in a single ciphertext block affects decipherment of that block only

Encryption modes: ECB

AN EXAMPLE OF BLOCK REPLAY

- A bank transaction transfers a client U’s amount of money D from bank B1 to bank B2
  - Bank B1 debits D to U
  - Bank B1 sends the “credit D to U” message to bank B2
  - Upon receiving the message, Bank B2 credits D to U
- Credit message format
  - Src bank: *M* (12 byte)
  - Rcv banck: *R* (12 byte)
  - Client: *C* (48 byte)
  - Bank account: *N* (16 byte)
  - Amount of money: *D* (8 byte)
- Cifrario (n = 64 bit; modalità ECB)

Encryption modes: ECB

AN EXAMPLE OF BLOCK REPLAY

- Mr. Lou Cipher is a client of the banks and wants to make a fraud.
- Lou Cipher is an active adversary and wants to replay a Bank B1's message "credit 100$ to Lou Cipher" many times
- Attack strategy
  - The adversary 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

Encryption modes: ECB

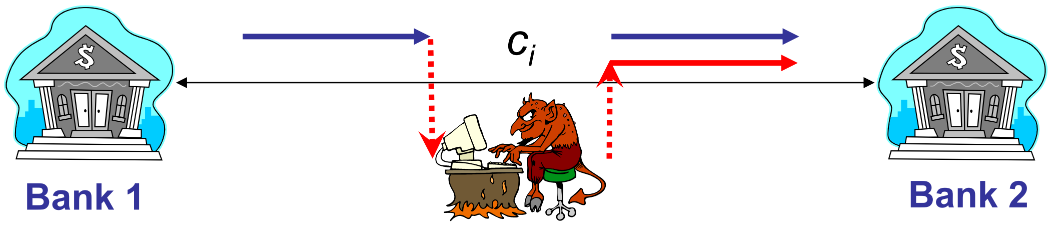
AN EXAMPLE OF BLOCK REPLAY

1. The adversary performs **k** equal transfers
   - credit 100$ to Lou Cipher ⇒ c_1
   - credit 100$ to Lou Cipher ⇒ c_2
   - …
   - credit 100$ to Lou Cipher ⇒ c_k

$c_1 = c_2 = \ldots = c_k$

COMMENT. ***k*** is large enough to allow the adversary to identify the cryptograms corresponding to its transfers

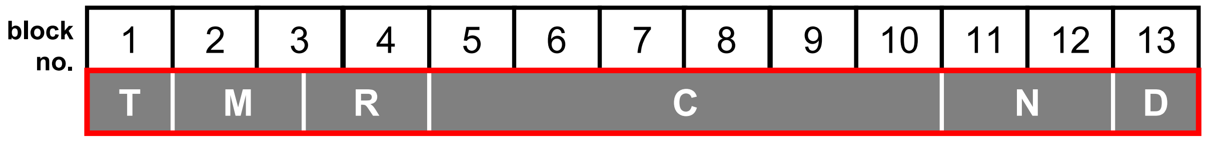
1. The adversary searches "his own" cryptograms over the network
2. The adversary replies one of these cryptograms



Encryption modes: ECB

AN EXAMPLE OF BLOCK REPLAY

- An 8-byte timestamp field ***T*** is added to the message to prevent replay attacks



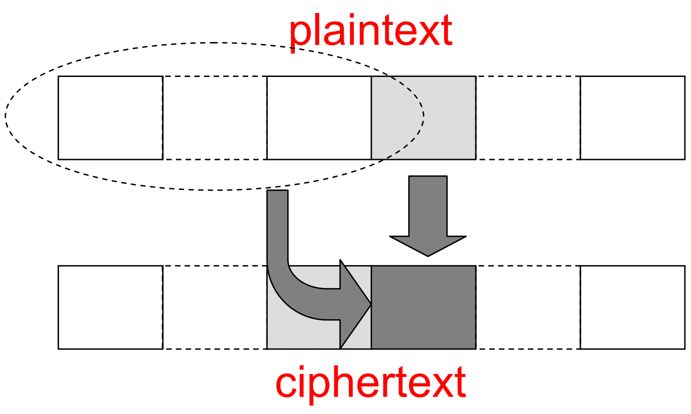
However, the adversary can

1. identify “his own” cryptograms as before by inspecting blocks 2–13;
2. intercept any “fresh” cryptogram;
3. substitute block 1 of “his own” cryptogram with block 1 of the “fresh” cryptogram

Encryption modes: Cipher Block Chaining

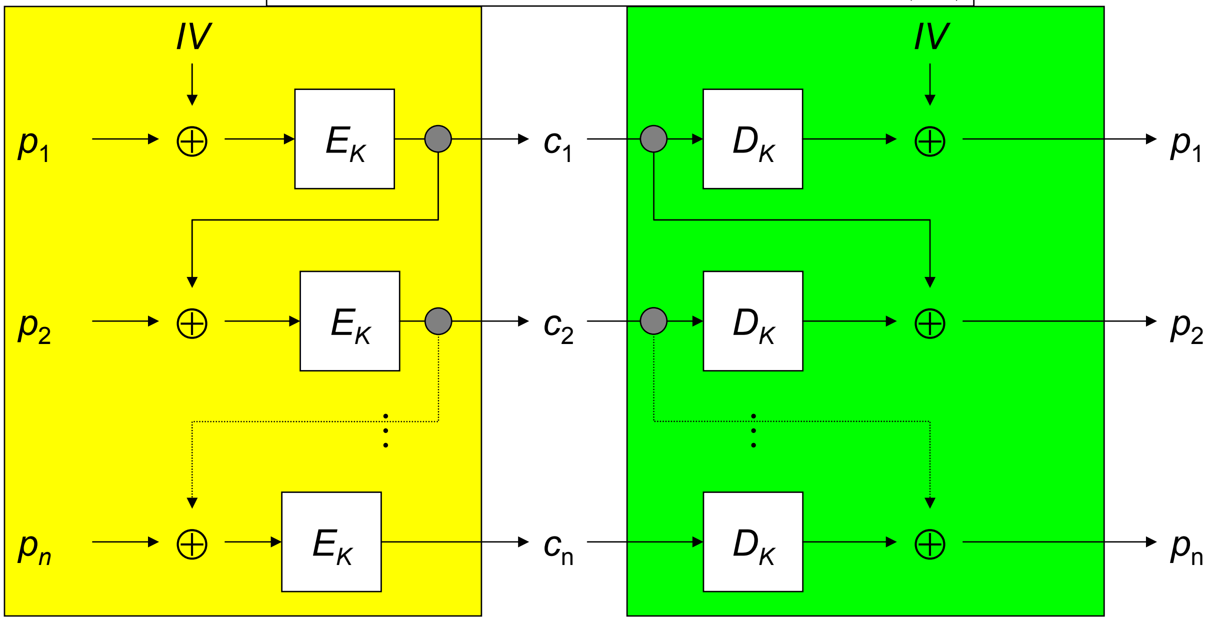
- CBC segue il **principio di diffusione** di Shannon introducendo una **dipendenza di posizione** tra il blocco in elaborazione e quelli precedenti
- CBC è un cifrario a blocchi in cui blocchi identici del messaggio vengono cifrati in modo **diverso** eliminando ogni periodicità

c_i depends on p_i and all preceding plaintext blocks



CBC

$$c_0 \leftarrow IV . \forall 1 \le i \le t, c_i \leftarrow E_k\left(p_i \oplus c_{i-1}\right)$$
$$c_0 \leftarrow IV . \forall 1 \le i \le t, p_i \leftarrow c_{i-1} \oplus D_k\left(c_i\right)$$



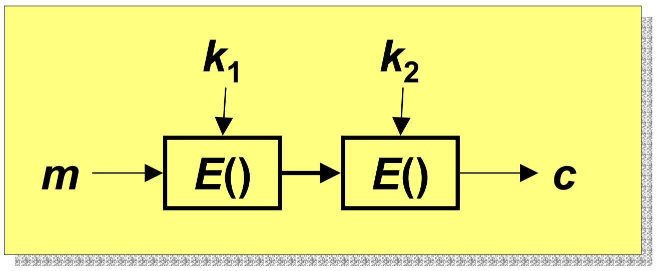
CBC: properties

- Identical ciphertext result from the same plaintext under the same key and IV
- IV can be sent in the clear; its integrity must be guaranteed
- Chaining dependencies: c_i depends on p_i and all preceding plaintext blocks
  - Ciphertext block reordering affects decryption
- Error propagation: bit errors in c_i affect decryption of c_i and c_{i+1}
- Error recovery: CBC is self-synchronizing or ciphertext autokey
- Framing errors: CBC does not tolerate “lost” bits

Multiple encryption

- If a cipher is subject to exhaustive key search, encipherment of a message more than once **may** increase security
- Multiple encryption may be extended to messages exceeding one block by using standard modes of operation
- Cascade cipher is the concatenation of L ≥ 2 ciphers, each with independent keys
- Multiple encryption is similar to a cascade cipher but the ciphers are identical (either *E* or *D*) and the keys need not be independent

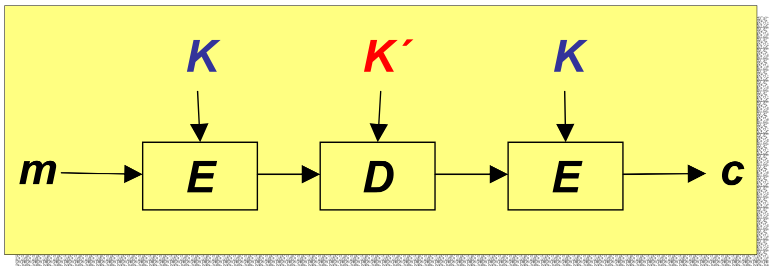
Double encryption



- Double encryption is subject to a known-plaintext attack called "meet-in-the-middle" attack which requires 2^k operations and 2^k storage units

Triple encryption

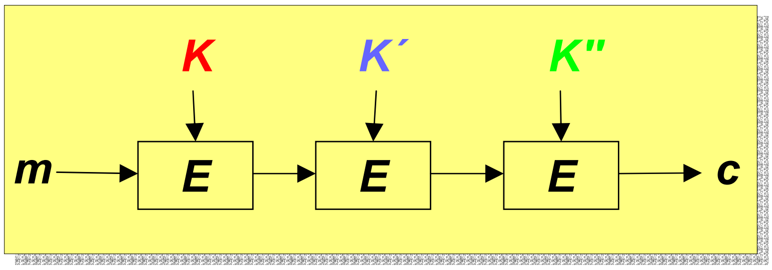
EDE



- Used in financial applications
- ANSI X9.17 e ISO 8732
- A chosen-plaintext attack which requires 2^k operations, 2^k data inputs and 2^k storage units
- A known-plaintext attack requires *p* data inputs, $2^{k+n}/p$ operations, and $O(p)$ storage units
- Backward compatibility: *E* when $K = K'$

Triple encryption

EEE



- A known-plaintext attack similar to meet-in-the-middle, which requires 2^{2k} operations and 2^k units of storage
- With DES, $k = 56$ (DES), the cipher is practically secure

One-time Pad (Vernam, 1917)

- Let m be a ***t***-bit message

  Let k be a sequence of ***t*** randomly chosen bits
- Encryption and decryption functions

  Encryption: $c_i = m_i \oplus k_i$, $0 \leq i \leq t$

  Decryption: $m_i = c_i \oplus k_i$, $0 \leq i \leq t$
- An alternative view of the encryption function

$$E_{k_i}(m_i) = \begin{cases} m_i & k_i = 0 \\ (m_i + 1) \bmod 2 & k_i = 1 \end{cases}$$

- Esempio
  - *m* = 01010101, *k* = 01001110, *c* = 00011011 (si noti che *m* è periodico ma *c* no)

One-Time Pad è un cifrario perfetto

- **TEOREMA**. One-Time Pad è un cifrario perfetto se la chiave è scelta in modo perfettamente random per ogni messaggio e
  1. tutti i messaggi hanno la stessa lunghezza ***t***
  2. tutte le sequenze di ***t*** bit sono messaggi possibili
- **Dimostrazione**.
  - Si deve provare che: $\mathcal{P}(M = m \mid C = c) = \mathcal{P}(M = m, C = c) / \mathcal{P}(C = c) = \mathcal{P}(M = m)$.
  - $\mathcal{P}(M = m, C = c)$ implica che esiste k′ tale che c = m ⊕ k′ e k′ è unica.
  - Sia K una variabile aleatoria che modella il processo di generazione della chiave.
  - $\mathcal{P}(M = m, C = c) = \mathcal{P}(M = m, C = c, K = k')$. (continua)

One-Time Pad è un cifrario perfetto

- Applicando la definizione di probabilità condizionata a ritroso si ottiene
  $\mathcal{P}(M=m, C = c, K = k') =$
  $\mathcal{P}(C = c \mid M = m, K = k') \times \mathcal{P}(M = m, K = k') =$
  $\mathcal{P}(C = c \mid M = m, K = k') \times \mathcal{P}(M = m \mid K = k') \times \mathcal{P}(K = k')$
- Si osservi che
  $\mathcal{P}(C = c \mid M = m, K = k') = 1$, per definizione di k′
  $\mathcal{P}(M = m \mid K = k') = \mathcal{P}(M = m)$, perché la generazione dei messaggi e delle chiavi sono processi indipendenti
  $\mathcal{P}(K = k') = (1/2)^t$, perché la chiave è generata in modo casuale (continua)

One-Time Pad è un cifrario perfetto

- Perciò
  $\mathcal{P}(M=m, C = c) = \mathcal{P}(M = m) \times (1/2)^t$ (*)
  $\mathcal{P}(M=m \mid C = c) = \mathcal{P}(M = m) \times (1/2)^t / \mathcal{P}(C= c)$ (**)
- Per valutare $\mathcal{P}(C= c)$ si consideri la famiglia di eventi $\mathcal{F}_m=\{M = m\}$. Gli eventi della famiglia sono disgiunti e $\mathcal{P}(\cup_m \mathcal{F}_m) = 1$. Ne segue quindi che $\mathcal{P}(C= c) = \Sigma_m \mathcal{P}(C= c, M = m)$ sostituendo la (*) si ottiene
  $\mathcal{P}(C= c) = \Sigma_m \mathcal{P}(M = m) \times (1/2)^t = (1/2)^t \times \Sigma_m \mathcal{P}(M = m) = (1/2)^t$ (***)

(continua)

One-Time Pad è un cifrario perfetto

- sostituendo la (***) nella (**) si ottiene $\mathcal{P}(M=m \mid C=c) = \mathcal{P}(M=m)$ cioè il cifrario è perfetto.
  C.V.D. □

- Si può osservare anche che

  TEOREMA. One-Time Pad utilizza anche il numero minimo di chiavi.

- **Dimostrazione**.
  - Siccome le chiavi sono sequenze arbitrarie di t bit e $N_m \leq N_k$ allora $|\mathcal{M}| = |\mathcal{K}| = 2^t$. C.V.D. □

One-Time Pad

- One-Time Pad is vulnerable to a **known-plaintext attack**
  - key k can be easily obtained from m and c: $\boldsymbol{k_i = m_i \oplus c_i}$
- **The key must be used only once**.

  Let us suppose that a key ***k*** **is used twice**, that is $c = m \oplus k$ and $c' = m' \oplus k$.

  It follows that $c \oplus c' = m \oplus m'$.

  This provides important information to a cryptanalyst who has both c and c'. For instance, a sequence of zeros in $c \oplus c'$ corresponds to equal sequences in m and m'
- One-Time Pad requires to generate a key of **many random bits:** this problem is not trivial!

Unconditional security

- **Unconditional security** (**perfect secrecy**)
  - An adversary is assumed to have **unlimited computational resources**
  - The uncertainty in the plaintext after observing the ciphertext must be equal to the a priori uncertainty about the plaintext
  - Observation of the ciphertext provides no information whatsoever to an adversary
- A **necessary condition** for a symmetric-key encryption scheme to be unconditionally secure is that the key bits are chosen randomly and independently and the key is at least as long as the message

Security of one-time pad

- One-time padding is **unconditionally secure** against ciphertext-only attack
- Any *t*-bit plaintext message *m** can be recovered from a *t*-bit ciphertect *c* by using a proper key $k^* = m^* \oplus c$
- The fact that the key should be at least as long as the plaintext complicates key distribution and key management

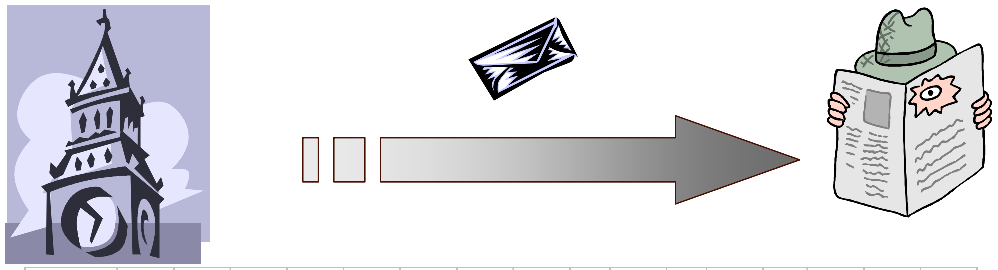
  For this reason, in practice, stream ciphers are used where the key stream is *pseudo randomly* generated from a smaller secret key

  These ciphers are not unconditionally secure but, hopefully, practically secure

One-time pad

- **c[i] = m[i] + k[i] mod 26**
- **m = “SUPPORT JAMES BOND”**

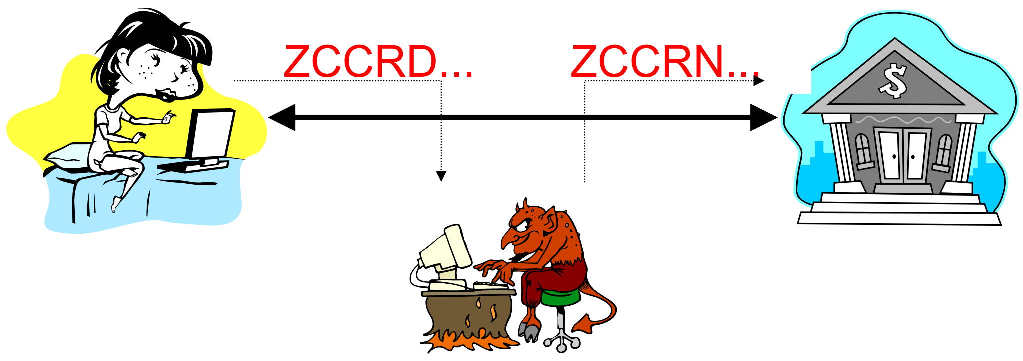
m =	S	U	P	P	O	R	T	J	A	M	E	S	B	O	N	D
k =	W	C	L	N	B	T	D	E	F	J	A	Z	G	U	I	R
c =	O	W	A	C	P	K	W	N	F	V	E	R	H	I	V	U



c =	O	W	A	C	P	K	W	N	F	V	E	R	H	I	V	U
k′ =	M	W	L	J	V	T	S	E	F	J	A	Z	G	U	I	R
m =	C	A	P	T	U	R	E	J	A	M	E	S	B	O	N	D

An insecure systems made by secure components

m =	D	A	R	E	C	E	N	T	O	E	U	R	O	A	B	O	B
k =	W	C	L	N	B	T	D	E	F	J	A	Z	G	U	I	R	X
c =	Z	C	C	R	D	X	Q	X	T	N	U	Q	U	U	J	F	Y



c' =	Z	C	C	R	N	B	O	P	J	N	U	Q	U	U	J	F	Y
k =	W	C	L	N	B	T	D	E	F	J	A	Z	G	U	I	R	X
m =	D	A	R	E	M	I	L	L	E	E	U	R	O	A	B	O	B

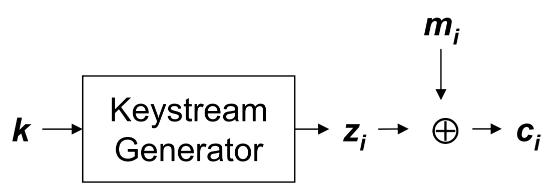
I cifrari a carattere

- WEP

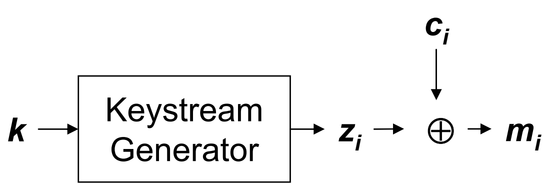
Stream ciphers

- In **stream ciphers**
  - **a plaintext block is as small as one bit** and
  - **the encryption function may vary as plaintext is processed** (stream ciphers have memory)
- **Stream ciphers are faster than block ciphers in hardware**, and have less complex hardware circuitry
- **Stream ciphers are more appropriate or mandatory**
  - when buffering is limited
  - when characters must be processed as they are received
  - when transmission errors are highly probable since they have limited or no error propagation

Synchronous stream ciphers



Encryption

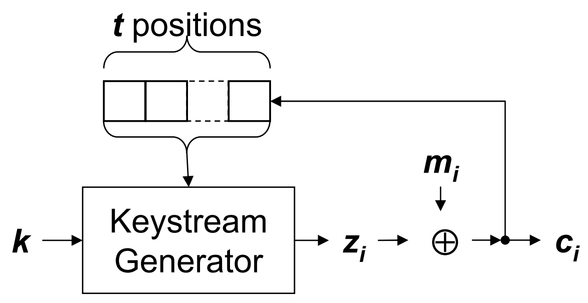


Decryption

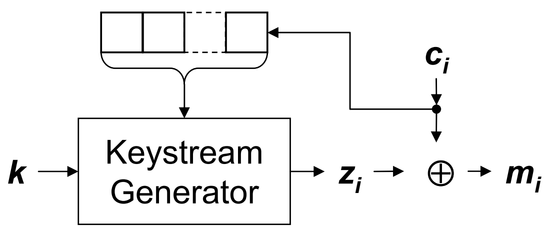
Properties

- Sender and receiver must be synchronized. If a bit is inserted or deleted, decryption fails.
- No error propagation
- Active attacks

Self-synchronizing stream ciphers



Encryption



Decryption

Properties

- Self-synchronization.
- No error propagation
- Active attacks
- Diffusion of plaintext statistics

Key stream generator

The key stream must have the following properties:

- large period
- unpredictable
- good statistics

These are only *necessary conditions* for a KSG to be considered cryptographically secure

- KSGs are computationally secure after public scrutiny (no mathematical proof)

Stream ciphers

- For hardware implementation
  - LFSR-based stream ciphers
- For software implementation
  - SEAL
    - New algorithm (1993) for software implementation on 32-bit processors. It has received not yet much scrutiny
  - RC4
    - commercial products
    - variable key
    - proprietary
  - Output Feedback (OFB), Cipher Feedback (CFB) (modes of block ciphers)

WEP (802.11)

- An example of insecure system made of secure components