Sécurité Informatique
Cryptographie et cryptanalyse

October 17, 2018

Houcemeddine HERMASSI
houcemeddine.hermassi@enit.rnu.tn

École Nationale d'Ingénieurs de Carthage ENI-CARTHAGE
Université Carthage
Tunisie

Plan de cour

Principe

Cryptographie: Transformer un texte clair pour en cacher le sens

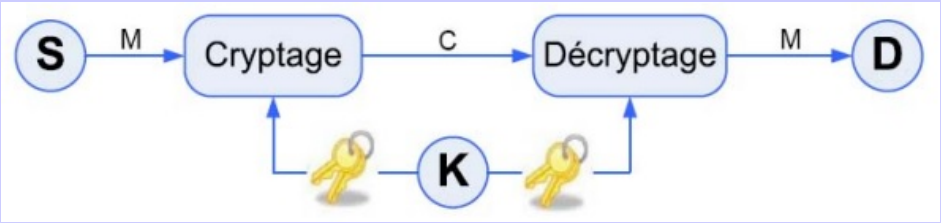
Classification

- Selon le nombre de **clés à utiliser**:
  - Une seul clé, cryptosystème à **clé privée** (symétrique)
  - 2 clés ou cryptosystème à **clé publique** (asymétrique)
- Selon le type d'opérations:
  - **substitution**
  - **transposition**
  - **produit des deux**
- Selon **la façon** dont le texte clair est traité:
  - **Bloc**
  - **flux (stream cipher)**

Terminologie basique

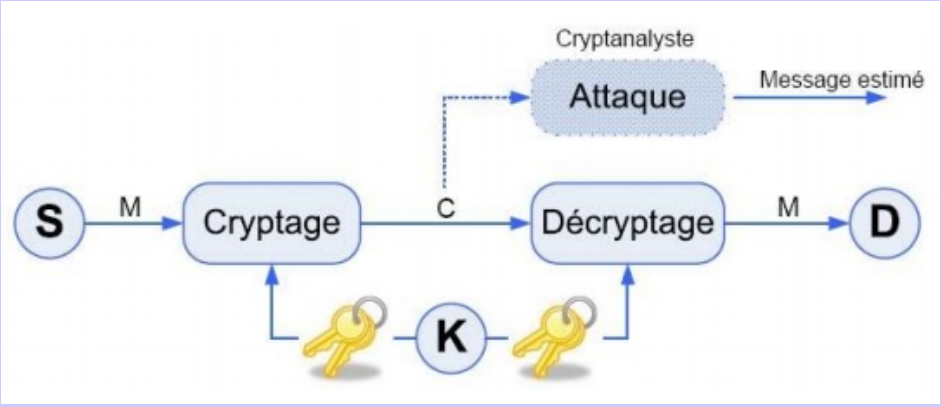
- **Plaintext** : le message original
- **Ciphertext** : le message chiffré
- **chiffrement ou cryptage** : le processus de conversion du plaintext vers le ciphertext
- **déchiffrement ou décryptage** : le processus de conversion du ciphertext vers le plaintext
- **cryptographie** : l'étude des méthodes de cryptage (science des messages secrets)
- **cryptanalyse** : l'étude des techniques pour casser les algorithmes de chiffrement
- **Cryptologie** : la cryptographie et la cryptanalyse

Modèle de cryptage symètrique



- Aussi connu comme cryptage conventionnel ou cryptage à clé secrète.
- c'était le seul type de cryptage jusqu'à invention du cryptage asymétrique ds les années 70.
- reste comme même le cryptage le plus répandu des deux

Principe

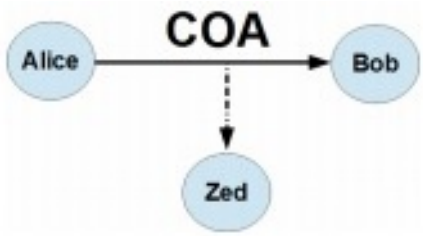


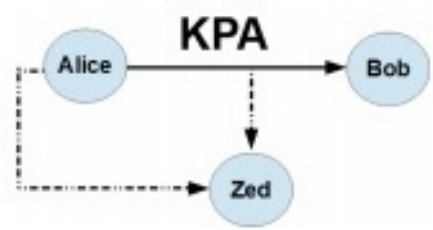
- Son objectif est de retrouver la clé secrète pas simplement le plaintext
- brute force attack (attaque à force brute) :
  - essayer **toutes les combinaisons** (sur une ciphertext pour le déchiffrer) de la clé jusqu'à trouver la bonne
  - En moyenne, il faut essayer au moins la moitié des clés disponibles pour arriver à casser un cryptosystème.
- cryptanalytic attack : **plus intelligente**, **exploite une connaissance** sur l'algorithme et la manière dont le plaintext est traité.

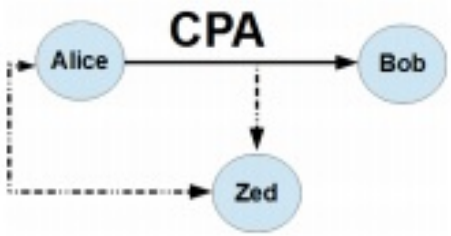
Collecte d'informations : faiblesses théoriques

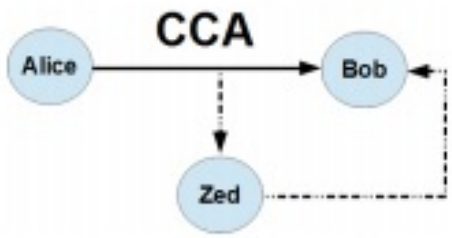
Observation ou action :Étape "on-line" connecté à la cible.

- **Ciphertext-only attack(COA)**
- **Known-plaintext attack(KPA)**
- **Chosen-plaintext attack(CPA)**
- **Chosen-ciphertext attack(CCA)**









Collecte d'informations : faiblesses physiques

Attaques par canal auxiliaire : Side Channel Attack

- Mesure **du temps de cryptage/décryptage** : étude du temps mis pour effectuer certaines opérations
- **Fuites électromagnétiques** : émet des rayonnements qui varient selon les opérations effectuées
- **Analyse du Comportement du processeur lors du calcul** : bruit acoustique
- **Analyse de la consommation d'énergie** : Une consommation accrue indique un calcul important et peut donner des renseignements sur la clé

Analyse, déduction et exploitation

- Étape "off-line" : Analyse & Déduction
  - **Attaque à force brute** : essayer toutes les clés possibles pour retrouver un texte en clair à partir du cryptogramme
  - **Attaque statistique** : Estimer la fréquence d'apparition des lettres dans un texte
  - **Attaque algébrique** : trouver des représentations équivalentes du cryptosystème, exploiter des linéarités existantes.
  - **Cryptanalyse linéaire** : approximation linéaire de l'algorithme de chiffrement, augmenter le nombre de couples pour améliorer l'approximation.
  - **Cryptanalyse différentielle** : étudier la manière dont les différences entre les entrées affectent les différences de leurs sorties pour découvrir des vulnérabilités.
- **Exploitation** : Estimation de la clé et Déchiffrement de tous les cryptogrammes.

Exemple: Brute force attack

Longueur clé (bits)	Nb de clés possibles	Temps requis à 1 dechiffrement/µs	Temps requis à 10^6 dechiffrement/µs
32	$2^{32} = 4.3 \times 10^9$	2^{31} µs = 35.8 minutes	2.15 millisecondes
56	$2^{56} = 7.2 \times 10^{16}$	2^{55} µs = 1142 années	10.01 heures
128	$2^{128} = 3.4 \times 10^{38}$	2^{127} µs = 5.4×10^{24} années	5.4×10^{18} années
168	$2^{168} = 3.7 \times 10^{50}$	2^{167} µs = 5.9×10^{36} années	5.9×10^{30} années
26 lettres (permutation)	$26! = 4 \times 10^{26}$	2×10^{26} µs= 6.4×10^{12} années	6.4×10^6 années

CESAR

- Consiste à **remplacer** les lettres du plaintext par d'autres lettres ou symboles ou bits.
- le plus connu est l'algorithme de Cesar : remplacer chaque lettre par celle qui la suit apres trois positions ds l'alphabet
- L'alphabet est enroulé de sorte que la lettre qui suit Z est A
- ex :

plain : meet me after the toga party
cipher : PHHW PH DIWHU WKH WRJD SDUWB

CESAR: modèlisation mathématique

- l'alg peut etre exprimé comme :

$$c = E(3, p) = (p + 3) mod 26$$

- le decalage peut etre généralisé à n'importe quel nombre k :

$$c = E(k, p) = (p + k) mod 26$$

- si $k \in [1, 25]$, alors le déchiffrement est :

$$p = D(k, c) = (c - k) mod 26$$

CESAR: Brute force attack

Brute force attack sur Cesar : essayer toute les 26 combinaisons

KEY	PHHW PH DIWHU WKH WRJD SDUWB
1	oggv og chvgt vjg vqic rctva
2	nffu nf bgufs uif uphb qbsuz
3	meet me after the toga party
4	ldds ld zesdq sgd snfz ozqsx
5	kccr kc ydrcp rfc rmey nyprw
6	jbbq jb xcqbo qeb qldx mxoqv
7	iaap ia wbpan pda pkcw lwnpu
8	hzzo hz vaozm ocz ojbv kvmot
9	gyyn gy uznyl nby niau julns
10	fxxm fx tymxk max mhzt itkmr
11	ewwl ew sxlwj lzw lgys hsjlq
12	dvvk dv rwkvi kyv kfxr grikp
13	cuuj cu qvjuh jxu jewq fqhjo
14	btti bt puitq iwt idvp epqin
15	assh as othsf hvs hcuo dofhm
16	zrrg zr nsgre gur gbtn cnegl
17	yqqf yq mrfqd ftq fasm bmdfk
18	xppe xp lqepc esp ezrl alcej
19	wood wo kpdob dro dyqk zkbdi
20	vnnc vn jocna cqn cxpj yjach
21	ummb um inbmz bpm bwoi xizbg
22	tlla tl hmaly aol avnh whyaf
23	skkz sk glzkx znk zumg vgxze
24	rjjy rj fkyjw ymj ytlf ufwyd
25	qiix qi ejxiv xli xske tevxc

Monoalphabetic cipher

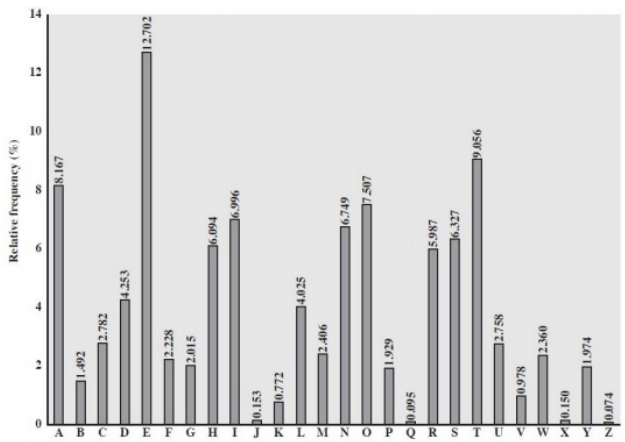
- consiste a remplacer chaque lettre arbitrairement (pas simple décalage)
- la clé est de longueur 26 :
  - Plain : a b c d e f g h i j k l m n o p q r s t u v w x y z
  - Cipher : D K V Q F I B J W P E S C X H T M Y A U O L R G Z N
- exemple :

Plaintext : if we wish to replace letters
Ciphertext : WI RF RWAJ UH YFTSDVF SFUUFYA

Sécurité du crypto monoalphabetique

- On a un total de $26! = 4 \times 10^{26}$ clés possibles, mais on peut le casser par analyse de fréquence : Al-Kindy
- Le langage humain est très redondant, ex ds le msg "th lrd s m shphrd shll nt wnt" les lettres de cette façon ne sont pas ordinaire en anglais
- En anglais la lettre "E" est la plus utilisée, suivie par : "T,R,N,I,O,A,S"
- les lettres comme "Z,J,K,Q,X" sont rares en utilisation.
- il ya des doublets ou des triplets qui sont plus répondu que d'autres.

Fréquences des lettres en anglais



Sécurité du crypto monoalphabetique: Exemple de cryptanalyse

- etant donné un ciphertext :**UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXU DBMETSXAIZVUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSXEPYE-POPDZSZUFPOMBZWPF UPZHMDJUDTMOHMQ**
- On compte la fréquence de chaque lettre ds le ciphertext
- On peut deviner que **P** et **Z** sont **e** et **t**
- On peut deviner que **ZW** est **th** et donc **ZWP** est **the**
- la séquence **ZWSZ** est remplacé par **th*t**, on peut deviner que **S** est **a**

```
UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ
 t a        e  e te  a that e e a       a
VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX
   e t   ta t ha e ee   a e  th    t  a
EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ
 e  e e tat  e   the   t
```

- on continu avec la technique essai-erreur-essai, on trouve le plaintext : "it was disclosed yesterday that several informal but direct contacts have been made with political representatives of the viet cong in moscow"

Principe

- L'algorithme le plus connu qui crypte plusieurs lettres en même temps
- traite les diagrammes (2 lettres) comme unité et la converti en diagramme ciphertext.
- basé sur une matrice 5×5 utilisant un mot clé
- inventé par le Britannique Sir Charles Wheatstone en 1854
- utilisé par l'armée Britannique en W.W.I et par l'USA et ses alliés durant la guerre W.W.II

Matrice Playfair

- copier les lettres du mots clé dans la matrice (sans duplication)
- completer le reste de la matrice par les lettres manquantes
- les lettres **I** et **J** sont traités comme une seule lettre
- ex : en utulisant le mot clé MONARCHY

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

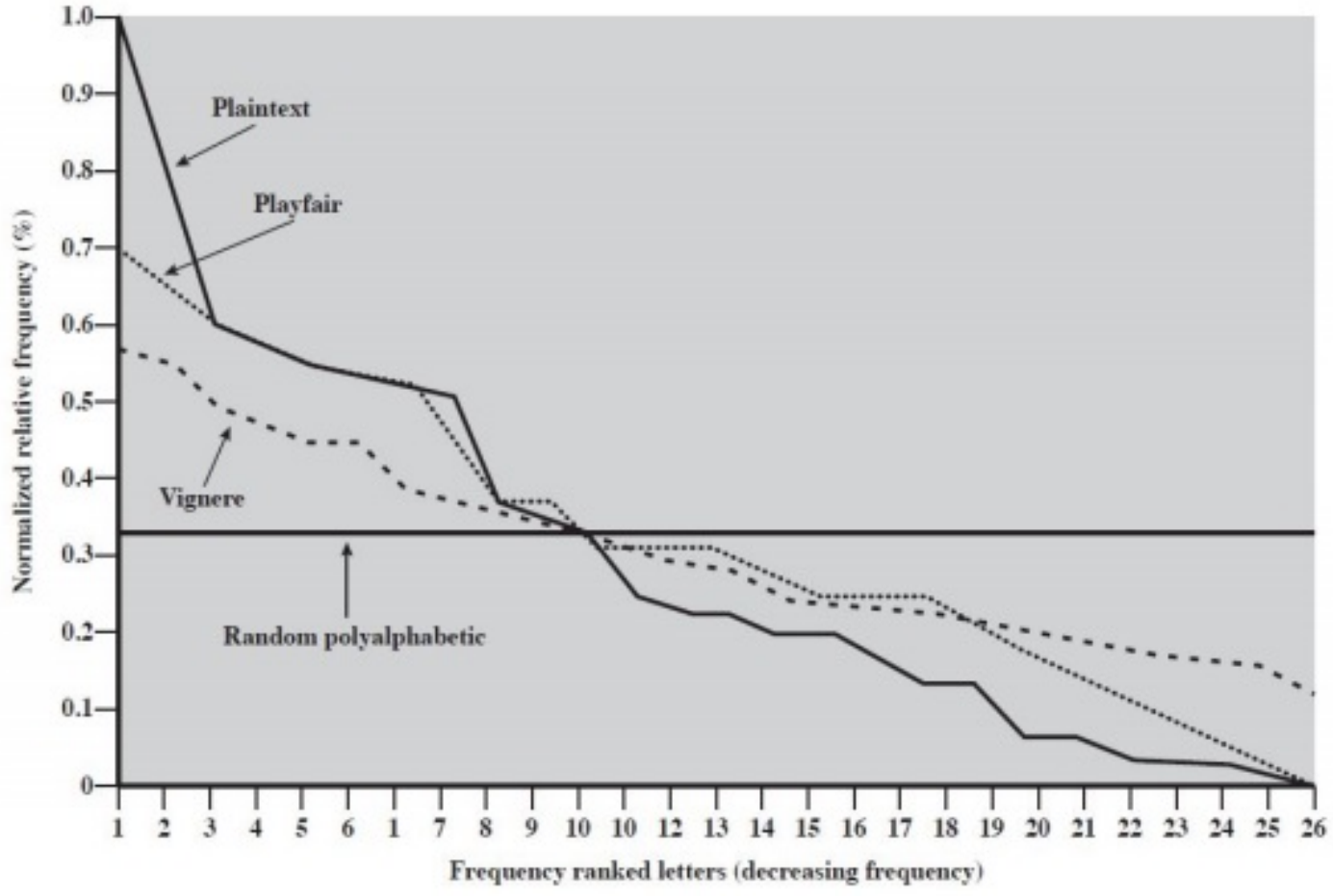
Cryptage Playfair

- opérer sur des diagrammes de lettres (2 lettres) à chaque fois
- cas particulier : si diagramme de même lettres, séparer par des lettres spéciales ex : x. par exemple : **balloon** est traité comme **ba lx lo on**
- Si plaintext dans la même ligne : remplacer par les lettres de droite. Ex1 **pq** est remplacé par **qs**. Ex2 **ar** est remplacé par **RM**.
- Si plaintext dans la même colonne : remplacer par les lettres en-dessous. ex **mu** est remplacé par **CM**
- sinon, remplacer par lettre en même ligne qu'elle et même colonne que l'autre lettre du plaintext. ex **hs** est remplacé par **BP**. ex2 ea devient **IM** ou **JM**

Sécurité de Playfair

- Sécurité amélioré puisque il ya en tout $26 \times 26 = 676$ diagrammes
- on a besoin d'une analyse fréquentielle sur 676 unité et non plus sur 26 comme le monoalphabetique
- donc l'alphabet du ciphertext est aussi énorme
- il peut être cassé si on connaît une centaine de plaintext/ciphertext

fréquence des lettres



Avantages

- améliore la sécurité en combinant plusieurs algorithme mono-alphabetiques
- rende la cryptanalyse plus difficile avec augmentation d'alphabets et une distribution fréquentielle plus platte
- utilise une clé pour choisir quel alphabet mono-alphabetique à utiliser pour chaque lettre du plaintext
- répéter du début si la fin de la clé est atteinte

Vigenère

- l'algorithme poly-alphabetique le plus simple : algorithmes de Cesar muliples
- la clé est constitué de caractères $K = k_1 k_2 \ldots k_d$
- la ième lettre de la clé spécifie le ième algorithme de Cesar à utiliser
- repeter des le debut chaque d lettres du plaintext

Key \ Plaintext	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
a	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
b	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
c	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
d	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
e	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
f	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
g	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
h	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
i	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
j	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
k	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
l	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
m	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
n	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
o	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
p	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
r	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
s	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
t	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
u	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
v	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
w	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
x	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X

Vigenère: Exemple

- écrire le plaintext
- écrire la clé et la répéter sur la longueur du plaintext
- utiliser chaque lettre de la clé comme clé de Cesar
- chiffrer chaque lettre indépendamment des autres
- ex : clé = deceptive

```
key:            deceptivedeceptivedeceptive
plaintext:      wearediscoveredsaveyourself
ciphertext:     ZICVTWQNGRZGVTWAVZHCQYGLMGJ
```

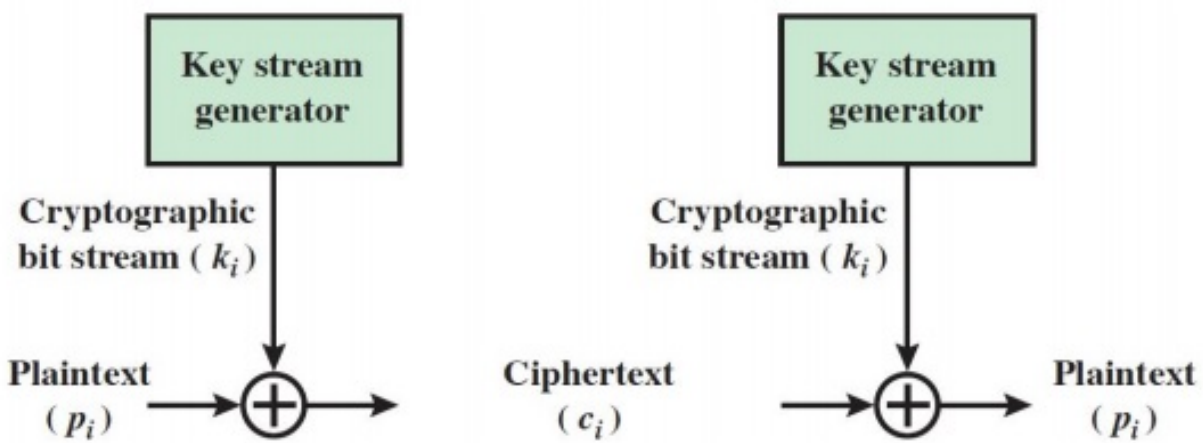
Vigenère: AutoKey Cipher

- voulant une clé aussi longue que le message
- vigenère propose l'autokey
- clé est préfixé au message pour générer une nouvelle clé
- connaissant la clé basique, on peut déchiffrer les premières lettres
- peut etre cassé par analyse fréquentielle
- ex : clé : deceptive

```
key:          deceptivewearediscoveredsav
plaintext:    wearediscoveredsaveyourself
ciphertext:   ZICVTWQNGKZEIIGASXSTSLVVWLA
```

Vernam cipher

- utilise une clé aussi longue que le plaintext
- inventé par un ingénieur AT&T Gilbert Vernam en 1918



Amélioration Vernam cipher: One Time Pad

- Amélioration de Vernam proposé par l'officier de l'armée, Joseph Mauborgne
- Utiliser une clé aléatoire qui est aussi longue que le message de sorte que la clé n'a pas besoin d'être répétée
- La Clé est utilisée pour chiffrer et déchiffrer un seul message, puis elle est jeté
- Chaque nouveau message nécessite une nouvelle clé de la même longueur que le nouveau message
- Ce cryptosystème est incassable
- problèmes dans la production et la distribution sécurisée de la clé
- Non pratique : reste utilisé ds les communications top-secrets et très coûteuses (teleph rouge entre Moscow et Washington)

Principe

- **Transposition= permutation**
- Chiffrer le message en réarrangeant l'ordre des lettres du plaintext
- le plaintext et le ciphertext ont même occurence (fréquence) des lettres

Fail hence cipher

- La transposition la plus simple
- plaintext est écrit en séquences de diagonales
- on le lit ligne par ligne
- pour chiffrer le message "meet me after the toga party" avec "Rail hence" de profondeur(nb de lignes) 2 :

```
m e m a t r h t g p r y
  e t e f e t e o a a t
```

- ciphertext est : MEMATRHTGPRYETEFETEOAAT

Raw Transposition Cipher

- Transposition plus complexe
- écrire le plaintext sous forme de rectangle, ligne par ligne
- ciphertext : lire le message colonne par colonne, mais permuter l'ordre des colonnes
- l'ordre de la lecture des colonnes est donc la clé

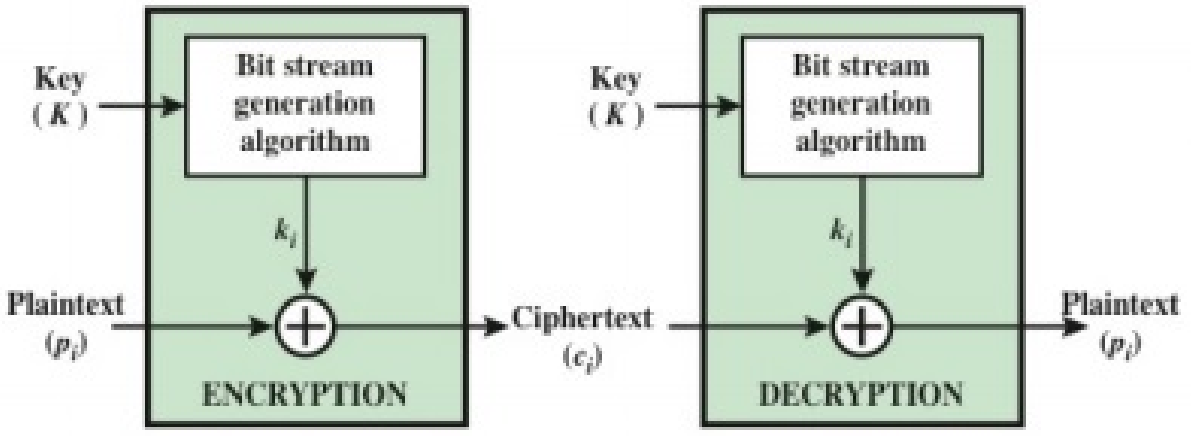
```
Key:            4 3 1 2 5 6 7
Plaintext:      a t t a c k p
                o s t p o n e
                d u n t i l t
                w o a m x y z
Ciphertext:     TTNAAPTMTSUOAODWCOIXKNLYPETZ
```

Product ciphers

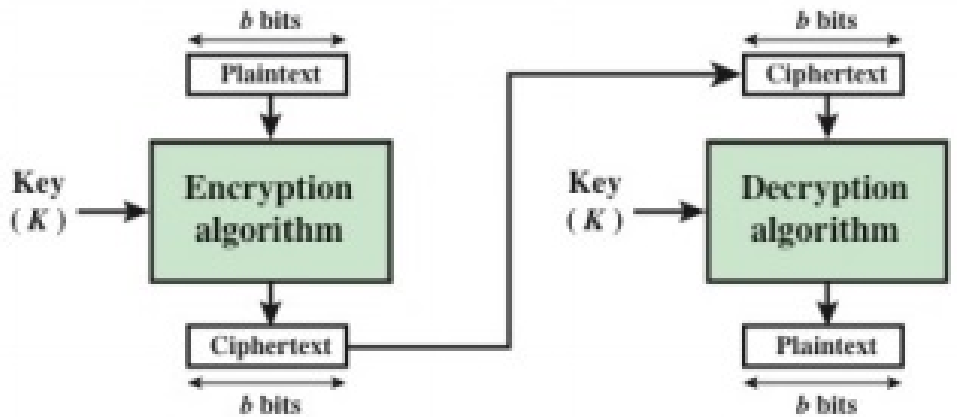
- Les algorithmes de substitutions ou transposition ne sont pas sécurisés à cause de l'analyse fréquentielle
- donc envisager d'utiliser plusieurs alg à la suite pour rendre la cryptanalyse plus difficile.
- exemple répéter la permutation du texte précédent avec la même clé (ou même avec une autre clé) :

```
Key:        4 3 1 2 5 6 7
Input:      t t n a a p t
            m t s u o a o
            d w c o i x k
            n l y p e t z
Output:     NSCYAUOPTTWLTMDNAOIEPAXTTOKZ
```

Cryptographie moderne
Stream cipher vs Block cipher



(a) Stream Cipher Using Algorithmic Bit Stream Generator



(b) Block Cipher

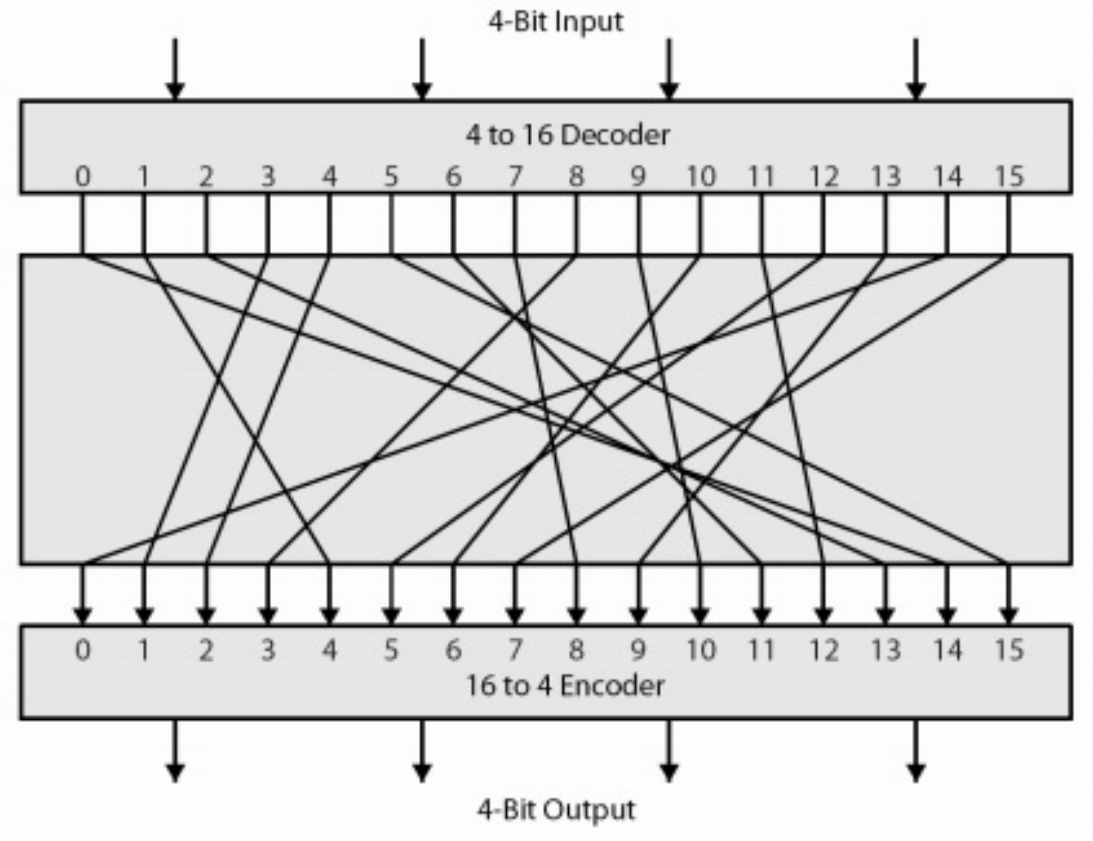
fonctions réversibles et irréversibles

- Un algorithme à chiffrement par bloc prend n bits du plaintext et le transforme en n bits de ciphertext
- il y a 2^n combinaisons possibles de plaintext
- le Cryptage doit être reversible
- chaque bloc du plaintext produit un bloc du ciphertext diffèrent (bijectivité)
- il ya 2^n transformations possibles

Reversible Mapping	
Plaintext	Ciphertext
00	11
01	10
10	00
11	01

Irreversible Mapping	
Plaintext	Ciphertext
00	11
01	10
10	01
11	01

modèle d'un block cipher



les tables du bloc cipher exemple

Plaintext	Ciphertext
0000	1110
0001	0100
0010	1101
0011	0001
0100	0010
0101	1111
0110	1011
0111	1000
1000	0011
1001	1010
1010	0110
1011	1100
1100	0101
1101	1001
1110	0000
1111	0111

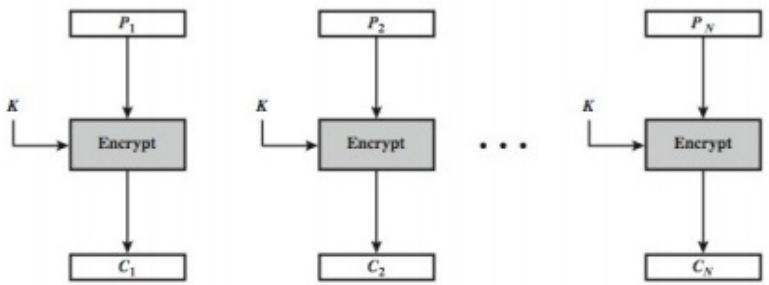
Ciphertext	Plaintext
0000	1110
0001	0011
0010	0100
0011	1000
0100	0001
0101	1100
0110	1010
0111	1111
1000	0111
1001	1101
1010	1001
1011	0110
1100	1011
1101	0010
1110	0000
1111	0101

Modes de cryptage par bloc

- NIST SP 800-38A definiti 5 modes de cryptages
  - **ECB** : Electronic codebook Book Mode
  - **CBC** : cipher block chaining Mode
  - **CFB** : cipher FeedBack Mode
  - **OFB** : Output FeedBack Mode
  - **CTR** : Counter Mode
- il y a ceux qui sont orientés bloc et ceux qui sont orienté flux
- Ceci est pour couvrir une large variété d'application ds la vie réelle
- ces modes peuvent être appliqués sur n'importe quel algorithme de bloc

ECB

- Le plaintext est divisé en blocs qui seront cryptés
- chaque block constitue une valeur qui sera substitué par cryptage comme un dictionnaire, d'où le nom (dictionnaire=codebook)
- Chaque bloc est crypté indépendamment des autres blocs : $C_i = E_K(P_i)$
- Application : transmission sécurisée de messages courts



(a) Encryption

Avantages et limitations de ECB

- Les répétitions dans le plaintext sont montrés aussi dans le ciphertext (peu de confusion)
- Non efficace pour les images : trop de redondance, trop de répétitions donc image peut rester visible après cryptage
- La faiblesse est dans l'indépendance dans le cryptage des différents blocs
- Utilisation principale est le cryptage de plaintext très court

Image originale

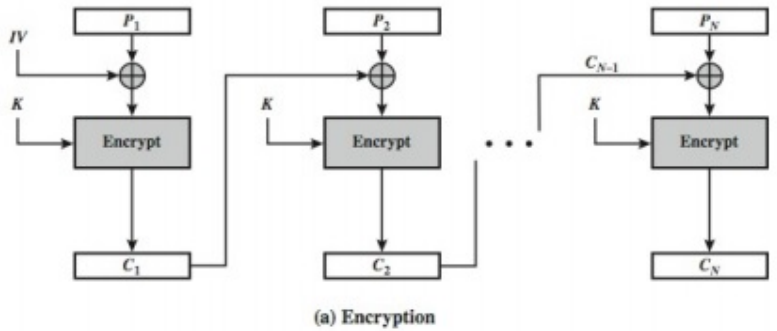
Image cryptée (AES) en mode ECB

CBC

- Le plaintext est divisé en blocs, ces blocs seront liés durant le cryptage
- chaque bloc du ciphertext est lié avec le bloc du plaintext correspondant et les bloc ciphertext précédents
- utilise un vecteur d'initialisation pour commencer le cryptage :

$$C_i = E_K(P_i XOR C_{i-1}) C_{-1} = IV$$

- Application : Le cryptage de données en vrac (de grande redondance) ; Authentification (CMAC)



(a) Encryption

Avantages et limitations de CBC

- chaque bloc du ciphertext dépend de tous les blocs qui le précèdent
- n'importe quel changement affecte tous les blocs du ciphertext qui le suivent
- CBC a besoin d'un IV pour l'initialisation :
  - l'IV doit être connu de l'émetteur et récepteur
  - S'il est transmis en clair, un adversaire peut changer les bits du premier bloc et changer IV pour compenser ce changement.
  - Donc IV doit être soit fixe
  - soit envoyé crypté en mode ECB avant de traiter le plaintext

Les modes de cryptage en bloc orientés flux

- Les modes de bloc chiffre tout le bloc
- dans certaines application, on pourra avoir besoin d'opérer sur des tailles plus petites
- application dans le cryptage du flux multimédia (temps réel)
- convertir les alg de bloc en algorithme de flux
  - **CFB**
  - **OFB**
  - **CTR**
- séquences pseudo-aléatoires

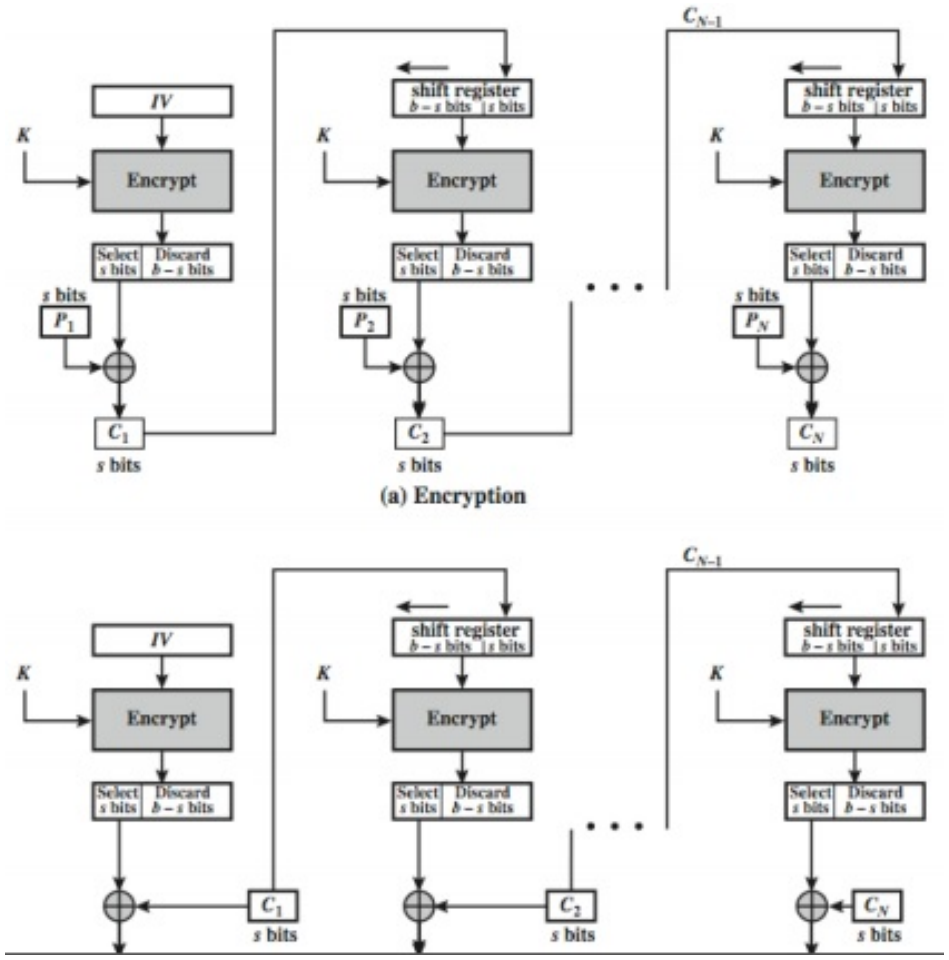
CFB

- Le message est traité comme un flux de bits
- le message est ajouté à la sortie du l'alg de bloc
- le résultat est retourné (feed-back) à l'étage précédent (d'où le nom)
- le standard permet plusieurs tailles de blocs (1 ; 8 ; 64 ; 128 ; etc) pour être feed-back
- notés CFB-1, CFB-8, CFB-64, CFB-128
- le cryptage est comme suit :

$$C_i = P_i XOR E_K(C_{i-1}) C_{-1} = IV$$

- Applications : cryptage du flux (temps réel), Authetification

CFB


IV
K
Encrypt
Select s bits
Discard b − s bits
s bits
P1
C1
s bits
shift register
b − s bits | s bits
P2
C2
PN
CN
CN−1
(a) Encryption

Avantages et limitations de CFB

- CFB est approprié si les données arrivent en bits ou octets
- approprié pour le mode en flux
- Noter que dans le cryptage et le décryptage, les deux opèrent avec le bloc de chiffrement E_K
- l'erreur (s'il y en a) peut se propager dans plusieurs blocs après le bloc erroné

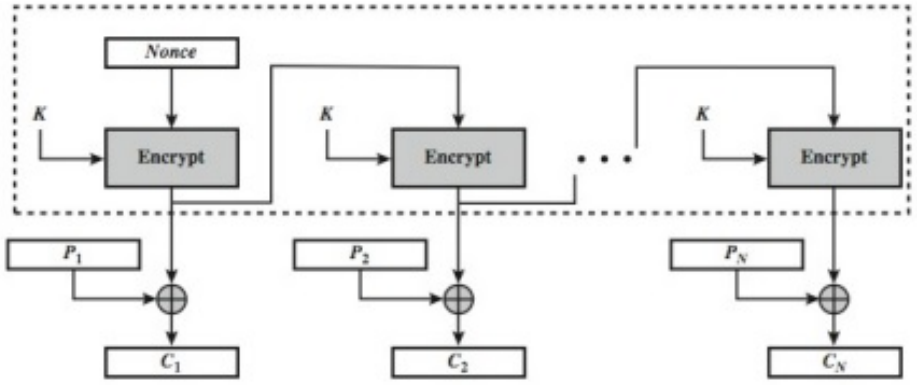
OFB

- Le message est traité comme flux de bits
- la sortie du cryptage est ajouté au message
- la sortie est ensuite retourné (Output feed-back) à l'entrée de l'étage suivant (d'où le nom)
- le feedback est indépendant du message (plaintext)
- il peut être calculé auparavant

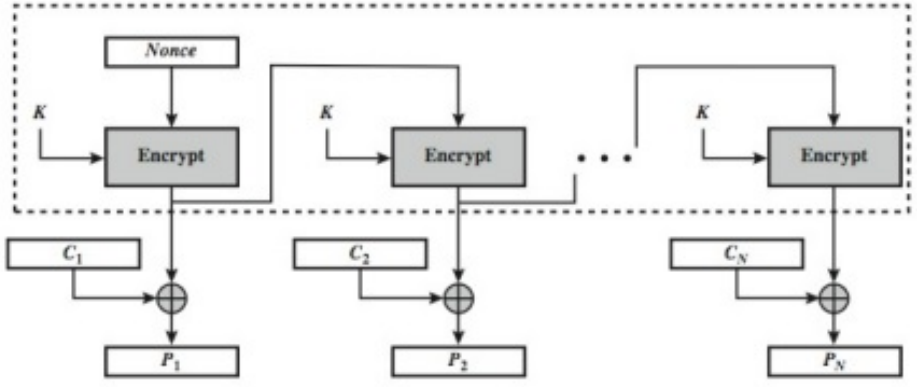
$$O_i = E_K(O_{i-1})C_i = P_i XORO_i O_{-1} = IV$$

- Utilisation : Cryptage de flux dans un canal bruité

OFB



(a) Encryption



(b) Decryption

Avantages et limitations de OFB

- OFB a besoin d'un IV qui doit être unique pour chaque utilisation
- si l'IV est réutilisé, l'adversaire peut retrouver les sorties
- Les erreurs ne se propagent pas
- émetteur et récepteur doivent être en synchronisation

CTR

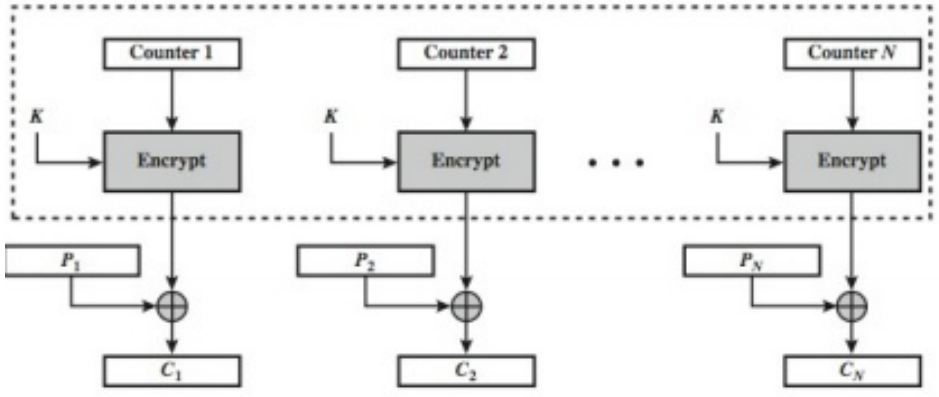
- un nouveau mode similaire à OFB mais chiffre un compteur au lieu de la sortie
- doit avoir une clé différente et une valeur de compteur différente pour chaque message

$$O_i = E_K(O_i)C_i = P_i XORO_i$$

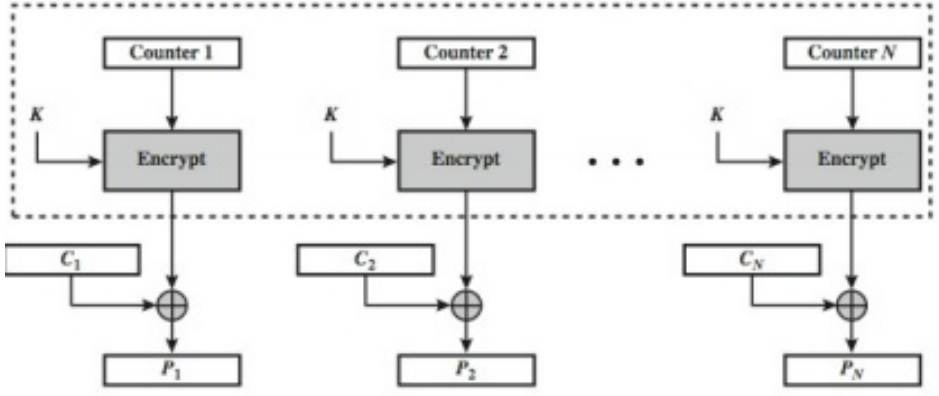
- Utilisation : Cryptage dans les réseaux haut débit

Cryptographie moderne
Block Cipher

CTR



(a) Encryption



(b) Decryption

Shannon et les alg de substitution-permutation

- Shannon a introduit l'idée des réseaux de substitution-permutation (S-P) en 1949
- c'est la base de tout les alg de cryptage moderne
- les réseaux S-P sont basés sur deux critère :
  - **substitution (S-box)**
  - **Permutation(P-box)**
- Ceci fourni les critères de confusion et de diffusion du plaintext et de la clé sur le ciphertext

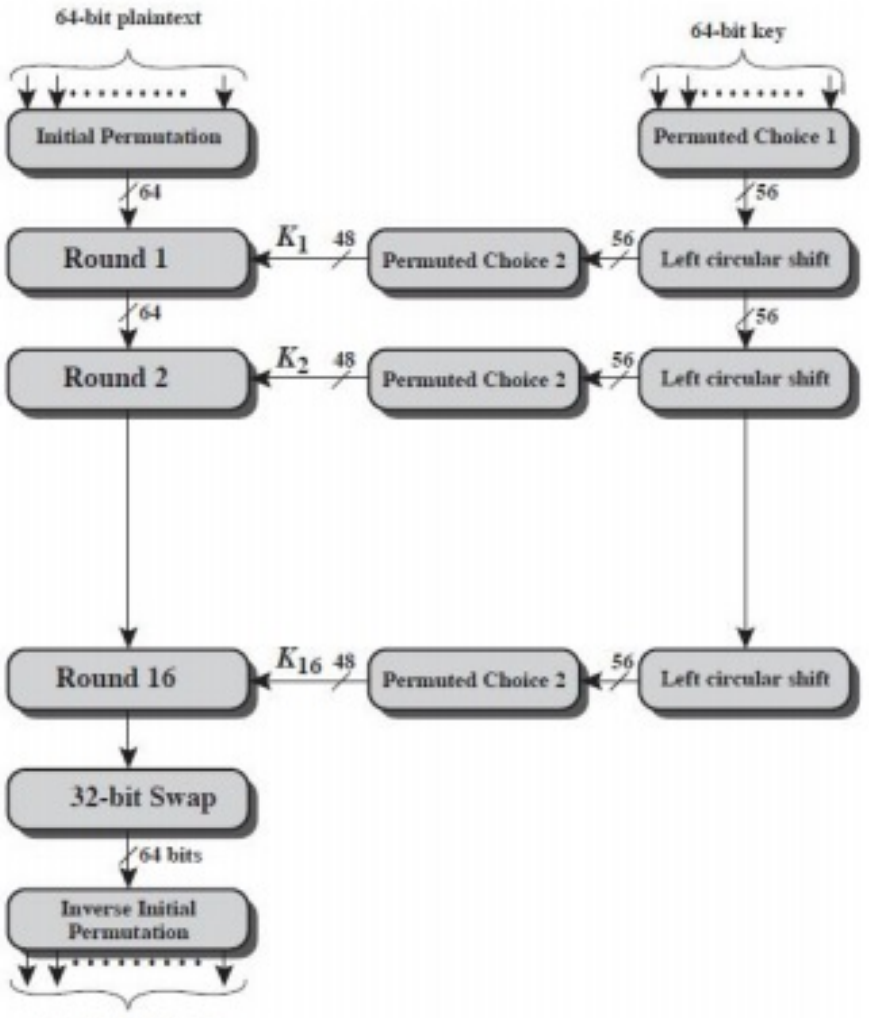
Confusion et diffusion

- Deux termes introduits par Shanon qui constituent les critères de base d'un algorithme de cryptage
- Son but était de concevoir des cryptosystèmes qui résistent l'analyse statistique
- **confusion** : Rend la relation entre le ciphertext et la clé aussi complexe que possible (apparence aléatoire)
- **diffusion** : Chaque bit du plaintext affecte tous les bits du ciphertext (avalanche)

DES

- Data Encryption standard (DES) est le standard de cryptage recommandé par NIST (National Institute of Standards and Technologies) en 1977.
- l'alg de cryptage le plus utilisé jusqu'à 2001 (l'arrivée de AES par NIST aussi)
- L'alg de DES est appelé DEA (Data Encryption Algorithm)
- Le plaintext est chiffré en 64-bit blocs en utilisant une clé de taille 56 bit
- l'alg transforme un bloc de 64-bit du plaintext à un bloc de 64-bit du ciphertext
- Les mêmes étapes, avec la même clé, conduisent au décryptage

DES


64-bit plaintext
64-bit key
Initial Permutation
Permuted Choice 1
64
56
Round 1
K_1
48
Permuted Choice 2
56
Left circular shift
64
56
Round 2
K_2
48
Permuted Choice 2
56
Left circular shift
Round 16
K_{16}
48
Permuted Choice 2
56
Left circular shift
32-bit Swap
64 bits
Inverse Initial Permutation

Initial permutation (IP) de DES

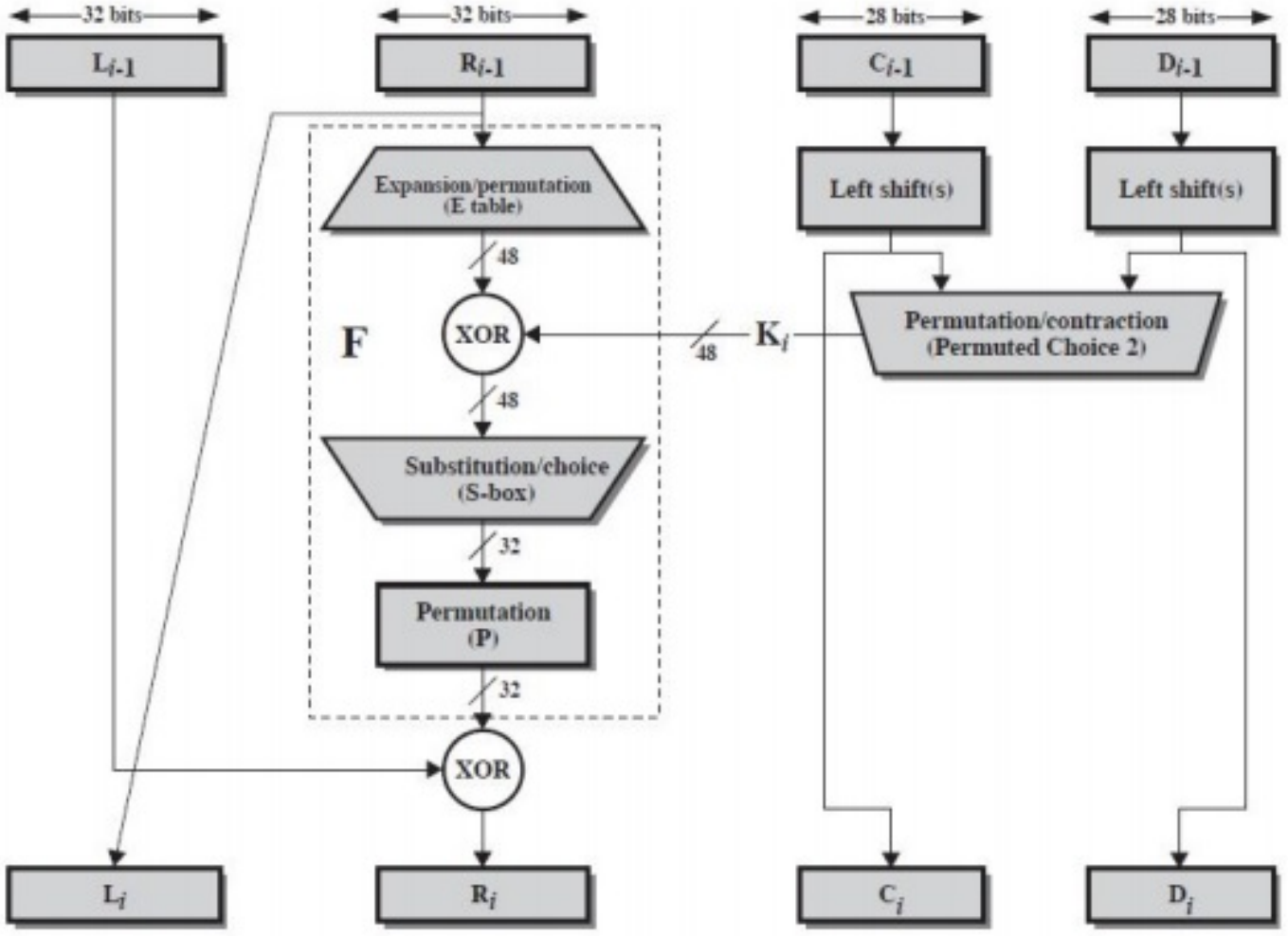
(a) Initial Permutation (IP)

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

(b) Inverse Initial Permutation (IP^{-1})

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Structure d'une ronde DES



Structure d'une ronde DES

- Deux moitiés L et R de taille 32-bits chacune
- Structure de Feistel est comme suit :
- F prend la moitié R de 32-bit et la clé intermédiaire de 48-bit et fait comme suit :

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$$

- Expansion de R à 48-bits en utilisant la permutation E
- L'ajouter à la clé intermédiaire par XOR
- La faire passer à travers 8 S-box pour avoir le résultat de 32-bits
- Finalement la permuter en utilisant une permutation P

Les fonction de permutation E et P

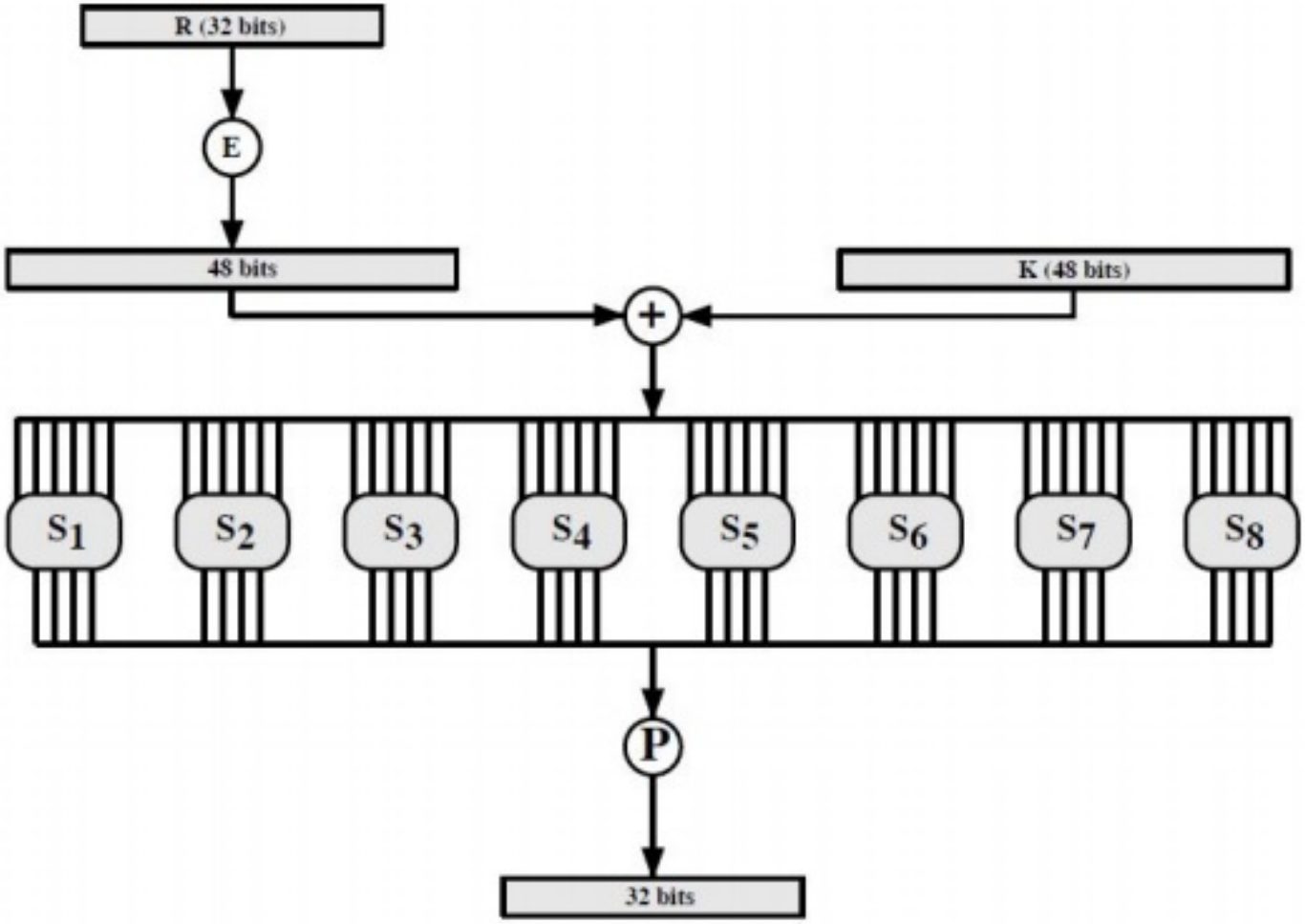
(c) Expansion Permutation (E)

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

(d) Permutation Function (P)

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Structure d'une ronde DES : F(R,K)



Les 8 S-box

- chaque S-box transforme 6-bits à 4-bits
- Pour chaque entrée de chaque S-box :
  - les bits 1 et 6 (bits extérieurs) sélectionne une ligne parmi 4.
  - les bits 2-5 (bits intérieurs) sont substitués par la sortie correspondante ds la ligne choisie
  - le résultat est 8 lots de 4-bit : ça fait 32-bits en tout
- la sélection de la ligne dépend du plaintext et de la clé

Les 8 S-box : (1-4)

S_1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

Cryptographie moderne
DES

Les 8 S-box : (5-8)

S_5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Key schedule

- Key schedule : préparation des clés intermédiaires (des 16 rondes) à partir de la clé originale de 56-bits
- Permutation initiale de la clé (PC1) qui sélectionne 56-bits (parmi 64) en 2 moitiés de 28-bits
- 16 stages qui consistent à :
  - "Rotation circulaire à gauche" de chaque moitié de 1 ou 2 bits en fonction de la fonction de rotation K
  - sélectionner 24 bits de chaque moitié et la permuter par (PC2) pour être l'entrée de la fonction F.

Key schedule

(b) Permuted Choice One (PC-1)

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

(c) Permuted Choice Two (PC-2)

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

(d) Schedule of Left Shifts

Round Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits Rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

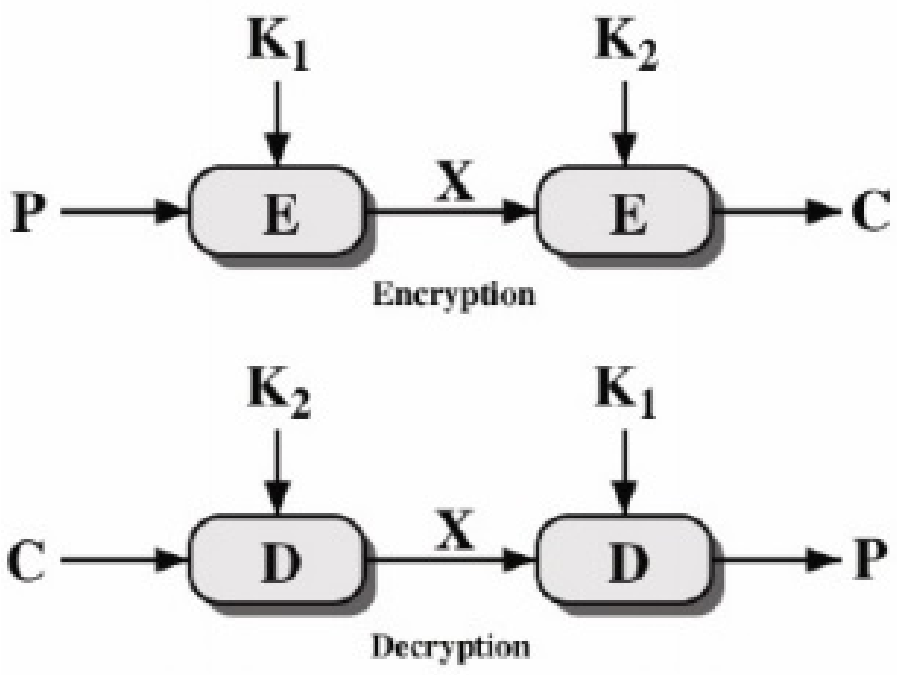
Analyse de sécurité

- taille de l'espace clé $2^{56} = 7.2 \times 10^{16}$
- une machine à 10^9 déchiffrement/s peut le casser en 1.125 année
- une machine à 10^{13} déchiffrement/s peut le casser en 1 heure
- l'AES-128 avec la même vitesse, la machine reste 5.3×10^{17} années
- plusieurs attaques sur DES :
  - differential cryptanalysis
  - linear cryptanalysis
  - related key attack
- Le besoin de trouver une alternative de DES devient nécessaire

Cryptage multiple

- DES est devenu vulnérable à la brute force attack
- Alternative : crypter plusieurs fois avec des clés différentes
- Options :
  - Double DES : n'est pas très performant
  - Triple DES (3DES) avec deux clés : brute force 2^{112}
  - Triple DES avec trois clés : brute force 2^{168}

Cryptographie moderne
DES



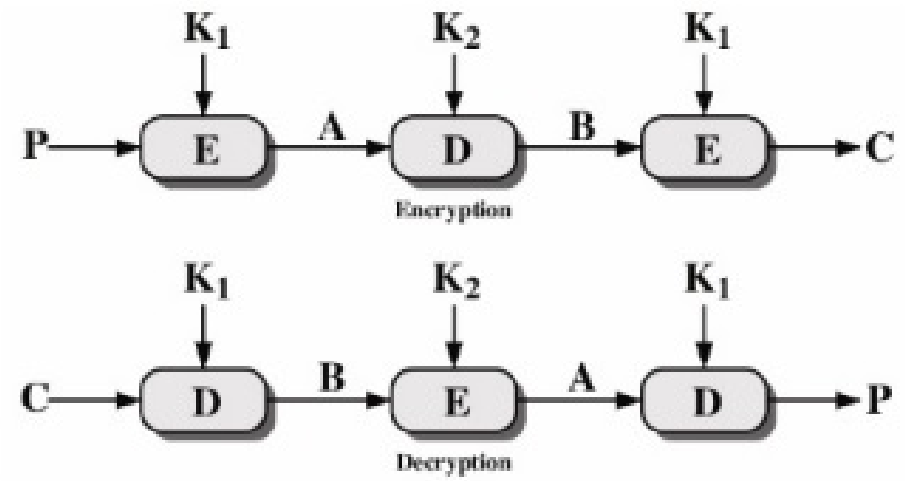
Double encryption

- clé de taille $2 \times 56 = 112$
- espace clé de 2^{112}
- il faut donc essayer en moyenne 2^{111} pour le casser en brute force
- attaque plus intelligente : Meet-in-the-middle attack

Meet-in-the-middle attack

- Double cryptage DES : $C = E(K_2, E(K_1, P))$
- soit $X = E(K_1, P) = D(K_2, C)$
- supposons que l'adversaire connaît 2 paires P/C : (P_a, C_a) et (P_b, C_b)
  - chiffrer Pa en utilisant toutes les possibilités 2^{56} de la clé K_1 pour avoir les possibilités de X
  - Enregistrer les valeurs possibles de X ds un tableau avec leurs clés correspondantes K_1
  - Déchiffrer C_a en utilisant toutes les possibilités 2^{56} de la clé K_2
  - Pour chaque résultat du décryptage, vérifier avec les valeurs du tableau
  - S'il y a correspondance, Prenez les valeurs correspondantes de K_1 et K_2. Et vérifier si $C_b = E(K_2, E(K_1, P_b))$, alors accepter les clés.
- Avec deux paires de P/C, la probabilité de succès est 1
- Cette attaque est de complexité 2×2^{56} qui est très inférieur à complexité brute force attack 2^{112}

Cryptographie moderne
DES



Cryptage triple

- 2 clés : 112 bits
- 3 clés : 168 bits
- Pourquoi E-D-E ? Pour être compatible avec DES simple :

$$C = E(K_1, D(K_1, E(K_1, P)))$$

- 3DES a été adopté par plusieurs application internet : PGP, S/MIME

Cryptographie moderne
AES

Principe

- Advanced encryption standard est le nouveau alg de cryptage adopté par NIST crée en 2001 (Rijandael)
  - Taille block : 128-bit (possibilité d'autres tailles)
  - taille clé : 128, 192, 256 bits
  - rondes : 10, 12, 14 dépend de la clé
  - XOR avec clés intermédiaires, Substitutions avec des S-box, mixage avec l'arithmétique du corps de Galois
- Largement utilisé dans les communications sécurisés en réseaux
- Considéré comme sécurisée jusqu'à ce moment.

Merci pour votre attention!