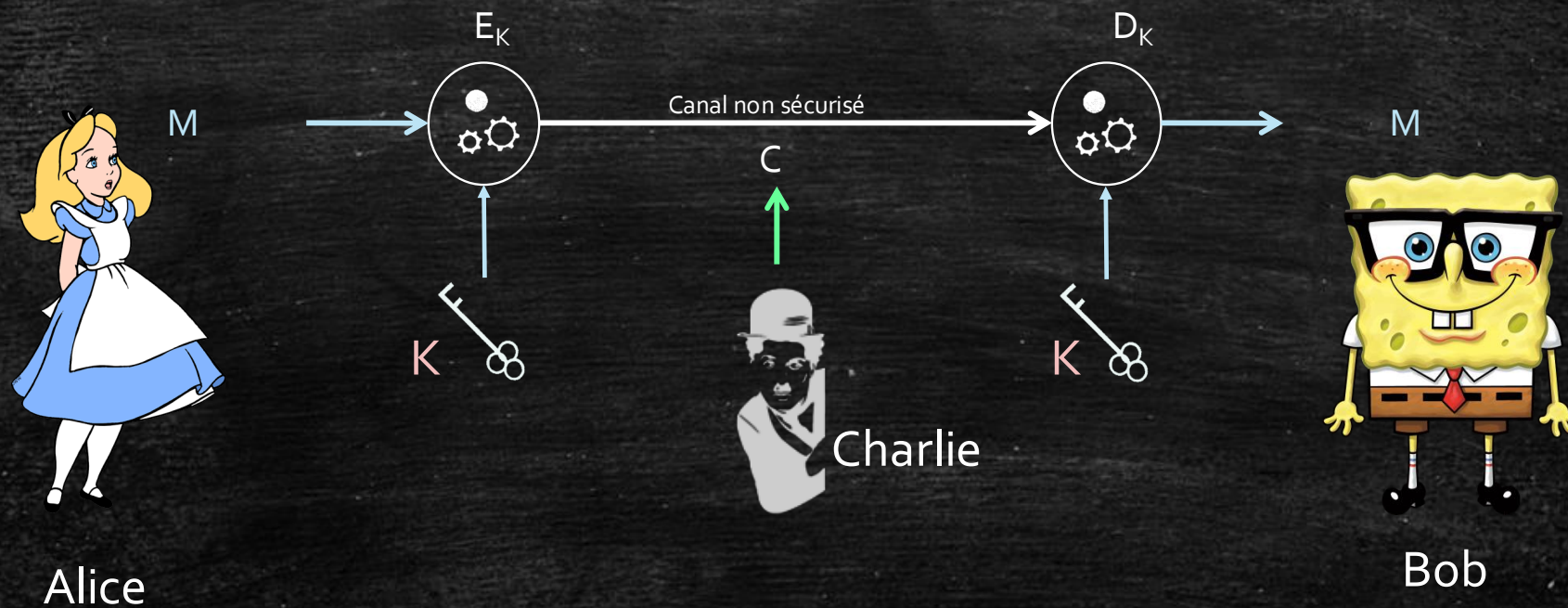
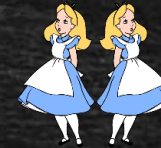


Introduction à la cryptographie

Louiza Khati

Cours 2

Rappels : Chiffrement symétrique



Chiffrement $C = E_K(M)$ et Déchiffrement $M = D_K(C)$
 $M = D_K(E_K(M))$

Confidentialité : niveau attaquant

- Intuitivement :
 - Un attaquant ne doit pas être capable de retrouver la clé secrète


Confidentialité : niveau attaquant

- Intuitivement :
 - Un attaquant ne doit pas être capable de retrouver la clé secrète
 - Un attaquant ne doit pas être capable de déchiffrer sans la clé secrète


Confidentialité : niveau attaquant

▪ Intuitivement :

- Un attaquant ne doit pas être capable de retrouver la clé secrète
- Un attaquant ne doit pas être capable de déchiffrer sans la clé secrète
- Un attaquant ne doit apprendre aucune information sur le message clair à la vue de son chiffré



Attaque de plus
en plus simple




Attaquant de plus
en plus fort!

Confidentialité : niveau attaquant


- Intuitivement :

- Un attaquant ne doit pas être capable de retrouver la clé secrète
- Un attaquant ne doit pas être capable de déchiffrer sans la clé secrète
- Un attaquant ne doit apprendre aucune information sur le message clair à la vue de son chiffré



Attaque de plus
en plus simple

Sécurité maximale



Attaquant de plus
en plus fort!

Modèles d'adversaire

- Moyens de l'attaquant

- **Chiffrés seuls (COA)** : l'adversaire ne connaît que des messages chiffrés (exemple : interception)
- **Clairs connus (KPA)** : l'adversaire a accès à des couples (M, C) de messages clairs et des chiffrés correspondants
- **Clairs choisis (CPA)** : l'adversaire a accès à des couples (M, C) de messages clairs / chiffrés pour M de son choix (oracle de chiffrement)
- **Chiffrés choisis (CCA)** : l'adversaire a accès à des couples (M, C) de messages clairs / chiffrés pour C de son choix (oracle de déchiffrement)

Modèles d'adversaire

- Moyens de l'attaquant

- **Chiffrés seuls (COA)** : l'adversaire ne connaît que des messages chiffrés (exemple : interception)
- **Clairs connus (KPA)** : l'adversaire a accès à des couples (M, C) de messages clairs et des chiffrés correspondants
- **Clairs choisis (CPA)** : l'adversaire a accès à des couples (M, C) de messages clairs / chiffrés pour M de son choix (oracle de chiffrement)
- **Chiffrés choisis (CCA)** : l'adversaire a accès à des couples (M, C) de messages clairs / chiffrés pour C de son choix (oracle de déchiffrement)

Chiffrement seul

Chiffrement symétrique

Chiffrement par bloc

m blocs

m_1

m_2

m_i

m_m

128 m bits

- Taille de message multiple taille de bloc
- Utilisé avec un mode opératoire
- Chiffrement avec sécurité prouvée
 - Sous l'hypothèse que le chiffrement par bloc est « robuste »
- Exemples :
 - AES-CBC, AES-CTR, Camellia-CBC

Chiffrement à flot

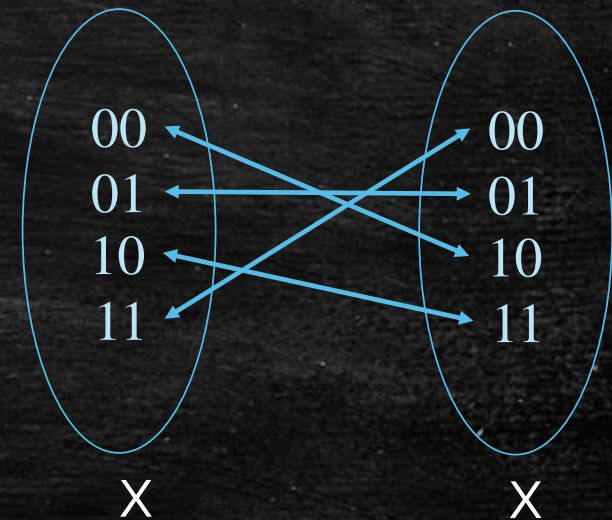
- Taille de message quelconque
 - Au bit/octet près
- Sécurité non prouvée
 - Robustesse donnée avec le temps (cryptanalyse sur plusieurs années)
- Exemples :
 - RC4, chacha

$M[1]$ $M[2]$ $M[i]$ $M[m]$

m bits

Rappels : Permutations

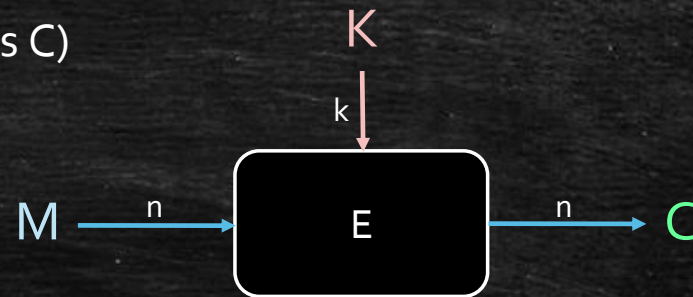
- Permutation $P : X \rightarrow X$ (bijection)
- X = ensemble des mots sur n bits (chiffrement par bloc)
 - X ensemble de cardinal 2^n
 - Exemple : permutation sur 2 bits ($n=2$)



(Primitive de)
chiffrement par bloc

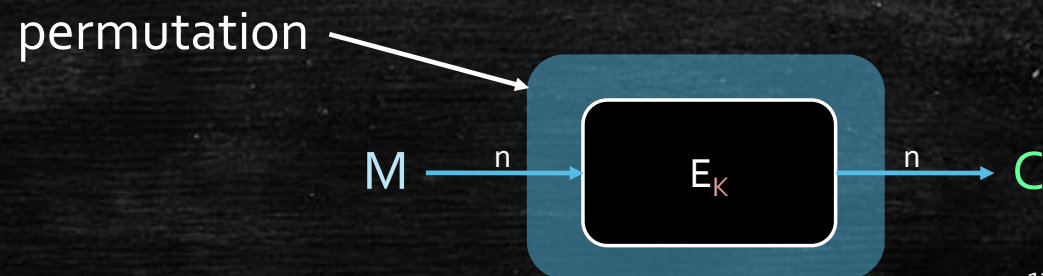
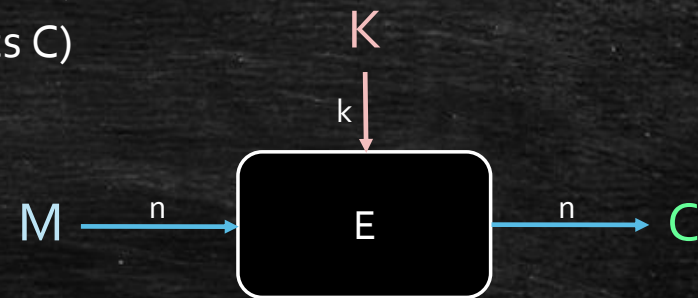
Brique de base : chiffrement par bloc

- A clé fixée, un chiffrement par bloc est une permutation
 - Ensemble d'entrée : mots sur n bits (les blocs M)
 - Ensemble de sortie : mots sur n bits (les blocs C)
 - n est la taille de bloc de ce chiffrement
 - La taille de bloc de l'AES est 128 bits
 - La taille de bloc du 3DES est de 64 bits
 - La taille de la clé \neq taille de bloc
 - AES : 128, 192, 256 bits
 - 3DES : 112 bits, 168 bits



Brique de base : chiffrement par bloc

- A clé fixée, un chiffrement par bloc est une permutation
 - Ensemble d'entrée : mots sur n bits (les blocs M)
 - Ensemble de sortie : mots sur n bits (les blocs C)
 - n est la taille de bloc de ce chiffrement
 - La taille de bloc de l'AES est 128 bits
 - La taille de bloc du 3DES est de 64 bits
 - La taille de la clé \neq taille de bloc
 - AES : 128, 192, 256 bits
 - 3DES : 112 bits, 168 bits



Brique de base : chiffrement par bloc

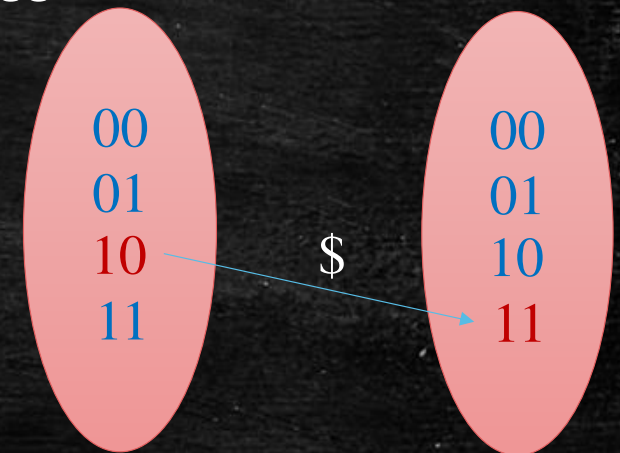
- E_K est « **déterministe** » (connaissant K)
 - Le résultat du chiffrement de M avec E_K est toujours le même C



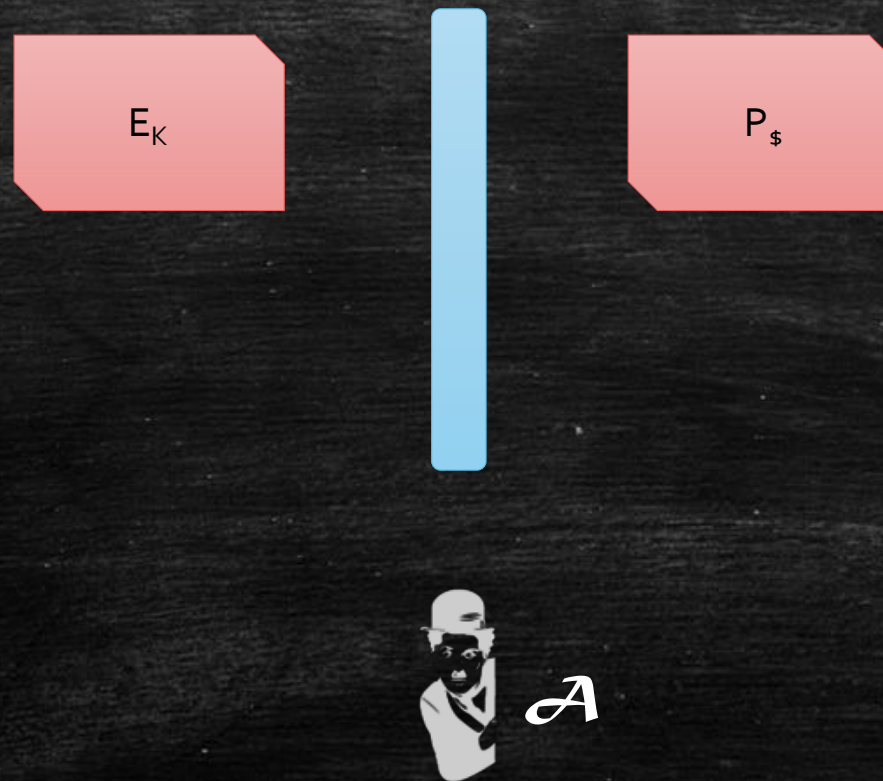
- Un chiffrement par bloc « robuste »
 - Si je vois plusieurs (M, C) sans connaître la clé, je ne vois pas de lien entre eux
 - A chaque fois que je vois un C sans connaître la clé, il me semble aléatoire
 - Est indistinguishable d'une permutation aléatoire

Rappels : Permutation aléatoire

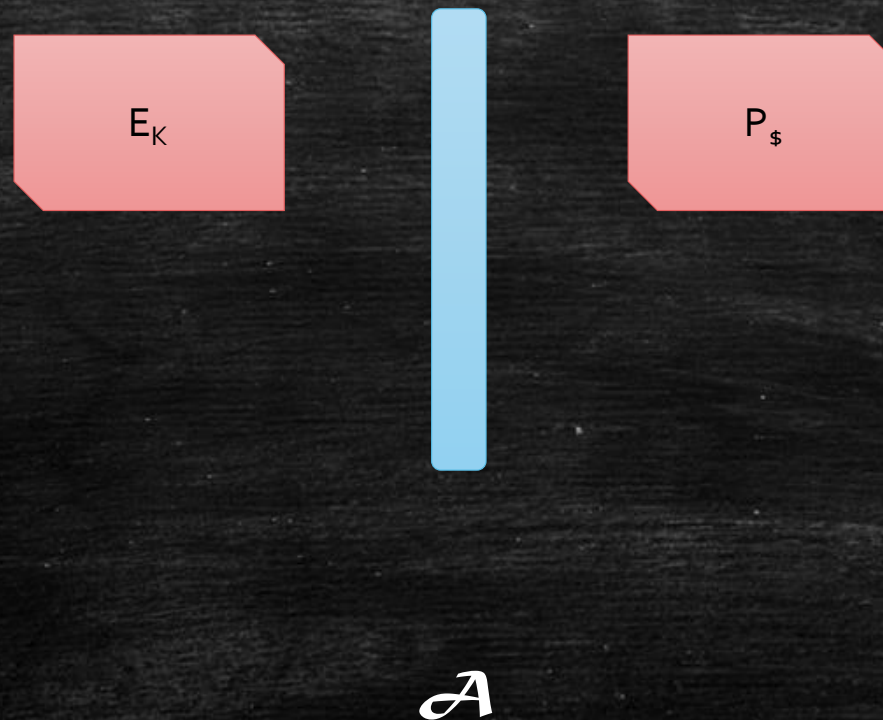
- Permutation aléatoire $P_{\$} : X \rightarrow X$
- X = ensemble des mots sur n bits (chiffrement par bloc)
 - Exemple : permutation sur 2 bits
- Pour chaque entrée, une valeur de sortie est tirée aléatoirement
 - Tirage sans remise (sinon fonction aléatoire)
- Outil théorique



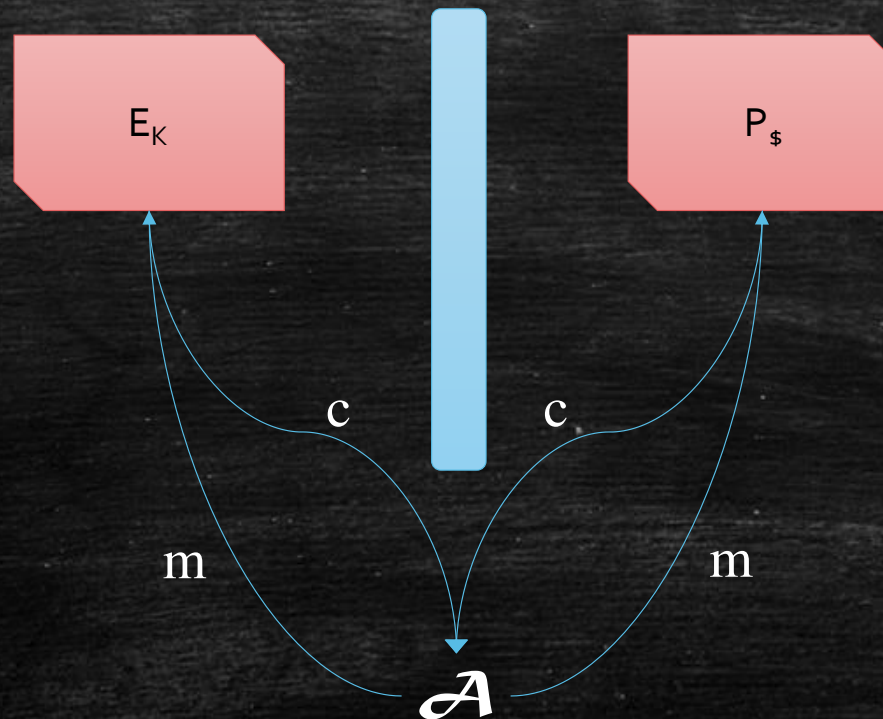
Sécurité d'un chiffrement par bloc



Sécurité d'un chiffrement par bloc

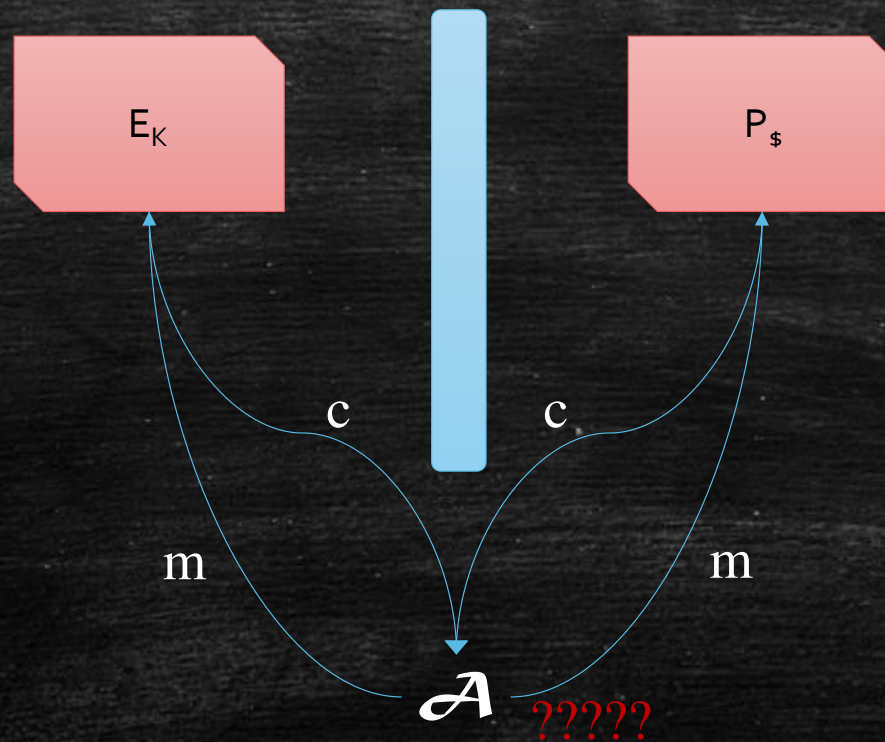


Sécurité d'un chiffrement par bloc



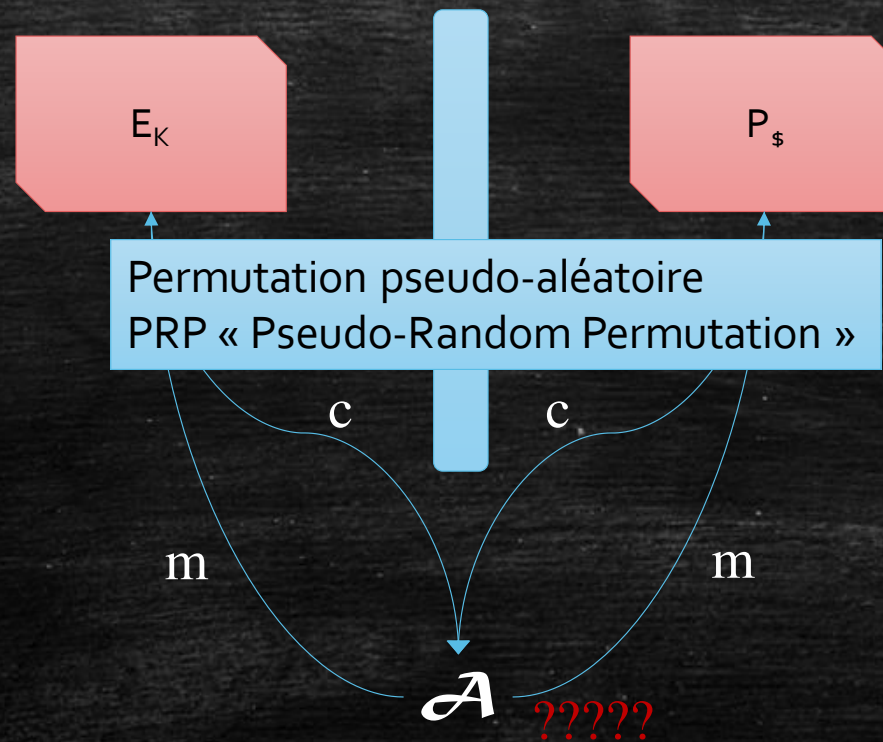
Sans connaître K , \mathcal{A} peut-il distinguer E_K et $P_{\$}$?

Sécurité d'un chiffrement par bloc



Sans connaître K , \mathcal{A} peut-il distinguer E_K et $P_{\$}$?

Sécurité d'un chiffrement par bloc

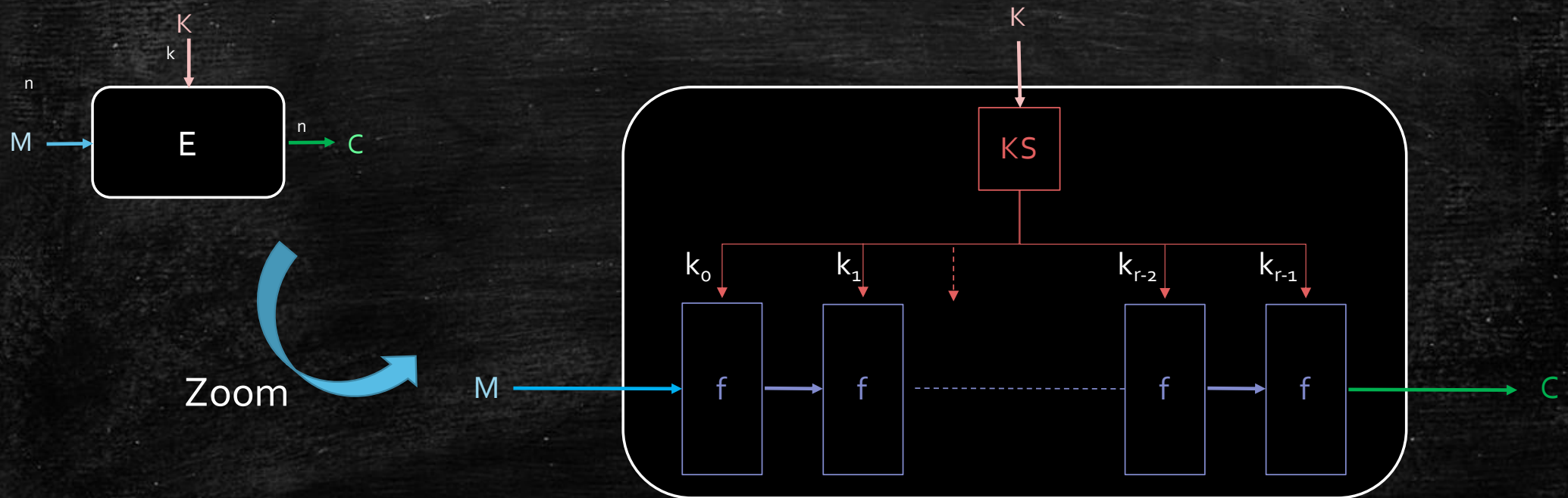


Notion d'indistinguabilité

Sans connaître K , A peut-il distinguer E_K et $P_{\$}$?

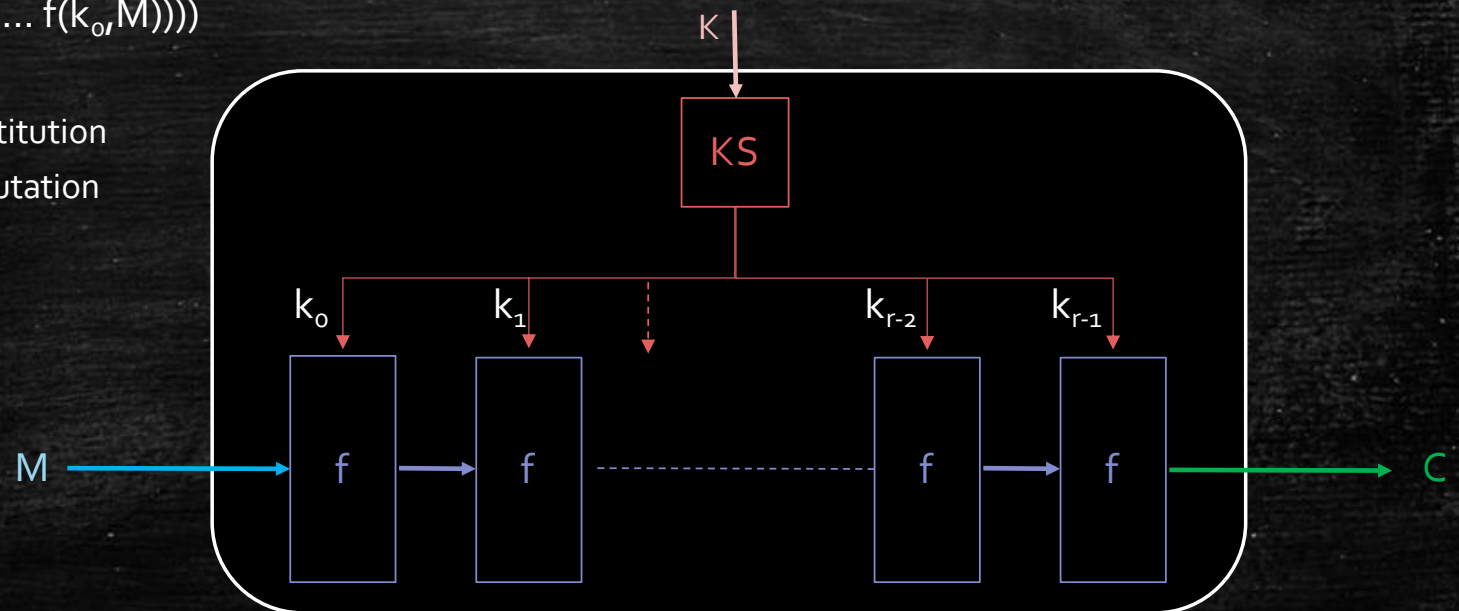
Chiffrement par bloc

- Chiffrement itératif :
 - Pour bien « mélanger » la clé et le message



Chiffrement par bloc

- Chiffrement itératif : (but E_K indistinguishable d'une permutation aléatoire)
 - Application d'une fonction de tour f itérée r fois (« r » pour « round »)
 - « Key Schedule » (KS) : génération de sous clés k_i pour le tour i
 - $C = f(k_{r-1}, f(k_{r-2}, f(\dots f(k_0, M))))$
 - Fonction f :
 - Confusion : substitution
 - Diffusion : permutation



AES- k (k = taille de clé)

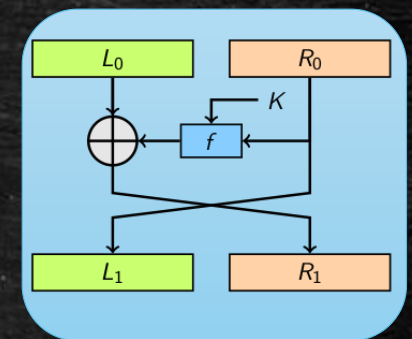
AES-128 : 10 tours

AES-192 : 12 tours

AES-256 : 14 tours

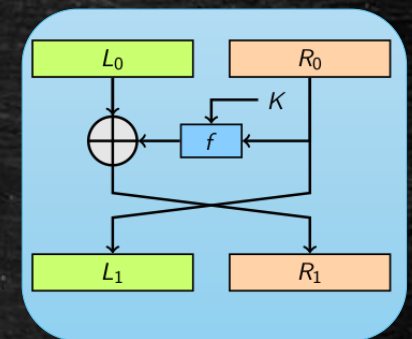
Chiffrement par bloc : DES

- Data Encryption Standard (DES)
 - Lucifer développé à IBM par Horst Feistel dans les années 1970
 - Modifié et standardisé par le NBS (National Bureau of Standards)
- Algorithme de chiffrement symétrique par bloc
 - Clé de 56 bits (2^{56} clés différentes)
 - Bloc de taille 64 bits
 - Schéma de Feistel
- Sécurité
 - En 1977, 256 représente un nombre de chiffrement très grand
 - Aujourd'hui, plusieurs outils cassent une clé en moins de 23 heures!



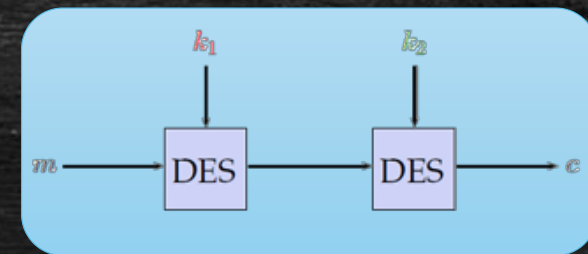
Chiffrement par bloc : DES

- Data Encryption Standard (DES)
 - Lucifer développé à IBM par Horst Feistel dans les années 1970
 - Modifié et standardisé par le NBS (National Bureau of Standards)
- Algorithme de chiffrement symétrique par bloc
 - Clé de 56 bits (2^{56} clés différentes) → Améliorations possibles ?
 - Bloc de taille 64 bits
 - Schéma de Feistel
- Sécurité
 - En 1977, 256 représente un nombre de chiffrement très grand
 - Aujourd'hui, plusieurs outils cassent une clé en moins de 23 heures!



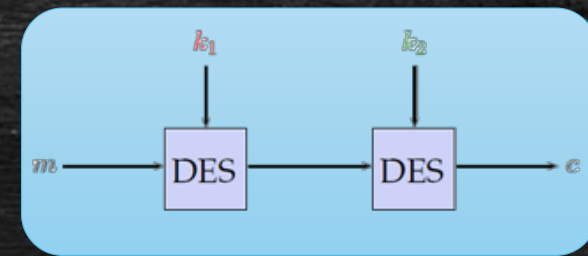
Chiffrement par bloc : DES

- Double DES avec 2 clés différentes :
 - 2 clés donc résistance 2^{112} ?



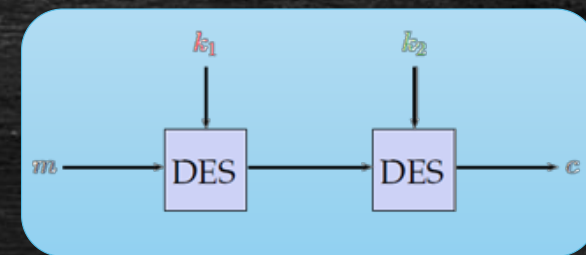
Chiffrement par bloc : DES

- Double DES avec 2 clés différentes :
 - 2 clés donc résistance 2^{112} ?
- Attaque par le milieu
 - Charlie récupère un couple (M,C)



Chiffrement par bloc : DES

- Double DES avec 2 clés différentes :
 - 2 clés donc résistance 2^{112} ?
- Attaque par le milieu
 - Charlie récupère un couple (M,C)

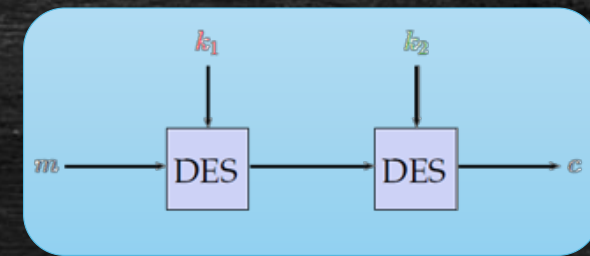


k	DES(k,M)
0..000	V_1
...	...
k_1	v
...	...
1....1	$V_{2^{56}}$

k	DES(k,C)
0..000	V'_1
...	...
k_2	v
...	...
1....1	$V'_{2^{56}}$

Chiffrement par bloc : DES

- Double DES avec 2 clés différentes :
 - 2 clés donc résistance 2^{112} ?
- Attaque par le milieu
 - Charlie récupère un couple (M,C)



k	DES(k,M)
0..000	V_1
...	...
k_1	v
...	...
1....1	$V_{2^{56}}$

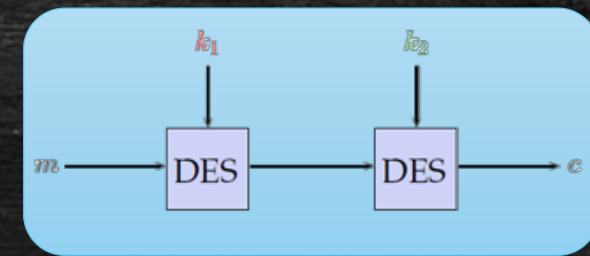
k	DES(k,C)
0..000	V'_1
...	...
k_2	v
...	...
1....1	$V'_{2^{56}}$



Coût total = ?

Chiffrement par bloc : DES

- Double DES avec 2 clés différentes :
 - 2 clés donc résistance 2^{112} ?
- Attaque par le milieu
 - Charlie récupère un couple (M,C)



k	DES(k,M)
0..000	V_1
...	...
k_1	v
...	...
1....1	$V_{2^{56}}$

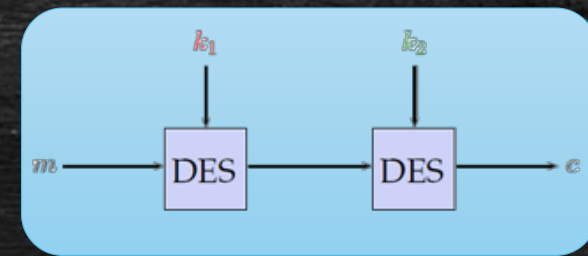
k	DES(k,C)
0..000	V'_1
...	...
k_2	v
...	...
1....1	$V'_{2^{56}}$



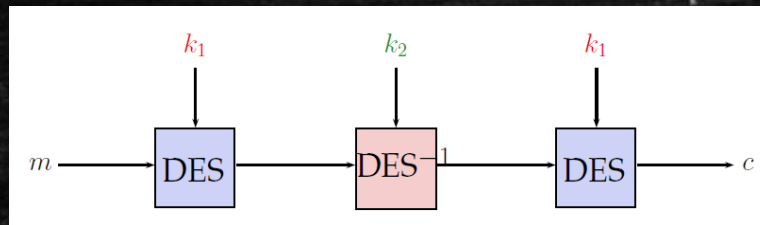
Coût total = $2^{56} + 2^{56} = 2^{57}$ opérations

Chiffrement par bloc : DES

- Double DES avec 2 clés différentes :
 - 2 clés donc résistance 2^{112} ?



- Triple-DES avec 2 clés différentes (1998) :



Résout le problème de la clé trop courte mais pas trop!

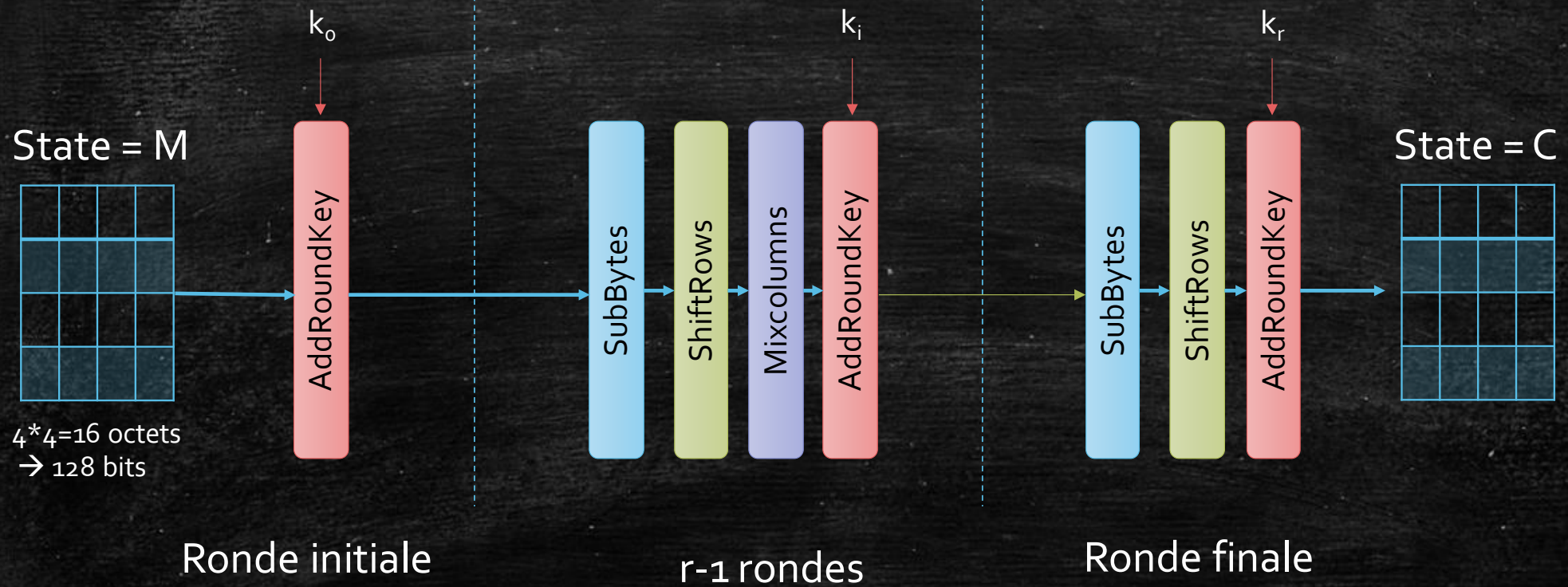
Advanced Encryption Standard

- Advanced Encryption Standard (AES)
 - Compétition NIST (lancée en 1997)
- Algorithme de chiffrement symétrique par bloc
 - Standardisé en 2001
 - Rijndael vainqueur du concours (Rijmen et Daemen) lancé en 1997 par le NIST
 - Clé de 128, 192 ou 256 bits (2^{128} , 2^{192} , 2^{256} clés différentes)
 - Bloc de taille 128 bits
- Algorithme de chiffrement par bloc le plus utilisé



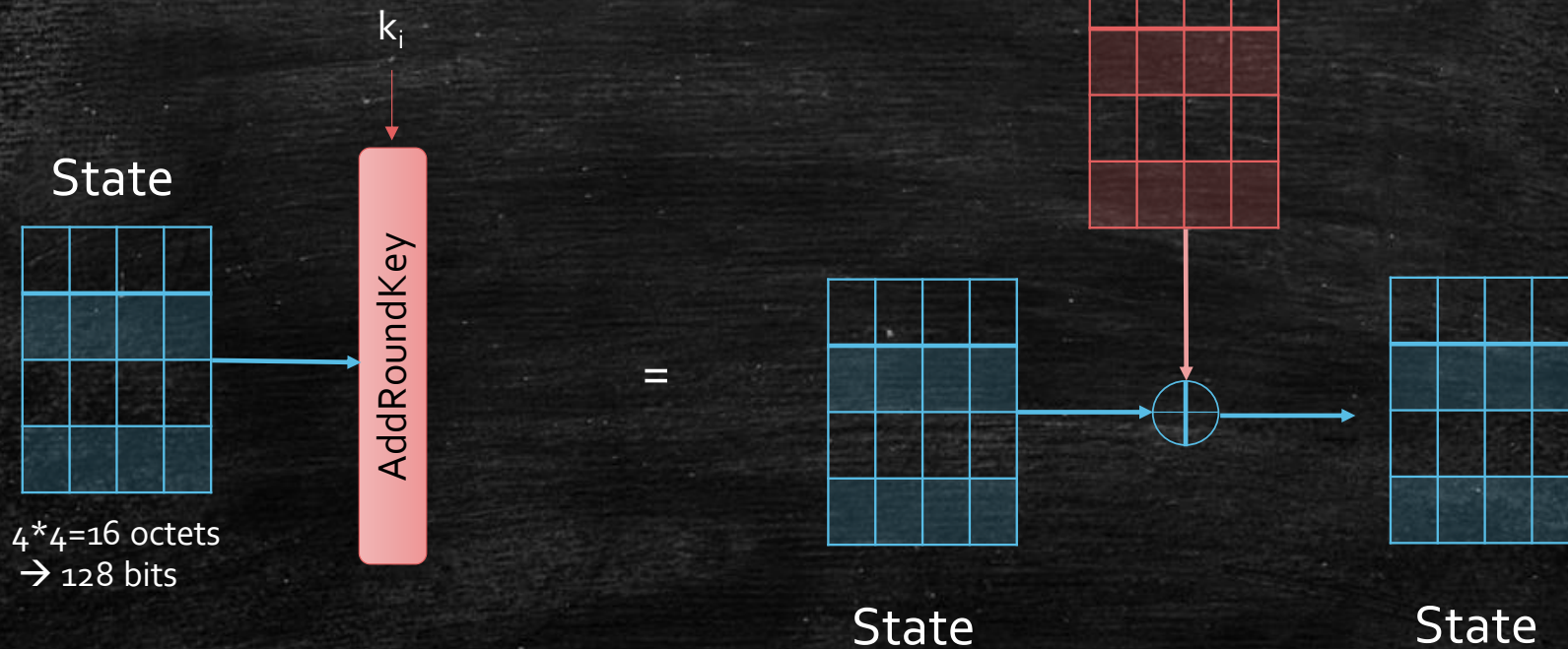
AES : Chiffrement

$$\text{KeyExpansion}(K) = (k_0, \dots, k_r)$$



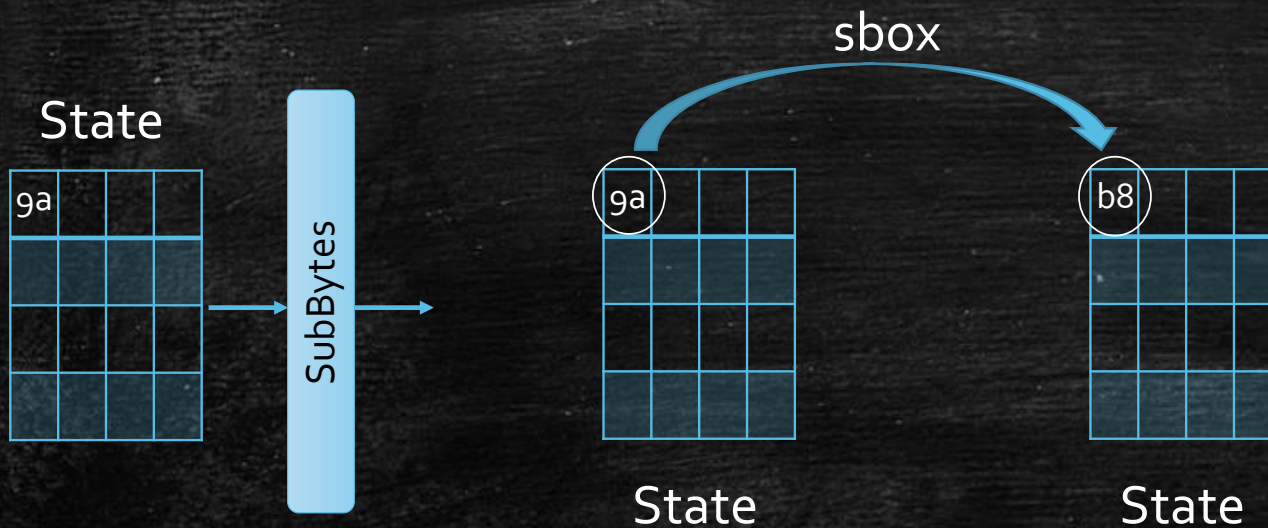
AES : AddRoundKey

Injecte la clé dans l'état (confusion)



AES : SubBytes

Utilisation de la Sbox : tables définies dans le standard
Fonction non linéaire (confusion)



Lecture sbox :
Pour chaque octet (8bits) :
Lire premier nibble (4bits) sur les
Lire second nibble sur les colonnes

AES S-box																
	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

The column is determined by the least significant nibble, and the row by the most significant nibble. For example, the value 9a₁₆ is converted into b8₁₆.

AES : ShiftRows

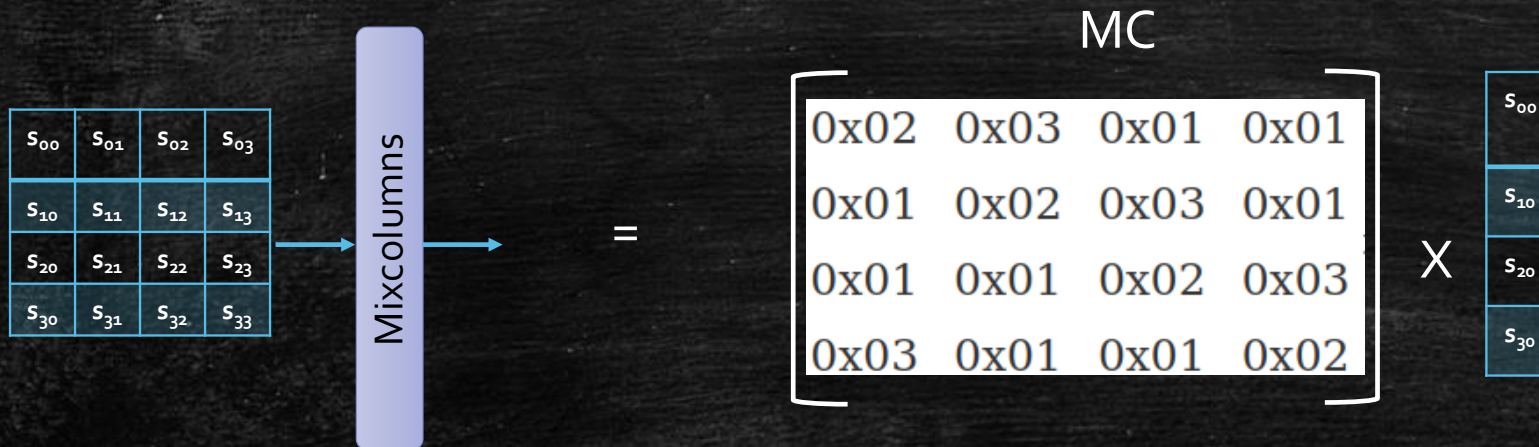
Permutation cyclique sur les lignes (diffusion)



Décalage de chaque ligne (row) vers la « gauche »

AES : Mixcolumn

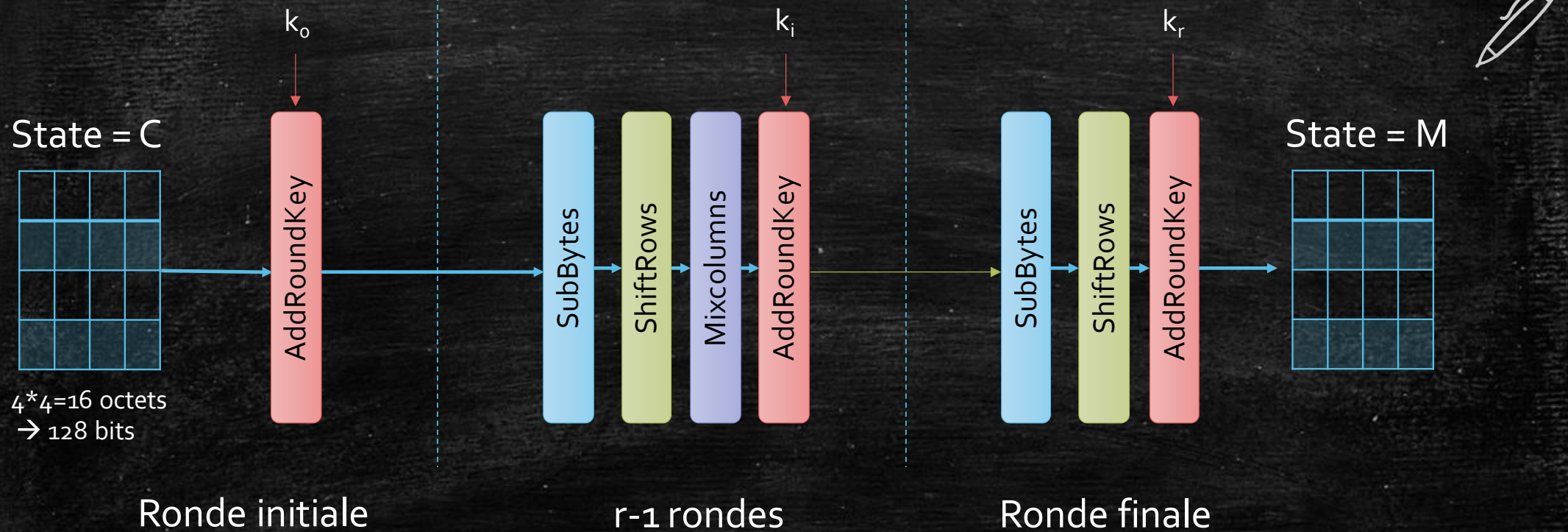
- Transformation linéaire (diffusion)



Pour chaque colonne, multiplication par la matrice MC

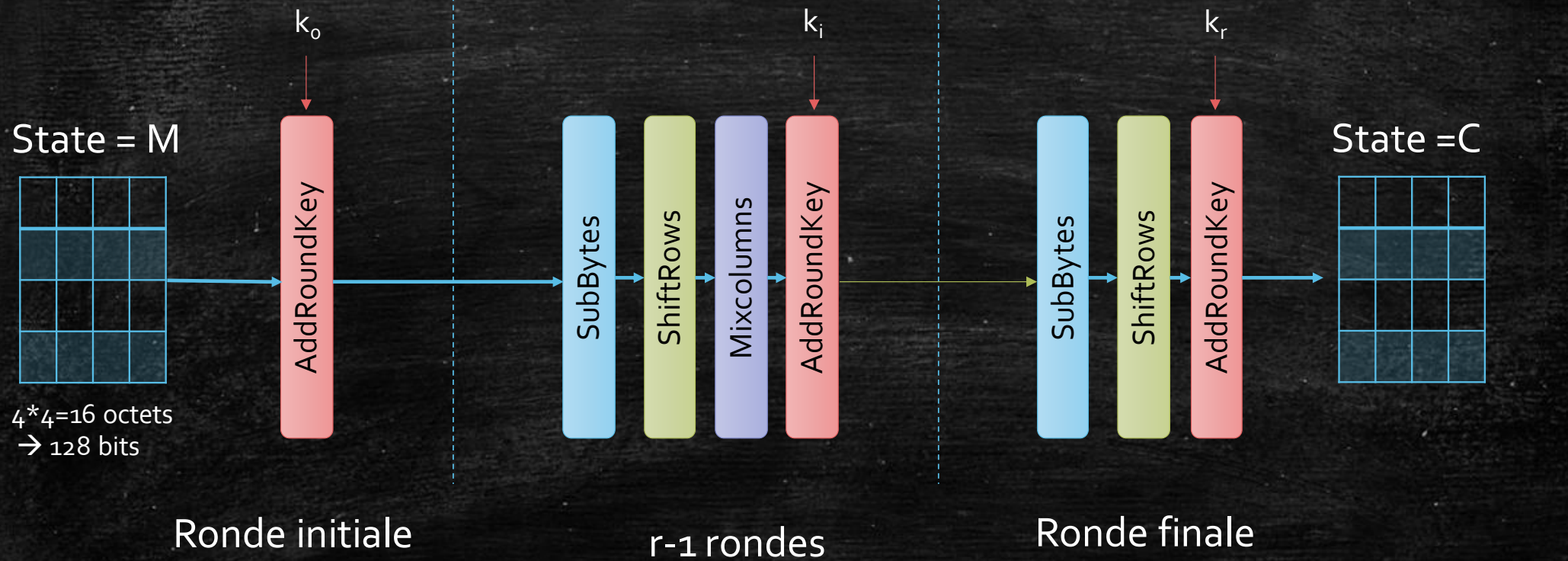
AES : Déchiffrement

KeyExpansion(K) = (k_0, \dots, k_r)



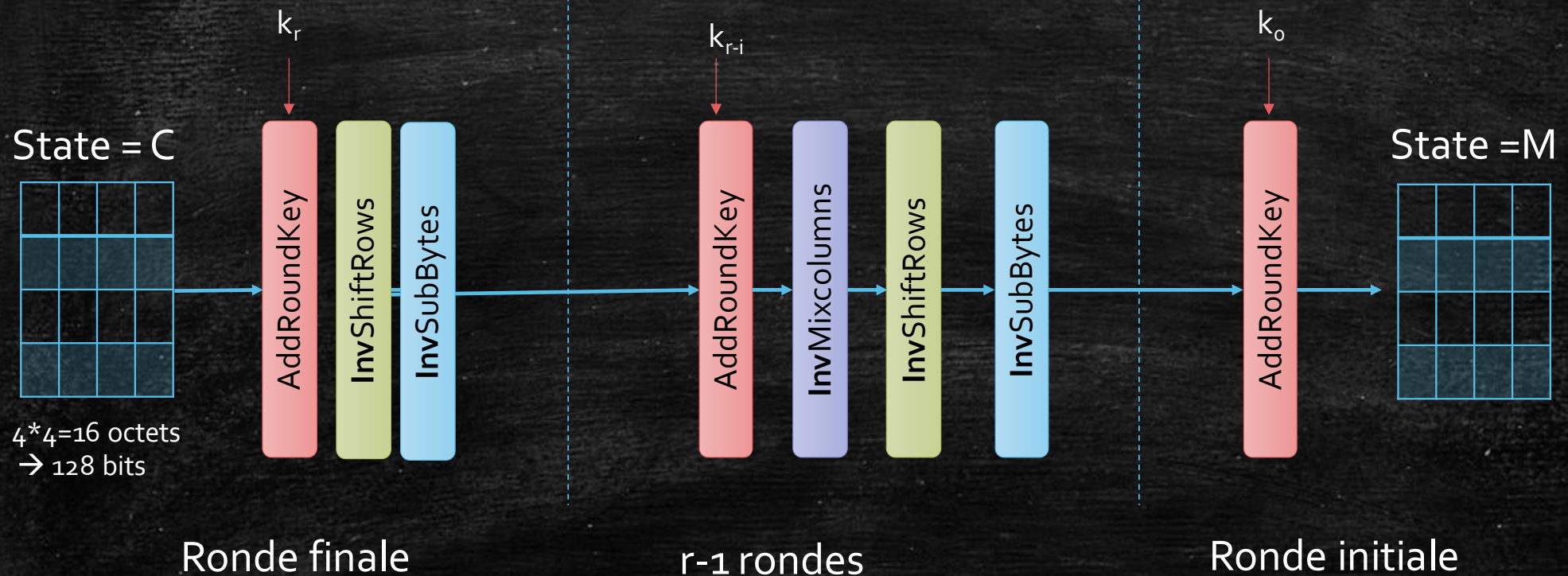
AES : Déchiffrement

KeyExpansion(K) = (k₀, ..., k_r)



AES : Déchiffrement

$$\text{KeyExpansion}(K) = (k_0, \dots, k_r)$$

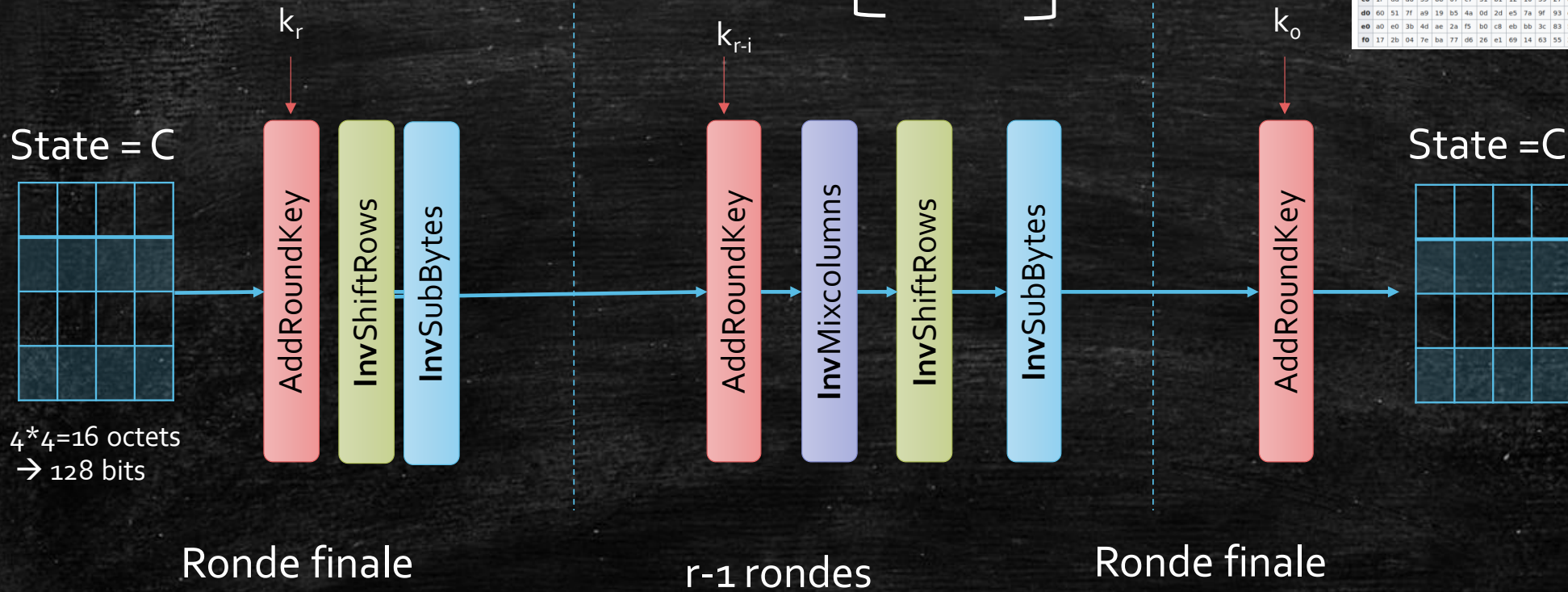


AES : Déchiffrement

KeyExpansion(K) = (k₀, ..., k_r)

$$MC^{-1} = \begin{bmatrix} 0x0E & 0x0B & 0x0D & 0x09 \\ 0x09 & 0x0E & 0x0B & 0x0D \\ 0x0D & 0x09 & 0x0E & 0x0B \\ 0x0B & 0x0D & 0x09 & 0x0E \end{bmatrix}$$

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
10	7c	e3	39	92	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
20	54	7b	94	32	a6	c2	23	3e	ee	4c	95	db	42	fa	c3	4e
30	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
40	72	fb	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
50	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
60	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
70	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
80	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
90	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
ao	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
bo	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
co	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
do	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
eo	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
fo	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d



Ronde finale

r-1 rondes

Ronde finale

Chiffrement par bloc : utilisations

- Exemples :
 - DES, 3DES
 - AES
 - IDEA
 - Camélia
 - Skinny (Lightweight cryptography)
 - Et bien d'autres!
- Sécurité
 - Pas de preuve
 - Cryptanalyse
 - Taille de clé : au moins 128 bits
 - Taille de bloc : acceptable 64 bits, recommandée 128 bits

# TLS 1.2 (suites in server-preferred order)			
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)	ECDH x25519 (eq. 3072 bits RSA)	FS	
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca9)	ECDH x25519 (eq. 3072 bits RSA)	FS	
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)	ECDH x25519 (eq. 3072 bits RSA)	FS	
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009)	ECDH x25519 (eq. 3072 bits RSA)	FS	WEAK
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a)	ECDH x25519 (eq. 3072 bits RSA)	FS	WEAK
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)	ECDH x25519 (eq. 3072 bits RSA)	FS	
TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8)	ECDH x25519 (eq. 3072 bits RSA)	FS	
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)	ECDH x25519 (eq. 3072 bits RSA)	FS	
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)	ECDH x25519 (eq. 3072 bits RSA)	FS	WEAK
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)	ECDH x25519 (eq. 3072 bits RSA)	FS	WEAK
TLS_RSA_WITH_AES_128_GCM_SHA256 (0x9c)			WEAK
TLS_RSA_WITH_AES_256_GCM_SHA384 (0x9d)			WEAK
TLS_RSA_WITH_AES_128_CBC_SHA (0x2f)			WEAK
TLS_RSA_WITH_AES_256_CBC_SHA (0x35)			WEAK
TLS_RSA_WITH_3DES_EDE_CBC_SHA (0xa)			WEAK

Chiffrement par bloc : sécurité

- Pas de preuve de sécurité
- La confiance dans la sécurité d'un chiffrement par bloc est sa **résistance à la cryptanalyse dans le temps**
- Cryptanalyse :
 - Différentielle : introduire une différence entre deux messages et prédire les conséquences sur les chiffrés correspondants
 - (Linéaire)

Chiffrement par bloc : utilisations

- Très utilisé en cryptographie
- Constructions de cryptosystèmes divers
 - Chiffrement symétrique (modes opératoires de chiffrement)
 - Calcul de tag (MAC)
 - Chiffrement authentifié
 - Générateur d'aléa
- Très pratique
 - Une implémentation pour plusieurs usages
 - Une **petite** clé
 - Implémentation très **rapide**

Chiffrement par bloc : utilisations

- Chiffrer un message de n'importe quelle taille avec un chiffrement par bloc
 - Modes opératoires utilisés seulement pour chiffrer une donnée
 - Garantir la **confidentialité** seulement
- Modes opératoires utilisés pour garantir **l'intégrité** d'une donnée
 - MAC
- Mode opératoire pour garantir à la fois la **confidentialité** et **l'intégrité** de la donnée (récent → chiffrement authentifié)

Modes opératoires de chiffrement

Avantages

Un mode de chiffrement

- permet de (ré)utiliser une primitive de chiffrement par bloc existante
- peut être prouvé moyennant des hypothèses sur la primitive sous-jacente
- peut fonctionner comme un chiffrement par flot
- peut combiner un mécanisme d'authentification des données (MAC)
- peut s'adapter aux différents contextes d'utilisation

Modes opératoires de chiffrement

- Garantir la confidentialité d'une donnée



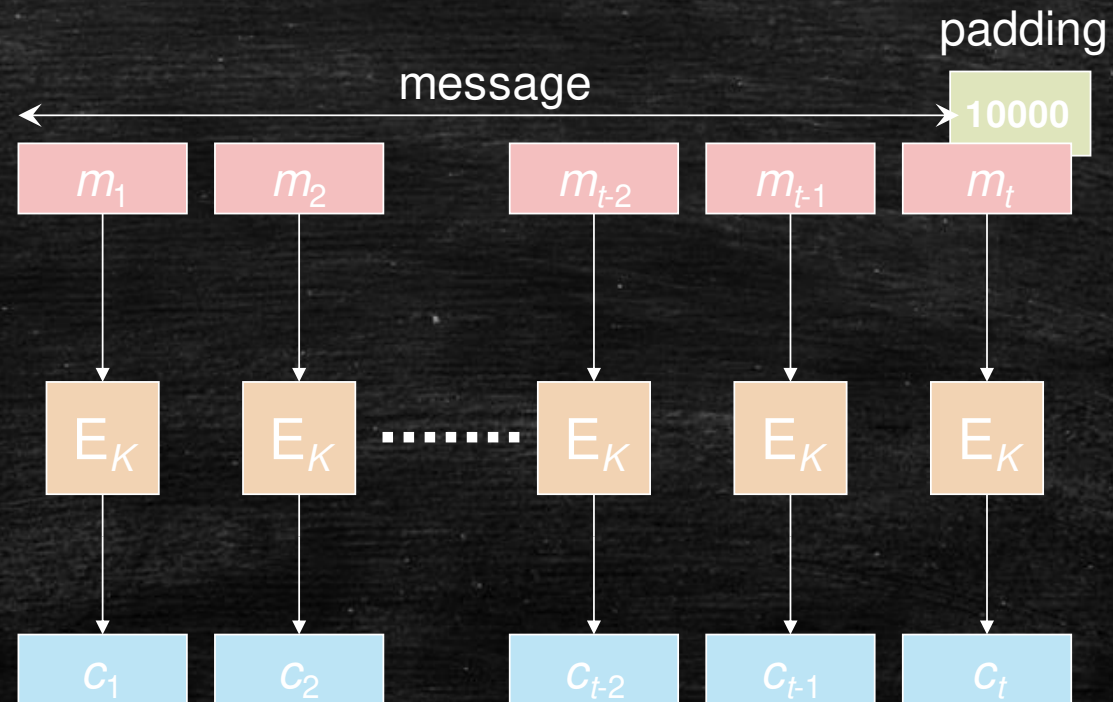
- Plusieurs **niveaux d'attaquants** (rappel)
 - Un attaquant ne doit pas être capable de retrouver la clé secrète
 - Un attaquant ne doit pas être capable de déchiffrer sans la clé secrète
 - Un attaquant ne doit apprendre aucune information sur le message clair à la vue de son chiffré

Modes opératoires de chiffrement

- Comment chiffrer un message de n'importe quelle taille ?
- Quatres modes spécifiés (standards FIPS post DES)
 - **Electronic Code Book Mode (ECB)**
 - **Cipher Block Chaining Mode (CBC)**
 - Cipher Feedback Mode (CFB)
 - Output Feedback Mode (OFB)
- Mode ajouté après publication de l'AES
 - Counter Mode (CTR)

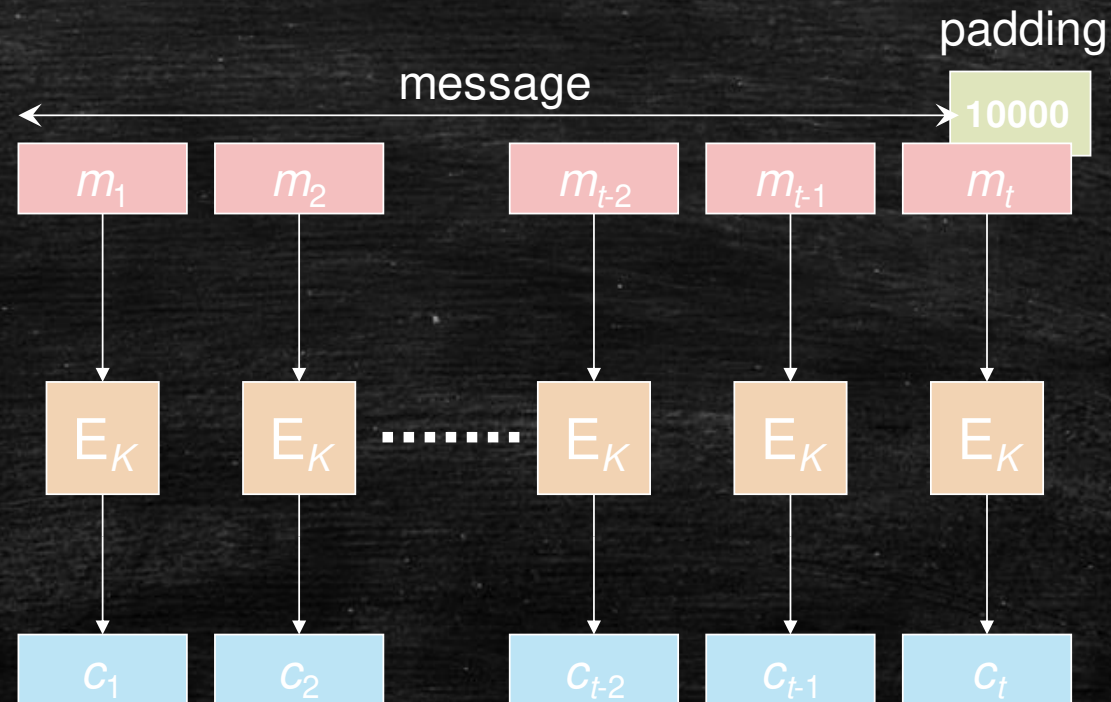
Mode naturel : ECB

- Message de taille quelconque, non multiple de n bits



Mode naturel : ECB

- Message de taille quelconque, non multiple de n bits



Parallélisable ?

Expansion ?

Primitives utilisées ? (E_K / D_K)

État ?

Mode déterministe/probabiliste ?

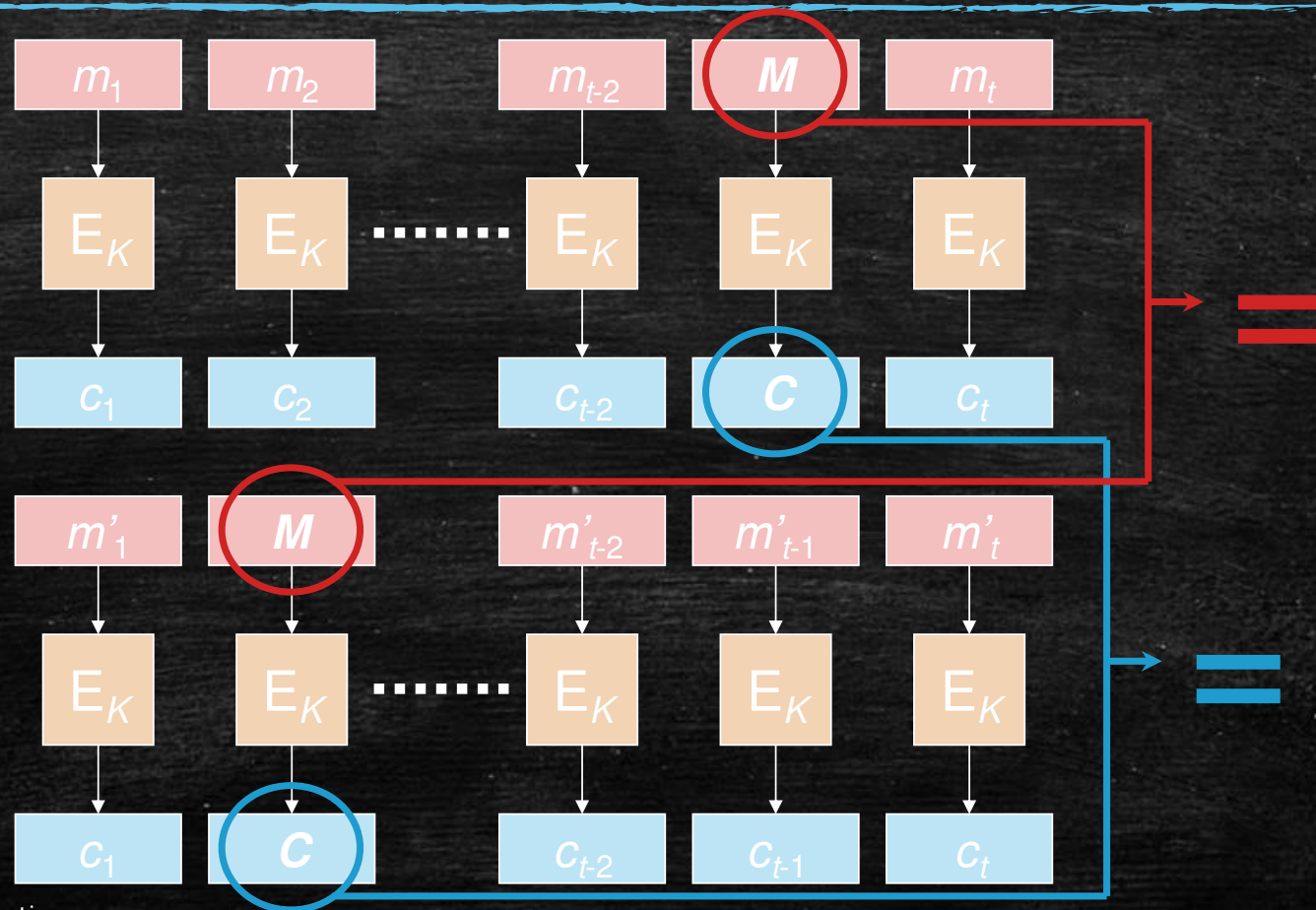
Pas de propagation d'erreurs ?

ECB : Propriétés

Mode **ECB** : Electronic CodeBook

- **Parallélisable** : chaque bloc est chiffré indépendamment des autres
- **Expansion** : taille du chiffré = taille du clair + bourrage
- **Primitives utilisées** : E_K pour chiffrer, D_K pour déchiffrer
- **Pas d'état** (stateless)
- **Mode déterministe** : le même clair donne toujours le même chiffré
- **Pas de propagation d'erreurs** : une erreur sur un bloc de chiffré modifie seulement un bloc de clair

ECB : Un mode (très) déterministe



ECB : Sécurité

Chaque bloc est chiffré indépendamment des autres :

- Un même bloc de chiffré correspond toujours au même bloc de clair
- Suppression d'un bloc
- Inversion possible des blocs
- Un attaquant peut créer un dictionnaire
- Attaque statistique parfois possible

Mode non sûr!

ECB : Sécurité

Chaque bloc est chiffré indépendamment des autres :

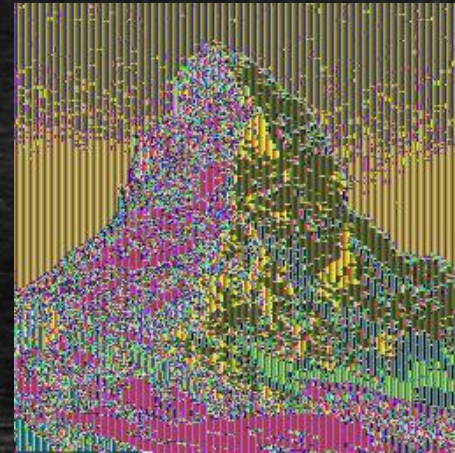
- Un même bloc de chiffré correspond toujours au même bloc de clair
- Suppression d'un bloc
- Inversion possible des blocs
- Un attaquant peut créer un dictionnaire
- Attaque statistique parfois possible

Mode non sûr!

Illustration



ECB



ECB : Mode déterministe

- Le mode ECB est **déterministe**
- Le déterminisme d'un algorithme de chiffrement peut être problématique dans certains contextes :
 - « si l'on chiffre plusieurs fois un message, on obtient le même chiffré chaque fois »
- Exemple : vote électronique

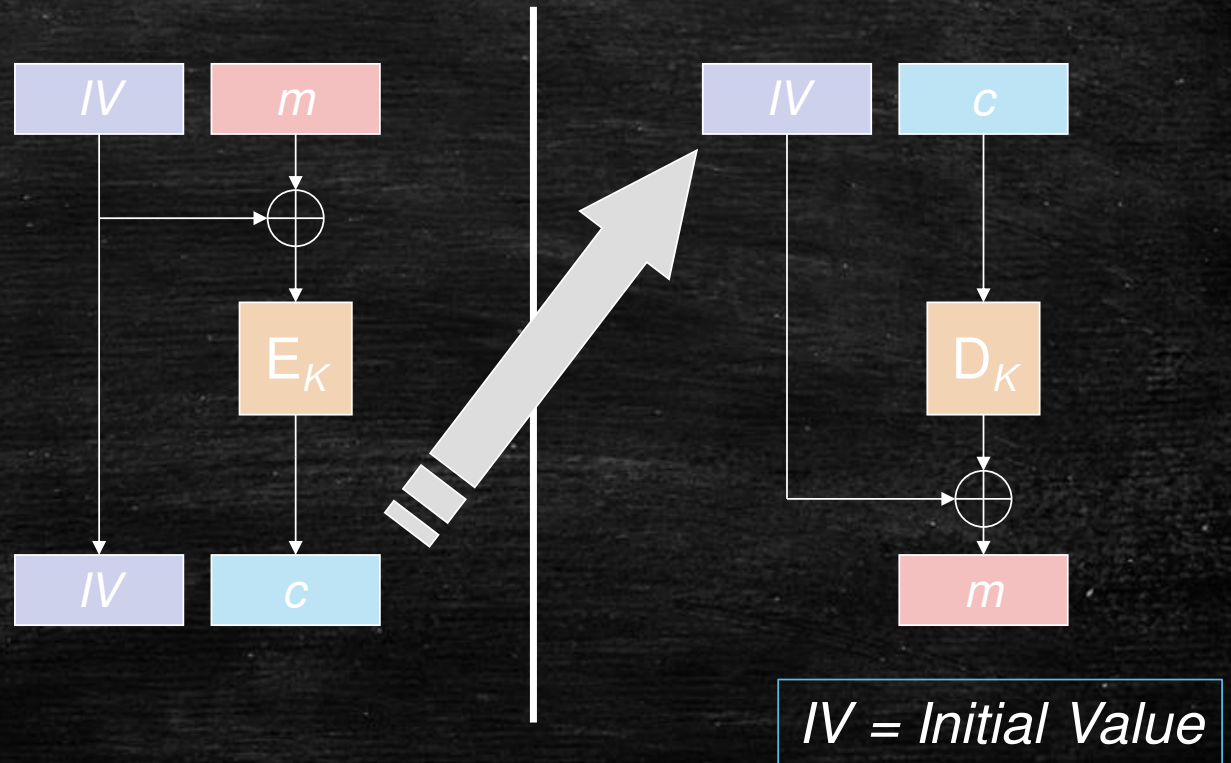


Solution

- « Randomiser » chaque bloc de clair
 - Chaque bloc de clair est masqué à l'aide d'une valeur aléatoire
 - Les attaques statistiques ne s'appliquent plus
 - Tous les blocs d'entrée sont équiprobables
 - Toutes les valeurs de sortie peuvent être atteintes avec même probabilité

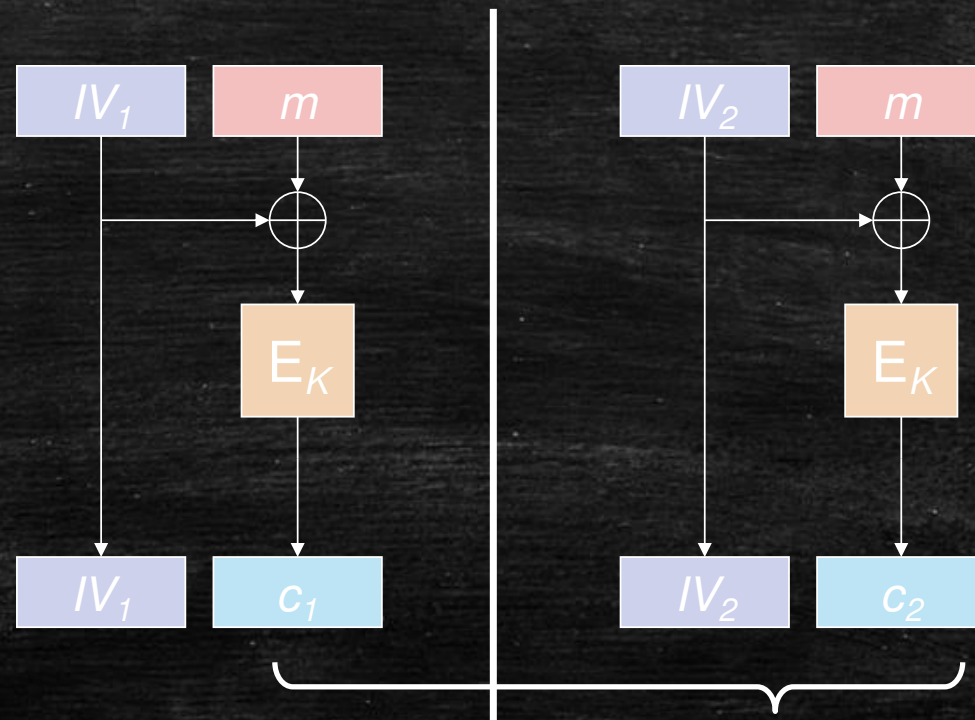
Vers le mode CBC

- Idée : rendre en partie aléatoire le chiffrement



Vers le mode CBC

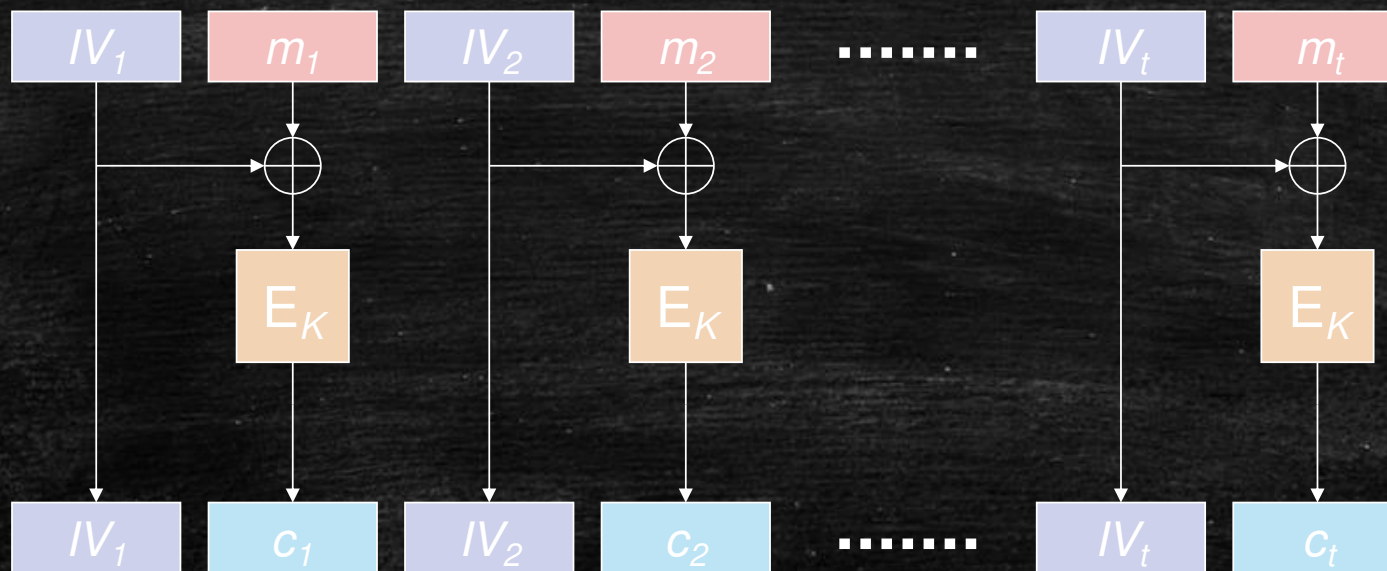
- Idée : rendre en partie aléatoire le chiffrement



c_1 et c_2 n'ont rien à voir

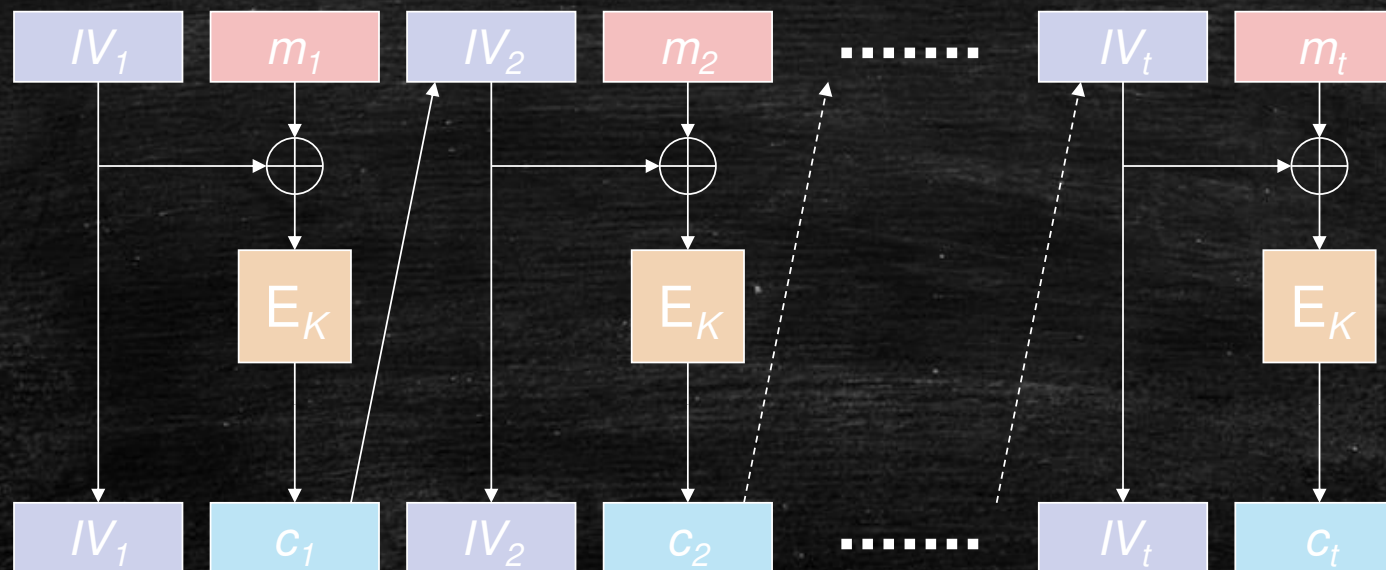
Mode CBC

- Un IV /bloc : *première idée...*



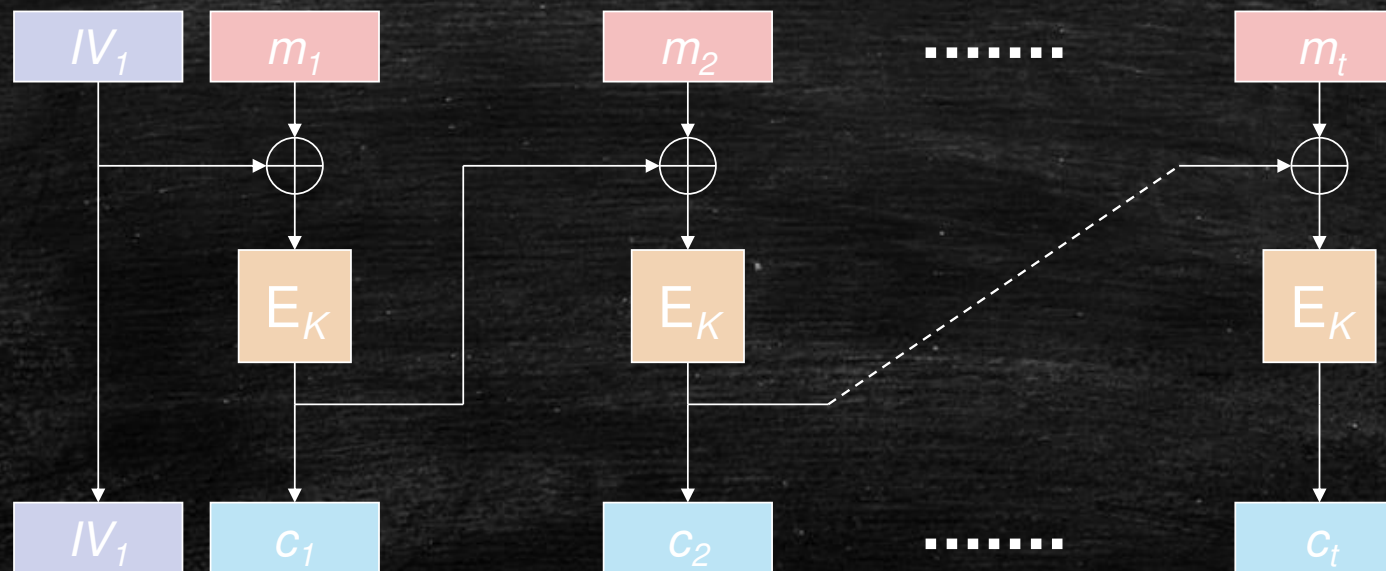
Mode CBC

- CBC = utiliser c_i comme IV_{i+1} ($IV_2 = c_1$, $IV_3 = c_2$, ...)

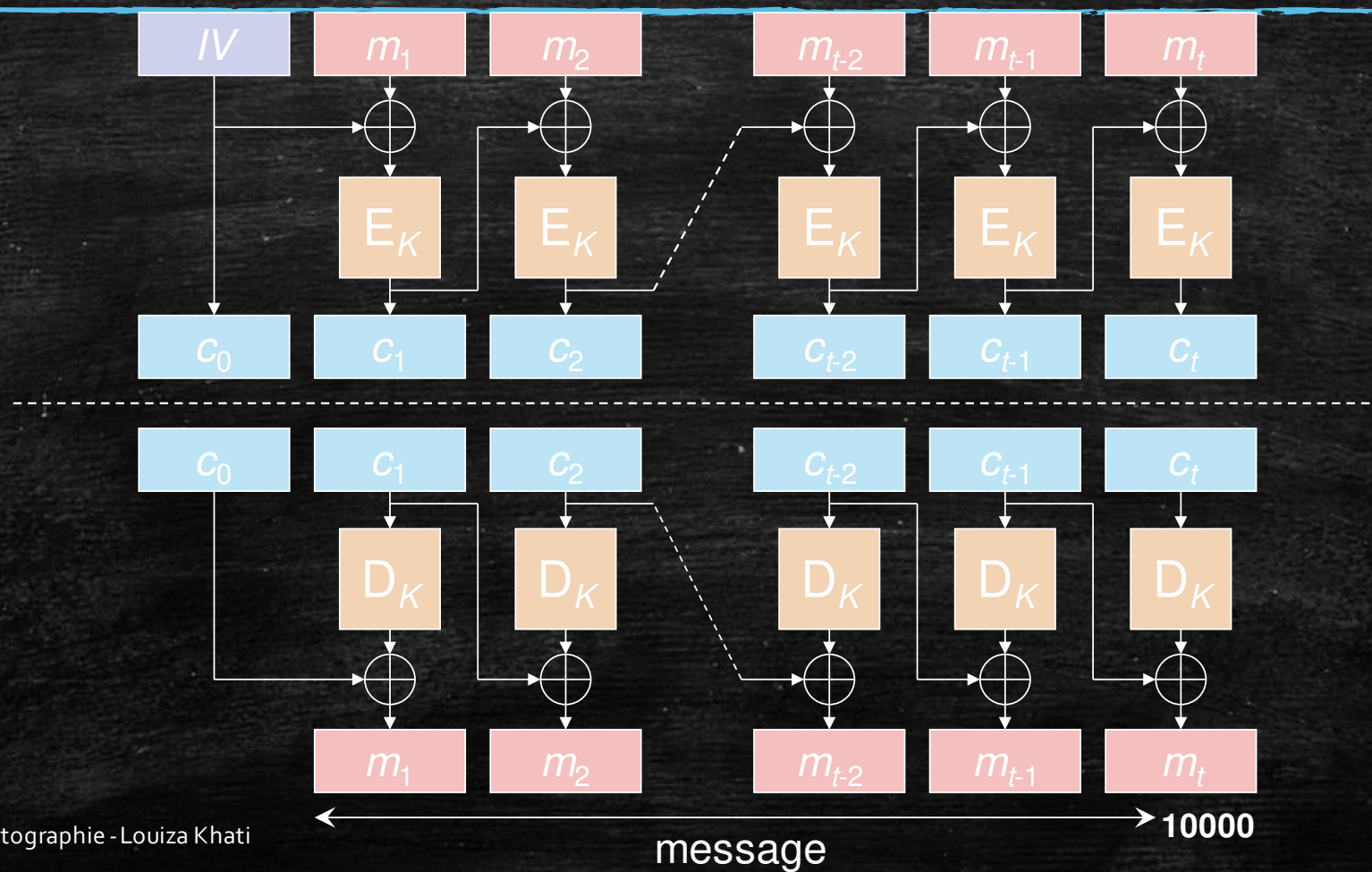


Mode CBC

- CBC = utiliser c_i comme IV_{i+1} ($IV_2 = c_1$, $IV_3 = c_2$, ...)

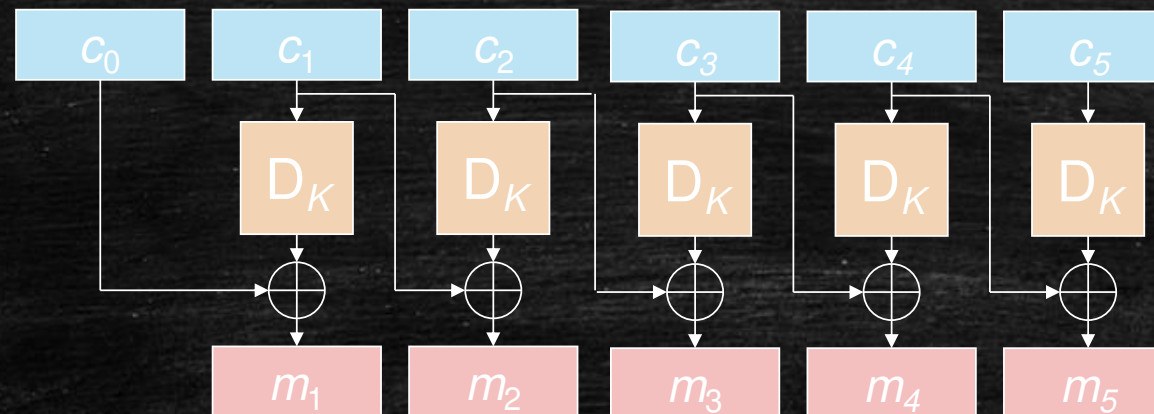


Mode CBC : chiffrement/déchiffrement

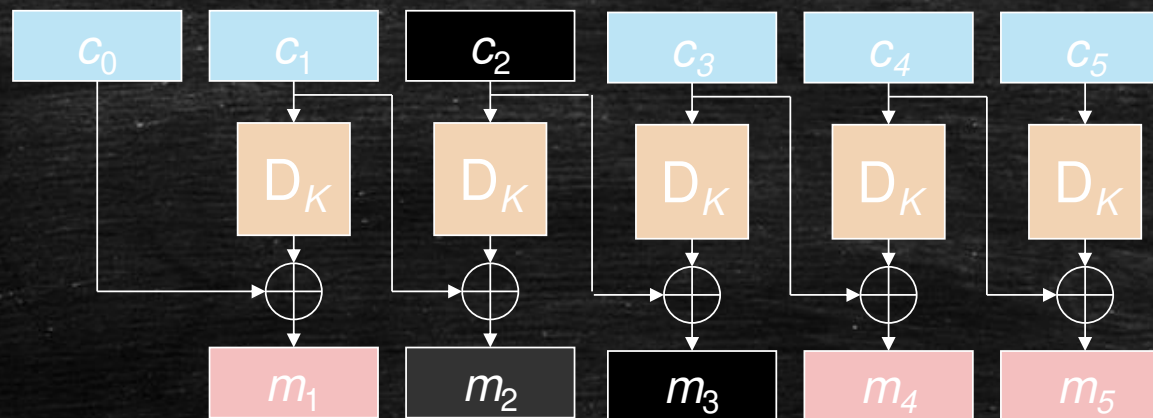


CBC : Propagation d'erreur

- Impact du chainage sur la propagation d'erreurs ?
- Erreur de transmission = erreur de déchiffrement



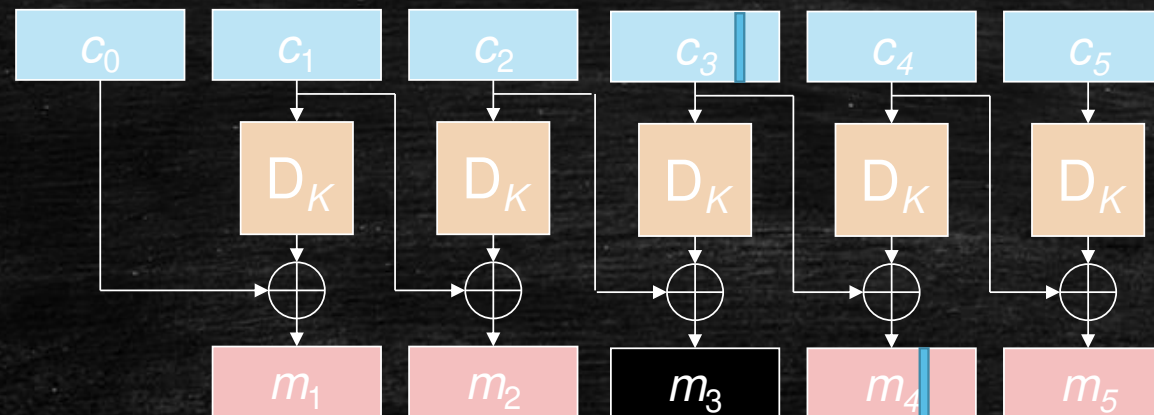
CBC : Propagation d'erreur



Propagation d'erreur limitée en déchiffrement

CBC : Malléabilité

- Malléabilité :
 - Modifier des bits en sortie et connaître les impacts sur les entrées
- Sécurité pour un adversaire CPA!



CBC : Propriétés

Mode **CBC** : Cipher Block Chaining

- **Non parallélisable** en chiffrement, **parallélisable** en déchiffrement
- **Expansion** : taille du chiffré = taille de l'IV + taille du clair + bourrage
- **Gestion de l'IV** : transmis en clair avec le chiffré, sa valeur doit être **aléatoire (c-à-d non prédictible)**
- **Primitives utilisées** : E_K pour chiffrer, D_K pour déchiffrer
- **Pas d'état**
- **Mode probabiliste (randomisé)** : pour un même clair, les chiffrés sont différents si IV choisis sont différents
- **Propagation d'erreurs limitée en déchiffrement** : une erreur sur le bloc de chiffré transmis C_i modifie seulement les clairs M_i et M_{i+1}

Modes opératoires : ECB VS CBC



Message



Chiffré ECB



Chiffré CBC
(avec IV aléatoire)

CBC : Sécurité

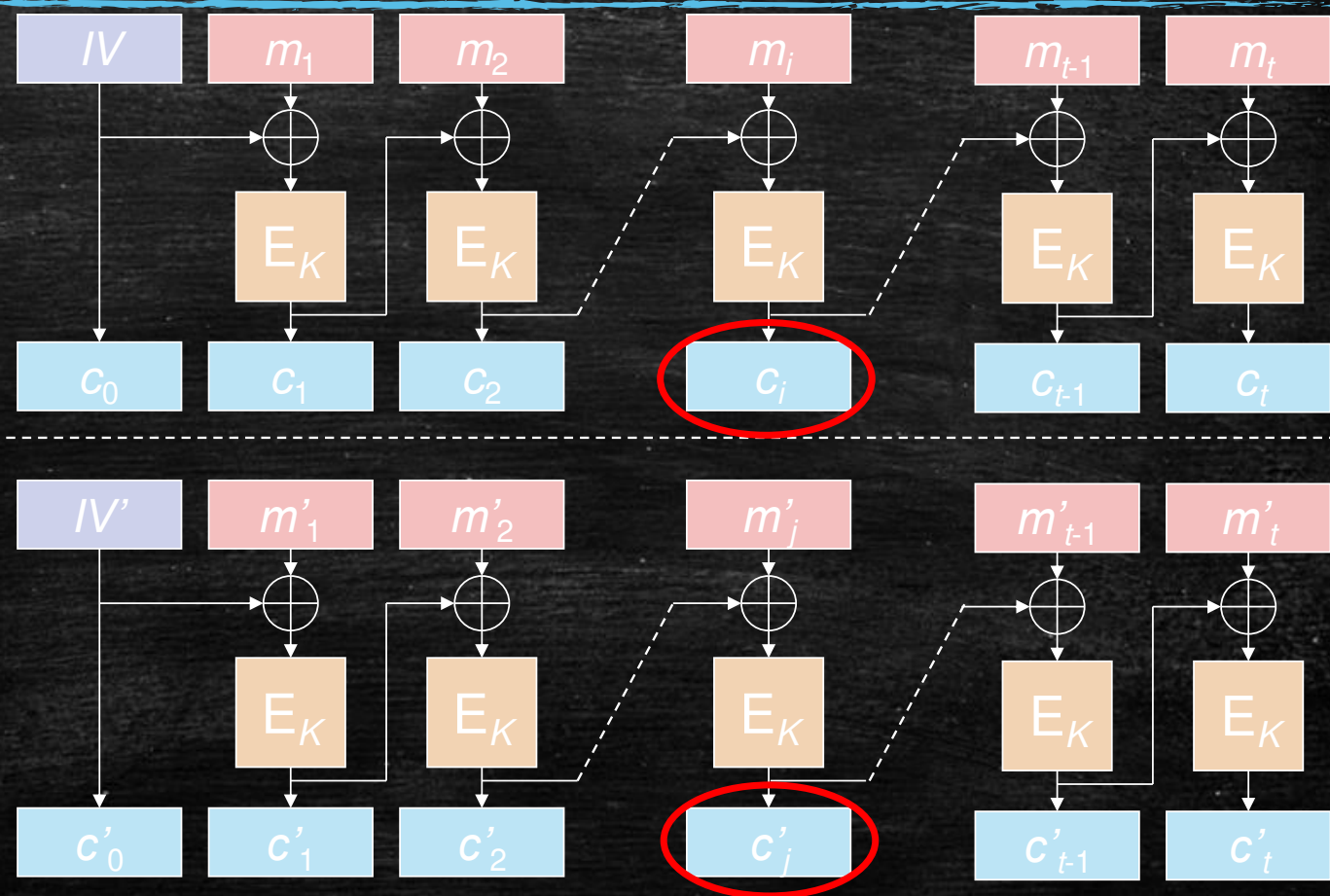
- Que se passe-t-il si deux blocs de chiffrés sont identiques = collision ?
- Quelle fuite d'information peut-on tolérer ?

CBC : Sécurité

$$C_i = C'_j$$

$$C_{i-1} \oplus M_i = C'_{j-1} \oplus M'_j$$

$$M_i \oplus M'_j = C_{i-1} \oplus C'_{j-1}$$



CBC : Sécurité

- Si deux blocs **collisionnent**, de l'information fuit : on obtient une relation linéaire entre deux blocs de clair

$$M_i \oplus M_j = C_{i-1} \oplus C_{j-1}$$

- La confidentialité au sens le plus fort n'est plus assurée
- Quelle est la **probabilité** qu'une telle collision se produise ?

CBC : Sécurité

- Les collisions se font sur des blocs de n bits
 - 2^n valeurs possibles
 - La primitive E_K est « sûre » : toutes ses valeurs sont équiprobables
- Dès que $\sqrt{2n} = 2^{n/2}$ blocs ont été chiffrés, deux d'entre eux collisionnent avec forte probabilité (paradoxe des anniversaires)



Admettre

CBC : Sécurité

- Quand $2^{n/2}$ blocs ont été chiffrés, de l'information fuit sur les messages clairs
- Importance de la taille des blocs pour la primitive de chiffrement
- En pratique :
 - Pour le DES : 2^{32} blocs (= 32 Go) avant une collision
 - Très réaliste sur un réseau gigabit
 - Discutable sur une carte à puce
 - Pour l'AES : 2^{64} blocs (~ 275 milliards de Go) avant collision, pas de risque pratique

CBC : Sécurité

- Fuite d'information au bout de $2^{n/2}$ blocs de messages chiffrés
- Preuve de sécurité :
 - Bellare, Desai, Jokipii, Rogaway : *A Concrete Security Treatment of Symmetric Encryption*. 1997
 - Hypothèse : E=PRP
 - Niveau de sécurité : $2^{n/2}$ blocs de messages
- Le niveau de sécurité donné par la preuve rejoint celui de la meilleure attaque connue

Padding

- Doit permettre de retrouver le clair sans ambiguïté
- Exemple 1 : « 10...0 » ISO/IEC 9797-1
 - Si dernier bloc d'exactly n bits :
 - On ajoute un bloc 10...00 de n bits
 - Si dernier bloc de n-1 bits :
 - On ajoute simplement un 1
 - Ce padding est sans ambiguïté et déterministe.
- Exemple 2 : Padding avec le nombre d'octets manquants (PKCS#7)
 - Bloc de 64 bits : 0x11223344 → 0x1122334404040404

Padding

- Propriété d'injectivité du padding
 - $\text{Pad}(M) = M \parallel \text{padding}$
 - Pour un message paddé $\text{pad}(M)$ un seul antécédent M
- Sinon possibilité de trouver des collisions
- Attaque par « oracle de padding »
 - Le serveur fait fuir de l'information : padding valide/invalid
 - Possibilité de récupérer le message en entier dans certains cas

Chiffrement symétrique

- « Chiffrement par bloc » (= Chiffrement par bloc+ mode opératoire)
 - Chiffre les données bloc par bloc (128 bits généralement)
 - Sécurité d'un mode prouvée sous des hypothèses raisonnables
 - Permet de réutiliser une primitive donnée (implémentation hardware)
 - Besoin de padding/bourrage
- Chiffrement par flots
 - Chiffre les données bit à bit ou octet par octet (pas de padding)
 - Très peu de schémas non cryptanalysés (Chacha20)
 - Pas de preuve de sécurité (tout comme le chiffrement par bloc)

Chiffrement symétrique

- Algorithme de chiffrement $ENC = \{\text{keygen}, \text{encrypt}, \text{decrypt}\}$
 - $K \leftarrow \text{keygen}(k)$, k taille de la clé
 - $C \leftarrow \text{Encrypt}(K, M)$ probabiliste (en général).
 - On considère que l'IV est généré dans l'algorithme
 - $M \leftarrow \text{Decrypt}(K, C)$ déterministe
 - S'il y a un IV, il fait généralement parti du chiffré
- Sécurité prouvée pour les modes opératoires :
 - Vérifier les hypothèses (sur les IVs notamment)
 - $E_K = \text{PRP}$

A retenir

- Primitives de chiffrement par bloc
 - Utilisées avec un modes opératoires
 - Sécurité :
 - Résistance à la cryptanalyse (pas de preuve de sécurité)
 - Paramètres de sécurité : taille de bloc recommandé 128 bits (paradoxe des anniversaires), taille de clé (128 bits minimum)
 - Exemple : 3DES, AES, Camélia etc.

A retenir

- Chiffrements symétriques
 - A-chiffrement par bloc (primitive de chiffrement + mode opératoire)
 - Preuve sur le mode opératoire : IV aléatoire
 - Une primitive sûr : AES-128 (exemple)
 - B-Chiffrement par flot
 - Sécurité : pas de preuve de sécurité
- Sécurité :
 - Chiffrement probabiliste
 - Besoin d'aléa
 - Il est recommandé d'ajouter un mécanisme d'intégrité

Merci pour votre
attention

Des questions?

Bibliographie

- Livres:
 - La Guerre des Codes (David Kahn)
 - Histoire des Codes Secrets (Simon Singh)
 - La Science du Secret (Jacques Stern)
 - Cryptographie Appliquée (Bruce Schneier)
 - Cryptographie : Théorie et Pratique (Stinson)
- Pointeurs internet
 - Handbook of Applied Cryptography
www.cacr.math.uwaterloo.ca/hac