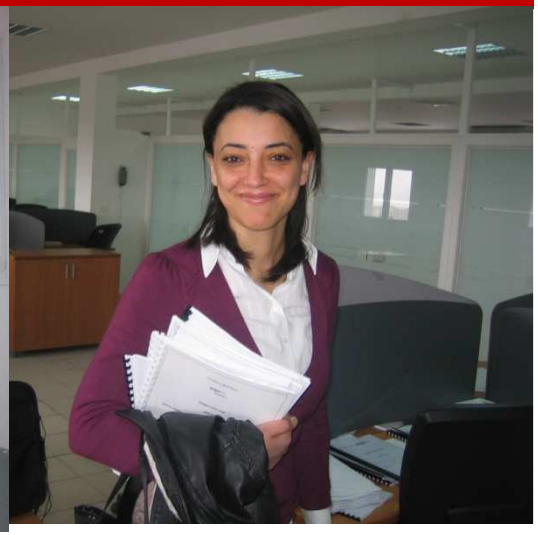
Ecole Supérieure Privée
d'Ingénierie et de Technologies

www.esprit.ens.tn

Sécurité des réseaux informatiques

Ramzi Ouafi
Assistant Technologue – Esprit
e-mail : Ramzi.ELOUAFI@esprit.ens.tn

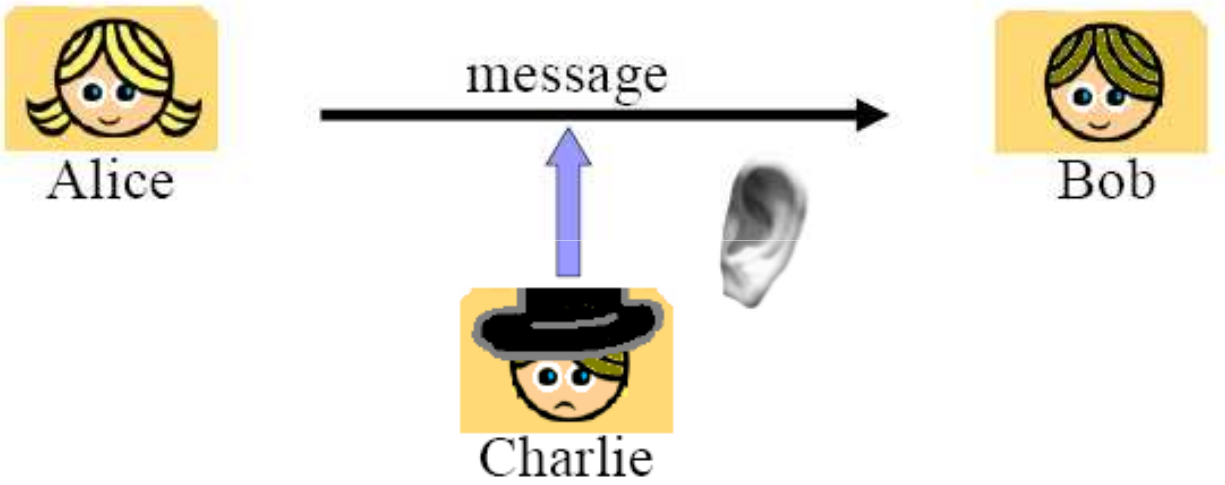


Menaces utilité de la cryptographie

Les fonctions Cryptographiques

Menaces utilité de la cryptographie

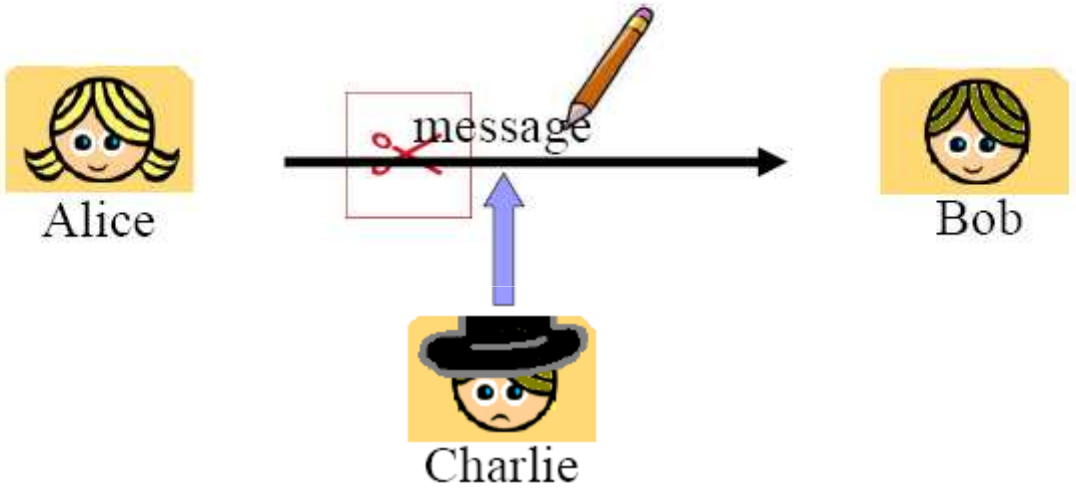
- Attaques passives



- Menace contre la *confidentialité* de l'information : une information sensible parvient à une personne autre que son destinataire légitime.

Menaces utilité de la cryptographie

- Attaques actives : interventions sur la ligne



- Menace contre *l'intégrité et l'authenticité* de l'information

Terminologie

- Chiffrement:
  - Opération de codage pour rendre une information incompréhensible en l'absence d'un décodeur particulier.

- Cryptogramme:
  - Message caché écrit en caractères secrets, en langage chiffré.

- Cryptographie:
  - La science des méthodes permettant de transmettre des données de manière confidentielle.

- La Cryptanalyse:
  - Etude des procédés cryptographiques afin de trouver des failles et pouvoir décrypter des textes chiffrés sans connaître la clé de déchiffrement.

Moyens cryptographique

- **Cryptage/décryptage (chiffrement)**
  - Confidentialité: données incompréhensibles lors du transit
  - Authentification: nécessité de disposer de « l'autre » clé
- **Hachage cryptographique**
  - Intégrité: toute modification est détectée
- **Signature numérique**
  - Authentification: s'assurer de l'identité de l'émetteur de données
  - Intégrité: s'assurer que, si la signature est valide, les données n'ont pas changées depuis leur signature.
  - Non-répudiation: prouver que l'émetteur a bien émis ces données, et est le seul a avoir pu le faire (autorisation de prélèvement bancaire)

1. Cryptographie symétrique

Historique

- Auparavant : la sécurité reposait sur le fait que l'algorithme utilisé était secret
  - Exemple : Alphabet de César, Vigenère
  - Exemple : Alphabet de César : décalage de trois positions des lettres de l'alphabet **mod (26)**.
    - => CESAR -> FHVDU

- Aujourd'hui : les algorithmes sont connus de tous : la sécurité repose uniquement sur le secret d'une clef.

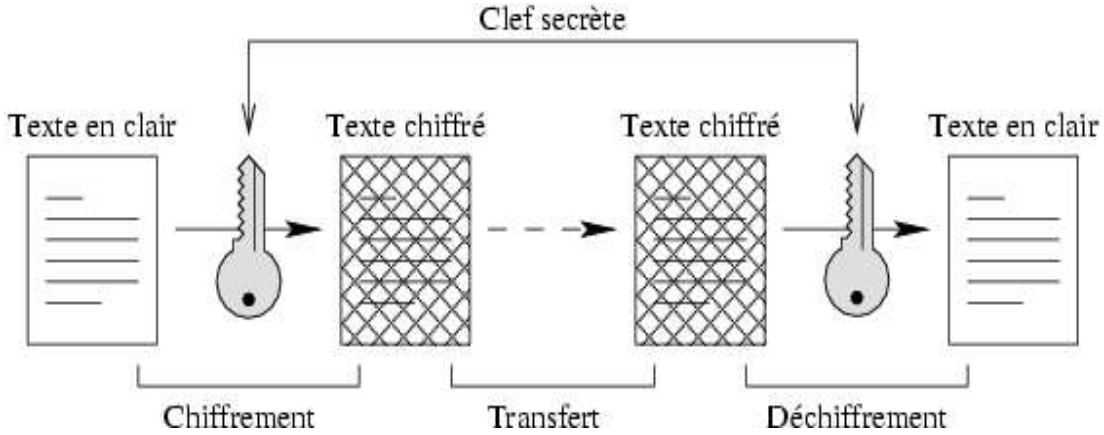
  - Premier Exemple : Dernière guerre : Machine Enigma
  - 75-77 : Premier standard de chiffrement américain, **le DES**
  - 76 : nouvelle forme de cryptographie : **la cryptographie à clé publique, introduite par Diffie et Hellman (Exemple : DH, RSA,)**

Le chiffrement

- Le chiffrement repose sur:
  - Une clé de chiffrement
  - Un algorithme de chiffrement
- La robustesse d'un système de chiffrement est déterminée par:
  - **La taille de la clé** : plus la clé est longue exemple clé de taille 128 bits 10^{38} possibilités.
    - Trouver la clé nécessite de la puissance et un temps de calcul énorme.
  - **La qualité de l'algorithme:**
    - la qualité d'un algorithme repose sur sa fiabilité mathématique et surtout pas sur le secret de sa réalisation.
  - **L'implémentation des algorithmes:**
    - Il est souvent bien plus facile de contourner une mauvaise implémentation que d'attaquer un algorithme
- Types d'algorithmes de chiffrement:
  - **Les Algorithmes Symétriques , à clef secrète**
  - **Les Algorithmes Asymétriques, à clef publique**

Le chiffrement symétrique

- **Caractéristiques**
  - Opérations de chiffrement et déchiffrement identiques
    - Clef identique pour les opérations de chiffrement / Déchiffrement**.**
    - La clef doit être partagée entre

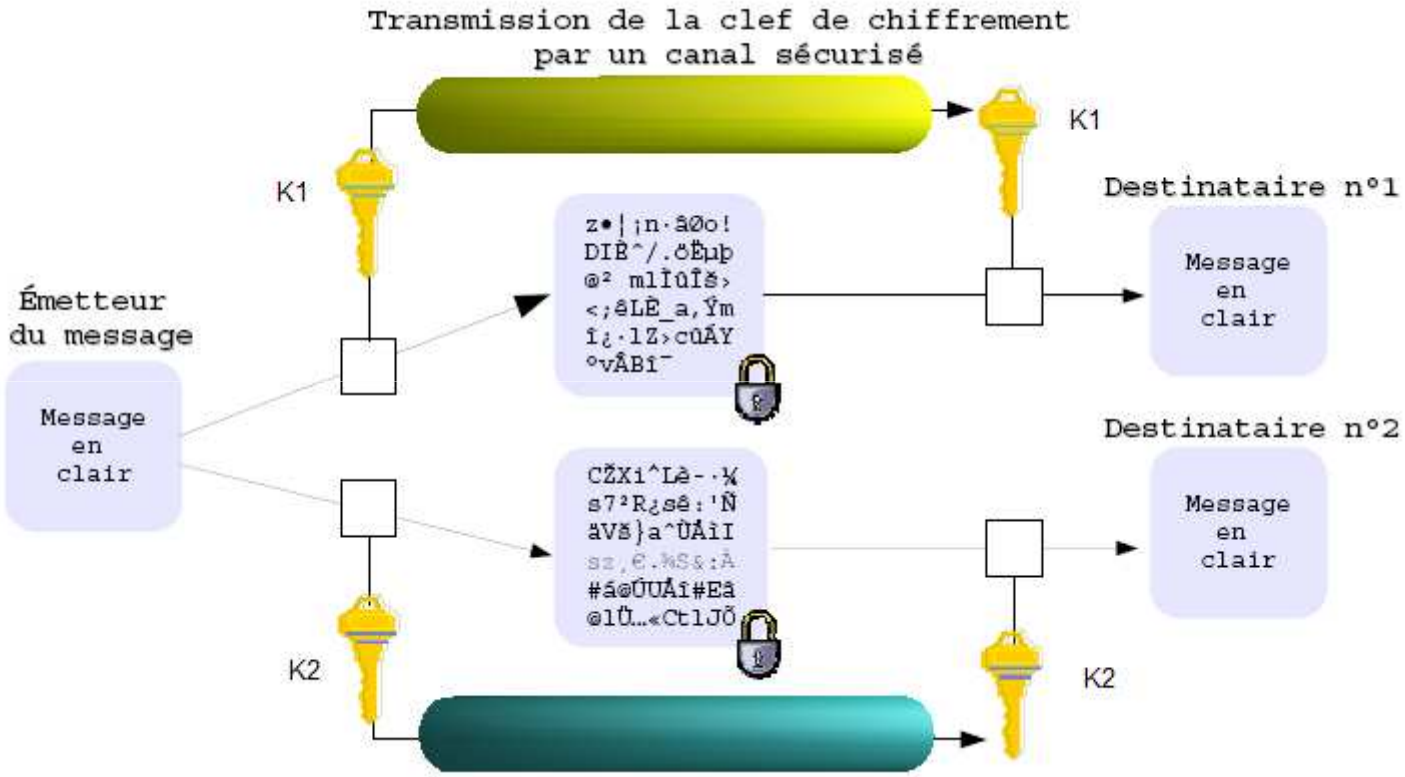


  - Utilisation pour le chiffrement des données.
  - **Rapidité**, et **facilité** de mise en œuvre sur des circuits
    - Se basent sur des fonctions mathématiques simples : Permutations, XOR, décalage
    - **1000** fois plus rapides que les algorithmes **asymétriques**

Le chiffrement symétrique

- **Problèmes**
  - Transport sécurisé de la clef de chiffrement
    - Problématique de l'échange de clef
  - Nombre de clefs échangées
    - autant de clés que de récepteurs n participants: n(n-1)/2 clés doivent être générées ($O(n^2)$) →
      - Problème pour la gestion et la diffusion des clés secrètes.

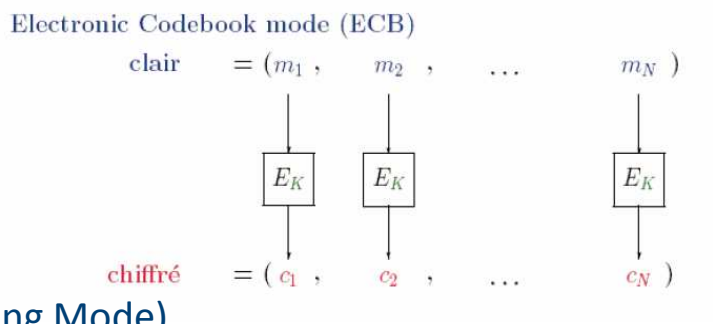
Le chiffrement symétrique



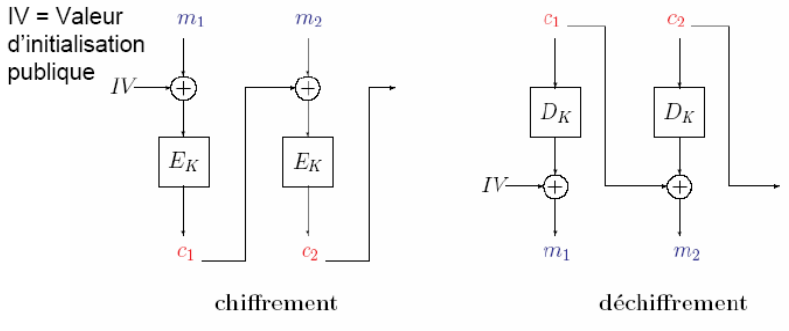
– **n** participants → n(n-1)/2 clefs : par récurrence !!!

Les différentes techniques

- **Algorithmes de chiffrement par blocs**
  - Le texte est découpé en plusieurs blocs.
  - On chiffre un bloc à la fois
  - Exemples : DES (64 bits), AES (128 bits), blowfish (168 bits jusqu'à 448 bits), IDEA (128 bits), etc.
  - **Mode ECB** (Electronic CodeBook Mode)

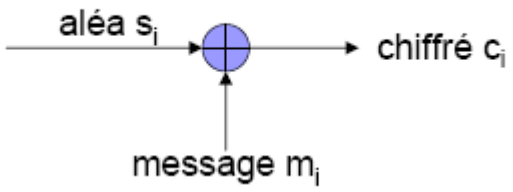


  - **Mode CBC** (Cipher Blok Chaining Mode)



Les différentes techniques

- **Algorithmes de chiffrement en continu à flot (Stream Cipher)**
  - Chiffrement bit à bit
    - Exemple RC4 (RSA Security) : taille de clef variable
  - L'aléa est remplacé par un générateur pseudo- aléatoire (ou chiffrement à flot)
    - Initialisé par la clé commune K
    - Sécurité repose sur les qualités du générateur (grande période, très bon aléa,…)



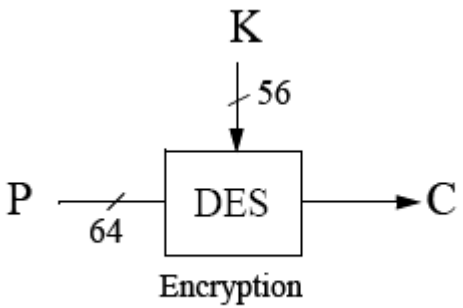
$$m = m_0\ m_1\ m_2\ m_3 \ldots$$
$$\oplus\ s = s_0\ s_1\ s_2\ s_3 \ldots$$
$$= c = c_0\ c_1\ c_2\ c_3 \ldots$$

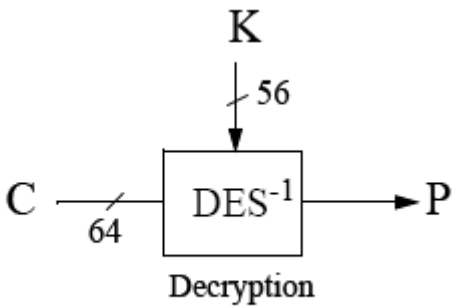
Exemple d'algorithmes

Algorithmes de chiffrement	Concepteur(s)	Taille de la clé	Taille du bloc	Commentaires
RC4 (Continu)	Ron Rivest (1994)	Variable, 128 bits en général	-	Adapté aux codages rapides 10 fois plus rapide que DES
RC2 (Par bloc)	Ron Rivest	Variable	-	Equivalent à DES
DES (Data Encryption Standard) Par bloc	IBM (1976)	56 bits	64 bits & 128 bits	publié par le NIST, normalisé ANSI Relativement rapide Destiné au chiffrement de gros volumes Clé craquée en 1998
Triple DES (Par bloc)	idem	3 clés différentes, 168 bits (EDE)	64 bits	Remplace DES
IDEA (International Data Encryption Algorithm)Par bloc	X.Lai et J. Massey (1990)	128 bits	64 bits	Conçu pour une efficacité maximale lors de calculs logiciels Nécessite une licence
BLOWFISH Par bloc	Bruce Schneier (1994)	Variable, de 32 à 448 bits	64 bits	Rapide Gratuit
CAST-128 Par bloc	Carlisle Adams, et Stafford Tavares	128 bits	64 bits	Alternative à IDEA et Blowfish, mais moins utilisé qu'eux
Rijndael (NIST finaliste de l'AES (Advanced Encryption Algorithm) compétition)	Joan Daemen, et Vincent Rijmen	Variable 128, 192 ou 256 bits	128 bits	Considéré comme l'algorithme de chiffrement le plus fiable actuellement

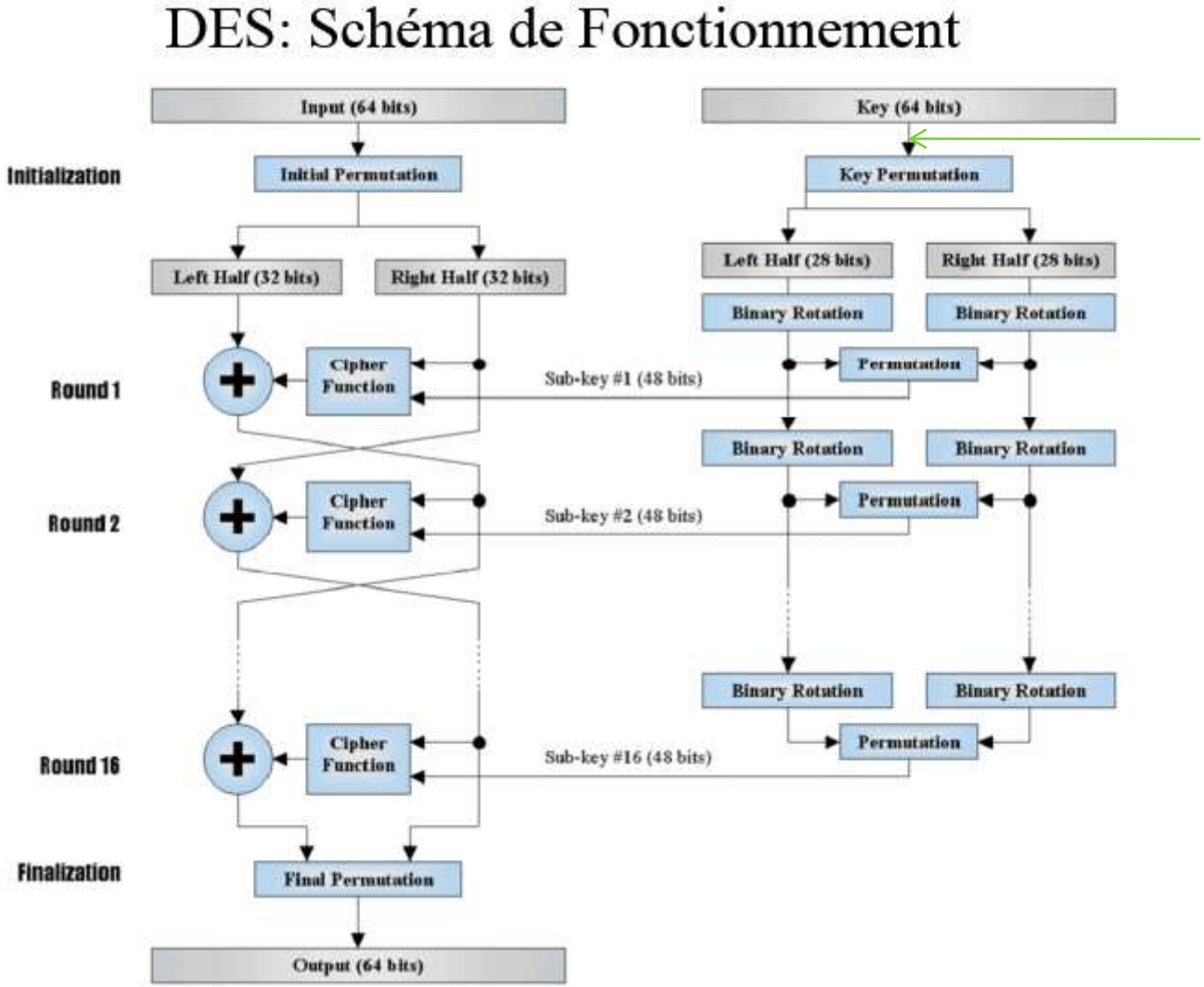
Data Encryption Standard

- DES = Data Encryption Standard Élaboré par le NBS (National Bureau of Standards, aujourd'hui NIST) à partir d'un algorithme d'IBM Lucifer.
- Standardisé en 1977, DES a été l'algorithme cryptographique le plus important de ces derniers 30 ans.
- DES est un ***Feistel Cipher*** avec des blocs de 64 bits ***(taille*** nominale).
- La taille effective de la clé est de 56 bits **(Un total de 64 bits** avec 8 bits de parité).
- L'algorithme est constitué de **16 étapes (tours)** avec **16 sous-clés de** 48 bits générées (une clé par étape).



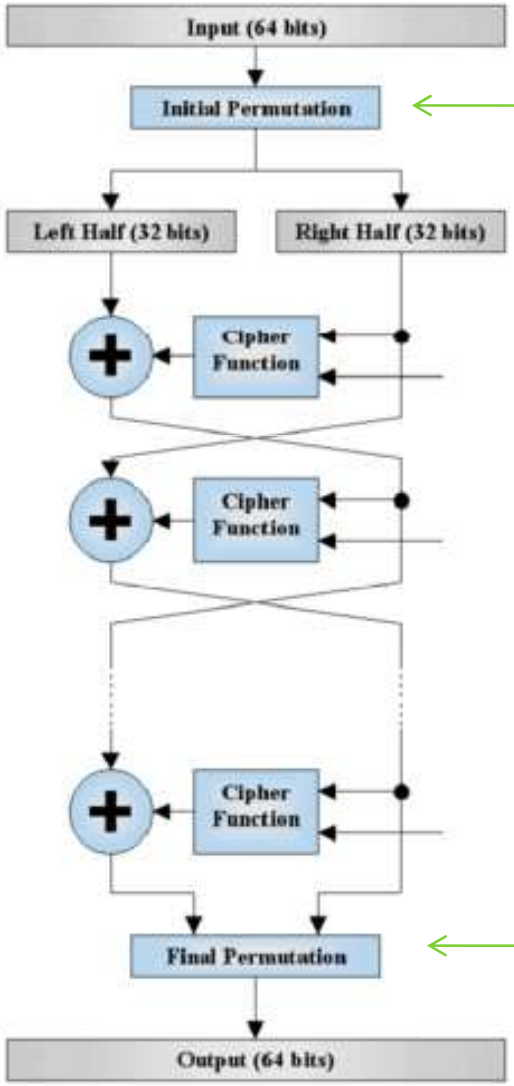


DES Principe de fonctionnement



Réduction de 8 bits (les bits de parité de chaque octet)

Permutations: Initiale & initiale inverse



Permutation Initiale

Permutation inverse (finale)

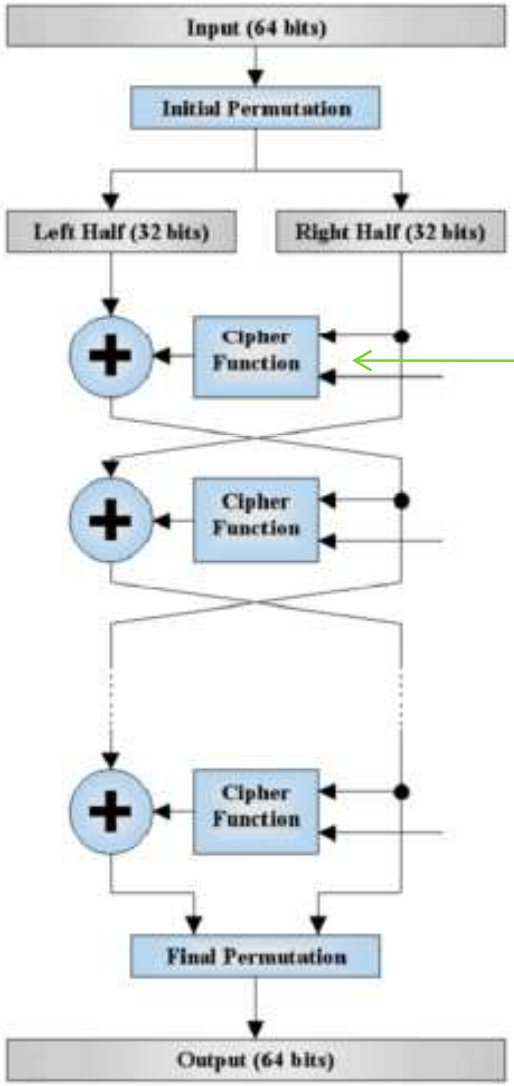
Permutations: Initiale & initiale inverse

- Agissent respectivement au début et à la fin du traitement du bloc et sur l'ensemble des 64 bits
- **Principe** :
  - Mettre le 58ème bit à la position 1, après le 50ème bit, ensuite le 42ème bit, etc …
  - De même pour IP^{-1}

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

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

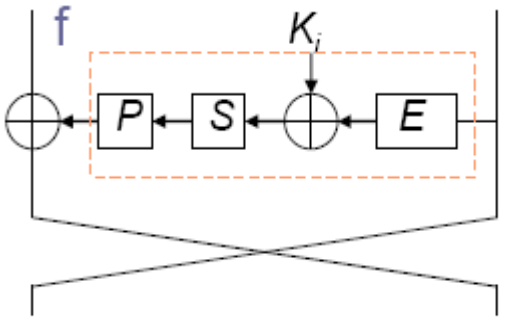
Cipher Function: Schéma de Feistel



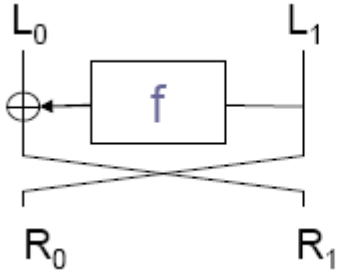
Fonction de cryptage

Cipher Function: Schéma de Feistel

- Appelée aussi la fonction d'étage
- Principe :
  - **E** : expansion de 32 vers 48 bits
  - **Xor** avec une sous clé Ki de 48 bits
  - **S -Box** : boites S (permutations non linéaires) de 48 bits vers 32 bits;
  - **P** : permutation de 32 bits vers 32 bits

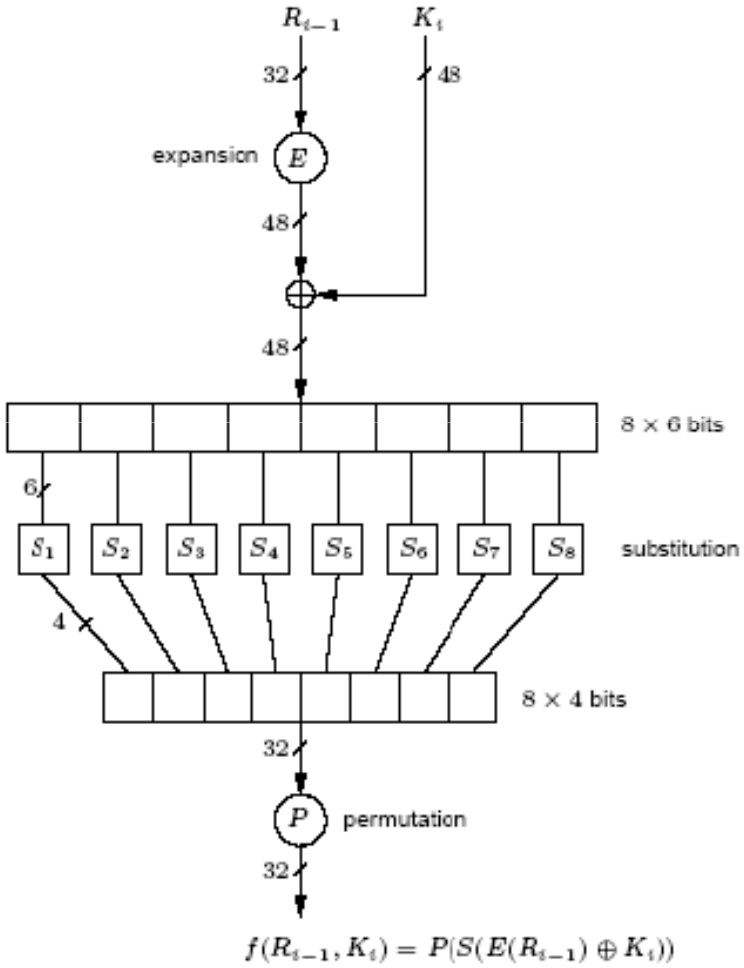


- Le schéma de **Feistel** est une permutation :



- $R_0 = L_1$ et $R_1 = L_0 \oplus f(L_1)$
- Inverse : $L_1 = R_0$ et $L_0 = R_1 \oplus f(L_1)$

Cipher Function: Schéma de Feistel



(Source [Men97])

Figure 7.10: *DES inner function* f.

Cipher Function: Schéma de Feistel

- **Expansion E: Les 32 bits de l'entrée sont transformés en un vecteur de 48 bits en** utilisant la table **E .**
- **La première ligne de cette table indique comment sera** généré le premier sous-bloc de 6 bits: on prendra en premier le 32e bit et après les bits 1,2,3,4,5. Le deuxième sous-bloc commence par le 4e bit ensuite les bits 5,6,7,8,9 et ainsi de suite…

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

Cipher Function: Schéma de Feistel

- **XOR** du vecteur de 48 bits avec la clef $\mathbf{K_i}$ intermédiaire .
- **S-Boxes: On applique 8 S-boxes sur le vecteur de 48 bits résultant du XOR précédent.**
  - Chacune de ces S-boxes prend un sous-bloc de 6 bits et le transforme en un sous-bloc de 4 bits.
  - L'opération s'effectue de la manière suivante:
    - Si on dénote les 6 bits d'input de la S-box comme: $\mathbf{a_1\ a_2\ a_3\ a_4\ a_5\ a_6}$ La sortie est donnée par le contenu de la cellule située dans la **Ligne** $\mathbf{a_1}$ +2 $\mathbf{a_6}$ et la **Colonne** $\mathbf{a_2}$ + 2 $\mathbf{a_3}$ + 4 $\mathbf{a_4}$ + 8 $\mathbf{a_5}$.
    - Exemple: soit un bloc **011111,** la **S-Box 1**le résultat est **0011 , 2 ligne 15 colonne**

S-Box 1: Substitution Box 1

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

Cipher Function: Schéma de Feistel

S-Box 2: Substitution Box 2

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

- **S-Box3, S-Box4, S-Box5, S-Box6, S-Box7, S-Box8.**
- **Permutation P: Le premier bit est** envoyé à la 16ème position, le deuxième à la 7ème position et ainsi de suite

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

Génération des clés

- **Attaques** :
  - Taille de la clé du DES : 56 bits soit 2^{56} essais !
  - **99 : Attaque des laboratoires RSA contre le DES (clé** retrouvée en 22 heures)
    - À l'aide d'une machine dédiée : Deep Crack (250 000 dollars)
    - De 100 000 PCs par calul distribué
- Solutions :
  - Changement de taille de clé : 128 bits minimum
  - **Triple DES** nous met à l'abri de ces attaques *brute force en augmentant l'espace des clés* possibles à $2^{112.}$
  - **97 : Appel d'offre du NIST pour choisir un nouvel** algorithme de chiffrement par blocs pour le 21ème siècle
    - Nom : **AES** choix de **Rijndael en octobre 2000.**

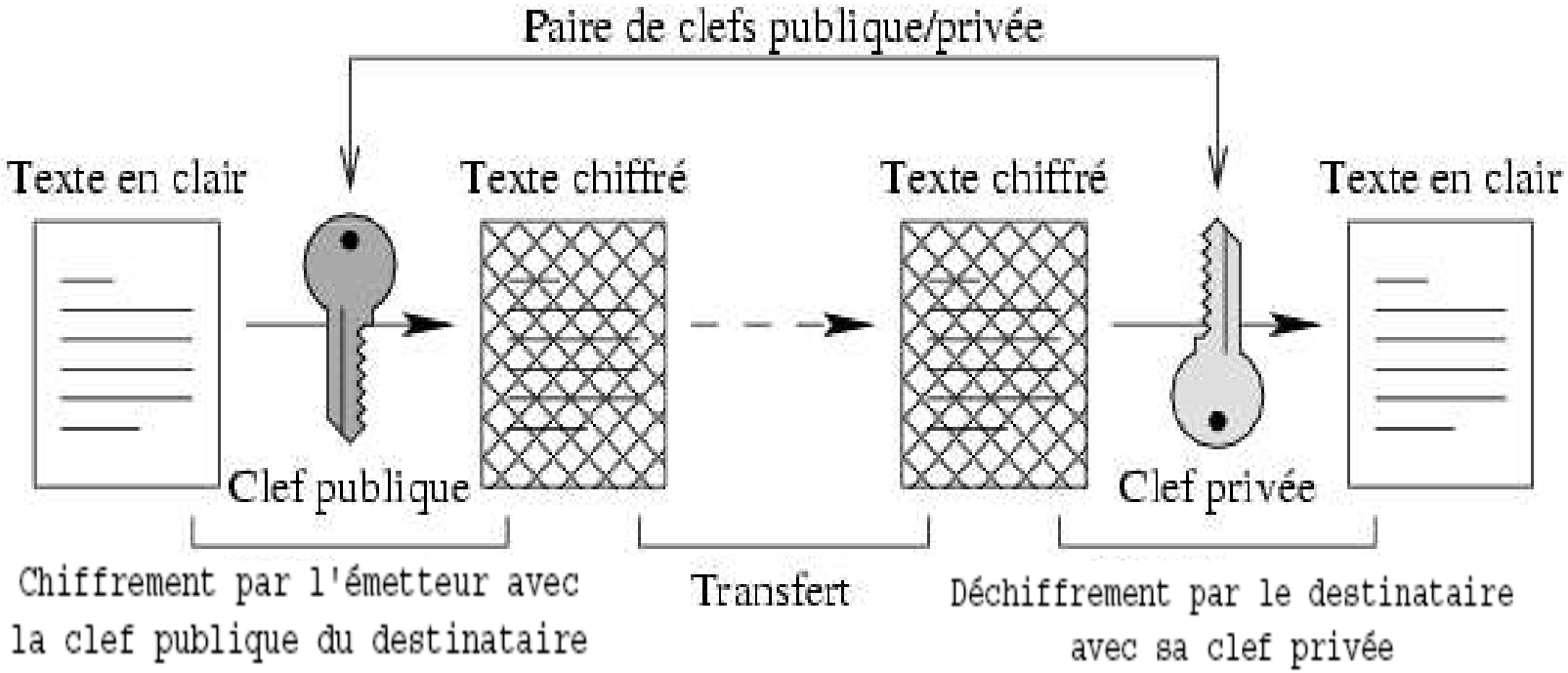
2. Cryptographie asymétrique

Le chiffrement asymétrique

- Dit aussi chiffrement à **clé publique.**
- Limitation des algorithmes à clef secrète:
  - Problématique de l'échange de la clef de chiffrement
    - Etablissement préalable d'un canal sur pour la transmission de la clef.
- Système à clef publique:
  - Clef de chiffrement **e**, rendu publique :
    - Publiée dans un annuaire.
    - Connue par tous.
  - Clef de déchiffrement **d**, gardée secrète:
    - N'est connue que de la personne à qui la paire.
- Chiffrement d'un message à l'aide de la clef publique **e**.
- Déchiffrement à l'aide de la clef privée **d**.

- **Impossibilité pratique** de déterminer la clef privée à partir de la clef publique:
  - **d** non déductible modulo la connaissance de **e.**

Le chiffrement asymétrique



Le chiffrement asymétrique

- **Avantages**
  - Une seule clef publique publiée vers tous les récepteurs
  - n participants: 2*n clés doivent être générées (O(**n**))*
  - Signature digitale → non-répudiation
  - Plus sur qu'un algorithme de cryptage symétrique
- **Inconvénients**
  - Requiert trop d'opérations et donc n'est pas recommandé pour de grande quantité d'information.
    - Utilisé pour l'échange de clefs de session (**voir ultérieurement dans le cours**)
  - Vulnérable aux attaques de type **MIME** (Man In The Middle)
    - Problème d'authentification d'un destinataire.
    - Il n'y a aucune certitude à savoir qui est le véritable propriétaire de la clef publique
    - Un attaquant peut fausser une personne et lui entraine d'utiliser sa clef publique

Le cryptosystème RSA

- L'implémentation fut achevée en 1978 par **Rivest, Shamir et Adleman.**
- Fondé sur la difficulté de la factorisation des grands nombres
- **Principe**
  - On choisit **p** et **q** deux nombre entiers premier entre eux.
  - On calcule **n=p*q**.
  - On choisit un entier **e** premier avec ϕ **(n)=(p-1) *(q-1)**.
  - L'entier d tel que **e*d = 1 mod (p-1)(q-1)**
  - Le couple **(n,e)** représente **la clef publique**
  - L'entier **d** représente **la clef privée.**
- **Chiffrement RSA**:
  - Découper le message en plusieurs blocs de taille égale B_i (**< n**).
  - Un bloc B_i est chiffré par la formule $C_i = B_i^e \bmod n$
- **Déchiffrement RSA**:
  - Chacun des blocs **C** du message chiffré sera déchiffré par la formule $B_i = C_i^d \bmod n$.

Le cryptosystème RSA

- **Exemple** :
  - Nous allons essayer de crypter le message **STOP.**
  - Prenons **p= 43** et **q=59.** (p et q sont premier entre eux)
  - **n**= **2537**.
  - Choix de **e:**
    - (p-1)*(q-1)= 42*58=2436
    - Prenons **e=13** il vérifie bien la condition PGCD(13, 2436)=1
  - Choix de **d:**
    - Chercher **d** tel que **(p-1)(q-1)** divise **(e*d)-1**
      - En pratique : chercher k € IN / **d= [k*(p-1)(q-1)+1]/e**
      - Chercher **d** tel que 2436 divise (13*d)-1 : [k*2436+1]/13
      - k=2, k=3, k=4, **k=5**
    - **d= 937.**
- Pour notre message on prend la positon de chaque lettre dans l'alphabet -1
  - ST OP=1819 1415
  - M_1= 1819
  - M_2 =1415 (taille de (M)=4 < n=2537)

Le cryptosystème RSA

- **Chiffrement** :
  - **$C = B^e$ mod n** (dans notre cas le bloc c'est le message)
  - 1819^{13} **mod** (2537) = 2081 et 1415^{13} **mod** (2537)= 2182.
  - Le message à envoyer **2081 2182**.
- **Déchiffrement** :
  - Le message reçu est **2081 2182**
  - **$B = C^d$ mod n**.
  - On calcule **2081^{937} mod** (2537)=1819 et **2182^{937} mod** (2537)=1415

- **Exercice :** déchiffrer le message suivant 0981 0461 avec les paramètres de l'exemple précédent
- **En pratique**
  - p et q doivent être grand (100 chiffres décimaux)
  - Taille minimum préconisé :
    - Au moins 768 bits
    - 1024 bits conseillé

Le cryptosystème RSA

- Le problème difficile sur lequel repose RSA : la factorisation des grands nombres
  - Il est très difficile de trouver p et q / n=p.q en ne connaissant que n
    - (pas d'algorithme en temps polynomiale)
  - ou alors, si c'est possible, les cryptanalystes qui ont trouvé la méthode la gardent secrète
- Nouveau record en 2005 : RSA-200 digits (663 bits)
- D'autres algorithmes

Algorithmes de chiffrement	Concepteur (s)	Taille de la clé	Taille du bloc	Commentaires
RSA (Rivest, Shamir, Adelman) (Par bloc)	Rivest, Shamir, et Adelman (1978)	Variable, 512 bits en général	Variable et < taille des clés	Système le plus utilisé Nécessite une licence
Diffie-Hellman	Whitfield Diffie, Martin Hellman (1976)			Système de cryptage le moins récent et toujours en exploitation
El Gamal	El Gamal (1985)			Variante de Diffie-Hellman, mais nécessitant l'envoi du double d'information

La cryptanalyse

- Il existe de nombreuses méthodes d'attaque parfois extrêmement sophistiquées.
  - **Force brute :**
    - Essayer toutes les clés possibles. Principal danger : l'augmentation de la puissance des machines. Parades : augmenter la longueur des clé, choisir des algorithmes coûteux
    - En 97 : 3h pour casser une clé de 40 bits
  - **Analyse statistique :**
    - basée sur les propriétés des messages en clair (exemple : le «e» représente 14.5 % des caractères utilisés dans un texte en français).
    - Il existe des tables statistiques pour des motifs de plusieurs lettres.
  - **Attaque à texte en clair connu:**
    - L'attaquant connaît un message en clair et son équivalent chiffré. Il tente d'en déduire la clé.
  - **Attaque de l'algorithme:**
    - Par exemple, pour les algorithmes qui génèrent une clé secrète aléatoirement, il arrive que l'aléa ne soit pas parfait et donc reproductible par l'attaquant.
- Cryptanalyse des algorithmes asymétriques concentrée sur la résolution ou la réduction de la complexité des fonctions inveres:
  - Exemple : en 2005 une équipe chinoise à annoncé qu'elle avait réduit la complexité de **RSA** de 2^{80} à 2^{69} opérations élémentaires.

3. Le hachage cryptographique

Les fonctions de hachage

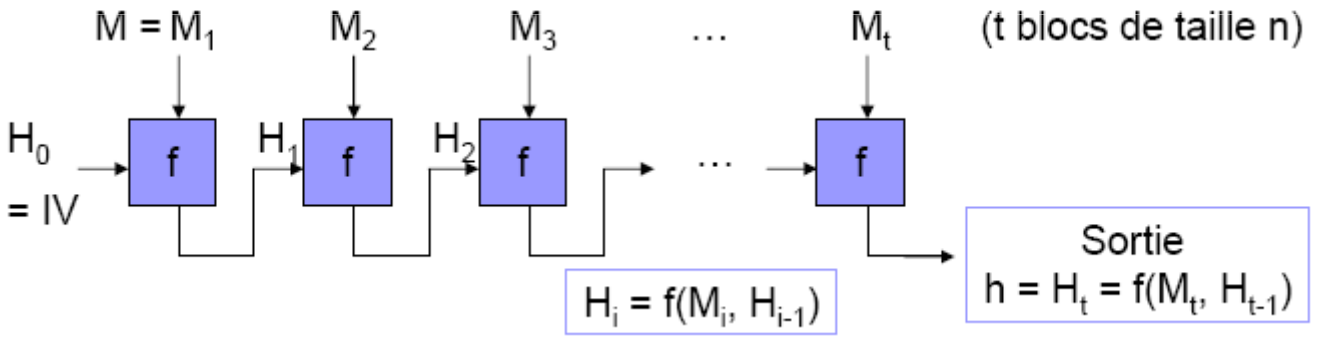
- **Fonction de hachage = fonction de condensation**
  - Convertir une chaîne de longueur quelconque en une chaîne de taille inférieur, et généralement fixe appelée empreinte (digest), ou condensé de la chaîne initiale.
  - **avec une probabilité très forte que deux fichiers différents donnent deux empreintes différentes**
- La plupart des fonctions de hachage à sens unique sans collision sont construites par itération d'une fonction de compression:
  - *M est décomposé en n blocs m1, m2, …, mn,*
  - Une fonction de compression ***f*** est appliquée à chaque bloc, et au résultat de la compression du bloc précédent.
  - L'empreinte ***H(M) = résultat de la dernière compression***.

Les fonctions de hachage

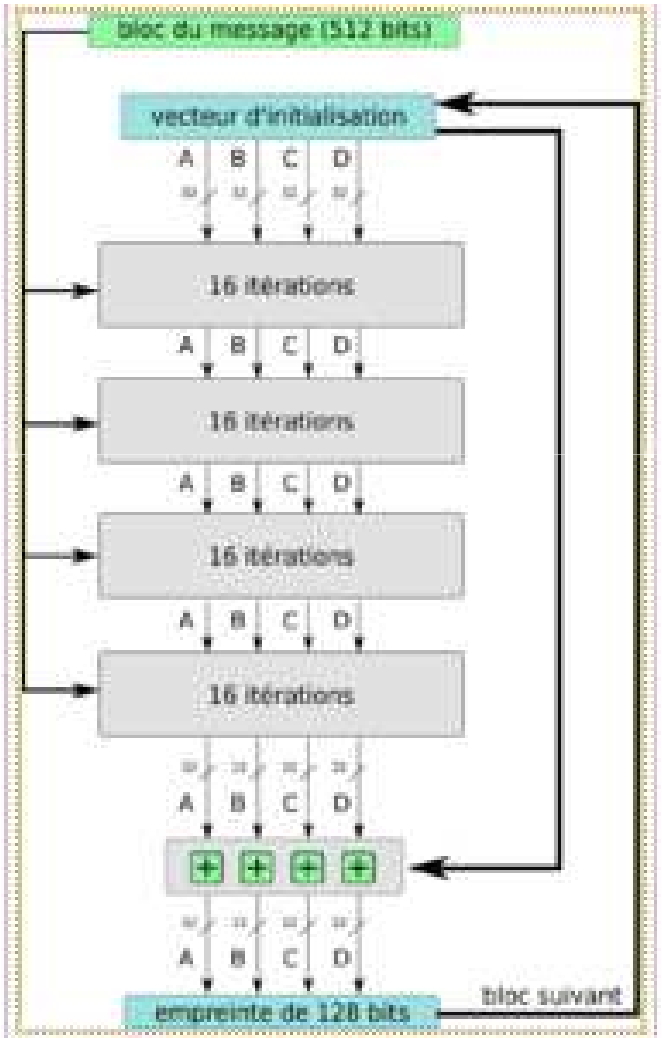
- **Propriétés**
  - A sens unique: il est difficile d'engendrer la chaîne initiale à partir de l'empreinte.
  - Sans collision: il est impossible de trouver deux messages ayant la même empreinte (**paradoxe de l'anniversaire**).
  - La moindre modification des données produit une valeur de hachage différente
- **Utilités** : Permet de garantir l'intégrité
  - Téléchargement de packages sur internet.
  - Vérification de l'intégrité des paquets par calcul du hash.
  - Utilisation avec une signature numérique (voir plus loin)
  - Calcul de mot de passe

Paradoxe des anniversaires

- Cas du hachage:
  - Si le haché fait 128 bits, alors il faut essayer environ $\mathbf{2^{64}}$ messages pour obtenir une collision.
- Méthodes de construction
  - Construite à partir d'une fonction de compression f



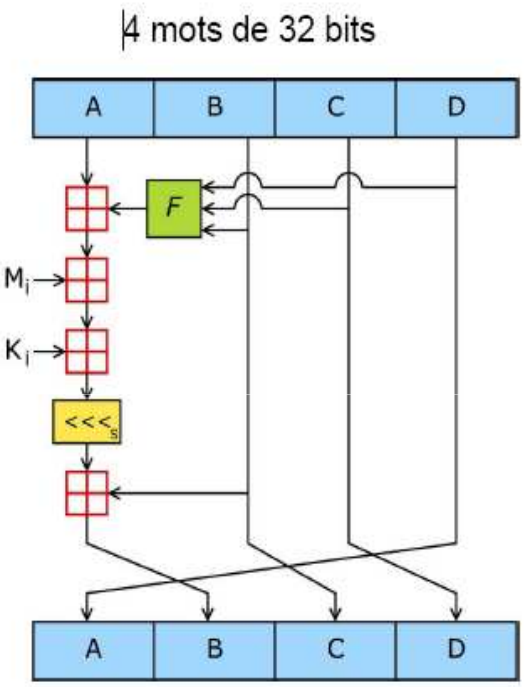
Exemple Message Digest 5 (MD5)



Vue générale de MD5

Exemple Message Digest 5 (MD5)

- MD5 : entrée de 512 bits -> hash de 128 bits.
  - Le message est divisé en blocs de 512 bits, on applique un remplissage de manière à avoir un message dont la longueur est un multiple de 512.
- Une opération de MD5.
  - MD5 comprend 64 sous blocs (d'un **bloc M_i de taille 512bits)** de ce type, groupés en quatre tours de 16 opérations. *F* est une fonction non-linéaire, qui varie selon le tour.
  - *A*, *B*, *C* et *D*. Ils sont initialisés au début avec des constantes
  - ***M_i*** symbolise un bloc de 32 bits provenant du message à hacher



- $[<<<]_s$ est une rotation de *s* bits vers la gauche, *s* varie pour chaque opération.
- [+] symbolise l'addition modulo 2^{32}.
- $\oplus, \wedge, \vee, \neg$ symbolisent respectivement les opérations booléennes XOR, AND, OR et NOT.

$$F(B, C, D) = (B \wedge C) \vee (\neg B \wedge D)$$

Les algorithmes standards

Fonctions de hachage	Concepteur (s)	Taille de l'empreinte	Commentaires
MD5 (*Message Digest 5*)	Ronald Rivest (1991)	128 bits	Successeur de MD4 Présente des problèmes de collision qui ont affecté son expansion
SHA (*Secure Hash Algorithm*)	NSA (National Security Agency), et normalisé par le NIST	160 bits	SHA-1 révision publiée en 1994, empreintes de 160 bits sur documents de au - 264 bits de longueur considérée plus sur que MD5 SHA-2 (octobre 2000) agrandit la taille de l'empreinte dans plusieurs versions en 224, 256, 384, et 512 bits

Attaques des algos de hashage

- **Attaques**
  - En **2004**, une équipe chinoise découvre des collisions complètes sur MD5
  - temps nécessaires pour trouver deux messages M et M’ fournissant le même haché h :
    - MD4 : 15 minutes
    - MD5 : 8 heures sur un PC à 1,6 GHz : MD5 n'est donc plus considéré comme sûr au sens cryptographique
- **Solutions** :
  - On suggère maintenant d'utiliser plutôt des algorithmes tels que SHA1, SHA-256.
  - On parle d’un éventuel appel d’offre du NIST

Exemple de collisions MD5

Example: MD5 collision with the standard IV

IV according to [2]:

```
  context->state[0] = 0X67452301;
  context->state[1] = 0xefcdab89;
  context->state[2] = 0x98badcfe;
  context->state[3] = 0X10325476;
```

First message:

```
0xA6,0x64,0xEA,0xB8,0x89,0x04,0xC2,0xAC,
0x48,0x43,0x41,0x0E,0x0A,0x63,0x42,0x54,
0x16,0x60,0x6C,0x81,0x44,0x2D,0xD6,0x8D,
0x40,0x04,0x58,0x3E,0xB8,0xFB,0x7F,0x89,
0x55,0xAD,0x34,0x06,0x09,0xF4,0xB3,0x02,
0x83,0xE4,0x88,0x83,0x25,0x71,0x41,0x5A,
0x08,0x51,0x25,0xE8,0xF7,0xCD,0xC9,0x9F,
0xD9,0x1D,0xBD,0xF2,0x80,0x37,0x3C,0x5B,
0x97,0x9E,0xBD,0xB4,0x0E,0x2A,0x6E,0x17,
0xA6,0x23,0x57,0x24,0xD1,0xDF,0x41,0xB4,
0x46,0x73,0xF9,0x96,0xF1,0x62,0x4A,0xDD,
0x10,0x29,0x31,0x67,0xD0,0x09,0xB1,0x8F,
0x75,0xA7,0x7F,0x79,0x30,0xD9,0x5C,0xEB,
0x02,0xE8,0xAD,0xBA,0x7A,0xC8,0x55,0x5C,
0xED,0x74,0xCA,0xDD,0x5F,0xC9,0x93,0x6D,
0xB1,0x9B,0x4A,0xD8,0x35,0xCC,0x67,0xE3.
```

Second message:

```
0xA6,0x64,0xEA,0xB8,0x89,0x04,0xC2,0xAC,
0x48,0x43,0x41,0x0E,0x0A,0x63,0x42,0x54,
0x16,0x60,0x6C,0x01,0x44,0x2D,0xD6,0x8D,
0x40,0x04,0x58,0x3E,0xB8,0xFB,0x7F,0x89,
0x55,0xAD,0x34,0x06,0x09,0xF4,0xB3,0x02,
0x83,0xE4,0x88,0x83,0x25,0xF1,0x41,0x5A,
0x08,0x51,0x25,0xE8,0xF7,0xCD,0xC9,0x9F,
0xD9,0x1D,0xBD,0x72,0x80,0x37,0x3C,0x5B,
0x97,0x9E,0xBD,0xB4,0x0E,0x2A,0x6E,0x17,
0xA6,0x23,0x57,0x24,0xD1,0xDF,0x41,0xB4,
0x46,0x73,0xF9,0x16,0xF1,0x62,0x4A,0xDD,
0x10,0x29,0x31,0x67,0xD0,0x09,0xB1,0x8F,
0x75,0xA7,0x7F,0x79,0x30,0xD9,0x5C,0xEB,
0x02,0xE8,0xAD,0xBA,0x7A,0x48,0x55,0x5C,
0xED,0x74,0xCA,0xDD,0x5F,0xC9,0x93,0x6D,
0xB1,0x9B,0x4A,0x58,0x35,0xCC,0x67,0xE3.
```

Common MD5 hash:

```
0x2B,0xA3,0xBE,0x5A,0xA5,0x41,0x00,0x6B,
0x62,0x37,0x01,0x11,0x28,0x2D,0x19,0xF5.
```

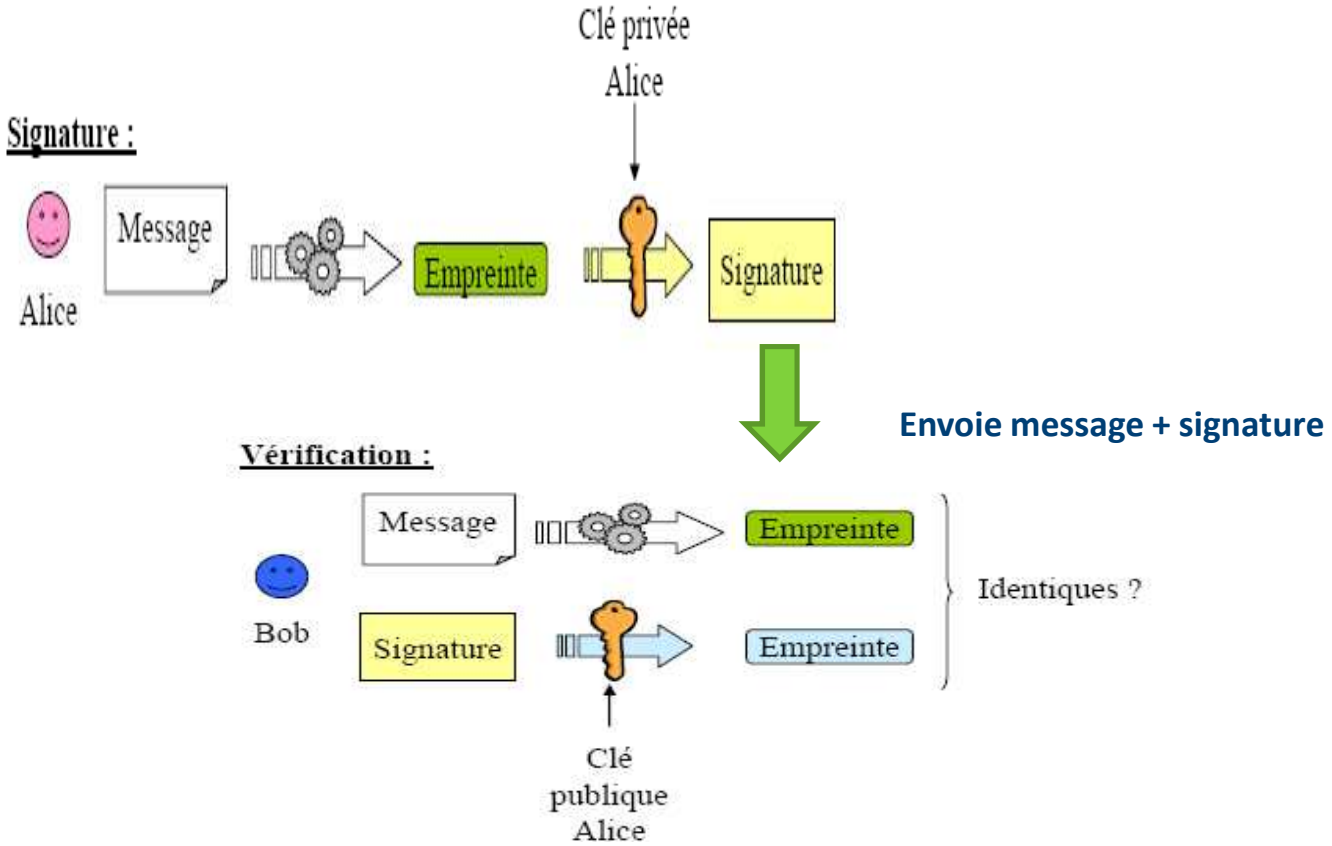
4. La signature numérique

La signature numérique

- La cryptage asymétrique possède deux inconvénients importants:
  - Il est gourmand en puissance de calcul et donc assez lent par rapport au cryptage symétrique.
  - Vulnérable à des attaques de type **MiM** (Man in the Middle):
    - Le pirate s'insère entre l'émetteur et le récepteur
    - Lorsque l'émetteur demande la clé publique du récepteur → le pirate envoie sa propre clé
    - L'émetteur crypte le message et l'envoie.
    - Intercepté par le pirate, il le décrypte avec sa propre clé privée.
    - Après il redirige le message vers le récepteur en le cryptant de nouveau avec la clé publique du récepteur .
- Les solutions:
  - Pour le premier problème: le cryptage asymétrique est généralement utilisé au début de la communication dont le but d'échanger secrètement la **clé publique**.
  - Pour le deuxième : l'émetteur doit s'assurer que la **clé publique** qui lui a été fourni est bien celle du récepteur.
    - N'accepter la clé que si elle est signée électroniquement (soit par le récepteur lui-même ou un **tiers de confiance**)

La signature numérique

- **Signature numérique (ISO 7498-2) : «** données ajoutées à une unité de données, ou transformation cryptographique d'une unité de données, permettant à un destinataire de prouver la source et l'intégrité de l'unité de données **»**.



La signature numérique

- Si les deux empreintes :
  - Calculée à partir du message reçu (H1).
  - Décryptée par la clé publique de l'émetteur (H2)
- Sont identiques la signature est **valide**.

- Grâce à ce mécanisme :
  - On peut vérifier est bien **signé** par la personne dont ont connaît la clé publique ⇒ **c'est l'authenticité.**
  - Le document n'a pas été modifié ⇒ **c'est l'intégrité.**
  - seul le signataire a pu produire ces données ⇒ **c'est la non-répudiation.**

- Pour vérifier une signature, il faut donc avoir:
  - Les deux algorithmes (hachage cryptographique et chiffrement)

- Remarque : Bien sur le document signé peut lui-même être crypté avec la clé publique du récepteur (ou avec un cryptage symétrique).

La signature numérique RSA

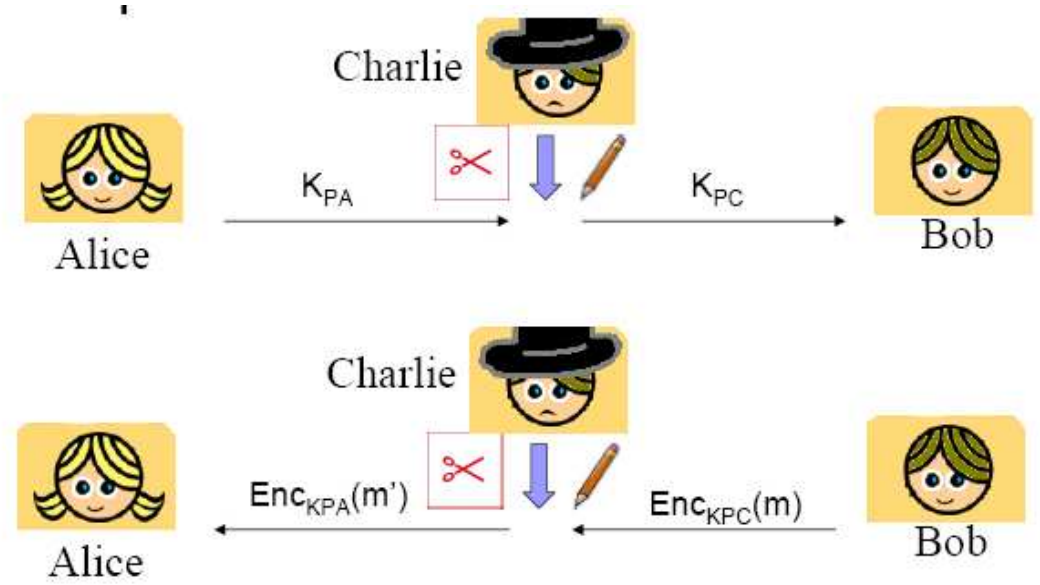
- La Signature RSA
  - Publique : **n** et **e**,
  - Secret : exposant Alice **d**
- Alice signe le message m en calculant :
  - **$S = M^d \bmod n$**
  - Bob vérifie en calculant : **$M = S^e \bmod n$**

Algorithme de Signature numérique	Concepteur (s)	Algorithme de chiffrement à clé publique	Fonction de hachage
DSA (Digital Signature Algorithm) DSS (Digital Signature Standard)	NIST(1991)	El Gamal	SHA-1
RSA Digital Signature	Rivest, Shamir, et Adelman	RSA	SHA-1 ou MD5

5. Le Certificat numérique

Le certificat numérique

- **Pourquoi ?**
  - Prenons toujours l'exemple d'une attaque MIME

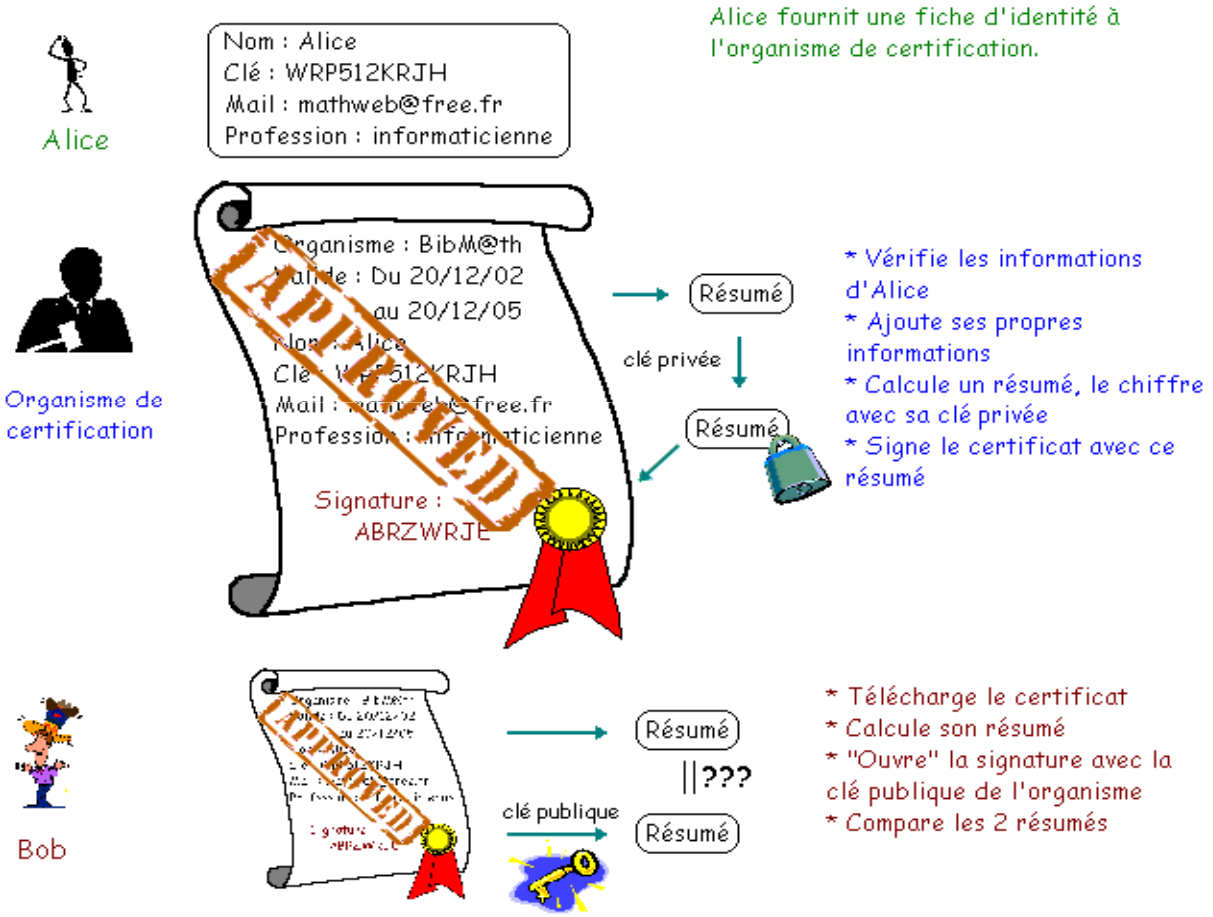


- La signature est « **vérifiable** », mais elle n'authentifie pas le signataire
- Garantir que la clef publique d'Alice est bien la clé publique d'Alice
  - Garantir **l'authentification**
- Annuaire de clefs publiques garanti par une **autorité** qui signe l'identité d'Alice et la clé publique de Alice

Le certificat numérique

- **Certificat numérique :**
  - un certificat numérique est un document crypté signé contenant des informations sur l'identification d'un serveur ou d'un utilisateur.
  - Le certificat numérique ou électronique est un lien entre l'entité physique et l'entité numérique (Virtuel).
  - L'autorité de certification fait foi de tiers de confiance et atteste du lien entre l'identité physique et l'entité numérique
- **Délivré** et **certifié** par une autorité de certification, **Certificate Authority (CA),**
- comporte généralement, et selon le format de sa norme:
  - un numéro d'identification du certificat : N° de série
  - nom du détenteur de la clé publique: Distinguished Name
  - une clé publique de chiffrement
  - un délai de validité (entre six mois et un an)
  - une catégorie
  - raison sociale de l'organisme de certification.

Le certificat numérique



Certification numérique d'une clé publique

Types de certificats numériques

- **Certificats de client SSL:** Utilisés pour identifier des client auprès de serveurs via SSL (authentification client):
  - Généralement, l'identité du client = identité d'un être humain.
  - Utilisation : authentification des clients pour accéder à leurs comptes.
- **Certificats de serveur SSL:** Utilisé pour identifier les serveurs auprès des client via SSL (authentification serveur):
  - L'authentification serveur est obligatoire lors de l'établissement d'une connexion SSL chiffrée.
  - **Utilisation:** Les sites internet de commerce électronique pour établir une session SSL chiffrée et assure les clients qu'ils traitent avec un site de confiance.

- **Certificats S/MIME (Secure Multipurpose Internet Mail Extensions** : Utilisés pour **signer** et **chiffrer** les courriers

- **Certificats de signature d'objet:** Utilisés pour identifier les signataires de code Java, de scripts JavaScript, ou d'autres fichiers signés

- **Certificats d'AC:** Utilisés pour identifier les autorités de certification (AC). Les logiciels client et serveur utilisent les certificats d'AC pour déterminer quels autres certifications peuvent être de confiance

Le certificat numérique X509

- X.509: Norme de description des certificats (**RFC 2459**)
- La norme X.509 version 3 en 1996.
- Chaque certificat contient les informations suivantes:
  - Numéro de version de X.509 utilisée
  - Numéro de série unique du certificat attribué par l'émetteur
  - Algorithme utilisé pour signer le certificat
  - Distinguished Name de l'émetteur du certificat
  - Période de validité du certificat
  - Distinguished Name du sujet
  - Clé publique du sujet et algorithme pour lequel elle est valable
  - Extensions éventuelles
  - Une partie signature
    - L'algorithme utilisé pour signer le certificat
    - La signature de ce certificat, chiffrée avec la clé privée de l'émetteur
- Les ***Distinguished Name*** *(DN) sont de la forme*
  - nom canonique (CN), organisme (O), ville (L), code pays (C), etc.

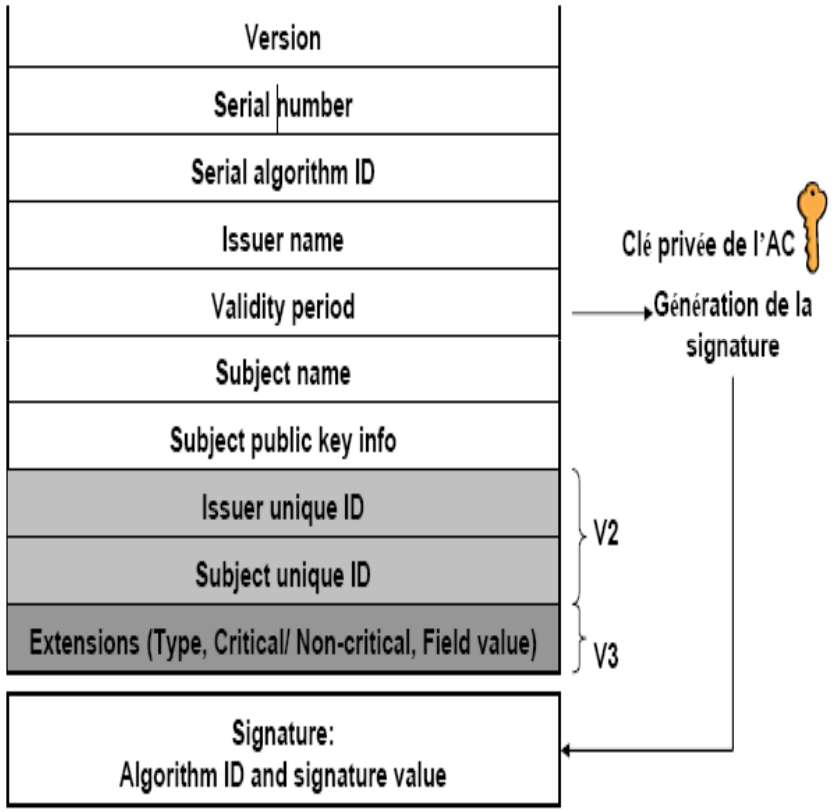
Le certificat numérique X509

- Un certificat X509 permet de :
  - Prouver l'identité d'une personne au même titre qu'une carte d'identité, dans le cadre fixé par l'autorité de certification qui l'a validé ;
  - Pour une application, assurer que celle-ci n'a pas été détournée de ses fonctions.
  - Pour un site il offre la garantie lors d'un accès vers celui-ci que l'on est bien sur le site auquel on veut accéder.

- Largement adopté par de nombreuses solutions du marché:
  - **IPSec** (Internet Protocol Security).
  - **SSL**(Secure Socket Layers).
  - **S/MIME** (Secure Multipurpose Internet Mail Extensions)

Le certificat numérique X509

- **Version** : Indique à quelle version de X.509 correspond ce certificat (v1, v2, v3).
- **Serial number** : Numéro de série du certificat (propre à chaque autorité de certification).
- **Signature Algorithm ID** : Identifiant du type de signature utilisée.
- **Issuer Name** : **Distinguished Name (DN)** de l'autorité de certification qui a émis ce certificat.
- **Validity period** : Période de validité.
- **Subject Name** : *Distinguished Name (DN)* du détenteur de la clef publique.
- **Subject public key info** : Infos sur la clef publique de ce certificat.
- **Issuer Unique ID / Subject Unique ID** : Extensions optionnelles introduites avec la version 2 de X.509.
- **Extensions** : Extensions génériques optionnelles, introduites avec la version 3 de X.509.
- **Signature** : Signature numérique de la CA sur l'ensemble des champs précédents. (Algorithme de signature, valeur de la signature)



Certificats X509 sous Windows

Certificats X509 sous Linux

```
Certificate:
Data:
 Version: v3 (0x2)
 Serial Number: 3 (0x3)
 Signature Algorithm: PKCS #1 MD5 With RSA Encryption
 Issuer: OU=Ace Certificate Authority, O=Ace Industry, C=US
 Validity:
  Not Before: Fri Oct 17 18:36:25 1997
  Not  After: Sun Oct 17 18:36:25 1999
 Subject: CN=Jane Doe, OU=Finance, O=Ace Industry, C=US
 Subject Public Key Info:
  Algorithm: PKCS #1 RSA Encryption
  Public Key:
   Modulus:
    00:ca:fa:79:98:8f:19:f8:d7:de:e4:49:80:48:e6:2a:2a:86:
    ed:27:40:4d:86:b3:05:c0:01:bb:50:15:c9:de:dc:85:19:22:
    43:7d:45:6d:71:4e:17:3d:f0:36:4b:5b:7f:a8:51:a3:a1:00:
    98:ce:7f:47:50:2c:93:36:7c:01:6e:cb:89:06:41:72:b5:e9:
    73:49:38:76:ef:b6:8f:ac:49:bb:63:0f:9b:ff:16:2a:e3:0e:
    9d:3b:af:ce:9a:3e:48:65:de:96:61:d5:0a:11:2a:a2:80:b0:
    7d:d8:99:cb:0c:99:34:c9:ab:25:06:a8:31:ad:8c:4b:aa:54:
    91:f4:15
   Public Exponent: 65537 (0x10001)
  Extensions:
   Identifier: Certificate Type
    Critical: no
    Certified Usage:
     SSL Client
   Identifier: Authority Key Identifier
    Critical: no
    Key Identifier:
     f2:f2:06:59:90:18:47:51:f5:89:33:5a:31:7a:e6:5c:fb:36:
     26:c9
 Signature:
  Algorithm: PKCS #1 MD5 With RSA Encryption
  Signature:
   6d:23:af:f3:d3:b6:7a:df:90:df:cd:7e:18:6c:01:69:8e:54:65:fc:06:
   30:43:34:d1:63:1f:06:7d:c3:40:a8:2a:82:c1:a4:83:2a:fb:2e:8f:fb:
   f0:6d:ff:75:a3:78:f7:52:47:46:62:97:1d:d9:c6:11:0a:02:a2:e0:cc:
   2a:75:6c:8b:b6:9b:87:00:7d:7c:84:76:79:ba:f8:b4:d2:62:58:c3:c5:
   b6:c1:43:ac:63:44:42:fd:af:c8:0f:2f:38:85:6d:d6:59:e8:41:42:a5:
   4a:e5:26:38:ff:32:78:a1:38:f1:ed:dc:0d:31:d1:b0:6d:67:e9:46:a8:
   d:c4
```

Confiance dans un certificat

- Les **CA** (autorités de certification) valident des identités et émettent des certificats
  - Ils peuvent être indépendants ou gérer leur propres mécanisme de certification

- Clients ou serveurs manipulent des certificats
  - Ils gèrent des listes de certificats des autorités de certification en qui ils ont confiance (***Trusted CA certificates)***

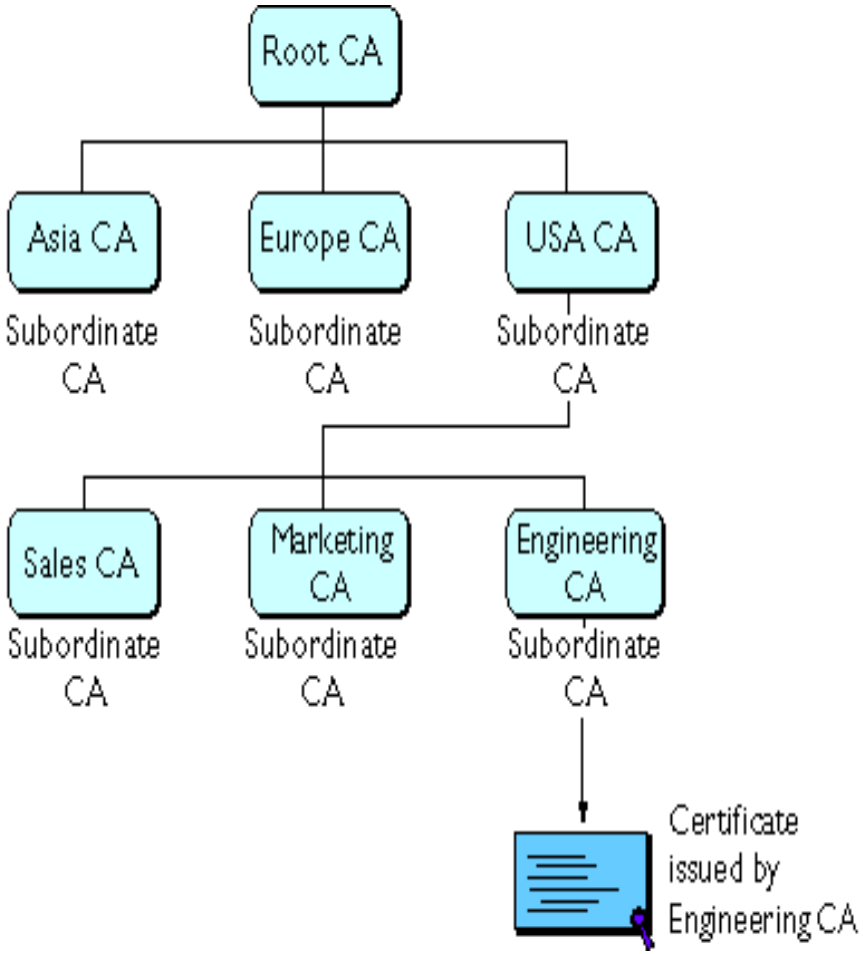
- Organisation des CA de manière **hiérarchique**
  - La racine de la hiérarchie s'auto-certifie
    - On suppose qu'elle sera « de confiance » pour l'utilisateur

- Chaque niveau en dessous est certifié par le CA du niveau au dessus

- **Utilité** : Dans les grandes organisations, il peut être judicieux de déléguer la responsabilité de l'émission des certificats à plusieurs autorités de certification

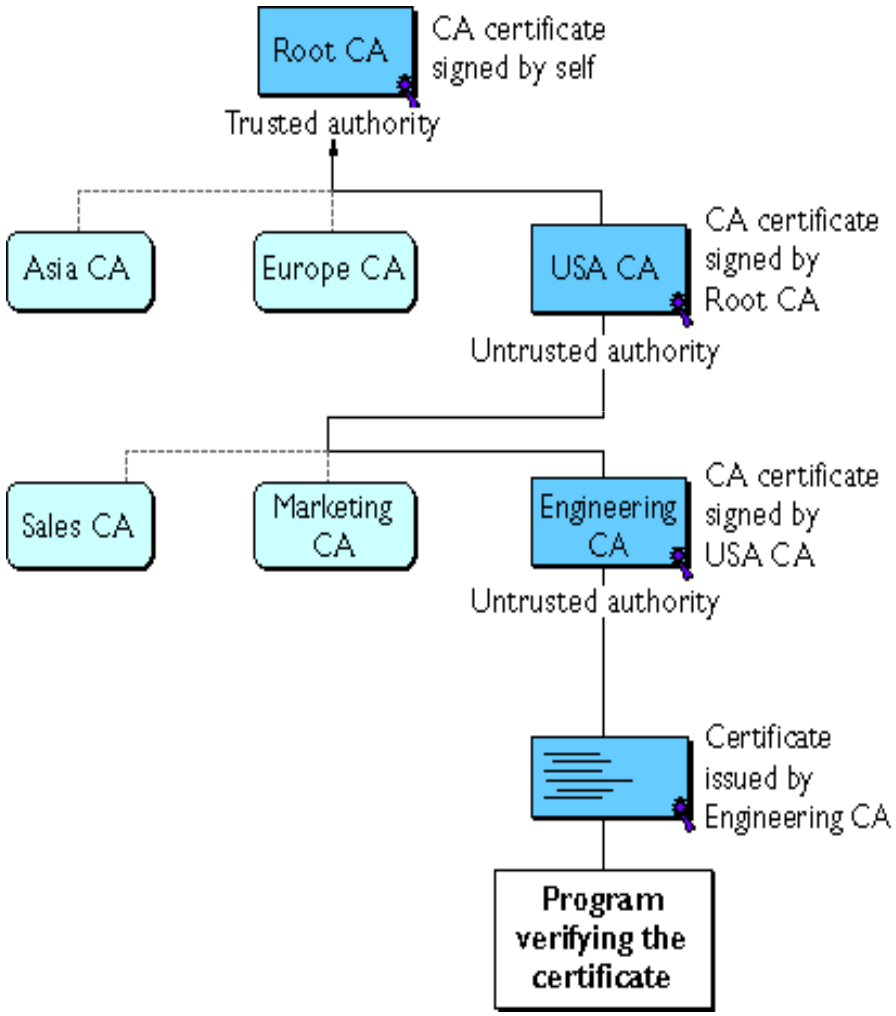
Organisation hiérarchique

- Un certificat émis par **Root CA** est certifié par **Root CA** (auto signature).
- Un certificat émis par USA CA est certifié par **RootCA**
- Un certificat émis par Marketing CA est certifié par **USACA**
- **Etc…**
- Ces hiérarchies sont représentées par des **chaînes de certificats**
- Chaque niveau en dessous est certifié par le CA du niveau au dessus.



Chaîne de certificats (1)

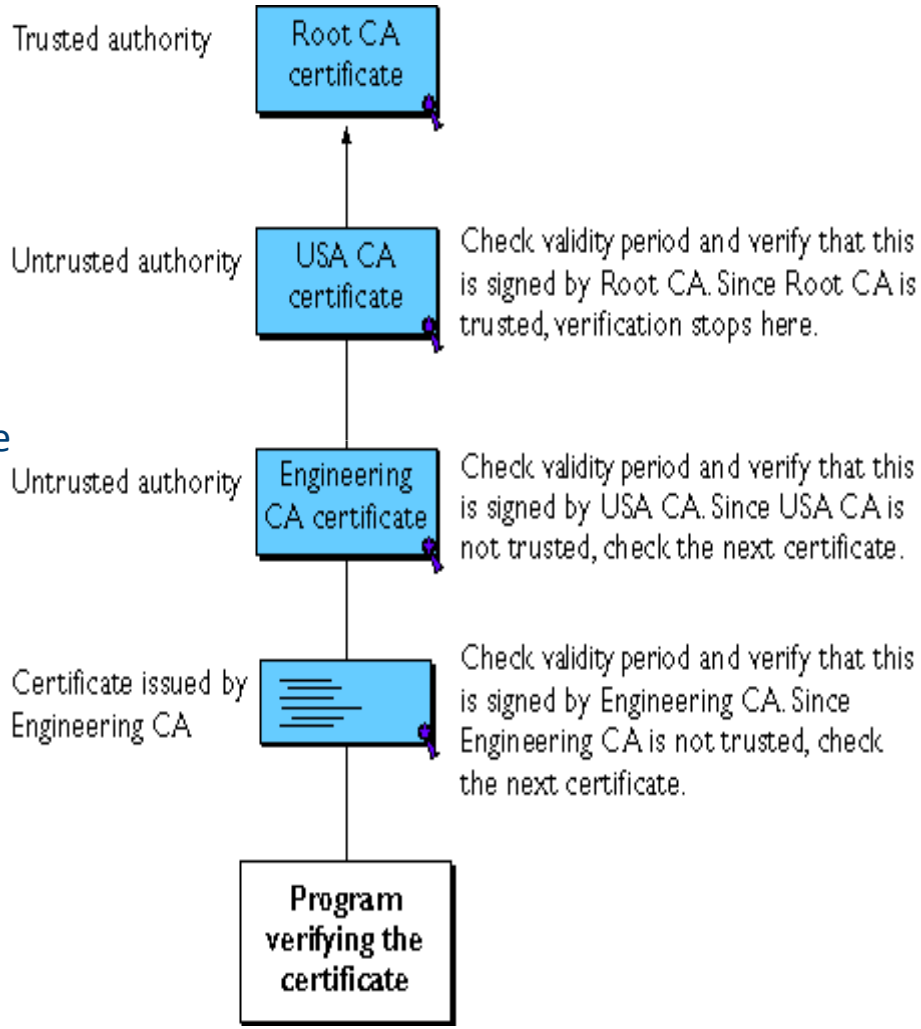
- **Une *chaîne de certificats* est une série de certificats émis par des AC successives.**
- Une chaîne de certificats trace le chemin des certificats d'une branche de la hiérarchie jusqu'à la racine
- Chaque certificat est suivi par le certificat de son émetteur.
- Chaque certificat contient le nom (DN) de son émetteur, qui est le sujet du certificat suivant dans la chaîne
- Chaque certificat est signé avec la clef privée de son émetteur. Cette signature peut être vérifiée avec la clef publique située dans le certificat de l'émetteur, qui est le prochain dans la chaîne



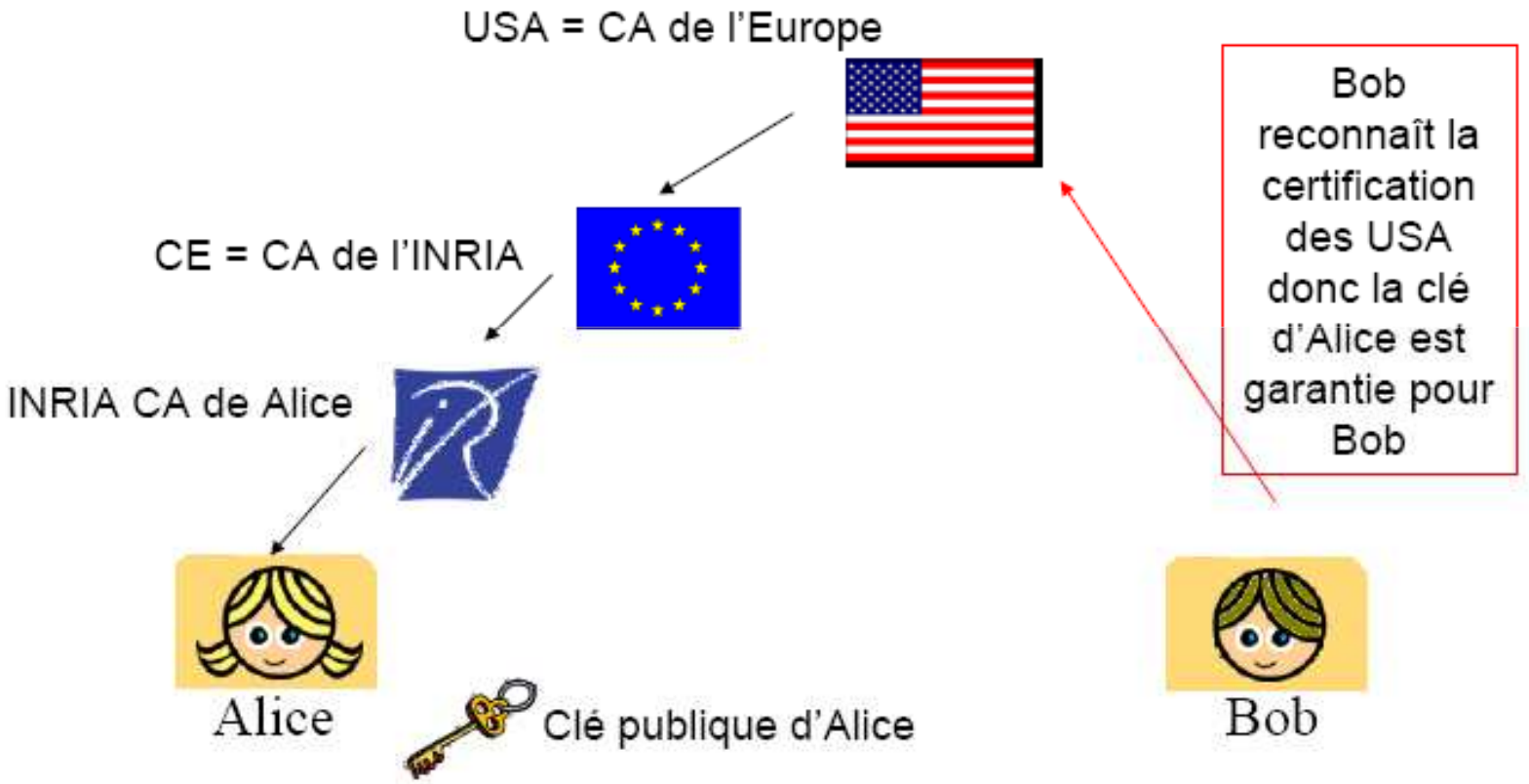
Chaîne de certificats (2)

Processus de vérification:

- La période de validité du certificat est vérifiée par rapport à la date actuelle fournie par l'horloge système du vérificateur.
- Le certificat de l'émetteur est localisé. La source peut être une base de certificats locale du vérificateur (du client ou du serveur) ou une chaîne de certificats fournit par le sujet (par exemple, par une connexion SSL).
- La signature du certificat est vérifiée à l'aide de la clef publique du certificat de l'émetteur.
- Si le certificat de l'émetteur est présent dans les certificats de confiance du vérificateur, la vérification s'arrête avec succès à cette étape.



Chaîne de certificats (3)



Public Key Infrastructure

- La sécurité du processus de chiffrement repose sur la sécurité et la confidentialité des clefs utilisées.
  - La **durée de vie** d'une clé dépend de son utilisation.
  - Il est conseillé de changer cette clé périodiquement.
  - De plus, il faut éviter toute modification fréquente qui rend la gestion difficile des clés.
  - $\Rightarrow$ utiliser une clé à usage unique à chaque session de travail.

- Des systèmes de gestion de clés sont envisageable et permettent de faire:

  - La génération d'une clé en fonction des besoins et des systèmes de chiffrement.
  - La distribution des clés aux entités (vérification, authentification des entités, etc.).
  - Stockage des clés de manière sécurisée (chiffrement des clés, archivage fiable afin d'assurer la confidentialité et l'intégrité des clés).
  - Surveillance d'enregistrement, d'audit, de test de bon fonctionnement des clés, etc.
  - Destruction des clés inutiles (destruction physique).

Public Key Infrastructure

- **PKI (Public Key Infrastructure)** (IGC infrastructure de Gestion de Clés publiques) : est un système de gestion des clés publiques qui permet de gérer des listes importantes de ces clés et d'en assurer la fiabilité, pour des entités généralement dans un réseau.

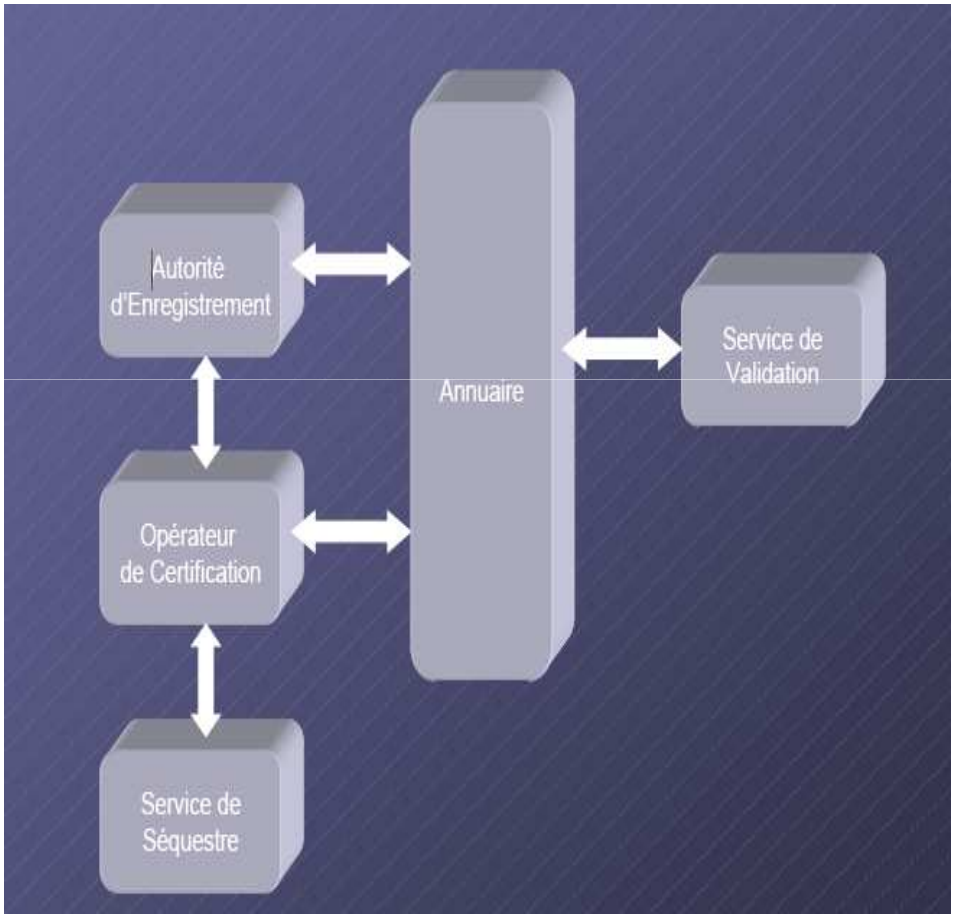
- Elle offre un cadre global permettant :.
  - **La gestion des clés** :
    - La génération et l'émission d'un couple unique de clés .
    - L'Enregistrement ,
    - Le stockage,
    - Le recouvrement en cas de pertes par l'utilisateur ou de demandes de mise à disposition par les autorités judiciaires.
  - **La gestion des certificats numériques**: création, signature, émission, validation, renouvellement

- Elle permet d'instaurer des éléments de sécurité tels que la confidentialité, l'authentification, l'intégrité et la non-répudiation tant au sein de l'entreprise que lors d'échanges d'information avec l'extérieur

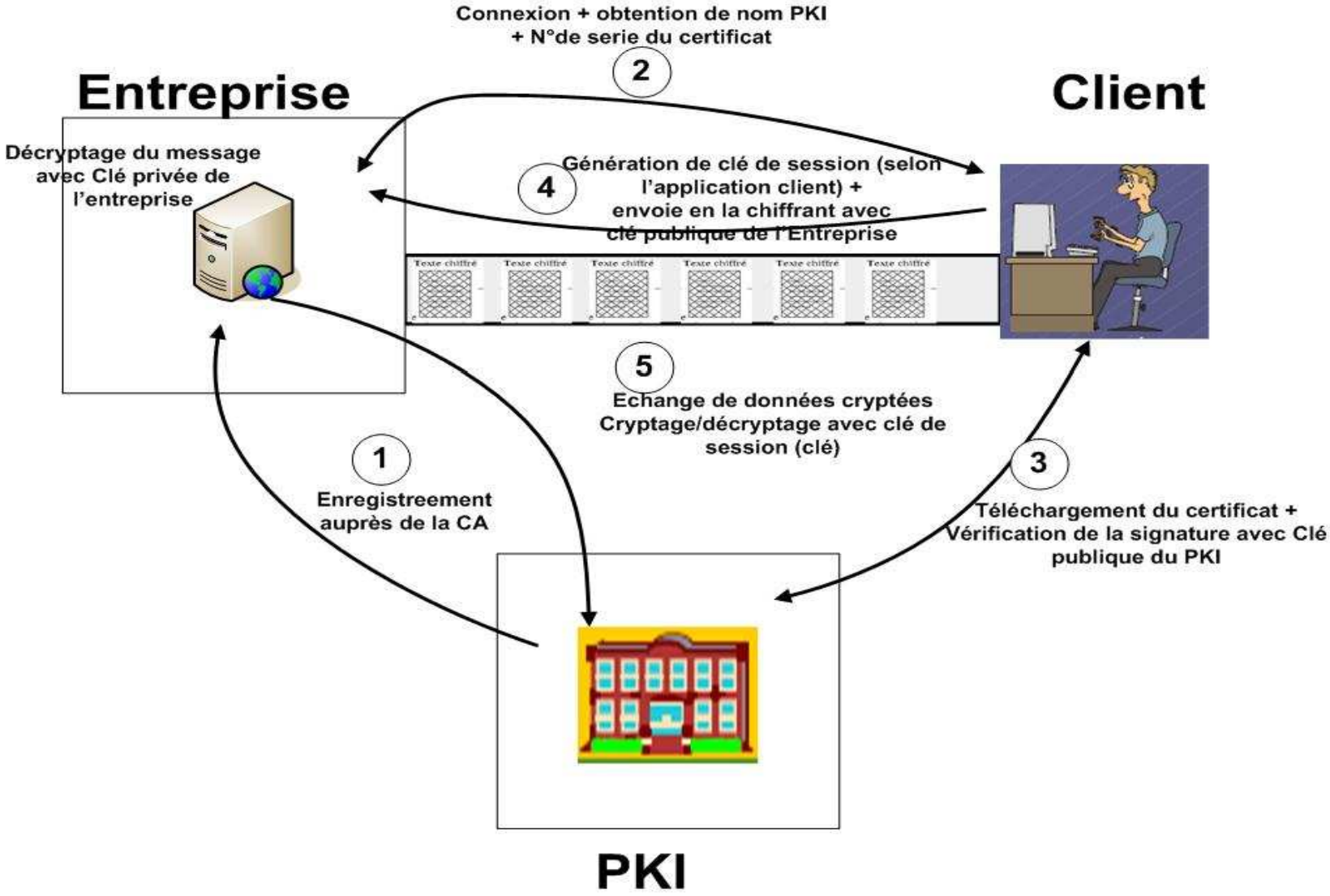
Public Key Infrastructure

Les composants d'une PKI

- Une autorité de certification (**AC)**
- Une autorité d'enregistrement (**AE)**
- Un opérateur de certification (**OC)**
- Un **annuaire** de publication de certificats
- Un **service de validation**
- Les utilisateurs finaux et les administrateurs



Transactions sécurisées avec PKI



Solutions

- **Solutions Commerciales**
  - Entrust http://www.entrust.com/
  - Baltimore
  - www.globalsign.fr
  - RSA
  - Verisign (www.verisign.com)
  - Cryptomatic
  - etc.
- Solutions open source
  - OpenSSL
  - OpenCA
  - IdealX
  - NewPKI
  - etc.

Les avantages d'une PKI

- La gestion centralisée de l'ensemble du cycle de vie des certificats.
- La compatibilité avec les différents standards de token (soft, carte puce, carte USB offrant différents niveaux de sécurité pour la protection des clés privées) mscapi, pkcs#11, ..
- La distribution sûre des clés publiques grâce au format de certificat normalisé X509.
- Le support des mécanismes de gestion des droits et privilèges (BD de gestions des droits, Annuaires)
- La compatibilité avec les applications majeures de sécurité (SSL, S/MiMe, IPSec, VPN…)
- **Le plus important : La PKI agit en tiers de confiance en se portant garant de l'identité du titulaire du certificat**

Fin du chapitre