University of khenchela
Departement MI

Computer security

Dr.Hichem Rahab

rahab.hichem@univ-khenchela.dz
L3 ISIL

2022 - 2023

Chapter 4: Public-Key Cryptography

Overview

Public-Key Cryptography

- Le concept du chiffrement asymétrique date officiellement de 1976 de Diffie et Hellman.
- La première implémentation a lieu en 1978 par Rivest, Shamir et Adleman sous la forme de l'algorithme RSA.
- La sécurité de tels systèmes repose sur des problèmes calculatoires :
  1. RSA : factorisation de grands entiers
  2. ElGamal : logarithme discret
  3. etc.

Public-Key Cryptography -(2)

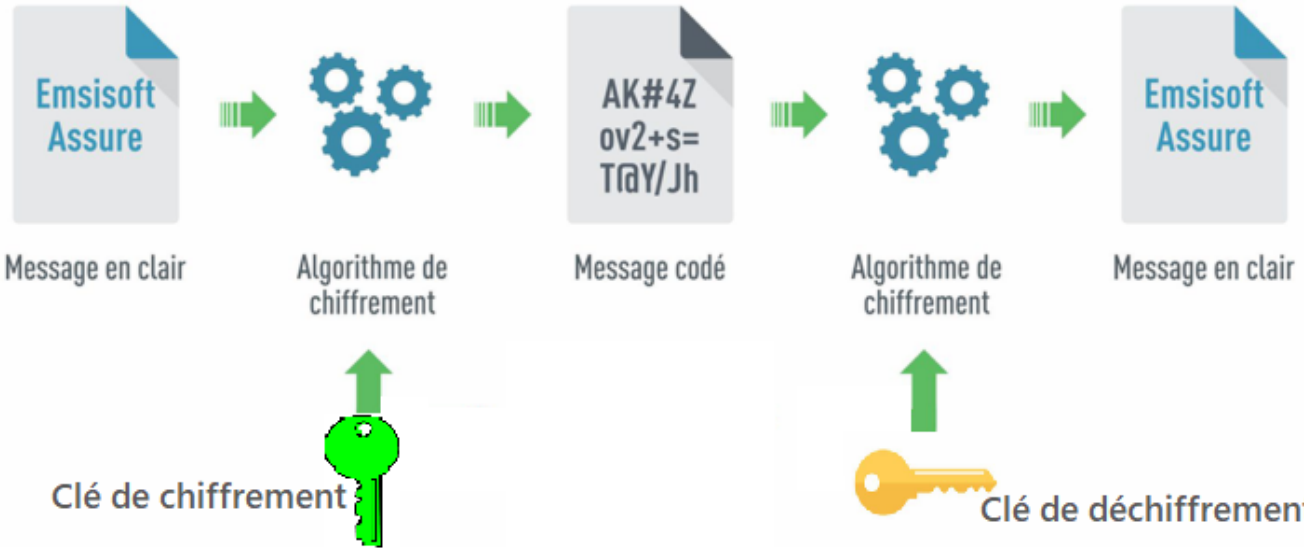
- Dans le cas des systèmes **symétriques**, on utilise une même clé pour le chiffrement et le déchiffrement. Le problème repose dans la transmission de la clé : il faut une clé par destinataire.
- Dans le cas des systèmes **asymétriques**, chaque personne possède 2 clés distinctes (une privée, une publique).
- La clé privée n'est pas déductible à partir de la clé publique. De ce fait, il est possible de distribuer librement cette dernière.
- On peut classer l'utilisation des algorithmes à clé publique en 3 catégories :
  1. Chiffrement/déchiffrement : cela fournit la confidentialité.
  2. Signature numériques: cela fournit l'authentification.
  3. Échange de clés symétrique (ou des clefs de session).

Public-Key Cryptography - (3)

- Une clé publique P_K (ou clé de chiffrement, symbolisée par la clé verticale),
- Une clé privée secrète S_K (symbolisée par la clé horizontale),
- Propriété : La connaissance de P_K ne permet pas de déduire S_K,
- $D_{SK}(E_{PK}(M)) = M$,
- La taille des clés s'étend de 512 bits à 2048 bits en standard. Dans le cas du RSA, une clé de 512 bits n'est plus sûre au sens "militaire" du terme, mais est toujours utilisable de particulier à particulier.

Public-Key Cryptography - (4)

- Le principe de ce genre d'algorithme est qu'il s'agit d'une fonction unidirectionnelle à trappe (piège). Une telle fonction à la particularité d'être facile à calculer dans un sens, mais difficile voire impossible dans le sens inverse.
- La seule manière de pouvoir réaliser le calcul inverse est de connaître une trappe. Une trappe pourrait par exemple être une faille dans le générateur de clés. Cette faille peut être soit intentionnelle de la part du concepteur (définition stricte d'une trappe) ou accidentelle.



Public-Key Cryptography (5)

- Au niveau des performances, le chiffrement par voie asymétrique est environ 1000 fois plus lent que le chiffrement symétrique.
- Cependant, à l'inverse du chiffrement symétrique où le nombre de clés est le problème majeur, ici, seules n paires sont nécessaires.
- En effet, chaque utilisateur possède une paire (S_K, P_K) et tous les transferts de message ont lieu avec ces clés.
- La distribution des clés est grandement facilitée car l'échange de clés secrètes n'est plus nécessaire.
- Chaque utilisateur conserve sa clé secrète sans jamais la divulguer. Seule la clé publique devra être distribuée ...

Public-Key Cryptography - Protocole

Protocole

- Algorithme de génération des clés KG(l) = (pk,sk) à partir d'un paramètre de sécurité, il produit une paire de clés,
- Algorithme de chiffrement E(pk, m) = c permet de générer le chiffré d'un message m, par la clé publique,
- Algorithme de déchiffrement D(sk, c) = m utilise la clé sécrete/privée sk pour retrouver m à partir de c

Fonctions à sens unique

Fonctions à sens unique

Pour une relation $y = f(x)$, calculer y est facile, et retrouver x à partir de y est difficile sans une "trappe".

Fonctions à sens unique (2)

Exemple :(fonction irréversible)

Prenant $X = 1, 2, 3, ..., 16$ et on définit $f(x)$ comme le reste de la division entière de 3^x par 17. Les résultats sont donnés:

Alors on a : $f(x) = 3^x mod(17)$

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$f(x)$	3	9	10	13	5	15	11	16	14	8	7	4	12	2	6	1

Alors donnant un nombre entre 1 et 16, il est facile de calculer son image par f. Cependant, en donnant un nombre tel que 7, sans avoir le tableau précédent, il est difficile de trouver x sachant que $f(x) = 7$. Si on a $f(x) = 3$ il est claire que $x = 1$, mais pour la plupart des autres nombres le calcule c'est pas assez facile.

RSA : Rivest - Shamir - Adleman

Principe de l'algorithme RSA

- Cet algorithme est fondé sur la théorie des nombres.
- L'utilisateur choisit deux grands nombres premiers, p et q, (En pratique, si p et q ont 100 chiffres décimaux, n possèdera 200 chiffres.), et calcule $n = p \times q$ et $\phi(n) = (p–1) * (q–1)$.
- Il choisit un nombre **d** premier avec $\phi(n)$ et cherche un nombre **e** tel que $e * d = 1(mod\phi(n))$.
- La clé publique est le couple $P_k = (e, n)$, la clé secrète le couple $S_k = (d, n)$.
- Considérons un message à chiffrer qui doit être expédié à un utilisateur qui a publié sa clé publique $P_k = (e, n)$.
- Le message en clair est découpé en une suite de blocs de telle sorte que chaque bloc en clair M soit un nombre inférieur à n.

RSA : Rivest - Shamir - Adleman -(2)

Résumé de l'algorithme RSA:

1. Génération de 2 nombres premiers p et q
2. Calcul de n = p * q
3. Calcul $\phi(n) = (p-1) * (q-1)$
4. Proposer d tel que: d et $\phi(n)$ n'ont pas de facteurs commun.
5. Déterminer e tel que $3 < e < n$ et $e * d \equiv 1 \pmod{\phi(n)}$
6. Clé publique : (e,n)
7. Clé privée : (d,n)
8. p et q doivent rester secrets, voire supprimés
9. C = $M^e mod(n)$ et M = $C^d mod(n)$

RSA : Rivest - Shamir - Adleman -(3)

Exemple:

Soit $p = 11$ et $q = 17$ d'où $n = 187$ et $\phi(n) = (11 - 1) \times (17 - 1) = 160$.
Choisissons $d = 7$, cette valeur convient puisque 7 et 160 n'ont pas de facteurs communs.
L'équation $e \times 7 = 1(mod160)$ donne: $e = 23$ puisque
$23 * 7 = 161 = 160 + 1$.
Alors: La clé publique $P_k = (7, 187)$ et la clé privée $S_k = (23, 187)$
Pour chiffrer le message $M = 88$, l'émetteur calcule $88^7(mod187)$ soit 11 et envoie ce " message".
Le récepteur qui connaît sa clé secrète calcule $11^{23}(mod187)$ et il trouve 88.

Exemple 2:

Vous avez reçu le cryptogramme m=7, calculer le message claire correspondent utilisant la clé privé $S_k = (23, 187)$

RSA : Rivest - Shamir - Adleman -(4)

Attaques sur RSA

Il existe trois approches pour attaquer le RSA :

1. Recherche par force brute de la clé (impossible étant donné la taille des données),
2. Attaques mathématiques (basées sur la difficulté de calculer $\phi(n)$, la factorisation du module n):
   - factoriser $n = p * q$ et par conséquent trouver n et puis d,
   - déterminer $\phi(n)$ directement et trouver d,
   - trouver d directement.
3. Attaques de synchronisation (sur le fonctionnement du déchiffrement).

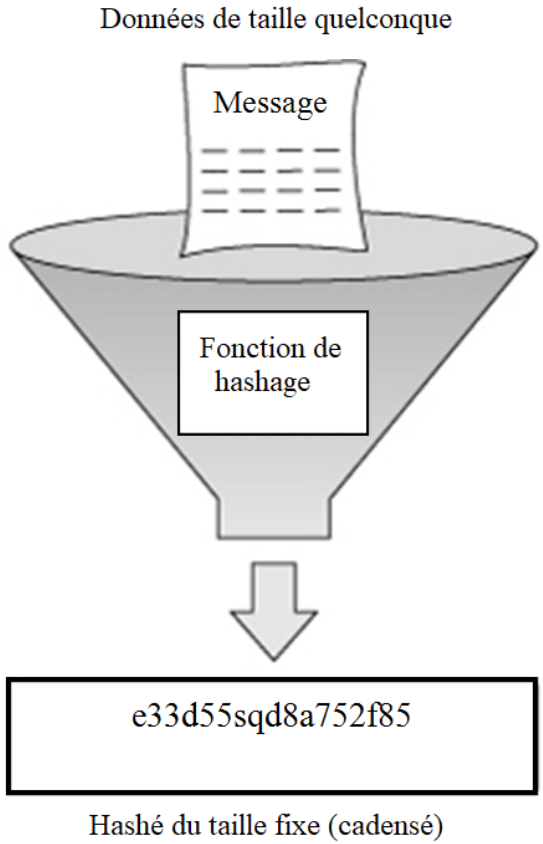
A l'heure actuelle, la factorisation connait de lentes améliorations au cours des années. La meilleure amélioration possible reste l'optimisation des algorithmes. Excepté un changement dramatique, le RSA-1024 restera sûr pour les prochaines années.

La fonction de hachage

Hash function

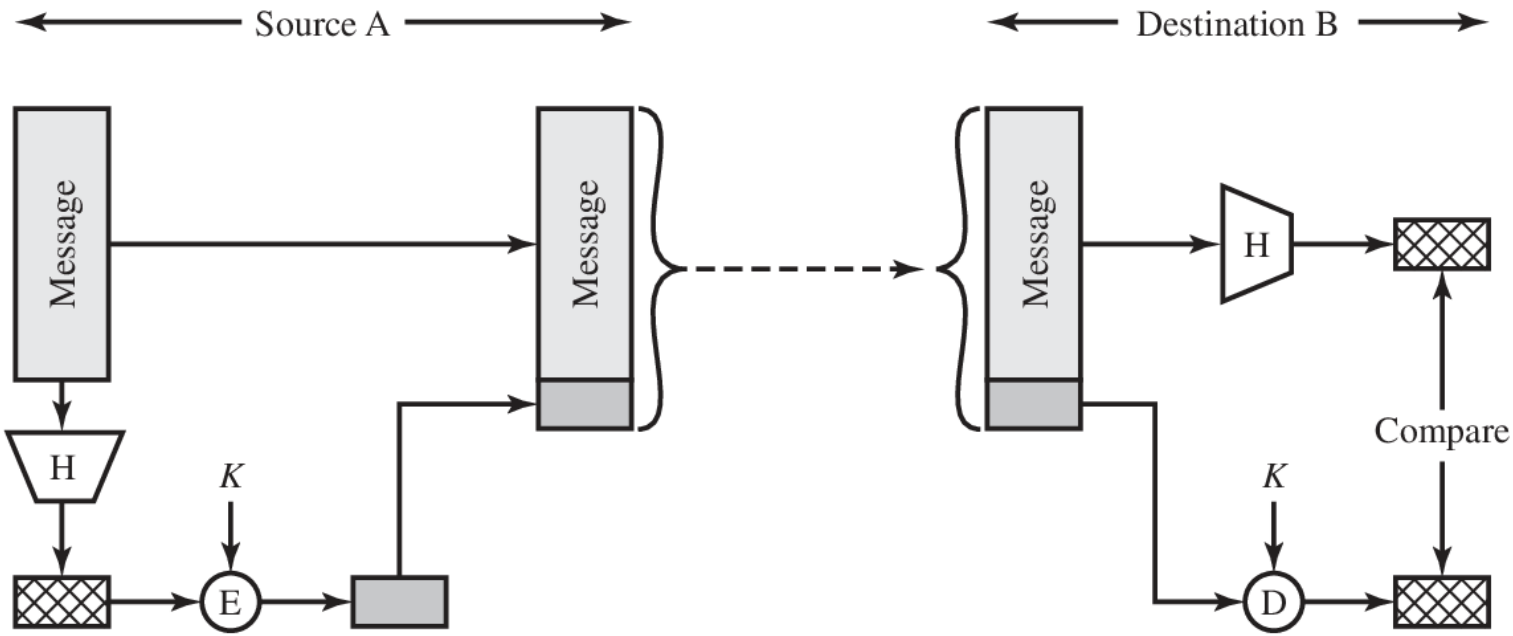
- Le principe est qu'un message clair de longueur quelconque va être transformé en un message de longueur fixe inférieure à celle de départ.
- L'algorithme utilisé est appelés **Hash function** fonction de hachage ou *fonction de condensation*.
- Le message réduit portera le nom de *"Haché"* ou de *"Condensé"*.
- L'intérêt est d'utiliser ce condensé comme empreinte digitale du message original afin que ce dernier soit identifié de manière univoque.
- Deux caractéristiques (théoriques) importantes sont les suivantes :
  1. Ce sont des fonctions unidirectionnelles :
     A partir de H(M) il est infaisable de retrouver M.
  2. Ce sont des fonctions sans collisions :A partir de H(M) et M
     il est impossible de trouver $M' \neq M$ tel que H(M') = H(M).

Hash function (2)



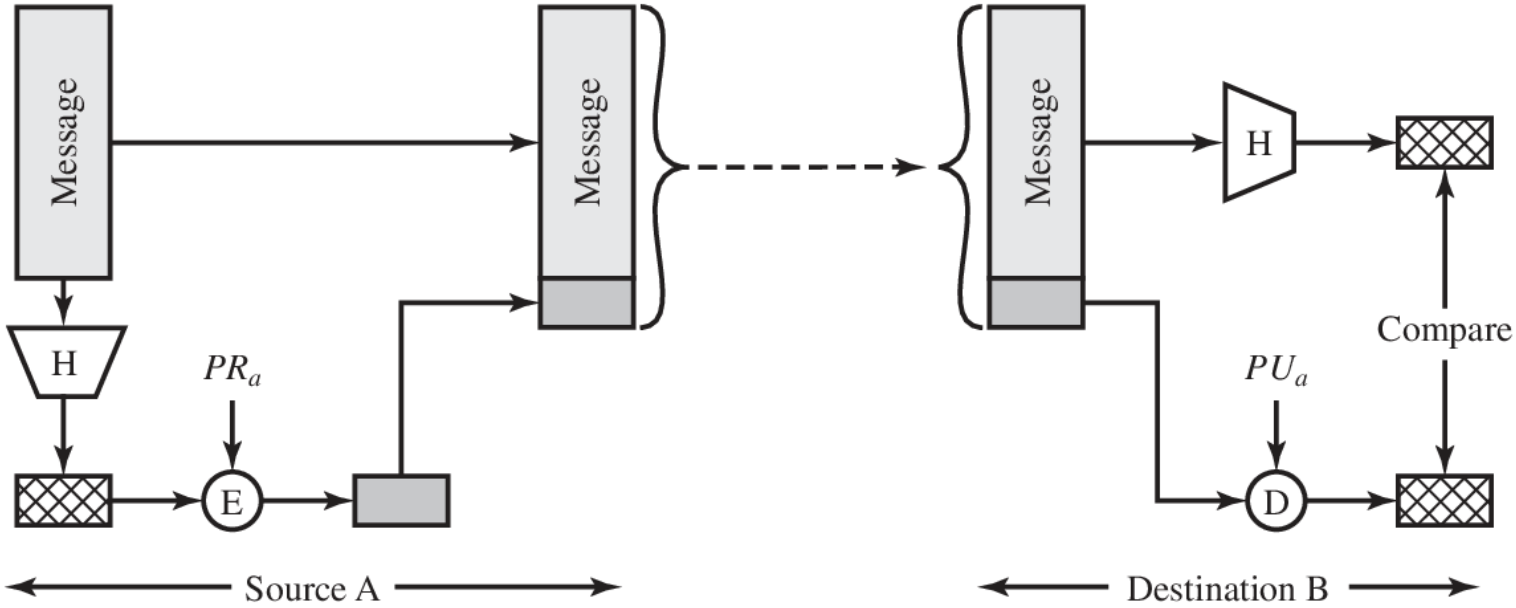
Hash function - symetric encryption

Utilisation du chiffrement symétrique.



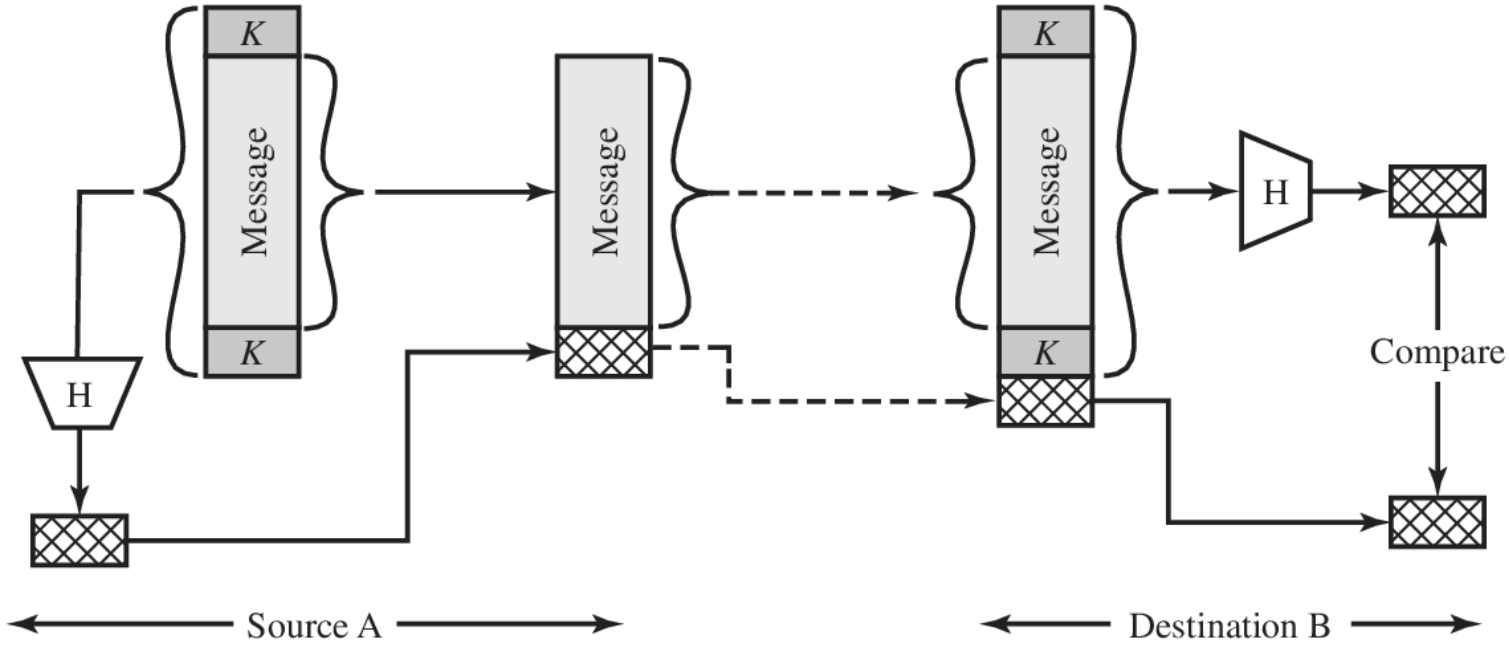
Hash function - asymetric encryption

Utilisation du chiffrement asymétrique (clé publique).



Hash function - secret key

Utilisation d'une clé secrète K.



Caractéristiques de fonctions de hachage

Le but de la fonction de hachage est d'avoir une empreinte de données. Afin d'être utile, une fonction de hachage H doit avoir les propriétés suivantes:

1. H doit applicable au block de données de n'importe quelle taille,
2. H produit un fichier de taille fixe,
3. H(x) est relativement simple à calculer, rendant une implémentation logicielle ou matériel faisable,
4. **Résistance à la preimage:** Pour un h donné, il est infaisable de trouver x tel que: $H(x) = h$
5. **Résistance à la seconde preimage:**(ou résistance faible à la collision) Pour un block de données x, il est infaisable de trouver $y \neq x$ avec $H(y) = H(x)$.
6. **Résistance à la collision:**(ou résistance forte à la collision) Il est infaisable de trouver un pair (x,y) tel que: $H(x) = H(y)$

Caractéristiques de fonctions de hachage

Résistance à la preimage: (Propriété du sens unique) Pour un h donné, il est infaisable de trouver x tel que: $H(x) = h$.

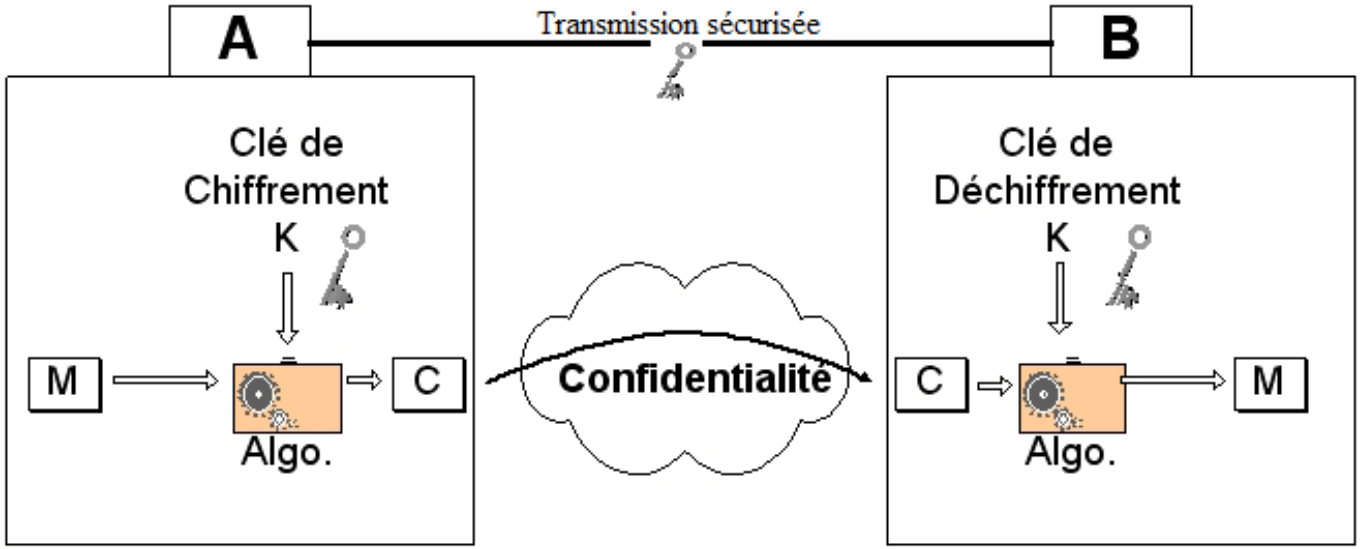
- Il est facile de générer un code pour un message, mais il est infaisable (virtuellement impossible) de récupérer un message depuis son code haché.
- Cette propriété est très importante dans le cas d'authentification par une clef secret.
- La valeur de la clef secrète n'est pas envoyée, cependant lorsque la fonction de hachage n'est pas a sens unique, un attaquant peut facilement récupérer le code secret.
- Alors, le cracker, une fois intercepter la transmission, peut obtenir le message et le code secret $MD_M = H(K\|M\|K)$.
- L'attaquant peut obtenir: $K\|M\|K = H^{-1}(MD_M)$.
- L'attaquant possède alors: M et $K\|M\|K$, il peut alors facilement trouver K.

Protocoles cryptographiques

Protocoles cryptographiques

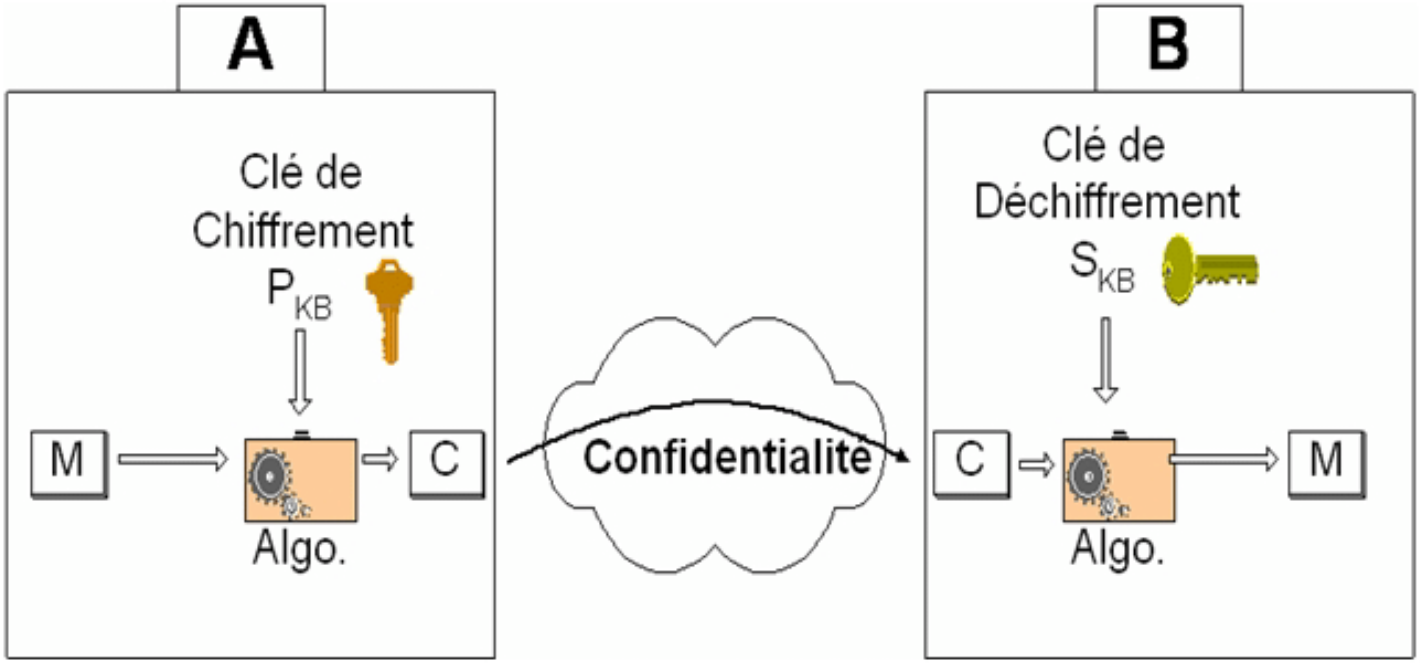
- Dès que plusieurs entités sont impliquées dans un échange de messages sécurisés, des règles doivent déterminer l'ensemble des opérations cryptographiques à réaliser, leur séquence, afin de sécuriser la communication C'est ce que l'on appelle les protocoles cryptographiques.
- Lorsque l'on parle de *"sécuriser un échange"*, on souhaite prêter attention aux 3 services suivants :
  1. La confidentialité,
  2. L'intégrité et
  3. L'authentification.
- Signalons la distinction entre **"services"** (confidentialité, intégrité, etc.) et **"mécanismes"** (les moyens utilisés : chiffrement, signature, hachage, etc.).

1. **Système symétrique:**
   - Elle est assuré par le chiffrement du message.
   - Dans le cas de systèmes à clés symétriques, la même clé est utilisée pour chiffrement $E_K(M)$ et déchiffrement $D_K(C)$.
   - Ce type de chiffrement nécessite un échange sûr préalable de la clé K entre les entités A et B.



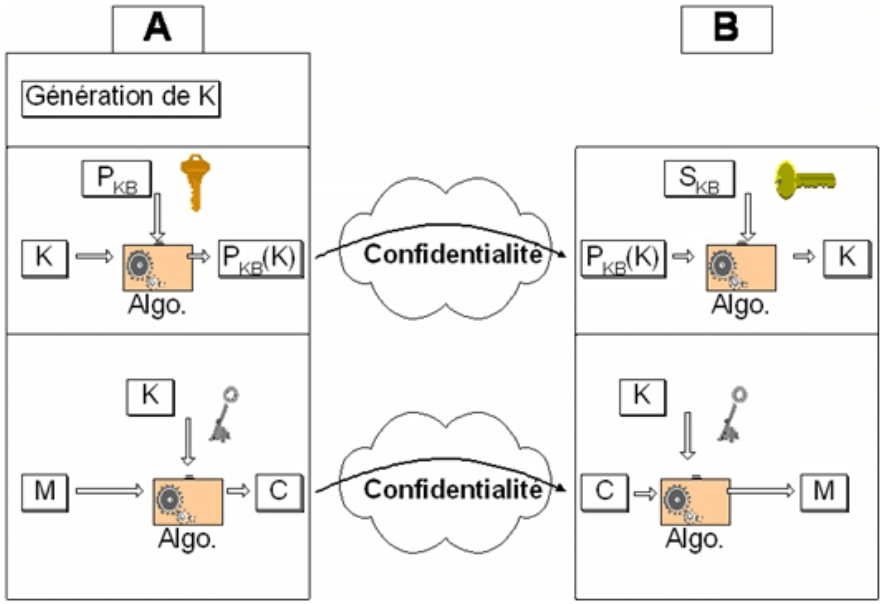
2. Système asymétrique:

- A l'aide d'un cryptosystème asymétrique, l'échange préalable de la clé n'est pas nécessaire.
- Chaque entité possède sa propre paire de clés. On aura donc pour l'entité A, la paire P_{KA}, S_{KA} et pour l'entité B la paire P_{KB}, S_{KB} .



3. Système hybride:

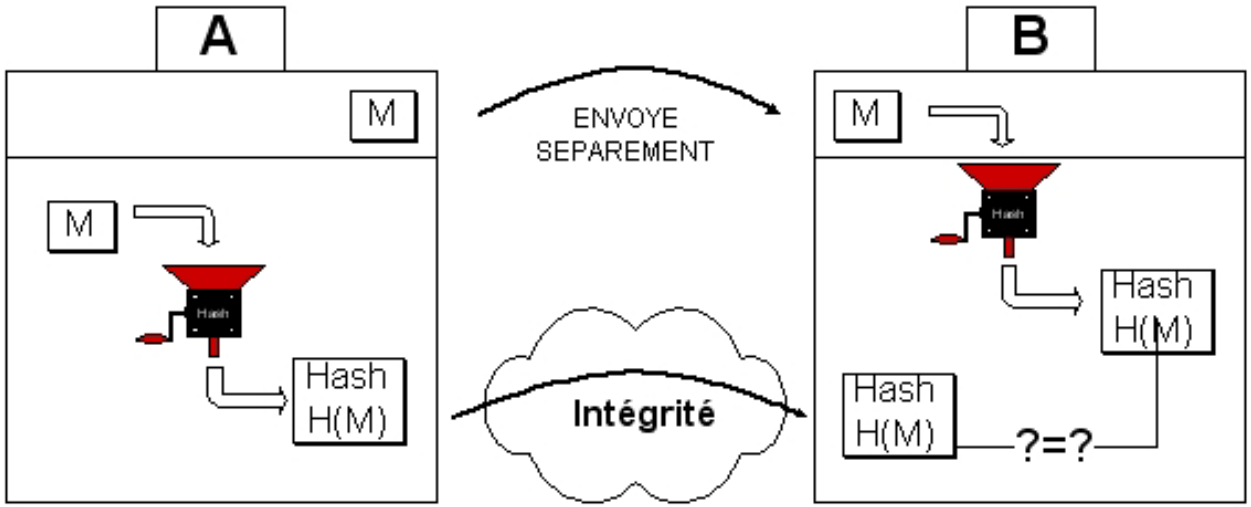
- Un système **"hybride"**, repose sur les deux systèmes précédents.
- A l'aide du système à clé publique, on sécurise l'échange de la clé K.
- Ensuite, les deux parties ayant acquis de manière sécurisée cette clé de chiffrement K, on utilisera le système à clé symétrique pour chiffrer le message.



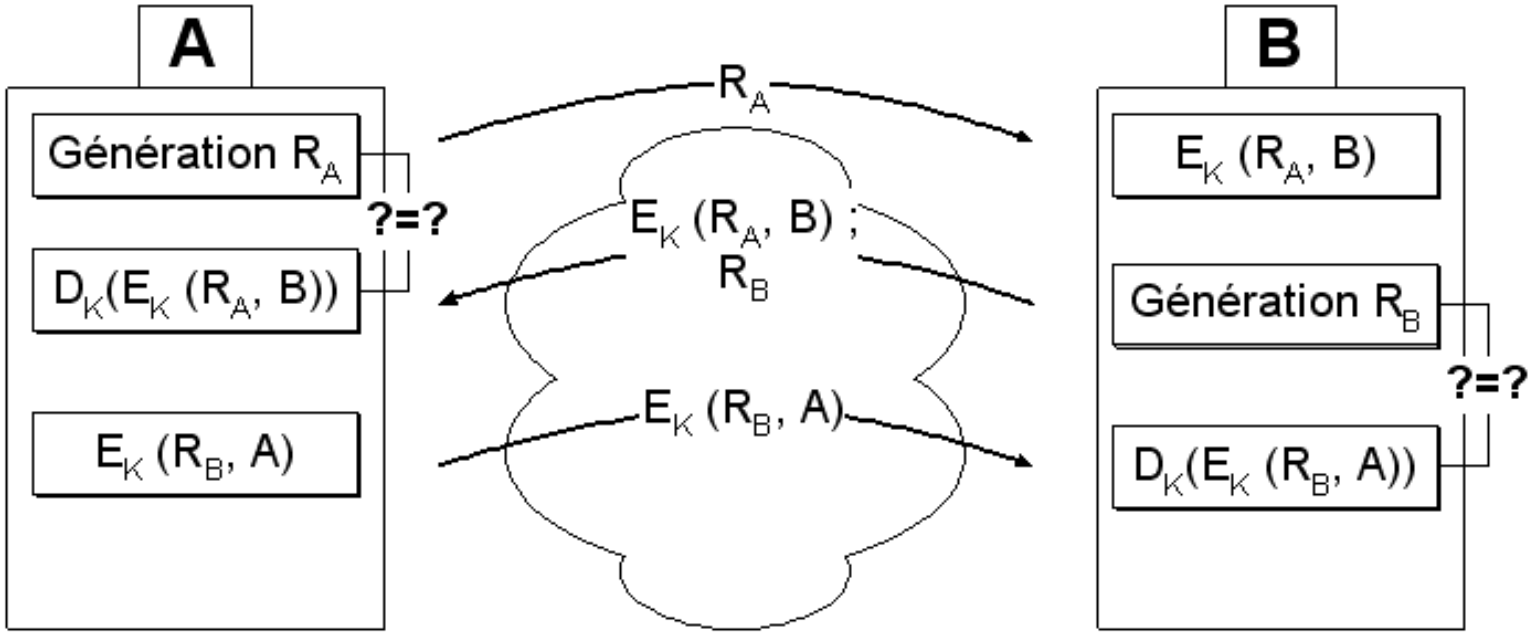
Protocoles cryptographiques

Intégrité

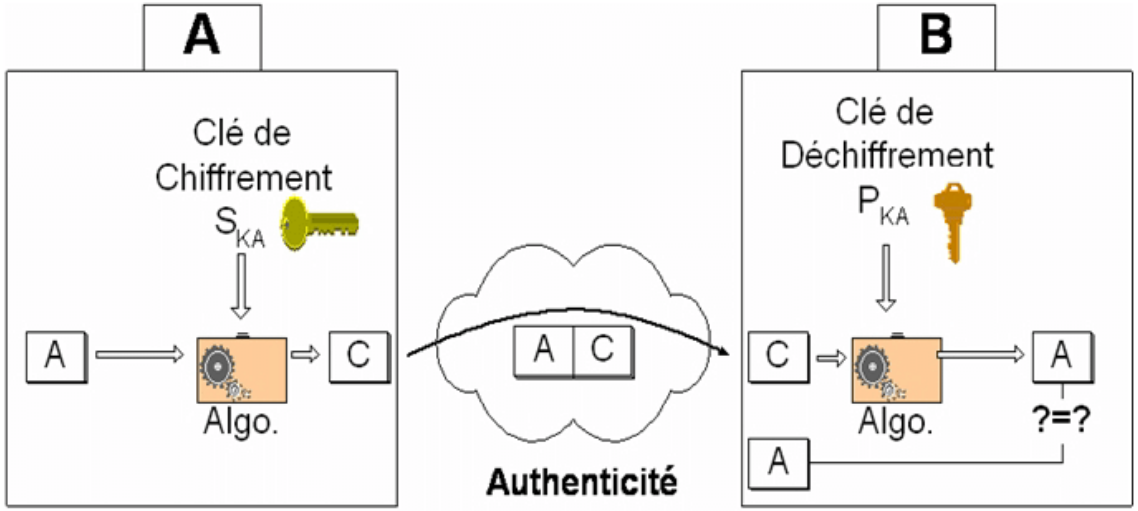
- Il s'agit ici de vérifier si le message n'a pas subi de modification durant la communication.
- C'est ici qu'interviennent les fonctions de hachage.



1. **Au niveau des parties communicantes:**
   - **A .Le cas d'un système symétrique :** R_A est une nonce (p. ex. nombre aléatoire), propre à l'utilisateur A. Les lettres A et B représentent des identificateurs personnels.

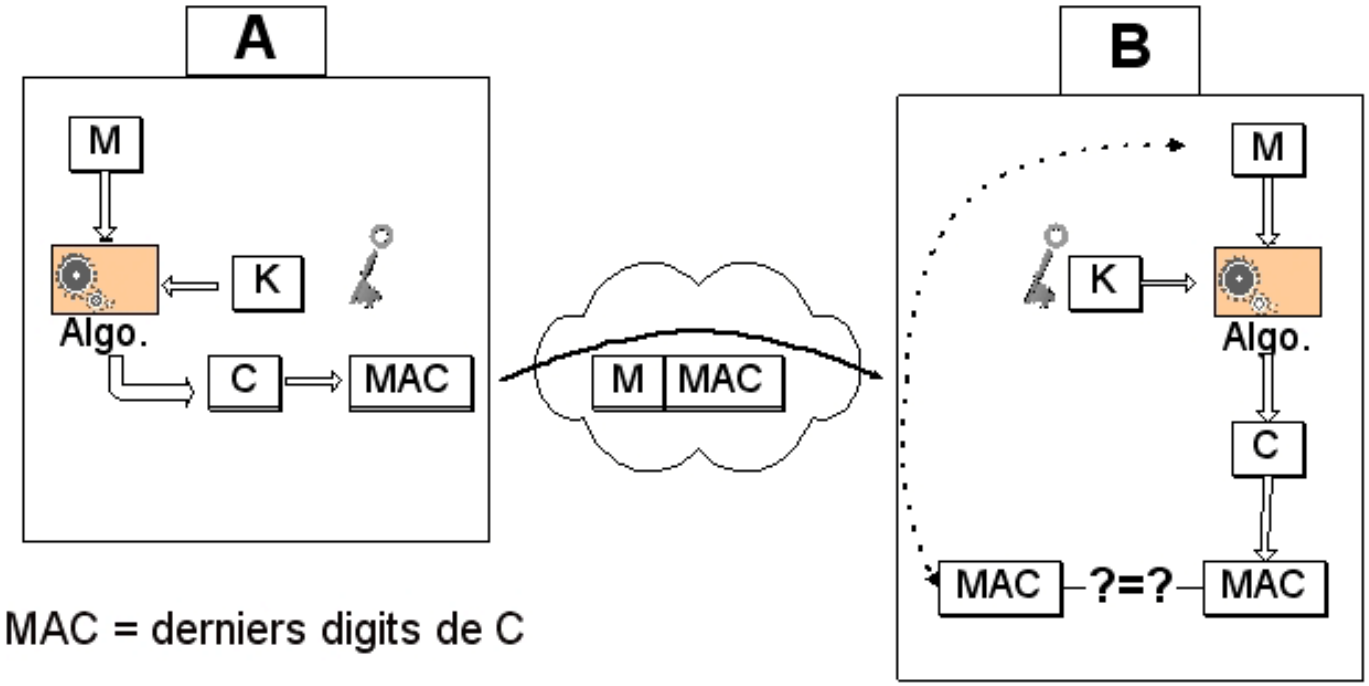


1. **Au niveau des parties communicantes:**
   - **B .Le cas d'un système asymétrique:** Comme le propriétaire de la clé secrète est le seul à la connaitre, cela prouve qu'il est bien la personne ayant chiffré le message.
     Dans cet exemple, seule l'authentification est souhaitée. Le message est envoyé en claire, la confidentialité n'est pas assurée ici.



2. Au niveau du message:

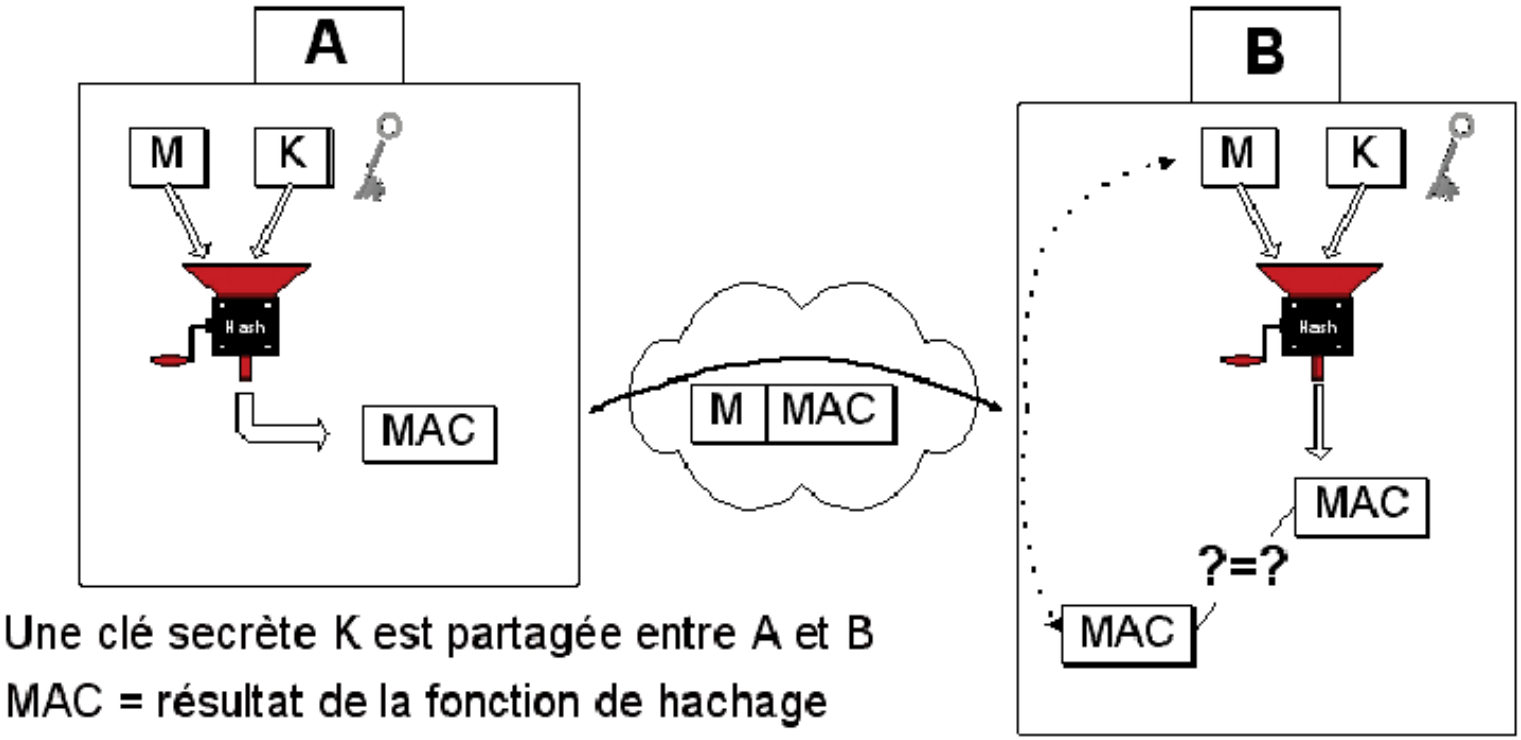
- A. L'utilisation d'un MAC (Message Authentication Code) généré à l'aide d'un cryptosystème à clé symétrique, où le MAC est constitué des derniers digits de C



MAC = derniers digits de C

2. Au niveau du message:

- B. L'utilisation d'un MAC (Message Authentication Code) généré à l'aide d'une fonction de hachage,

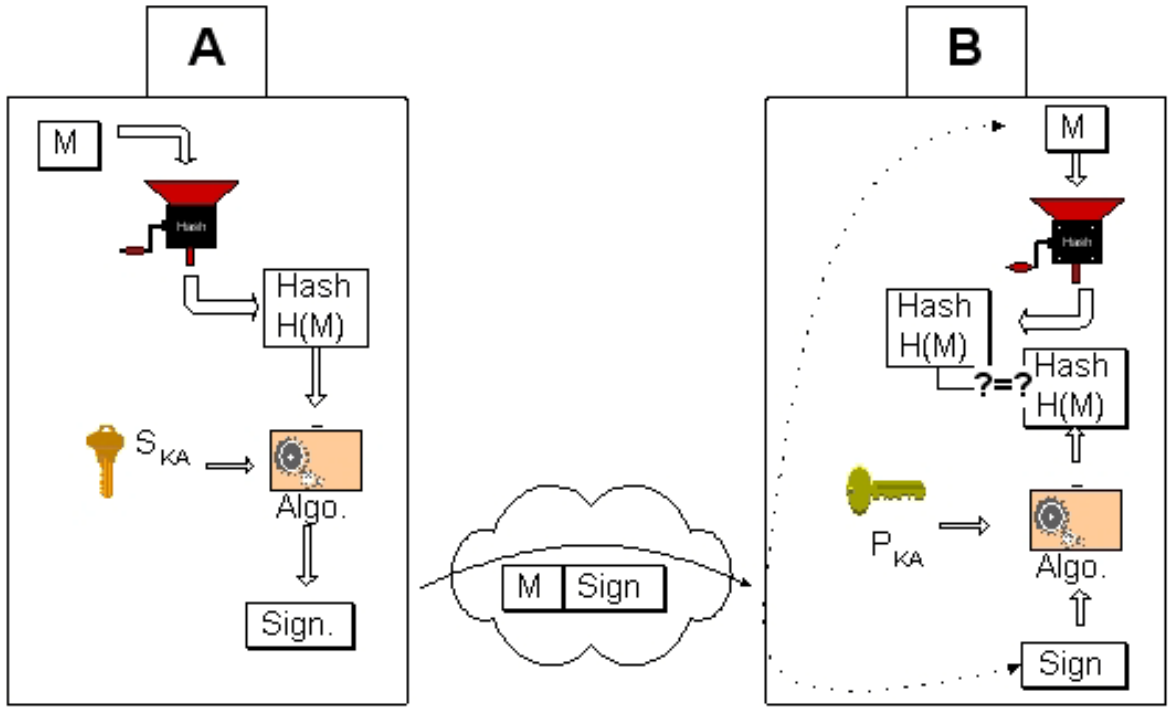


Une clé secrète K est partagée entre A et B

MAC = résultat de la fonction de hachage

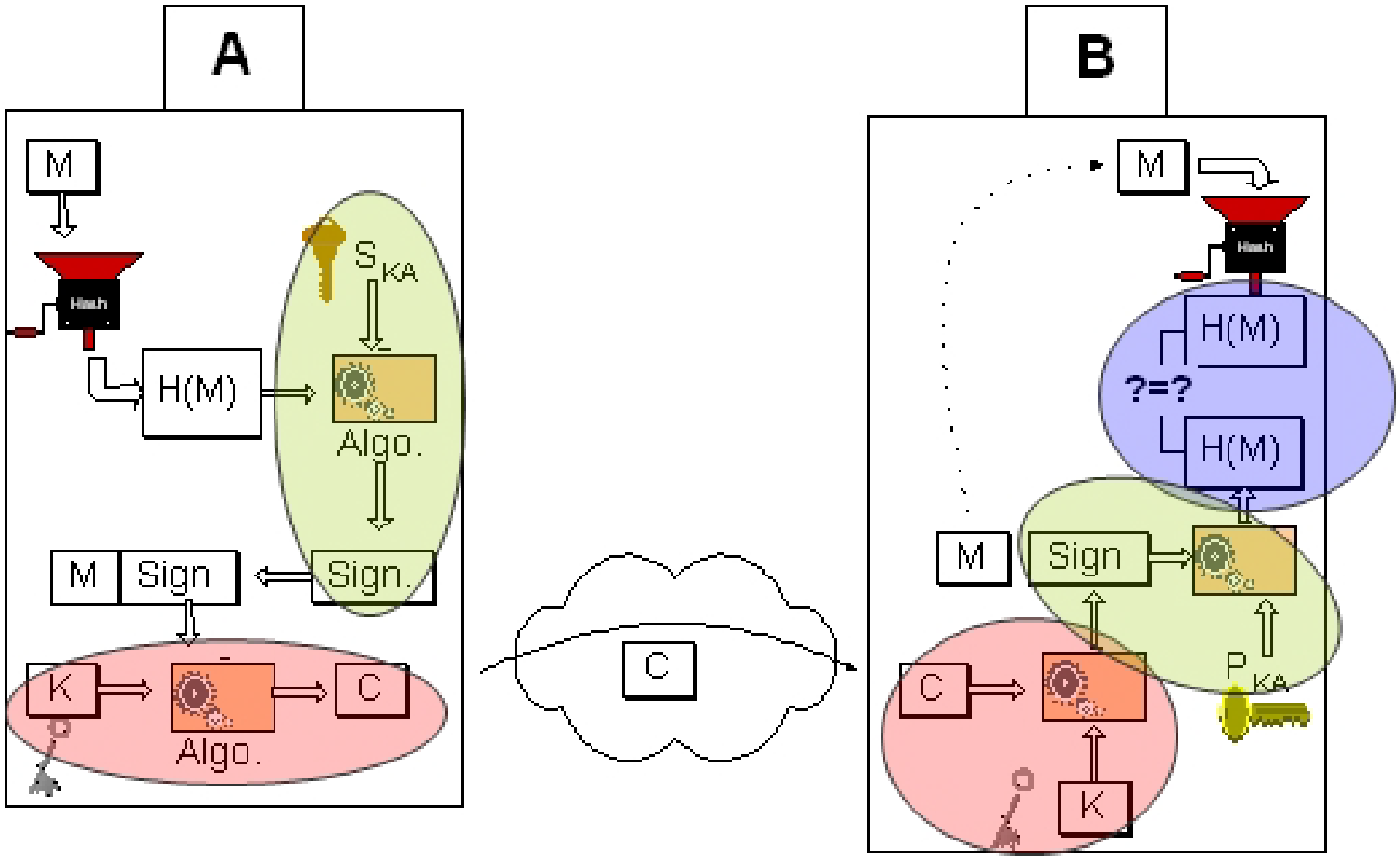
2. Au niveau du message:

- C. Par l'utilisation d'une signature digitale. Parmi les propriétés remarquables de ces signatures, on peut dire qu'elles doivent être authentiques, infalsifiables, non-réutilisables, non-répudiables, et inaltérables.



Synthèse

Confidentialité(Rouge), Intégrité(Violet), Authentification(Vert)



The end.