



Université Assane Seck de Ziguinchor
UFR Sciences & Technologies
Département d'Informatique

Chapitre 3

Les mécanismes de sécurité

La cryptographie



Pr Youssou FAYE
★ ★ ★ ★ ★



La cryptographie

Sommaire

- ☛ **Introduction**
- ☛ **La cryptologie**
- ☛ **Cryptographie symétrique**
- ☛ **Cryptographie asymétrique**
- ☛ **Le hachage**
- ☛ **Les signatures numériques**
- ☛ **Les certificats numériques**



Introduction

Définitions

- **Cryptologie**, ancien art du secret ou nouvelle science des messages secret comprend la **cryptographie et la cryptanalyse**
- **Cryptographie:** des principes, méthodes et techniques dont l'application assure le **chiffrement et le déchiffrement** des données, afin d'en préserver la confidentialité, l'authenticité et l'intégrité
- **Cryptanalyse:** méthodes et procédés de **décryptage** visant à rétablir en clair un cryptogramme, sans connaissance préalable de la clé de **chiffrement**.

Introduction

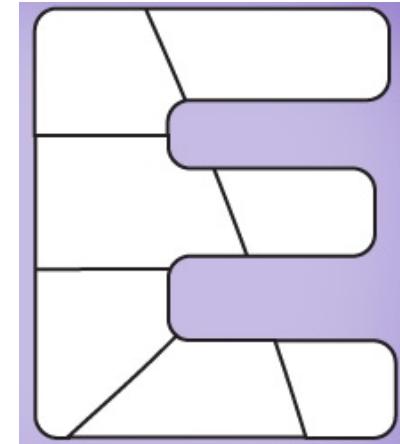
Définitions

👉 **Crypter:** rendre illisible un **texte en clair** sans connaître la **clé** après.

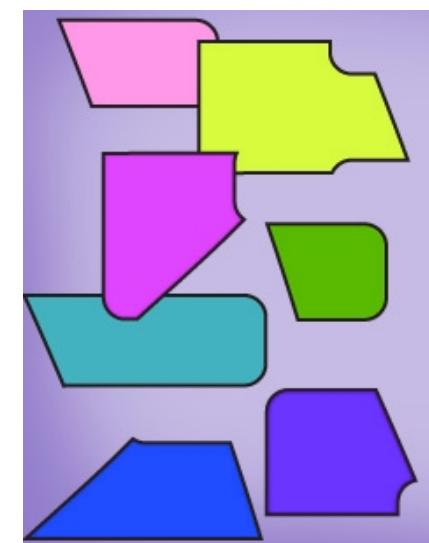
👉 La **clé** est le paramètre utilisé en entrée avec les données lors de l'opération de chiffrement/ cryptage

Exemple: puzzle

👉 **décrypter:** retrouver le **texte en clair** d'origine sans posséder la **clé**.



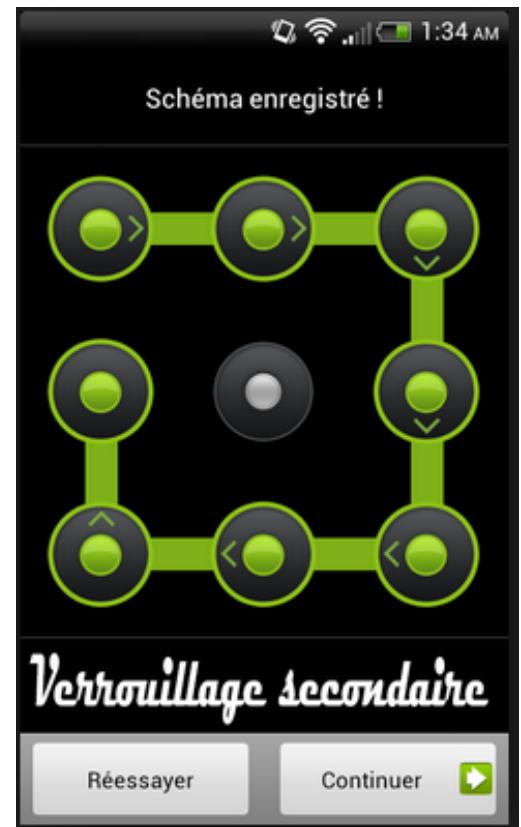
Crypter



Introduction

Définitions

- ☞ **Chiffrer:** opération qui transforme un texte en clair en un **texte chiffré ou cryptogramme**, inexploitable pour quiconque ne possède pas la clé.
- ☞ **Déchiffrer:** opération inverse du chiffrement, permettant à partir de la **clé**, de rétablir, en clair un **cryptogramme**.



Quelques services de sécurité

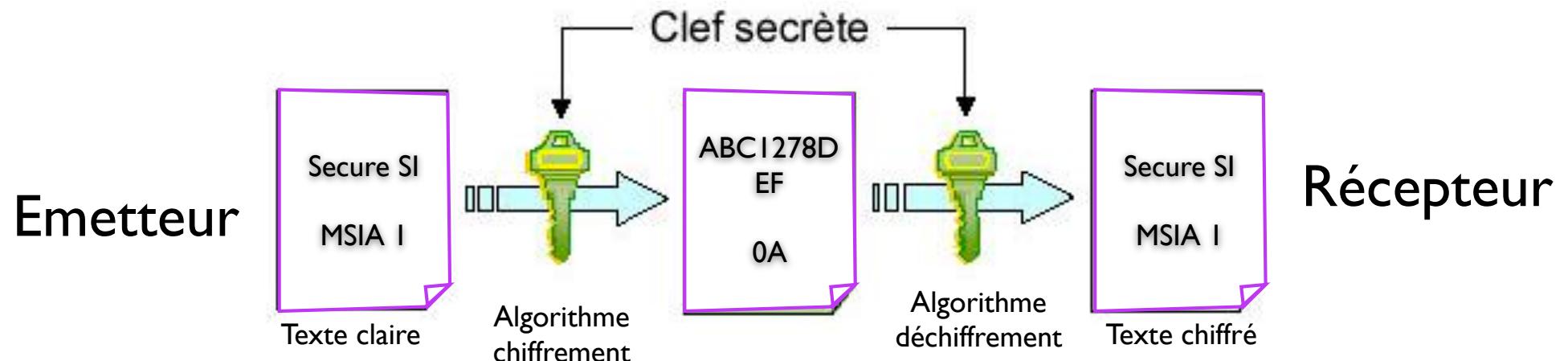
- La confidentialité
- L'authentification
- L'intégrité
- La disponibilité
- La non répudiation
- La fraîcheur
-



Cryptographie Symétrique

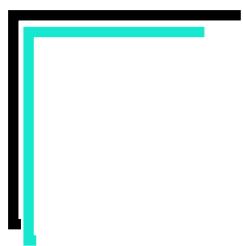
Cryptographie symétrique ou cryptographie à clé secrète

- En cryptographie à clé privée ou chiffrement symétrique, les données sont chiffrées et déchiffrées avec la **même clé** connue seulement de l'émetteur et du récepteur



Un algorithme cryptographique comprend deux fonctions:

- La fonction **E** (Encryption) noté $C=E(K,M)$: où C (**le cryptogramme**) est le résultat du message M chiffré avec la clé K .
- La fonction **D** (Decryption) noté $M=D(K,C)$: où M est le résultat du déchiffrement du cryptogramme C .



Cryptographie symétrique

Chiffrement traditionnel

Les méthodes de chiffrement traditionnelles utilisaient généralement des opérations de décalage, de permutation, de transposition



Chiffrement traditionnel

Opération de décalage

- chiffrement alphabétique par décalage (*Jules Cesar*): utilisé dans l'antiquité romaine
- Avec la clé K=3 (chiffre de César), le texte ABC est chiffré par DEF

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

Chiffrement traditionnel

➤ Algorithme du chiffrement alphabétique par décalage: substitution monoalphabétique

Numérotation de 0 à 25 les lettres de l'alphabet.

Choix d'une clé K comprise entre 1 et 25

On chiffre le caractère X par: $E(K,X) = (X+K) \text{ mod } 26$

Exemple: chiffrement du message M=ZIG avec K = 3

- Z = 25, $25 + 3 = 28$, $28 \text{ mod } 26 = 2$: C
- I = 8; $8 + 3 = 11$, $11 \text{ mod } 26 = 11$: L
- G = 6, $6 + 3 = 9$, $9 \text{ mod } 26 = 9$: J
- ZIG chiffré donne CLJ
- Problème: lorsque l'algorithme est connu, il est facile d'essayer toutes les 25 clés possibles.
- Si on connaît la langue qui est utilisée, on peut user de la fréquence d'apparition des lettres pour estimer le décalage (méthode d'AL-Kindi au neuvième siècle après J.C)

Chiffrement traditionnel

☞ Si la langue utilisée est le Français par exemple, la fréquence d'apparition des lettres est:

A	B	C	D	E	F	G	H	I	J	K	L	M
7,68	0,8	3,32	3,6	17,76	1,06	1,1	0,64	7,23	0,19	0	5,89	2,72

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
7,61	5,34	3,24	1,34	6,81	8,23	7,3	6,05	1,27	0	0,54	0,21	0,07

- AL-Kindi:
- Le texte: « Le problème est que lorsque l'algorithme est connu » est chiffré par « Oh sqreohph hvw txh orutxh odojrulwkph hvw frqqx » avec une clé de 3 .
- La lettre la plus fréquente est E, donc codée par h dans le texte chiffré.
- Donc il faut faire en sorte qu'une lettre ne soit toujours pas chiffrée de la même façon

Chiffrement traditionnel

Opération de permutation

- Algorithme : remplacer chaque lettre par une autre lettre (substitution monoalphabétique)

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

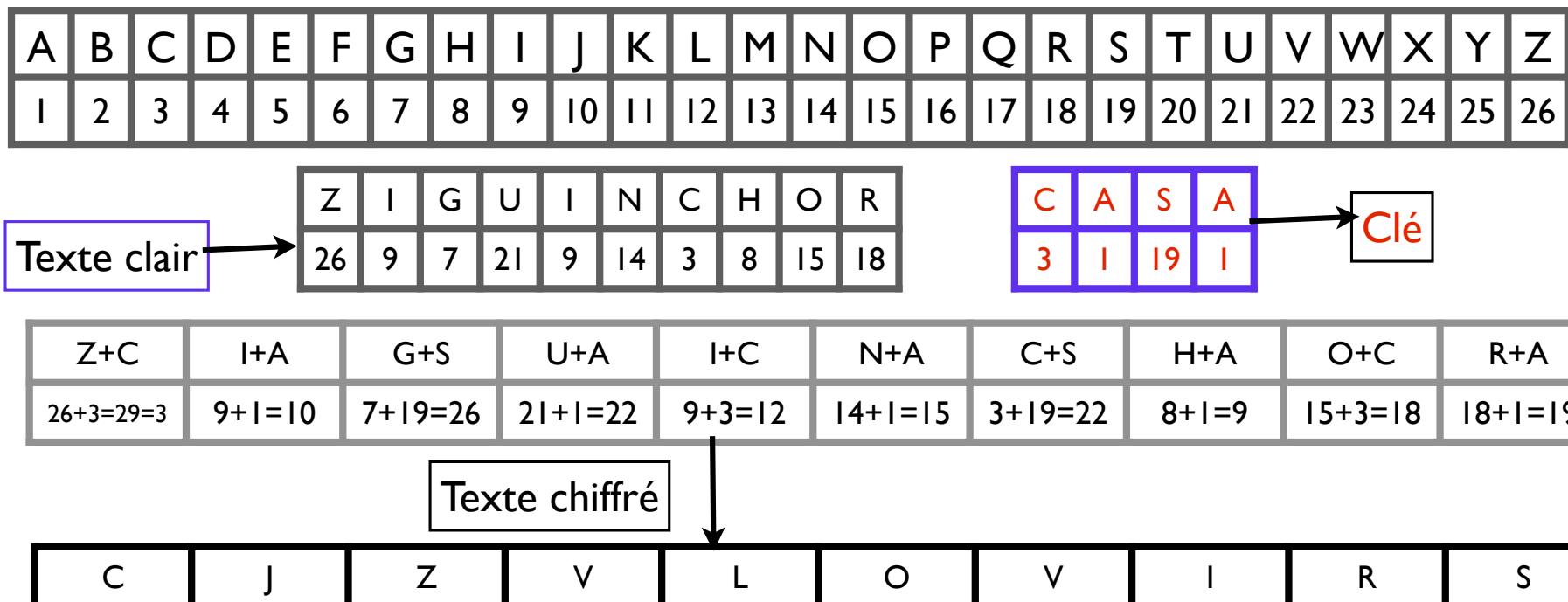
WELCOMETOUASZ
KBFZTRBPTQXON

- ☞ Même si l'algorithme est connu, il est impossible d'essayer les $26!-1$ clés possibles c.-à-d.
403291461126605635583999999

Chiffrement traditionnel

● Chiffrement par substitution poly-alphabétique (chiffrement de Vigenère)

- Utilise un mot comme clé au lieu d'un simple caractère (comme dans Cesar).
- A chaque lettre du texte clair on fait correspondre une lettre de la clef (la clef étant répétée autant de fois que nécessaire): **substitution polyalphabétique**



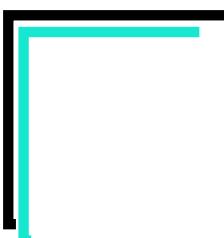
☞ Si l'algorithme est connu, il y a 26^4 Clés possibles



Chiffrement traditionnel

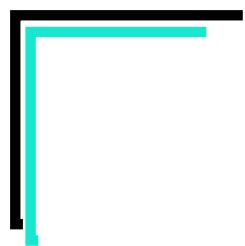
Opération de transposition

- Algorithme : permuter les lettres selon une permutation fixe
 - Clé: 123456 -> 642531
 - WELCOME TO UASZ
 - MCEOLWSUTAOEZ



Chiffrement traditionnel

- Quand on étudie un système, on part du principe que l'attaquant connaît le système de chiffrement utilisé sauf la clé: principe de **Kerckhoffs**.
- La sécurité du système doit toujours se reposer sur la clé, et le système restera sûr même si l'algorithme est connu.



Cryptographie symétrique

Chiffrement moderne

- Chiffrer

- Exemple avec
OU Exclusif

- Déchiffrer

100111=texte claire

110011=Clé

100111=texte claire

110011= Clé

010100=texte chiffré

010100=texte chiffré

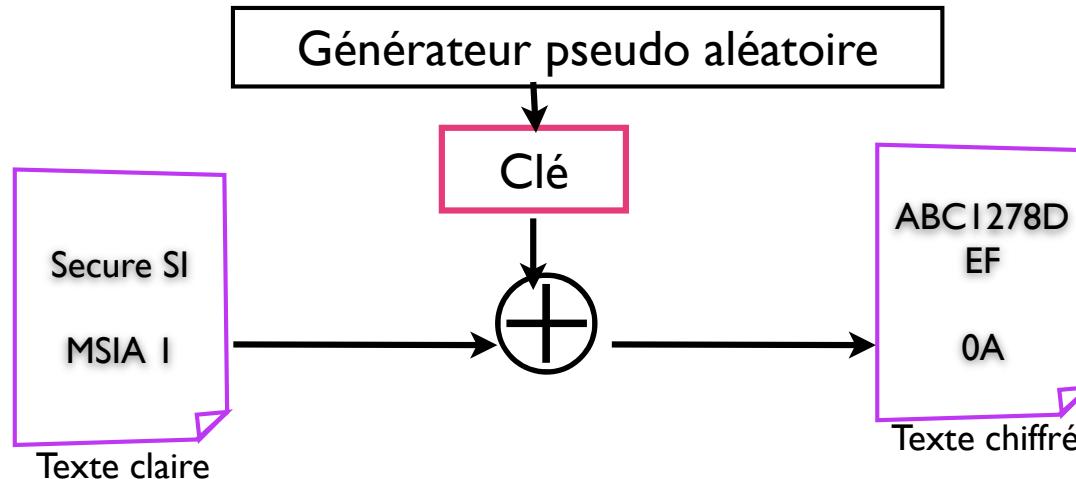
110011= Clé

100111=texte claire

Cryptographie symétrique

Chiffrement moderne

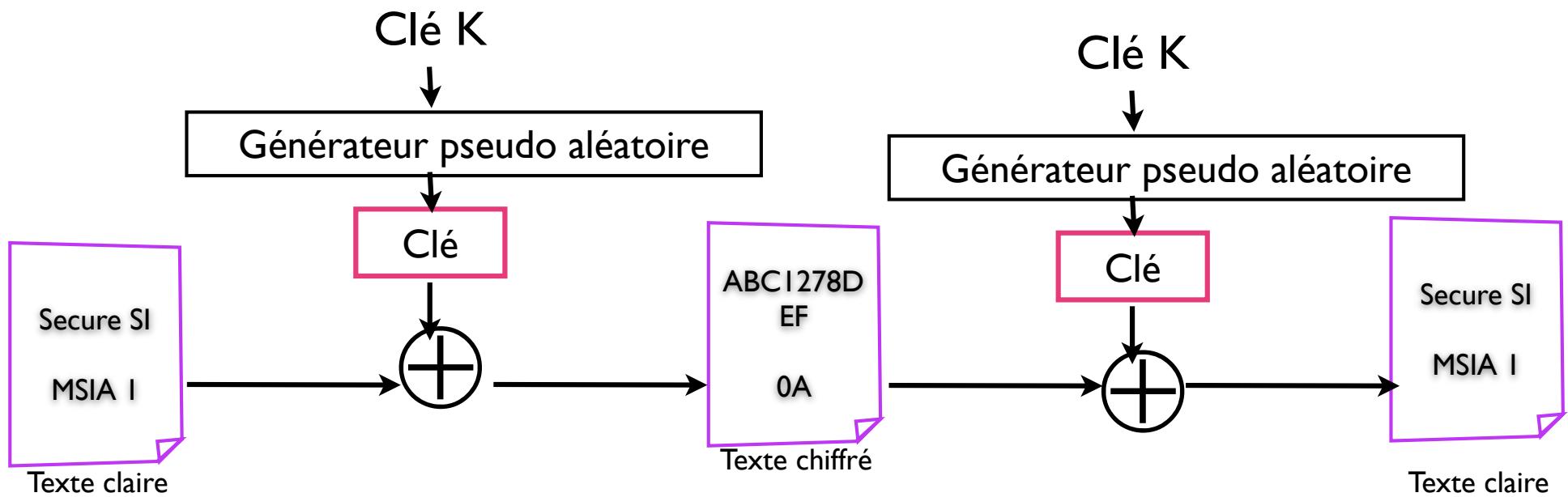
☞ Chiffrement de Vernam (1917) avec un simple opérateur XOR par exemple comme mode opératoire et la clé (pseudo-aléatoire)



- La clé est aussi longue que les données
- La clé est utilisée une seule fois (pas commode pour la transmission secrète de la clé aussi longue que le message): partager le message à la place de la clé.
- Les clés doivent être plus courtes que les messages à chiffrer et réutilisées plusieurs fois
 - Deux modes de chiffrement
 - Chiffrement par flux
 - Chiffrement par bloc

Chiffrement moderne: par flux

👉 C'est un chiffrement à la volée(bit par bit), pas besoin d'avoir la longueur du message pour commencer à chiffrer, il peut se faire avec un simple opérateur XOR



- Mode adapté pour la communication à temps réel, pas besoin d'attendre l'arrivée du bloc entier
- Implémenté en générale sur des supports matériels

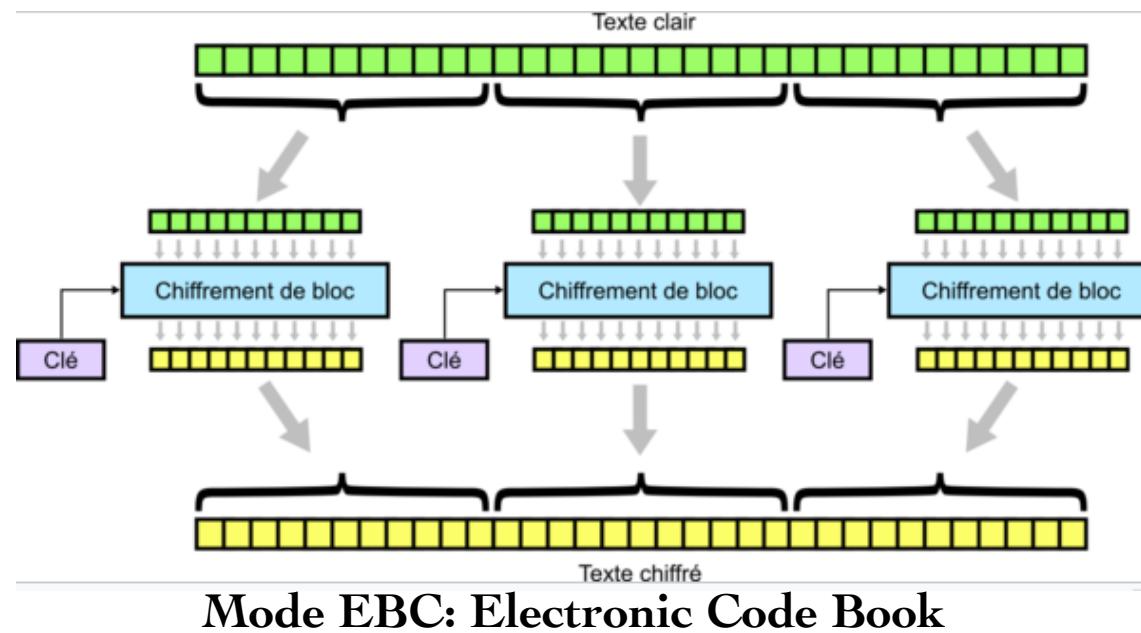
Chiffrement moderne: par flux

☞ Algorithmes très utilisés pour protéger les données multimédia.

- **RC4 (Rivest Cipher 4)**: plus utilisé, surtout en SSL et dans le Wifi 802.11
- **A5/1, A5/2, A5/3** utilisés en télécommunication dans le GSM
- E0 pour les liaison bluetooth

Chiffrement moderne: par bloc

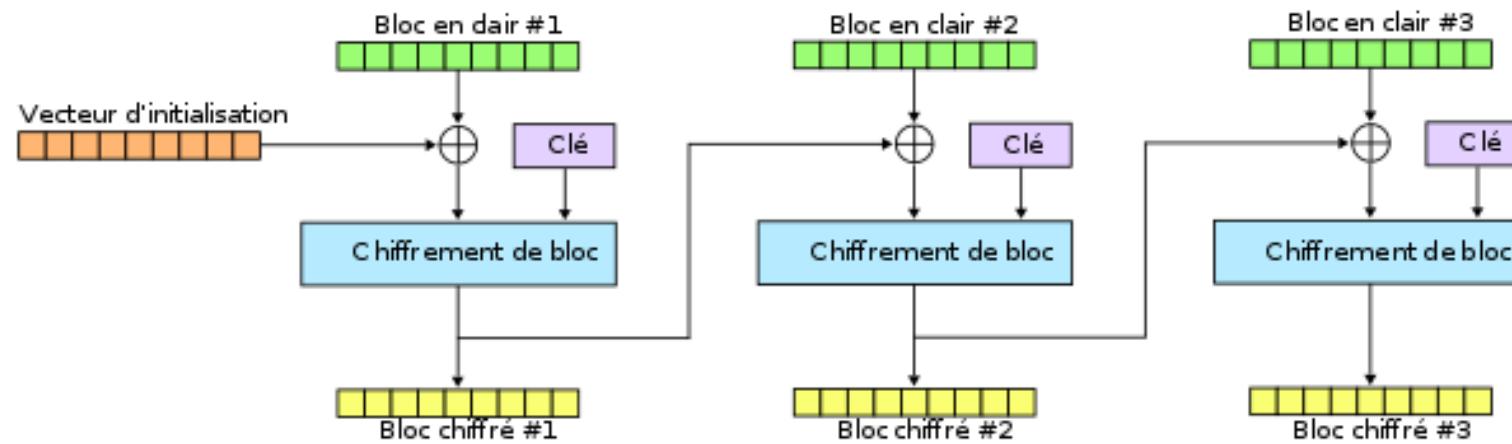
→ Texte divisé en blocs de taille fixe, et chaque bloc est chiffré donnant un bloc chiffré obtenu par substitution, permutation et combinaison avec XOR.



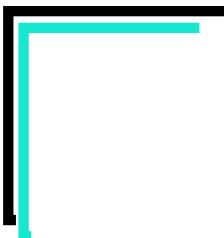
- chaque bloc doit être entièrement disponible avant le traitement
- La même clé et la même fonction sont utilisées pour chiffrer les blocs suivants

Chiffrement moderne: par bloc

☞ Enchaînement des blocs : « Cipher Block Chaining » (CBC): on applique sur chaque bloc un XOR avec le chiffrement du bloc précédent avant qu'il soit lui-même chiffré. Afin de rendre chaque message unique, un Vecteur d'Initialisation (IV) est utilisé.



Autres modes: Chiffrement à rétroaction : « Cipher Feedback » (CFB), Chiffrement à rétroaction de sortie : « Output Feedback » (OFB), Chiffrement basé sur un compteur : « CunTeR » (CTR), Chiffrement avec vol de texte : « CipherText Stealing » (CTS), Chiffrement par propagation des chiffrés en chaîne « Propagating Cipher Block Chaining » (PCBC), OU exclusif-Chiffrement-OU exclusif : « Xor-Encrypt-Xor » (XEX) etc...



Chiffrement moderne: par bloc

Algorithmes de chiffrement par bloc

- DES (Data Encryption Standard)
 - Triple DES
 - IDEA (Internation Data Encryption Algorithm)
 - RC5
 - AES (Advanced Encryption Standard)
 - Blowfish
- 

Résistance aux attaques

- La capacité de résistance d'un mécanisme cryptographique dépend:
 - 👉 L'algorithme choisi
 - 👉 La confidentialité de la clé
 - 👉 Le nombre clés
 - 👉 La taille des clés.
 - 👉 etc.....

Cryptographie symétrique

Principale problématique

Comment se partager la clé ?
Difficile à échanger?

- Rencontres personnelles
- Parties de clé envoyées par des canaux séparés

- Sans l'aide d'un tier, beaucoup de clés à gérer
- 2 personnes: 1 clés
- N personnes: $n*(n-1)/2$ clés
- 100 personnes: 4950 clés

Alternative:
Cryptographie à clé publique! ou échange de Diffie Hellman?

Cryptographie Asymétrique

Cryptographie asymétrique ou cryptographie à clé publique

☞ En cryptographie asymétrique, deux clés différentes et liées mathématiquement sont utilisées.

- **Une clé publique:** **connue de tous**

- Utilisée pour envoyer des messages chiffrés au propriétaire qu'il pourra déchiffrer avec la clé privée correspondante

- **Une clé privée:** **confIDENTIELLE**

- Permet de déchiffrer tout message chiffré avec la clé publique correspondante

☞ Il est facile de dériver la clé publique de la clé privée, l'inverse revient à résoudre un problème de calcul qui doit être difficilement résolvable selon le niveau de sécurité.



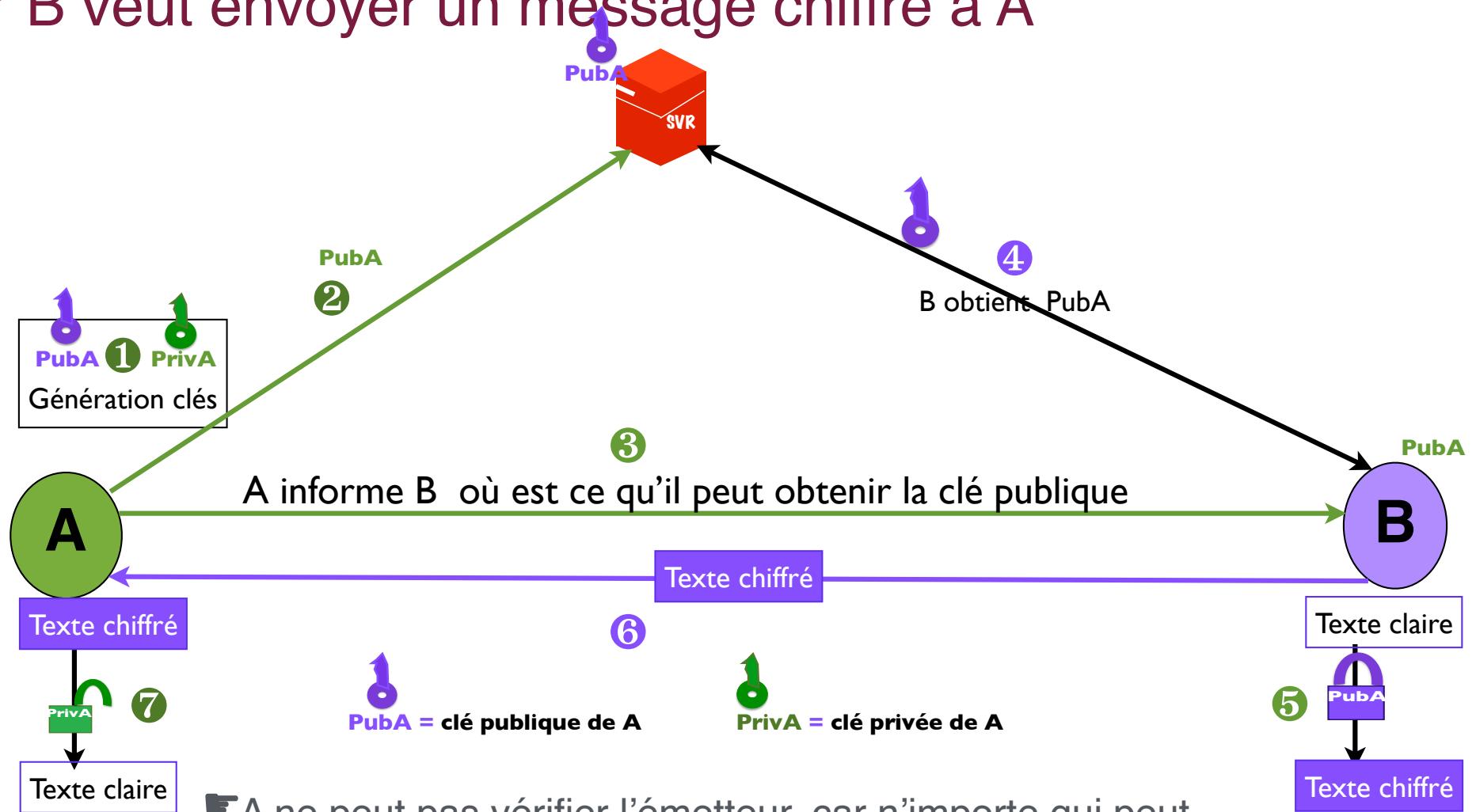
Cryptographie asymétrique chiffrer avec la clé publique

👉 En cryptographie asymétrique, le processus est initialisé par le récepteur pour l'envoi de message chiffré (confidentiel). Exemple: B veut envoyer un message à A / A veut recevoir un message de B

- Accord d'utiliser la cryptographie asymétrique pour chiffrer le message
- 1-A génère la paire de clés(clé publique de A et clé privée de A)
- 2-A garde la clé privée et donne à B la clé publique
- 3-B chiffre son message avec la clé publique de A
- 4-A reçoit le message et le déchiffre avec sa clé privée.

Cryptographie asymétrique chiffrer avec la clé publique

☛ B veut envoyer un message chiffré à A



☛ A ne peut pas vérifier l'émetteur, car n'importe qui peut disposer de sa clé publique et lui envoyer un message?

Cryptographie asymétrique

chiffrer avec la clé privée

👉 En cryptographie asymétrique, le processus est initialisé par l'émetteur pour l'envoi de message chiffré (non confidentiel): authentification.

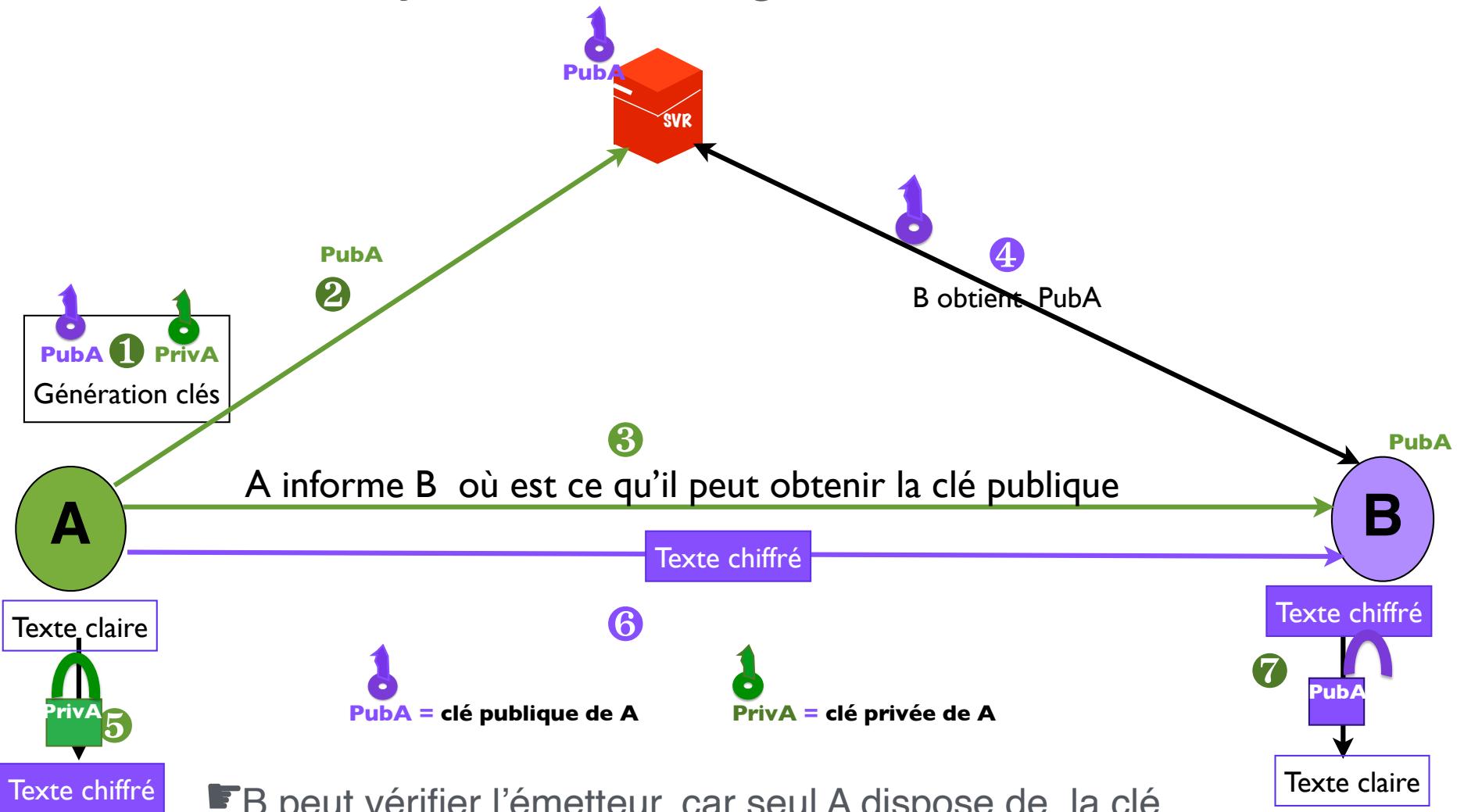
Exemple: A veut envoyer un message à B

- Accord d'utiliser la cryptographie asymétrique pour chiffrer le message
- 1-A génère la paire de clés(clé publique de A et clé privée de A)
- 2-A garde la clé privée et donne à B la clé publique
- 3-A chiffre son message avec sa clé privée
- 4-B reçoit le message et le déchiffre avec la publique de A.

Cryptographie asymétrique

chiffrer avec la clé privée

☞ A veut envoyer un message chiffré à B

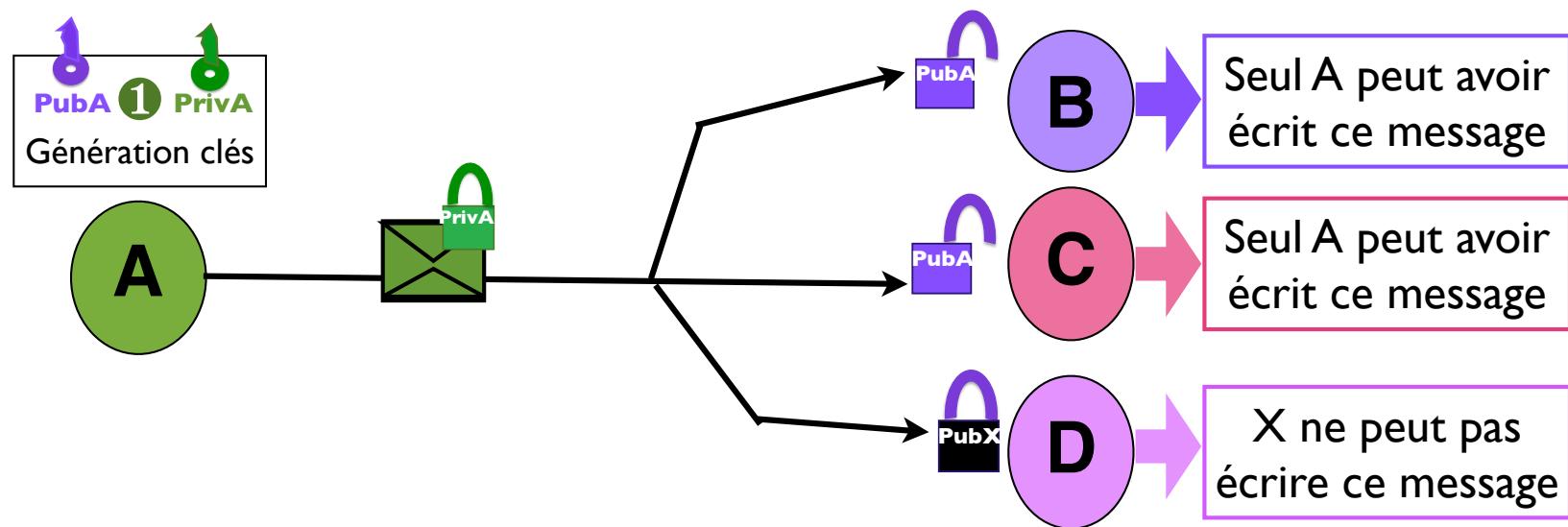


☞ B peut vérifier l'émetteur, car seul A dispose de la clé privée permettant de chiffrer ce message

Cryptographie asymétrique

chiffrer/déchiffrer avec une clé

☞ A veut envoyer un message chiffré à B

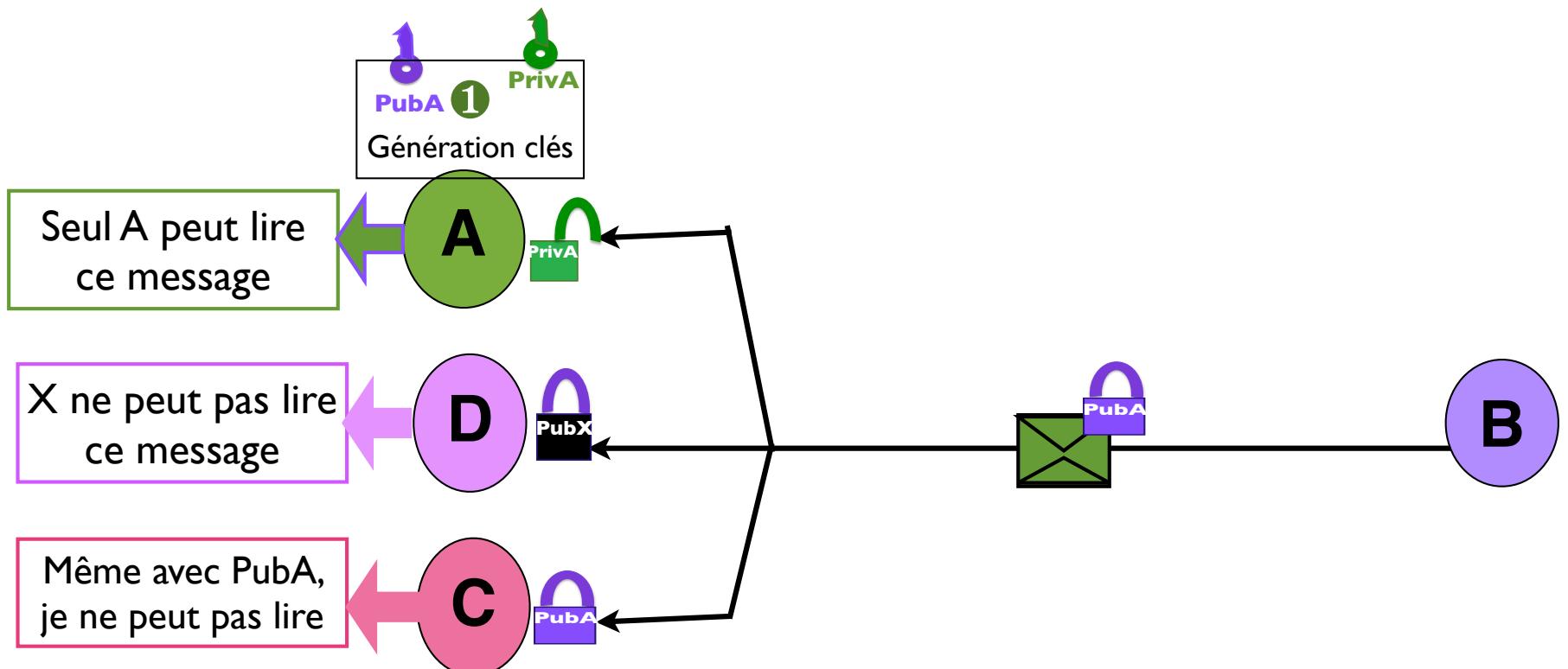


Authentification de A par B et C.
Le message est-il secret même s'il a été envoyé chiffré??

Cryptographie asymétrique

chiffrer/déchiffrer avec une clé

👉 B veut envoyer un message chiffré à A



Confidentialité du message
Qui l'a envoyé ???

Sécurité des algorithmes asymétriques

Le problème de la factorisation des entiers

- Etant donné deux nombres premiers p et q .
 - Facile de faire la multiplication $N = p \cdot q$
- Etant donné N
 - Difficile de trouver p et q (surtout si N grand)

- C'est la difficulté présumée de la factorisation des grands nombres
- **RSA (1977)** repose sur ce système
 - Génération de clé
 - Chiffrement/Déchiffrement/Signature/Vérification

Sécurité des algorithmes asymétriques

Le problème de la factorisation des entiers

- Exemple élémentaire: trouver les deux facteurs premiers des produits suivants
- $35 = 5 \times 7$
- $221 = 13 \times 17$
- $4453 = 61 \times 73$
- $503807 = 521 \times 967$
- $50123093 = 7297 \times 6869$
- Pour RSA-640 (193 chiffres décimaux ou 640 chiffres binaires)

Sécurité des algorithmes asymétriques

Le problème de la factorisation des entiers

- Pour RSA-640 (193 chiffres décimaux ou 640 chiffres binaires):
- Trouver les deux facteurs premiers de:
- $N=31074182404900437213507500358885679300373460228427275$
 $4572016194882320644051808150455634682967172328678243791$
 $6272838033415471073108501919548529007337724822783525742$
 $386454014691736602477652346609$
- $P=163473364580925384844313388386509085984178367003309231218$
 $1110852389333100104508151212118167511579$
 X
- $q=190087128166482211312685157393541397547189678996851549366$
 $6638539088027103802104498957191261465571$
- Équivalent à 30 années de calcul d'une machine 2.2GHz-Opteron-CPU [F. Bahr *et al.*, 2006]

Génération de clés avec RSA

FA veut générer une paire de clés (privée et publique)

Algorithm Génération de clés avec RSA

Entrées : l // taille (en bits) qui détermine le niveau de sécurité

Sorties : Clé publique (N, e) et clé privée d générées

begin

1. Sélection au hasard de deux nombres premiers p et q
2. Calculer le produit $N : pq$ et $\phi=(p - 1)(q - 1)$
3. Sélectionner arbitrairement un entier e tel que $1 < e < \phi$ et le PGDC (e, ϕ)= 1
4. Calculer d tel que $1 < d < \phi$ et $ed \equiv 1 \pmod{\phi}$
5. Retourner(N, e, d)
 ● PubA= (e, N) et PrivA= (d)

- Choisir deux nombres premiers secrets p et q soient 3 et 7
- Calculer le produit $N=p.q=> N=3x7 = 21$, et diffuser **N**
- Calculer $\phi(N)=(p-1)(q-1)=> \phi(N)2x6=12$
- Choisir **e** premier avec $\phi(N)$, soit **e**=5, premier avec 12, et diffuser **e**
- Calculer **d** inverse de $e \bmod \phi => ed=1 \bmod \phi, => 5d=1 \bmod 12=5x5=1 \bmod 12$ donc $d=5$
- Clé publique **PubA= (e, N)=(5,21)** et Clé privée **PrivA =d=5**

Chiffrement/déchiffrement avec RSA

B veut envoyer un message chiffré à **A**

- PubA= $(e, N)=(5, 12)$
- PrivA= $(d)=(5)$

① PubA= $(e, N)=(5, 12)$

Algorithm Chiffrement avec RSA

Entrées : (N, e) et m // la clé publique et le texte claire $m \in [0, N-1]$
Sorties : c , // le texte chiffré

begin

1. Calculer $c = m^e \text{mod } N$
2. Retourner(c)

- $m=2$, message à chiffrer doit être $< N$, sinon on le découpe en bloc $m_i < N$
- Calculer $c = m^e \text{mod } N = 2^5 \text{mod } 21 = 11$, puis transmettre $c=11$

② $c=11$

Algorithm déchiffrement avec RSA

Entrées : $(N, e), d$ et c // la clé publique, la clé privée et le texte chiffré
Sorties : m , // le texte claire

begin

1. Calculer $m = c^d \text{mod } N$
2. Retourner(m)

- Calculer $m = c^d \text{mod } N = 11^5 \text{mod } 21 = 2$

- $c^d \text{mod } N = (m^e)^d \text{mod } N = m^{ed} \text{mod } N = m \text{ mod } N$ car $ed = 1 \text{mod } \phi = 1$ d'après le théorème de Fermat-Euler

Sécurité des algorithmes asymétriques

Le problème du logarithme discret

- Etant donné deux nombres g et x .
 - Facile de faire $y = g^x$
- Etant donné y et g
- Difficile de trouver x tel que $y = g^x$

- C'est la difficulté présumée du logarithme discret
 - **Diffie-Hellman (1976) et El-Gamal (1984)** reposent sur ce système
 - Génération de clé (Diffie-Hellman)
 - Chiffrement/Déchiffrement (El-Gamal)
 - Signature/Vérification (DSA 1991 (NIST))

Génération de clés avec Le logarithme Discret

FA veut générer une paire de clés (privée et publique)

Algorithm Génération de clés avec le Logarithme Discret

Entrées : (p, q, g) // les paramètres publiques générés

Sorties : Clé publique (y) et clé privée x générées

begin

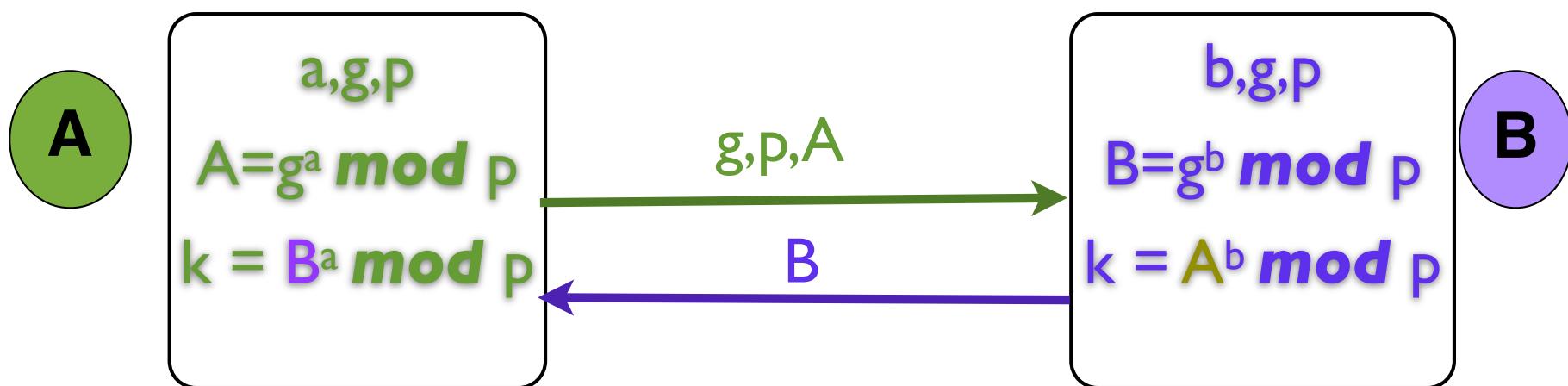
1. Sélection au hasard x dans l'intervalle $[1, q-1]$
2. Calculer $y=g^x \text{mod } p$
3. Retourner(x, y)

● PubA= y et PrivA= x

- p est un nombre premier et \mathbb{Z}^*_p le groupe cyclique de générateur $g \in \mathbb{Z}^*_p$
- $g \in [1, p-1]$ est d'ordre q , q est diviseur premier de $(p-1)$
- x la clé privée doit être choisi entre $[1, p-1]$
- $y=g^x \pmod{p}$.

Echange de clés avec Diffie-Hellman

- En cryptographie, l'échange de clés Diffie-Hellman, du nom de ses auteurs Whitfield Diffie et Martin Hellman, est une méthode, publiée en 1976,



$$K = A^b \text{ mod } p = (g^a \text{ mod } p)^b \text{ mod } p = g^{ab} \text{ mod } p$$

Chiffrement/déchiffrement avec El-Gamal

B veut envoyer un message chiffré à **A**

B

Algorithm Chiffrement avec ElGamal

Entrées : (p, q, g) , y et $m \in [0, p-1]$ // les paramètres publiques, la clé publique et le message $m \in [0, p-1]$

Sorties : (C_1, C_2) // le texte chiffré

begin

1. Sélectionner $k \in [1, q-1]$
2. Calculer $C_1 = g^k \text{mod } p$
3. Calculer $C_2 = m \cdot y^k \text{mod } p$
2. Retourner (C_1, C_2)

① PubA=Y

② c1c2

Algorithm Déchiffrement avec ElGamal

Entrées : (p, q, g) , x et C_1, C_2 // les paramètres publiques, la clé privée et le message chiffré

Sorties : m // message en claire

begin

1. Calculer $m = C_2 \cdot C_1^{-x} \text{mod } p$
2. Retourner (m)

$$\bullet C_1^{-x} = (g^k)^{-x} = (g^x)^{-k} = y^{-k} \Rightarrow C_2 C_1^{-x} \text{mod } p = m \cdot y^k y^{-k} \text{mod } p = m$$

A

Chiffrement/déchiffrement avec El-Gamal

B veut envoyer un message chiffré à **A**

- $m=1299$
- $\text{PrivB}=k= 853$

- $p=2579, g=2, x=765$
- $\text{PubA}= Y=g^x \bmod p$
- $Y=2^{765} \bmod 2579=949$
- $\text{PrivA}=x=765$



- $C_1=2^{853} \bmod 2579=435$
- $C_2=1299 \times 949^{853} \bmod 2579=2396$



- $C_1 C_2^{-x} = (2^{853})^{-765} \times 1299 \times 949^{853} \bmod 2579$
- $C_1 C_2^{-x} = (2^{765})^{-853} \times 1299 \times 949^{853} \bmod 2579$
- $C_1 C_2^{-x} = (949)^{-853} \times 1299 \times 949^{853} \bmod 2579$
- $C_1 C_2^{-x} = 1299 \bmod 2579 = 1299$

Sécurité des algorithmes asymétriques

Le problème du logarithme discret elliptique

- Etant donné un point P d'une courbe elliptique et un scalaire d .
 - Facile de calculer le point $Q = dP$
- Etant donné Q et P deux points de la courbe
 - Difficile de trouver le scalaire d tel $Q = dP$.

- C'est la difficulté présumée du logarithme discret elliptique
 - **Diffie-Hellman (1976) et El-Gamal(1984)** reposent sur ce système
 - Génération de clé (Diffie-Hellman)
 - Chiffrement/Déchiffrement (El-Gamal)
 - Signature/Vérification (ECDSA 1992 Scott Vanstone (NIST))

Génération de clés avec le logarithme Discret Elliptique

FA veut générer une paire de clés (privée et publique)

Algorithm : Génération de clés avec le Logarithme Discret Elliptique

Entrées : p, E, P, n // les paramètres publiques générés

Sorties : Clé publique (Q) et clé privée d générées

begin

1. Sélectionner au hasard d dans l'intervalle $[1, n-1]$
 2. Calculer $Q=dP$
 3. Retourner(Q, d)
-

- E est une courbe elliptique d'équation $y^2=x^3+ax^2+b \text{ mod } p$ (forme simplifiée)
- P est un point générateur d'ordre n (un nombre premier) de E d'(donc $nP=\infty$)
- p est un nombre premier
- d la clé privée doit être choisi entre $[1, n-1]$
- $Q=dP$ est la clé publique.

Chiffrement/déchiffrement avec El-Gamal

B veut envoyer un message chiffré à **A**

B

Algorithm Chiffrement ElGamal avec les courbes elliptiques

Entrées : $(p, E, P, n), Q_A$ et m // les paramètres publiques, la clé publique et le message clair m

Sorties : (C_1, C_2) // le texte chiffré

begin

1. Représenter le message m en tant que point $M \in E(\mathbb{F}_p)$
2. Sélectionner $k \in [1, n-1]$
3. Calculer $C_1 = kP$
4. Calculer $C_2 = M + kQ_A$
5. Retourner (C_1, C_2)

1 PubA= Q

2 c1c2

Algorithm Déchiffrement ElGamal avec les courbes elliptiques

Entrées : $(p, E, P, n), d$ et C_1, C_2 // les paramètres publiques, la clé privée et le message chiffré

Sorties : m // message en clair

A

begin

1. Calculer $M = C_2 - dC_1$, et extraire m de M
2. Retourner (m)

$$\begin{aligned} \bullet C_2 - dC_1 &= C_2 - dkP \\ \bullet &= M + kQ_A - dkP \\ \bullet &= M + kQ_A - kQ_A = M \end{aligned}$$

Les fonctions de hachage

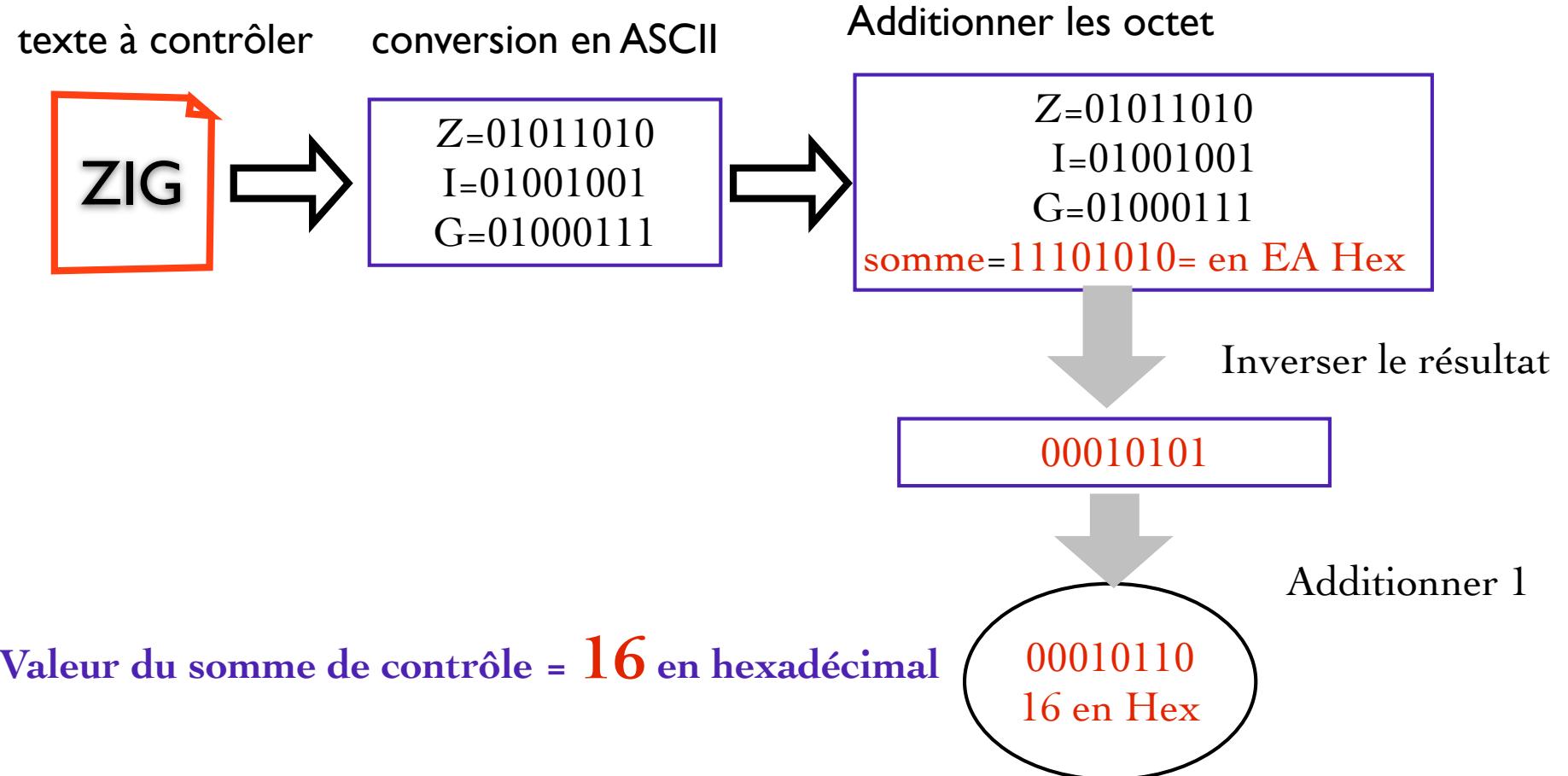
Usage de fonction de Hachage

- En cryptographie asymétrique, plus la taille des données est grande plus les opérations sont plus coûteuses
- Les fonctions de hachage permettent de réduire la taille des données par la création d'empreinte
- ✓ Une fonction de hache est un algorithme difficilement irréversible, qui, appliquée à des données produit une unique empreinte (hach ou condensé etc.). A partir de l'empreinte, il est difficile de retrouver les données. X



Usage de fonction de Hachage

☞ Algorithme de hachage simple: somme de contrôle de 8 bits.



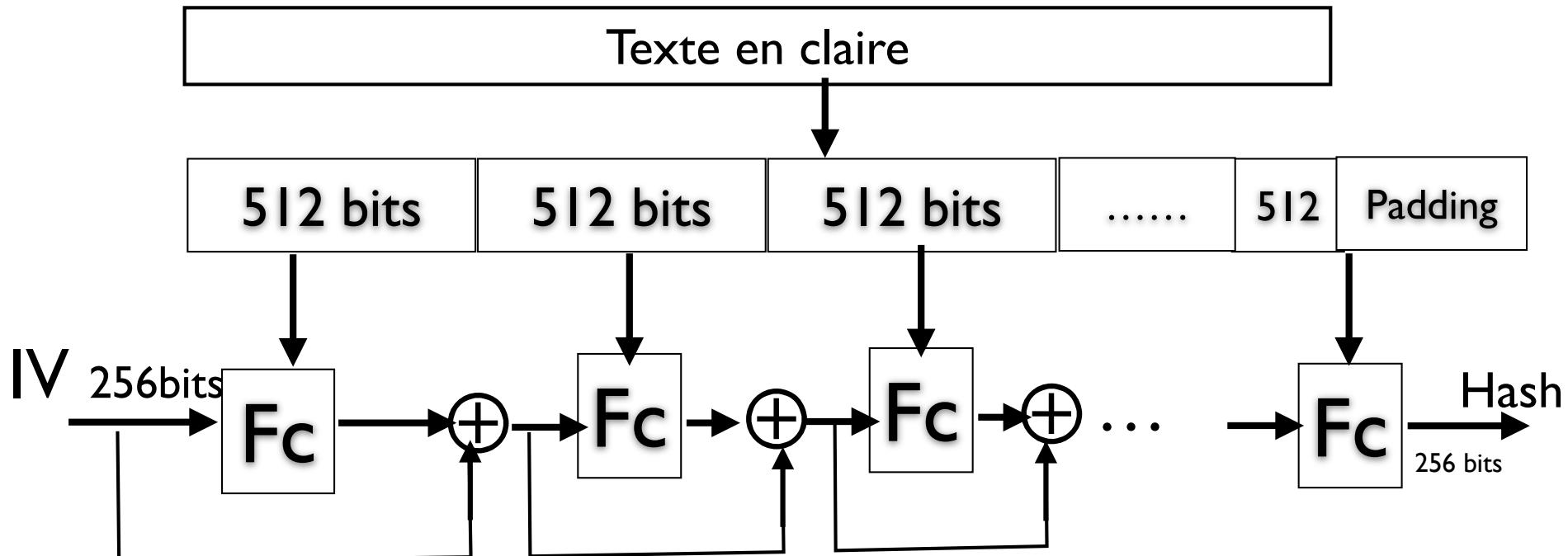
Usage de fonction de Hachage

- ☛ Hormis le caractère irreversible, une propriété très importante des fonctions de hachage est: la résistance au collisions
- ☛ Quelques algorithmes de hachage:
 - Construction de Merkle-Damgard
 - ☛ MD5 128 bits 1991 (empreinte de 128bits)
 - ☛ SHA1 160 bits 1995 (plus résistant au collision que MD5)
 - ☛ SHA2 256-512 bits 2001 (plus résistant au que SHA1)
 - Construction de Keccak
 - ☛ SHA2 256-512 bits 2001 (plus résistant au que SHA1)

Usage de fonction de Hachage

Construction de Merkle-Damgard

- Emploie une fonction de compression F avec une entrée et une sortie de taille fixe, et divise le message à hacher en blocs de taille fixe (bourrage si nécessaire sur le dernier bloc)

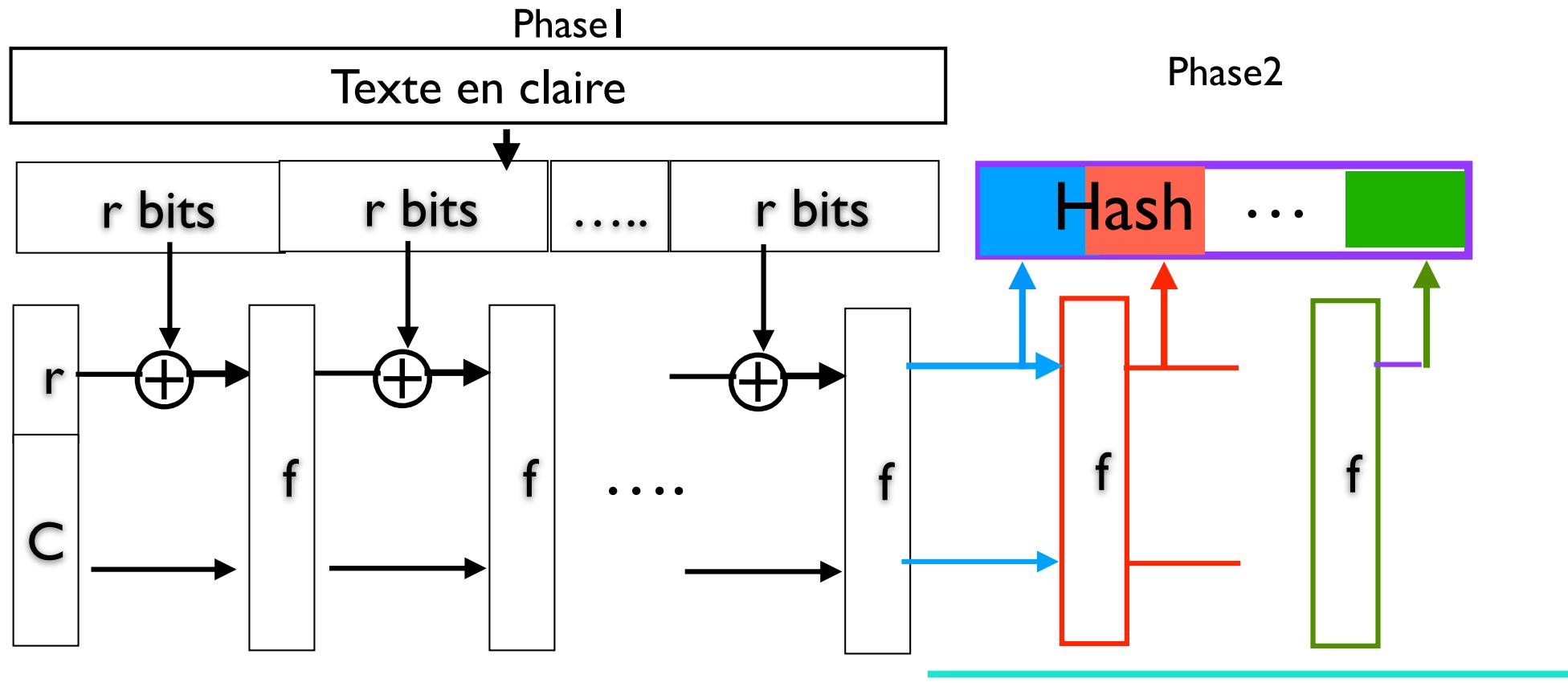


F_c est une fonction de compression qui peut être un même algorithme de chiffrement par blocs par exemple....

Usage de fonction de Hachage

Keccak ou Construction, de l'éponge

- Il fonctionne en deux étapes: on absorbe l'information, puis on l'essore pour obtenir le hash
- Découper l'entrée en longueur de r bits
- Faire un XOR entre le bloc 1 et les r bits de l'IV et donner le résultat en 1er entrée avec les c bits comme 2eme entrée de la fonction f (fonction de permutation pseudo-aléatoire)



Usage de fonction de Hachage

Application des fonctions de Hachage H

- ✓ Diminution de la taille des données
- ✓ Pour la comparaison et non pour le chiffrement

Propriétés de base

- ✓ Sécurité: fonction non réversible
- ✓ Taille fixe: les données longues et courtes produisent la même taille de données en sortie
 - Exemple: les données “ab” et “abcdefg” engendrent à la sortie de la fonction H la même taille 128 bits.
 - ✓ Des données différentes ne produisent pas la même empreinte
 - Exemple: les données “ab” et “abcdefg” n’engendent pas la même empreinte à la sortie de la fonction H

Attaque sur les fonctions de Hachage

■ Attaque de la pré-image

- ✓ Etant donné une empreinte, comment retrouver l'image correspondante appelée pré-image
- ✓ Attaque utilisée pour le crackage des mots de passe
 - ✓ Si on a une image y , une taille de hash de m bits
 - ✓ Le nombre de hash possibles est $n=2^m$
 - ✓ La probabilité de succès $p= 1/n$
 - ✓ Nombre d'essais est égale à 2^m
 - ✓ La longueur de l'empreinte est importante, plus elle est longue, plus c'est difficile de trouver une pré-image

Attaque sur les fonctions de Hachage

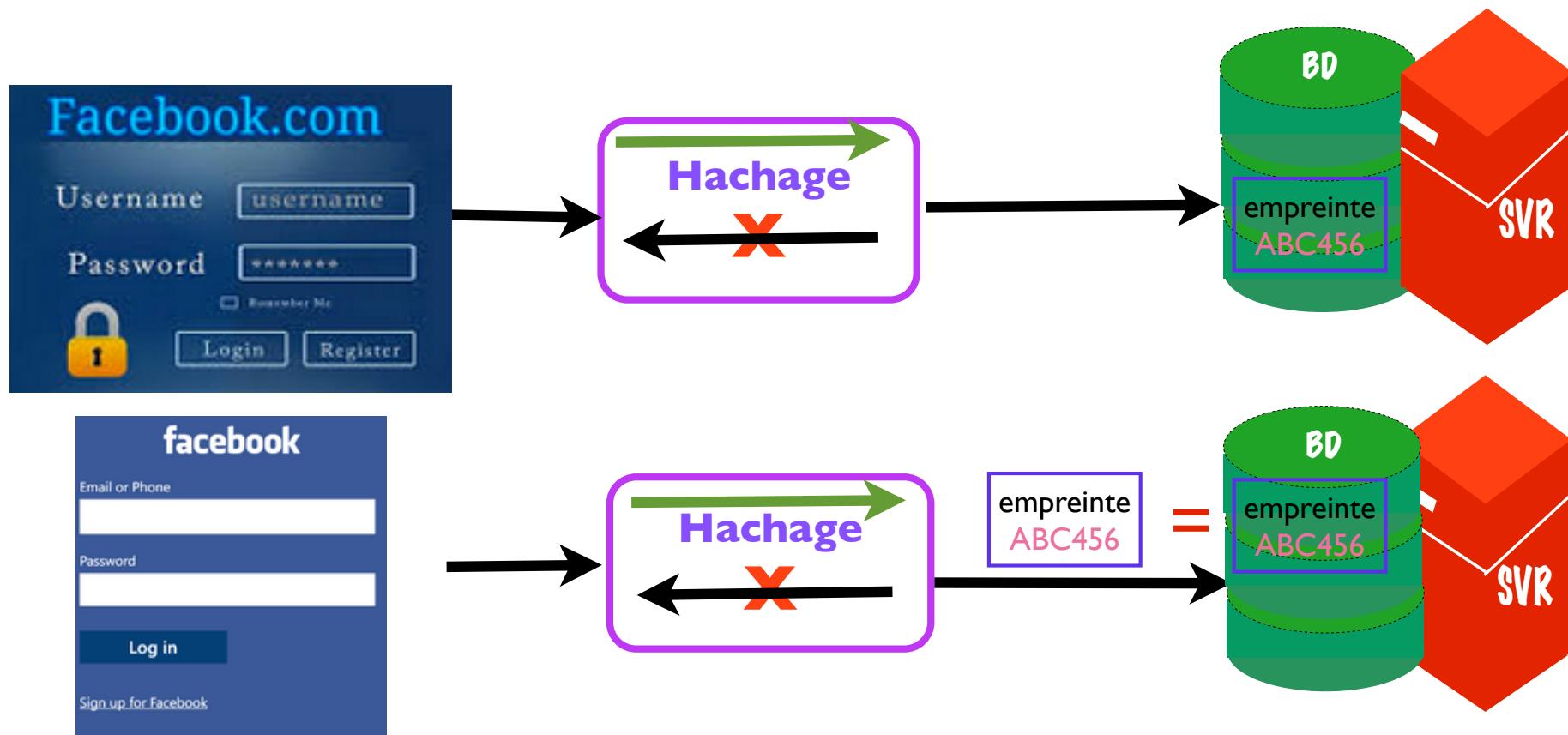
■ Attaque par collision

- ✓ Etant donné une empreinte $h(a)$, trouver b telle que $h(a)=h(b)$
- ✓ A chaque fois qu'on calcule une empreinte, voir s'il a déjà été calculé, si oui, il y a une collision
- ✓ Combien de valoir en moyenne doit-on considérer pour avoir une collision sachant qu'on a n empreintes différentes possibles: c'est la paradoxe des anniversaires (en probabilité)

- Nombre de hash: n
- Nombre moyen d'essais pour trouver une collision:
$$\sqrt{\frac{\pi}{2}n}$$

Avec fonction de Hachage

Application: Principalement utilisé pour les mots de passe



Les mots de passe doivent toujours être stockés cryptés.

Les signatures numériques

Cryptographie asymétrique

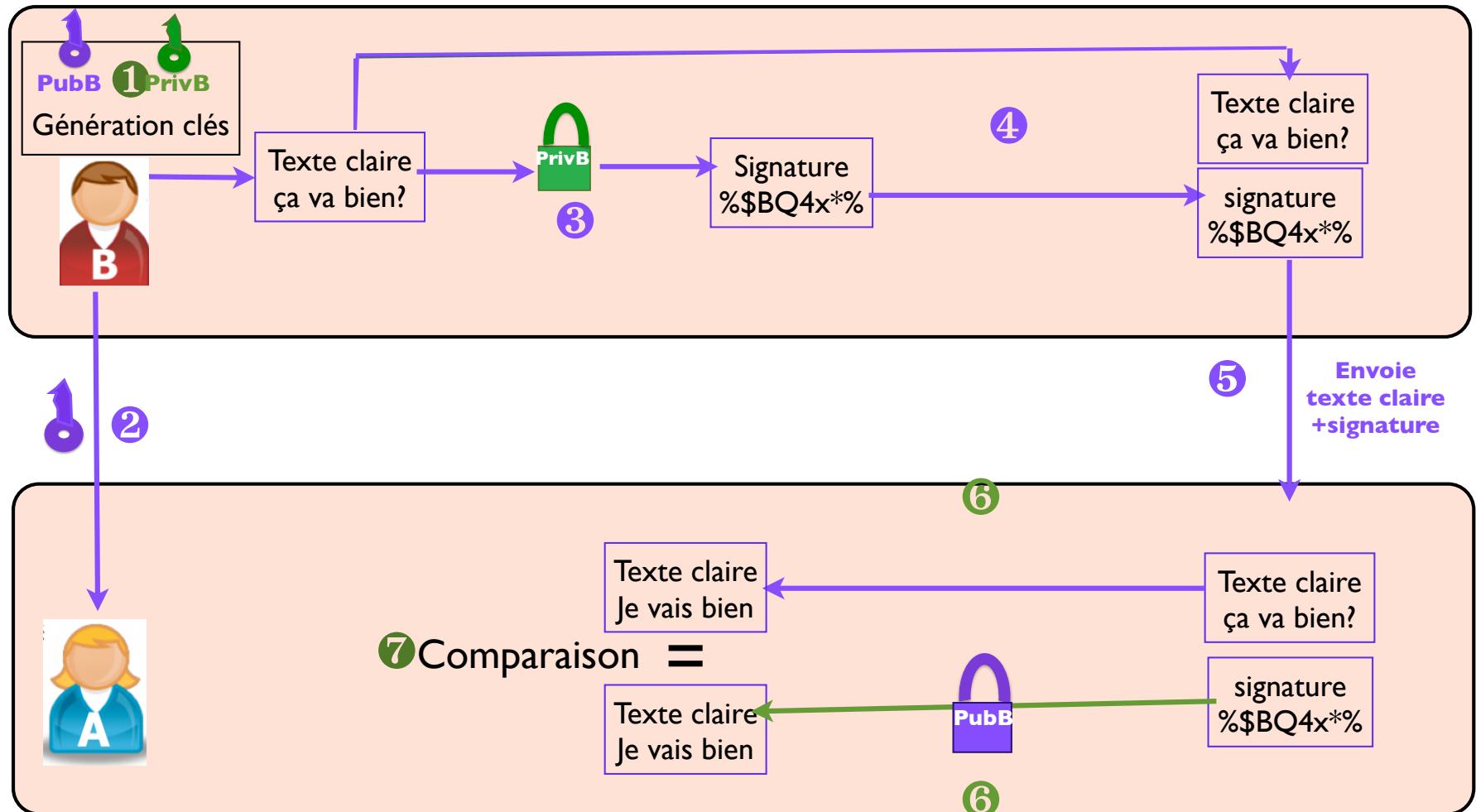
signature numérique

- ☛ La signature (SN) numérique est équivalente à la signature manuelle, c'est une vérification électronique de l'identité de l'émetteur:
- ☛ Elle offre trois services de sécurité
 - ✓ Authentification: SN permet au récepteur de vérifier (d'authentifier le message) que le message est envoyé par le supposé émetteur.
 - ✓ Non répudiation: avec SN, l'émetteur ne peut pas nier avoir émis le message
 - ✓ Intégrité: SN vous assure que le message n'a pas été modifié ou altéré durant sa transmission.
- ☛ SN est très utilisée pour la distribution de logiciels et les transactions financières
- ☛ SN est aussi populaire avec les transmission d'emails

Cryptographie asymétrique

signature numérique

■ B veut envoyer un message signé à A

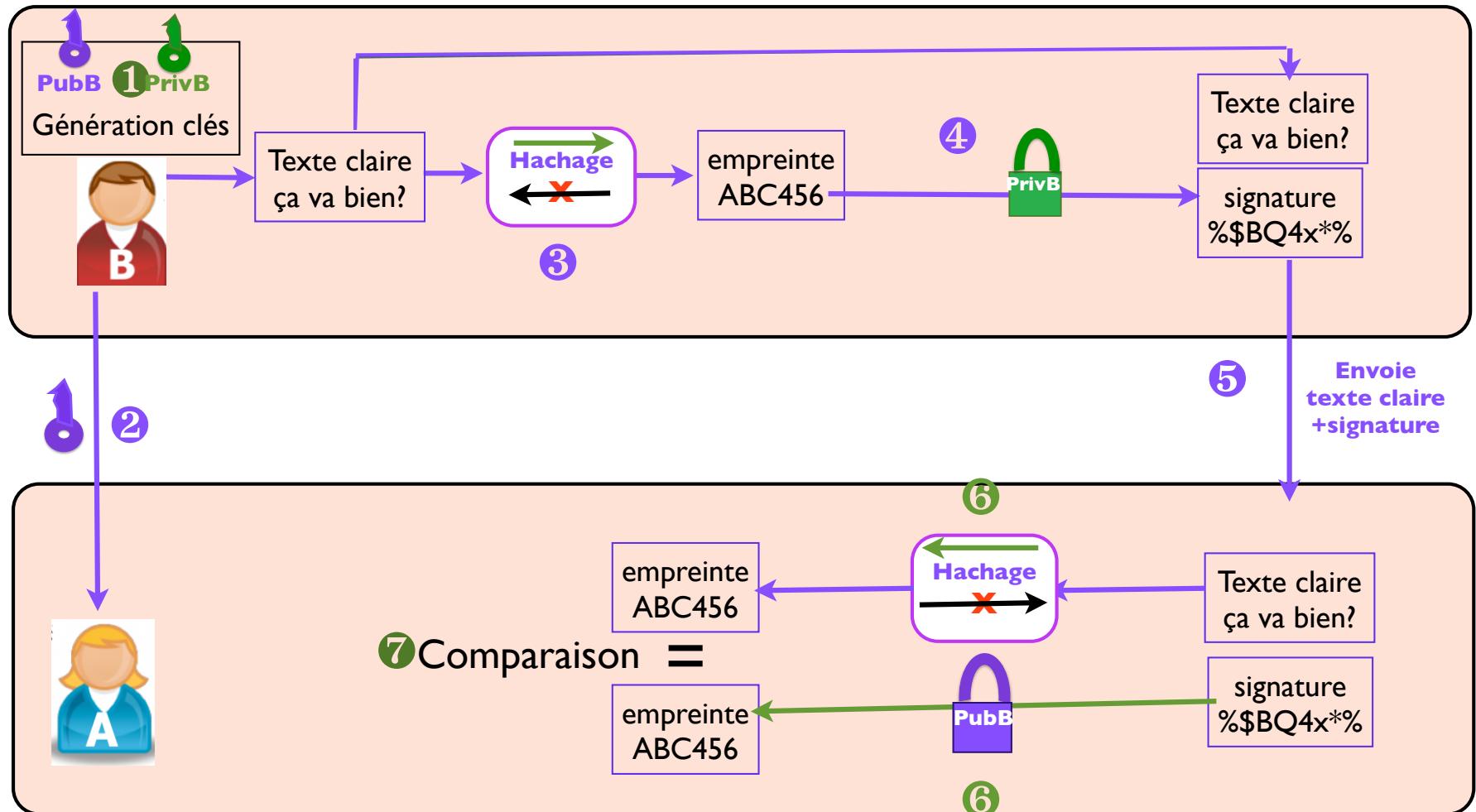


■ L'intérêt de la signature numérique n'est pas de crypter le message, car celui-ci est envoyé en clair.

Cryptographie asymétrique

signature numérique

■ B veut envoyer un message signé à A



■ L'intérêt de la signature numérique n'est pas de crypter le message, car celui-ci est envoyé en claire.

Signature Avec RSA

 A veut envoyer un message signé à B

- A envoie à B sa clé publique



$$\text{PubA} = (e, N)$$
$$\text{PrivA} = d$$

Entrées : (N, e) , d et m

Sorties : s ,

begin

1. Calculer $h = H(m)$
2. Calculer $s = h^d \bmod N$
3. Retourner(s)



$$\text{PubA} = (e, N)$$



Entrées : (N, e) , m et s

Sorties : Acceptation ou rejet s

begin

1. Calculer $h = H(m)$
2. Calculer $h' = s^e \bmod N$
3. Accepter si $h = h'$ et rejeter sinon

- Si on ne hache pas, A calcule $s = m^d \bmod N = s$ et envoie à B $m | s$ qui vérifie par $m' = s^e \bmod N$ et vérifie si $m = m'$

○ Démonstration

- $e.d \bmod \phi(N) = 1$
- $s^e \bmod N = (h^d)^e \bmod N = h^{de} \bmod N$
- $= h \bmod N = h'$

- Le calcul peut être très coûteux si on doit effectuer cette opération sur chaque octet du fichier à signer: donc on hache puis on signe pour réduire les opérations

Signature Avec el-Gamal

☛ A veut envoyer un message signé à B

- A doit envoyer à B sa clé publique

PrivA= a

PubA=A=g^a mod N



Paramètres publice: **g, N, q**
Paramètre privé: **a**
Message: **m**

PubA= A



$m | (r, s)$



V(A, (r,s), m)

Vérifier si $g^m \text{ mod } N = Ar \cdot rs$

$m = ar + sk$

$g^m \text{ mod } N = g^{(ar+sk)} \text{ mod } N$

$= g^{ar} \cdot g^{sk} \text{ mod } N$

$= (g^a)^r \cdot (g^k)^s \text{ mod } N$

$= Ar \cdot rs$

Choisir $k \in [1, q-1]$

Calculer $r = g^k \text{ mod } N$

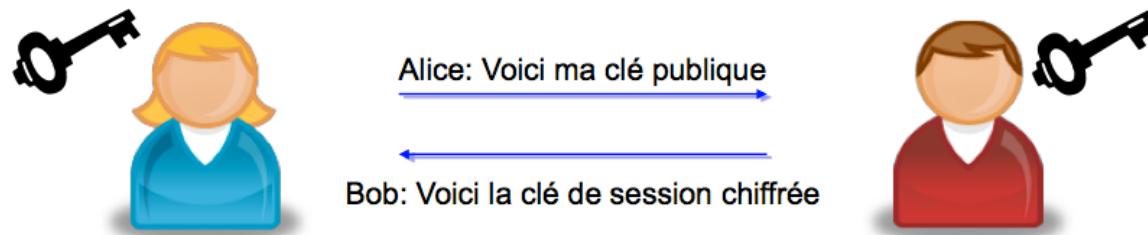
Calculer $S(\text{PrivA}, m) = (m - ar)k^{-1} \text{ mod } N = s$

○ Si on hache, A calcule $h = H(m)$, $r = g^k \text{ mod } N$,
 $s = (h - ar)k^{-1} \text{ mod } N$

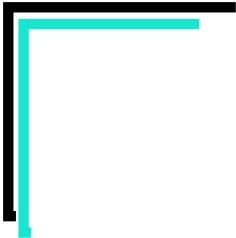
○ envoie à B $m | (r,s)$ qui vérifie si $g^h \text{ mod } N = Ar \cdot rs$

La gestion des clés (Echange de clés)

- Puisque les chiffrements à **clé publique** sont trop **coûteux**, ils servent généralement à **échanger** des **clés secrètes**.
- Génération et distribution de clés
 - Chiffrement **rapide** avec une clé secrète: clé de session
 - La clé de session est chiffrée grâce à un algorithme de cryptographie à clé publique tel que **RSA**.



- La clé de session peut aussi être obtenue grâce à un protocole d'échange de clé tel que **Diffie-Hellman**.

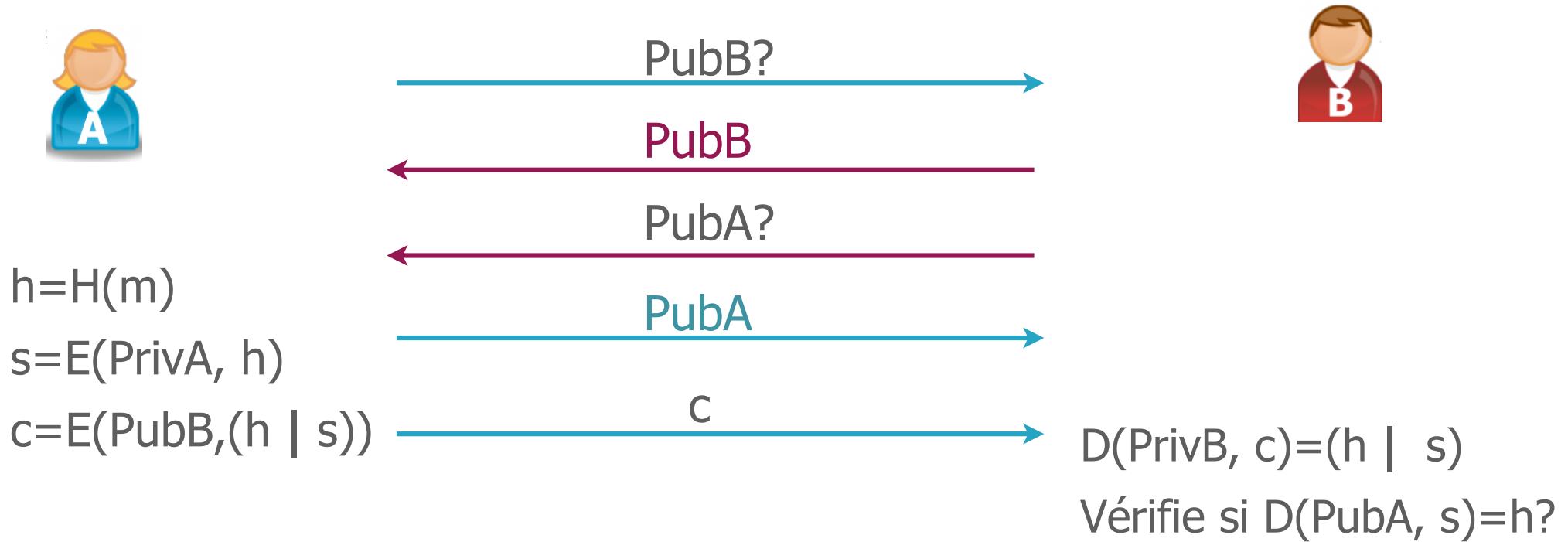


Les protocoles cryptographiques



Protocoles cryptographiques

- Chiffrement hybride: Confidentialité, Authentification, Intégrité
- A veut envoyer un message secret à B



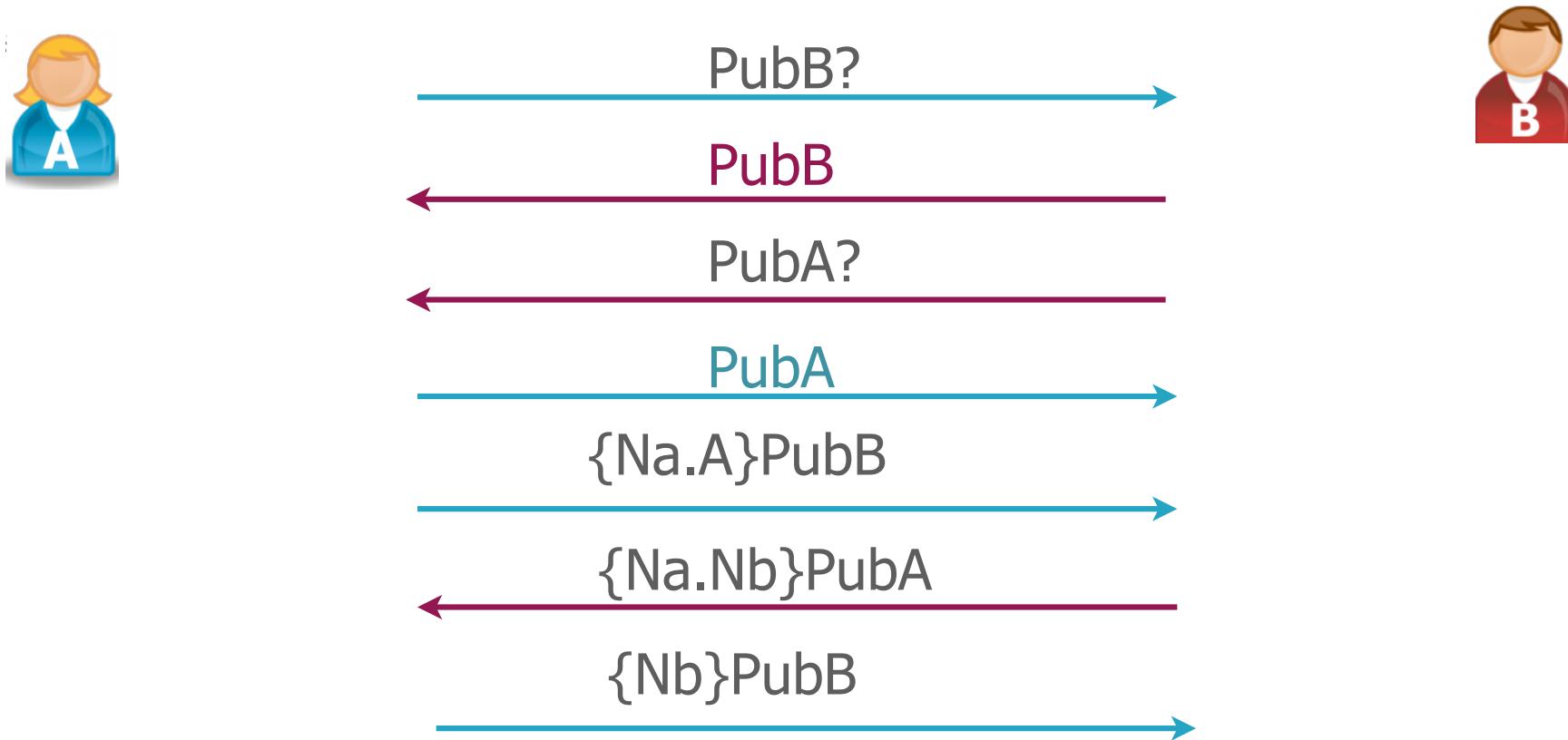
Chiffrer puis signer: $c=E(m)$ et $s=S(h(c))$, envoyer $c \mid s$

Chiffrer et signer: $c=E(m)$ et $s=S(h(m))$, envoyer $c \mid s$

Signer puis chiffrer: $s=S(h(m))$ et $c=E(m \mid s)$ et, envoyer c

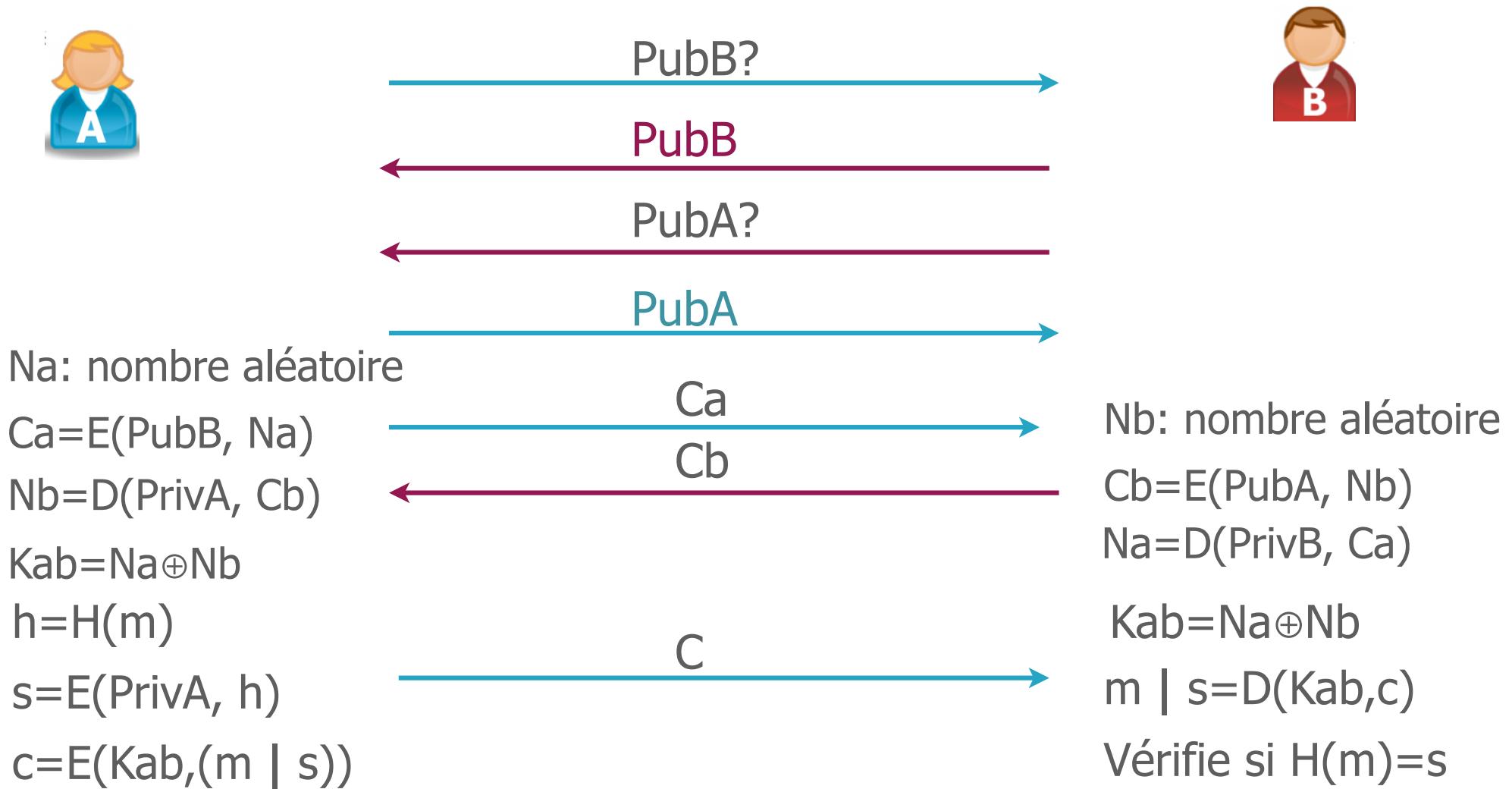
Protocoles cryptographiques

- Authentification mutuelle: Needham-schroeder
- A veut s'authentifier mutuellement avec B



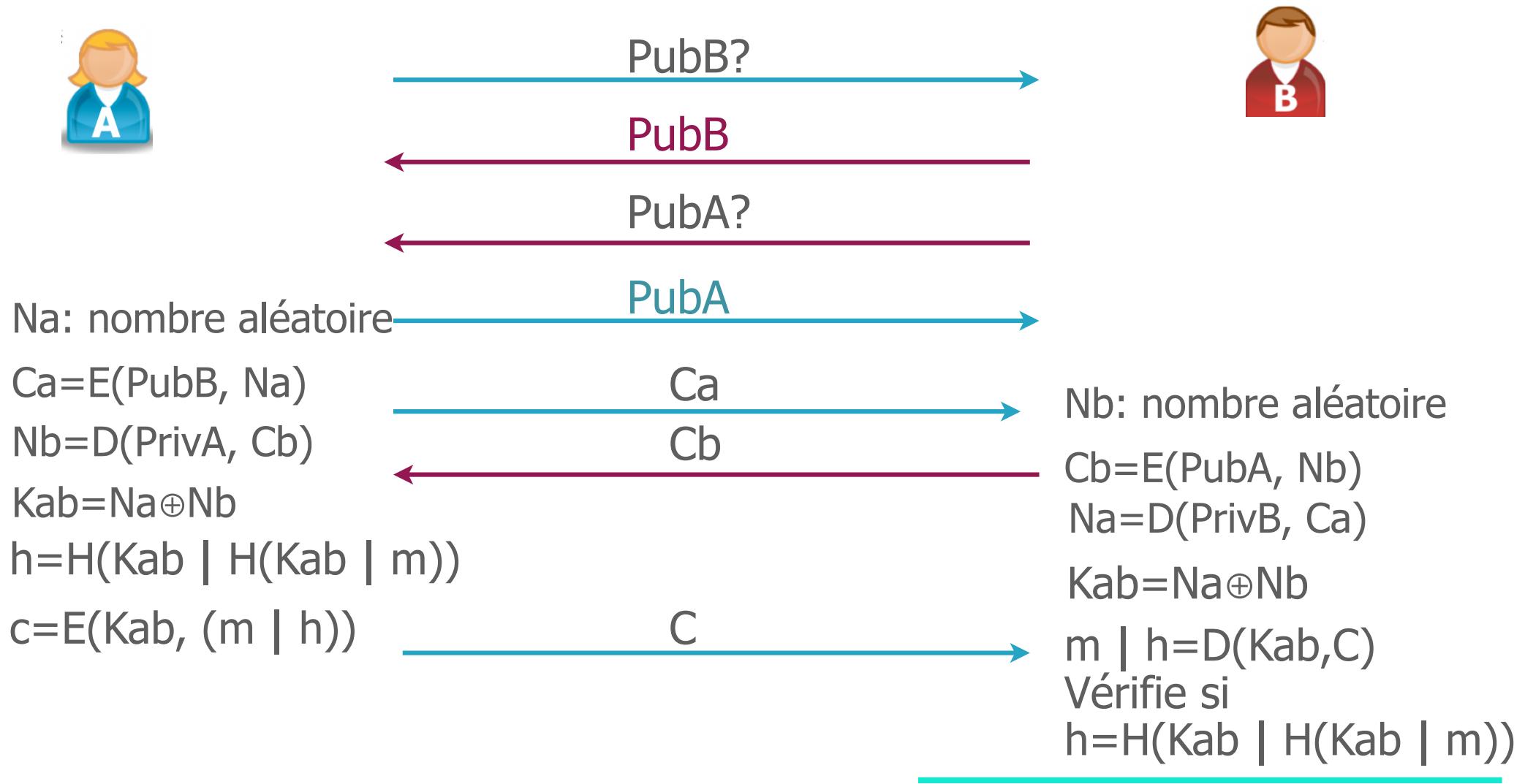
Protocoles cryptographiques

- Chiffrement hybride: Authentification de clé partagé
- A veut partager une clé symétrique avec B sans utiliser Diffie-Hellman



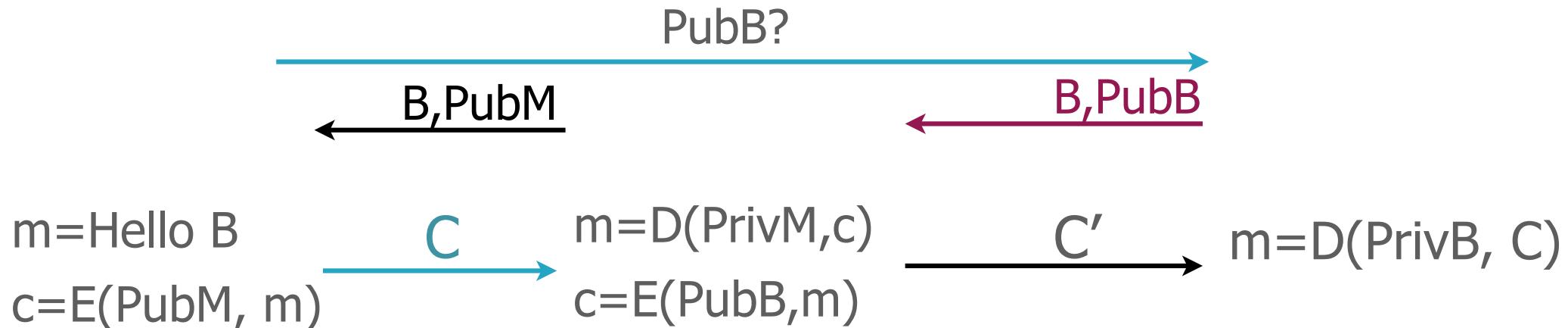
Protocoles cryptographiques

- Chiffrement symétrique pour l'authenticité et l'intégrité: HMAC
- A veut envoyer un message secret à B



Protocoles cryptographiques

- Attaque **Man In the Middle** sur l'authenticité, la confidentialité et l'intégrité
- A veut envoyer un message secret à B

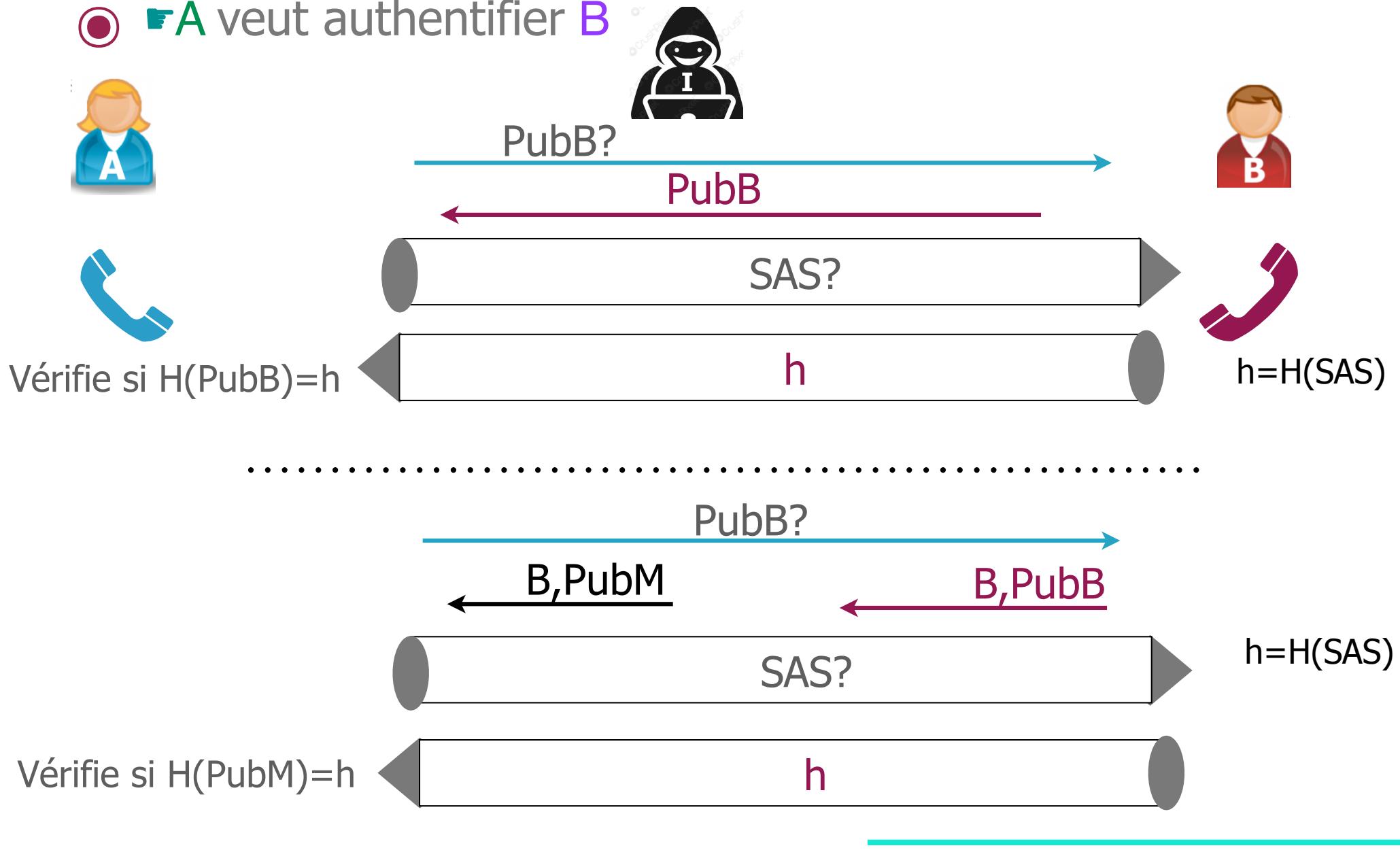


- Cette attaque n'est possible que si A et B sont à plus d'un saut l'un de l'autre
- Nécessité d'authentifier la clé publique
 - Short Authenticated Strings
 - Web of Trust
 - Certificat numérique

Authenticité de la clé publique

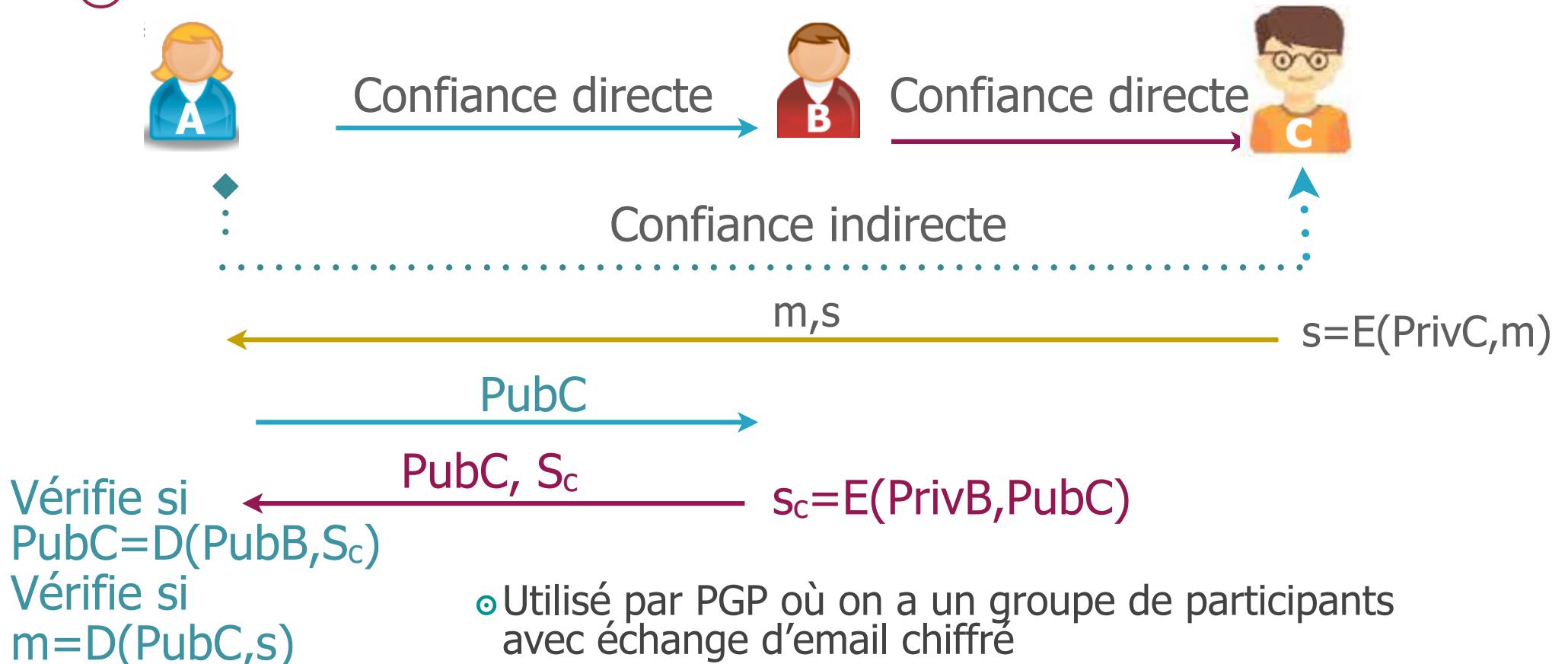
Authentification de clé publique

- SAS: utilise des chaines de caractères courtes via un canal secret
- A veut authentifier B



Authentification de clé publique

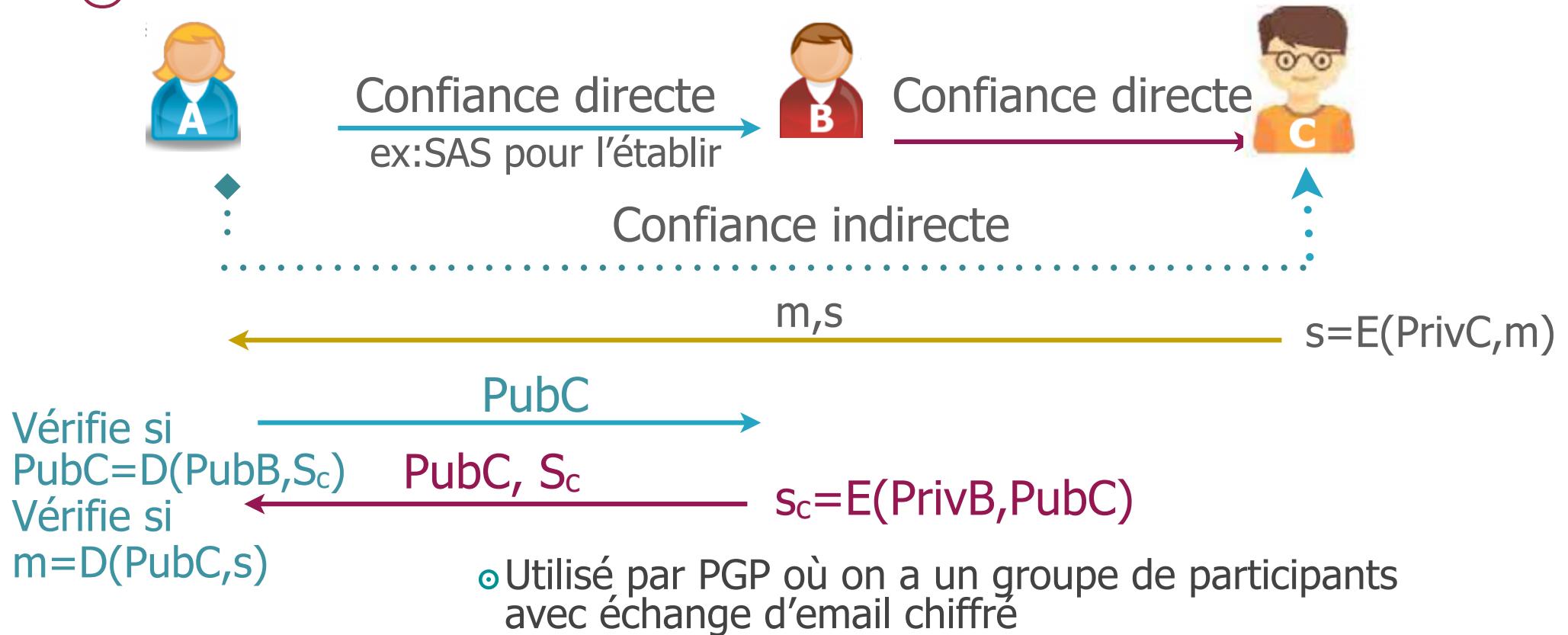
- **Web of Trust:** fonctionne selon le principe de confiance
- A veut authentifier C via B



- Utilisé par PGP où on a un groupe de participants avec échange d'emails chiffrés et signés: Web of Trust règle le problème d'authenticité de clés publiques

Authentification de clé publique

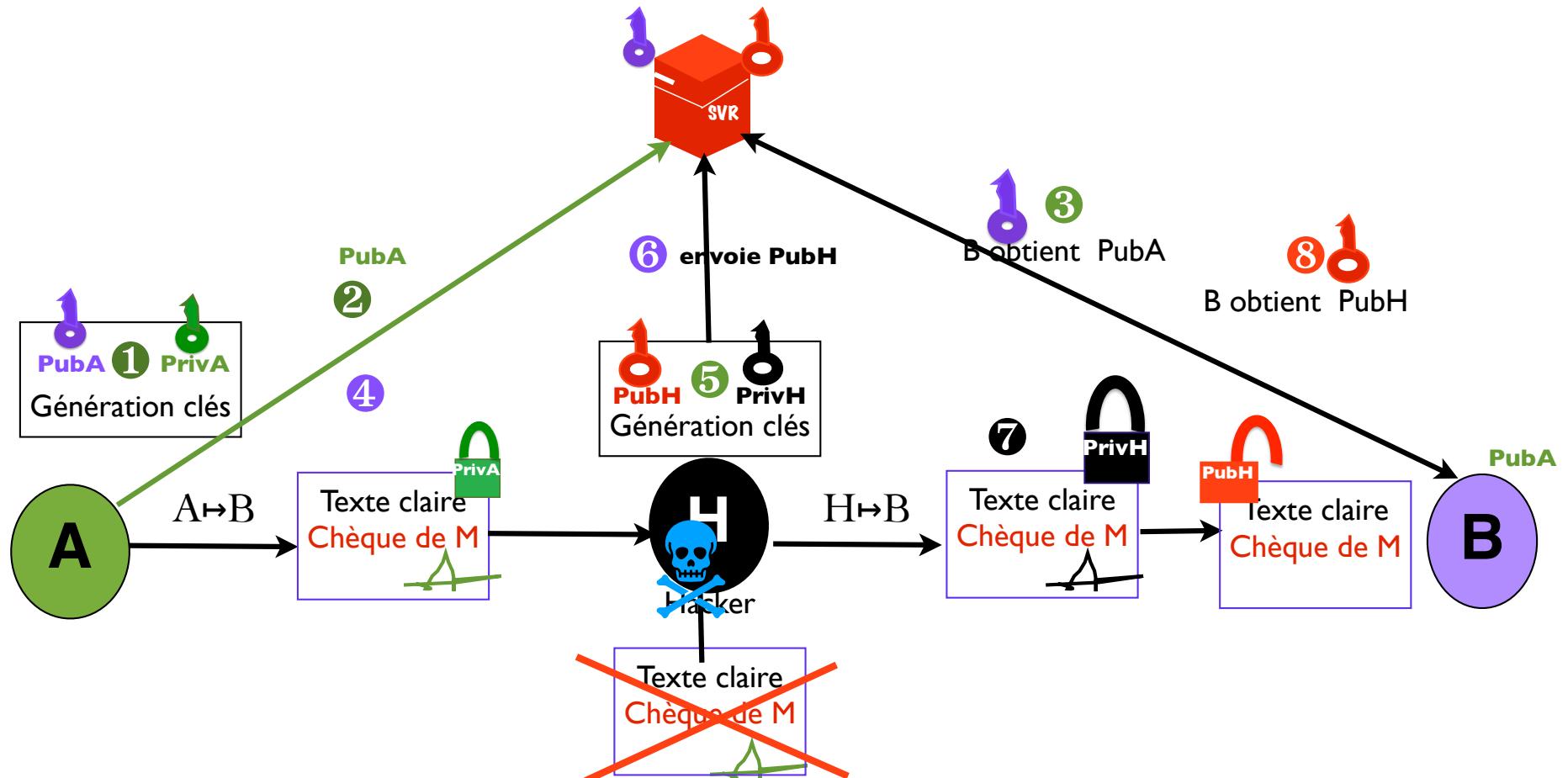
- **Web of Trust:** fonctionne selon le principe de confiance
- A veut authentifier C via B



- Utilisé par PGP où on a un groupe de participants avec échange d'emails chiffrés et signés: Web of Trust règle le problème d'authenticité de clés publiques
- PGP définit des règles pour la validité d'une clé publique
 - Signée par une seule personne (avec un niveau de confiance full),
 - Signée par trois personnes (avec un niveau de confiance marginale)
 - Le chemin jusqu'à la clé publique ne doit pas dépasser 5 personnes

Cryptographie asymétrique certificat numérique

- La signature numérique pose problème si les clés publique sont gardées dans un espace accessible par tous et sans mécanisme de gestion approprié



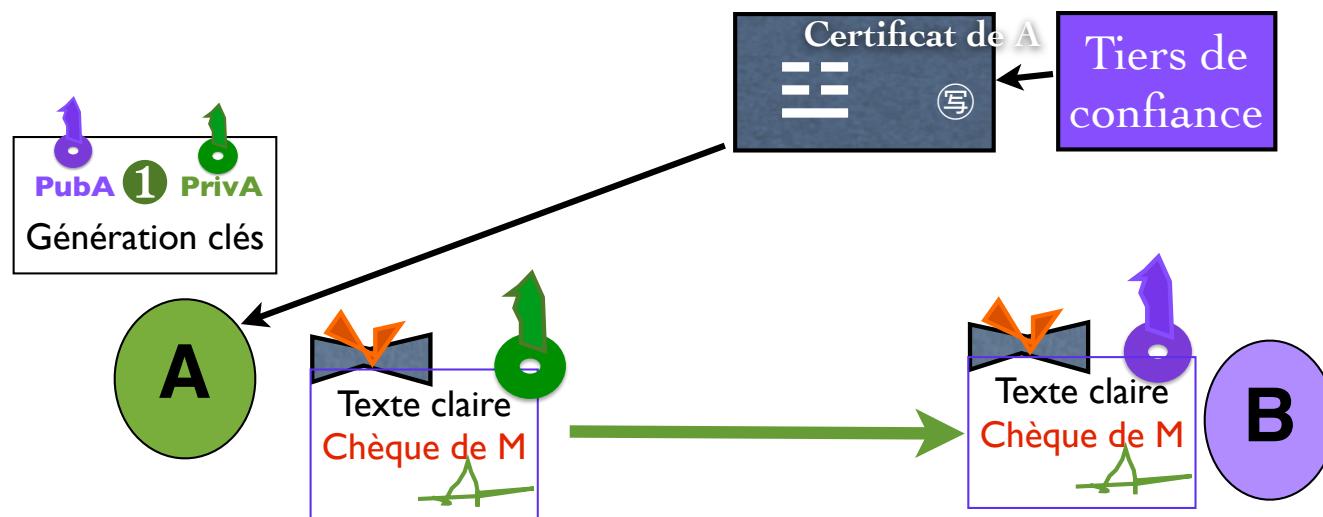
- La vraie identité de l'émetteur n'est pas vérifiée, B pense que le message vient de H. Le certificat numérique aide à résoudre ce problème.

Cryptographie asymétrique

certificat numérique

Le certificat électronique est composé d'identités issues d'une tierce partie, à qui on a confiance. Il vérifie les entités et leur clé publique associée.

✓ Quand A doit envoyer sa clé publique PubA à B, il ne demande pas à B de récupérer la clé PubA sur un site central, mais attache son certificat et PubA sur son message et envoie le tout (message signé+certificat+PubA) à B



- Informations contenues dans le certificat
 - ✓ Nom du propriétaire du certificat
 - ✓ La clé publique du propriétaire et sa date d'expiration
 - ✓ Nom du producteur du certificat
 - ✓ La signature numérique du producteur du certificat
 - ✓



Cryptographie asymétrique

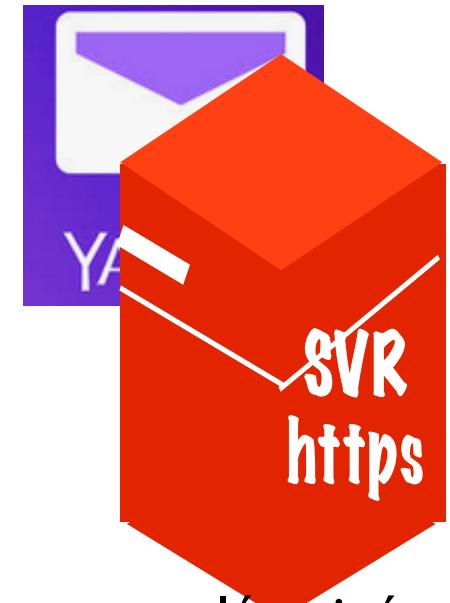
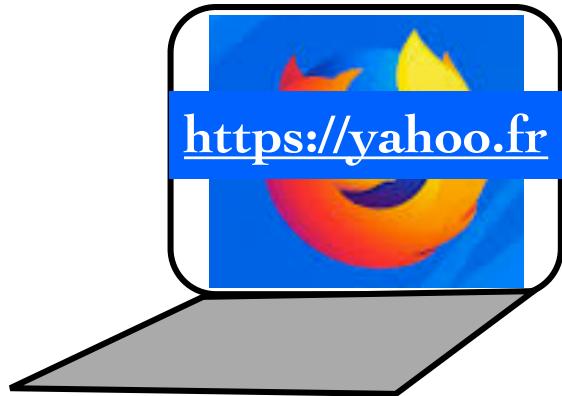
certificat numérique

- Le certificat numérique est basé sur la confiance ou une chaîne de confiance.
- ✓ Le tiers de confiance est une autorité de certification, une entité qui produit des certificats
 - ✓ Exemple: un passeport vérifie l'identité d'une personne si on a une confiance à l'organisme l'ayant délivré.
 - ✓ Nous faisons souvent appel à un notaire (tier de confiance) lors de la signature d'un contrat important
- ✓ Le certificat numérique vérifie l'identité de la clé publique du propriétaire i.e si l'entité prétendant signer le message est celle qu'elle prétend être.

Cryptographie asymétrique certificat numérique (SSL)

Les certificat SSL permettent d'établir la confiance navigateur-serveur-Web dans https.

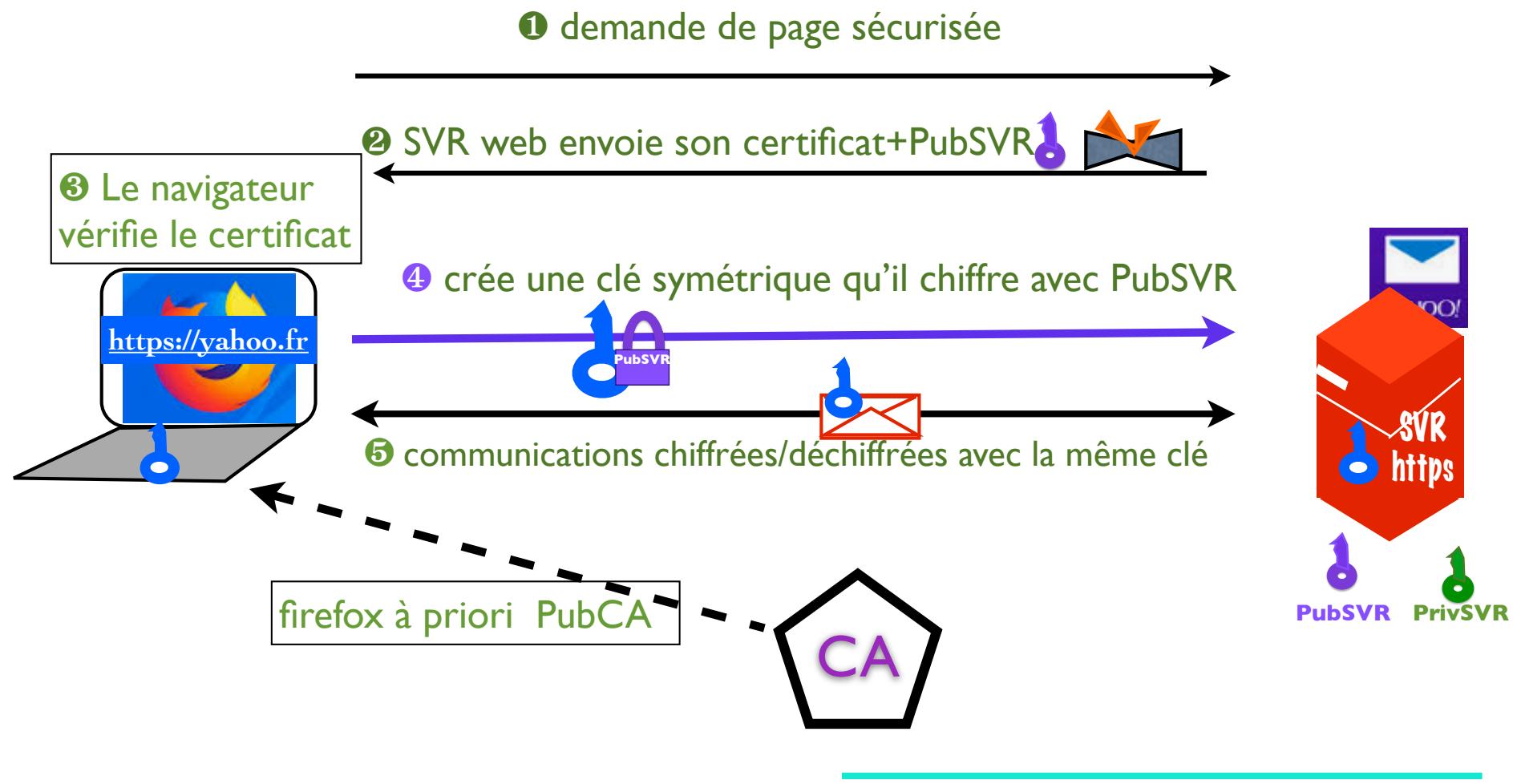
- ✓ Un certificat SSL, est un certificat numérique d'un serveur web délivré par une tierce parti de confiance (la CA) qui le signe
- ✓ Il vérifie l'identité du serveur et sa clé publique



- L'autorité de certification (CA) signe le certificat avec sa clé privée
- Le navigateur est à priori installé avec des clés publiques de CA, il peut vérifier les certificats

Cryptographie asymétrique certificat numérique (SSL)

https://yahoo.fr sur la barre d'adresse du navigateur, indique une demande de page sécurisée au serveur web



Cryptographie asymétrique certificat numérique (SSL)

- ☛ Le cadenas gris indique que la clé publique du serveur est à lui
- ☛ Les communications entre le browser et le serveur web sont chiffrées par une clé symétrique: SSL associe les deux formes cryptographique
- ☛ se connecter à un site avec https, indique que les communications sont chiffrées et non que le site est sûr, car aussi bien une personne bien intentionnée, un hacker peut obtenir un certificat SSL pour son site.
- ☛ Connecter vous à des sites avec une bonne réputation



Cryptographie asymétrique

certificat numérique (SSL) auto-signé

■ 2 méthodes pour la sécurité **serveur web-client web**

- 1. Payer un certificat SSL auprès d'une CA (comme Symantec, DigiCert etc.)
- 2 Crée librement son propre certificat SSL: **certificat auto-signé**
 - Le certificat auto-signé, **est signé par la même entité**
 - Il peut être créé en utilisant OpenSSL, java's keytools, Adobe Reader etc.

■ Certificat SSL auto-signé

- Crois moi, je suis celui que je prétend être
- Il n'est pas connu des browser



■ Certificat SSL commercialisé

- Crois moi, Symantec dit que je suis celui que je prétend être
- Il est connu des navigateurs



Cryptographie asymétrique certificat numérique (SSL) auto-signé

- Les certificats auto-signés ont le même niveau de sécurité que les certificats commercialisés, seulement ils sont moins réputés.
- Si un certificat auto-signé est distribué et installé sur les navigateurs qui accèdent à votre site, il est autant sécurisé qu'un certificat commercialisé.
- Le certificat auto-signé est limité: il marche bien pour un petit groupe de personnes qui utilisent votre site.

Sécurité des mots de passe

Sécurité des mots de passe

résistance aux attaques par force brute



Quelques mesures à prendre en compte faces aux attaques sur les mots de passe

- La robustesse (mot de passe fort, mot de passe faible)
- Mot de passe unique pour chaque compte
- Le changement périodique du mot de passe
- Etc....

Sécurité des mots de passe

résistance aux attaques par force brute



Robustesse des mot de passe

- Les mots de passe sont plus ou moins résistants aux attaques selon leur composition. Leur résistance aux attaques par force brute dépend:
 - ▶ plus il est long, plus il est résistant
 - ▶ Avec $26^6 = 308\ 915\ 776$ combinaisons d'attaques par force brute on trouve un mot de passe composé de 6 lettres alphabétiques, alors qu'avec 4 lettres, il en faut juste $26^4=456975$ combinaisons
- Une composition mixte (lettres, chiffres, caractères spéciaux, minuscules, majuscules etc..)
 - ▶ Comparé à un mot de passe de 6 lettres alphabétiques, un mot de passe de 6 lettres alphanumériques nécessite $36^6=2\ 176\ 782\ 336$ combinaisons d'attaques par force brute pour être connu

Sécurité des mots de passe

résistance aux attaques par force brute

