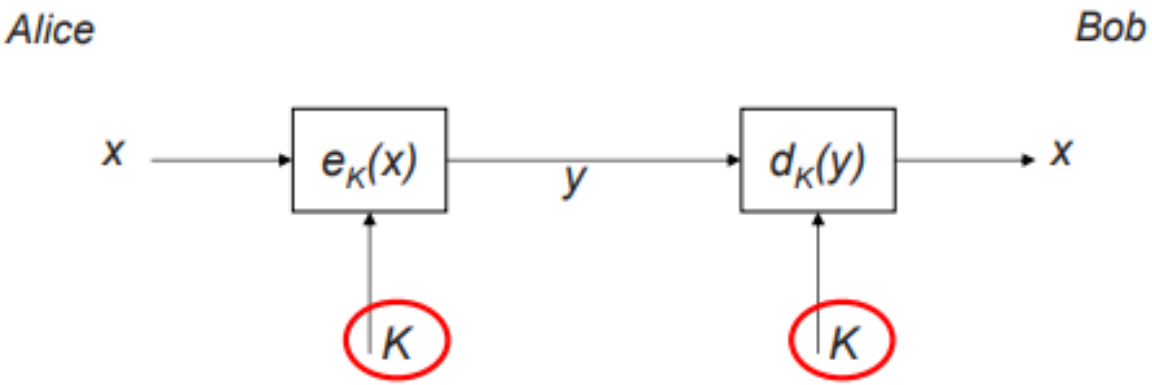
Chapitre IV
Chiffrement Asymétrique

Par Ilyas Bambrik

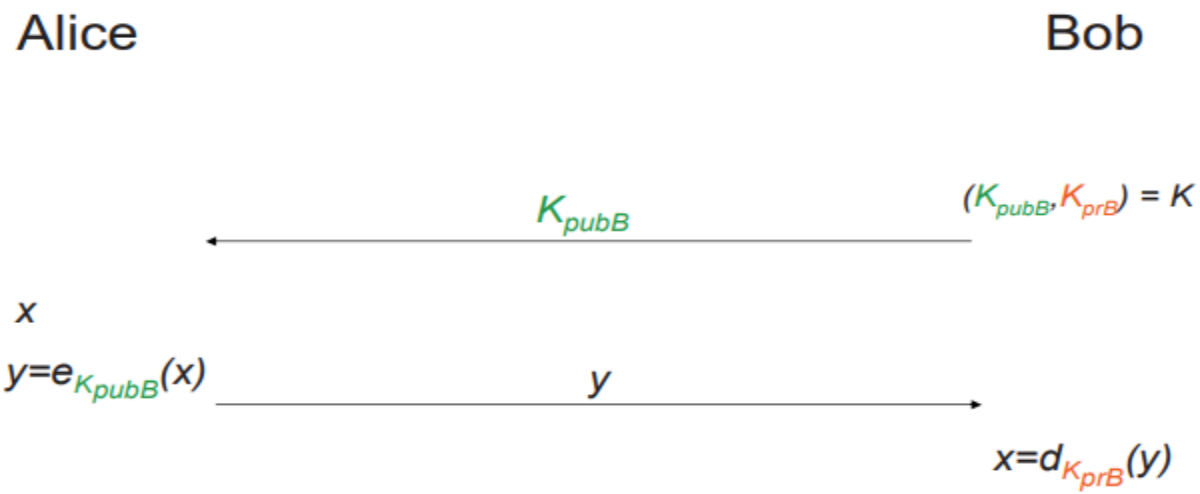
I. Problèmes de la cryptographie symétrique (cryptographie avec clé privée)

- Les deux parties qui participent à la communication doivent partager la clé privée d'une manière sécurisée.
- Chaque session de communication sécurisée nécessite une nouvelle clé (si on souhaite communiquer avec 10 personnes, on doit générer une clé pour chaque communication).
- **Solution :** Cryptographie avec clé publique. La clé utilisée pour le chiffrement (clé publique) est différente de la clé de déchiffrement (clé privée).



II. Chiffrement asymétrique

- Dans un système de chiffrement asymétrique, la clé publique est disponible publiquement (n'importe qui a accès à celle-ci)
- Si une personne souhaite transmettre un message chiffré, celui-ci obtient la clé publique et chiffre son message avec celle-ci avant de le transmettre.
- Même si tout le monde a accès à la même clé, seul le possesseur de la clé privée peut déchiffrer le message. Ainsi, une clé suffit pour communiquer avec plusieurs parties (contrairement au chiffrement symétrique).
- Ce type de chiffrement a été inventé par Diffie et Hellman



III. RSA

- L'algorithme de cryptographie asymétrique le plus utilisé est RSA.
- Inventé en 1976 et utilisé exclusivement par le gouvernement américain jusqu'à l'année 2000.

Algorithme RSA:

1. Sélectionne **deux nombres premiers** p et q aléatoires(p et q > 2^{512})
2. La première partie de la clé publique est N= p x q (N > 2^{1024})
3. Calcule de Φ(N) =| *Nombre premiers par rapport à N && inférieurs à* **N**| = (q-1) x (p-1)
4. Choisir un chiffre **E** t.q.: 2<=E <= Φ(N)-1 et GDC(Φ(N),E) == 1 (E et Φ(N) sont premiers)
5. Calculer l'inverse de E par rapport à Φ(N): (E x D) modulo Φ(N) == 1 (D est l'inverse de E)
6. **(N,E)** et **(N,D)** sont respectivement la clé publique et la clé privée.

Chiffrement/ déchiffrement RSA

- Pour chiffrer une partie d'un message X tel que X < N (N représente la première partie de la clé de chiffrement), il suffit de calculer:

$$X^E \text{ modulo } N = Y$$

- Afin de déchiffrer Y par le récepteur, il faudra calculer:

$$Y^D \text{ modulo } N = X$$

- La puissance de RSA repose sur le fait que pour un chiffre de N supérieur à 2^{1024} , il n'existe pas d'algorithme pour factoriser celui-ci rapidement (Si q et p sont connues par une partie non autorisé, le chiffrement est cassé)

Exemple Chiffrement RSA

- P = 17 , Q = 11
- N = 17 x 11 = 187
- $\Phi(N) = (17 -1)x(11 -1) = 160$
- $E \epsilon \{2 \ldots \Phi(N) \}$ t.q. GDC(E, $\Phi(N)$) =1
- Supposant que E = 151 [GDC(151,160) =1]
- $D=E^{-1}= 71$ (voir comment retrouver l'inverse de E dans la section suivante)
- Clé de chiffrement = (187, 151)
- Maintenant pour chiffrer un message X= 142 (142< N):

  X^E modulo N = 142^{151} modulo 187 = 65
- Pour déchiffrer un message Y= 65:

  Y^D modulo N = 65^{71} modulo 187 = 142

Calcule de l'inverse modulo

- **Rappel:** Pour retrouver l'inverse (A^{-1}) d'un nombre A par rapport au modulo M, A doit être premier avec M. Trois méthodes existent:

1. **Brute force :** Tester touts les produits A x B modulo M avec B < M.
2. **Théorème de Fermat:** Si GDC(A, M) =1 alors $A^{\Phi(M)} = A^{-1}$
3. **Algorithme Euclidien étendu**

Fast Exponential Algorithm

- Le problème du chiffrement Symétrique réside dans le calcul de X^E avec E un entier large. Ainsi pour chaque bloque de valeur < N, le même calcul est répété.
- Le résultat est un chiffrement plus long que le chiffrement symétrique car le chiffrement symétrique s'effectue sur des bits ou des octets.
- Pour calculer X^E et Y^D plus rapidement une simple méthode existe pour le calcul de l' exponentiel (Square and Multiply).

Fast Exponential Algorithm

```
Fast Exponential Algorithm(x,n):
Calculer la représentation binaire de n
N={ b_m,b_m-1,b_m-2, …,b_0} [b_i = 0 ou 1]
P=1
Pour i = m jusqu'à 0:
      P=P * P // Square (Etape 1a)
      si b_i == 1:
            P= P * x // Multiply (Etape 2a)
Retourner P
```

Exemple

- Calcule de $2^{10} = 128$
- Etape initiale

  Représentation binaire de 10 = 1010

  P= 2

Itération	Etape	Valeur de l'exponent	Valeur	Opération
1	1a	10	4	Square
1	2a	-	4	Multiply
2	1a	100	16	Square
2	2a	101	32	Multiply
3	1a	1010	1024	Square
3	2a	-	1024	Multiply

Exemple

- Calcule de $3^{13}=1594323$
- Etape initiale

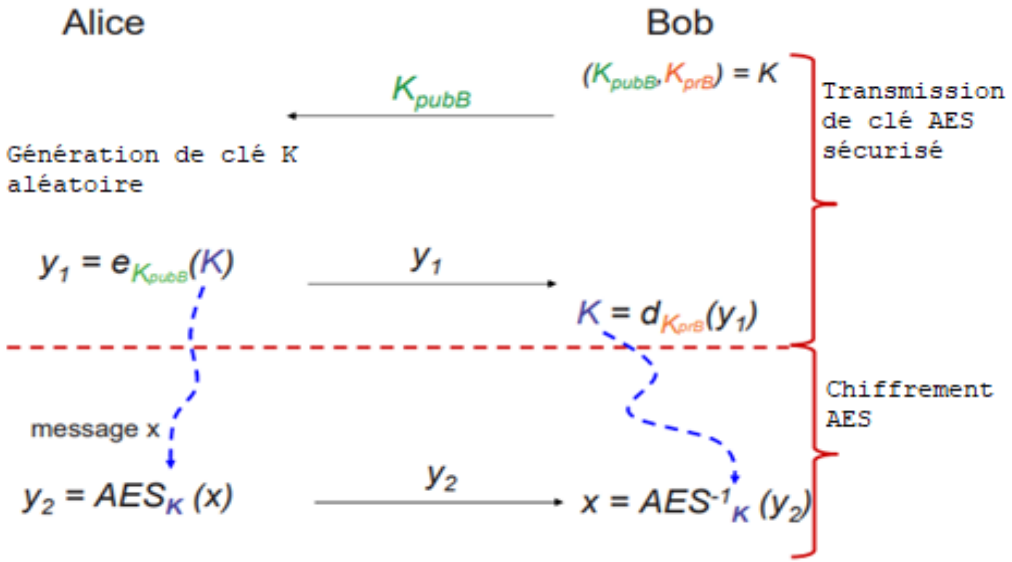
  Représentation binaire de 10 = 1101

  P= 3

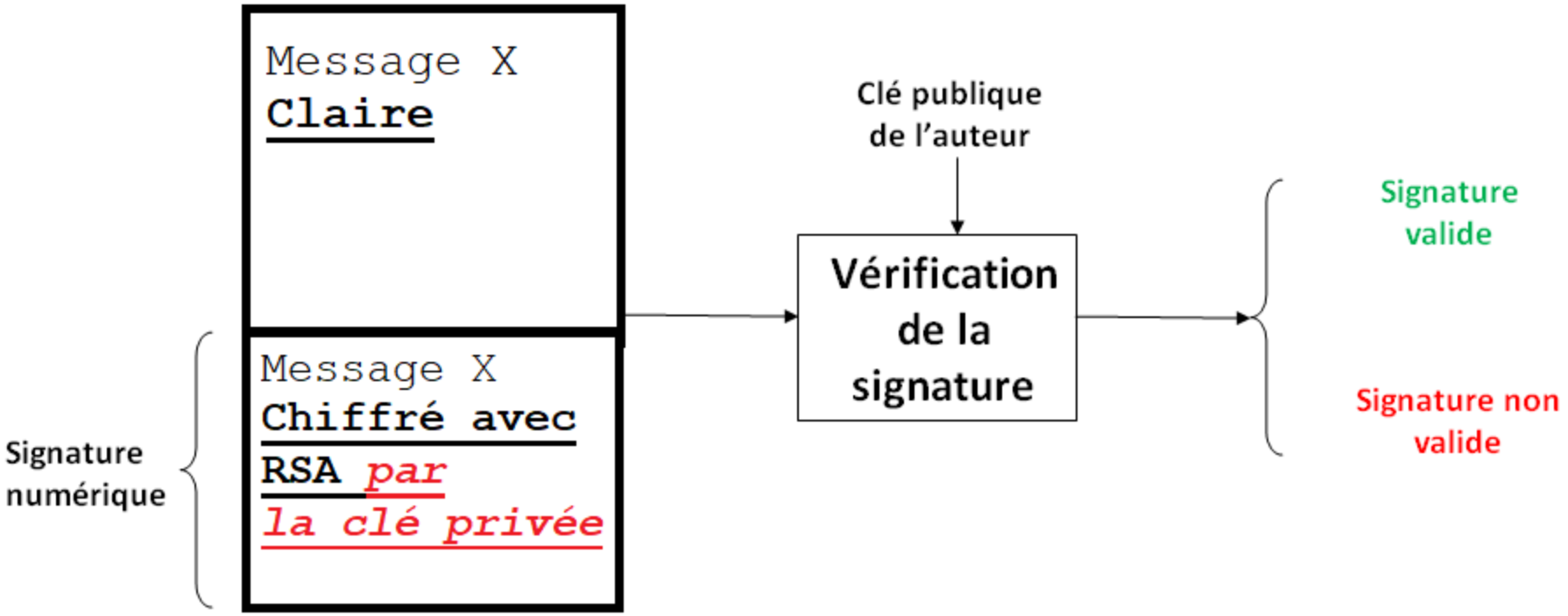
Itération	Etape	Valeur de l'exponent	Valeur	Opération
1	1a	10	9	Square
1	2a	11	27	Multiply
2	1a	110	729	Square
2	2a	-	729	Multiply
3	1a	1100	531441	Square
3	2a	1101	1594323	Multiply

Application de la cryptographie Asymetrique

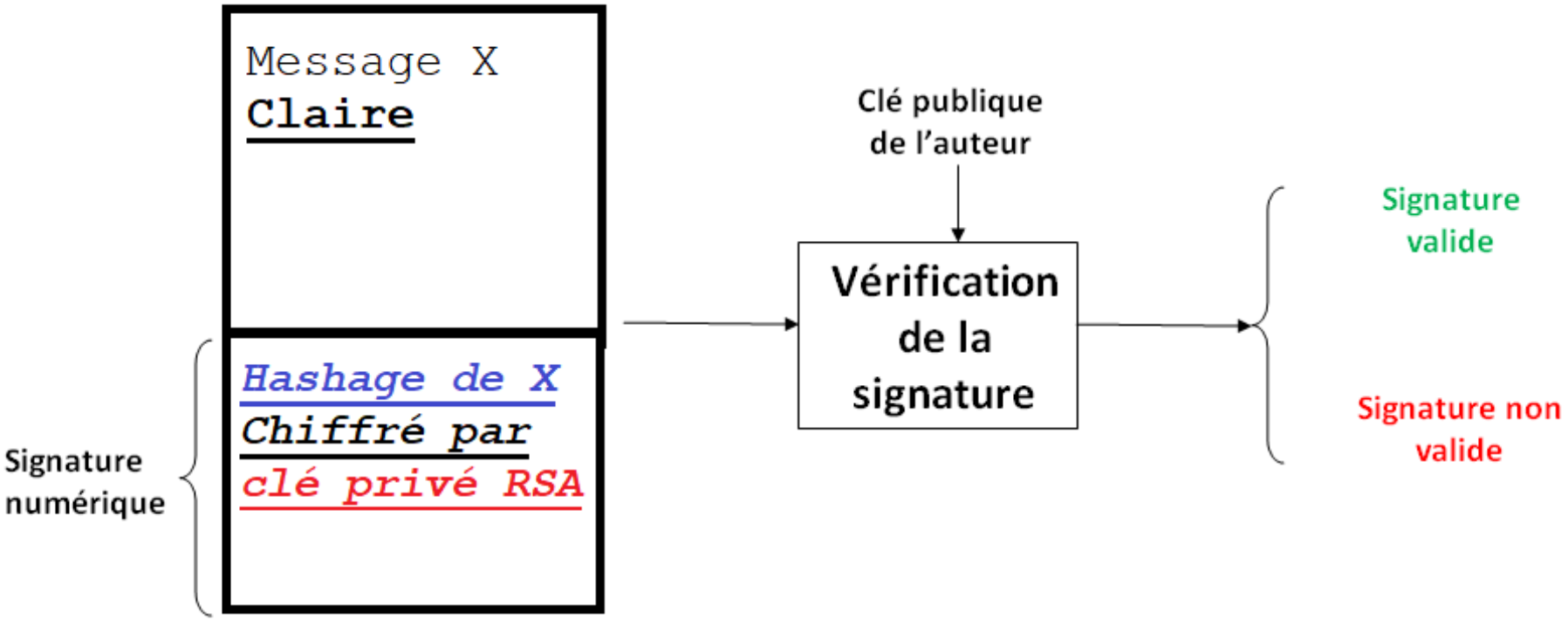
- Echange sécurisé de la clé symétrique avec RSA:
- Les deux parties commencent par l'échange des clés publiques RSA.
- La clé symétrique AES est générée aléatoirement et ensuite chiffrée par la clé publique RSA du vis-à-vis (Bob).
- Par la suite, la communication entre les deux parties est chiffrée avec AES.



Signature numérique



Signature numérique avec Hashage



HTTPS (HyperText Transfer Protocol Secure) - Certification

- HTTPS combine entre le chiffrement symétrique et asymétrique pour sécuriser les échanges entre le client et le serveur.
- Les requêtes HTTPS sont les mêmes requêtes/réponses HTTP (port TCP 443) mais sont chiffrées avec un algorithmes de chiffrement symétrique.
- Pour lancer un serveur HTTPS, l'organisation propriétaire de serveur doit demander un Certificat SSL (Secure Sockets Layer) d'une Autorité de Certification (AC). Le travail de ces autorités est de vérifier l'identité du serveur.
- Si la réponse de l'Autorité de Certification est favorable, celle-ci offre au serveur un certificat SSL signé avec la clé privé de l' Autorité de Certification. La signature de l'autorité est simplement les information d'identification du serveur chiffré la clé privée de l' Autorité de Certification.

HTTPS (HyperText Transfer Protocol Secure) – Etablissement de connexion sécurisée

- Pour initier une connexion HTTPS, le client envoie un message **Client-Hello** après le three-way-handshake TCP. Ce message contient les différents algorithmes de chiffrement supportés par le client et d'autres informations relative à cette procédure.
- Par la suite, le serveur répond avec **Server-Hello** qui contient l'algorithme choisi par le serveur **(selon les préférences communiqués dans Client-Hello)**
- **Echange de Certificat:** Dans cette étape, le serveur envoie au client ça certificat fournie par l'AC. Ce certificat comporte l'identité du serveur ainsi que ça clé publique. Ce certificat est signé par la AC (chiffré avec ça clé privée) afin d'assurer l' authenticité de celui-ci
- A la réception le client génère une clé symétrique aléatoire selon l'algorithme selectionné (de AES par exemple), chiffre cette clé avec la clé publique du serveur et envoie celle-ci au serveur (seul le serveur possède la clé privé pour déchiffrer).
- Par la suite le client et le serveur commencent à transmettre les requêtes /réponses HTTP chiffrées avec cette clé symétrique.

HTTPS – Authentification du serveur

- Le certificat SSL signé par l'AC est utilisée par le navigateur du client afin de vérifier si le serveur est bien ce qu'il prétend être.
- Pour ceci, chaque navigateur est pré-chargé avec les clés publiques des AC de confiance connues. A la réception du certificat SSL du serveur, le client déchiffre la signature numérique fournie par l'AC avec la clé publique de celle-ci (disponible pour le navigateur). Si le résultat du déchiffrement est identique aux informations d'identité du serveur, alors ce certificat est valide. Sinon, le certificat est forgé.