Sécurité des systèmes d’information

(initiation à la cryptographie)
Partie 3: cryptographie asymétrique

université d’Alger 1 -
Benyoucef Benkhedda

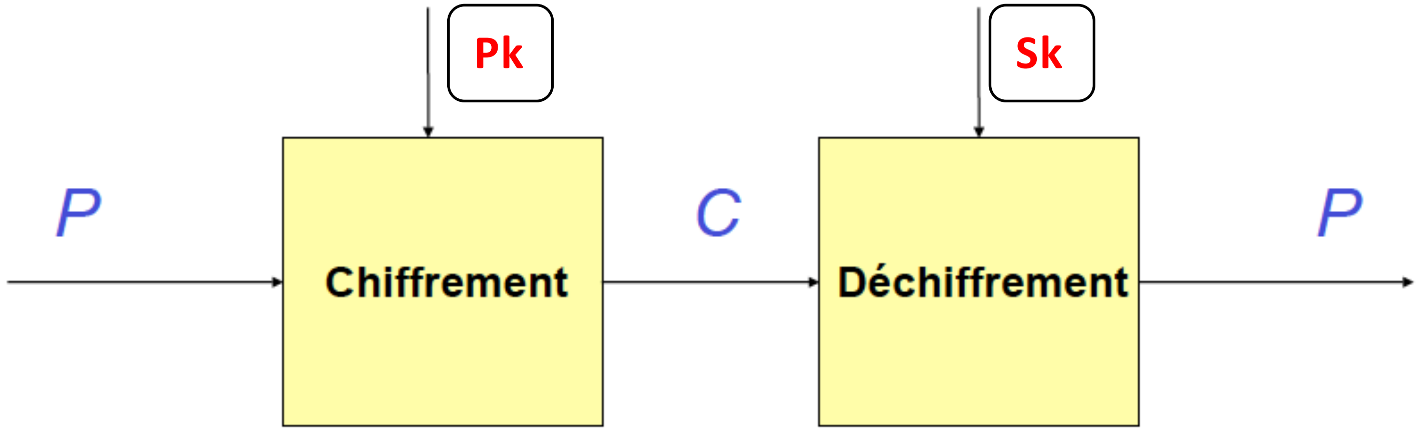
Rappel

- Chiffrement symétrique utilise la même clé pour chiffrer et déchiffrer.
  - Pratiquement sûr
  - Efficace en terme de temps de calcul
  - Problèmes d’utilisation dans une communication
  - N’assure que la confidentialité

Principe

- Chiffrer un message claire m en utilisant une **clé publique Pk**.

- Seule une clé secrète **Sk** déduite à partir de Pk peut être utilisée pour retrouver le message claire m



Principe

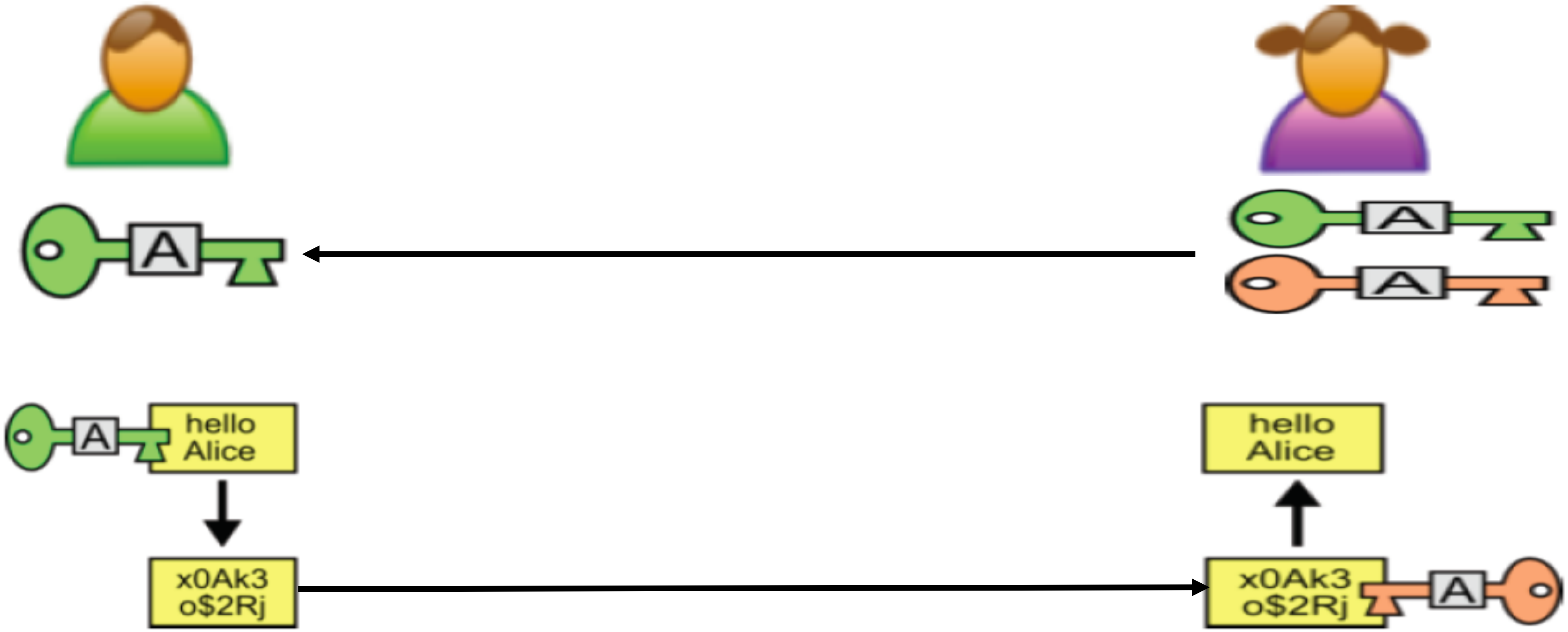
- Fondé sur le principe de **fonction à trappe**, ou bien **fonction a sens unique** ou **à brèche secrète**
  - Peu de gens qui connaissent une certaine information peuvent revenir en arrière

Fonction à sens unique

- Soit S et S' deux ensembles et $f: S \to S'$ une fonction. On dit f est à sens unique si elle satisfait:
  - Pour tout $x \in S$, on peut facilement calculer $y \in S'$ tel que $y=f(x)$.
  - Pour tout $y \in S'$ tel que $y=f(x)$, il est quasi-impossible de trouver x sans une connaissance secrète.

Scénario de communication

- Bob souhaite envoyer un message à Alice



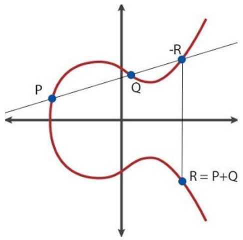
Quelques algorithmes asymétriques

Plusieurs algorithmes

- Chacun d'eux se repose sur un problème mathématique d'une fonction à trappe
- **RSA:** se repose sur le problème de factorisation des entiers en nombres premiers
- **Al-Gamal et Diffie-Hellman:** se reposent sur le problème de logarithme discret

D'autres algorithmes plus modernes

- Cryptographie à courbes elliptiques
- Cryptographie homomorphique
- ...etc.



Quelques algorithmes asymétriques

L'algorithme RSA

- Proposé par Rivest, Shamir et Adlmen en 1977 à MIT.

- Basé sur le problème de factorisation en nombres premiers pour sécuriser les données.

- Utilise des grands entiers (jusqu'à 1024 bits)

Quelques algorithmes asymétriques

L'algorithme RSA

1. Générer aléatoirement deux nombres premiers p et q
2. Mettre $n = p * q$ et $\varphi(n) = (p - 1) * (q - 1)$
3. Générer aléatoirement un entier e tel que $\mathrm{PGCD}(\mathrm{e}, \varphi(n)) = 1$
   - La clé publique **Pk** est donc **(e, n)**
4. Calculer un entier d (une trapdoor) en utilisant le **théorème Bézout** et l'algorithme d'**Euclide étendu** de tel que $\mathrm{e} * \mathrm{d} \equiv 1 \bmod(\varphi(n))$
   - La clé secrète **Sk** est donc **(d, n)**

- Pour chiffrer un message clair m, il suffit de calculer

$$ENC_{Pk}(m) = m^e \; mod \; n$$

- Pour déchiffrer un message chiffré CT, il suffit de calculer

$$DEC_{Sk}(CT) = CT^d \; mod \; n$$

Quelques algorithmes asymétriques

L'algorithme RSA

Théorème de Bézout:

Soit deux entiers x et y, on dit que x et y sont premiers entre eux (PGCD (x, y) = 1) si et seulement si:

$$\exists u, v \in \mathbb{Z} : x * u + y * v = 1$$

Cela peut aussi être écrit comme:

$$x * u \equiv 1 \bmod \mathrm{y}$$

$$\Rightarrow u^{-1} = x$$

On dit que u est l'**inversible** de x et il est calculé en utilisant l'algorithme d'Euclide étendu

Quelques algorithmes asymétriques

L’algorithme RSA

l’algorithme d’Euclide étendu:

Soit x et y deux entiers naturels premiers entre eux. Pour calculer l’inversible de x on doit résoudre l’équation x*u + y*v=1. l’algorithme d’Euclide étendu est comme suit:

```
fonction euclide_etendu(x, y) {

r = x, r′ = y, u = 1, v = 0, u′ = 0, v′ = 1

tant que (r′ != 0) faire
q = r ÷ r′ (division entiere)
       r = r′, u = u′, v = v′,
       r′ = r - q*r′, u′ = u - q*u′, v′ = v - q*v′
fin tant que

retourner (r, u, v) (u est l’inversible de x)
```

Quelques algorithmes asymétriques

L'algorithme RSA

Exemple :

Soit p = 13, q = 7 et e = 23. soit le message m = 123. on voulait calculer la clé publique Pk, la clé secrète Sk et le message chiffré CT

Étape 1: Calculer la clé publique Pk:

1. n = p * q => n = 13 * 7 = 91
2. $\varphi(n) = (p-1) * (q-1)$ => $\varphi(n) = 12 * 6 = 72$
3. La clé publique Pk = (e, n) = (23, 91)

Quelques algorithmes asymétriques

L'algorithme RSA

Exemple :

Soit p = 13, q = 7 et e = 23. soit le message m = 123. on voulait calculer la clé publique Pk, la clé secrète Sk et le message chiffré CT

Étape 2: Calculer la clé secrète Sk:

On doit trouvé un entier d tel que e $* d + v * \varphi(n) = 1$ => e $* d + v * 72 = 1$

1. On commence par l'initialisation des variables:

r = e = 23 et r' = $\varphi(n)$ = 72

u = d = 1 et u' = 0

v = 0 et v' = 1

Quelques algorithmes asymétriques

L'algorithme RSA

Exemple :

Soit p = 13, q = 7 et e = 23. soit le message m = 123. on voulait calculer la clé publique Pk, la clé secrète Sk et le message chiffré CT

Étape 2: Calculer la clé secrète Sk:

2. On commence par l'itération 1 puisque r' != 0 :

$$q = r \div r' = 23 \div 72 = 0$$

$$r = r' = 72 \text{ et } r' = r - q * r' = 23 - 0 * 72 = 23$$

$$u = u' = 0 \text{ et } u' = u - q * u' = 1 - 0 * 0 = 1$$

$$v = v' = 1 \text{ et } v' = v - q * v' = 0 - 0 * 1 = 0$$

La valeur de r' devient 23, elle est toujours différente de 0 donc on continue

Quelques algorithmes asymétriques

L'algorithme RSA

Exemple :

Soit p = 13, q = 7 et e = 23. soit le message m = 123. on voulait calculer la clé publique Pk, la clé secrète Sk et le message chiffré CT

Étape 2: Calculer la clé secrète Sk:

2. On passe à l'itération 2 puisque r' != 0 :

$q = r \div r' = 72 \div 23 = 3$

$r = r' = 23$ et $r' = r - q * r' = 72 - 3 * 23 = 3$

$u = u' = 1$ et $u' = u - q * u' = 0 - 3 * 1 = -3$

$v = v' = 0$ et $v' = v - q * v' = 1 - 3 * 0 = 1$

La valeur de r' devient 3, elle est toujours différente de 0 donc on continue

Quelques algorithmes asymétriques

L'algorithme RSA

Exemple :

Soit p = 13, q = 7 et e = 23. soit le message m = 123. on voulait calculer la clé publique Pk, la clé secrète Sk et le message chiffré CT

Étape 2: Calculer la clé secrète Sk:

2. On passe à l'itération 3 puisque r' != 0 :

q = r ÷ r' = 23 ÷ 3 = 7

r = r' = 3 et r' = r – q * r' = 23 – 7 * 3 = 2

u = u' = -3 et u' = u – q * u' = 1 – 7 * (-3) = 22

v = v' = 1 et v' = v – q * v' = 0 – 7 * 1 = -7

La valeur de r' devient 2, elle est toujours différente de 0 donc on continue

Quelques algorithmes asymétriques

L'algorithme RSA

Exemple :

Soit p = 13, q = 7 et e = 23. soit le message m = 123. on voulait calculer la clé publique Pk, la clé secrète Sk et le message chiffré CT

Étape 2: Calculer la clé secrète Sk:

2. On passe à l'itération 4 puisque r' != 0 :

$q = r \div r' = 3 \div 2 = 1$

$r = r' = 2$ et $r' = r - q * r' = 3 - 1 * 2 = 1$

$u = u' = 22$ et $u' = u - q * u' = -3 - 1 * 22 = -25$

$v = v' = -7$ et $v' = v - q * v' = 1 - 1 * (-7) = 8$

La valeur de r' devient 1, elle est toujours différente de 0 donc on continue

Quelques algorithmes asymétriques

L'algorithme RSA

Exemple :

Soit p = 13, q = 7 et e = 23. soit le message m = 123. on voulait calculer la clé publique Pk, la clé secrète Sk et le message chiffré CT

Étape 2: Calculer la clé secrète Sk:

2. On passe à l'itération 5 puisque r' != 0 :

q = r ÷ r' = 2 ÷ 1 = 2

r = r' = 1 et r' = r – q * r' = 2 – 2 * 1 = 0

u = u' = -25 et u' = u – q * u' = 22 – 2 * (-25) = 72

v = v' = 8 et v' = v – q * v' = -7 – 2 * 8 = -23

La valeur de r' devient 0, donc on s'arrete dans cette itération et on peut vérifier la validité de solution:

d = u = -25 et v = 8 donc $e * d + v * \varphi(n) = 1$ donne $23 * (-25) + 8 * 72 = 1$

alors la clé secrète Sk = (d, n) = (-25, 91) on la transfère à une valeur positive par un modulo, ça devient -25 mod 72 = 47 la clé secrète Sk=(47, 91)

Quelques algorithmes asymétriques

L'algorithme RSA

Exemple :

Soit p = 13, q = 7 et e = 23. soit le message m = 123. on voulait calculer la clé publique Pk, la clé secrète Sk et le message chiffré CT

Étape 3: chiffrer le message m:

L'équation de chiffrement de message m est:

$$ENC_{Pk}(m) = m^e \ mod \ n$$
$$ENC_{Pk}(m) = 123^{23} \ mod \ 91$$

Pour calculer ce chiffrement il y a aussi un algorithme simple appelé « **exponentiation rapide modulaire** »

Quelques algorithmes asymétriques

Étape 3: chiffrer le message m:

Pour calculer 123^{23} mod 91, on commence par un codage de 23 en binaire ça donne $(23)_{10}=(10111)_2$

On a 5 bits, donc on va calculer 5 valeurs:

1. Correspond au bit 0 (1er bit a gauche), $123^{2^0}\ mod\ 91 = 123\ mod\ 91 = 32$ => bit =1
2. Correspond au bit 1, $123^{2^1}\ mod\ 91 = 123^2\ mod\ 91 = 15129\ mod\ 91 = 23$ => bit =0
3. Correspond au bit 2, $123^{2^2}\ mod\ 91 = 23^2\ mod\ 91 = 529\ mod\ 91 = 74$ => bit =1
4. Correspond au bit 2, $123^{2^3}\ mod\ 91 = 74^2\ mod\ 91 = 5476\ mod\ 91 = 170$ => bit =1
5. Correspond au bit 2, $123^{2^4}\ mod\ 91 = 170^2\ mod\ 91 = 28900\ mod\ 91 = 53$ => bit =1

On déduit donc que 123^{23} mod 91 = $123^{2^0} * 123^{2^2} * 123^{2^3} * 123^{2^4}\ mod\ 91 = 32$ *74*170*53 mod 91 = 21335680 mod 91 = 2

Le texte chiffré CT = 2

Quelques algorithmes asymétriques

L'algorithme de Diffie-Hellman:

- Est un algorithme de partage de clé secrète entre deux entités
- Il se base sur le problème de **logarithme discret**

Logarithme discret:

- Soit G un groupe monogène fini d'ordre n et p un nombre premier. on appel g ∈ G, un générateur modulo p si pour tout b ∈ Z_p, in existe un entier a tel que b = g^a mod p
- Le problème de logarithme discret consiste à résoudre le problème suivant:
  - Étant donné G, g et b, trouver l'entier unique a=$\log_g$ b

Quelques algorithmes asymétriques

L'algorithme de Diffie-Hellman:

- Choisir un groupe G et son générateur g
- Choisir aléatoirement un entier a

- Choisir aléatoirement un entier b

(G, g, A=g^a mod p) $\longrightarrow$

- Calculer B = g^b mod p

$\longleftarrow$ (B)

- Calculer K = B^a mod p

- Calculer K = A^b mod p

B^a mod p = g^{b^a} mod p = g^{ba} mod p = g^{ab} mod p = A^b mod p

Quelques algorithmes asymétriques

L’algorithme d’Al-Gamal:

- Est un algorithme de chiffrement à clé publique

- Il se base aussi sur le problème de **logarithme discret**

Étape 1: génération des clés

1. Choisir un groupe G d’ordre n et son générateur g
2. Choisir aléatoirement un entier $0 < a < n-1$
3. Calculer $A=g^a$
4. La clé publique **Pk=(G, g, A, p)**
5. La clé secrète **Sk = a**

Quelques algorithmes asymétriques

L'algorithme d'Al-Gamal:

- Est un algorithme de chiffrement à clé publique

- Il se base aussi sur le problème de **logarithme discret**

Étape 2: chiffrement

Soit le message clair m.

1. Choisir aléatoirement un entier $0 < k < n-1$
2. Calculer $y_1 = g^k$ et $y_2 = m * A^k$
3. le message chiffré **CT = (y_1, y_2)**

Étape 3: déchiffrement

1. Calculer $m = \frac{y_2}{y_1^a}$

En effet, $\frac{y_2}{y_1^a} = \frac{m * A^k}{\boldsymbol{g}^{k\boldsymbol{a}}} = \frac{m * \boldsymbol{g}^{ak}}{\boldsymbol{g}^{k\boldsymbol{a}}} = m$

Chiffrement asymétrique en pratique

Openssl

- Open source
- Préinstallé dans toute les distributions de Linux
- Simple et pratique
- Contient aussi une bibliothèque en c « openssl.h »

Avantages et inconvénients

Avantages:

- Peut être facilement utilisé dans les communications à cause de possibilité de publication des clés publiques

Inconvénients:

- Lourd en terme de calcule (besoin de calculer plusieurs clés)
- Problèmes de non-répudiation des clés publiques