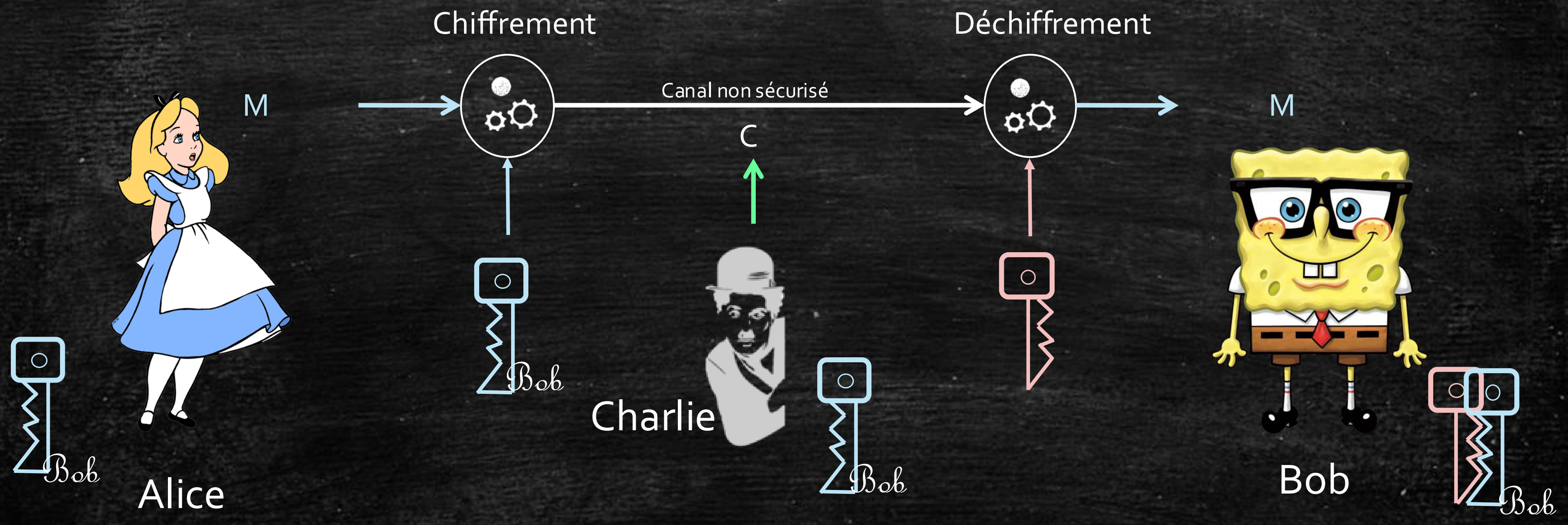
Introduction à la cryptographie

Louiza Khati

Cours 4

Chiffrement asymétrique



Alice envoie un message à Bob (utilisation du bi-clé de Bob)

Chiffrement asymétrique

- Propriétés sur les clés
  - La connaissance de la clé publique **ne doit pas permettre** de retrouver la clé privée
  - La clé privée et la clé publique sont **liées**

- Propriétés sur le schéma : fonction à sens unique
  - **Chiffrer** un message doit être facile
  - **Déchiffrer** un message **sans la clé** doit être **très difficile!**
- Repose sur un problème difficile (preuve par réduction)
  - La **factorisation** d'entiers
  - Le **logarithme** discret

Chiffrement asymétrique : Factorisation

- **Factorisation** des nombre entiers
  - $53 \times 37 = ?$
  - $1403 = ?$

Chiffrement asymétrique : Factorisation

- **Factorisation** des nombre entiers
  - 53 x 37 = 1961
  - 1403 = 61 * 23

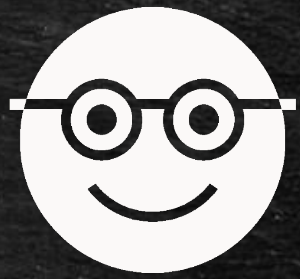
Chiffrement asymétrique : Factorisation

- **Factorisation** des nombre entiers
  - 53 x 37 = 1961
  - 1403 = 61 * 23

- Et
251959084756578934940271832400483985714292821262040320277771378
360436620207075955562640185258807844069182906412495150821892985591
491761845028084891200728449926873928072877767359714183472702618
963750149718246911650776133798590957000973304597488084284017974
291006424586918171951187461215151726546322822168699875491824224
336372590851418654620435767984233871847744479207399342365848238
242811981638150106748104516603773060562016196762561338441436038
339044149526344321901146575444541784240209246165157233507787077
498171257724679629263863563732899121548314381678998850404453640
23527381951378636564391212010397122822120720357 ????

Complexité : produit et factorisation

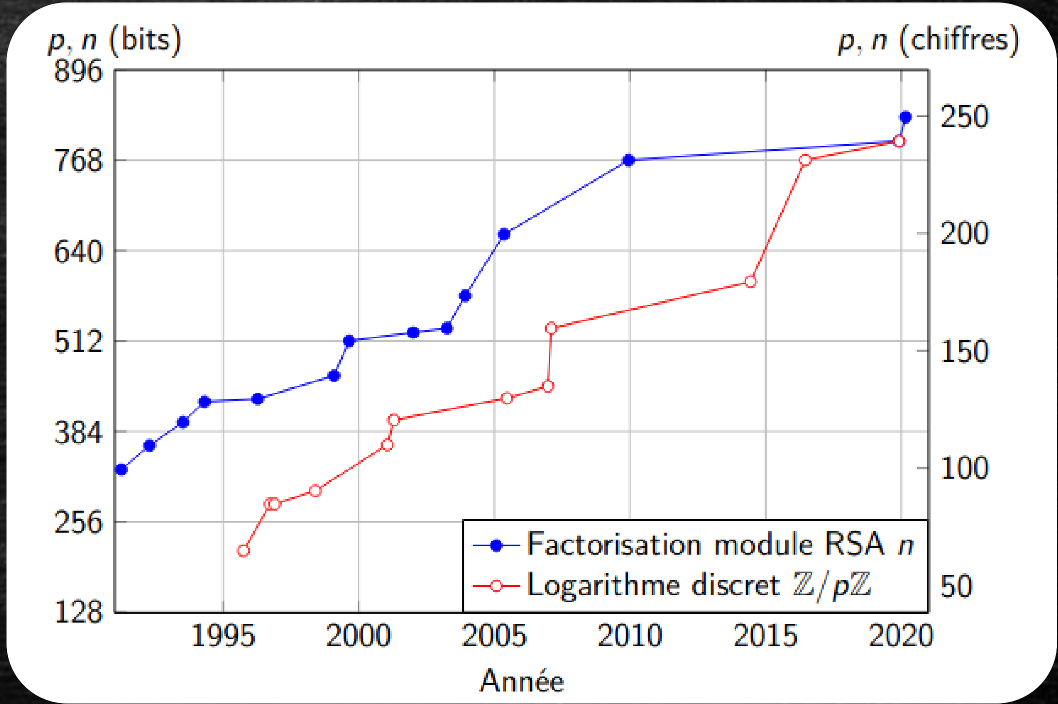
- Produit de deux nombres de n bits
- Coût d'un produit : $N_1 \times N_2$
  - Soient N_1 et N_2 deux nombres premiers sur n bits
  - Méthode naïve : coût $O(n^2)$
  - Méthodes plus efficaces : complexité **quasi-linéaire**



- Factorisation
  - Pour un entier sur N sur n bits, complexité **exponentielle**
  - Algorithme naïf : $O(\sqrt{N}) \rightarrow O(2^{n/2})$

Factorisation

- Record de factorisation
  - 768 bits et 232 chiffres décimaux



https://members.loria.fr/AGuillevic/files/teaching/NFS/techniques-de-l-ingenieur-record-calcul-RSA240.pdf

Rappels : Algorithme d'Euclide

- Soient a et b deux entiers : pgcd(a,b) = pgcd(b, r) où r = a mod(b)
- Exemple : pgcd (119, 91) = ?
  - 119 = 1* 91 + 28
  - 91 = 3*28 + $\boxed{7}$
  - 28 = 4* 7 + 0

PGCD!

Rappels : Algorithme d’Euclide

- Soient a et b deux entiers : pgcd(a,b) = pgcd(b, r) où r = a mod(b)
- Exemple : pgcd (119, 91) = 7
  - 119 = 1* 91 + 28
  - 91 = 3*28 + $\boxed{7}$
  - 28 = 4* 7 + 0

Rappels : Théorème de Bezout

- Soient a et b deux entiers naturels tel que pgcd(a,b) = d alors il existe deux entiers relatifs u et v tel que :

$$d = a*u+b*v$$

  u et v sont appelés les coefficients de Bezout.

- Exemple : a = 21 et b = 12 → d = 3
  - (-1)*21 + 2 * 12 = 3 → u = -1 et v = 2

- Cas particulier d = 1 : a et b premiers entre eux alors il existe u et v tel que

$$1 = a*u+b*v$$

(calcul inverse modulaire)

Rappels : Algorithme d'Euclide étendu

- Version récursive de l'algorithme d'Euclide
- Permet de trouver les coefficients de Bezout
- Exemple : pgcd (119, 91) = ?
  - (1) 119 = 1* 91 + 28
  - (2) 91 = 3*28 + 7
  - (3) 28 = 4* 7 + 0
- Reconstruction (trouver les coefficients de Bezout) :
  - (2)→ 91 – 3*28 = 7
  - Avec (1) →91 – 3 * (119 – 1*91) = 7
  - Finalement 4*91 – 3* 119 = 7

Rappels : Indicatrice d'Euler

- Fonction qui à tout entier naturel N non nul associe le nombre d'entiers compris entre 1 et N (inclus) et premiers avec N.

- Exemples :
  - $\phi(5) = 4$ ({1,2, 3, 4})
  - Si p est premier : $\phi(p) = p-1$
  - Si p_i premier et $n = \Pi\, p_i$ alors $\phi(n) = \Pi\,(p_i-1)$
- Remarque : Pour calculer $\phi(n)$, il faut connaitre la décomposition en facteurs premiers de n!