Fonctionnement du RSA

choix de d coeff. Bézout

$e \cdot d = 1 - \varphi(n) \cdot y$

$$\mu^d \bmod n = \underbrace{(m^e \bmod n)}_{\text{dif. de } \mu}{}^d \bmod n \underset{\substack{\uparrow \\ \text{dist.} \\ \text{Modulo}}}{\equiv_n} (m^{ed}) \bmod n \equiv_n \left(m^{1-\varphi(n)\cdot y}\right) \bmod n$$

$$e \cdot \underline{d} + \varphi(n) \cdot \underline{y} = 1$$

$$\underset{\substack{\uparrow \\ \text{dist.} \\ \text{Modulo}}}{\equiv_n} (m \bmod n) \cdot \underbrace{\left(m^{\varphi(n)} \bmod n\right)}_{\text{Fermat}}{}^{-y} \equiv_n m \bmod n = m$$

Fermat : $m^{\varphi(n)} \equiv_n 1$

m et n premiers entre eux

car $m \in [0, n-1]$

Pourquoi est-ce sûr d'utiliser le RSA !

Rappel : la clé PUBLIQUE est [n, e] et la clé privée est un coefficient de Bézout (l'inverse modulaire de $\varphi(n)$)

Si on connaît p, q et e, alors on peut calculer tout les reste (n, $\varphi(n)$ et donc d).

Choisit 2 premiers p, q

$n = p \cdot q$

Choisit e premier avec $\varphi(n) = (p-1)(q-1)$

Euclide étendu donne

$e \cdot x + \varphi(n) \cdot y = 1$

d privé ! ($d = x \bmod \varphi(n)$)

n étant une partie de la clé publique, il suffit de

$e \cdot \underbrace{x}_{} + \underbrace{\varphi(n)}_{} \cdot y = 1$

d privé ! ($d = x \bmod \varphi(n)$)

n étant une partie de la clé publique, il suffit de retrouver p et q (les deux nombres premiers) pour retrouver la clé privée d !!!

Craquer une clé RSA revient à FACTORISER n = p * q !

Le RSA repose UNIQUEMENT sur le fait que trouver p et q, sachant n est DIFFICILE !!!

Ce n'est difficile QUE par la taille de n !!!!

Aujoud'hui on parle du RSA-2048 (introduit petit à petit le 4096 et la NASA utilise parfois le 8192).
Cela signifie que n est sur 2048 bits !

Avec le RSA-2048 $n \approx 10^{640}$

Pour trouver p (ou q), il y en a forcément 1 entre 1 et $\sqrt{n}$.
Au pire, on doit tester $\sqrt{n} \approx 10^{300}$ candidats.

Avec 10^{30} opérations par seconde (largement plus que la puissance combinée de TOUS les ordinateurs de la terre).

Ordre de grandeur : on estime à env 10^{80} atomes dans l'univers.

$$\sqrt{n} \approx 10^{300} = \underbrace{10^{30}}_{\text{Tous les ordi. de la planète}} \cdot \underbrace{10^{270}}_{} s$$

$\Rightarrow$ il faudrait $10^{270} s \approx 10^{262}$ années !!

1 an fois $3 \cdot 10^7 s. \approx 10^8$

C'est IMPENSABLE d'y arriver avec notre technologie actuelle en moins de plusieurs milliards d'années !!!!

RSA-512 : il faut 8'000 ans de calcul a raison de 1 Mégaflop (10^6 calculs par seconde).

300 ordinateurs dédiés, la factorisation du RSA-512 a été effectuée en 3 mois de calcul !

$$\underbrace{x^2}_{\approx n^2} \bmod n$$

$$\approx n^2 < 2^{64} \Rightarrow n < 2^{32}$$

TP:

$= 43223 \cdot 27551$

$\mu =$

$\hookrightarrow m = 2123082 \left(\xrightarrow{\text{UTF-8}} \text{"Je\_"} \right)$

(1·11·0010 . .)

$$m = \mu^{\boxed{d}} \bmod n$$

1) Trouver $d = 58122075$

1.1.) factoriser $n = \underbrace{p \cdot q}_{\text{premiers pour sûr!}}$

1.2) $\varphi(n) = (p-1)(q-1)$

1.3) Euclide étendu pour calculer

$\text{PGCD}(\varphi(n), e) = 1 = e \cdot x + \varphi(n) \cdot y$

$d = x \bmod \varphi(n)$

2) déchiffrer $m = \mu^d \bmod n$ exp. rapide

3) Convertir int $m \xrightarrow{\text{ASCII !!!}}$ string