Le chiffrement RSA

Valentin Defransure

November 23, 2023

le plan

1. Fonctionnement de RSA (historique, rappels de maths et l'algorithme)
2. Enjeux et risques autour de RSA (résister aux attaques, et l'arrivée du quantique)

un peu de culture

- RSA est très utilisé pour l'envoi de données confidentielles
- RSA a été décrit pour la première fois en 1977 par Ron **R**ivest, Adi **S**hamir et Leonard **A**dleman du MIT
- RSA est le sytème de chiffrement le plus répandu au monde
- le nom de RSA vient des initiales des 3 auteurs

un peu de culture

- RSA est très utilisé pour l'envoi de données confidentielles
- RSA a été décrit pour la première fois en 1977 par Ron **R**ivest, Adi **S**hamir et Leonard **A**dleman du MIT
- RSA est le sytème de chiffrement le plus répandu au monde
- le nom de RSA vient des initiales des 3 auteurs

un peu de culture

- RSA est très utilisé pour l'envoi de données confidentielles
- RSA a été décrit pour la première fois en 1977 par Ron **R**ivest, Adi **S**hamir et Leonard **A**dleman du MIT
- RSA est le sytème de chiffrement le plus répandu au monde
- le nom de RSA vient des initiales des 3 auteurs

un peu de culture

- RSA est très utilisé pour l'envoi de données confidentielles
- RSA a été décrit pour la première fois en 1977 par Ron **R**ivest, Adi **S**hamir et Leonard **A**dleman du MIT
- RSA est le sytème de chiffrement le plus répandu au monde
- le nom de RSA vient des initiales des 3 auteurs

exemples d'utilisations

- cartes bancaires, et plus généralement par les banques
- les communications militaires

Mais pourquoi utiliser un système de chiffrement qui, vous le verrez, et non trivial ?

exemples d'utilisations

- cartes bancaires, et plus généralement par les banques
- les communications militaires

Mais pourquoi utiliser un système de chiffrement qui, vous le verrez, et non trivial ?

exemples d'utilisations

- cartes bancaires, et plus généralement par les banques
- les communications militaires

Mais pourquoi utiliser un système de chiffrement qui, vous le verrez, et non trivial ?

codage naif

Imaginons que l'on crypte un mot en remplaçant chaque lettre par sa lettre voisine . . .

alors le mot "CVSHFS" code le mot : "BURGER" Ce codage est faible niveau sécurité.

codage naif

Imaginons que l'on crypte un mot en remplaçant chaque lettre par sa lettre voisine . . .
alors le mot "CVSHFS" code le mot : "BURGER" Ce codage est faible niveau sécurité.

codage naif

Imaginons que l'on crypte un mot en remplaçant chaque lettre par sa lettre voisine . . .
alors le mot "CVSHFS" code le mot : "BURGER" Ce codage est faible niveau sécurité.

codage naif

Imaginons que l'on crypte un mot en remplaçant chaque lettre par sa lettre voisine . . .
alors le mot "CVSHFS" code le mot : "BURGER" Ce codage est faible niveau sécurité.

un exemple un peu moins naif

En fait, on veut une fonction f, bijective (pour pouvoir l'inverser) qui soit facile à inverser pour le destinataire du message, mais presque-impossible pour un attaquant extérieur.
En pratique, le destinataire du message doit posséder une information en plus, qui lui rend le calcul de f^{-1} facile.
On étend évidemment f par $f(a_0 \dots a_n) = f(a_0) \dots f(a_n)$

mais ça ne suffit pas . . .

La raison est que l'attaque fréquentielle est un succès dans ce cas.
Exemple : Admettons que l'on veuille déchiffrer le mot 1X'2X5X6 8X9 10X2X8 11X8X4 (le X désigne la séparationde 2 lettres)
On sait que la langue est le français, et on remarque le caractère 1. C'est soit un C, soit un L. On teste le C. Statistiquement, le premier mot a de fortement chances d'être C'est. Du coup, en remplacant on a : C'est 8X9 2XeX8 11X8X4. Après, on voit que la lettre 8 revient souvent. C'est probablement un a, un i ou un u, ou un o. Pb : elle commence un mots de 2 lettre. C'est donc un u.
Du coup on se retrouve avec C'est uX9 10Xeu 11XuX4, et le 9 est un n. A partir de là, on déchiffre sans peine.

mais ça ne suffit pas . . .

La raison est que l'attaque fréquentielle est un succès dans ce cas.
Exemple : Admettons que l'on veuille déchiffrer le mot 1X'2X5X6 8X9 10X2X8 11X8X4 (le X désigne la séparationde 2 lettres)
On sait que la langue est le français, et on remarque le caractère 1. C'est soit un C, soit un L. On teste le C. Statistiquement, le premier mot a de fortement chances d'être C'est. Du coup, en remplacant on a : C'est 8X9 2XeX8 11X8X4. Après, on voit que la lettre 8 revient souvent. C'est probablement un a, un i ou un u, ou un o. Pb : elle commence un mots de 2 lettre. C'est donc un u.
Du coup on se retrouve avec C'est uX9 10Xeu 11XuX4, et le 9 est un n. A partir de là, on déchiffre sans peine.

mais ça ne suffit pas . . .

La raison est que l'attaque fréquentielle est un succès dans ce cas.
Exemple : Admettons que l'on veuille déchiffrer le mot 1X'2X5X6 8X9 10X2X8 11X8X4 (le X désigne la séparationde 2 lettres)
On sait que la langue est le français, et on remarque le caractère 1. C'est soit un C, soit un L. On teste le C. Statistiquement, le premier mot a de fortement chances d'être C'est. Du coup, en remplacant on a : C'est 8X9 2XeX8 11X8X4. Après, on voit que la lettre 8 revient souvent. C'est probablement un a, un i ou un u, ou un o. Pb : elle commence un mots de 2 lettre. C'est donc un u.
Du coup on se retrouve avec C'est uX9 10Xeu 11XuX4, et le 9 est un n. A partir de là, on déchiffre sans peine.

mais ça ne suffit pas . . .

La raison est que l'attaque fréquentielle est un succès dans ce cas.
Exemple : Admettons que l'on veuille déchiffrer le mot 1X'2X5X6 8X9 10X2X8 11X8X4 (le X désigne la séparationde 2 lettres)
On sait que la langue est le français, et on remarque le caractère 1. C'est soit un C, soit un L. On teste le C. Statistiquement, le premier mot a de fortement chances d'être C'est. Du coup, en remplacant on a : C'est 8X9 2XeX8 11X8X4. Après, on voit que la lettre 8 revient souvent. C'est probablement un a, un i ou un u, ou un o. Pb : elle commence un mots de 2 lettre. C'est donc un u.
Du coup on se retrouve avec C'est uX9 10Xeu 11XuX4, et le 9 est un n. A partir de là, on déchiffre sans peine.

mais ça ne suffit pas . . .

La raison est que l'attaque fréquentielle est un succès dans ce cas.
Exemple : Admettons que l'on veuille déchiffrer le mot 1X'2X5X6 8X9 10X2X8 11X8X4 (le X désigne la séparationde 2 lettres)
On sait que la langue est le français, et on remarque le caractère 1. C'est soit un C, soit un L. On teste le C. Statistiquement, le premier mot a de fortement chances d'être C'est. Du coup, en remplacant on a : C'est 8X9 2XeX8 11X8X4. Après, on voit que la lettre 8 revient souvent. C'est probablement un a, un i ou un u, ou un o. Pb : elle commence un mots de 2 lettre. C'est donc un u.
Du coup on se retrouve avec C'est uX9 10Xeu 11XuX4, et le 9 est un n. A partir de là, on déchiffre sans peine.

mais ça ne suffit pas . . .

La raison est que l'attaque fréquentielle est un succès dans ce cas.
Exemple : Admettons que l'on veuille déchiffrer le mot 1X'2X5X6 8X9 10X2X8 11X8X4 (le X désigne la séparationde 2 lettres)
On sait que la langue est le français, et on remarque le caractère 1. C'est soit un C, soit un L. On teste le C. Statistiquement, le premier mot a de fortement chances d'être C'est. Du coup, en remplacant on a : C'est 8X9 2XeX8 11X8X4. Après, on voit que la lettre 8 revient souvent. C'est probablement un a, un i ou un u, ou un o. Pb : elle commence un mots de 2 lettre. C'est donc un u.
Du coup on se retrouve avec C'est uX9 10Xeu 11XuX4, et le 9 est un n. A partir de là, on déchiffre sans peine.

mais ça ne suffit pas . . .

La raison est que l'attaque fréquentielle est un succès dans ce cas.
Exemple : Admettons que l'on veuille déchiffrer le mot 1X'2X5X6 8X9 10X2X8 11X8X4 (le X désigne la séparationde 2 lettres)
On sait que la langue est le français, et on remarque le caractère 1. C'est soit un C, soit un L. On teste le C. Statistiquement, le premier mot a de fortement chances d'être C'est. Du coup, en remplacant on a : C'est 8X9 2XeX8 11X8X4. Après, on voit que la lettre 8 revient souvent. C'est probablement un a, un i ou un u, ou un o. Pb : elle commence un mots de 2 lettre. C'est donc un u.
Du coup on se retrouve avec C'est uX9 10Xeu 11XuX4, et le 9 est un n. A partir de là, on déchiffre sans peine.

mais ça ne suffit pas . . .

La raison est que l'attaque fréquentielle est un succès dans ce cas.
Exemple : Admettons que l'on veuille déchiffrer le mot 1X'2X5X6 8X9 10X2X8 11X8X4 (le X désigne la séparationde 2 lettres)
On sait que la langue est le français, et on remarque le caractère 1. C'est soit un C, soit un L. On teste le C. Statistiquement, le premier mot a de fortement chances d'être C'est. Du coup, en remplacant on a : C'est 8X9 2XeX8 11X8X4. Après, on voit que la lettre 8 revient souvent. C'est probablement un a, un i ou un u, ou un o. Pb : elle commence un mots de 2 lettre. C'est donc un u.
Du coup on se retrouve avec C'est uX9 10Xeu 11XuX4, et le 9 est un n. A partir de là, on déchiffre sans peine.

RSA est un chiffrement difficle à casser, et qui résiste à des attaques fréquentielles et autres

bases mathématiques

- p est un nombre premier
- p et q sont premiers entre eux
- $a \equiv b\ [n]$
- lemme de Gauss
- théorème de Bézout
- petit théorème de Fermat
- théorème des restes chinois

bases mathématiques

- p est un nombre premier
- p et q sont premiers entre eux
- $a \equiv b\ [n]$
- lemme de Gauss
- théorème de Bézout
- petit théorème de Fermat
- théorème des restes chinois

bases mathématiques

- p est un nombre premier
- p et q sont premiers entre eux
- $a \equiv b\ [n]$
- lemme de Gauss
- théorème de Bézout
- petit théorème de Fermat
- théorème des restes chinois

bases mathématiques

- p est un nombre premier
- p et q sont premiers entre eux
- $a \equiv b\ [n]$
- lemme de Gauss
- théorème de Bézout
- petit théorème de Fermat
- théorème des restes chinois

bases mathématiques

- p est un nombre premier
- p et q sont premiers entre eux
- $a \equiv b\ [n]$
- lemme de Gauss
- théorème de Bézout
- petit théorème de Fermat
- théorème des restes chinois

bases mathématiques

- p est un nombre premier
- p et q sont premiers entre eux
- $a \equiv b\ [n]$
- lemme de Gauss
- théorème de Bézout
- petit théorème de Fermat
- théorème des restes chinois

bases mathématiques

- p est un nombre premier
- p et q sont premiers entre eux
- $a \equiv b\ [n]$
- lemme de Gauss
- théorème de Bézout
- petit théorème de Fermat
- théorème des restes chinois

théorèmes de Fermat et des restes chinois

(petit) théorème de Fermat

Soit p un nombre premier, alors $\forall a \in \mathbb{N}, a^p \equiv a\ [p]$

corollaire

Soit p un nombre premier, alors $\forall a \in \mathbb{N}, a^{p-1} \equiv 1\ [p]$ si p ne divise pas a, et 0 sinon

théorèmes des restes chinois

Soient $k \in \mathbb{N}$, $n_1, \ldots, n_k$ des nombres premiers entre eux 2 à 2 et $y_1, \ldots, y_k$ des entiers. Alors il existe un unique $x \leq n_1 \times \ldots \times n_k$ tel que, $\forall\ 1 \leq i \leq k,\ x \equiv y_i\ [n_i]$

théorèmes de Fermat et des restes chinois

(petit) théorème de Fermat

Soit p un nombre premier, alors $\forall a \in \mathbb{N}, a^p \equiv a\ [p]$

corollaire

Soit p un nombre premier, alors $\forall a \in \mathbb{N}, a^{p-1} \equiv 1\ [p]$ si p ne divise pas a, et 0 sinon

théorèmes des restes chinois

Soient $k \in \mathbb{N}$, $n_1, \ldots, n_k$ des nombres premiers entre eux 2 à 2 et $y_1, \ldots, y_k$ des entiers. Alors il existe un unique $x \leq n_1 \times \ldots \times n_k$ tel que, $\forall\ 1 \leq i \leq k,\ x \equiv y_i\ [n_i]$

théorèmes de Fermat et des restes chinois

(petit) théorème de Fermat

Soit p un nombre premier, alors $\forall a \in \mathbb{N}, a^p \equiv a \ [p]$

corollaire

Soit p un nombre premier, alors $\forall a \in \mathbb{N}, a^{p-1} \equiv 1 \ [p]$ si p ne divise pas a, et 0 sinon

théorèmes des restes chinois

Soient $k \in \mathbb{N}$, $n_1, \ldots, n_k$ des nombres premiers entre eux 2 à 2 et $y_1, \ldots, y_k$ des entiers. Alors il existe un unique $x \leq n_1 \times \ldots \times n_k$ tel que, $\forall\ 1 \leq i \leq k$, $x \equiv y_i \ [n_i]$

comment RSA fonctionne

- 2 personnes, A et B. A envoie veut envoyer un message à B. Dans un premier temps, ce message est uniquement une suite de chiffres
- A choisit 2 nombres premiers p et q et on calcule $p \cdot q$. On appelle ce nombre n. On pose $\phi(n) = (p-1) \times (q-1)$
- A choisit un nombre e, premier avec $\phi(n)$. Ce nombre e est appelé l'exposant de chiffrement, tel que $e \leq \phi(n)$
- A **crypte** le message en remplacant chaque chiffre c par $c^e[n]$. on note $z = c^e[n]$
- Soit d l'inverse modulaire modulaire de e (modulo $\phi(n)$), ie l'unique nombre $\leq \phi(n)$ tel que $e \times d \equiv 1\ [\phi(n)]$
- B **décrypte** le message crypté en remplaçant chaque chiffre c par $c^d\ [n]$. En effet, on a $c^{e \times d} \equiv c\ [n]$

comment RSA fonctionne

- 2 personnes, A et B. A envoie veut envoyer un message à B. Dans un premier temps, ce message est uniquement une suite de chiffres
- A choisit 2 nombres premiers p et q et on calcule $p \cdot q$. On appelle ce nombre n. On pose $\phi(n) = (p-1) \times (q-1)$
- A choisit un nombre e, premier avec $\phi(n)$. Ce nombre e est appelé l'exposant de chiffrement, tel que $e \leq \phi(n)$
- A **crypte** le message en remplacant chaque chiffre c par $c^e[n]$. on note $z = c^e[n]$
- Soit d l'inverse modulaire modulaire de e (modulo $\phi(n)$), ie l'unique nombre $\leq \phi(n)$ tel que $e \times d \equiv 1\ [\phi(n)]$
- B **décrypte** le message crypté en remplaçant chaque chiffre c par $c^d\ [n]$. En effet, on a $c^{e \times d} \equiv c\ [n]$

comment RSA fonctionne

- 2 personnes, A et B. A envoie veut envoyer un message à B. Dans un premier temps, ce message est uniquement une suite de chiffres
- A choisit 2 nombres premiers p et q et on calcule $p \cdot q$. On appelle ce nombre n. On pose $\phi(n) = (p-1) \times (q-1)$
- A choisit un nombre e, premier avec $\phi(n)$. Ce nombre e est appelé l'exposant de chiffrement, tel que $e \leq \phi(n)$
- A **crypte** le message en remplacant chaque chiffre c par $c^e[n]$. on note $z = c^e[n]$
- Soit d l'inverse modulaire modulaire de e (modulo $\phi(n)$), ie l'unique nombre $\leq \phi(n)$ tel que $e \times d \equiv 1\ [\phi(n)]$
- B **décrypte** le message crypté en remplaçant chaque chiffre c par $c^d\ [n]$. En effet, on a $c^{e \times d} \equiv c\ [n]$

comment RSA fonctionne

- 2 personnes, A et B. A envoie veut envoyer un message à B. Dans un premier temps, ce message est uniquement une suite de chiffres
- A choisit 2 nombres premiers p et q et on calcule $p \cdot q$. On appelle ce nombre n. On pose $\phi(n) = (p-1) \times (q-1)$
- A choisit un nombre e, premier avec $\phi(n)$. Ce nombre e est appelé l'exposant de chiffrement, tel que $e \leq \phi(n)$
- A **crypte** le message en remplacant chaque chiffre c par $c^e[n]$. on note $z = c^e[n]$
- Soit d l'inverse modulaire modulaire de e (modulo $\phi(n)$), ie l'unique nombre $\leq \phi(n)$ tel que $e \times d \equiv 1\ [\phi(n)]$
- B **décrypte** le message crypté en remplaçant chaque chiffre c par $c^d\ [n]$. En effet, on a $c^{e \times d} \equiv c\ [n]$

comment RSA fonctionne

- 2 personnes, A et B. A envoie veut envoyer un message à B. Dans un premier temps, ce message est uniquement une suite de chiffres
- A choisit 2 nombres premiers p et q et on calcule $p \cdot q$. On appelle ce nombre n. On pose $\phi(n) = (p-1) \times (q-1)$
- A choisit un nombre e, premier avec $\phi(n)$. Ce nombre e est appelé l'exposant de chiffrement, tel que $e \leq \phi(n)$
- A **crypte** le message en remplacant chaque chiffre c par $c^e[n]$. on note $z = c^e[n]$
- Soit d l'inverse modulaire modulaire de e (modulo $\phi(n)$), ie l'unique nombre $\leq \phi(n)$ tel que $e \times d \equiv 1\ [\phi(n)]$
- B **décrypte** le message crypté en remplaçant chaque chiffre c par $c^d\ [n]$. En effet, on a $c^{e \times d} \equiv c\ [n]$

comment RSA fonctionne

- 2 personnes, A et B. A envoie veut envoyer un message à B. Dans un premier temps, ce message est uniquement une suite de chiffres
- A choisit 2 nombres premiers p et q et on calcule $p \cdot q$. On appelle ce nombre n. On pose $\phi(n) = (p-1) \times (q-1)$
- A choisit un nombre e, premier avec $\phi(n)$. Ce nombre e est appelé l'exposant de chiffrement, tel que $e \leq \phi(n)$
- A **crypte** le message en remplacant chaque chiffre c par $c^e[n]$. on note $z = c^e[n]$
- Soit d l'inverse modulaire modulaire de e (modulo $\phi(n)$), ie l'unique nombre $\leq \phi(n)$ tel que $e \times d \equiv 1\ [\phi(n)]$
- B **décrypte** le message crypté en remplaçant chaque chiffre c par $c^d\ [n]$. En effet, on a $c^{e \times d} \equiv c\ [n]$

avant de montrer pourquoi ça fonctionne, un exemple

- p, q des nombres premiers, $n = p \cdot q$
- c est crypté par c^e $[n]$, on note ce nombre z
- z est décrypté avec $z^d = c$ $[n]$

Prenons $p = 3$ et $q = 11$. On a alors $n = 33$ et $\phi(n) = 20$. Admettons que A choisisse $e = 3$, alors $d = 7$, et que l'on veuille crypter le chiffre 9.
$3 \cdot 7 = 21$, et $21 = 1$ $[20]$ $9^3 = 9 \times 9 \times 9 = 81 \times 9$, et $81 = 2 \times 33 + 15$. et $15 \times 9 = 135 = 4 \times 33 + 3$ et
$3^7 = 9 \times 9 \times 9 \times 3 = 81 \times 27 \equiv 15 \times -6$ $[33] \equiv -90$ $[33] \equiv 9$.

avant de montrer pourquoi ça fonctionne, un exemple

- p, q des nombres premiers, $n = p \cdot q$
- c est crypté par c^e $[n]$, on note ce nombre z
- z est décrypté avec $z^d = c$ $[n]$

Prenons $p = 3$ et $q = 11$. On a alors $n = 33$ et $\phi(n) = 20$. Admettons que A choisisse $e = 3$, alors $d = 7$, et que l'on veuille crypter le chiffre 9.
$3 \cdot 7 = 21$, et $21 = 1$ [20] $9^3 = 9 \times 9 \times 9 = 81 \times 9$, et $81 = 2 \times 33 + 15$. et $15 \times 9 = 135 = 4 \times 33 + 3$ et
$3^7 = 9 \times 9 \times 9 \times 3 = 81 \times 27 \equiv 15 \times -6$ $[33] \equiv -90$ $[33] \equiv 9$.

avant de montrer pourquoi ça fonctionne, un exemple

- p, q des nombres premiers, $n = p \cdot q$
- c est crypté par c^e $[n]$, on note ce nombre z
- z est décrypté avec $z^d = c$ $[n]$

Prenons $p = 3$ et $q = 11$. On a alors $n = 33$ et $\phi(n) = 20$. Admettons que A choisisse $e = 3$, alors $d = 7$, et que l'on veuille crypter le chiffre 9.
$3 \cdot 7 = 21$, et $21 = 1$ $[20]$ $9^3 = 9 \times 9 \times 9 = 81 \times 9$, et $81 = 2 \times 33 + 15$. et
$15 \times 9 = 135 = 4 \times 33 + 3$ et
$3^7 = 9 \times 9 \times 9 \times 3 = 81 \times 27 \equiv 15 \times -6$ $[33] \equiv -90$ $[33] \equiv 9$.

avant de montrer pourquoi ça fonctionne, un exemple

- p, q des nombres premiers, $n = p \cdot q$
- c est crypté par c^e $[n]$, on note ce nombre z
- z est décrypté avec $z^d = c$ $[n]$

Prenons $p = 3$ et $q = 11$. On a alors $n = 33$ et $\phi(n) = 20$. Admettons que A choisisse $e = 3$, alors $d = 7$, et que l'on veuille crypter le chiffre 9.
$3 \cdot 7 = 21$, et $21 = 1$ $[20]$ $9^3 = 9 \times 9 \times 9 = 81 \times 9$, et $81 = 2 \times 33 + 15$. et $15 \times 9 = 135 = 4 \times 33 + 3$ et
$3^7 = 9 \times 9 \times 9 \times 3 = 81 \times 27 \equiv 15 \times -6$ $[33] \equiv -90$ $[33] \equiv 9$.

avant de montrer pourquoi ça fonctionne, un exemple

- p, q des nombres premiers, $n = p \cdot q$
- c est crypté par c^e $[n]$, on note ce nombre z
- z est décrypté avec $z^d = c$ $[n]$

Prenons $p = 3$ et $q = 11$. On a alors $n = 33$ et $\phi(n) = 20$. Admettons que A choisisse $e = 3$, alors $d = 7$, et que l'on veuille crypter le chiffre 9.
$3 \cdot 7 = 21$, et $21 = 1$ $[20]$ $9^3 = 9 \times 9 \times 9 = 81 \times 9$, et $81 = 2 \times 33 + 15$. et
$15 \times 9 = 135 = 4 \times 33 + 3$ et
$3^7 = 9 \times 9 \times 9 \times 3 = 81 \times 27 \equiv 15 \times -6$ $[33] \equiv -90$ $[33] \equiv 9$.

quelques premières justifications

- Déjà, comme annoncé, d est unique, à e fixé.
  Preuve : Soient d_1 et d_2 deux inverses modulaires de e, alors $e \times d_1 \times d_2 \equiv d_2\ [\phi(n)] \equiv d_1\ [\phi(n)]$, donc $d_1 - d_2 \equiv 0\ [\phi(n)]$, donc $\phi(n)$ divise $d_1 - d_2$. Or $|d_1 - d_2| < \phi(n)$, donc $d_1 = d_2$
- e **doit** être premier avec $\phi(n)$, sinon e n'est pas inversible modulo $\phi(n)$
  Preuve : e inversible modulo $\phi(n)$ d'inverse équivaut à $\exists d, l \in N$, $e \times d + l \times \phi(n)$, ie e et premier avec $\phi(n)$, cf théorème de Bézout.
- On a bien $c^{e \times d} \equiv c\ [n]$. En effet, il existe l tel que $e \times d = 1 + l \times (p-1) \times (q-1)$. Donc, avec le petit théorème de Fermat, $c^{e \times d} \equiv c\ [p]$, ie p divise $c^{e \times d} - c$. De même, q divise ce nombre. Par le lemme de Gauss, n divise ce nombre, ce qui montre que le chiffrement RSA est correct.

quelques premières justifications

- Déjà, comme annoncé, d est unique, à e fixé.
  Preuve : Soient d_1 et d_2 deux inverses modulaires de e, alors $e \times d_1 \times d_2 \equiv d_2\ [\phi(n)] \equiv d_1\ [\phi(n)]$, donc $d_1 - d_2 \equiv 0\ [\phi(n)]$, donc $\phi(n)$ divise $d_1 - d_2$. Or $|d_1 - d_2| < \phi(n)$, donc $d_1 = d_2$
- e **doit** être premier avec $\phi(n)$, sinon e n'est pas inversible modulo $\phi(n)$
  Preuve : e inversible modulo $\phi(n)$ d'inverse équivaut à $\exists d, l \in N$, $e \times d + l \times \phi(n)$, ie e et premier avec $\phi(n)$, cf théorème de Bézout.
- On a bien $c^{e \times d} \equiv c\ [n]$. En effet, il existe l tel que $e \times d = 1 + l \times (p-1) \times (q-1)$. Donc, avec le petit théorème de Fermat, $c^{e \times d} \equiv c\ [p]$, ie p divise $c^{e \times d} - c$. De même, q divise ce nombre. Par le lemme de Gauss, n divise ce nombre, ce qui montre que le chiffrement RSA est correct.

quelques premières justifications

- Déjà, comme annoncé, d est unique, à e fixé.
  Preuve : Soient d_1 et d_2 deux inverses modulaires de e, alors $e \times d_1 \times d_2 \equiv d_2\ [\phi(n)] \equiv d_1\ [\phi(n)]$, donc $d_1 - d_2 \equiv 0\ [\phi(n)]$, donc $\phi(n)$ divise $d_1 - d_2$. Or $|d_1 - d_2| < \phi(n)$, donc $d_1 = d_2$
- e **doit** être premier avec $\phi(n)$, sinon e n'est pas inversible modulo $\phi(n)$
  Preuve : e inversible modulo $\phi(n)$ d'inverse équivaut à $\exists d, l \in N$, $e \times d + l \times \phi(n)$, ie e et premier avec $\phi(n)$, cf théorème de Bézout.
- On a bien $c^{e \times d} \equiv c\ [n]$. En effet, il existe l tel que $e \times d = 1 + l \times (p-1) \times (q-1)$. Donc, avec le petit théorème de Fermat, $c^{e \times d} \equiv c\ [p]$, ie p divise $c^{e \times d} - c$. De même, q divise ce nombre. Par le lemme de Gauss, n divise ce nombre, ce qui montre que le chiffrement RSA est correct.

quelques premières justifications

- Déjà, comme annoncé, d est unique, à e fixé.
  Preuve : Soient d_1 et d_2 deux inverses modulaires de e, alors $e \times d_1 \times d_2 \equiv d_2\ [\phi(n)] \equiv d_1\ [\phi(n)]$, donc $d_1 - d_2 \equiv 0\ [\phi(n)]$, donc $\phi(n)$ divise $d_1 - d_2$. Or $|d_1 - d_2| < \phi(n)$, donc $d_1 = d_2$
- e **doit** être premier avec $\phi(n)$, sinon e n'est pas inversible modulo $\phi(n)$
  Preuve : e inversible modulo $\phi(n)$ d'inverse équivaut à $\exists d, l \in N$, $e \times d + l \times \phi(n)$, ie e et premier avec $\phi(n)$, cf théorème de Bézout.
- On a bien $c^{e \times d} \equiv c\ [n]$. En effet, il existe l tel que $e \times d = 1 + l \times (p-1) \times (q-1)$. Donc, avec le petit théorème de Fermat, $c^{e \times d} \equiv c\ [p]$, ie p divise $c^{e \times d} - c$. De même, q divise ce nombre. Par le lemme de Gauss, n divise ce nombre, ce qui montre que le chiffrement RSA est correct.

quelques premières justifications

- Déjà, comme annoncé, d est unique, à e fixé.
  Preuve : Soient d_1 et d_2 deux inverses modulaires de e, alors $e \times d_1 \times d_2 \equiv d_2\ [\phi(n)] \equiv d_1\ [\phi(n)]$, donc $d_1 - d_2 \equiv 0\ [\phi(n)]$, donc $\phi(n)$ divise $d_1 - d_2$. Or $|d_1 - d_2| < \phi(n)$, donc $d_1 = d_2$
- e **doit** être premier avec $\phi(n)$, sinon e n'est pas inversible modulo $\phi(n)$
  Preuve : e inversible modulo $\phi(n)$ d'inverse équivaut à $\exists d, l \in N$, $e \times d + l \times \phi(n)$, ie e et premier avec $\phi(n)$, cf théorème de Bézout.
- On a bien $c^{e \times d} \equiv c\ [n]$. En effet, il existe l tel que $e \times d = 1 + l \times (p-1) \times (q-1)$. Donc, avec le petit théorème de Fermat, $c^{e \times d} \equiv c\ [p]$, ie p divise $c^{e \times d} - c$. De même, q divise ce nombre. Par le lemme de Gauss, n divise ce nombre, ce qui montre que le chiffrement RSA est correct.

en pratique

- En pratique, A partage la clé dite publique (n, e), et seul A possède sa clé privée (n, d). B utilise la clé publique de A.
- Tous les systèmes de chiffrement ne repose pas sur ce système de clés privés/publiques : c'est le cas de la crypto symétrique

en pratique

- En pratique, A partage la clé dite publique (n, e), et seul A possède sa clé privée (n, d). B utilise la clé publique de A.
- Tous les systèmes de chiffrement ne repose pas sur ce système de clés privés/publiques : c'est le cas de la crypto symétrique

retour à la situation initiale

- La fonction de cryptage est une fonction facile à inverser pour le destinatiare, qui connaît d, mais très difficile voir impossible à inverser pour un attaquant extérieur (il faudrait trouver les facteurs premiers de n)
- sauf si celui du crypte fait des bêtises, mais j'en parlerai plus loin
- quid de l'attaque fréqentielle ?

retour à la situation initiale

- La fonction de cryptage est une fonction facile à inverser pour le destinatiare, qui connaît d, mais très difficile voir impossible à inverser pour un attaquant extérieur (il faudrait trouver les facteurs premiers de n)
- sauf si celui du crypte fait des bêtises, mais j'en parlerai plus loin
- quid de l'attaque fréqentielle ?

retour à la situation initiale

- La fonction de cryptage est une fonction facile à inverser pour le destinatiare, qui connaît d, mais très difficile voir impossible à inverser pour un attaquant extérieur (il faudrait trouver les facteurs premiers de n)
- sauf si celui du crypte fait des bêtises, mais j'en parlerai plus loin
- quid de l'attaque fréqentielle ?

le cas d'un texte

On pourrait remplacer chaque lettre par un chiffre, et faire ce que l'on a fait avec les chiffres

mais ceci serait stupide, car une attaque fréquentielle serait un succès !

La solution, on regroupe les codages des lettres par paquet de 3, et on travaille avec ces nombres là

Exemple : si on a la chaîne ant, (a est 97, n est 110, t est 116), alors on travaille avec le nombre 97 110 116.

Ceci est **très** efficace, et peu couteux pour le cryptage.

le cas d'un texte

On pourrait remplacer chaque lettre par un chiffre, et faire ce que l'on a fait avec les chiffres
mais ceci serait stupide, car une attaque fréquentielle serait un succès !
La solution, on regroupe les codages des lettres par paquet de 3, et on travaille avec ces nombres là
Exemple : si on a la chaîne ant, (a est 97, n est 110, t est 116), alors on travaille avec le nombre 97 110 116.
Ceci est **très** efficace, et peu couteux pour le cryptage.

le cas d'un texte

On pourrait remplacer chaque lettre par un chiffre, et faire ce que l'on a fait avec les chiffres
mais ceci serait stupide, car une attaque fréquentielle serait un succès !
La solution, on regroupe les codages des lettres par paquet de 3, et on travaille avec ces nombres là
Exemple : si on a la chaîne ant, (a est 97, n est 110, t est 116), alors on travaille avec le nombre 97 110 116.
Ceci est **très** efficace, et peu couteux pour le cryptage.

le cas d'un texte

On pourrait remplacer chaque lettre par un chiffre, et faire ce que l'on a fait avec les chiffres
mais ceci serait stupide, car une attaque fréquentielle serait un succès !
La solution, on regroupe les codages des lettres par paquet de 3, et on travaille avec ces nombres là
Exemple : si on a la chaîne ant, (a est 97, n est 110, t est 116), alors on travaille avec le nombre 97 110 116.
Ceci est **très** efficace, et peu couteux pour le cryptage.

pourquoi ce chiffrement est fort

Ce chiffrement repose sur deux hypothèses fondammentales

- Casser RSA revient à savoir résoudre le problème de la factorisation d'un grand nombre
- Il n'existe aucun algorithmes permettant de résoudre ce problème en temps raisonnables

Si ces hypothèses, ou seulement une des 2 tombe, le chiffrement RSA serait en danger.
Les attaques qui aboutissent sont celles qui réussissent à ne pas factoriser des grands nombre (cf le cas des textes)

pourquoi ce chiffrement est fort

Ce chiffrement repose sur deux hypothèses fondammentales

- Casser RSA revient à savoir résoudre le problème de la factorisation d'un grand nombre
- Il n'existe aucun algorithmes permettant de résoudre ce problème en temps raisonnables

Si ces hypothèses, ou seulement une des 2 tombe, le chiffrement RSA serait en danger.
Les attaques qui aboutissent sont celles qui réussissent à ne pas factoriser des grands nombre (cf le cas des textes)

pourquoi ce chiffrement est fort

Ce chiffrement repose sur deux hypothèses fondammentales

- Casser RSA revient à savoir résoudre le problème de la factorisation d'un grand nombre
- Il n'existe aucun algorithmes permettant de résoudre ce problème en temps raisonnables

Si ces hypothèses, ou seulement une des 2 tombe, le chiffrement RSA serait en danger.
Les attaques qui aboutissent sont celles qui réussissent à ne pas factoriser des grands nombre (cf le cas des textes)

pourquoi ce chiffrement est fort

Ce chiffrement repose sur deux hypothèses fondammentales

- Casser RSA revient à savoir résoudre le problème de la factorisation d'un grand nombre
- Il n'existe aucun algorithmes permettant de résoudre ce problème en temps raisonnables

Si ces hypothèses, ou seulement une des 2 tombe, le chiffrement RSA serait en danger.

Les attaques qui aboutissent sont celles qui réussissent à ne pas factoriser des grands nombre (cf le cas des textes)

pourquoi ce chiffrement est fort

Ce chiffrement repose sur deux hypothèses fondammentales

- Casser RSA revient à savoir résoudre le problème de la factorisation d'un grand nombre
- Il n'existe aucun algorithmes permettant de résoudre ce problème en temps raisonnables

Si ces hypothèses, ou seulement une des 2 tombe, le chiffrement RSA serait en danger.
Les attaques qui aboutissent sont celles qui réussissent à ne pas factoriser des grands nombre (cf le cas des textes)

quelques contraintes sur p et q : contraintes intuitives

- déjà, on travaille modulo $p \times q$, donc on a un nombre fini de valeur. Il faut donc prendre p et q grands. Mais aussi, pour assurer la sécurité du chiffrement, on les prend typiquement sur 1024 ou 2048 bits, taille qui n'est pas non plus excessivement grande.
- p et q ne doivent pas être friables, ie leurs facteurs premiers doivent être assez grands.
- même chose pour leur différence, car tester tous les facteurs entre $\sqrt{p \times q}$ et $\sqrt{p \times q} + |p - q|$ aurait un coût faible.

quelques contraintes sur p et q : contraintes intuitives

- déjà, on travaille modulo $p \times q$, donc on a un nombre fini de valeur. Il faut donc prendre p et q grands. Mais aussi, pour assurer la sécurité du chiffrement, on les prend typiquement sur 1024 ou 2048 bits, taille qui n'est pas non plus excessivement grande.
- p et q ne doivent pas être friables, ie leurs facteurs premiers doivent être assez grands.
- même chose pour leur différence, car tester tous les facteurs entre $\sqrt{p \times q}$ et $\sqrt{p \times q} + |p - q|$ aurait un coût faible.

quelques contraintes sur p et q : contraintes intuitives

- déjà, on travaille modulo $p \times q$, donc on a un nombre fini de valeur. Il faut donc prendre p et q grands. Mais aussi, pour assurer la sécurité du chiffrement, on les prend typiquement sur 1024 ou 2048 bits, taille qui n'est pas non plus excessivement grande.
- p et q ne doivent pas être friables, ie leurs facteurs premiers doivent être assez grands.
- même chose pour leur différence, car tester tous les facteurs entre $\sqrt{p \times q}$ *et* $\sqrt{p \times q} + |p - q|$ aurait un coût faible.

autres contraintes

il existe des alogrithmes de factorisation assez puissants (algorithme de rho de Pollard, $p-1$ de Pollard, algorithme de factorisation de Williams) qui impose respectivement que :

1. p et q ne soient pas friables
2. $p-1$ et $q-1$ ne soient pas friables
3. $p+1$ et $q+1$ ne soient pas friables

comment choisir alors p et q

- beaucoup de contraintes mathématiques
- mais il existe aussi des contraintes "humaines" : il faut que même ceux qui connaissent les algos de chiffrement ne puissent pas faire fuiter les clés
- La solution trouvée consiste à générer au hasard deux nombres premiers, et à vérifier qu'ils respectent les contraintes évoquées.

comment choisir alors p et q

- beaucoup de contraintes mathématiques
- mais il existe aussi des contraintes "humaines" : il faut que même ceux qui connaissent les algos de chiffrement ne puissent pas faire fuiter les clés
- La solution trouvée consiste à générer au hasard deux nombres premiers, et à vérifier qu'ils respectent les contraintes évoquées.

comment choisir alors p et q

- beaucoup de contraintes mathématiques
- mais il existe aussi des contraintes "humaines" : il faut que même ceux qui connaissent les algos de chiffrement ne puissent pas faire fuiter les clés
- La solution trouvée consiste à générer au hasard deux nombres premiers, et à vérifier qu'ils respectent les contraintes évoquées.

les contraintes sur e et d

- si on change e, il est préférable de changer n aussi
- il faut prendre e assez grand
- il faut prendre e tel que d vérifie $d \geq \frac{1}{4} \times n^{\frac{1}{4}}$

les contraintes sur e et d

- si on change e, il est préférable de changer n aussi
- il faut prendre e assez grand
- il faut prendre e tel que d vérifie $d \geq \frac{1}{4} \times n^{\frac{1}{4}}$

les contraintes sur e et d

- si on change e, il est préférable de changer n aussi
- il faut prendre e assez grand
- il faut prendre e tel que d vérifie $d \geq \frac{1}{4} \times n^{\frac{1}{4}}$

justification de la première contrainte

1. garder le même e et le même n ne présente pas de risques qu'un attaquant extérieur décrypte le message
2. ***MAIS***, si on envoie le même message, simultanément, à B et C, avec le même n, mais des e différent, alors B peut savoir quel codage correspond à quel codage pour C, et ainsi lire tout ce que C recevra de la part de A
3. exemple : si le message envoyé est "hi", et que B reçoit le message crypté : 18 42 et C reçoit 24 65. Si, dans un autre message de A vers C, B voit un 24, il saura que c'est un h

justification de la première contrainte

1. garder le même e et le même n ne présente pas de risques qu'un attaquant extérieur décrypte le message
2. ***MAIS***, si on envoie le même message, simultanément, à B et C, avec le même n, mais des e différent, alors B peut savoir quel codage correspond à quel codage pour C, et ainsi lire tout ce que C recevra de la part de A
3. exemple : si le message envoyé est "hi", et que B reçoit le message crypté : 18 42 et C reçoit 24 65. Si, dans un autre message de A vers C, B voit un 24, il saura que c'est un h

justification de la première contrainte

1. garder le même e et le même n ne présente pas de risques qu'un attaquant extérieur décrypte le message
2. ***MAIS***, si on envoie le même message, simultanément, à B et C, avec le même n, mais des e différent, alors B peut savoir quel codage correspond à quel codage pour C, et ainsi lire tout ce que C recevra de la part de A
3. exemple : si le message envoyé est "hi", et que B reçoit le message crypté : 18 42 et C reçoit 24 65. Si, dans un autre message de A vers C, B voit un 24, il saura que c'est un h

illustration, l'attaque de Hastad

- on suppose e petit. Pour illustrer, prenons $e = 5$, et que A envoie ces messages avec le même exposant e, mais n changeant.
- on rappelle :

théorèmes des restes chinois

Soient $k \in \mathbb{N}$, $n_1, \ldots, n_k$ des nombres premiers entre eux 2 à 2 et $y_1, \ldots, y_k$. Alors il existe un unique $x \leq n_1 \times \ldots \times n_k$ tel que, $\forall\ 1 \leq i \leq k,\ x \equiv y_i\ [n_i]$

- si on envoie le même message x (ici, x est par exemple un groupe de 3 lettres) à e personnes, alors x va vérifier : $\forall 1 \leq i \leq e,\ x^e \equiv y_i\ [n_i]$, et l'attaquant connait les y_i.
- le lien avec le théorème : en plus de pouvoir s'appliquer, le thm des restes a une preuve constructible, qui est facile à faire algorithmiquement
  comme $x \leq n_i$, on a $x^e \leq n_1 \times \ldots n_e$, et donc on peut trouver x^e, le message avec la réduction modulo les n_i Puis on calcule x.

illustration, l'attaque de Hastad

- on suppose e petit. Pour illustrer, prenons $e = 5$, et que A envoie ces messages avec le même exposant e, mais n changeant.
- on rappelle :

théorèmes des restes chinois

Soient $k \in \mathbb{N}$, $n_1, \ldots, n_k$ des nombres premiers entre eux 2 à 2 et $y_1, \ldots, y_k$. Alors il existe un unique $x \leq n_1 \times \ldots \times n_k$ tel que, $\forall\ 1 \leq i \leq k,\ x \equiv y_i\ [n_i]$

- si on envoie le même message x (ici, x est par exemple un groupe de 3 lettres) à e personnes, alors x va vérifier : $\forall 1 \leq i \leq e,\ x^e \equiv y_i\ [n_i]$, et l'attaquant connait les y_i.
- le lien avec le théorème : en plus de pouvoir s'appliquer, le thm des restes a une preuve constructible, qui est facile à faire algorithmiquement
  comme $x \leq n_i$, on a $x^e \leq n_1 \times \ldots n_e$, et donc on peut trouver x^e, le message avec la réduction modulo les n_i Puis on calcule x.

illustration, l'attaque de Hastad

- on suppose e petit. Pour illustrer, prenons $e = 5$, et que A envoie ces messages avec le même exposant e, mais n changeant.
- on rappelle :

théorèmes des restes chinois

Soient $k \in \mathbb{N}$, $n_1, \ldots, n_k$ des nombres premiers entre eux 2 à 2 et $y_1, \ldots, y_k$. Alors il existe un unique $x \leq n_1 \times \ldots \times n_k$ tel que, $\forall\ 1 \leq i \leq k,\ x \equiv y_i\ [n_i]$

- si on envoie le même message x (ici, x est par exemple un groupe de 3 lettres) à e personnes, alors x va vérifier : $\forall 1 \leq i \leq e,\ x^e \equiv y_i\ [n_i]$, et l'attaquant connait les y_i.
- le lien avec le théorème : en plus de pouvoir s'appliquer, le thm des restes a une preuve constructible, qui est facile à faire algorithmiquement
  comme $x \leq n_i$, on a $x^e \leq n_1 \times \ldots n_e$, et donc on peut trouver x^e, le message avec la réduction modulo les n_i Puis on calcule x.

illustration, l'attaque de Hastad

- on suppose e petit. Pour illustrer, prenons $e = 5$, et que A envoie ces messages avec le même exposant e, mais n changeant.
- on rappelle :

théorèmes des restes chinois

Soient $k \in \mathbb{N}$, $n_1, \ldots, n_k$ des nombres premiers entre eux 2 à 2 et $y_1, \ldots, y_k$. Alors il existe un unique $x \leq n_1 \times \ldots \times n_k$ tel que, $\forall\ 1 \leq i \leq k,\ x \equiv y_i\ [n_i]$

- si on envoie le même message x (ici, x est par exemple un groupe de 3 lettres) à e personnes, alors x va vérifier : $\forall 1 \leq i \leq e,\ x^e \equiv y_i\ [n_i]$, et l'attaquant connait les y_i.
- le lien avec le théorème : en plus de pouvoir s'appliquer, le thm des restes a une preuve constructible, qui est facile à faire algorithmiquement

  comme $x \leq n_i$, on a $x^e \leq n_1 \times \ldots n_e$, et donc on peut trouver x^e, le message avec la réduction modulo les n_i Puis on calcule x.

illustration, l'attaque de Hastad

- on suppose e petit. Pour illustrer, prenons $e = 5$, et que A envoie ces messages avec le même exposant e, mais n changeant.
- on rappelle :

théorèmes des restes chinois

Soient $k \in \mathbb{N}$, $n_1, \ldots, n_k$ des nombres premiers entre eux 2 à 2 et $y_1, \ldots, y_k$. Alors il existe un unique $x \leq n_1 \times \ldots \times n_k$ tel que, $\forall\ 1 \leq i \leq k,\ x \equiv y_i\ [n_i]$

- si on envoie le même message x (ici, x est par exemple un groupe de 3 lettres) à e personnes, alors x va vérifier : $\forall 1 \leq i \leq e,\ x^e \equiv y_i\ [n_i]$, et l'attaquant connait les y_i.
- le lien avec le théorème : en plus de pouvoir s'appliquer, le thm des restes a une preuve constructible, qui est facile à faire algorithmiquement
  comme $x \leq n_i$, on a $x^e \leq n_1 \times \ldots n_e$, et donc on peut trouver x^e, le message avec la réduction modulo les n_i Puis on calcule x.

illustration, l'attaque de Hastad

- on suppose e petit. Pour illustrer, prenons $e = 5$, et que A envoie ces messages avec le même exposant e, mais n changeant.
- on rappelle :

théorèmes des restes chinois

Soient $k \in \mathbb{N}$, $n_1, \ldots, n_k$ des nombres premiers entre eux 2 à 2 et $y_1, \ldots, y_k$. Alors il existe un unique $x \leq n_1 \times \ldots \times n_k$ tel que, $\forall\ 1 \leq i \leq k,\ x \equiv y_i\ [n_i]$

- si on envoie le même message x (ici, x est par exemple un groupe de 3 lettres) à e personnes, alors x va vérifier : $\forall 1 \leq i \leq e,\ x^e \equiv y_i\ [n_i]$, et l'attaquant connait les y_i.
- le lien avec le théorème : en plus de pouvoir s'appliquer, le thm des restes a une preuve constructible, qui est facile à faire algorithmiquement
  comme $x \leq n_i$, on a $x^e \leq n_1 \times \ldots n_e$, et donc on peut trouver x^e, le message avec la réduction modulo les n_i Puis on calcule x.

quelques remarqques

1. pour appliquer ce thm, il faut que les n_i soient 2 à 2 premiers entre eux, ce qui est le cas bien sûr, car le calcul du pgcd est logarithmique.
2. normalement, le calcul de la racine e^{ieme} est compliqué, mais ici e est petit

2 solutions

- on peut soit rajouter un décalage
- ou juste prendre e suffisamment grand

2 solutions

- on peut soit rajouter un décalage
- ou juste prendre e suffisamment grand

la dernière contrainte

la justification mathématique de cette attaque repose sur le développement en fraction continue de $\frac{e}{d}$

quand l'informatique quantique s'en mêle . . .

l'apport de l'informatique quantique est que l'on possède des algorithmes quantiques qui devraient pouvoir factoriser très rapidement des grands nombres, et ainsi casser RSA.

par exemple, l'algorithme de Shor

quand l'informatique quantique s'en mêle . . .

l'apport de l'informatique quantique est que l'on possède des algorithmes quantiques qui devraient pouvoir factoriser très rapidement des grands nombres, et ainsi casser RSA.
par exemple, l'algorithme de Shor

l'algorithme de Shor, en rapide

Entrée : un entier N, sortie : un facteur non trivial de N

1. choisir $2 \leq a < N$
2. claculer $pgcd(a, N)$. Si on obtient autre chose que 1, on s'arrête et on a eu de la chance. Sinon on continue
3. Soit f la fonction te que $f(x) = a^x \ [N]$. On calcule sa période p. **C'est ici que l'on a besoin d'un ordi quantique pour aller vite**
4. Si r est impair ou si $a^{r/2} \equiv 1 \ [N]$, on retourne à la première étape.
5. On calcule le $pgcd(a^{r/2} + 1, N)$ et $pgcd(a^{r/2} - 1, N)$, On obtient des nombres différents de 1 car N a un facteur commun avec $a^{r/2} + 1$ et avec $a^{r/2} - 1$

l'algorithme de Shor, en rapide

Entrée : un entier N, sortie : un facteur non trivial de N

1. choisir $2 \leq a < N$
2. claculer $pgcd(a, N)$. Si on obtient autre chose que 1, on s'arrête et on a eu de la chance. Sinon on continue
3. Soit f la fonction te que $f(x) = a^x \ [N]$. On calcule sa période p. **C'est ici que l'on a besoin d'un ordi quantique pour aller vite**
4. Si r est impair ou si $a^{r/2} \equiv 1 \ [N]$, on retourne à la première étape.
5. On calcule le $pgcd(a^{r/2} + 1, N)$ et $pgcd(a^{r/2} - 1, N)$, On obtient des nombres différents de 1 car N a un facteur commun avec $a^{r/2} + 1$ et avec $a^{r/2} - 1$

l'algorithme de Shor, en rapide

Entrée : un entier N, sortie : un facteur non trivial de N

1. choisir $2 \leq a < N$
2. claculer $pgcd(a, N)$. Si on obtient autre chose que 1, on s'arrête et on a eu de la chance. Sinon on continue
3. Soit f la fonction te que $f(x) = a^x$ $[N]$. On calcule sa période p. **C'est ici que l'on a besoin d'un ordi quantique pour aller vite**
4. Si r est impair ou si $a^{r/2} \equiv 1$ $[N]$, on retourne à la première étape.
5. On calcule le $pgcd(a^{r/2} + 1, N)$ et $pgcd(a^{r/2} - 1, N)$, On obtient des nombres différents de 1 car N a un facteur commun avec $a^{r/2} + 1$ et avec $a^{r/2} - 1$

l'algorithme de Shor, en rapide

Entrée : un entier N, sortie : un facteur non trivial de N

1. choisir $2 \leq a < N$
2. claculer $pgcd(a, N)$. Si on obtient autre chose que 1, on s'arrête et on a eu de la chance. Sinon on continue
3. Soit f la fonction te que $f(x) = a^x \ [N]$. On calcule sa période p. **C'est ici que l'on a besoin d'un ordi quantique pour aller vite**
4. Si r est impair ou si $a^{r/2} \equiv 1 \ [N]$, on retourne à la première étape.
5. On calcule le $pgcd(a^{r/2} + 1, N)$ et $pgcd(a^{r/2} - 1, N)$, On obtient des nombres différents de 1 car N a un facteur commun avec $a^{r/2} + 1$ et avec $a^{r/2} - 1$

l'algorithme de Shor, en rapide

Entrée : un entier N, sortie : un facteur non trivial de N

1. choisir $2 \leq a < N$
2. claculer $pgcd(a, N)$. Si on obtient autre chose que 1, on s'arrête et on a eu de la chance. Sinon on continue
3. Soit f la fonction te que $f(x) = a^x \ [N]$. On calcule sa période p. **C'est ici que l'on a besoin d'un ordi quantique pour aller vite**
4. Si r est impair ou si $a^{r/2} \equiv 1 \ [N]$, on retourne à la première étape.
5. On calcule le $pgcd(a^{r/2} + 1, N)$ et $pgcd(a^{r/2} - 1, N)$, On obtient des nombres différents de 1 car N a un facteur commun avec $a^{r/2} + 1$ et avec $a^{r/2} - 1$

l'algorithme de Shor, en rapide

Entrée : un entier N, sortie : un facteur non trivial de N

1. choisir $2 \leq a < N$
2. claculer $pgcd(a, N)$. Si on obtient autre chose que 1, on s'arrête et on a eu de la chance. Sinon on continue
3. Soit f la fonction te que $f(x) = a^x \ [N]$. On calcule sa période p. **C'est ici que l'on a besoin d'un ordi quantique pour aller vite**
4. Si r est impair ou si $a^{r/2} \equiv 1 \ [N]$, on retourne à la première étape.
5. On calcule le $pgcd(a^{r/2} + 1, N)$ et $pgcd(a^{r/2} - 1, N)$, On obtient des nombres différents de 1 car N a un facteur commun avec $a^{r/2} + 1$ et avec $a^{r/2} - 1$

l'algorithme de Shor, en rapide

Entrée : un entier N, sortie : un facteur non trivial de N

1. choisir $2 \leq a < N$
2. claculer $pgcd(a, N)$. Si on obtient autre chose que 1, on s'arrête et on a eu de la chance. Sinon on continue
3. Soit f la fonction te que $f(x) = a^x \ [N]$. On calcule sa période p. **C'est ici que l'on a besoin d'un ordi quantique pour aller vite**
4. Si r est impair ou si $a^{r/2} \equiv 1 \ [N]$, on retourne à la première étape.
5. On calcule le $pgcd(a^{r/2} + 1, N)$ et $pgcd(a^{r/2} - 1, N)$, On obtient des nombres différents de 1 car N a un facteur commun avec $a^{r/2} + 1$ et avec $a^{r/2} - 1$

preuve (rapide)

On a $N|a^r - 1$, donc $N|(a^{r/2} - 1) \times (a^{r/2} + 1)$. par défintion de la période, N ne peut pas diviser $a^{r/2} - 1$ et on on aussi que N ne divise pas $a^{r/2} - 1$, d'où le résultat.

conclusion ?

- RSA est très puissant, incassable à l'heure actuel si on fait attention à bien choisir les clés et les exposants
- RSa serait facilement cassable avec un ordi quantique
- Mais jusqu'à maintenant, aucun ordi quantique ne marche ou ne semble près de fonctionner
- Du coup on fait quoi?

conclusion ?

- RSA est très puissant, incassable à l'heure actuel si on fait attention à bien choisir les clés et les exposants
- RSa serait facilement cassable avec un ordi quantique
- Mais jusqu'à maintenant, aucun ordi quantique ne marche ou ne semble près de fonctionner
- Du coup on fait quoi?

conclusion ?

- RSA est très puissant, incassable à l'heure actuel si on fait attention à bien choisir les clés et les exposants
- RSa serait facilement cassable avec un ordi quantique
- Mais jusqu'à maintenant, aucun ordi quantique ne marche ou ne semble près de fonctionner
- Du coup on fait quoi?

conclusion ?

- RSA est très puissant, incassable à l'heure actuel si on fait attention à bien choisir les clés et les exposants
- RSa serait facilement cassable avec un ordi quantique
- Mais jusqu'à maintenant, aucun ordi quantique ne marche ou ne semble près de fonctionner
- Du coup on fait quoi?

vraie conclusion

- il existe d'autre problèmes mathématiques utilisés en crypto (logarithme discret).
- mais Shor peut aussi les casser
- mais une piste assez explorée est de chiffrer en utilisant les réseaux euclidiens, ce qui résisterait aux algorithmes quantiques

vraie conclusion

- il existe d'autre problèmes mathématiques utilisés en crypto (logarithme discret).
- mais Shor peut aussi les casser
- mais une piste assez explorée est de chiffrer en utilisant les réseaux euclidiens, ce qui résisterait aux algorithmes quantiques

vraie conclusion

- il existe d'autre problèmes mathématiques utilisés en crypto (logarithme discret).
- mais Shor peut aussi les casser
- mais une piste assez explorée est de chiffrer en utilisant les réseaux euclidiens, ce qui résisterait aux algorithmes quantiques