Texier Benjamin - Roblin Clément - Le Mesle Maxime

Projet d'informatique quantique

Sacha et ses amis dans …

L'algorithme de Shor et le chiffrement RSA

Polytech - ET5 INFO 2021/2022

I / Le chiffrement RSA

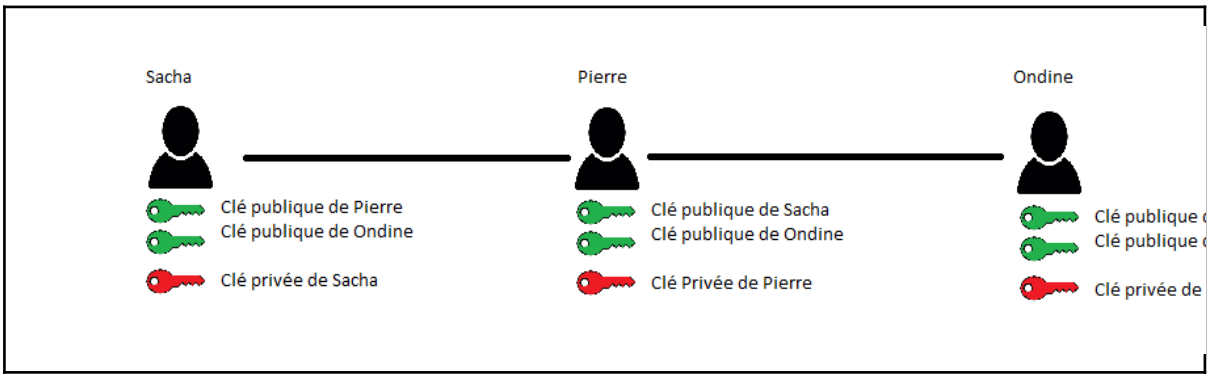
1.1 Principe de fonctionnement

Le chiffrement RSA est un algorithme de cryptage utilisé pour encoder des messages. Il est utilisé pour sécuriser des échanges entre les utilisateurs d'un réseau.

Chaque utilisateur du réseau possède 2 clés (i. e. un couple de 2 nombre), une privée qu'il garde et une publique qu'il diffuse à tous le monde.

- La clé publique permet aux autres utilisateurs du réseau de crypter le message de tel manière que seul l'émetteur de cette clé puisse le lire.
- La clé privée permet au destinataire de décrypter le message.

Ce schéma représente comment les clés sont réparties sur un réseau de 3 utilisateurs :



Ce qui sécurise le chiffrement RSA est l'usage de 2 nombres premiers, p et q , lors de la génération des clés (qui sera détaillé par la suite). RSA utilisant en partie de clé publique n=p*q , tout le monde peut récupérer sa valeur.

Pour craquer RSA , à savoir retrouver la clé privée d'un utilisateur pour décoder les messages qui lui sont destinés, il suffirait de retrouver p et q à partir de n.

Cependant la décomposition en nombre premiers est un algorithme très coûteux et choisis dans RSA sont très grands. C'est ce qui fait la force du RSA.

1.2 Algorithme de RSA

La création des clés se déroule ainsi :

1) Choix d'un p et q premiers : p et q doivent être très grands pour avoir un chiffrement robuste. On peut les obtenir via le crible d'Ératosthène. Il faut également que le message encodé m soit plus petit que n

2) Calcul de n = p * q

3) Calcul de phi(n) = (p - 1) * (q - 1)

4) Choix d'un exposant e vérifiant pgcd(e , phi(n)) =1 et tel que e<phi(n)

5) Calcul de d, qui est l'inverse de e % phi(n) via l'algorithme d'Euclide étendu

On obtient ainsi la clé publique (n,e), et la clé privée d

Avec la clé publique on peut ensuite crypter un message m :

message_crypte = m^e % n

et le décrypter avec la clé privée :

message_decrypte= message_crypte^d % n

Exemple d'utilisation :

Sacha prend p=7 et q=19 comme nombres premiers, il a donc n=133 et phi(n)=108.
Il choisit e=5 qui est bien premier avec phi(n), et il obtient avec Euclide étendu d=65.

Sacha diffuse alors sa clef publique (n et e) à Pierre et Ondine.

Ondine souhaite lui envoyer le message 10,qui correspond au nombre de pokéball que Sacha doit apporter. Elle récupère sa clé publique et crypte le message :

message crypté= 10^5 % 133 = 117

Sacha reçoit le message et le décrypte avec sa clé privée :

message décrypté = 117^65 % 133 =10

II / L'algorithme de Shor

2.1 Principe de fonctionnement

Le but de l'algorithme de Shor est , à partir d'un nombre N, de réussir à le décomposer en facteurs non triviaux. L'objectif est de trouver r et a tels que $a^r = 1\%N$, avec r pair

Pour cela l'algorithme se décompose en une partie quantique et une partie classique:

- La partie quantique sert à résoudre le problème de recherche d'ordre, soit pour un a pris aléatoirement de trouver un r tel que l'équation précédente est vérifiée

- La partie classique permet à partir du a et du r trouvés, de factoriser N car en général (a^(r/2)-1)(a^(r/2)+1)= 0 % N sont des facteurs non triviaux de N

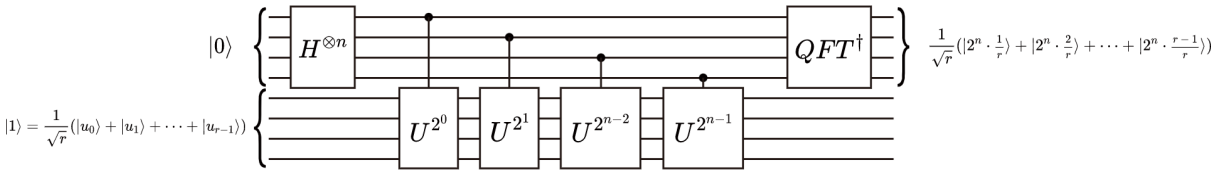
2.2 Partie classique

L'algorithme se déroule ainsi :

- On choisit un a , avec 1<a<N
- On calcule pgcd(a,N) au cas où a serait un facteur non trivial de N ce qui donnerait directement une solution
- Sinon on cherche r tel que a^r=1 % N . Pour cela on effectue cette recherche par période avec l'algorithme d'estimation de phase quantique
- Une fois le r obtenu , on vérifie que r est pair. Si c'est le cas, on vérifie notre condition (a^(r/2)-1)(a^(r/2)+1)= 0 % N ainsi que si r!=0 . Si c'est vrai r est trouvé
  Sinon, si r est impair ou que les conditions précédentes ne sont pas respectées on recommence l'algorithme du début

2.2 Partie quantique

Le cœur de cet algorithme est la partie quantique avec l'estimation de phase à implémenter avec Qiskit. L'idée est d'implémenter l'oracle suivant :



Les portes H servent à obtenir un état intriqué, en changeant les ket 0 en ket +. C'est grâce à ces portes que l'algorithme peut s'exécuter rapidement sur un ordinateur quantique en travaillant sur plusieurs états superposés. Il y a autant de H que de qubits (disons ici nb_qubits), qui est déterminé par la conversion en binaire de N.

Les portes U font office d'opérateurs unitaires. Chaque porte effectue le calcul suivant :

$$U \mid y \rangle \equiv \mid a*y \bmod N \rangle$$

L'idée est de répéter sur elle-même cette application jusqu'à obtenir l'état initial à nouveau. La superposition des états de ce cycle représente un vecteur propre de la porte U:4

$$|us\rangle = 1/\sqrt{r} = \sum_{k=0}^{r-1} e^{(-2\pi i s k r)} \mid a^k \bmod N \rangle$$

$$\text{Donc } U|us\rangle = e^{((2\pi i s)/r)} * |us\rangle$$

La partie QFT est utile pour mesurer la phase que nous cherchons. Il suffit d'additionner tous les vecteurs propres obtenus précédemment puisque l'état de base est la superposition de ces vecteurs propres. C'est ce que fait la porte QFT, qui représente une transformée de Fourier discrète quantique.

Enfin on ajoute des portes de mesure pour obtenir les différentes phases avec leurs probabilités. On déduit des résultats obtenus la phase réelle qui est le nombre trouvé divisé par 2^nb_qubits. Enfin, puisque la phase s'exprime sous la forme phase = s / r, il suffit d'approximer une fraction de numérateur et dénominateur entier égale à cette phase pour en déduire r. On impose que le dénominateur soit plus petit que N dans l'approximation pour que le r obtenu soit raisonnable. On conserve ensuite le r qui revient le plus parmi les r de toutes les phases mesurées. Il est possible que ce r ne soit pas juste, si tel est le cas, on répète à nouveau l'opération depuis la partie classique. Sinon on a trouvé un r, on peut donc poursuivre la partie classique.

III / Détails sur l'implémentation:

Notre implémentation de l'algorithme de Shor est théoriquement fonctionnelle pour tout N, mais en pratique n'est utilisable que jusqu'à 25 inclus d'après nos tests. Au-delà, soit le temps d'exécution est trop important, soit des messages d'erreurs nous sont renvoyés.

L'algorithme du RSA est parfaitement implémenté, de même que la génération de nombres premiers. Cependant nous utilisons une adaptation de ces algorithmes afin d'obtenir des entiers suffisamment petits pour que Shor fonctionne.

Conclusion :

S'il pouvait être implémenté correctement, l'algorithme de Shor pourrait mettre à mal la majorité des systèmes de sécurité actuels, puisque leur principe repose sur le temps inhumain nécessaire afin de pouvoir craquer les algorithmes utilisés, ce que Shor saurait faire en très peu de temps. Il faudrait alors trouver des algorithmes que même les calculs sur ordinateurs quantiques ne pourraient pas surpasser en un temps humain.

Actuellement, les limitations sur le matériel quantique ne permettent pas encore une bonne utilisation de l'algorithme de Shor, celui-ci fonctionnant que pour de très petites valeurs. On est encore bien loin des nombres gigantesques utilisés pour le chiffrement RSA. IBM a réussi en 2001 à trouver les solutions pour N=15, puis pour N=21 en 2012. En 2019, une tentative pour trouver les solutions pour N=35 a échoué suite à une accumulation d'erreurs, là où d'autres algorithmes sur ordinateur quantique ont réussi, comme la factorisation quantique variationnelle. Seulement, ces algorithmes sont de simples algorithmes gloutons qui n'ont pas la prétention d'obtenir de meilleures performances que les techniques classiques.

La team Rocket devra attendre encore un peu avant de pouvoir déchiffrer les messages de nos trois héros.

Bibliographie/Sitographie :

- Chiffrement RSA,dCode, consulté le 14/02/2022
https://www.dcode.fr/chiffre-rsa

- A. Bodin, F. Recher ,*Cryptographie* -Exo7Math - Arnaud Bodin, François Recher, consulté le 14/02/2022
http://exo7.emath.fr/cours/ch_crypto.pdf

- E. Poinsignon,S. Gendreau, *Cryptographie, le système RSA* 2009, consulté le 14/02/2022
https://www-fourier.ujf-grenoble.fr/~mcshane/STUDENT%20PROJECTS/gendreau/Projet.pdf

- Wikipédia, Shor algorithm
https://en.wikipedia.org/wiki/Shor%27s_algorithm

- Wikipédia, Algorithme d'Euclide étendu
https://fr.wikipedia.org/wiki/Algorithme_d'Euclide_%C3%A9tendu

-Le Crible D'Ératosthène En Python
https://www.mathweb.fr/euclide/2021/02/24/le-crible-deratosthene-en-python/

-Analyzing the performance of variational quantum factoring on a superconducting quantum processor
https://www.nature.com/articles/s41534-021-00478-z