RSA est-il fiable ?

Rémi Cros, Damien Dumas, Alex Fabre, Vincent Fougeras

Table des matières

Répertoire contenant le dossier RSA et les programmes python :
https://drive.google.com/folderview?id=0B4qaPhFcWCsaNElXUWdyYUxwdUk&usp=sharing

Résumé

La cryptographie est utilisée depuis la nuit des temps pour permettre à des hommes, des organismes ou des nations de transmettre des informations en s'assurant une certaine confidentialité. Depuis le chiffre de César jusqu'aux systèmes plus performants utilisés aujourd'hui, l'être humain n'a cessé de chercher des moyens plus efficaces les uns que les autres pour coder ses messages. Dans ce dossier, nous allons parcourir l'histoire en analysant différents systèmes de cryptage mis en pratique au fil des siècles. Nous présenterons chaque système, étudierons leurs fonctionnements et expliqueront leurs failles. Ensuite, nous nous attellerons à étudier un des systèmes les plus performants aujourd'hui : le RSA. Du nom de ses trois créateurs, ce système a révolutionné la cryptographie en proposant un code pratiquement inviolable. Après avoir expliqué son fonctionnement, ses faiblesses et fais une démonstration d'un chiffrement par RSA, nous pourrons ensuite parler du futur du RSA, et plus encore, du futur de la cryptographie.

Abstract

Cryptography is used since ancient times to allow men, organizations or nations to communicate information and to ensure a degree of confidentiality. Since the Caesar Cipher to more efficient systems used today, man has continued to look for more effective ways to encode messages, and in this presentation, we will go through history analyzing different encryption system in practice for centuries. We will present each system, study how they work and explain their faults. Then we will study one of the most powerful systems today: RSA. From the names of its three creators, this system revolutionized cryptography by providing a virtually inviolable code. After explaining its operation, its weaknesses and a demonstration of a RSA encryption, we will talk about the future of the RSA, and generaly, the future of cryptography.

I. Exemples de chiffrement

1. Définition : Différence entre système symétrique et asymétrique

Les systèmes de cryptographies se séparent en deux familles. Les systèmes symétriques, ou à clef privée, et les systèmes asymétriques, ou à clef publique. Le premier d'entre eux est le plus ancien système. Dans celui ci la clef de chiffrement est la même que celle de déchiffrement. Ce système possède l'avantage d'être plus rapide que son homologue asymétrique, mais il nécessite un partage de la clef privée à tous les correspondants. Mathématiquement il s'agit d'une fonction f qui permet de chiffrer un message et également de déchiffrer les messages chiffrés par cette fonction. Prenons A un message clair et B le message chiffré correspondant.

$$A \xrightarrow{f} B \xrightarrow{f} A$$

Le système asymétrique utilise deux types de clef, une clef privée et une clef publique. La clef publique est partagée avec les correspondants, elle leur sert à chiffrer le message. La clef privée n'est connue que par la personne qui déchiffre les messages. Ce système propose donc une sécurité supplémentaire dans le sens où la clef privée n'est jamais partagée avec les correspondants, donc il n'y a pas de risque qu'elle soit dérobée. D'un point de vue mathématique la fonction f correspond à la clef publique et permet de chiffrer un message, tandis que la fonction b correspond à la clef privée et permet de déchiffrer un message.

$$A \xrightarrow{f} B \xrightarrow{b} A$$

2. Système de César

Présentation

Ce système de cryptographie est un des plus anciens systèmes et aurait été utilisé par Jules César, qui utilisait un décalage de 3. Il s'agit d'un système symétrique, ou à clef privée. Le système de César est très basique et n'offre plus aucune sécurité depuis longtemps.

Fonctionnement

Il est facile de chiffrer un mot à l'aide de ce système, mais il est tout aussi facile de le déchiffrer. Il consiste à passer d'un mot A en un mot B en décalant la position des lettres dans l'alphabet d'un rang donné, on appelle cela un système par substitution mono-alphabétique.

Exemple

Nous allons illustrer ce système à l'aide d'un exemple. Alice souhaite envoyer un message chiffré à Bob. Elle chiffre son message selon le système de César avec un décalage de 3. Le message va être chiffré en décalant toutes les lettres de 3 dans l'alphabet.

```
  C E S A R
+ 3 3 3 3 3
= F H V D U
```

Bob obtient ainsi un message chiffré. En ayant connaissance du décalage, il peut déchiffrer le message en décalant les lettres dans le sens inverse à celui du chiffrement.

```
  F H V D U
- 3 3 3 3 3
= C E S A R
```

Analyse

Le système de César ne propose aujourd'hui aucune difficulté à déchiffrer même sans connaître le décalage. L'alphabet ne possédant que 26 lettres, il n'existe que 25 substitutions possibles (On ne compte pas une substitution d'une lettre par elle-même). Il est possible de le déchiffrer plus rapidement en analysant la fréquence des lettres.

Apparté sur l'analyse par fréquence

Dans chaque langage les lettres apparaissent plus ou moins fréquemment. En français, le E apparaît très fréquemment, tandis que le W est très rare. Si l'on connaît la fréquence d'apparition des lettres dans une langue, on peut déchiffrer un message codé et écrit dans cette langue sans posséder la clef de déchiffrement, en réalisant une analyse par fréquence.

Cette méthode de déchiffrement a cependant ses limites. Il est nécessaire que le chiffrement d'un message conserve la fréquence des lettres. Il faut également que le message soit suffisamment long pour obtenir une fréquence semblable à celle du langage.

E	S	A	N	T	I	R	U	L	O
17.76	8.23	7.68	7.61	7.30	7.23	6.81	6.05	5.89	5.34

Tableau des lettres les plus fréquentes dans la langue française

3. Système de Vigenère

Présentation

Le système de Vigenère, plus récent que le chiffre de César, est le premier à introduire la notion de “clef”. En effet, pour pouvoir chiffrer/déchiffrer un message, les utilisateurs doivent au préalable choisir une clef. Cette clef est le plus souvent représenté par un mot.

Fonctionnement

Pour coder un message en Vigenère, Alice doit choisir un mot servant de clef. Ensuite, elle réécrit la clef sous le texte, afin que chaque lettre du texte soit associée à une lettre de la clef. En partant du principe de l’alphabet correspond à un nombre de 0 à 25 (A−>0, B−>1,...,Z−>25), on ajoute les deux lettres, celle du mot à crypter, et celle de la clef. Le total(modulo 26) donne la position de la nouvelle lettre. Ainsi, une même lettre peut avoir une correspondance différente. Ainsi, il résout le problème du système de César, qui pouvait être cassé en analysant la fréquence des lettres.

Pour le décrypter, Bob n’a qu’à faire le chemin inverse, soit réécrire la clé qu’Alice lui a donné sous le mot crypter, et soustraire les lettres en elles.

Exemple

Si dessous, nous pouvons voir un exemple de message chiffré avec le système de Vigenère.

Texte en clair :

Bonjour je m'appelle Damien

B	O	N	J	O	U	R	J	E	M	A	P	P	E	L	L	E	D	A	M	I	E	N
1	14	13	9	14	20	17	9	4	12	0	15	15	4	11	11	4	3	0	12	8	4	13

Clef : ALIGOT

A	L	I	G	O	T
0	11	8	6	14	19

Message crypté :

B	O	N	J	O	U	R	J	E	M	A	P	P	E	L	L	E	D	A	M	I	E	N
1	14	13	9	14	20	17	9	4	12	0	15	15	4	11	11	4	3	0	12	8	4	13
0	11	8	6	14	19	0	11	8	6	14	19	0	11	8	6	14	19	0	11	8	6	14
A	L	I	G	O	T	A	L	I	G	O	T	A	L	I	G	O	T	A	L	I	G	O
1	25	21	15	2	13	17	20	12	18	14	8	15	15	19	17	18	22	0	23	16	10	1
B	Z	V	P	C	N	R	U	M	S	O	I	P	P	T	R	S	W	A	X	Q	K	B

Le mot crypté est : BZVPCNRUMSOIPPTRSWAXQKB

Ainsi, on peut voir que la même lettre n'a pas forcément la même traduction (le E qui est traduit par un M, un P, un S ou un K).

Analyse

Au XIXe siècle, le chiffre de Vigenère a été "cassé". Depuis lors, ce système n'est plus utilisé, de part sa facilité à le décrypter, même sans être en possession de la clef de cryptage.

Exemple de déchiffrage d'un code de Vigenère sans possession de la clef

Bob donne à Alice le code suivant, chiffré selon le système de Vigenère avec une clef de longueur connue 7.

```
cbntjdefhccavtvfdvfbpufdftgnzqwkysdufwcnfjguznzutvtlveonybyvldcf
lacryptographieestunedesdisciplinesdelacryptologiesattachantapro
upipfovtxgdglxfdcdgfibyvncywjogyhtrmtvpofkippewnzupgewykfrtthpvo
tegerdesmessagesassurantconfidentialiteauthenticiteetintegriteen
dctrleudqfjpeuogdsnifeuzincfdgwzpjfokdhtehfgoswrtegroyfhccavtvrf
saidantsouventdesecretsouclesellesedistinguedelasteganographiequ
kqotkqludsczolrpfnlvyopgdrhpflbdlolwefpdfduluprmztdefvmlecmakprt
ifaitpasserinapercuunmessagedansunautremessagealorsquelacryptogr
ldszfcgyrfenpudorvjykyhpcmtitpwvblwefphvpsfwovecqthpcmpgdhfkjwkd
aphierendunmessageinintelligibleaautrequequidedroitelleestutilis
spufawtgwroekbitkfxctgnvsectbpjepupgxvusqosdcfdrwidznaqchleupunc
eedepuislantiquitemaiscertainesdesesmethodeslesplusimportantesco
xdfwcnfjguzicoayjpcdmxvuckbipubegyhovmlhtbolwtprhtvnputsncf
mmelacryptographieasymetriquedatentdelafinduvingtiemesiecle
```

Alice doit déchiffrer le message sans informations additionnelles sur la clef. Il est possible de déchiffrer un tel message à l'aide, entre autres, d'une analyse par fréquence.
La première étape est de séparer la chaine de caractères en sous-chaines de longueur égale à celle de la clef. Alice doit donc créer des chaines de longueur 7.

```
cbntjde - fhccavt - vfdvfbp - ufdftgn - zqwkysd - ufwcnfj - ...
```

Alice peut ensuite réaliser une analyse fréquentielle des caractères situés à la même position dans leur sous-chaine. Dans cet exemple, Alice va analyser la fréquence des lettres parmi {c, f, v, u, z, u, …} afin d'obtenir la première lettre de la clef. En répétant cette opération plusieurs fois, on obtient la clef de chiffrage *rblclol* avec laquelle Alice va pouvoir déchiffrer le message complet.

4. Systèmes de Vernam

Présentation

Ce système de chiffrement aussi appelé masque jetable est connu pour être un chiffre sûr, c'est à dire que le message publique est impossible à décoder sans la clef. Cependant la préciosité de la clef rend donc sa transmission tout aussi périlleuse que le message lui-même.

Cette méthode à été mise en évidence par G.Verman en 1917 et améliorée par J.Mauborgne en introduisant la notion de clef aléatoire.

Fonctionnement

Pour encoder un message, Alice va devoir commencer par produire une clef "parfaitement" aléatoire, unique, et de longueur égale à son message. Elle va ensuite combiner son message à sa clef en additionnant par exemple le code de chaque lettre avec celle de la clef modulo 26. Le message ainsi produit est parfaitement incompréhensible et se révèle impossible à interpréter pour qui que ce soit. De plus puisque la clef est générée aléatoirement, une lettre n'est pas forcément codée par la même autre lettre (pas d'attaque par fréquence possible).

Son message est maintenant chiffré et peut être transmit publiquement à Bob. Elle doit aussi lui transmettre la clef pour qu'il puisse déchiffrer.
Pour déchiffrer le message Bob soustrait la clef au message publique et obtient facilement le message clair.

Exemple (voir programme Vernam.py en parallèle)

Message à chiffrer:	**ALLONSMANGER**
Code de chaque lettre:	01.12.12.15.14.19.13.01.14.07.05.18
Clef aléatoire:	UFKLWNDFPZTR
	21.06.11.12.23.14.04.06.16.26.20.18
	22.18. 23.01.11.07.17.07.04.07.25.10
Message publique:	**VRWAKGQGDGYJ**

Analyse

Aucune attaque ne peut être encourue contre le message publique. En effet le mot "DQK" pourrait aussi bien signifier "OUI" que "NON". En revanche une attaque peut être envisagée sur la clef si celle-ci est utilisé plus d'une fois.

Problème de l'utilisation unique de chaque clef

Soient M1 un message en clair et C1 son chiffre par la clef K et M2 un deuxième message en clair et C2 son chiffre par la même clef K.
L'attaque se fait sur C1 et C2 et a pour objectif de retrouver M1 et M2.
On va appliquer l'opérateur XOR (OU exclusif) sur chacun des bit de C1 et C2.

Conclusion

Ainsi, si un message publique encodé par Verman est parfaitement inviolable, la faiblesse de cette méthode réside dans sa clef. En effet elle doit respecter un aléa le plus naturel possible et être connue du destinataire pour décoder le message. Ainsi la transmission de la clef est tout aussi importante que le message à coder. Sachant que le clef fait par définition la même taille que le message, si l'on peut transmettre la clef secrètement autant transmettre le message directement.

II Le système RSA

1. Présentation générale

Les systèmes de chiffrement que nous avons vu jusqu'à présent sont tous dans la même catégorie. Ils font partie des systèmes à clef privée, ou chiffre symétrique. Il faut que Bob donne sa clef à Alice pour qu'elle décode le message. Et si la clef tombe dans de mauvaises mains alors l'intégrité du message est compromise. Le RSA fonctionne différemment, il utilise un système de clef publique et fait parti de la famille des systèmes de chiffre asymétrique.

Il a été mis au point par **R**ivest, **S**hamir et **A**dleman en 1976. L'idée est que Alice va émettre une clef publique dont Bob va se servir pour encoder son message. Pour déchiffrer, Alice va utiliser une clef privée qu'elle seule connaît.

Les inventeurs du RSA : de gauche à droite Adi Shamir, Don Rivest, et Len Adleman.

2. Fonctionnement du RSA

Fonctionnement

Pour créer un couple clef publique/privée qui permette un chiffrement et une sécurité optimale, le RSA se base sur une propriété mathématique simple:

Exemple (voir programme RSA.py en parallèle)

Alice souhaite envoyer un message à Bob. Il doit donc commencer par générer le couple de clefs publique/privée.

Il commence par choisir deux grands nombres premiers:

p=1459
q=1721

n=p*q=2510939

le produit p*q=n sera le module de chiffrement.

On va calculer l'indicatrice d'Euler phi=(p-1)(q-1) qui va permettre de trouver les indices de chiffrement et de déchiffrement.

phi=3003289

On cherche ‘e’ premier avec phi. Il sera notre indice de chiffrement.
On trouve e=7

Et ‘d’ indice de déchiffrement sera l’inverse de ‘e’ modulo phi.
On trouve d=716503

Nous avons ainsi tous les éléments pour constituer les deux clefs:

La clef publique (n,e) = (2510939,7)
La clef privée (n,d) = (2510939,716503)

Bob envoi à Alice la clef publique pour qu’elle chiffre son message.

Message à chiffrer : Lisa

Maintenant pour chaque lettre ‘m’ du message, l’algorithme va en prendre la valeur binaire puis réaliser l’exponentiation modulaire dessus.

```
foreach message as lettre
      lettre = pow(bin(lettre),e,n)   //pow(valeur,exposant,modulo)
      code+=lettre
```

Le code généré : 12938952108404

A noter qu’aucune analyse fréquentielle n’est possible sur ce code !
Pour déchiffrer le message Bob n’a plus qu’a réaliser la même opération en utilisant d comme indice de l’exponentiation modulaire.

3. Faiblesses du RSA

Soient:

La clef publique (n,e) = (2510939,7)
Le message chiffré : 12938952108404

N*'importe quel attaquant a accès à ces deux données.*
Pour déchiffrer le message, l'attaquant doit chercher l'indice de déchiffrement 'd' en utilisant seulement la clef publique (n,e) = (2510939,7)

*Il commence par chercher le couple unique (p,q) tel que p*q=n. Cette opération peut être longue. C'est cette opération qui doit etre rendu quasi impossible par l'utilisation de très grand nombres premiers (>4096 bits !). Ici nous avons des nombres à 8 bits !*

p=1459
q=1721

L'attaquant s'il parvient à retrouver p et q dispose maintenant de tout le matériel nécessaire pour déchiffrer le message.
Il va recalculer l'indicatrice d'Euler phi=(p-1)(q-1) pour trouver 'd' indice de déchiffrement premier avec 'e' modulo n.

phi=492350
et e=7
On trouve d=716503 premier avec 'e'.

Nous allons maintenant retrouver le message d'origine en utilisant la clef privée reconstituée. (n,d)=(2510939,492350)

Le message déchiffré : Lisa

La faiblesse du RSA est donc sérieusement mise en évidence par l'utilisation de petits nombres premiers. Reste que même si les clefs sont aujourd'hui construites avec des nombres de taille >4096 bits il faut renouveler les clefs fréquemment, environ une fois par mois pour les organismes de sécurité et que avec l'émergence d'ordinateurs toujours plus puissant, le RSA est sans cesse menacé.

4. Conditions nécessaires à la sûreté du RSA

Une des principales techniques pour garantir la sûreté du chiffrement est le renouvellement des clefs. Il est conseillé d'utiliser de nouvelles clefs régulièrement pour contourner les techniques de crack demandant du temps.

La longueur des clefs est également importante. Dans le cas d'un RSA naïf, avec une clef codée sur 256 bits, le crack peut être effectué avec peu de ressources et rapidement. C'est pour cela que les chiffrements RSA s'effectuent aujourd'hui avec des longueurs supérieures à 1024 bits. Le système SSL autorise les clefs RSA à partir de 2048 bits.

La génération aléatoire des deux nombres premiers n'est pas réellement aléatoire. L'utilisation des fonctions natives aux langages utilisés, comme random() pour Python, est déconseillée car il possible de retrouver les nombres qui ont été générés. Certains logiciels utilisent des fonctions jouant sur différents facteurs pour améliorer l'aléa.

```
We need to generate a lot of random bytes. It is a good idea to
perform some other action (type on the keyboard, move the mouse,
utilize the disks) during the prime generation; this gives the
random number generator a better chance to gain enough entropy.
```

5. Utilisation du RSA

Le protocole SSL (Secure Sockets Layer), et son successeur TLS (Transport Layer Security) sont des sur-couches au protocole TCP/IP, qui est un protocole utilisé pour gérer les connexions réseaux sur internet. Plusieurs services réseaux sécurisés utilisent ces protocoles : HTTPS, SSH, TCPS.

Ces protocoles de sécurité utilisent plusieurs systèmes cryptographiques, à différentes étapes du protocole. Lors de la communication, un système symétrique sera utilisé, comme le DES. Lors de l'échange de la clef symétrique avec le serveur, un système asymétrique est nécessaire pour éviter le vol de la clef. Dans 99% des cas le système utilisé est le RSA.

Les protocoles de sécurité comme le SSL ont été déchiffrés par la NSA d'après les dires de The Guardian. Cela serait dû à l'exploitation de failles dans le code des protocoles internet, et non à un déchiffrement du RSA ou des autres systèmes cryptographiques utilisés. Les nombreuses protections du RSA ne sont donc pas utiles si elles sont mal utilisées par les protocoles.

6. Avenir du RSA

Cryptographie acoustique

Décrypter une clé RSA grâce aux bruits du processeur ? C'est la mission un peu folle que se sont donnés trois chercheurs. Et les résultats furent concluant. En effet, ils ont réussi à casser une clé RSA de 4.096 bits. Ce tour de force a été réalisé en analysant les sons qu'émettait le processeur durant l'envoi de la clé. Avant leur expérience, les trois chercheurs avaient au préalable démontré que chaque clef RSA possédait un signal sonore spécifique. Ces sons, préalablement enregistrés à l'aide d'un microphone placé à côté du processeur, émettent différentes fréquences.

L'envoi de la clef RSA a été réalisé grâce au logiciel GnuPG 1.x. Suite à la publication de cette expérience, la société qui gère le logiciel a annoncé que le logiciel avait été mis à jour pour éviter ce genre d'attaque.

Shamir, un des créateurs du RSA, fait parti des trois chercheurs qui ont découvert ce moyen de craquer une clef RSA. Il indique que, bien que pour l'instant ce principe de décryptage n'est pas réalisable au quotidien, on peut facilement imaginer un futur proche où il suffira de poser son smart-phone à côté d'un ordinateur pour intercepter et déchiffrer un message chifré en RSA.

Ordinateurs quantiques

Le 9 décembre 2015, Google annonçait le bon fonctionnement de son ordinateur quantique, réalisé en partenariat avec la startup D-Wave et la NASA. Cet ordinateur est basé sur une technologie quantique, les qubit (quantum bit). A l'inverse des bits qui sont toujours dans une des 2 positions, 0 ou 1, les qubit peuvent être dans les deux états à la fois : c'est la superposition. Au moment de l'observation de ces qubit, ils se placent dans un des deux états. De plus, deux qubit ayant précédemment interagit entre eux sont liés par l'intrication quantique. Cet état permet de deviner l'état de plusieurs qubit intriqués en ne regardant qu'un seul d'entre eux.

En se servant de ces propriétés quantiques, l'ordinateur quantique D-Wave est capable de réaliser de nombreux calculs en parrallèle. Pour certains problèmes informatiques, il est théoriquement possible pour ce calculateur d'effectuer en 600 secondes (10 minutes) ce qu'un ordinateur classique effectuerait en $2*10^{90}$ secondes. Une technologie pareille, si elle venait à se répandre, permettrait de calculer la clef privée d'un code RSA très rapidement, même s'il est basé sur des nombres premiers très grands, ce qui supprimerait son intérêt.

Cependant, le D-Wave n'est pas encore capable de réaliser de tels calculs. Il est pour le moment capable de gérer un problème d'optimisation bien précis. La technologie quantique pourrait porter atteinte à la sécurité du RSA dans le futur mais ne pose pas de problème aujourd'hui.

Conclusion

Il est encore possible aujourd'hui de communiquer de façon sécurisé et beaucoup d'entreprise s'attellent à trouver les outils sécurisées du futur. Cependant le chiffre parfait n'existe pas et le RSA bien que prometteur depuis plus d'une quinzaine d'années, devra se renouveler et évoluer pour faire face à l'informatique de demain. Les attaquants sont de plus en plus nombreux, le nombre de connexions sécurisé croit avec le nombre d'utilisateurs en plus sur internet chaque jour, et c'est bien en multipliant les chiffres parfois en mixant les méthodes comme avec la cryptographie hybride que la sécurité continuera à évoluer.

Damien Dumas, Rémi Cros, Alex Fabre, Vincent Fougeras
Janvier 2015

Bibliographie

Système symétrique et asymétrique
http://www.di.ens.fr/~bresson/P12-M1/P12-M1-Crypto_3.pdf
http://www.di.ens.fr/~bresson/P12-M1/P12-M1-Crypto_4.pdf
César
http://www.primenumbers.net/Renaud/fr/crypto/Cesar.htm
Vigénère
http://www.nymphomath.ch/crypto/vigenere/index.html
http://www.apprendre-en-ligne.net/crypto/vigenere/decodevig.html (Rémi)
http://www.nymphomath.ch/crypto/vigenere/index.html (Rémi)
Vernam
https://fr.wikipedia.org/wiki/Masque_jetable
http://richard.esplins.org/static/downloads/linux_book.pdf chapitre 6.5.4
RSA
https://fr.wikipedia.org/wiki/Chiffrement_RSA
http://nciespoitiers.free.fr/math/index.html
Faiblessses
http://www.bortzmeyer.org/nouvelle-cle-pgp.html
SSL
http://www.sebsauvage.net/comprendre/ssl/
https://tools.ietf.org/html/rfc2818
https://www.sans.org/reading-room/whitepapers/protocols/ssl-tls-beginners-guide-1029
https://www.ssl247.fr/certificats-ssl/rsa-dsa-ecc
http://www.theguardian.com/world/2013/sep/06/nsa-encryption-revelations-roadmap-us
Quantique
http://www.futura-sciences.com/magazines/matiere/infos/actu/d/ordinateur-quantique-buzz-google-encore-loin-ordinateur-quantique-miracle-60811/
http://www.sciencesetavenir.fr/high-tech/informatique/20151214.OBS1309/les-incroyables-promesses-de-l-ordinateur-quantique.html
http://www.lexpress.fr/actualite/sciences/l-ordinateur-quantique-de-google-calcule-t-il-vraiment-plus-vite-que-son-ombre_1744865.html