Le coffre de HMS Titanic

La cryptographie RSA vingt ans après

JEAN-PAUL DELAHAYE

Comme tout le monde, par l'intermédiaire du système RSA quasi universel, vous utilisez des nombres premiers pour payer vos achats.

Le système de cryptage RSA a été inventé en 1977 par Ron Rivest, Adi Shamir et Len Adleman (dont les initiales forment RSA). Ces trois auteurs avaient décidé de travailler ensemble pour établir qu'un nouveau système de codage révolutionnaire, dénommé «système à clef publique» que W. Diffie et M. Hellman venaient d'inventer, était une impossibilité logique (autrement dit, que tout système de cryptage de cette nature présentait des failles). Ils ne réussirent pas dans leur projet, mais, au contraire, découvrirent un nouveau système à clef publique qui supplanta vite celui de W. Diffie et M. Hellman.

Nous expliquerons le fonctionnement du système RSA en gardant en tête une dissymétrie qui sous-tendra notre compréhension : il est facile (une seule opération) de multiplier deux nombres premiers ayant un nombre important de chiffres, 100 pour fixer les idées, et il est très difficile de déterminer les facteurs premiers d'un autre nombre de cette taille. Si l'on s'y prend naïvement, il faut diviser ce nombre par beaucoup de nombres plus petits et déterminer le reste : si celui-ci est nul, on a déterminé ainsi un diviseur. Les mathématiciens ont mis au point des moyens plus rapides que les divisions successives, mais le nombre d'opérations à effectuer reste considérable pour des nombres de taille importante. Ainsi il est actuellement impossible, pour des nombres appartenant à certaines catégories, de «factoriser» ceux de plus de 200 chiffres, même avec les ordinateurs les plus rapides et les algorithmes les plus performants.

Le système RSA est utilisé à la fois pour crypter des messages et pour les signer. Le message est envoyé par *Émetteur* et reçu par *Destinataire*. C'est un système à clef publique, ce qui signifie que :
– l'algorithme de calcul n'est pas caché, ni la clef de codage (appelée de ce fait clef publique) ;
– la connaissance de la clef publique de *Destinataire* permet à tous les *Émetteurs* de crypter les messages qu'ils destinent à *Destinataire*, mais seul ce dernier peut décrypter les messages qu'il reçoit des *Émetteurs*, grâce à sa clef de décodage privée, qu'il cache soigneusement. Les clefs publiques des *Destinataires* qui souhaitent recevoir des messages peuvent être publiées dans un annuaire ou sont obtenues, à la demande, en contactant préalablement celui à qui l'on veut faire parvenir un message.

Ce type de systèmes possédant une clef de décodage différente de la clef de codage (le système est dissymétrique) présente un avantage sur les systèmes classiques (dits symétriques, car une seule et même clef sert à la fois au codage et au décodage) : avant un échange, les deux interlocuteurs n'ont pas besoin de

1. CODAGE DES MESSAGES

Pour coder, *Émetteur* consulte un annuaire où il trouve la clef publique **Pub** de *Destinataire (a). Émetteur* code son texte avec cette clef publique **Pub** de *Destinataire* et l'envoie (message secret A). Tout le monde peut envoyer un tel message crypté à *Destinataire (b). Destinataire* décode, avec sa clef privée **Pri**, le message d'*Émetteur (c).* Seul *Destinataire* peut décoder les messages qui lui parviennent. Le cryptage-décryptage est fondé sur la propriété : **Pri**(**Pub**(texte)) = texte. Aucune méthode rapide pour avoir **Pri** à partir de **Pub** ne semble exister.

J. - M. Thiriet

se rencontrer pour convenir d'une clef secrète, qui devra rester connue d'eux seuls, ni n'ont besoin de faire circuler – sur un réseau informatique ou autre – une clef secrète, transmission qui bien sûr engendre un risque. Seule la clef publique, insuffisante pour le décryptage, est connue préalablement aux échanges cryptés entre les deux interlocuteurs.

Le RSA est un système à clef publique qui permet de signer – on dit aussi authentifier – des messages : pour cela, *Émetteur* transforme, avec sa clef privée, le texte à signer en un message codé. *Destinataire* reçoit le message et la signature qu'il peut décoder avec la clef publique d'*Émetteur* (le fait de pouvoir décoder la signature assure que c'est bien *Émetteur* qui a signé le message). On peut combiner les méthodes de codage et d'identification, comme indiqué sur la figure 2.

L'algorithme RSA est moins rapide que les algorithmes classiques à une seule clef, ce qui est un handicap lorsque l'on doit coder des messages volumineux. Aussi est-il fréquemment utilisé en combinaison avec un algorithme classique. Dans une première phase, le RSA (avec ses deux clefs) permet à *Émetteur* de transmettre secrètement à *Destinataire* une troisième clef secrète qu'ils seront seuls à connaître : *Émetteur* la fait parvenir à *Destinataire* en la cryptant avec la clef publique de *Destinataire*, qui, à l'arrivée, la décrypte avec sa clef secrète.

Dans un second temps, la troisième clef est utilisée avec un algorithme symétrique convenu pour envoyer rapidement à *Destinataire* le long message. Les algorithmes à clefs symétriques étant de cent à dix mille fois plus rapides que RSA (selon qu'il s'agit de versions matérielles ou logicielles), ce procédé à deux étages est aujourd'hui d'utilisation très commune.

VOUS L'UTILISEZ... SANS LE SAVOIR

Insistons sur le fait que ce jeu numérique élémentaire est aujourd'hui un système universel servant dans une multitude d'applications. Au cours des années, il a supplanté tous ses concurrents, qui sont abandonnés (au profit du RSA) soit parce qu'on y a trouvé des points faibles, soit parce qu'ils n'ont pas été assez étudiés pour qu'on soit convaincu de leur solidité.

La technologie RSA est protégée par un brevet dans certains pays (aux États-Unis, ce brevet expire en septembre 2000 ; en France, il n'y a pas de brevet). Elle a été commercialisée par plus de 350 entreprises, et l'on estime que plus de 300 millions de programmes installés utilisent le RSA : votre ordinateur contient sans doute de tels programmes, qui sont activés – sans même que vous le sachiez – lorsque, par exemple, vous souhaitez mener des transactions sécurisées sur l'Internet.

Le RSA est inclus dans de nombreux standards informatiques, dont le *Society for Worldwide Interbank Financial Telecommunication Standard* ou le *French Financial Industry's ETEBAC-5 Standard*, ou encore le *X9.44 draft Standard for the U.S. Banking Industry*. Le RSA est programmé aussi dans les systèmes d'exploitation de *Microsoft*, d'*Apple*, de *Sun* et de *Novel*. Il est intégré dans les cartes Ethernet et dans certaines cartes à puce bancaires. Un très grand nombre d'institutions gouvernementales, universitaires ou militaires l'utilisent de façon interne. Le réseau Internet en fait un usage systématique pour assurer la confidentialité du courrier électronique et authentifier les utilisateurs. Bref, le RSA est partout.

Nous avons vu que le système RSA est fondé sur la particularité qu'une opération inverse est plus difficile que l'opération directe. Examinons alors les opérations arithmétiques de RSA qui ont cette propriété.

Protocole RSA pour envoyer un message crypté :

(a) Création des clefs. Destinataire construit un quadruplet de nombres *(p, q, e, d)* tel que : p et q sont deux nombres premiers ; on pose $n = pq$; e est un entier premier avec le produit $(p-1)(q-1)$; d est un entier positif tel que $ed-1$ est un multiple de $(p-1)(q-1)$, c'est-à-dire tel que $ed = 1 \bmod (p-1)(q-1)$.

On sait alors d'après l'énoncé du théorème du RSA que, si A est un entier quelconque, alors : $A^{ed} = A \pmod n$, et c'est cette identité qui va tout faire fonctionner.

Grâce aux algorithmes probabilistes, on sait engendrer des nombres premiers de plusieurs milliers de chiffres de long en peu de temps ; le calcul de p et q est donc rapide. Trouver e et d est aussi une opération rapide, car l'algorithme d'Euclide qui permet de calculer ces deux nombres est un algorithme peu coûteux à exécuter. Au total, la constitution d'un quadruplet *(p, q, e, d)* est donc une opération rapide pour un ordinateur, même si l'on souhaite que p et q aient quelques centaines ou quelques milliers de chiffres.

2. SIGNATURE ET CODAGE SIMULTANÉS

SI *Émetteur*, non content d'envoyer un message, désire aussi le signer, de manière que *Destinataire* soit certain que c'est *Émetteur* qui l'a écrit, il code un texte de signature avec sa clef secrète **Pri'** et obtient le message codé B *(d)*. Il adjoint alors le message B au texte initial et il code le tout avec la clef publique **Pub** de *Destinataire (e)*. Il obtient ainsi le message A'. *Destinataire* décode le message A' avec sa clef privée **Pri**, puis le message B avec la clef publique d'*Émetteur* **Pub'**. Cette technique de codage repose sur la propriété : **Pub'**(**Pri**(**Pub**(**Pri'**(texte)))) = (texte).

3. LE THÉORÈME DU RSA

Le protocole RSA se fonde sur le théorème suivant.
Soient p et q deux nombres premiers. On pose $n = pq$. Si le nombre e est un entier premier avec le nombre $(p-1)(q-1)$, alors il existe un entier d positif, tel que $ed = 1 \bmod (p-1)(q-1)$, et, pour cet entier d et un entier A quelconque, on a : $A^{ed} = A \pmod n$.
La démonstration n'utilise que des mathématiques parfaitement connues depuis le XVIIIe siècle. Il est amusant de voir que les mathématiques du RSA attendaient depuis deux siècles qu'on leur trouve une application : les mathématiques étaient en avance. Ce n'est pas toujours le cas, et, aujourd'hui, en matière de preuve de complexité, la cryptographie attend des résultats mathématiques qui ne viennent pas (concernant les liens entre RSA et factorisation, ou concernant la difficulté de la factorisation) : les mathématiques sont en retard!

(b) Destinataire rend publics n et e, qui constituent la clef publique. Il la publie dans un annuaire ou la communique à *Émetteur*, qui la lui demande. Il ne communique surtout pas p, q ou d. Les nombres p et q peuvent être oubliés, car ils ne serviront plus à personne. Le nombre d constitue la clef secrète de *Destinataire*.

(c) Émetteur, qui veut transmettre une information secrète à *Destinataire*, transforme son information en un nombre entier A, inférieur à n (ou en plusieurs si nécessaire), en utilisant des conventions connues de tous (provenant, par exemple, des codes numériques des caractères typographiques, ou en prenant $a = 01$, $b = 02$, etc.).

(d) Émetteur calcule, *(voir la figure 4)*, grâce à la méthode d'exponentiation rapide, $B = A^e \pmod n$, envoie B à *Destinataire* par un canal qui n'a pas besoin d'être protégé (par exemple, le courrier électronique).

(e) Destinataire, pour décoder B, calcule $B^d \pmod n$, ce qui lui redonne A, car, d'après le théorème du RSA, on a $B^d = A^{ed} = A \pmod n$.

UTILISATION ET SÉCURITÉ DU RSA

Avec les techniques utilisées aujourd'hui pour programmer les systèmes RSA, les spécialistes estiment que doubler la longueur des clefs multiplie le temps de création des clefs par 16 (croissance en L^4, L étant la longueur des clefs) et multiplie, en revanche, le temps des opérations de codage, décodage, signature et authentification seulement par 4 (croissance en L^2). En se fondant sur l'hypothèse que casser le RSA demanderait des algorithmes non polynomiaux, c'est-à-dire plus longs, certains déduisent que, plus les machines seront puissantes, plus on pourra utiliser facilement de longues clefs, et que donc plus l'écart entre la puissance disponible et celle nécessaire pour casser le RSA sera grand. Autrement dit, plus le temps passe, plus RSA serait robuste et sûr. Nous verrons plus loin que cette présentation des choses est très optimiste et loin d'être mathématiquement établie.

Sur le plan pratique, les experts recommandent aujourd'hui d'utiliser des nombres n de 768 bits (232 chiffres décimaux) pour mettre en œuvre le RSA dans le cas de données pas trop importantes, mais ils conseillent 1 024 bits (309 chiffres décimaux) pour un usage commercial et 2 048 bits (617 chiffres décimaux) pour avoir une garantie se prolongeant sur une longue période de temps. Les clefs de 512 bits (155 chiffres décimaux), encore très utilisées, ne devraient plus l'être, car on a réussi, en août 1999, à factoriser un nombre n (produit de deux grands nombres premiers) de 512 bits.

La confiance dans la sécurité du RSA n'est pas due à la démonstration théorique que ce système est sûr, car une telle démonstration n'existe pas (nous allons y revenir). La confiance affichée provient de l'échec, répété depuis plus de 20 ans, de toutes les tentatives entreprises pour casser ce système, tentatives qui n'ont conduit qu'à la formulation de quelques recommandations pour le choix des paramètres p, q, e, d. D'autres systèmes concurrents du RSA sont peut-être aussi bons ou même meilleurs, mais aucun ne bénéficie de la validation par l'usage et la robustesse aux attaques dont le RSA peut se prévaloir. Le fait d'avoir été l'un des premiers lui a donné un avantage qui, puisqu'il résiste aux tentatives d'effraction, le maintient en tête.

Sur le plan théorique, la situation est décevante et le restera certainement longtemps. Celui qui sait factoriser $n = pq$ retrouve ensuite facilement d. Inversement, les mathématiques montrent que celui qui connaît n, e et d peut trouver rapidement p et q. La robustesse du RSA apparaît donc liée à la difficulté de la factorisation. Malheureusement, il n'est pas exclu que quelqu'un, sans réussir à obtenir d ni p ni q, puisse décrypter un message : autrement dit, il n'a pas été prouvé que la difficulté du RSA est équivalente à celle de la factorisation. Deux attaques théoriques du RSA sont donc envisageables. Celle passant par la factorisation : quiconque sait factoriser les nombres de la taille de $n = pq$ sait lire comme à livre ouvert tous les messages cryptés par le RSA. Celle contournant la factorisation, dont personne n'a su établir qu'elle était impossible et pour laquelle, au contraire, on a proposé récemment des arguments indiquant qu'elle devait être sérieusement crainte. Dan Boneh et Ramarathnam Venkatesan ont en effet établi en 1998 que casser le RSA, lorsqu'il est utilisé avec des exposants e trop petits, n'est pas équivalent à factoriser n.

TENTATIVES POUR BRISER LE RSA

D'autres consignes et mises en garde sont d'ailleurs formulées à destination des utilisateurs du RSA. Certains nombres premiers p et q doivent être évités : on conseille de prendre p et q de taille proche, et d'éviter que les nombres $p + 1$, $p - 1$, $q + 1$, $q - 1$ soient faciles à factoriser (car alors n risquerait de l'être aussi par l'utilisation d'algorithmes spécialisés de factorisation qui savent tirer parti des factorisations de $p + 1$, $p - 1$, $q + 1$, $q - 1$). Un conseil évident, mais à ne pas oublier, est que le nombre n ne doit pas être choisi par plus d'une entité : sinon, l'utilisation des diverses clefs publiques des utilisateurs du même nombre n risquerait de donner accès aux facteurs de n.

Plus subtil, mais très important, il ne faut jamais accepter de signer un message en apparence aléatoire soumis par un inconnu. En effet, en choisissant bien ce message, l'inconnu peut en déduire la signature d'un message qu'il aura choisi lui-même au préalable, et vous vous retrouveriez donc à devoir assumer une signature que vous n'avez pas donnée.

L'utilisation d'exposants e trop petits doit être évitée.

Il faut compléter les textes, avant de les coder, par des bouts choisis aléatoirement et surtout pas par de longues séries de blancs ou de zéros.

Une attaque astucieuse proposée par P. Kocher consiste à mesurer minutieusement le temps de calcul (ou la consommation électrique) demandé pour le codage, ce qui permet, si l'on dispose de suffisamment de données, de retrouver les clefs utilisées. Appliquée aux cartes à puce, cette méthode se révélerait catastrophique. Heureusement, se prémunir contre cette attaque est assez facile : il suffit de s'arranger pour que le temps de calcul (ou la consommation électrique)

apparent soit faussé, en opérant des calculs supplémentaires inutiles une fois le codage réalisé, avant d'envoyer l'information que le codage est terminé.

ON NE S'IMPROVISE PAS CRYPTOGRAPHE

De nombreuses erreurs dans la gestion imprudente ou maladroite des clefs sont aussi à l'origine de décryptages inopinés des messages considérés inviolables, car obtenus avec RSA. Une prudence à tous les niveaux est indispensable pour en arriver à des systèmes fiables. Dans un article paru en 1999, Dan Boneh, qui est le grand spécialiste des attaques du RSA proposait cette conclusion rassurante : «En ce moment, il apparaît que l'on peut avoir confiance dans la sécurité que procure le RSA au monde numérique.» Bruce Schneier, un autre spécialiste international de la cryptographie très préoccupé des réalisations opérationnelles, insiste sur le fait que la plupart des systèmes de cryptographie qu'on a réussi à casser l'ont été, non par l'attaque réussie du noyau mathématique du système, mais à cause de faiblesses dans la mise en œuvre : mauvaise génération des clefs, mots de passe mal choisis ou mal protégés, contournement possible de l'algorithme mathématique, etc.

L'idée défendue que le noyau mathématique du RSA est inviolable est seulement une conclusion expérimentale tirée de l'échec des tentatives connues d'attaques du RSA (lorsqu'on l'utilise en respectant les consignes formulées par les spécialistes… dont la liste s'allonge chaque année). Cette conclusion n'interdit pas que le risque théorique lié au RSA puisse être jugé grand, puisqu'on n'a ni réussi à montrer que la factorisation est difficile (aucun espoir n'existe à court terme qu'on y arrive), ni même réussi à montrer que le RSA est aussi difficile que la factorisation.

À l'évidence, si une agence de renseignements a découvert quelque chose concernant la factorisation ou s'appliquant directement au RSA sans passer par la factorisation, elle ne rendra pas publique sa découverte et, au contraire, fera tout pour la maintenir secrète le plus longtemps possible. Dire cela n'est pas faire preuve d'une méfiance maladive, mais du simple bon sens. La tranquille assurance exprimée par les spécialistes à propos de la sécurité du RSA apparaît incompréhensible, et parfois même assez louche.

Dix fois, dans l'histoire des services d'espionnage, un système de codage a été considéré cent pour cent sûr et inviolable par ceux qui l'utilisaient, alors qu'au même moment un service ennemi ayant percé le secret du système lisait tranquillement tous les messages qui tombaient entre ses mains. Cela s'est produit pendant la Première Guerre mondiale : les services français décodaient chaque nuit les messages destinés aux U-boot tapis au fond de la Méditerranée, ce qui permettait de modifier avant l'aube les programmes de navigation des bateaux menacés. À nouveau, pendant la Seconde Guerre mondiale, cela s'est produit avec la fameuse affaire de la machine *Enigma*, où s'illustra Alan Turing. Signalons aussi l'extraordinaire secret qui a entouré cette aventure impliquant les centaines de personnes qui ont travaillé au Bletchley Park, près de Londres, qui toutes sont restées muettes pendant trente ans : le sens patriotique est une motivation suffisante pour faire taire même le mathématicien le plus soucieux du partage de ses découvertes. Il me semble que, si un mathématicien canadien (même s'il est universitaire) découvre quelque chose d'important sur la factorisation, il le gardera pour les militaires canadiens, de même qu'un mathématicien français réservera ses théorèmes cryptographiques cruciaux pour les militaires français, etc.

Pourquoi donc, alors que des moyens considérables sont consacrés à la cryptanalyse – et donc à toutes les mathématiques et techniques qui y sont liées – par les agences spécialisées de renseignements puissantes et fortunées, ne pas envisager sérieusement, qu'ignorée des circuits de la recherche universitaire (et provenant peut-être d'un universitaire patriote), une découverte fondamentale faisant tomber le RSA ait été faite et reste cachée?

Même si l'on n'est pas paranoïaque, en l'absence de résultats mathématiques établissant la sécurité du RSA, il est sage de ne pas trop lui faire confiance. Réalise-t-on, quand on utilise la méthode de cryptage à deux étages évoquée plus haut (RSA pour communiquer une clef, puis algorithme symétrique), que si un seul des deux étages est percé c'est tout le système qui coule? Comment ne pas penser à de la désinformation quand on lit, en contradiction avec tout bon sens, que le système à deux étages est d'autant plus solide qu'il utilise deux méthodes cryptographiques !

A. M. Odlyzko a publié un bilan sur le cryptosystème à double clef de Merkle et Hellman fondé sur le problème du sac à dos (un problème combinatoire de rangement). Grâce à certaines propriétés mathématiques le reliant à d'autres problèmes assez bien compris, ce cryptosystème fut considéré un temps comme plus prometteur que le RSA. Malheureusement, il fut cassé, et les variantes qui ont été proposées aujourd'hui de ce système se sont révélées pour l'essentiel aussi fragiles que la version originale, tant et si bien que du système le plus prometteur d'il y a 15 ans il ne reste rien aujourd'hui. N'en sera-t-il pas de même pour le RSA dans 15 ans, et tout ce qui

4. LE CALCUL DE L'EXPONENTIATION RAPIDE

Dans la mise en œuvre aussi bien des algorithmes probabilistes pour engendrer des nombres premiers que pour l'implémentation du RSA, on a besoin de calculer rapidement A^e mod n. Voici une méthode (dont le principe général est emprunté à la multiplication égyptienne) qui permet ce calcul.

Calcul de A^e mod n

- partir du triplet $(A, e, 1) = (P, J, R)$ puis :
- si J est pair, passer à $(P^2 \bmod n, J/2, R \bmod n)$
- si J est impair, passer à $(P^2 \bmod n, (J-1)/2, R \times P \bmod n)$
- s'arrêter si $J = 1$, le résultat est $R = P \bmod n$

Exemple :

$A^{13} \bmod n \longrightarrow (A,13,1) \longrightarrow (A^2 \bmod n, 6, A) \longrightarrow (A^4 \bmod n, 3, A) \longrightarrow (A^8 \bmod n, 1, A^5) \longrightarrow R = A^{13} \bmod n$. Avec $n = 5$ et $A = 2$:

$213 \bmod 5 \longrightarrow (2,13,1) \longrightarrow (4, 6, 2) \longrightarrow (1, 3, 2) \longrightarrow (1, 1, 2) \longrightarrow R = 2$

5. EXEMPLE NUMÉRIQUE SIMPLIFIÉ DE CODAGE RSA

$p = 47$, $q = 71$, $n = pq = 3337$, $(p-1)(q-1) = 3220$, $e = 79$, $d = 1019$

On vérifie que $ed = 80501 = 25 \times 3220 + 1 = 1 \bmod 3220$

Prenons un message préalablement transformé en un nombre :

$m = 6882326879666683$

On découpe m : $m_1 = 688$ $m_2 = 232$ $m_3 = 687$ $m_4 = 966$ $m_5 = 668$ $m_6 = 3$

Le premier bloc est chiffré par le nombre $c_1 = 688^{79} \bmod 3337 = 1570$;

de même : $c_2 = 2756$ $c_3 = 2091$ $c_4 = 2276$ $c_5 = 2423$ $c_6 = 158$.

Le déchiffrement du message se fait en calculant :

$c_1 = 1570^{1019} \bmod 3337 = 688$, etc.

aura été crypté avec lui ne sera-t-il pas alors accessible à chacun facilement?

Citons encore le cas du *RSA for paranoids* proposé par A. Shamir (le «S» du RSA) en 1995. Il s'agit d'une variante du RSA, destinée à rassurer les utilisateurs inquiets du RSA classique en leur permettant d'utiliser, sans trop augmenter les coûts de codage, des nombres *n* plus longs. Trois ans après, ce prétendu renforcement du RSA était cassé par H. Guilbert, D. Gupta, A. Odlyzko et J.-J. Quisquater, qui en identifiaient de graves points faibles. Ainsi, l'un des inventeurs du RSA, croyant renforcer sa méthode dans le but de satisfaire les paranoïaques, a renforcé leurs craintes. Si nous souhaitons protéger des données pour une longue période de temps, soyons conscients que tout s'appuie, en cryptographie mathématique, sur un état de l'art qui peut changer rapidement et dont la réalité est peut-être déjà totalement autre que celle présentée au public.

Le grand nombre de systèmes nouveaux inventés chaque année et qu'on exhibe dans le but de calmer les inquiets («de toutes les façons, si tel système est cassé vous pourrez toujours utiliser celui-là») ne résout en rien le problème, qui est de disposer d'UN système prouvé robuste. Rien ne ressemble plus à un système cryptographique solide qu'un système cryptographique faible, et aujourd'hui personne ne dispose vraiment des moyens de garantir sérieusement aucun des systèmes largement utilisés.

6.LE RECORD DE FACTORISATION DU 22 AOÛT 1999

Le travail nécessaire aux grandes factorisations est complexe : voici quelques détails sur le calcul mené pour venir à bout du RSA-155. Le 22 août 1999, une équipe de l'Institut national de recherche en mathématiques et en science informatique d'Amsterdam annonçait avoir cassé la clef numérique de 512 bits (155 chiffres décimaux), problème connu sous le nom du RSA-155. Le résultat trouvé tient en quelques lignes ; c'est la factorisation d'un nombre de 155 chiffres en deux nombres premiers de 78 chiffres :

109417386415705274218097073220403576120037329454492059909138
421314763499842889347847179972578912673324976257528997818337
970765372440271467435315933543338 97
=102639592829741105772054196573991675900716567808038066803341
933521790711307779 × 106603488380168454820927220360012878 67
920795857598929152227060823719306280864 3

Trois cents ordinateurs divers (stations de travail et PC) ont travaillé à ce résultat cumulant un total de calcul évalué à 8 000 Mips année (1 Mips = un million d'instructions par seconde). Le déroulement des opérations s'est étalé sur une période de trois mois et demi. La primalité des facteurs a été évaluée par deux méthodes différentes. L'algorithme utilisé est le GNFS *(General Number Field Sieve)*, qui déjà avait permis de battre le précédent record de factorisation, le RSA-140, en février 1999.

Une légère amélioration de la méthode utilisée pour le RSA-140 a été utilisée. Si la méthode du RSA-140 avait été reprise sans modification, 14 000 Mips années auraient été nécessaires. Ainsi, en six mois, les progrès théoriques ont économisé 40 pour cent du calcul. En y ajoutant les progrès technologiques des microprocesseurs et la capacité accrue des réseaux pour faire coopérer des ordinateurs, la puissance a été doublée en six mois.

Le travail s'est déroulé en une série complexe de phases nécessitant des ordinateurs et des combinaisons d'ordinateurs variées. La première phase a consisté à rechercher un bon jeu de paramètres pour la méthode. On évalue à 100 Mips année la quantité de calculs faite pour cette phase, qui a duré neuf semaines en utilisant une seule machine (de type 250 Mhz *GI Origin* 2000). Cette partie du calcul ne se laisse pas distribuer sur un réseau.

La seconde phase, qui a été distribuée sur un réseau de 300 machines réparties dans le monde entier, est aussi la plus coûteuse en calcul. Elle a fourni des relations qui sont au cœur de la méthode. Un total de 124 millions de relations a ainsi été engendré, dont 39 millions apparaissaient en double (à cause de la méthode de calcul distribué).

Le tri des relations et la préparation de la matrice du système linéaire à résoudre ont été réalisés au CWI, ce qui a demandé un mois de travail. La matrice résultante comportait 6 699 191 lignes et 6 711 336 colonnes.

La résolution du système d'équations a alors demandé 224 heures et 2 gigaoctets de mémoire centrale à un ordinateur *Cray C916*, un ordinateur vectoriel destiné à mener de très gros calculs (par exemple en météorologie), calculs impossibles à répartir sur un réseau de machines moyennes.

ALTERNATIVE QUANTIQUE?

L'usage de la cryptographie quantique, dont la garantie ne repose pas sur des conjectures mathématiques, mais sur des principes de physique fondamentale, résoudra peut-être ce problème. La maturité de cette technique la rend utilisable aujourd'hui pour les applications où l'émetteur et le récepteur ne sont pas éloignés de plus de cent kilomètres.

Un autre résultat lié aussi à la mécanique quantique contribue à rendre encore plus méfiant vis-à-vis de la cryptographie mathématique. Peter Shor a montré en 1994 qu'avec un ordinateur quantique on peut factoriser efficacement des nombres entiers. Si l'on réussit à construire de telles machines, tout un pan de la cryptographie mathématique sera définitivement compromis. Les ordinateurs quantiques aujourd'hui sont de simples fictions théoriques dont la difficile mise au point devrait demander des années (si elle se révèle possible) ; cependant, là encore, l'avance possible de certains services secrets n'est pas à exclure et, si elle est réellement sensible, alors il se peut que pour eux la factorisation des clefs de 1 024 bits ou de 2 048 bits du RSA soit devenue un jeu d'enfants. Gilles Brassard, chercheur canadien renommé, passé de la cryptographie mathématique à la recherche sur les techniques quantiques, remarque que la mécanique quantique non seulement fait percevoir la fragilité fondamentale de la cryptographie mathématique, mais propose une solution pour échapper aux incertitudes dont elle ne réussit pas à se libérer.

D BONEH, *Twenty Years of Attacks on the RSA Cryptosystem,* in *Notice of the AMS* (American Mathematical Society), pp. 203-213, février 1999.

D. BONEH et R. VENKATESAN, *Breaking RSA May Be Easier Than Factoring,* Eurocrypt 1998, LNCS 1403, Springer-Verlag, pp. 59-71.

P. C. KOCHER, *Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS and other Systems,* Crypto 1996, LNCS 1109, Springer-Verlag, pp. 104-113.

A.M. ODLYZKO, *The Rise and Fall of Knapsack Cryptosystems,* ATT Bell Laboratories, Murray Hill, 1990.

B. SCHNEIER, *Cryptographie appliquée,* Thomson Publishing France, Paris, 1995.

J. STERN, *La science du secret,* Odile Jacob, Paris, 1998.