行政院國家科學委員會專題研究計畫成果報告

RSA 大數分解

RSA Factoring

計畫編號：NSC 89-2213-E-009-007
執行期限：88 年 08 月 01 日至 89 年 07 月 31 日

主持人：陳榮傑　　國立交通大學 資訊工程系
計畫參與人員：張仁俊　　國立交通大學 資訊工程系
胡鈞祥　　國立交通大學 資訊工程系
洪政宏　　國立交通大學 資訊工程系
楊淑華　　國立交通大學 資訊工程系

一、中文摘要

在 RSA 公鑰密碼系統中，每個使用者都擁有兩把鑰匙—公鑰與私鑰。公鑰公布給所有的人知道，私鑰則由自己秘密保存著，若甲方想要把訊息送給乙方且不想讓其他人知道訊息的內容，甲方可以用乙方的公鑰將訊息加密後送出，加密過的訊息只有擁有私鑰的人才能解開讀到真正的內容。在正常的情形下，只有乙方一人知道這把解密的鑰匙。

公鑰密碼系統的安全性在於幾乎無法由公鑰計算推導出私鑰，也就是靠這種計算不可行性(computational infeasibility)才得以保密。假如藉計算推導確實不易取得私鑰，那麼我們可以稱這系統是安全的。RSA 密碼系統就是利用大數分解相當困難的這個事實來製造公鑰與私鑰。因為大家公認分解一個大的數字是困難的，所以說 RSA 系統可說是相當安全的。

在本計畫中我們想更深入探討大數分解這個主題：到底我們要取多大的數字才會讓 RSA 不可能被破解？八０年代初期，人們認為 100 位的數字便足夠安全，但現在的電腦已經有足夠的計算能力可以分解。1997 年 RSA 的三位發明者 Rivest、Shamir、Adleman 於 Scientific American 提出著名的 129-digit RSA 挑戰題，預言需要四億億年才可分解出來。但在 1994 年，有人在 Internet 上用二次篩選分解法只花了八個月便完成這項工作。1996 年四月更有人以六至七倍快的一般代數體篩選分解法，成功地解決 130-digit RSA 挑戰題。現在，很多人是用 155-digit 數字來保護他們重要的資料。

大數分解的進步主要歸功於硬體的速度越來越快及更多快速的分解演算法被提出，目前著名的演算法有 Rho 分解法、因數基集分解法、連分數分解法、二次篩選分解法、p-1 分解法、橢圓曲線分解法、代數體篩選分解法。鑑於國內密碼學學術界對於基礎理論的研究並不多，我們將對這領域做深入的研究，設計出更有效率的大數分解演算法。在應用方面，我們也將發展一套大數分解的軟體，以供國內學界使用。

關鍵詞：RSA 公鑰密碼系統、大數分解、二次篩選分解法、橢圓曲線分解法、代數體篩選分解法

Abstract

RSA is a public key cryptosystem, where each party holds two keys: a public key and a corresponding secret key. The public key is accessible by the public while the secret key is always kept secret. You can encrypt a message using the recipient's public key and only the recipient can decrypt the message with secret key. No other people know the secret key.

The security of the public key

cryptosystem is ensured by the fact that it is too hard or, computationally infeasible, to derive the secret key from the public key. If a third party wants to decrypt the message, he should factor a number that is part of the public key. Since factoring a large number is commonly believed too hard, RSA utilizes this fact to produce the public key and the secret key to ensure security.

In this project we would like to explore factoring : to find out how large numbers we should take so that RSA becomes impossible to break. In early 1980s, people believed 100-digit numbers provided enough security, but now the computer can factor them efficiently. For example, in 1977, the inventors of RSA (Ron Rivest, Adi Shamir, and Len Adleman) posed the famous 129-digit RSA challenge and predicted it would take 40 quadrillion years to factor the challenge. However, it was factored in April 1994 in 8 months. In 1996, a 130-digit RSA challenge was factored using the general number field sieve. Nowadays many people use 155-digit numbers to protect their important data.

The progress in factoring results from faster hardware and better factorization methods. Currently the available factoring methods include rho factorization, factor base factorization, continued fraction factorization, quadratic sieve factorization, p-1 factorization, elliptic curve factorization, and number field sieve factorization. Since the basic theoretical research in cryptography is scarce in Taiwan, we dedicate ourselves to explore this area trying to design more efficient factoring algorithms. For application purpose, we will develop a factoring software for academic users in the country.

Keywords: RSA public key cryptosystem, Factoring, Quadratic sieve factorization, Elliptic curve factorization, Number field sieve factorization

二、緣由與目的

1980 年代，非對稱性的加密方法 RSA 演算法被提出並有成熟的應用加密軟體後，大數分解變成一件很重要的事。目前很多的密碼加密方法都是採用 RSA 演算法，其安全度是基於對大數作質因數分解的困難度。所有的正整數都可以用唯一的質因數乘積來表示，而且可以很容易證明這樣的質因數分解存在。給定二個以上的質數可以很容易的算出其乘積，但反之，若給定一個大的正整數卻很難將其質因數分解。

最直接的大數分解法就是試除法(The trial division method)，即對整數 n，用 2, 3, …, $\lfloor\sqrt{n}\rfloor$ 去試除，來分解大數。這個簡單的方法對 20 左右的數就要耗費很多時間。在四十年代電子計算機出現以前，儘管發明了許多大數分解的方法，但因需要手算，故即使十幾位數也需好幾天的時間，而對更大的數更是無能為力。隨著電子計算機的發明，人們開始利用這些歷史上留下來的方法並創造新的方法。到了七十年代隨著大數分解在密碼學上有了重大的價值，數論學家與計算機專家們已深入地研究這個問題，而且得到了許多有效的方法。目前著名的演算法有

1. Rho 分解法(The rho factorization method)
2. 因數基集分解法(The factor base factorization method)
3. 連分數分解法(The continued fraction factorization method)
4. 二次篩選分解法(The quadratic sieve factorization method)
5. p-1 分解法(The p-1 factorization method)
6. 橢圓曲線分解法(The elliptic curve factorization method)
7. 代數體篩選分解法(The number field factorization method)

根據影響其執行效率的因素，我們將其分成二大類，第一類的執行時間與分解之大數的因數所具有的特性有關，這類的演算法有試除法、Rho 分解法[11]、Pollard p-1 分解法[10]及橢圓曲線分解法[5]。而第二類的執行時間只跟大數本身

的特性有關，這類的演算法有連分數分解法[9]、二次篩選分解法[13]及代數體篩選分解法[6][1][7][12]。實際上這二類的演算法都非常的重要，假設要分解任一大數，若先前沒有任何關於此大數的因數的提示，則大部份都先使用第一類的方法作因數分解，若仍有無法分解出來的因數時，再使用第二類的方法作完全的因數分解。

本計畫主要是研究有關大數分解的理論和實際的應用，依照不同的考量因素去改良與設計出不同的演算法：這些研究需要有數論、現代代數、複雜度分析、以及演算法設計等方面的知識並加以整合。在理論方面，我們將詳細的研讀所有相關於大數分解及數論的文獻，包括前述的各種分解方法。再者，我們也會透過網際網路得知世界上最新最快有關於分解大數的問題和新的方法，以及它們在複雜度分析方面的結果；在應用方面，首先我們會測試目前現有的程式，並分析它們的優缺點。其次，我們會嘗試把前述的演算法以及我們所設計的演算法做實際上軟體的撰寫；初期我們希望能有一個網際網路的介面供大家分解因數，進而我們試著撰寫可供利用的 library 供有興趣的學者使用。

大數分解的進步主要歸功於硬體的速度越來越快及更多快速的分解演算法被提出。鑑於國內密碼學學術界對於基礎理論的研究並不多，我們將對這領域做深入的研究，設計出更有效率的大數分解演算法。在應用方面，我們也將發展一套大數分解的軟體，以供國內學界使用。

三、結果與討論

我們蒐集了到目前為止各種有關大數分解的理論演算法及實作方法，整理完成大數分解的基礎理論與實作技巧[2]，提供給國內密碼學學術界參考。

在實作方面，目前被認為對於 129 位數以下的大數，最有效率的大數分解方法是二次篩選分解法，而大於 130 位數以上的大數則以代數體篩選分解法的分解效率較好，基於硬體設備及時間等因素考量，我們以 100 位數左右的大數為目標，使用二次篩選分解法來分解大數。我們實作了雙質數多多項式二次篩選分解法[3]，並使用交通大學資訊工程學系英特爾實驗室的設備，設計分散式主從架構環境來完成 RSA 大數分解。根據我們的實作，成功的分解了 90 位數的大數。

據此，我們完成本計畫的目的，並在大數分解的領域上提供一個理論與實作並重的研究報告。

四、計畫成果自評

依上節所提之結果，我們達成了此計劃預期的目標。此計畫的研究結果除了為國內密碼學術界提供大數分解理論研究的基礎；另外透過實作的過程，發展國內相關的基礎技術，除了應用在大數分解的領域外，未來更可以運用在密碼學上基本的計算及安全強度的分析。成果極具有學術上的價值與貢獻，相當適合學術期刊上發表。

五、參考文獻

[1] J. P. Buhler, H. W. Lenstra, C. Pomerance, “Factoring Integers with the Number Field Sieve”, in A. K. Lenstra and H. W. Lenstra, jr. (eds), The Development of the Number Field Sieve, Lecture Notes in Mathematics v1554, Springer-Verlag, New York, 1993, pp. 50-94.

[2] J. H. Hung and R. J. Chen, Parallel RSA Factoring, Master thesis, the National Chiao Tung University, (2000).

[3] J. H. Hung, J. S. Hwu, and R. J. Chen, (2000) "Parallel RSA Factoring," Proceedings of the 10th National Conference on Information Security, Taiwan, pp. 51-61.

[4] B. Kurowski, “The multiple polynomial quadratic sieve: a platform-independent distributed implementation”, http://www.marlboro.edu/~kurowski/

[5] H. W. Lenstra, Jr., “Factoring integers with elliptic curves”, Annals of Math., v126(3), 1987, pp. 649-673.

[6] A. K. Lenstra, H. W. Lenstra, Jr., M. S. Manasse, J. M. Pollard, “The number field Sieve”, Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing, Baltimore, May 14-16, 1990, ACM, 1990, pp. 564-562.

[7] A. K. Lenstra, H. W. Lenstra, Jr., M. S. Manasse, J. M. Pollard, “The Factorization of the Ninth

Fermat Number", Math. Comp. v61, 1993, pp. 319-349.

[8] A. K. Lenstra, M. S. Manasse, "Factoring by electronic mail", Proc. Eurocrypt'89, LNCS v434, 1990, pp. 355-371

[9] M. A. Morrison, J. Brillhard, "A method of factoring and the factorization of ", Math. of Comp., v29, 1975, pp. 183-205.

[10] J. M. Pollard, "Theorems on factorization and primality testing", Proc. Camb. Phil. Soc., v76(2), 1974, pp.521-528.

[11] J. M. Pollard, "A Monte Carlo method for factorization", BIT. v15(3), 1975, pp. 331-334.

[12] J. M. Pollard, "Factoring with Cubic Integers", in A. K. Lenstra and H. W. Lenstra, jr. (eds), The Development of the Number Field Sieve, Lecture Notes in Mathematics v1554, Springer-Verlag, New York, 1993, pp. 4-10.

[13] C. Pomerance, "The quadratic sieve factoring algorithm", Advances in Cryptology, Proceeding of Eurocryp'84, LNCS v209, 1985, pp. 169-182.

[14] C. Pomerance, J. W. Smith, Randy Tuler, "A pipeline architecture for factoring large integers with the quadratic sieve algorithm", SIAM J. Comp., v17, 1988, pp. 387-403.