EFFICIENT FACTORING BASED ON PARTIAL INFORMATION

Ronald L. Rivest* and Adi Shamir**

*MIT Laboratory for Computer Science
Cambridge, Mass. 02139, U.S.A.

**Applied Math. Dept., The Weizmann Institute of Science
Rehovot 76100, Israel

Many recently proposed cryptosystems are based on the assumption that factoring large composite integers is computationally difficult. In this paper we examine this assumption when the cryptanalyst has "side information" available.

Let N be the product of two large primes P and Q, where N is n bits in length, and P, P are each $n/2$ bits in length. Given N, it is possible to compute P and Q in time approximately

$$L(N) = \exp(\mathrm{sgrt}(\ln(N) \ln\ln(N))) \quad [1]$$

using, for example, the recent algorithm of Lenstra.

In cryptographic applications, however, the cryptanalyst may have available additional "side information" above and beyond the number N itself. In practice, one of the parties typically knows P and Q already, and uses these factors explicitly during his cryptographic computations. The results of these computations may become known to the cryptanalyst, who thereby may find himself at an advantage compared to a pure factoring situation.

For example, the cryptanalyst might become privy to:

(1) the procedure that generated P and Q (but not the random inputs to that procedure).

(2) the lengths of P and Q.

(3) a square root of 2, modulo N.

(4) the RSA signature of a message M using modulus N corresponding to a public RSA exponent of 3.

(5) the least-significant $n/4$ bits of P.

The point to be understood is that in practice additional side information may become available to the cryptanalyst, for one of the following reasons:

- loss of the equipment that generated P and Q.
- explicit release of partial side information as part of a protocol (e.g., "exchange of secrets" [B183]).
- routine usage of P, Q to decrypt mail, sign messages, etc.
- poor physical or electrical security by crypto equipment that uses and guards P and Q.

We formalize this notion, in a worst-case manner, as follows. Suppose that the cryptanalyst is allowed to ask a certain number k of arbitrary "Yes/No" questions at the beginning. He is given the answers to these questions before he attempts to factor N. (We do not care about the difficulty of answering these questions -- the answers are supplied free of charge to the cryptanalyst.) To be precise, we assume he is given the answer to question i before he poses question $i + 1$.

As we increase k, the cryptanalyst's task becomes easier and easier. For example, with $k = n/2$ his task is trivial: he asks for the binary representation of P. We ask our fundamental question: for what values of k (as a function of N) can the cryptanalyst factor N in polynomial time? Our main result is the following:

Theorem. The cryptanalyst can factor $N = P\cdot Q$ (where P and Q are $n/2$-bit numbers, and N is an n-bit number in time polynomial in n, if he is first given the answers to $n/3 + O(1)$ "Yes/No" questions about N for free.

This is not a dramatic improvement over the obvious $n/2$ result mentioned above. However, the proof is not trivial, and we do not know how to improve on this result. We conjecture that $O(n^{\varepsilon})$ questions suffice, for some $\varepsilon < 1$.

Proof (sketch): Suppose the cryptanalyst asks for the top $k = n/3$ bits of the factor P. He can then represent P in the form

$$P = P_1\cdot 2^m + P_o \quad [2]$$

where $m = (n/2) - k = n/6$,

$$0 \le P_1 \le 2^k \quad [3]$$

$$0 \le P_o \le 2^m \quad [4]$$

P_1 is known, and P_o is unknown. The factor Q can be represented similarly:

$$Q = Q_1\cdot 2^m + Q_o \quad [5]$$

where

$$0 \le Q_1 \le 2^k \text{ , and} \quad [6]$$

$$0 \le Q_o \le 2^m. \quad [7]$$

Since N and P_1 are known, Q_1 can be easily computed. (We know N and P to at least k bits of precision, so we know their quotient to k bits of precision.) The unknowns to be solved for are P_o and Q_o.

Compute

$X = N - P_1Q_1 2^{2m}$, [8]

$A = P_1 \cdot 2^m$, and [9]

$B = Q_1 \cdot 2^m$. [10]

Then we have the equation

$X = A \cdot P_o + B \cdot Q_o + P_oQ_o$ [11]

to solve for P_o and Q_o. When k is large, m is small, and the product P_oQ_o (of length $2m$) is also small. We can thus attempt to solve [11] by trying to find a linear combination of A and B that closely approximates X. (We treat the term P_oQ_o as similar to the "approximation error".) We set this up as a two-dimensional integer programming problem:

<u>Minimize</u>: $Z = X - A \cdot P_o - BQ_o$ [12]

<u>Subject to</u>: $0 \leq P_o \leq 2^m$ [13]

$0 \leq Q_o \leq 2^m$ [14]

We note that X is approximately $n - k$ bits in length. We use a heuristic argument here that for each degree of freedom (bit) we can set in P_o or Q_o, we can reduce the length of Z by one bit. Since we have

$|P_o| + |Q_o| = 2m$ [15]

we expect that Z will be $|X| - 2m = n - 2k = n/3$ bits in length; our "approximation error" is about $n/3$ bits in length. We note that P_oQ_o also has length $2m = n/3$, so that the "modelling error" we introduced by moving from the nonlinear equation [11] to the linear approximation [12] will also be about $n/3$ bits in length. We can thus expect the solution to [12]-[14] to be a solution for [11] as well. We note that [12]-[14] can be solved in polynomial time using Lenstra's algorithm for integer programming in a fixed number of dimensions. [Le81]

The preceding proof sketch is not a rigorous argument, but can be made so (although the number of questions may need to be increased by $O(1)$ to handle some details about the precision).

A similar argument can be made to show that the cryptanalyst can factor N using the <u>low</u>-order k bits of N rather than the high-order k bits. ■

Open Problems

Prove or disprove that $\Omega(n)$ questions are necessary in the theorem, if the cryptanalyst may only ask for bits in the binary representation of P.

Prove or disprove that $\Omega(n)$ questions are necessary in general.

Acknowledgment

This research was supported in part by NSF grant MCS 8006938.

References

[Bl83] Blum, Manuel. "How to exchange secrets," Proc. 15th Annual ACM STOC Conference (1983), 440-447.

[Le81] Lenstra, H. W., Jr. "Integer programming in a fixed number of variables," Report 81-03, Mathematisch Institut, Universitat ban Amsterdam (1981).