A chosen text attack on the RSA cryptosystem and some discrete logarithm schemes

Y. Desmedt

Aangesteld Navorser NFWO
Katholieke Universiteit Leuven
Laboratorium ESAT
B-3030 Heverlee, Belgium

A. M. Odlyzko

AT&T Bell Laboratories
Murray Hill, NJ 07974, USA

ABSTRACT

A new attack on the RSA cryptosystem is presented. This attack assumes less than previous chosen ciphertext attacks, since the cryptanalyst has to obtain the plaintext versions of some carefully chosen ciphertexts only once, and can then proceed to decrypt further ciphertexts without further recourse to the authorized user's decrypting facility. This attack is considerably more efficient than the best algorithms that are known for factoring the public modulus. The same idea can also be used to develop an attack on the three-pass system of transmitting information using exponentiation in a finite field.

1. Introduction

The RSA cryptosystem [13] is perhaps the most famous public key cryptosystem and, together with the Diffie-Hellman key exchange scheme [6], is one of the most important public key systems. It is often thought that breaking the RSA system is as hard as factoring the public modulus n used in the system, but this has never been proved. The attack by Simmons and Norris [14] involving repeated encryptions has been shown to be unlikely to succeed if the primes dividing the modulus n are chosen carefully [12]. On the other hand, it has been pointed out that there are ways to employ the RSA system that can be cryptanalyzed without factoring n. For example, Knuth's proposal to use a small encryption exponent (to speed up operation) was shown to be unsafe when the same message is being sent to several destinations simultaneously [1] (see also [4; pp. 57-58] and [7]).

Another attack on some particular ways to employ the RSA system is due to Davida [3] (see also [5]), and our result is similar to it and can be considered as a generalization of it. In that attack, suppose that n is the public modulus of user A and E the public encryption exponent. Suppose that the cryptanalyst intercepts the ciphertext $c \equiv m^E \pmod{n}$, and wishes to recover the plaintext m. He chooses a random integer x and forms

$$c' \equiv c\, x^E \pmod{n} \ .$$

If he can now get user A to decrypt c' (for example, if A uses the pair (E, n) for signatures and is willing to sign challenge messages, or else if A discards decrypted messages that appear meaningless, and the cryptanalyst can get access to these discards), then he obtains

$$(c')^D \equiv c^D\, x \equiv m\, x \pmod{n} \ ,$$

and can recover m. Thus the cryptanalyst can decipher any single ciphertext at the cost of using A's decryption mechanism once.

In this note we present a slightly different attack. As in the Davida-style attacks, it requires that user A decrypt a certain number of chosen ciphertexts and make the decrypted versions available to the cryptanalyst. (We are assuming here for simplicity that user A's public key (E, n) is used to send private information to him. A similar procedure applies if these keys are used for authentication, in which case a chosen plaintext attack is used.) The differences between our scheme and Davida's is that once the decrypted plaintexts of the chosen ciphertexts are obtained, no further decryptions by A are needed to read individual messages. More precisely, let $L = L(n)$ denote any quantity that satisfies [11]

$$L = \exp((1+o(1))\ ((\log n)\ (\log\log n))^{1/2}) \quad \text{as} \quad n \to \infty \ .$$

Then, if the cryptanalyst succeeds in obtaining decrypted versions of $L^{1/2}$ chosen ciphertexts, he can decipher any specific ciphertext in $L^{1/2}$ bit operations on his own computer. (It is possible to decrease the number of operations required to decrypt individual ciphertexts at the cost of increasing the number of uses of A's decryption facility, and vice versa.) The importance of this result is that the best currently known algorithms for factoring integers of the same size as n require L bit operations [2,8,9]. (The memory required is $L^{1/2}$ bits for our attack, although it can be a very slow memory, such as a tape.

Some factoring algorithms require negligible memory, while others also require $L^{1/2}$.) Therefore our attack on the RSA cryptosystem, although based on very special assumptions, appears to be the most general one that has been proposed so far and is substantially faster than factoring *n*.

Our basic assumptions are not completely unrealistic. It is easy to imagine situations where decryptions that are not intelligible might not be classified and would be thrown away (either by poorly trained secretaries or by software programs) in a form where they could be intercepted by the cryptanalyst. Also, one can imagine situations in which repairmen servicing either the decryption ‘‘black box’’ or nearby pieces of equipment could obtain the desired decryptions. Another scenario where our attack might apply arises when a whole group of users uses the same decryption key, which is not accessible to them. By using the decryption ‘‘black box,’’ any one user can accumulate data required by our attack which would let him break the scheme even after he was denied access to the ‘‘black box’’ if the key was not changed. (The reader could remark that group members can sign all the messages they wish and use them later on, so this method is not necessary. However, not all fraudulent messages that the forgers might wish to use can be predicted beforehand.)

Our attack can also be applied to the well-known ‘‘three-pass’’ system for transmitting information using exponentiation in a finite field [8; pp. 345-346]. In it, user A wishes to send message *m* to user B, where *m* is regarded as belonging to some fixed and known finite field $GF(q)$. User A selects an integer *a* such that $(a, q-1) = 1$ and transmits m^a to B. User B then selects an integer *b* with $(b, q-1) = 1$ and sends m^{ab} to A. Next, A computes a' such that $aa' \equiv 1 \pmod{q-1}$, and sends $m^{aba'} = m^b$ to B, who now obtains *m* from $m = m^{bb'}$, where $bb' \equiv 1 \pmod{q-1}$. Should user B always use the same integer *b*, our attack could again be applied. In it the cryptanalyst would send $L^{1/2}$ messages *u* to B, (where $L = \exp((1 + o(1))\ ((\log q)\ (\log \log q))^{1/2})$ this time), would receive u^b for each one of them, and would then be able to decipher any messages that might be sent to B using this protocol in $L^{1/2}$ bit operations on his own computer. The same kind of attack applies if user A always uses the same integer *a*. (This kind of attack could also be applied to the basic Diffie-Hellman key distribution scheme, but in that context is less realistic.)

The basic lesson of our attack is that one has to be very careful in using the RSA cryptosystem and discrete exponentiation schemes to keep them secure. If the attacks that are outlined above have to be guarded against, moduli somewhat larger than those currently being recommended are likely to be required. However, as we note at the end of the next section, in practical situations the necessary increase in the modulus size is likely to be quite small.

2. The attack

Our attack is a modification of an algorithm used for computing discrete logarithms in fields $GF(p)$ for p a prime [2]. Many of the number theoretic estimates that we utilize can be found there and in [11].

Let $\alpha > 0$ be fixed, and let $k = \lfloor n^{1/2} \rfloor$. In the first stage we utilize user A's decrypting facility to obtain $x^D \pmod{n}$ for all $x \in S = S_1 \cup S_2$, where

$$S_1 = \{p \colon p \le L^{\alpha} , \; p \text{ a prime}\} , \tag{2.1}$$

$$S_2 = \{k+1, k+2 , \ldots , k + \lfloor L^{\alpha} \rfloor\} .$$

In order to avoid detection by any simple screening program, we choose a random y_x for each x and obtain $\left[x\, y_x^E \right]^D \equiv x^D \, y_x \pmod{n}$ from the decrypting algorithm, which then allows us to obtain $x^D \pmod{n}$.

Once we have obtained $x^D \pmod{n}$ for all $x \in S$, we can proceed to decrypt individual ciphertexts c. (At this stage we will not need to use the decrypting facility any more.) The basic idea is to find a representation

$$c \equiv y^E \prod_{x \in S} x^{a_x} \pmod{n} \tag{2.2}$$

for some integers a_x and y, since then

$$c^D \equiv y \prod_{x \in S} \left[x^D \right]^{a_x} \pmod{n} ,$$

where y and all the x^D are known to the cryptanalyst as explained above.

To obtain the representation (2.2), we proceed in two stages. In the first stage we find a y and primes $q_i \le L^{2\alpha}$ such that

$$c \equiv y^E \prod_{i=1}^{h} q_i \pmod{n} . \tag{2.3}$$

To obtain the representation (2.3), we choose a random y, compute

$$b \equiv cy^{-E} \pmod{n} , \ 1 \le b \le n ,$$

and check whether b factors into primes $q \le L^{2\alpha}$. We expect to test approximately $L^{1/(4\alpha)}$ values of y before a factorization of c of the form (2.2) is found [2,11], and for each y, it takes $L^{o(1)}$ bit operations to test whether such a factorization exists by using Lenstra's elliptic curve factorization algorithm [9]. Therefore this stage is expected to take time $L^{1/(4\alpha) + o(1)} = L^{1/(4\alpha)}$.

Once a factorization of the form (2.3) is obtained, we proceed to the second stage, in which we represent each of the at most $O(\log n) = L^{o(1)}$ primes $q = q_i \le L^{2\alpha}$ in the form

$$q \equiv \prod_{x \,\varepsilon\, S} x^{u_x} \pmod{n} , \tag{2.4}$$

where only $O(\log n)$ of the u_x are non-zero (possibly negative). Once such a representation is obtained for each of the q's, we quickly obtain (2.2).

To see how to represent a prime $q \le L^{2\alpha}$ in the form (2.4), let

$$m = \left\lfloor n^{1/2} q^{-1} \right\rfloor$$

and determine those integers among

$$m+1, m+2 , \ldots , m + \left\lfloor L^{\beta} \right\rfloor \tag{2.5}$$

that are divisible solely by primes $p \le L^{\alpha}$. There will be $L^{\beta - 1/(4\alpha)}$ such integers, and finding them will take L^{β} bit operations if we employ the Lenstra algorithm, for example.

We next consider two cases. If $\alpha \ge 1/2$, we take $\beta = 1/(4\alpha) + \delta$ for any $\delta > 0$. We then have L^{δ} integers $m+j$, $1 \le j \le L^{\beta}$, all of whose prime factors are $\le L^{\alpha}$. For each such integer and any

$i,\ 1 \le i \le L^{1/(4\alpha)}$, (note that $1/(4\alpha) \le \alpha$)

$$q(m+j)\ (k+i) \equiv t \pmod{n}\ , \tag{2.6}$$

where $t \le n^{1/2 + o(1)}$ and k was defined at the beginning of this section. Therefore, if the t's factor like random integers of the same size, we will find L^{δ} t's that factor into primes $\le L^{\alpha}$, and any single one will give us a factorization of the form (2.4), which gives the desired result. Since testing of each t takes $L^{o(1)}$ bit operations, this stage requires $L^{\beta + o(1)}$ bit operations as $n \to \infty$, and since this holds for all $\delta > 0$, we conclude that for $\alpha \ge 1/2$, this stage can be carried out in $L^{1/(4\alpha)}$ bit operations.

It remains to consider the case $\alpha < 1/2$. Here we take $\beta = 1/(2\alpha) - \alpha + \delta$. We expect to find $L^{\beta - 1/(4\alpha)} = L^{1/(4\alpha) - \alpha + \delta}$ values of $m+j,\ 1 \le j \le L^{\beta}$, which factor into primes $\le L^{\alpha}$, and it takes $L^{\beta + o(1)} = L^{\beta}$ bit operations to find them. For each one, and for $1 \le i \le L^{\alpha}$, we test whether the t defined by (2.6) is composed of primes $\le L^{\alpha}$. We expect to find L^{δ} of them. Letting $\delta \to 0$, we discover that this case takes $L^{1/(2\alpha) - \alpha}$ bit operations.

We thus conclude that if the cryptanalyst can obtain decryptions of L^{α} chosen ciphertexts he will be able to decrypt any individual ciphertext in $L^{1/(4\alpha)}$ bit operations for $\alpha \ge 1/2$, and in $L^{1/(2\alpha) - \alpha}$ bit operations for $0 < \alpha \le 1/2$. For $\alpha = 1/2$, both stages require $L^{1/2}$ steps. Since the modulus n can be factored in L bit operations, our attack has an asymptotically smaller running time precisely for

$$\frac{-1 + \sqrt{3}}{2} = 0.366... < \alpha < 1\ .$$

In practice, our $L^{1/2}$ attack does not offer too much of a speed improvement over the L attacks that factor the modulus. The main reason is that for moduli of practical size (500 to 1000 binary bits) $L^{1/2}$ is quite small, and there is no known way to test whether an integer of size about n (or even $n^{1/2}$) is divisible only by primes $\le L^{1/2}$ that is much more efficient than trial division. The Lenstra algorithm can be avoided by utilizing somewhat more involved, although probably more efficient methods (cf. [2]), but even they probably would not offer too much of a speedup.

References

1. M. Blum, A potential danger with low-exponent modular encryption schemes, to be published.

2. D. Coppersmith, A. M. Odlyzko, and R. Schroeppel, Discrete logarithms in $GF(p)$, *Algorithmica*, to appear.

3. G. Davida, Chosen signature cryptanalysis of the RSA (MIT) public key cryptosystem, Tech. Rept. TR-CS-82-2, Dept. of Electrical Engineering and Computer Science, Univ. of Wisconsin, Milwaukee, Wisconsin, Oct. 1982.

4. R. A. DeMillo, G. I. Davida, D. P. Dobkin, M. A. Harrison, and R. J. Lipton, *Applied Cryptology, Cryptographic Protocols, and Computer Security Models*, Proc. Symp. Appl. Math. #29, Am. Math. Soc. 1983.

5. D. E. Denning, Digital signatures with RSA and other public-key cryptosystems, *Comm. ACM 27* (1984), 388-392.

6. W. Diffie and M. E. Hellman, New directions in cryptography, *IEEE Trans. Inform. Theory*, *IT-22* (1976), 644-654.

7. J. Hastad, On using RSA with low exponent in a public key network, to be published.

8. A. G. Konheim, *Cryptography: A Primer*, Wiley, 1981.

9. H. W. Lenstra, Jr., manuscript in preparation.

10. A. M. Odlyzko, Discrete logarithms in finite fields and their cryptographic significance, *Proc. Eurocrypt ’84*, to appear.

11. C. Pomerance, Analysis and comparison of some integer factoring algorithms, pp. 89-139 in *Computational Methods in Number Theory: Part 1*, H. W. Lenstra, Jr., and R. Tijdeman, eds., Math. Centre Tract 154, Math. Centre Amsterdam, 1982.

12. R. L. Rivest, Remarks on a proposed cryptanalytic attack on the M.I.T. public-key cryptosystem, *Cryptologia 2* (1978), 62-65.

13. R. L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, *Comm. ACM 21* (1978), 120-126.

14. G. T. Simmons and J. N. Norris, Preliminary comments on the M.I.T. public-key cryptosystem, *Cryptologia 1* (1977), 406-414.