On Shamir's Unbalanced RSA cryptosystem

Dipankar Gupta *

24.1.96

Abstract

We examine Adi Shamir's proposal (presented at EUROCRYPT '95 and in [2]) to enhance RSA's security without an undue increase in the cost of computation. We point out a case of protocol failure and suggest a modification to the decryption stage that protects against this failure.

1 Unbalanced RSA

A variant of the usual RSA cryptosystem has been proposed by Adi Shamir in [2] to address the issue of increasing it's security while keeping the cost of encryption and decryption competitive. Shamir proposes to use a modulus $n = pq$ with unbalanced factors (eg. $\log q \sim 10 \log p$) to provide greater security while allowing for a minor increase in computation effort during encryption. It is based on the following observations:

- Techniques for factoring integers that are based on algebraic number fields have complexity determined by the size of the modulus n. This is in contrast with those based on elliptic curves that have complexity determined by the size of the smaller factor p. We use the following notation for the asymptotic complexity:

$$L[c, a](x) = (\exp((c + o(1)) \cdot (\log x)^a (\log \log x)^{1-a})).$$

  Algorithms of the first type have complexity $\mathcal{O}(L[c, \frac{1}{3}](n))$ whereas those of the second type have complexity $\mathcal{O}(L[c^*, \frac{1}{2}](p))$. Shamir suggests that the class of algorithms that are of interest to factoring RSA moduli will be the first type on account of the lower $a = \frac{1}{3}$. To provide additional security against this class of algorithms, the size of n can be increased to a much greater extent, assuming that p is already large enough to make an ECM attack infeasible. He therefore proposes to increase $\log p$ by a factor of 2 whilst increasing $\log n$ by a factor of 10. This means that q is much larger than p.

- A balanced RSA cryptosystem with larger factors would increase the computation load significantly. Modular exponentiation of an element x of $\mathbb{Z}/n\mathbb{Z}$ by exponent e takes time $\mathcal{O}(\log^2 n \log |e|)$ (see [1, p. 8]), and thus

*Hewlett–Packard Laboratories, Bristol BS12 6QZ UK; dg@hplb.hpl.hp.com

would be slowed down by a factor of 100 for balanced RSA. Shamir proposes that for certain common uses of RSA cryptosystems like sharing session keys, if the message m (equiv. the element x) is smaller than p, then operation $c = m^e \bmod n$ can be computed in time comparable to that for the usual sized RSA moduli. This is because in addition to e being small, the $(\log^2 n)$ term in the estimate is related to the complexity of squaring an arbitrary element. The precondition $m < p$ ensures that the results of a majority of the multiplications would be smaller than n. Exponents less than 10 are dangerous because they don't provide any wraparound. Shamir suggests that the value of the encryption exponent e be between 20 and 100. Shamir now suggests that the decryption stage $c^d \bmod n$ can be reduced to $r^{d'} \bmod p$ where $d' \equiv d \bmod \phi(p)$ and $r \equiv c \bmod p$, which is of comparable complexity as (balanced) RSA.

2 Comments

We note the following issues with the proposal:

- It assumes that the message m is smaller than the smallest factor p, which may be reasonable when public key cryptography is being used to exchange session keys for bulk encryption, for example. We denote by M the maximum permissible value of m; this is publically known.
- The speedup suggested in the decryption stage requires the factor p. In hardware (balanced) RSA cryptosystem, only d need be on the chip to perform the decryption (in addition to the publically available n), and may be relevant to hardware implementations.

3 Protocol Failure

The main purpose of this note is to remark that the cryptosystem is insecure if used in certain common ways, and hence can lead to protocol failure if this is not taken into account. If Alice uses Bob's public key n to share a secret with him, and can get access to his decrypted plaintext, then she can recover the factorization of n. This happens if she encrypts a message m which is bigger than p and sends the ciphertext $m^e \bmod n$ to Bob. Bob decrypts the ciphertext to get $\widetilde{m}$. As $m > p$, $\widetilde{m} \neq m$. Now, if Alice can access $\widetilde{m}$, she knows that $p \mid (m - \widetilde{m})$. We also know (using Chinese Remainder theorem) that $q \nmid (m - \widetilde{m})$ under the constraints $0 < m, \widetilde{m} < n$ and $m \neq \widetilde{m}$. Alice can therefore recover p by computing $\gcd(m - \widetilde{m}, n)$. We therefore conclude that this cryptosystem should not be used in protocols where Alice can verify the decrypted message $\widetilde{m}$, for instance in challenge-response situations [4, p. 204] to verify the identity of Bob. Shamir has pointed out [3] that this is similar to the known ciphertext attack on Rabin's variant. He notes that it may be resolved by adding redundancy into the encrypted message, and to reject the decrypted ciphertext if it does not have this redundancy. This attack can also be countered by adding a verification step in the decryption stage where c is compared with $\widetilde{m}^e \bmod n$ or equivalently $\widetilde{m}^e \bmod q$, and rejected if they are not identical.

However, if this (or any) verification step has been added which fails if $m > p$, Alice could find the prime p in $o(\log n)$ protocol attempts using known plaintexts. She does a binary search for p, using M and $2^{\lfloor \frac{1}{2}(\log n) \rfloor + 1} - 1$ as the initial lower and upper bounds, and narrowing the bounds each time a protocol request fails. This attack is independent of whether the decryption result is accessible to the attacker as only one bit of information (whether the protocol ran to completion) is required per request. Thus, the verification step should be strengthened to include the re-encryption step and verify that $m' < M$. In addition, the key generation phase should ensure the strict inequality: $\lfloor \log p \rfloor > \lfloor \log M \rfloor$.

References

[1] H. Cohen, *A Course in Computational Algebraic Number Theory*, vol. 138 of Graduate Texts in Mathematics, Springer, 1994.

[2] A. Shamir, *RSA for paranoids*, CryptoBytes, 1 (1995), pp. 1,3–4.

[3] ——, *Personal communication.* (electronic mail), 1996.

[4] D. R. Stinson, *Cryptography: Theory and Practice*, CRC Press, 1995.