Cryptanalysis of a Zero-Knowledge Identification Protocol of Eurocrypt '95

Jean-Sébastien Coron and David Naccache

Gemplus Card International
34 rue Guynemer, 92447 Issy-les-Moulineaux, France
{jean-sebastien.coron, david.naccache}@gemplus.com

Abstract. We present a cryptanalysis of a zero-knowledge identification protocol introduced by Naccache *et al.* at Eurocrypt '95. Our cryptanalysis enables a polynomial-time attacker to pass the identification protocol with probability one, without knowing the private key.

Key Words: Zero-knowledge, Fiat-Shamir Identification Protocol.

1 Introduction

An identification protocol enables a verifier to check that a prover knows the private key corresponding to a public key associated to its identity. A protocol is zero-knowledge when the only additional information obtained by the verifier is that the prover knows the corresponding private key [2]. A famous zero-knowledge identification protocol is Fiat-Shamir's protocol [1], which is provably secure assuming that factoring is hard. The protocol requires performing multiplications modulo an RSA modulus.

A space-efficient variant of the Fiat-Shamir identification protocol was introduced by Naccache [3] and by Shamir [5] at Eurocrypt' 94. This variant requires only a few bytes of RAM, even for an RSA modulus of several thousands bits, and is provably as secure as the original Fiat-Shamir protocol. This variant is particularly interesting when the prover is implemented in a smart-card, in which the amount of RAM is very limited.

However, the time complexity of the previous variant is still quadratic in the modulus size, and its implementation on a low-cost smart-card is likely to be inefficient. At Eurocrypt '95, Naccache *et al.* introduced another Fiat-Shamir variant [4]. It uses the same idea for reducing the space-complexity, but the prover's time complexity is now quasi-linear in the modulus size (instead of being quadratic). As shown in [4], the new identification protocol can be executed on a low-cost smart-card in less than a second.

In this paper, we describe a cryptanalysis of one of [4]'s time-efficient variants. Our cryptanalysis enables a polynomial-time attacker to pass the identification protocol with probability one, without knowing the private key. We would like to stress that the basic quasi-linear time protocol introduced by [4] remains secure, since it is in fact equivalent to standard Fiat-Shamir and hence to factoring.

2 The Fiat-Shamir Protocol

We briefly recall Fiat-Shamir's identification protocol [1]. The objective of the prover is to identify itself to any verifier, by proving knowledge of a secret s corresponding to a public value v, which is associated to its identity. The protocol is zero-knowledge in that it does not reveal any additional information about s to the verifier. The security relies on the hardness of factoring an RSA modulus.

Key generation: The authority generates a k-bit RSA modulus $n = p \cdot q$, and an integer v which is a function of the identity of the prover. Using the factorization of n, it computes a square root s of v modulo n, *i.e.* $v = s^2 \mod n$. The authority publishes (n, v) and sends s to the prover.

Identification protocol:

1. The prover generates a random $x \leftarrow Z_n$, and sends $z = x^2 \mod n$ to the verifier.
2. The verifier sends a random bit b to the prover.
3. If $b = 0$, the prover sends $y = x$ to the verifier, otherwise it sends $y = x \cdot s \mod n$.
4. The verifier checks that $y^2 = z \cdot v^b \mod n$.
5. Steps 1-4 are repeated several time to reduce the cheating probability.

3 The Space-Efficient Variant of Fiat-Shamir's Protocol

Fiat-Shamir's protocol requires to perform multiplications modulo an RSA modulus n. It has a quadratic time and linear space complexity. Therefore, the original protocol could not be implemented on low-cost smart-cards, which in 1994 contained about 40 bytes of random access memory (RAM). Naccache [3] and Shamir [5] introduced a space-efficient variant which requires only a few bytes of RAM, even for an RSA modulus of several thousands bits, and which is provably as secure as the original Fiat-Shamir protocol.

The idea is the following: assume that the prover is required to compute $z = x \cdot y \mod n$, where x and y are two large numbers which are already stored in the smart-card (*e.g.*, in its EEPROM[1]), or whose bytes can be generated on the fly. Then instead of computing $z = x \cdot y \mod n$, the prover computes

$$z' = x \cdot y + r \cdot n$$

for a random r uniformly distributed in $[0, B]$, for a fixed bound B. The verifier can recover $x \cdot y \mod n$ by reducing z' modulo n. Moreover, when computing z', the prover does not need to store the intermediate result in RAM. Instead, the successive bytes of z' can be sent out of the card as soon as they are generated. Therefore, a smart-card implementation of the prover needs only a few bytes of RAM (see [5] or [3] for more details).

As shown in [5], if B is sufficiently large, there is no loss of security in sending z' instead of z. Namely, from z one can generate $z'' = z + u \cdot n$ where u is a random integer in $[0, B]$. Letting $z = x \cdot y - \omega \cdot n$, we have:

$$z'' = x \cdot y + (u - \omega) \cdot n$$

Then, the statistical distance between the distributions induced by z' and z'' is equal to the statistical distance between the uniform distribution in $[0, B]$ and the uniform distribution in $[-\omega, B - \omega]$, which is equal to ω/B. Then, assuming that x and y are both in $[0, n]$, this gives $\omega \in [0, n]$, and the previous statistical distance is lesser than n/B. Therefore, by taking a B much larger than n (for example, $B = 2^{k+80}$, where k is the bit-size of n), the two distributions are statistically indistinguishable, and any attack against the protocol using z' would be as successful against the protocol using z.

The identification protocol is then modified as follows:

Space-efficient Fiat-Shamir identification protocol:

1. The prover generates a random $x \leftarrow Z_n$ and a random $r \in [0, B]$, and sends $z = x^2 + r \cdot n$ to the verifier.
2. The verifier sends a random bit b to the prover.
3. If $b = 0$, the prover sends $y = x$ to the verifier, otherwise it sends $y = x \cdot s + t \cdot n$ for a random $t \in [0, B]$.

[1] The smart-card EEPROM is a re-writable memory, but the operation of writing is about one thousand time slower than writing into RAM, and can not be used to store fast-changing intermediate data during the execution of an algorithm.

4. The verifier checks that $y^2 = z \cdot v^b \mod n$.
5. Steps 1-4 are repeated several time to reduce the cheating probability.

4 The Time-Efficient Variant of Fiat-Shamir's protocol

The time complexity of the previous variant is still quadratic in the modulus size, and its implementation on a low-cost smart-card is likely to be inefficient. At Eurocrypt '95, Naccache *et al.* introduced yet another Fiat-Shamir variant [4]. It uses the same idea as Shamir's variant for reducing the space-complexity, but the prover's time complexity is now quasi-linear in the modulus size (instead of being quadratic). As shown in [4], the identification protocol can then be executed on a low-cost smart-card in less than a second.

The technique consists in representing the integers modulo a set of ℓ small primes p_i (usually, one takes the first ℓ primes). This is called the Residue Number System (RNS) representation. Letting $\Pi = \prod_{i=1}^{\ell} p_i$, by virtue of the Chinese Remainder Theorem, any integer $0 \leq x < \Pi$ is uniquely represented by the vector:

$$(x \mod p_1, \ldots, x \mod p_\ell)$$

The advantage of this representation is that multiplication is of quasi-linear complexity (instead of quadratic complexity): if x and y are represented by the vectors $(x_1, \ldots, x_\ell)$ and $(y_1, \ldots, y_\ell)$, then the product $z = x \cdot y$ is represented by:

$$(x_1 \cdot y_1 \mod p_1, \ldots, x_\ell \cdot y_\ell \mod p_\ell)$$

The size ℓ of the RNS representation is determined so that all integers used in the protocol are strictly smaller than Π; the bijection between an integer and its modular representation is then guaranteed by the Chinese Remainder Theorem. The time-efficient variant of the Fiat-Shamir protocol is the following:

Time-efficient variant of the Fiat-Shamir protocol:

1. The prover generates a random $x \in [0, n]$ and a random $r \in [0, B]$, and sends $z = x^2 + r \cdot n$ to the verifier. The integers x, r and z are represented in RNS.
2. The verifier sends a random bit b to the prover.
3. If $b = 0$, the prover sends $y = x$ to the verifier, otherwise it sends $y = x \cdot s + t \cdot n$ for a random $t \in [0, B]$. The integers x, s and t are represented in RNS.

4. The verifier checks that $y^2 = z \cdot v^b \mod n$.
5. Steps 1-4 are repeated several time to reduce the cheating probability.

The only difference between this time-efficient variant and Shamir's space-efficient variant is that integers are represented in RNS. Therefore, from a security standpoint, those variants are strictly equivalent.

However, another time-efficient variant is introduced in [4], whose goal is to increase the efficiency of the verifier. The goal of this second variant is to enable the verifier to check the prover's answer in linear time when $b = 0$. In this variant, when $b = 0$, the prover also reveals r, which enables the verifier to check that $z = x^2 + r \cdot n$ by performing the computation in the RNS representation (the equality $z = x^2 + r \cdot n$ is checked modulo each of the primes p_i), which takes quasi-linear time instead of quadratic time. More precisely, this variant is the following:

Second Time-efficient Variant of the Fiat-Shamir Protocol:

1. The prover generates a random $x \in [0, n]$ and a random $r \in [0, B]$, and sends $z = x^2 + r \cdot n$ to the verifier. The integers x, r and z are represented in RNS.
2. The verifier sends a random bit b to the prover.
3. If $b = 0$, the prover sends x and r to the verifier, in RNS representation. If $b = 1$, the prover sends $y = x \cdot s + t \cdot n$ for a random $t \in [0, B]$, where y is represented in RNS.
4. If $b = 0$, the verifier checks that $z = x^2 + r \cdot n$. The test is performed in the RNS representation. If $b = 1$, the verifier checks that $y^2 = z \cdot v \mod n$.
5. Steps 1-4 are repeated several time to reduce the cheating probability.

This second time-efficient variant is more efficient for the verifier, because when $b = 0$, the check at step 3 is performed in RNS representation, which is of quasi-linear complexity instead of quadratic complexity. Therefore, the time-complexity of this second time-efficient variant is expected to be divided by a factor of approximately two.

5 Cryptanalysis of the Second Time-Efficient Variant of Eurocrypt '95

We show that the second time-efficient variant is insecure. We describe an attacker $\mathcal{A}$ that passes the identification protocol with probability one, without knowing the private key s.

The key observation is the following: since for $b = 0$, the verifier checks that $z = x^2 + r \cdot n$ in the RNS representation, the equality checked by the verifier is actually:

$$z = x^2 + r \cdot n \mod \Pi \tag{1}$$

Since the attacker can choose $x, r \in [0, \Pi]$ instead of $x \in [0, n]$ and $r \in [0, B]$, we may have $x^2 + r \cdot n > \Pi$, and therefore equation (1) does not necessarily imply that $z = x^2 + r \cdot n$ holds over the integers (or equivalently, that x is a square root of z modulo n). Therefore the zero-knowledge security proof does not apply anymore, which leads to the following attack:

Since Π is the product of small primes, it is easy to compute square roots modulo Π, as opposed to computing square roots modulo n. Therefore, the attacker can generate an integer z at step 1 so that he is guaranteed to succeed if $b = 1$. Then if $b = 0$, the attacker will also succeed by computing a square root modulo Π, which is easy.

More precisely, at step 1, the attacker generates a random $u \in \mathbb{Z}_n$ and a random $r' \in [0, B]$, and sends $z = (u^2/v \mod n) + r' \cdot n$ to the verifier. Then at step 3, if $b = 0$, the attacker generates a random $r \in [0, \Pi]$, and solves:

$$x^2 = z - r \cdot n \mod \Pi$$

Since Π is the product of small primes, it suffices to take a square root of $z - r \cdot n$ modulo each of the small primes p_i. If $z - r \cdot n$ is not a square modulo a given prime p_j, it suffices to modify the value of $r \mod p_j$ without changing $r \mod p_i$ for $i \neq j$. This is possible since from the protocol, r is not required to belong to $[0, B]$. Eventually the attacker sends x and r to the verifier in RNS representation, and the attacker is successful with probability one.

Otherwise, if $b = 1$, then the attacker sends $y = u + t \cdot n$ for a random $t \in [0, B]$, and the verifier can check that $y^2 = z \cdot v \mod n$ since $u^2 = z \cdot v \mod n$.

Therefore, in both cases, the attacker passes the identification protocol with probability one, without knowing the private key.

6 Conclusion

We have shown that one of the time-efficient Fiat-Shamir variants introduced at Eurocrypt' 95 by Naccache *et al.* is insecure. Namely, a polynomial-time attacker can pass the identification protocol with probability one, without knowing the private key. Consequently, for practical implementations, we recommend to use [4]'s first time-efficient variant rather than [4]'s second time-efficient variant, which should be avoided. We believe that our attack illustrates the importance of careful security analysis of even apparently harmless variations of known secure protocols

References

1. A. Fiat and A. Shamir, *How to prove yourself: Practical solutions to identification and signature problems*, Proceedings of Crypto' 86, LNCS vol. 263, 1986.
2. S. Goldwasser, S. Micali and C. Rackoff, *The knowledge complexity of interactive proof-systems*, Proceedings of the 17th Annual ACM Symposium on Theory of Computing, 291-304, 1985.
3. D. Naccache, *Method, sender apparatus and receiver apparatus for modulo operation*, European patent application no. 91402958.2, November 5, 1991.
4. D. Naccache, D. MRaihi, W. Wolfowicz and A. di Porto, *Are Crypto-Accelrators really inevitable ?* 20 *bit zero-knowledge in less than a second on simple* 8*-bit microcontrollers*, Proceedings of Eurocrypt '95, Lecture Notes in Computer Science, Springer-Verlag.
5. A. Shamir, *Memory efficient variants of public-key schems for smart-card applications*, Proceedings of Eurocrypt '94, Lecture Notes in Computer Science, Springer-Verlag.