A PRACTICAL ZERO-KNOWLEDGE PROTOCOL FITTED TO SECURITY MICROPROCESSOR MINIMIZING BOTH TRANSMISSION AND MEMORY

Louis C. Guillou [1] and Jean-Jacques Quisquater [2]

[1] Centre Commun d'Etudes de Télédiffusion et Télécommunications
CCETT, BP 59
F–35 512 Cesson-Sevigné Cédex, France

[2] Philips Research Laboratory Brussels
Avenue Van Becelaere, 2
B–1 170 Brussels, Belgium
E-mail: jjq@prlb2.uucp

ABSTRACT

Zero-knowledge interactive proofs are very promising for the problems related to the verification of identity. After their (mainly theoretical) introduction by S. Goldwasser, S. Micali and C. Rackoff (1985), A. Fiat and A. Shamir (1986) proposed a first practical solution: the scheme of Fiat-Shamir is a trade-off between the number of authentication numbers stored in each security microprocessor and the number of witness numbers to be checked at each verification.

This paper proposes a new scheme which requires the storage of only one authentication number in each security microprocessor and the check of only one witness number. The needed computations are only 2 or 3 more than for the scheme of Fiat-Shamir.

1 INTRODUCTION

Interactive proofs and zero-knowledge protocols were recently introduced (Goldwasser, Micali and Rackoff, 1985). These concepts are very interesting but, at the moment, it is not possible to imagine such protocols in very small components (security microprocessor, tamperfree devices, smart cards, etc).

A new method based on these concepts was found by Fiat and Shamir (1986) and is very promising. But the main problems are the number of iterations (interaction between the prover and the verifier) and/or the memory needed by the prover. We propose an optimization of this protocol where we attain very few steps (3 steps, that is, one iteration) and low memory. The price to pay is longer computations.

Before explaining the new protocol, we need some definitions. We recall also the basic protocol of Fiat-Shamir.

2 DEFINITIONS: SHADOWS AND IMPRINTS FOR (RSA-BASED) SIGNATURES

- *Shadow:* One first completes a short message (half the length of the public modulus n) with a similar-sized redundancy, named *shadow*, then extracts the v^{th} root of this element in the chosen ring based on the composite integer n. The composition of these two consecutive operations is the secret operation S. The v^{th} power of a random element has a negligible probability of being shadowed. This method *with shadow* produces credentials, the most compact signatures. Due to multiplicative properties of RSA, the shadow must not be expressed multiplicatively in terms of the message.

- *Imprint:* Rather than signing long messages as chained blocks, one first uses a *hash function* to compute an *imprint* (shorter than n) of message M, then extracts as *appendix* H the v^{th} root of this imprint h. The composition of these two consecutive operations now is the secret operation S. The hash function must be *one-way*, such that it is infeasible to construct *collisions* of equivalent messages.

3 THE BASIC PROTOCOL OF FIAT AND SHAMIR

Let us remember that one must use factorization of n in order to extract efficiently a v^{th} root (such as a credential $A = X^{1/v} \bmod n$) in the ring of integers modulo n. The verification of such a credential reveals an element X carrying some identification data reflected by a redundant shadow. Let us name x, the identification data, and X, the resulting shadowed identity.

Suppose there exist a security device able to pick values at random and to multiply numbers modulo n (with about 512 bits) in a fast way. Each device receives from some trusted authority an authentication value A related to x using the method just described.

To authenticate such a processor claiming identification data x, the verifier negotiates a transaction with this device by repeating 20 to 30 times the elementary sequence described in the following paragraph. The number of iterations is a *security parameter* which exponentially limits the chances of a cheater.

The elementary sequence is (here $v = 3$):

- The processor picks at random an element in the ring $(1 < r < n - 1)$, raises it to the cube ($r^3 \bmod n$), and sends this cube to the verifier as a *test* T with the identity x.

- The verifier tosses a coin and transmits the outcome as a *question* q: `head` or `tail`.

- The processor transmits as witness t: either element r for `head`, or product $r \cdot A \bmod n$ for `tail`. The verifier raises this *witness* t to the cube mod$n$ in order to reveal, according to `head` or `tail`, either test T, or its product mod$n$ by shadowed identity X.

Each successful exchange increases verifier's confidence, because the value of credential A is needed to produce simultaneously the two values of witness t, while the first error reveals an unlucky cheater. Provers and verifiers make use of similar computing resources; they are both using the same composite number n. This method may, as well, be reversed. This method may use any exponent in place of the cube, with some caution for the square.

This was a first version of the method; various optimizations are possible, and some are already published. The next section will show a very interesting new version.

This zero-knowledge interactive procedure of demonstration leads to the emergence of new methods of signature, by replacing the random role of the verifier by a deterministic function, accepted by everybody, and difficult to invert, that is to say a *one-way* function. This is a summary of a method, due to Adi Shamir (for security reasons, k, the equivalent number of elementary iterations, is now about 60 so as to avoid forgery of signed messages). Our new method is also possible for this scheme of signature (see forthcoming paper: same authors).

4 THE NEW PROTOCOL: A DEEP VERSION

In this version, each security device with identity I receives an authentication value B (the inverse of A modulo n) computed by some authority from

$$A = J^{1/v} \bmod n$$

where J is the shadowed identity I; the factorization of n is only known by the authority.

The composite integer n (ala RSA) is distributed to everybody.

Here is the complete protocol for one verification:

- The processor picks at random an element r in the ring $(1 < r < n-1)$, computes $(r^v \bmod n)$, and sends the result to the verifier as a *test* T (or at least a part of the result) with the identity I.
- The verifier "tosses" a "deep" coin with integer values between 0 and $v-1$ and transmits the outcome as a *question* d.
- The processor transmits as *witness* t:

$$r \cdot B^d \bmod n$$

- The verifier computes

$$J^d \cdot t^v \bmod n$$

  and compares with the given bits of T.

In this version, there are only one exchange between the prover and the verifier (after the sending of the witness) and only one authentication value needed in the security device!

By definition, a cheater does not know B. Let us precisely evaluate the possibilities of a cheater.

- If a cheater guesses the question d, he can pick at random any new witness number t and then deduce the corresponding test number T by computing exactly as the verifier will do. There is an evident winning strategy for any lucky guesser.
- When the test number T has been transmitted to the verifier, let us evaluate the situation of a cheater which would be able to propose two witness numbers t' and t'' for two different questions d' and d''. The following short technical demonstration proves that such a cheater should no more be a cheater because he should easily deduce authentication number B from any pair (t', t'') of such witness numbers.

Proof of security
By hypothesis, $0 \leq d'' < d' \leq v - 1$.

Let us write the equation:

$$J^{d'} \cdot t'^{v} \bmod n = J^{d''} \cdot t''^{v} \bmod n,$$

which may transformed into:

$$J^{(d'-d'')} \cdot (t'/t'')^{v} \bmod n = 1.$$

Let us notice that $d' - d''$ is a positive integer, smaller than v, and prime with v (because v is prime). So, there exists a unique pair of positive integers k and m, in the range from 1 to $v - 1$, currently named Bezout coefficients of v and $d' - d''$, easily computed by the Euclidean algorithm, such that

$$m \cdot v - k \cdot (d' - d'') = \pm 1.$$

Let us raise the last equation to the power k and substitute: thus,

$$B = (J^{m} \cdot (t'/t'')^{k})^{\mp 1}\ modn.$$

Q.E.D.

At each use of the procedure, a cheater has exactly one chance on v to fool the verifier. The verifier has exactly $v - 1$ chances on v to defeat a cheater. After the procedure, the verifier has essentially learned nothing about the authentication value B because he cannot distinguish between an honest user and a very very lucky cheater.

No repetition of the procedure is needed as long as the size of the exponent v is sufficient to reach directly the level of security requested by the application. It is easy to specify: ten to sixteen bits for a local authentication, twenty to thirty bits for a remote authentication, and at least sixty bits for signature schemes based upon non-interactive zero-knowledge techniques.

The complete paper will give more explanations about the number of operations which related to the size of v.

A paper by Shamir (1984) uses a similar function but in a very different context.

REFERENCES

1. Gilles Brassard, David Chaum and Claude Crépreau, *Minimum disclosure proofs of knowledge*, July 1987.
2. Amos Fiat and Adi Shamir, *How to prove yourself: practical solutions to identification and signature problems.* Springer–Verlag, Lecture notes in computer science, N° 263, Advances in cryptology, Proceedings of CRYPTO '86, pp. 186–194, 1987.
3. Shafi Goldwasser, S. Micali and C. Rackoff, *The knowledge of interactive proof systems*, 17th ACM symposium on theory of computing, 1985, pp. 291-304.
4. Oded Goldreich, Silvio Micali and Avi Wigderson, *Proofs that yields nothing but the validity of the proof*, Workshop on probabilistic algorithms, Marseille, March 1986.
5. Adi Shamir, *Identity-based cryptosystems and signatures schemes*, Springer–Verlag, Lecture notes in computer science, N° 196, Advances in cryptology, Proceedings of CRYPTO '84, pp. 47–53, 1985.