A new identification scheme based on syndrome decoding *.

Jacques Stern

Laboratoire d'informatique, École Normale Supérieure

Abstract. Zero-knowledge proofs were introduced in 1985, in a paper by Goldwasser, Micali and Rackoff ([6]). Their practical significance was soon demonstrated in the work of Fiat and Shamir ([4]), who turned zero-knowledge proofs of quadratic residuosity into efficient means of establishing user identities. Still, as is almost always the case in public-key cryptography, the Fiat-Shamir scheme relied on arithmetic operations on large numbers. In 1989, there were two attempts to build identification protocols that only use simple operations (see [11, 10]). One appeared in the EUROCRYPT proceedings and relies on the intractability of some coding problems, the other was presented at the CRYPTO rump session and depends on the so-called Permuted Kernel problem (PKP). Unfortunately, the first of the schemes was not really practical. In the present paper, we propose a new identification scheme, based on error-correcting codes, which is zero-knowledge and is of practical value. Furthermore, we describe several variants, including one which has an *identity based* character. The security of our scheme depends on the hardness of decoding a word of given syndrome w.r.t. some binary linear error-correcting code.

1 The signature scheme

Since the appearance of public-key cryptography, basically all practical schemes have been based on hard problems from number theory. This has remained true with zero-knowledge proofs, introduced in 1985, in a paper by Goldwasser, Micali and Rackoff ([6]) and whose practical significance was soon demonstrated in the work of Fiat and Shamir ([4]). In 1989, there were two attempts to build identification protocols that only use simple operations (see [11, 10]). One relied on the intractability of some coding problems, the other on the so-called Permuted Kernel problem (PKP). Unfortunately, the first of the schemes was not really practical.

In the present paper, we propose a new identification scheme, based on the syndrome decoding problem for error-correcting codes (SD), which is zero-knowledge and seems of truly practical value. The proposed scheme uses a fixed $(n\text{-}k)$-matrix H over the two-element field. This matrix is common to all users and is originally built randomly. Thus, considered as a parity-check matrix, it should provide a linear binary code with a good correcting power.

[*] PATENT CAUTION: This document may reveal patentable subject matter

D.R. Stinson (Ed.): Advances in Cryptology - CRYPTO '93, LNCS 773, pp. 13-21, 1994.

Any user receives a secret key s which is an n-bit word with a prescribed number p of 1's. This prescribed number p is also part of the system. The public identification is computed as

$$i = H(s)$$

The identification scheme relies heavily on the technical notion of a *commitment*. If u is an sequence of bits, a *commitment* for u is the image of u via some cryptographic hash function. A commitment will be used as a one-way function: in order to disclose it, one announces the original sequence from which it was built. Once this is done, anyone can check the correctness of the commitment.

We now describe the basic interactive zero-knowledge protocol that enables any user (which we will call the prover as usual) to identify himself to another one (which we call the verifier). The protocol includes r rounds, each of these being performed as follows:

1. The prover picks a random n-bit word y together with a random permutation σ of the integers $\{1 \cdots n\}$ and sends commitments c_1, c_2, c_3 respectively for $\langle \sigma, H(y) \rangle$, $\langle y.\sigma \rangle$ and $\langle (y \oplus s).\sigma \rangle$ to the verifier. Note that $\langle , \rangle$ denotes the action of the hash function on the concatenation of all the bits of information of its arguments. A permutation σ is being considered in this setting as a vector of bits which encodes it; also note that $y.\sigma$ refers to the image of y under permutation σ.
2. The verifier sends a random element b of $\{0, 1, 2\}$.
3. If b is 0, the prover reveals y and σ. If b is 1, the prover reveals $y \oplus s$ and σ. Finally, if b equals 2, the prover discloses $y.\sigma$ and $s.\sigma$.
4. If b equals 0, the verifier checks that commitments c_1 and c_2, which were disclosed in step 2, have been computed honestly.
   If b equals 1, the verifier checks that commitments c_1 and c_3, were correct: note that σ is known from step 3 and that $H(y)$ can be recovered from $H(y \oplus s)$ by the equation

   $$H(y) = H(y \oplus s) \oplus i$$

   wher i is the user's public key.
   Now, if b is 2, the verifier checks commitments c_2 and c_3 and the property that the weight of $s.\sigma$ has the prescribed value p.

The number r of consecutive rounds depends on the required level of security and will be discussed further on as well as the values of the parameters n, k, p.

2 Security of the scheme

Of course, the security of the scheme relies on the difficulty of inverting the function

$$s \longrightarrow H(s)$$

when its arguments are restricted to valid secret keys. In order to give evidence of this difficulty, let us recall from [1] that it is NP-complete to determine whether a code has a word s of weight p whose image is a given k-bit word i. Let us also observe that, if p is small enough, finding s is exactly equivalent to finding the codeword w minimizing the weight of $t \oplus w$, when an element t of $H^{-1}(i)$ is chosen. But this is the problem of decoding unstructured codes which is currently believed to be unsolvable. Algorithms known for this problem (such as those described in [7, 12]) have a computing time that grows exponentially. A correct asymptotic evaluation of those algorithms has been recently given in [2] and confirmed by experiments in moderate sizes. Thus, as is the case for factoring, it is possible to state an *intractability assumption for the Syndrome Decoding Problem*, upon which the security of our scheme rests. Several versions of this assumption can be given, depending on the underlying model of security: for example, one may claim that no polynomial time probabilistic algorithm that takes as an input the parity check matrix H of an $(n, n-k)$ code, together with a binary k-bit vector i, can output the minimum-weight solution of the equation $H(s) = i$, with a non-negligible probability of success and a for a significant part of the inputs that correspond to a ratio k/n lying in a fixed interval. As usual, the precise formulation of these intractability assumptions is rather cumbersome: we keep it for the final version of our paper.

In order to counterfeit a given signature without knowing the secret key, various strategies can be used.

- Having only y and σ ready for the verifier's query and replacing the unknown s by some arbitrary vector t of weight p. In this case, the false prover hopes that b is 0 or 2 and the probability of success is $(2/3)^r$, where r is the number of rounds. A similar strategy can be defined with $y \oplus s$ in place of y.
- Having both y and $y \oplus t$ ready where t is some element such that $H(t) = i$, distinct from s and whose weight is not p. This yields the same probability of success.

It is fairly clear that shifting beetween one strategy to another has also the same probability of success. Furthermore, it can be proven formally that any probabilistic interactive algorithm that is accepted by the verifier with a probability of success that is significantly above $(2/3)^r$ can be turned into another that discloses the secret key or else outputs collisions for the cryptographic hash functions used for commitments. This is along the lines of the proof of theorem 3 of [5] and will appear in the full paper. Thus, if secure cryptographic hash functions do exist and if the intractability assumption for the Syndrome Decoding Problem holds, our system actually provides a proper proof of knowledge.

It can also be proved formally that the scheme is zero-knowledge. We will only give a brief hint. As we observed above, anyone can be ready to answer two queries among the three possible ones at each round. Hence, by using the standard idea of resettable simulation (see [6]), one can devise a polynomial-time simulation algorithm that mimics the fair communication between the prover and the verifier in expected time $O(2/3.r)$. Speaking more informally, the only information that comes out of a round is either a random word, y or $y \oplus s$

together with a random permutation, or else one random word y together with another random word of fixed weight p. Therefore, it is virtually impossible to undertake any statistical analysis that might reveal s.

We now discuss the size of the various parameters. Although a large variety of choices are opened, we recommend that n equals $2k$. Thus possible sizes are:

- $n = 512, k = 256$
- $n = 1024, k = 512$

As for the value of p, we observe that, although it is tempting to lower it, this would be rather dangerous: using the arguments of [12], one can see that secret keys of small weight (e.g. $p = 20$) will presumably be found. We recommend that p is chosen slightly below the value given by the so-called Warshamov Gilbert bound ([9]), which provides a theoretical limit value for the minimal weight d of a (n, k) random code, namely:

$$\frac{d}{n} = H_2(\frac{k}{n})$$

where H_2 is the entropy function defined by:

$$H_2 = -x \log_2(x) - (1 - x) \log_2(1 - x)$$

When n is $2k$, d is approximately $0.11n$ and values of p corresponding to the three possible data mentionned above are about 56 in the first case and 110 in the second. From the estimations of [2], the workfactor for the known algorithms that might reveal the secret is above $O(\Omega^k)$ where Ω is about 1.18 for the chosen values. This yields a value of 2^{61} for $k = 256$, which can be considered as unfeasible, especially as this is an estimate of a work factor, which means that the constant under the big-O notation is fairly large.

3 Performances of the scheme.

We will restrict ourselves to various remarks.

1. It might be thought that the proposed scheme requires a large amount of memory. THis is not accurate: on one hand, because the operations to perform are very simple, they can be implemented in hardware in a quite efficient way; on the other hand, if the scheme is implemented, partially or totally, in software, it is not neccessary to store all of H. One can only store words corresponding to some chosen locations and extend these by a fixed software random number generator.
2. The communication complexity of the protocol is comparable to what it is in the Fiat-Shamir scheme. If we assume that permutations come from a a seed of of 120 bits via a pseudo-random generator and that hash values are 64 bits long, which is perfectly acceptable from a security point of view, we obtain an average number of bits per round which is close to 950 bits when $n = 512$. Using a trick that saves space for sending $s.\sigma$, this can be lowered to approximately 860 bits.

3. The security of the scheme can be increased by taking n, k and p larger. The figures $n = 512$, $k = 256$ are to be considered as a minimum.
4. The heaviest part of the computing load of the prover (which is usually a portable device with a limited computing power) is the computation of $H(y)$, which is done in step 1. This load can be drastically reduced by extending the protocol to a 5-pass version which will be discussed in the next section
5. Considering that the probability of any cheating strategy is bounded above by $(2/3)^r$, where r is the number of rounds, we see that the basic protocol has to be repeated 35 times in order to achieve a level of security of 10^{-6}. A key difference between the proposed schemes and previous proposals is the fact that a single round offers only security 1/3 instead of 1/2. There is a variant of our protocol that achieves security 1/2. It will be discussed further on in this paper
6. As is the case for Shamir's PKP, our scheme is not identity based. This means that public keys have to be certified by the issuing authority. We will consider below a variant of this scheme with an identity-based character.

4 A variant that minimizes the computing load.

In order to minimize the computing load, we introduce a 5-pass variant. this variant depends on a new parameter q.

Step 1 is the same except that commitment c_1 is replaced by $\langle\sigma\rangle$. Thus $H(y)$ is not computed at this stage.

After step 1, the verifier sends back a choice of q indices from $\{1 \cdots k\}$ (these refer to a choice of q rows of the matrix H).

The prover answers by sending the list of bits $b_1, \cdots, b_q$ corresponding to the selected indices of the vector $H(y)$.

The rest of the protocol is similar (with obvious changes for the checking step 6).

Of course this opens up new strategies for cheating: basically, one will try to have both y and $y \oplus t$ ready where t is some element of weight p such that $H(t)$ differs from i on a small number of bits, say h. This will increase the probability of success by an amount which is close to $\frac{1}{3}(1-\frac{h}{k})^q$. In the case $n = 512$, $k = 256$, $p = 56$, $q = 64$, $h = 15$, this extra amount is roughly 0.007 and the loss can be compensated by adding only one extra round of the protocol.

Of course, the new strategy becomes more and more successful as h decreases; for example, making $h = 4$ and keeping all other figures unchanged increases the probability of cheating successfully to 0.78. But it can be shown that finding a t as above is equivalent to finding a word s' of weight at most $p + h$ with a given syndrome $H(s') = i$ and it is believed that, when h is very small, this remains unfeasable. Of course, many other trade-offs between n, k, p, h, q are possible.

5 A variant that minimizes the number of rounds.

In this variant, the secret key s is replaced by a simplex code generated by $s_1, \cdots, s_m$. Recall that a simplex code of dimension m has all its non zero codewords of weight 2^{m-1} (see [9]). It is easy to construct such a code with length $2^m - 1$ and to extend the length to any larger value n. The corresponding public key is the sequence $H(s_1), \cdots, H(s_m)$.

It is unknown whether or not it is easier to recover the family of secret vectors than to recover a single one. As a set of minimal values, we recommend $m = 7$ together with $n = 576$ and $k = 288$. This ensures consistency with our previous estimates.

We now describe one 5-pass round of a protocol that achieves identification.

1. The prover picks a random n-bit word y together with a random permutation σ of the integers $\{1 \cdots n\}$ and sends commitments c_1, c_2 respectively for $\langle \sigma, H(y) \rangle$, $\langle y.\sigma, s_1.\sigma, \cdots, s_m.\sigma \rangle$ to the verifier.
2. The verifier sends a random binary vector $b_1, \cdots, b_m$.
3. The prover computes
$$z = (y \oplus \bigoplus_{j=1}^{m} b_j s_j).\sigma$$
and sends z to the verifier
4. The verifier responds with a one bit challenge b.
5. If b is 0, the prover reveals σ. If b is 1, the prover discloses $y.\sigma$ as well as the full sequence $s_1.\sigma, \cdots, s_m.\sigma$.
6. If b equals 0, the verifier checks that commitment c_1 has been computed honestly.Note that $H(y)$ can be recovered from $H(z.\sigma^{-1})$, the sequence of public keys and the binary vector issued at step 2.
If b equals 1, the verifier checks that commitment c_2, was correct, that the computation of z is consistent and that $s_1, \cdots, s_m$ actually form a simplex code of the required weight.

This basic round can be repeated and it can be shown that the probability of success of a single round, when no information about the secret keys is known, is at most $\frac{1+2^{m-1}}{2^m}$, which is essentially 1/2. On the other hand, it is clear that the communication complexity is worse than in the single-key case.

6 An identity based version.

One attractive feature of the Fiat-Shamir scheme is that the public key can be derived from the user's identity, thus avoiding the need to link both by some signature from the issuing authority. Neither Shamir's PKP scheme nor our basic scheme have this feature. We now investigate various modifications that can turn our scheme into an identity-based scheme.

A first possibility is to use a set of t simplex codes of dimension m. If $s_1, \cdots, s_m$ is the first of these codes, then m bits can define a specific key

$$\bigoplus_{j=1}^{m} b_j s_j$$

and therefore, assuming that the identity of a user is given by tm bits, one can define t secret keys for each user. Now, these secret keys can be used randomly to perform identification. The verifier has to store tm vectors of k bits (the images of the basis vectors of the codes), which is much less than a full directory of users. We suggest $m = 7$ $t = 6$ as a reasonable implementation.

The other possibility we describe is a bit more intricate. It uses a "master code" consisting of t vectors $s_1, \cdots, s_t$ whose one-bits only cover a subset T of the possible n locations. Given the identity of a user as a sequence of t bits $e_1, \cdots, e_t$ it is easy, by Gaussian elimination, to find a linear combination s of the s_j's such that

$$H(s) = \sum_{j=1}^{t} e_j H(s_j)$$

and whose weight p is approximately $|T| - t/2$. This will be the secret key of the user, computed by the issuing authority. The security of this variant is more difficult to analyze. Typically, the existence of the master code implies that some code that can de defined from the public data has a vector with a small number of ones located within the (unknown) set T. The dimensions should be designed in order that the weight of this vector is large enough. We suggest, as an example, $t = 56$, $p = 95$, $n = 864$, $k = 432$.

7 An analogous scheme based on modular knapsacks.

In this section, we briefly mention an analogous scheme that can be devised by replacing the $\{0,1\}$-matrix H by a matrix over a finite field with an extremely small number of elements (typically 3, 5 or 7). In this situation, the weight constraint is replaced by the constraint that the secret solution s to the equation $H(s) = i$ consists entirely of zeros and ones. Thus the underlying difficult problem is a modular knapsack. Although it is known that knapsacks can be attacked by methods based on lattice reduction (see [8, 3]), it is clear also that these methods do not apply to the modular case, at least when the modulus m is very small. Possible values for the scheme are (with the same notations as above)

- $n = 196$, $k = 128$, $m = 3$
- $n = 384$, $k = 256$, $m = 3$
- $n = 128$, $k = 64$, $m = 5$
- $n = 192$, $k = 96$, $m = 5$

One round of the protocol is performed as follows:

1. The prover picks a random vector y with coefficients from the m-element field, together with a random permutation σ of the integers $\{1 \cdots n\}$ and sends commitments c_1, c_2, c_3 respectively for $\langle \sigma, H(y) \rangle$, $\langle y.\sigma \rangle$ and $\langle (y \oplus s).\sigma \rangle$.
2. The verifier sends a random element b of $\{0, 1, 2\}$.
3. If b is 0, the prover reveals y and σ. If b is 1, the prover reveals $y + s \bmod m$ and σ. Finally, if b equals 2, the prover discloses $y.\sigma$ and $s.\sigma$.
4. The verifier makes the obvious checks.

8 Conclusion

We have defined a new practical identification scheme based on the syndrome decoding problem (SD). We have also described several variants of this scheme. The scheme only uses very simple operations and thus widens the range of techniques that can be applied in crytography. We welcome attaccks from readers and, as is customary when introducing a new cryptographic tool, we suggest that the scheme should not be adopted prematurely for actual use.

References

1. E. R. Berlekamp, R. J. Mc Eliece and H. C. A. Van Tilborg. On the inherent intractability of certain coding problems, *IEEE Trans. Inform. Theory,* (1978) 384-386.
2. F. Chabaud, Asymptotic analysis of probabilistic algorithms for finding short codewords, *Proceedings of EUROCODE 92*, Lecture Notes in Computer Science, to appear.
3. M. J. Coster, A. Joux, B. A. LaMacchia, A. M. Odlyzko, C. P. Schnorr and J. Stern, Improved low-density subset sum algorithms, *Computational Complexity*, to appear.
4. A. Fiat and A. Shamir, How to prove yourself: Practical solutions to identification and signature problems, *Proceedings of Crypto 86*, Lecture Notes in Computer Science 263, 181-187.
5. U. Feige, A. Fiat and A. Shamir, Zero-knowledge proofs of identity, *Proc. 19th ACM Symp. Theory of Computing*, 210-217, (1987) and *J. Cryptology*, **1** (1988), 77-95.
6. S. Goldwasser, S. Micali and C. Rackoff, The knowledge complexity of interactive proof systems, *Proc. 17th ACM Symp. Theory of Computing*, 291-304, (1985).
7. J . S. Leon, A probabilistic algorithm for computing minimum weights of large error-correcting codes. *IEEE Trans. Inform. Theory*, IT-34(5): 1354-1359.
8. J. C. Lagarias and A. M. Odlyzko, Solving low-density subset sum problems, *J. Assoc. Comp. Mach.* **32** (1985), 229-246.
9. F. J. MacWilliams and N. J. A. Sloane, The theory of error-correcting codes, North-Holland, Amsterdam-New-York-Oxford (1977).
10. A. Shamir, An efficient identification scheme based on permuted kernels, *Proceedings of Crypto 89,* Lecture Notes in Computer Science 435, 606-609.
11. J. Stern, An alternative to the Fiat-Shamir protocol, *Proceedings of Eurocrypt 89*, Lecture Notes in Computer Science 434, 173-180.
12. J. Stern, A method for finding codewords of small weight, *Coding Theory and Applications*, Lecture Notes in Computer Science 388 (1989), 106-113.