An Efficient Identification Scheme Based on Permuted Patterns*

Shahrokh Saeednia

Université Libre de Bruxelles
Département d'Informatique
CP 212, Boulevard du Triomphe
1050 Bruxelles, Belgium
saeednia@ulb.ac.be

Abstract. This paper proposes a new identification scheme based on a hard partition problem rather than factoring or discrete logarithm problems. The new scheme minimizes at the same time the communication complexity and the computational cost required by the parties. Since only simple operations are needed for an identification, our scheme is well suited for smart cards with very limited processing power. With a good implementation, the scheme is much faster than the Fiat-Shamir or Shamir's PKP schemes.

Keywords: Identification, Zero-knowledge, Smart cards.

1 Introduction

Since the introduction of zero-knowledge proofs [2], many identification schemes have been proposed from which the Fiat-Shamir [1], the Guillou-Quisquater [3] and the Schnorr [5] schemes have attracted most attention. The security of these schemes is mainly based on the intractability assumption of either factoring large integers or the discrete logarithm problem, two closely related problems that actually constitute the basis of the public key cryptography. The major disadvantage of these schemes is the important computing cost required by modular maltiplications of very large integers, which is significant when the computing power offered by the processor in use is low, as is the case of smart cards.

Many efforts have been devoted to create identification schemes using other hard problems allowing a fast implementation. In 1989, Shamir [6] proposed the first identification scheme that makes use of simple operations, while relies on some NP-complete problem, known as permuted kernels. The scheme needs 5 moves at each iteration and 20 iterations to achieve the classical security level of 10^{-6}. In 1993, Stern [7] proposed another linear scheme based on syndrome decoding that is a three-move one, but requires about 34 iterations to attain the same security level as in Shamir's scheme. In 1996, another scheme is proposed

* Patent pending. This work is supported by the Belgian Ministry of Wallone Region, grant no. 981/3743.

by Pointcheval [4] based on permuted perceptrons problem that uses the same framework as Stern's scheme but needs more than 48 iterations in order to be secure in practice.

The problem with all these schemes is, however, the high communication rate between the prover and the verifier that considerably reduces the efficiency of the schemes. The problem is due to the number of iterations in the protocols, which is closely connected to the high probability of fraud at each iteration that is a constant value (1/2 for Shamir's, 2/3 for Stern's and 3/4 for pointcheval's schemes). In the following sections, we propose the first linear scheme in which the probability of fraud at each iteration depends on some parameters of the scheme and so allows reducing that probability and subsequently, the number of iterations. Some initial analyses show that we may choose system parameters in such a way that this probability be reduced to 1/10 (so that, we only need 6 iterations to attain the security level of 10^{-6}), while a very fast implementation is assured. Since only basical operations such as byte-multiplications are needed, the scheme may be implemented efficiently on smart cards with very limited computing power.

2 Notations and Problems

We call

- a (k, p)-identity vector, a vector of size kp in which each value belonging to Z_p occurs exactly k times.
- a (k, p)-partitionable vector, a vector of size kp that may be partitioned into k disjoint subvectors of size p, each of which containing the same value $\in Z_p$.

So, any permutation of the vector

$$(\underbrace{0, \ldots, 0}_{k}, \underbrace{1, \ldots, 1}_{k}, \ldots, \underbrace{p-1, \ldots, p-1}_{k})$$

is a (k, p)-identity vector, and any permutation of one of the p^k vectors

$$(\underbrace{e_1, \ldots, e_1}_{p}, \underbrace{e_2, \ldots, e_2}_{p}, \ldots, \underbrace{e_k, \ldots, e_k}_{p})$$

is a (k, p)-partitionable vector, where $e_1, \ldots, e_k \in Z_p$.

For example, $(1, 1, 2, 1, 1, 2, 1, 2, 2, 1, 2, 1, 1, 1, 1)$ is a $(3, 5)$-partitionable vector, as it may be partitioned into $(1, 1, 1, 1, 1)$, $(1, 1, 1, 1, 1)$ and $(2, 2, 2, 2, 2)$.

Problem 1 :

Given an odd prime p and an $(m \times n)$ matrix H of integers $\in Z_p$, where $n = kp$ for some fixed $k > 0$, and a vector Y of size m, with $y_i \in Z_p$, find a (k, p)-identity vector X such that

$$Y = HX \pmod{p}. \tag{1}$$

The problem is NP-complete even for $k = 1$ and $Y = (0, \ldots, 0)$, since this is actually a particular hard instance of the permuted kernels problem (PKP). More generally, we can show that any arbitrary instance of the PKP may be transformed to an instance of the above problem.

Problem 2 :

Given an odd prime p and an $(m \times n)$ matrix H of integers $\in Z_p$, where $n = kp$ for some fixed $k > 1$, and a vector Y of size m, with $y_i \in Z_p$, find a (k, p)-partitionable vector X such that

$$Y = HX \pmod{p}.$$

Note that, unlike the problem 1, we require here that $k > 1$. The problem of finding a (k, p)-partitionable vector is somehow different from the problem 1, because here we are not limited to find a solution with fixed values for entries, but from a particular set of values. What makes this problem hard is the predetermined number (that is p) of each value appearing in the solution. The problem consists of finding k integers $e_1, \ldots, e_k \in Z_p$ and k disjoint subsets $S_1, \ldots, S_k$ of the set $(1, 2, \ldots, kp)$, each of which having p elements, such that $\forall i = 1, \ldots, m$

$$\sum_{\ell=1}^{k} e_\ell (\sum_{j \in S_\ell} h_{ij}) \pmod{p} = y_i$$

If $k \geq m$ then the problem is easy, whatever the partition is. In fact, one can choose an arbitrary partition and form a system of equations with m equations and k unknowns that may be solved in polynomial time by attributing arbitrary values to $k - m$ unknowns and solving the resulting $m \times m$ system. Contrarily, when $k < m$ one should solve a system in which the number of unknowns is less than that of the equations. So, the system may have a solution or not, depending on the chosen partition. We believe that this is a hard partition problem, but its NP-completenss remains an open problem.

We may also combine the two problems as follows that is more suitable for the security analysis of our scheme.

The main problem :

Given an odd prime p and an $(m \times n)$ matrix H of integers $\in Z_p$, where $n = kp$ for some fixed $k > 1$, and a vector Y of size m, with $y_i \in Z_p$, find a combination of a (ℓ, p)-identity vector and a $(k - \ell, p)$-partitionable vector X such that

$$Y = HX \pmod{p}$$

for any value of $0 \leq \ell \leq k$.

3 The Scheme

Initialization: The authority chooses a relatively small odd prime $p > 3$ and an $m \times n$ matrix H of integers in Z_p, where $n = kp$ for some fixed $k \geq 2$.

Key Generation: Every user chooses a random permutation π over $\{1, \ldots, n\}$ as his secret key and forms the (k,p)-identity vector

$$X = (\underbrace{0, \ldots, 0}_{k}, \underbrace{1, \ldots, 1}_{k}, \ldots, \underbrace{p-1, \ldots, p-1}_{k})_\pi$$

and computes his public key as $Y = HX \pmod{p}$.

Protocol:

1. $\bar{A}$ chooses a (k,p)-partionable vector R by randomly and independently selecting k numbers $e_1, \ldots, e_k \in Z_p$ and p permutations $\sigma_1, \ldots, \sigma_p$ over $\{1, \ldots, k\}$ and computing
$$R = ((e_1, \ldots, e_k)_{\sigma_1}, \ldots, (e_1, \ldots, e_k)_{\sigma_p})_\pi$$
and sends $Z = HR \pmod{p}$ to $\bar{B}$.
2. $\bar{B}$ selects a random number $0 < c < p$ and sends it to $\bar{A}$.
3. $\bar{A}$ sends $V = R + cX \pmod{p}$ to $\bar{B}$.
4. $\bar{B}$ verifies whether V is a (k,p)-identity vector and whether $Z + cY \equiv HV \pmod{p}$.

If after t rounds, A correctly answered to all $\bar{B}$'s questions, $\bar{B}$ accepts A.

4 Security assessment

4.1 Completeness

Theorem 1. *If A follows the protocol, $\bar{B}$ always accepts A's proof.*

Proof. There exists a permutation τ such that

$$R_{\tau^{-1}} = (\underbrace{e_1, \ldots, e_1}_{p}, \ldots, \underbrace{e_k, \ldots, e_k}_{p})$$

and

$$X_{\tau^{-1}} = (\underbrace{(0, \ldots, p-1), \ldots, (0, \ldots, p-1)}_{k \text{ times}}).$$

Hence,

$$cX_{\tau^{-1}} = ((0, \ldots, p-1)_\delta, \ldots, (0, \ldots, p-1)_\delta)$$

for some permutation δ. Clearly, $R_{\tau^{-1}} + cX_{\tau^{-1}}$ is a (k,p)-identity vector and any permutation of the latter remains so. Thus, $(R_{\tau^{-1}} + cX_{\tau^{-1}})_\tau = R + cX$ is also (k,p)-identity.

The second part of the verification, i.e., the fact that $Z + cY \equiv H(R + cX) \pmod{p}$ is trivial.

4.2 Security from the verifier's point of view (Soundness)

As usual, a cheating prover may guess the verifier's question c in advance and choose a (k,p)-identity vector V, compute $R = V - cX$ and send $Z = HR$ in step 1. Evidently, if verifier's question happens to be c then the cheating prover can successfully answer to it. However, the probability of this event is just $1/(p-1)$ for one round and $1/(p-1)^t$ for the whole protocol.

To answer all verifier's possible questions, an impersonator may clearly find X or a similar (k,p)-identity vector satisfying (1). However, if he finds a combination of a (ℓ,p)-identity vector and a $(k-\ell,p)$-partitionable vector for any value of ℓ (for $\ell = 0$, it will be a (k,p)-partitionable vector), he can still answer correctly to all verifier's questions. In fact, it suffices to determine a permutation τ, such that

$$\tilde{X}_{\tau^{-1}} = (\underbrace{(e_1,\ldots,e_1}_{p},\ldots,\underbrace{e_{k-\ell},\ldots,e_{k-\ell})}_{p},\underbrace{(0,\ldots,p-1),\ldots,(0,\ldots,p-1))}_{\ell \text{ times}}$$

and set R in step 1 of the protocol as

$$R = (\underbrace{(0,\ldots,p-1),\ldots,(0,\ldots,p-1)}_{k-\ell \text{ times}},\underbrace{(e_1,\ldots,e_1}_{p},\ldots,\underbrace{e_\ell,\ldots,e_\ell)}_{p})_\tau$$

This guarantees that for any $0 < c < p$, $c\tilde{X} + R$ is a (k,p)-identity vector. However, we believe that finding such vectors is as hard as finding a (k,p)-identity vector satisfying (1).

Now, let us see what is the probability of success if $\tilde{X}$ is neither of these vectors. First, let us note that $\tilde{X}$ may contain subvectors that are (ℓ,p)-identity or (j,p)-partitionable, with $j + \ell < k$. In this case, we may reduce the problem to a smaller one by just considering the subvector with $k-(\ell+j)$ elements of $\tilde{X}$ that are neither in the (ℓ,p)-identity nor in the (j,p)-partitionable subvectors. Let us call this subvector X'.

We believe that, when p is not too small, the number j of values $0 < c_i < p$ for which there exist a vector R such that all vectors $R + c_i X'$, $i = 1,\ldots j$ are (k,p)-identity is at most 2.

In fact, a cheating prover, knowing a particular X' with $\sum_{i=1}^{n} x'_i = 0 \pmod p$ (that is neither of the above vectors) satisfying (1), may guess 2 out of $p-1$ possible verifier's questions and manage to find the corresponding answers. To do so, he has first to find two (k,p)-identity vectors V_1 and V_2 such that $V_1 - V_2 = (c_1 - c_2)X'$ and set $R = V_1 - c_1 X'$ (or $R = V_2 - c_2 X'$) afterward. We believe, however, that for no other c_i, $V_i = R + c_i X'$ can be so, if p is not too small.

Examples exist for very small p, that allow to prepare answers to 3 verifier's questions. For example, for $p = 7$ and $k = 2$ if we find

$$X' = (0,5,1,4,5,4,3,4,5,3,4,0,3,1)$$

then we may decide to answer to questions $c_1 = 1$ and $c_2 = 2$. So, we can find $(2,7)$-identity vectors V_1 and V_2 such that

$$V_2 - V_1 = (c_2 - c_1)X' = X'$$

as

$V_1 = (3,0,1,6,1,4,6,0,2,2,3,4,5,5)$ and $V_2 = (3,5,2,3,6,1,2,4,0,5,0,4,1,6)$.

With such choices, we find

$$R = V_1 - X' = (3,2,0,2,3,0,3,3,4,6,6,4,2,4).$$

Curiously, the vector $R + 5X'$ is also a $(2,7)$-identity vector. Note that, neither of the vectors $R + 3X'$, $R + 4X'$ and $R + 6X'$ is $(2,7)$-identity.

A simulation shows that the value of j depends essentially on p and less on k. There are examples with $p = 7$ and $k = 20$, for which $j = 3$, while for $p = 31$, even with $k = 1$, we didn't find any X' for which $j > 2$. Also, no X' has been found with $j > 3$, for any value of p and k.

So, combining the above assumption with the underlying hard problems allows us to state a more general intractability assumption that is more suitable for further analysis of our scheme:

Conjecture 1. Given a prime p, an $(m \times n)$ matrix H of integers $\in Z_p$, where $n = kp$ for some fixed $k \geq 2$, and a vector Y of size m, with $y_i \in Z_p$, when p is not too small, it is difficult to find more than two (k,p)-identity vectors V_i such that for some arbitrarily chosen vector Z, $HV_i \equiv Z + c_i Y \pmod{p}$, where $0 < c_i < p$.

We assume that for small values of p, $j = 3$, whatever the value of k. If the conjecture above is true, then in the best case, a cheating prover can only answer to 2 (or 3, for small values of p) out of $p - 1$ verifier's possible questions. This precisely means that the probability of fraud at each round is only $2/p$ (or $3/p$, for small values of p). To take the maximum security, we consider the probability $3/p$ of cheating when choosing the size of the parameters.

4.3 Security from the prover's point of view (Zero-knowledge)

Even though the vector R is at last permuted with π, the protocol is zero-knowledge, due to the use of partial permutations σ_i and the fact that the k values e_i are chosen randomly and independently.

If we don't use σ_i's, besides the fact that the number of possible vectors R becomes too limited, it is possible to identify k disjoint subvectors of X, each of which containing values $0, \ldots, p-1$, without knowing the position of each value. In fact, all vectors R may be partitioned into k subvectors of p elements, having the same value (e_i). Without using σ_i's, these subvectors occupy always the same positions in all R's (corresponding to different execution of the protocol). Let us note these subvectors as $R_1, \ldots, R_k$. Clearly, each of these subvectors corresponds to a subvector of X, that we call X_i, consisting of a permutation of $0, \ldots, p-1$. So, if we can identify each subvector R_i, then we may determine the related X_i.

Since R is never communicated in the protocol, this is not directly possible. However, by considering two values of V, that we note V' and V", corresponding

to the same verifier's question, one may determine $R' - R" = V' - V"$. Without σ_i's, this vector consists of k disjoint subvectors containing equal values (like any R) and thus allows to identify $X_1 \ldots, X_k$.

Note that, if two or more subvectors in $R' - R"$ have the same value, one may distinguish between them by considering other pairs of $(V', V")$.

Hence, trying $(p!)^k$ permutations and for each one a matrix product, we may determine the exact value of X. Although this attack is not efficient, it shows that without σ_i's the protocol is not zero-knowledge.

On the other hand, if the e_i's are not chosen randomly and independently, then it is possible to find X. For example, if we require that e_i's be all different (that is only possible when $k \leq p$), then no pair of entries in V corresponding to equal values in X may never be equal:

$$\nexists v_i, v_j : x_i = x_j \text{ and } v_i = v_j$$

Consequently, by fixing a position in V and discarding the positions that have the same value as the fixed position, after considering a polynomial number of V's, only k positions remain. These positions correspond obviously to those of equal values in X.

If we continue in this way, we can sequentially identify all the p subvectors of X, each of them having the same value, without knowing their exact value. Hence, trying $p!$ permutations, X may be determined. The attack may fail for large values of p, nevertheless the protocol is not zero-knowledge in this case.

If both conditions are satisfied (that is the case in our scheme), the protocol is zero-knowledge. Firstly, because of the random σ_i's, $R' - R"$ does not consist, with very high probability, of k subvectors of equal values and so does not yield any knowledge about X. Secondly, due to the fact that e_i's are chosen independently and randomly, for any fixed position i, the probability that $v_j = v_i$, for any $j \neq i$, is $1/p$. This precisely means that, we can no longer distinguish between different positions with respect to the value of v_i, as we did above.

A formal proof of zero-knowledgeness will be given in the full paper.

To conclude this section, let us notice that if we interchange the role of X and R in the protocol, i.e., if X is a (k, p)-partitionable vector and R is a (k, p)-identity one, not only the protocol functions correctly, but also this allows us to consider an additional verifier's questions ($c = 0$). The reason for which we did not adopt this choice is just because the protocol will no longer remain secure as it leaks enough information to retrieve X from some values of V.

5 Parameters and Performances

First of all, we recall that p may be any prime > 3, but not 3 or lower, because, as we noticed above, a cheating prover may prepare to answer correctly to 3 verifier's questions out of $p - 1$ possible questions, so that for $p = 3$, it would answer to all questions. However, p should not be too small. Since the probability

of success for a cheating prover is $3/(p-1)$ at each iteration, for small values of p (e.g. $p = 7$), we must have a considerable number of iterations (e.g., $t = 20$) in order to attain a sufficiently high security level (e.g., 2^{-20}). For this reason, we propose to choose $p = 31$ that allows us to have only 6 iterations to achieve the practical security of 10^{-6}.

On the other hand, the number of possible vectors R is closely connected to the choice of k. In fact, the number of basic (k, p)-partitionable vectors (i.e., without considering partial permutations σ_i) is p^k, while the number of permutations in each subvector (to which σ_i is applied) is $k!$. This means that, the total number of possible (k, p)-partitionable vectors is $p^k(k!)^p$. So, we propose to choose $k = 8$ that, with the chosen value of p, provides a large space of about 2^{514} random voctors R.

With these values for k and p, we have $n = kp = 248$ and we propose to take $m = 124$. Thus, H will be a large matrix and it seems that we deal with a large matrix product. However, Since we know the number of occurrence of each value e_i in R (that is p), we may reduce the matrix product HR into an $m \times k$ matrix product, by means of k passes on R, which minimize the computational cost required by the prover[1].

The matrix product algorithm works as follows:

$Z = (0, \ldots, 0)$
FOR $j = 1$ to k DO
  $W = (0, \ldots, 0)$
  FOR $i = 1$ to n DO
    IF $r_i = e_j$ THEN
      FOR $\ell = 1$ to m DO $w_\ell := w_\ell + H_{\ell i}$ END
    END
  END
  FOR $i = 1$ to m DO $z_i := z_i + (w_i \times e_j)$ END
END

Although this algorithm performs kn tests (that are not necessary in classical matrix product methods), the number of multiplications is reduced by $1/p$, which allows us to consider large matrices.

For the above values, a matrix product of this form needs $mk = 992$ multiplications of 5-bit integers, in our case. Since we only need 6 iterations, we have 5952 byte-multiplications in the whole protocol that, considering the kn additional tests, is equivalent to less than 2 multiplications of two 512-bit integers.

With this performance, our scheme, with only one key of short size (see below), is 7 times faster than the Fiat-Shamir scheme (with 5 keys) and may be efficiently implemented on smart cards with limited processing power.

To compare with other linear schemes, we recall that, unlike all previous proposals, in our scheme the probability of fraud at each iteration, is a function of p and may be decreased arbirarily, by choosing larger values of p. In the following

[1] The same technique allows the verifier to compute HV efficiently.

table, we consider Stern's syndrome decoding [7], Shamir's permuted kernels [6] and Poincheval's permuted perceptrons [4] protocols and show that our scheme is much more efficient than those schemes by far, from the computation and communication points of view.

	SDP Stern	PKP Shamir	PPP Pointcheval	New Scheme
Public key (bits)	256	296	144	620
Secret key (bits)	512	384	117	248
Bits sent by round	954	832	896	1865
Byte-multiplications. by round (prover)	1260	1369	42676	992
Number of rounds	34	20	48	6
Total bits sent	32436	16640	43008	11190
Total byte-Multiplications (prover)	42840	27380	2048448	5952

Note that the number of byte-multiplications, in the case of Stern's and Pointcheval's schemes, is computed by transforming bit operation timings into byte-multiplication clocks.

Since we believe that most permutations are suitable for our scheme, instead of a mapping, we choose the secret key as a vector of $n = 248$ bits that may be used by a very simple algorithm to form X and R. The storage of the public key needs 620 bits, but may be reduced to 128 bits if we consider $f(Y)$ rather than Y itself, where f is a hash function. In this case, the verifier should check, in the last step, whether $f(HV - Z \pmod p)$ equals the public key.

As is the case of all other linear schemes, the storage of the common matrix H is not necessary in our scheme. Since the above matrix product algorithm performs operations on columns of H at the same time, entries of each column of H may be generated by a simple pseudo-random function with i; $i = 1, \ldots, n$; as the argument.

6 Conclusion

We proposed a new identification scheme whose security depends on a new partition problem. Unlike all previously proposed linear schemes, the probability of fraud at each iteration of our scheme is not constant and may be decreased by considering larger values of one of the system parameters. The scheme uses only simple operations and needs a few iterations to attain a sufficiently high security level. This precisely means that, it minimizes both computation and communication complexity. To compare with existing identification schemes, our scheme is about 7 times faster than the Fiat-Shamir scheme that is one of the fastest and the most used scheme in practice. We intoduced a conjecture that constitutes the security basis of our scheme and encourage readers to analyze that conjecture and attack the scheme from all points of view.

References

1. A. Fiat and A. Shamir, "How to prove yourself: practical solutions to identification and signature problems", *Advances in Cryptology* (Proceedings of *Crypto* '86), Lecture Notes in Computer Science, vol. 263, Springer-Verlag, 1987, pp. 186-194
2. S. Goldwasser, S. Micali and C. Rackoff, "The knowledge complexity of interactive proof systems", *SIAM J. Comp.*, vol. 18, 1989, pp. 186-208
3. L. Guillou and J. J. Quisquater, "A practical zero-knowledge protocol fitted to security microprocessors minimizing both transmission and memory" *Advances in Cryptology* (Proceedings of *EuroCrypt* '88), Lecture Notes in Computer Science, vol. 339, Springer-Verlag, 1989, pp. 123-128
4. D. Poincheval, " A new identification scheme based on the perceptrons problem" *Advances in Cryptology* (Proceedings of *EuroCrypt* '95), Lecture Notes in Computer Science, vol. 921, Springer-Verlag, 1995, pp. 319-328
5. C. Schnorr, "Efficient signature generation by smart cards", *Journal of Cryptology*, vol. 4, no. 3, 1991, pp. 161-174
6. A. Shamir, "An efficient identification scheme based on permuted kernels" *Advances in Cryptology* (Proceedings of *Crypto* '89), Lecture Notes in Computer Science, vol. 435, Springer-Verlag, 1990, pp. 606-609
7. J. Stern, "A new identification scheme based on syndrome decoding" *Advances in Cryptology* (Proceedings of *Crypto* '93), Lecture Notes in Computer Science, vol. 773, Springer-Verlag, 1994, pp. 13-21