

Analysis of Some Natural Variants of the PKP Algorithm

Rodolphe Lampe and Jacques Patarin

University of Versailles Saint-Quentin, Laboratoire PRISM, Versailles, France

Keywords: Public-key Cryptography, Identification Scheme, Zero Knowledge, Permuted Kernel Problem.

Abstract: In 1989, (Shamir, 1989) proposed a new zero-knowledge identification scheme based on a NP-complete problem called PKP for Permuted Kernel Problem. For a given prime p , a given matrix A and a given vector V , the problem is to find a permutation π such that the permuted vector V_π verifies $A \cdot V_\pi = 0 \pmod{p}$. This scheme is still in 2011 known as one of the most efficient identification scheme based on a combinatorial problem. However, we will see in this paper that it is possible to improve this scheme significantly by combining new ideas in order to reduce the total number of computations to be performed and to improve very efficiently the security against side channel attacks using precomputations. We will obtain like this a new scheme that we have called SPKP. Moreover, if we use precomputed values in the scheme SPKP, then the prover will need to perform no computations (i.e. only selection and transmission of precomputed values). This is very interesting for security against side channel attacks because our scheme is zero-knowledge and we don't perform any computations using the key during the identification so we prove that any attacker (even using side channel attacks) being successfully identified implies that he has a solution to the NP-complete problem PKP.

1 INTRODUCTION

The articles published on PKP after Adi Shamir's article of 1989 focussed on the study of various attacks on PKP. In 1992, (Georgiades, 1992) introduced symmetric polynomials equations. The symmetric polynomial equation of degree 1 is very useful and will be used by every other attacks. The symmetric polynomial equations of bigger degrees seems to be very difficult to exploit though. The same year, (Bari-taud et al., 1992) attacked PKP using a time-memory trade-off. In 1993, (Patarin and Chauvaud, 1993) combined the previous attacks and used a few new ideas. In 1997, (Poupard, 1997) created a program to find the best attack's parameters improving the previous techniques. In 2001, (Jaulmes and Joux, 2001) used a new time-memory trade-off technique, dividing equations in 4 parts, to further improve the attack.

However, these attacks didn't break Shamir's PKP scheme: they are all exponential and PKP is still very efficient. For example, the best attack known from (Jaulmes and Joux, 2001) is in 2^{106} . Nevertheless, they show that the initial parameters of Shamir PKP(16,32) are too weak, specially with today's power computation.

Some articles (Girault, 1990), (Courtois et al., 2001) compared the PKP scheme with other identification scheme like CLE (Stern, 1989) (Stern, 1994)

and SD (Stern, 1993), PPP (Pointcheval, 1995) and MQ (Sakumoto et al., 2011). These papers show that PKP scheme is one of the most efficient in terms of computations needed and bits transferred.

In this article, we will try to describe variants of PKP that could make it even more efficient. It seems like this subject has not been studied so far. In fact, as we will see, the simplest variants don't give very good results. In this way, we could say that Shamir's PKP scheme seems quite "stable". Nevertheless we will see that, combining some simple ideas, we can create a scheme, named SPKP, that seems to be really more efficient.

For example, standard parameters PKP(37,64) needs 2^{15} multiplications of 8 bits numbers and 2^{15} additions of 8 bits numbers (for a 2^{106} security and a 2^{-30} impersonation probability) and the number of operations remains the same with a 32 bits microprocessor.

Our new version SPKP needs $2^{14.4}$ additions of 8 bits numbers (still for 2^{106} security against the best known attacks) and $2^{12.4}$ additions of 32 bits numbers if we use a 32 bits microprocessor.

On modern microprocessors, 8 bits additions and 8 bits multiplications cost about the same but it may be interesting to use additions instead of multiplications on very cheap RFID, or when the modulo p becomes large.

We will also see that our scheme SPKP is perfectly safe against SCA (side channel attacks). With PKP, we need $2^{23.4}$ bits of precomputed values to be perfectly safe against SCA, this is not realistic. With SPKP, we need 2^{17} bits of precomputed values to be perfectly safe against SCA, this a major improvement.

2 DEFINITIONS OF PKP AND THE CORRESPONDING IDENTIFICATION SCHEME

Let p be a prime, V a vector of \mathbb{Z}_p^n , A a matrix of $\mathbb{Z}_p^{m \times n}$. For each permutation $\sigma \in S_n$, we note V_σ the vector defined by $V_\sigma = (v_{\sigma(i)})_i$ and A_σ the matrix defined by $A_\sigma = (a_{i,\sigma(j)})_{i,j}$. We can notice that, for each permutation σ , we have $A_\sigma R_\sigma = AR$.

Given a prime p , a matrix A and a vector V , the Permuted Kernel Problem is to find a permutation π such that $A \cdot V_\pi = 0 \pmod p$.

This problem is NP-complete and has many advantages to be used in an identification scheme. Indeed, the following identification scheme is Zero-Knowledge (the prover doesn't reveal anything about the secret during the identification), it uses very basic operations (multiplications mod p), it is very fast and it differs from many other schemes by not depending of the factorisation or discrete log problem. Since the problem is NP-complete, it is expected to be secure against quantum computers (unlike schemes based on factorisation or discrete log).

The identification scheme is the following:

PKP 5 Rounds Identification Scheme (Shamir, 1989)

The users agree on a matrix A and a prime p . Each user chooses a random vector W in $\text{Ker}(A)$, a random permutation π and computes $V = W_{\pi^{-1}}$. The public key will be V and the secret key will be π . V has been defined such that V_π is in $\text{Ker}(A)$. Each user can now prove their identity by proving they know π :

1. The prover chooses a random vector R and a random permutation σ . The prover computes the hashed values of $(\sigma, A \cdot R)$ and $(\pi\sigma, R_\sigma)$ and sends both of them to the verifier.
2. The verifier chooses a random $c \in \mathbb{Z}/p\mathbb{Z}$ and sends it to the prover.
3. The prover sends $W = R_\sigma + cV_{\pi\sigma}$.
4. The verifier sends a bit b .
5. The prover sends σ if $b = 0$ and sends $\pi\sigma$ if $b = 1$.

In the first case, the verifier checks that the hash of $(\sigma, A_\sigma W)$ is equal to the hash of (σ, AR) .

In the second case, the verifier checks that the hash of $(\pi\sigma, W - cV_{\pi\sigma})$ is equal to the hash of $(\pi\sigma, R_\sigma)$.

An honest prover is obviously passing the test successfully: in the first case, we verify that

$$A_\sigma W = A_\sigma(R_\sigma + cV_{\pi\sigma}) = A_\sigma R_\sigma + cA_\sigma V_{\pi\sigma} = AR + cAV_\pi = AR.$$

In the second case, we verify that

$$W - cV_{\pi\sigma} = R_\sigma.$$

As shown in (Shamir, 1989), the scheme is Zero-Knowledge and the probability of success for someone who doesn't know π is less or equal to $\frac{p+1}{2p}$. For 31 iterations, the probability of success is approximately 2^{-30} .

3 PARAMETERS

The probability for a random vector to be in the Kernel of A is p^{-m} because there are m equations. The cardinal of the orbit of V under the permutations (ie the set $\{V_\sigma\}_\sigma$) is $n!$ if V has distinct coordinates. In order to have only one solution and to give the good proportion of information, we need to have $n! \approx p^m$. This is the first constraint.

Now, we have to care about security. The naive attack is to choose the first $n - m$ coordinates of the vector V_π (using the coordinates of V) and use the m equations to find the last m coordinates. The complexity of this naive attack is $\frac{n!}{m!}$. We need n to be big enough so that $\frac{n!}{m!}$ is big enough, this is the second constraint.

Later, we will use the best known attack from Joux but in the next sections, we'll only need the naive attack to understand that the simplest variants are not efficient.

Shamir proposed to use $p = 251$ (the largest prime number on 8 bits) so that we can use the scheme on small devices like 8 bits microprocessors of smart cards. This is a good choice and we'll see in section 5 if we can choose other values for p (for example for 32 or 64 bits processors, are larger values of p more efficient?). Considering the two constraints, values of n and m were proposed: $PKP(16, 32)$ (which gives a security in 2^{46} against the best known attack at present and therefore is not sufficient) and $PKP(37, 64)$ (which gives a security in 2^{106} against the best known attack).

4 PERFORMANCES

Let's count how many multiplications we need to do in the identification scheme.

The matrix A is public so everyone can use Gauss elimination so we can assume A is given by $A = [A'|I]$ where A' is a $m \times (n - m)$ matrix and I is the $m \times m$ identity matrix.

The prover has to compute $A.R$ at step 1 and $c.V_{\pi\sigma}$ at step 3. This is $m \times (n - m) + n$ multiplications of 8bits numbers and the same number of additions. For $PKP(16, 32)$, after 31 rounds, this is $2^{14.1}$ operations (half of them are multiplications). For $PKP(37, 64)$, after 31 rounds, this is 2^{16} operations. This is very fast compared to many other schemes.

In each round, we send two hashed values (128 bits for both), one vector ($8n$ bits) and one permutation ($\log_2(n!)$ bits). For $PKP(16, 32)$, after 31 rounds, this is $2^{13.9}$ bits and for $PKP(37, 64)$ this is $2^{14.8}$ bits.

5 FIRST VARIANT: DIFFERENT VALUES OF P

5.1 Why $2 \leq p < 251$ is not a Good Choice in PKP

$p = 2$: There are many issues in using $p = 2$. The first one is that we don't have $n!$ different possible solutions anymore because there are many equal coordinates.

Moreover, if two public keys V_1 and V_2 have the same number of ones and zeros, the user knowing π_1 can compute π_2 and inversely. The proof is in the appendice of (Lampe and Patarin, 2011). This limits the number of possible keys to $n + 1$ at best but most of them are weak.

$3 < p < 251$: For those values of p , we have the same problems that we had with $p = 2$. It's difficult to build public keys with different coordinates and there is a limited number a possible public keys (specially for small values of p). All the details are in the appendice of (Lampe and Patarin, 2011).

5.2 Why $p > 251$ Improve the Number of Operations Needed but not the Transmissions

Nowadays, we have access to 32 and 64bits processor so we could use those to compute modulo prime numbers of 32 or 64 bits. Therefore it is rather natural to consider PKP on computers (instead of 8 bits smartcards) with values of p of 32 or 64 bits instead of $p = 251$ (8 bits). As we will see, we will improve like this the number of computations (but not the number of transmissions).

The equation $n! \approx p^m$ tell us that, using 4 or 8 times more bits for p , divides m by 4 or 8. We know that all the attack heavily use those m equations (for example, the naive attack is in $n!/m!$).

Let see in the next array some parameters, the corresponding Joux's attack complexity and the number of multiplications mod p needed. We can notice that the extra equation of degree one of Georgiades (Georgiades, 1992) is used here in Joux attack and is not negligible when p is large. We made different arrays for the various ranges of attack's complexity (cf appendice of (Lampe and Patarin, 2011) for more details).

Table 1: Security in 2^{80} .

p	m	n	Security	Operations	Transmissions
251	24	46	2^{79}	$2^{15.1}$	$2^{14.4}$
2^{16}	8	34	2^{80}	$2^{13.8}$	$2^{14.6}$
2^{32}	4	34	2^{104}	$2^{13.2}$	$2^{15.3}$
2^{64}	2	34	2^{80}	$2^{12.5}$	$2^{16.2}$

Table 2: Security in 2^{100} .

p	m	n	Security	Operations	Transmissions
251	34	60	2^{102}	$2^{15.8}$	$2^{14.7}$
2^{16}	10	40	2^{106}	$2^{14.4}$	$2^{14.8}$
2^{32}	4	34	2^{104}	$2^{13.2}$	$2^{15.3}$
2^{64}	3	46	2^{151}	$2^{13.4}$	$2^{16.6}$

Those results tell us that using big values of p permits to reduce significantly the number of operations. As shown above, with a 2^{80} security, it needs $2^{12.5}$ operations for p with 64 bits, $m = 2, n = 34$ and needs $2^{15.1}$ operations for the standard $PKP(24, 46)$. This is 6 times faster. However, the number of transmissions are bigger with big values of p .

6 SECOND VARIANT: 3 ROUNDS PKP (INSTEAD OF 5 ROUND PKP)

We found an identification scheme with 3 rounds (instead of 5) based on the PKP problem:

PKP 3 Rounds Identification Scheme

1. The prover chooses a random vector R and a random permutation σ . He sends 4 hashed values: $h_1 = H(\sigma), h_2 = H(R_\sigma + (V_\pi)_\sigma), h_3 = H(AR), h_4 = H(R_\sigma)$.
2. The verifier sends a challenge $b = 1, 2$ or 3 .
3. • If $b = 1$, the prover reveals σ and $W_1 = R_\sigma$. The verifier verifies that $H(\sigma) = h_1, H(A_\sigma(W_1)) = h_3$ and $H(W_1) = h_4$.

- If $b = 2$, the prover reveals σ and $W_2 = R + V_\pi$. The verifier verifies that $H(\sigma) = h_1, H((W_2)_\sigma) = h_2$ and $H(AW_2) = h_3$.
- If $b = 3$, the prover reveals $\sigma\pi$ and $W_3 = R_\sigma$. The verifier verifies that $H(W_3 + V_{\sigma\pi}) = h_2$ and $H(W_3) = h_4$.

Theorem:

An honest prover will pass the test successfully all the time while a dishonest prover has, at best, a probability $\frac{2}{3}$ to pass the test successfully. Indeed, if someone can answer to the 3 questions then he has a solution for the PKP problem.

Proof: See (Lampe and Patarin, 2011). □
 Considering this probability, we need 52 rounds to have a 2^{-30} impersonation probability.

Theorem:

The PKP 3-rounds scheme is Zero-Knowledge.

Proof: See (Lampe and Patarin, 2011). □
 About performances, we need to compute AR and $R_\sigma + (V_\pi)_\sigma$ at step 1. This is $m(n - m)$ multiplications and $m(n - m) + n$ additions. About transmissions, there are 4 hash (256 bits), one vector (8n bits) and one permutation ($\log_2(n!)$ bits).

7 THIRD VARIANT: USING MORE VECTORS AND SOME SYMMETRY IN THE PKP PROBLEM

What happens if we use more than one vector V i.e. with $l \geq 2$ vectors V instead of $l = 1$? This is the exact same thing to consider V as a matrix of size $m \times l$. If we increase the number of vectors, we have to decrease the number of equations m (because $n! \approx p^{m \cdot l}$) which should increase the attack complexity. Here we will explain why $l = m$ (the maximal possible value for l) is not more secure than $l = 1$ (they actually have the exact same security, using a symmetry argument). However, in section 8 we will see that $l = \sqrt{m}$ is indeed interesting because, in this situation, there is an equal number of equations and vectors which is the fixed point of the following symmetry:

Theorem (Symmetry in PKP):

Given a prime p and an integer n , solving the PKP problem with m equations and l vectors has the exact same complexity that solving the PKP problem with l equations and m vectors.

Proof: See (Lampe and Patarin, 2011). □

As said before, decreasing m increase the attack's complexity but it slightly increase the number of operations needed as well. Using p of 8, 16, 32 or 64 bits didn't lead to interesting results.

Our New Idea. With one vector, as we have seen above, we couldn't use low prime numbers. However, with multiple vectors, we noticed that it is now possible to use low prime numbers and we will see that it can be interesting to consider $p = 2$. As long V doesn't have two equal lines, there are $n!$ possibilities for π .

8 DEFINITION OF SPKP

Our new scheme SPKP combines the three previous ideas:

- 3 rounds (instead of 5 for PKP).
- $p = 2$ (instead of $p = 251$ for Shamir's PKP recommended parameters).
- multiple vectors (typically $l = 9$ instead of $l = 1$ for PKP).

Considering multiple vectors, we studied what's the best choice for p and it seems like $p = 2$ gives the best results. Using the 3 rounds scheme gives better results as well. At step 1, to compute h_2 , we have to do $n \times l$ additions. At step 1, to compute h_3 , we have to compute $A.R$ which has $m \times l$ coordinates. For each of them, we have to do $\frac{n-m}{2}$ bits additions. Indeed, the last m coordinates of each line of A has $m - 1$ zeros and 1 one, half of the other coordinates are zeros.

At step 3, when $b = 2$, we need $\frac{nl}{2}$ bits additions to compute W_2 .

The total is $\frac{m \times l \times (n - m)}{2} + \frac{3 \times n \times l}{2}$ bits additions.

We need 52 rounds to have a 2^{-30} impersonation probability.

We need to compute the best parameters n, m and l for SPKP and we'll see the number of additions needed and conclude on its potential efficiency. To find those parameters, we have to analyze how SPKP is resisting to the best attack known. This is the object of the next section.

9 ATTACKS AND EFFICIENCY OF SPKP

SPKP is NP-complete because PKP is NP-complete and PKP is a particular case of SPKP. SPKP is Zero-Knowledge, the proof is exactly the same we gave for PKP 3 rounds. We think that all existing attacks are

less efficient on SPKP. Indeed, for given values n and p , we have the equation

$$n! = p^{m \times l},$$

where m is the number of equations and l is the number of vectors. If we use l vectors instead of 1, we have to divide the number of equations m by l and every attacks are very much dependent of the number of equations m .

The best attack on SPKP seems to be, as far as we know, similar to the best attack on PKP such as Joux's attack (Jaulmes and Joux, 2001). It'd be too long and complicate to describe the Joux's attack there. We adjusted it to SPKP by taking $p = 2$, changing n -vectors in $n \times l$ -matrix and numbers (in the D_i sets) in $1 \times l$ -vectors. In the next arrays, we present the PKP parameters and the SPKP parameters with the corresponding attack's complexity and the number of computations needed for a 2^{-30} impersonation (31 rounds for PKP and 52 rounds for SPKP). Nowadays, recent smart cards use 32 bits microprocessor so we combined bits operations together to divide the number of operations needed.

Table 3: Security in 2^{80} .

Parameters	Security	32 bits operations
<i>SPKP</i> (15, 38, 10)	2^{79}	$2^{11.9}$ additions
<i>PKP</i> (24, 46)	2^{79}	$2^{15.1}$ operations

Table 4: Security in 2^{100} .

Parameters	Security	32 bits operations
<i>SPKP</i> (15, 42, 11)	2^{98}	$2^{12.2}$ additions
<i>PKP</i> (34, 60)	2^{102}	$2^{15.8}$ operations

This results shows that SPKP needs less and simpler operations so it seems to be more efficient than the original PKP scheme. For a 2^{100} security, SPKP needs 12 times less operations than PKP and all operations are additions compared to PKP using multiplications and additions.

Now, we will compare SPKP with other combinatorial schemes. In this array, we'll show bits operations for SPKP instead of combining them and we give the number field used as well. We used parameters for a 2^{-80} security and 2^{-30} impersonation probability.

This shows that SPKP is the scheme using the less operations (if we combine bits operations together).

Moreover, using precomputations, we can make this scheme even more efficient as we will see in the next section.

Table 5: Number of operations for a 2^{80} security and 2^{-30} impersonation probability.

CLE	SD	PP
$2^{15}/\mathbb{F}_{257}$	$2^{18}/\mathbb{F}_{256}$	$2^{21}/\mathbb{F}_{127}$
MQ	PKP	SPKP
$2^{26}/\mathbb{F}_2$	$2^{15}/\mathbb{F}_{251}$	$2^{17}/\mathbb{F}_2$

10 PRECOMPUTATIONS WITH SPKP OR OUR PKP 3-ROUNDS

In the original PKP scheme (5 rounds, presented at section 2), the prover has to compute $W = R_\sigma + cV_{\pi\sigma}$ where c is a value with p possibilities ($p = 251$ typically) chosen by the verifier then he will face one of the two challenges. Therefore, for the 31 rounds, if the prover want to precompute all the possible answers to the prover questions in advance, he has to prepare $62p$ answers, approximately 15000 values. This is not very realistic.

However, in our scheme (PKP 3 rounds, section 6), the verifier will face one of the three challenges at each round. Therefore, for the 52 rounds, the prover has to prepare 104 answers for one identification. This is realistic if we use devices with enough memory.

Therefore, we see that all the prover's computations can be precomputed so that the prover doesn't have to compute anything during the identification. We can create a smart card which contains only the datas $R_i, h_1, \dots, W_1, \dots$. The prover uses this card for identification, his only need is to send and receive datas from the verifier.

Precomputation with other Combinatorial Schemes. This property is possible on every other schemes as long the number of possible challenges is limited. This is why our scheme is efficient for precomputation: there are only 3 possible challenges while the standard PKP has 502 possible challenges. The memory needed for one identification is the number of bits for transmission times the number of challenges.

We see that SPKP is one of the most efficient scheme if we want to use precomputation. In 2004, Samsung realised a smart card with 256kbytes of EEPROM which permits to save datas for about 16 identifications using SPKP.

In the next few years, the memory size augmentation could permit to create smart cards with more than a thousand of identifications saved.

Table 6: Memory needed for one identification using pre-computations (with 2^{80} security and 2^{-30} impersonation probability).

MQ	SPKP	SD	PP	PKP	CLE
$2^{16.4}$	$2^{16.9}$	$2^{17.5}$	$2^{18.6}$	$2^{23.4}$	$2^{23.8}$

11 SECURITY AGAINST SIDE CHANNEL ATTACKS

Since a few years, very efficient physical attacks have been discovered on smart cards and microprocessors, for example: timing attacks, power attacks (SPA, DPA (Kocher et al., 1999)), fault attacks (DFA), ... Generally some ways to fix those problems was found by the scientific community, but sometimes it is really difficult to design secure hardwares against some physical attacks, and it is expected that new attacks could be found. A lot of those attacks use the fact that the microprocessor has to manipulate secret datas. In the variants of PKP 3 rounds and SPKP that we presented, it is possible to precompute everything. That way, no secret datas are manipulated by the microprocessor which greatly simplifies the security against physical attacks.

The precomputed datas have to be encrypted or saved in protected areas because, even if a single data doesn't reveal anything about the secret, the combination of some datas could reveal the secret. The microprocessor needs to be able to transmit one of those values but not all of them and eventually decrypt this value with a key K. The other values have to be encrypted with different keys or saved in protected areas to assure a good security. In fact, it seems to be much easier to secure such a scheme from physical attacks than to secure the traditional schemes that manipulate a secret data s in the computation of an identification against physical attacks (where s needs to be still secret after the identification). That's why we think those schemes present a real interest for the security against physical attacks.

We compared SPKP with other schemes that use precomputations like GPS (Girault et al., 2006) or Lamport (Lamport, 1981) and his variants. We give more details in appendice of (Lampe and Patarin, 2011).

ACKNOWLEDGEMENTS

The PhD of Rodolphe LAMPE is financially supported by the Direction Générale des Armées (DGA).

REFERENCES

Baritaud, T., Campana, M., Chauvaud, P., and Gilbert, H. (1992). On the security of the permuted kernel identification scheme. In *CRYPTO*, pages 305–311.

Courtois, N., Finiasz, M., and Sendrier, N. (2001). How to achieve a mceliece-based digital signature scheme. In *ASIACRYPT*, pages 157–174.

Georgiades, J. (1992). Some remarks on the security of the identification scheme based on permuted kernels. *J. Cryptology*, 5(2):133–137.

Girault, M. (1990). A survey of identification schemes. In *EUROCODE*, pages 168–179.

Girault, M., Poupard, G., and Stern, J. (2006). On the fly authentication and signature schemes based on groups of unknown order. *J. Cryptology*, 19(4):463–487.

Jaulmes, É. and Joux, A. (2001). Cryptanalysis of pkp: A new approach. In *Public Key Cryptography*, pages 165–172.

Kocher, P. C., Jaffe, J., and Jun, B. (1999). Differential power analysis. In *CRYPTO*, pages 388–397.

Lampe, R. and Patarin, J. (2011). Analysis of some natural variants of the pkp algorithm. *IACR Cryptology ePrint Archive*, 2011:686.

Lamport, L. (1981). Password authentication with insecure communication. *Commun. ACM*, 24(11):770–772.

Patarin, J. and Chauvaud, P. (1993). Improved algorithms for the permuted kernel problem. In *CRYPTO*, pages 391–402.

Pointcheval, D. (1995). A new identification scheme based on the perceptrons problem. In *EUROCRYPT*, pages 319–328.

Poupard, G. (1997). A realistic security analysis of identification scheme based on combinatorial problems. In *European transactions on telecommunications*, pages 471–480.

Sakumoto, K., Shirai, T., and Hiwatari, H. (2011). Public-key identification schemes based on multivariate quadratic polynomials. In *CRYPTO*, pages 706–723.

Shamir, A. (1989). An efficient identification scheme based on permuted kernels (extended abstract). In *CRYPTO*, pages 606–609.

Stern, J. (1989). An alternative to the fiat-shamir protocol. In *EUROCRYPT*, pages 173–180.

Stern, J. (1993). A new identification scheme based on syndrome decoding. In *CRYPTO*, pages 13–21.

Stern, J. (1994). Designing identification schemes with keys of short size. In *CRYPTO*, pages 164–173.