PKP-Based Signature Scheme

Ward Beullens[1], Jean-Charles Faugère[2], Eliane Koussa[3], Gilles Macario-Rat[4], Jacques Patarin[5], and Ludovic Perret[2]

[1] imec-COSIC, KU Leuven ward.beullens@esat.kuleuven.be
[2] INRIA and Sorbonne Universities/UPMC Uni Paris 6
jean-charles.faugere@inria.fr, ludovic.perret@lip6.fr
[3] Versailles Laboratory of Mathematics, UVSQ EJKoussa@outlook.com
[4] Orange gilles.macariorat@orange.com
[5] Versailles Laboratory of Mathematics, UVSQ, CNRS, University of Paris-Saclay, jpatarin@club-internet.fr

Abstract. In this document, we introduce PKP-DSS: a Digital Signature Scheme based on the Permuted Kernel Problem(PKP) [23]. PKP is a simple NP-hard [10] combinatorial problem that consists of finding a kernel for a publicly known matrix, such that the kernel vector is a permutation of a publicly known vector. This problem was used to develop an Identification Scheme (IDS) which has a very efficient implementation on low-cost smart cards. From this zero-knowledge identification scheme, we derive PKP-DSS with the traditional Fiat-Shamir transform [9]. Thus, PKP-DSS has a security that can be provably reduced, in the *(classical) random oracle model*, to the hardness of random instances of PKP (or, if wanted, to any specific family of PKP instances). We propose parameter sets following the thorough analysis of the State-of-the-art attacks on PKP presented in [**17**]. We show that PKP-DSS is competitive with other signatures derived from Zero-Knowledge identification schemes. In particular, PKP-DSS-128 gives a signature size of approximately 20 KBytes for 128 bits of classical security, which is approximately 30% smaller than MQDSS. Moreover, our proof-of-concept implementation shows that PKP-DSS-128 is an order of magnitude faster than MQDSS which in its turn is faster than Picnic2, SPHINCS,...
Since the PKP is NP-hard and since there are no known quantum attacks for solving PKP significantly better than classical attacks, we believe that our scheme is post-quantum secure.

Keywords: public-key cryptography, Fiat-Shamir, post-quantum cryptography, 5-pass identification scheme, Public-key Signature, Permuted Kernel Problem.

1 Introduction

The construction of large quantum computers would break most public-key cryptographic schemes in use today because they rely on the discrete logarithm

problem or the integer factorization problem. Even though it isn't clear when large scale quantum computation would be feasible, it is important to anticipate quantum computing and design new public key cryptosystems that are resistant to quantum attacks. Therefore, there currently is a large research effort to develop new post-quantum secure schemes, and a Post-Quantum Cryptography standardization process has been initiated by the American National Institute of Standards and Technology (`https://www.nist.gov/`). Because of this, there has been renewed interest in constructing signature schemes by applying the Fiat-Shamir transform [9] to Zero-Knowledge Identification Schemes. In particular, we are interested in post-quantum cryptographic schemes whose security relies on the quantum hardness of some NP-Hard problem [2]. One of those problems is the Permuted Kernel Problem: the problem of finding a permutation of a known vector such that the resulting vector is in the kernel of a given matrix. This is a classical NP-Hard combinatorial problem which requires only simple operations such as basic linear algebra and permuting the entries of a vector. For quite some time, no new attacks on PKP have been discovered, which makes it possible to confidently estimate the concrete hardness of the problem.

In 1989, Shamir [23] introduced a five-pass ZK-Identification scheme, based on the PKP. This work uses the Fiat-Shamir transform [9] on this identification scheme to develop a signature scheme that is provably secure in the Random Oracle Model (ROM). However, since our goal is to have a post-quantum scheme, we should also consider attackers in the Quantum Random Oracle Model (QROM). The security of the Fiat-Shamir transform in the QROM has been studied in [27,26,25], where the authors of [27,25] explain that the Fiat-Shamir transform might not be secure against quantum computers. Thus, new techniques with extra properties (such as "lossy IDS") were developed to obtain a quantum-secure transform. However, more recently, a number of works have proven the Fiat-Shamir construction secure in the QROM[26,13] under very mild conditions. So far, none of these works apply to five-round protocols (which is the kind of protocol we are considering in this work), but it is conceivable that the results can be generalized to five-pass protocols, including ours. We consider this an important open problem in post-quantum cryptography.

Previous work and State-of-the-art. Since quantum computers are expected not to be capable of solving *NP*-Hard problems in sub-exponential time (in worst case), Zero-knowledge Identification schemes based on such problems are interesting candidates for Post-Quantum Cryptography. The Fiat-Shamir transform [9] is a technique that can convert such a zero-knowledge authentication scheme into a signature scheme. This approach was taken by Chen et al. [7], who applied the Fiat-Shamir transform to a 5-pass identification scheme of Sakumoto et al. [22]. This identification scheme relies on the hardness of the (*NP*-Hard) problem of finding a solution to a set of multivariate quadratic equations. Chen et al. proved that, in the random oracle model, applying the Fiat-Shamir transform to this 5-pass identification scheme results in a secure

signature scheme. A concrete parameter choice and an efficient implementation of this signature scheme (which is called MQDSS) were developed, and this was one of the submissions to the NIST PQC standardization project. At a security level of 128 bits, the MQDSS scheme comes with a public key of 46 Bytes, a secret key of 16 Bytes and a signature size of approximately 28 Kilobytes.

A different line of work resulted in the Picnic signature scheme. Chase et al. [6] constructed this digital signature scheme by applying the Fiat-Shamir transform to an identification scheme whose security relies purely on symmetric primitives. At the 128-bit security level Picnic has a public key of 32 Bytes, a secret key of 16 Bytes and signatures of approximately 33 Kilobytes. There is a second version of this signature scheme, where the signatures are only 13.5 Kilobytes, but Picnic2 is 45 times slower than the original Picnic for signing and 25 times slower for verification.

Main results. The main contribution of this paper is to present PKP-DSS, a new post-quantum secure signature scheme. Similar to the approaches cited above, we use the Fiat-Shamir transform to construct a signature scheme from the 5-pass PKP identification scheme by Shamir [23]. Following the complexity analysis of the PKP [17], we choose secure parameter sets of the signature scheme for 128/192/256 of classical security level. To date, there are no known quantum algorithms for solving PKP (other than combining Grover search with the classical algorithms), so we claim that our signatures achieve the NIST security levels I/III and V respectively. However, we recognize that the (quantum) hardness of PKP deserves more research, and we hope that this work will inspire researchers to investigate this topic further.

We have developed a constant-time C implementation of the new signature scheme. By constant-time we mean that the running time and the memory access pattern of the implementation are independent of secret material, therefore blocking attacks from timing side channels. The resulting signature scheme compares well with MQDSS and Picnic/Picnic2. Our scheme is much faster than MQDSS and Picnic/Picnic2 in terms of signing and verification, we have small public and private keys, and the signature sizes of our scheme are comparable to those of MQDSS and Picnic2. This makes our signature scheme based on PKP competitive with state of the art post-quantum signature schemes.

2 Preliminaries

2.1 The Permuted Kernel Problem (PKP)

The Permuted Kernel Problem (PKP) [23,10] is the problem on which the security of PKP-DSS is based. PKP is a linear algebra problem which asks to find

a kernel vector of a given matrix under a vector-entries constraint. It's a generalization of the Partition problem [10, pg.224]. More precisely, it is defined as follows:

Definition 1 (Permuted Kernel Problem). *Given a finite field $\mathbb{F}_p$, a matrix $\mathbf{A} \in \mathbb{F}_p^{m \times n}$ and a n-vector $\mathbf{v} \in \mathbb{F}_p^n$, find a permutation $\pi \in S_n$ such that $\mathbf{A}\mathbf{v}_\pi = 0$, where $\mathbf{v}_\pi = (v_{\pi(1)}, \cdots, v_{\pi(n)})$*

A reduction of the 3-Partition problem proves PKP to be NP-Hard [10]. Moreover, solving random instances of PKP seems hard in practice. In fact, this is the fundamental design assumption of PKP-DSS. The hardness of PKP comes from, on the one hand, the big number of permutations, on the other hand, from the small number of possible permutations that satisfy the kernel equations. Note that, to make the problem more difficult, the n-vector $\mathbf{v}$ should have distinct coordinates. Otherwise if there are repeated entries, the space of permutations of $\mathbf{v}$ gets smaller. In the next section, we give the best known algorithm to solve the PKP problem.

Best known algorithms for solving PKP The implementation's efficiency of the first IDS, proposed by Shamir [23], based on PKP problem has led to several solving tools. There are various attacks for PKP, which are all exponential. We will not describe them here. Instead, we refer to [17] for further details. To estimate the concrete security of PKP, the authors of [17] review and compare the efficiency of the best known attacks in terms of the number of operations performed, for different finite fields. They bring together the Patarin-Chauvaud attack [21] and Poupard's algorithm [18] to provide an accurate program. The paper gives security estimates that we used to pick secure parameters sets for the Permuted Kernel Problem.

2.2 Commitment schemes

In our protocol, we use a commitment scheme $\mathsf{Com} : \{0,1\}^\lambda \times \{0,1\}^\star \to \{0,1\}^{2\lambda}$, that takes as input λ uniformly random bits bits, where λ is the security parameter, and a message $m \in \{0,1\}^\star$ and outputs a 2λ bit long commitment $\mathsf{Com}(\mathsf{bits}, m)$. In the description of our protocols, we often do not explicitly mention the commitment randomness. We write $\mathsf{S} \leftarrow \mathsf{Com}(m)$, to denote the process of picking a uniformly random bit string r, and setting $\mathsf{C} \leftarrow \mathsf{Com}(\mathsf{r}, m)$. Similarly, when we write check $\mathsf{C} = \mathsf{Com}(m)$, we actually mean that the prover communicates r to the verifier, and that the verifier checks if $\mathsf{C} = \mathsf{Com}(\mathsf{r}, m)$.

We assume that Com is computationally binding, which means that no computationally bounded adversary can produce a $\mathsf{r}, \mathsf{r}', m, m'$ with $m \neq m'$ such that $\mathsf{Com}(\mathsf{r}, m) = \mathsf{Com}(\mathsf{r}', m')$. We also assume that Com is computationally hiding, which means that for every pair of messages m, m', no computationally bounded adversary can distinguish the distributions of $\mathsf{Com}(m)$ and $\mathsf{Com}(m')$.

2.3 $2q$-Identification schemes and $2q$-extractors

In this paper, we will describe a so-called $2q$-Identification Scheme [24]. This is a 5-round identification scheme, where the first challenge is drawn uniformly at random from a challenge space of size q, and the second challenge is a random bit. Therefore, a transcript of an execution of a $2q$-protocol looks like $(\mathsf{com}, c, \mathsf{rsp}_1, b, \mathsf{rsp}_2)$. We now state the properties of a $2q$-protocol more formally:

Definition 2 ($2q$-Identification scheme,[24]). *A $2q$-Identification scheme is a canonical five-pass identification scheme $(\mathsf{KeyGen}, \mathcal{P}, \mathcal{V})$ with challenge spaces $\mathcal{C}h_1$ and $\mathcal{C}h_2$ for which it holds that $|\mathcal{C}h_1| = q$ and $|\mathcal{C}h_2| = 2$. Moreover, we require that the probability that the commitment com take a certain value is a negligible function of the security parameter.*

Definition 3 (Completeness). *An Identification scheme $(\mathsf{KeyGen}, \mathcal{P}, \mathcal{V})$ is called complete if when both parties follow the protocol honestly, the verifier accepts with probability 1. That is, we have*

$$\Pr\begin{bmatrix}(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ \langle \mathcal{P}(\mathsf{sk}), \mathcal{V}(\mathsf{pk})\rangle = 1\end{bmatrix} = 1\,,$$

where $\langle \mathcal{P}(\mathsf{sk}), \mathcal{V}(\mathsf{pk})\rangle$ stands for the common execution of the protocol between $\mathcal{P}$ with input sk and $\mathcal{V}$ with input pk.

Definition 4 (Soundness with soundness error κ). *An identification scheme $(\mathsf{KeyGen}, \mathcal{P}, \mathcal{V})$ is called Sound with soundness error κ, if for any probabilistic polynomial time adversary $\mathcal{A}$, we have*

$$\Pr\begin{bmatrix}(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ \langle \mathcal{A}(1^\lambda, \mathsf{pk}), \mathcal{V}(\mathsf{pk})\rangle = 1\end{bmatrix} \leq \kappa + \epsilon(\lambda)\,,$$

for some negligible function $\epsilon(\lambda)$.

Definition 5 ((computational) Honest-Verifier Zero-Knowledge). *We say an identification scheme $(\mathsf{KeyGen}, \mathcal{P}, \mathcal{V})$ is HVZK if there exists a probabilistic polynomial time Simulator $\mathcal{S}$ that outputs transcripts that are computationally indistinguishable from transcripts of honest executions of the protocol.*

Finally, we define the notion of a $2q$-extractor, which is an algorithm that can extract the secret key from 4 transcripts that satisfy some properties. This is useful because when there exists a $2q$-extractor for a $2q$-Identification scheme, this implies that the identification scheme has soundness with knowledge error at most $\frac{q+1}{2q}$. Moreover, this implies that applying the Fiat-Shamir transform to the identification scheme results in a secure signature scheme.

Definition 6 ($2q$-extractability). *We say a $2q$-identification scheme* $(\mathsf{KeyGen}, \mathcal{P}, \mathcal{V})$ *has $2q$-extractability, if there exists a polynomial-time algorithm that given four transcripts* $(\mathsf{com}, c^{(i)}, \mathsf{rsp}_1^{(i)}, b^{(i)}, \mathsf{rsp}_2^{(i)})$ *for i from 1 to 4, such that*

$$c^{(1)} = c^{(2)} \neq c^{(3)} = c^{(4)}$$
$$\mathsf{rsp}_1^{(1)} = \mathsf{rsp}_1^{(2)} \quad \mathsf{rsp}_1^{(3)} = \mathsf{rsp}_1^{(4)}$$
$$b^{(1)} = b^{(3)} \neq b^{(2)} = b^{(4)}$$

can efficiently extract a secret key.

Theorem 1 ([24], Theorem 3.1 and Theorem 4.3). *A $2q$-extractable $2q$-identification scheme is sound with knowledge error at most $\frac{q+1}{2q}$. Moreover, applying the Fiat-Shamir transform to such an identification scheme results in a EUF-CMA secure signature scheme in the Random Oracle Model.*

3 Identification scheme (IDS) based on PKP

In this section, we first present the 5-pass Zero-Knowledge Identification Scheme (ZK-IDS) based on the computational hardness of PKP [23,19], noted here PKP-IDS. Then, we introduce our optimized version of PKP-IDS and we prove that the optimized identification scheme is secure.

3.1 The original 5-pass PKP IDS

In this section, we present the original PKP-IDS [23,19], and we propose its slightly modified version. It consists of three probabilistic polynomial time algorithms $IDS = (\textsc{KeyGen}, \mathcal{P}, \mathcal{V})$ which we will describe now.

Generation of the public key and secret key in PKP-IDS. The users first agree on a prime number p, and on n, m, the dimensions of the matrix $\mathbf{A}$. The public-key in PKP-IDS is an instance of PKP, a solution to this instance is the secret-key. Thus, the prover picks a (right) kernel-vector $\mathbf{w} \in \mathrm{Ker(A)}$, then randomly generates a secret permutation of n elements $\mathsf{sk} = \pi$ and finishes by computing $\mathbf{v} = \mathbf{w}_{\pi^{-1}}$. We summarize the key generation algorithm in Alg. 1.

5-pass identification protocol: Prover $\mathcal{P}$ and Verifier $\mathcal{V}$.
The prover and verifier are interactive algorithms that realize the identification protocol in 5 passes. The 5 passes consist of one commitment and two responses transmitted from the prover to the verifier and two challenges transmitted from the verifier to the prover. The identification protocol is summarized in Alg. 2.

Algorithm 1 KeyGen(n,m,p)

$\mathbf{A} \xleftarrow{\$} \mathbb{F}_p^{m \times n}$
$\mathbf{w} \xleftarrow{\$} \mathrm{Ker}(\mathbf{A})$
$\pi \xleftarrow{\$} S_n$
$\mathbf{v} \leftarrow \mathbf{w}_{\pi^{-1}}$
Return $(\mathsf{pk} = (\mathbf{A}, \mathbf{v}), \mathsf{sk} = \pi)$

Algorithm 2 The original 5-pass PKP identification protocol

$\mathcal{P}(\mathsf{sk}, \mathsf{pk})$		$\mathcal{V}(\mathsf{pk})$
$\sigma \xleftarrow{\$} S_n$		
$\mathbf{r} \xleftarrow{\$} \mathbb{F}_p^n$		
$\mathsf{C}_0 \leftarrow \mathrm{Com}(\sigma, \mathbf{Ar})$		
$\mathsf{C}_1 \leftarrow \mathrm{Com}(\pi\sigma, \mathbf{r}_\sigma)$		
	$\xrightarrow{\mathsf{C}_0, \mathsf{C}_1}$	
		$c \xleftarrow{\$} \mathbb{F}_p$
	$\xleftarrow{c}$	
$\mathbf{z} \leftarrow \mathbf{r}_\sigma + c\mathbf{v}_{\pi\sigma}$		
	$\xrightarrow{\mathbf{z}}$	
		$b \xleftarrow{\$} \{0, 1\}$
	$\xleftarrow{b}$	
if $b = 0$ **then**		
$\quad \mathsf{rsp} \leftarrow \sigma$		
else		
$\quad \mathsf{rsp} \leftarrow \pi\sigma$		
end if		
	$\xrightarrow{\mathsf{rsp}}$	
		if $b = 0$ **then**
		$\quad$ accept if $\mathsf{C}_0 = \mathrm{Com}(\sigma, A\mathbf{z}_{\sigma^{-1}})$
		else
		$\quad$ accept if $\mathsf{C}_1 = \mathrm{Com}(\pi\sigma, \mathbf{z} - c\mathbf{v}_{\pi\sigma})$
		end if

Theorem 2. *PKP-IDS is complete, moreover, if the used commitment scheme is computationally hiding then PKP-IDS is computationally honest-verifier zero-knowledge and if the commitment scheme is computationally binding, then PKP-IDS is sound with soundness error* $\kappa = \frac{p+1}{2p}$.

Proof. We refer to [23] for the complete proof.

In such ZK-IDS, it is usually possible to cheat if a cheater can correctly guess some challenges, so there is a nonzero probability (called the soundness error) that the verifier accepts the proof, even though the prover does not know the witness. In the case of PKP-IDS, this soundness error is $\frac{p+1}{2p}$. Thus, it is necessary to repeat the protocol several times to reduce the probability of fraud. Sequentially repeating the zero-knowledge proof N times results in an Identification scheme with knowledge error $\kappa_{\text{repeated}} = \kappa^N$, hence it suffices to repeat the protocol $\lceil \lambda / \log_2(\frac{2p}{p+1}) \rceil$ times to get a soundness error $\kappa \leq 2^{-\lambda}$. The systems are constructed such that executing the protocol does not reveal any secrets (Zero-knowledge).

3.2 The modified version of PKP-IDS

We now describe several optimizations to reduce the communication cost of the identification scheme, as well as the computational cost of the algorithms. We will start by explaining a few standard optimizations that are common for identification protocols based on zero-knowledge proofs. Then, we will explain some novel optimizations that apply to the specific context of PKP-IDS.

Hashing the commitments. In the commitment phase of the protocol, instead of transmitting all the $2N$ commitments $\mathsf{C}_0^{(1)}, \mathsf{C}_1^{(1)}, \cdots, \mathsf{C}_0^{(N)}, \mathsf{C}_1^{(N)}$ the prover can just hash all these commitments together with a collision resistant hash function $\mathcal{H}$ and only transmit the hash $h = \mathcal{H}(\mathsf{C}_0^{(1)}, \cdots, \mathsf{C}_1^{(N)})$. Then, the prover includes the N commitments $\mathsf{C}_{1-b_i}^{(i)}$ in the second response. Since the verifier can reconstruct the $\mathsf{C}_{b_i}^{i}$ himself, he now has all the $2N$ commitments, so he can hash them together and check if their hash matched h. With this optimization, we reduce the number of communicated commitments from $2N$ to N, at the cost of transmitting a single hash value.

Use seeds and PRG. Instead of directly choosing the permutation σ at random, we can instead choose a random seed of λ bits and use a PRG to expand this seed into a permutation σ. This way, instead of transmitting σ, we can just transmit the λ-bit seed. This reduces the communication cost per permutation from $\log_2(n!)$ bits to just λ bits. For example for 128-bits of security, we have $n = 69$, so the communication cost per permutation drops from $\log_2(69!) \approx 327$ bits to just 128 bits.

Matrix A in systematic form. Now we get to the PKP-IDS-specific optimizations. With high probability, we can perform elementary row operations on $\mathbf{A}$ to put it in the form $\left(I_m \ \mathbf{A}'\right)$, for some $(n-m)$-by-m matrix $\mathbf{A}'$. Since row operations do not affect the right kernel of A, we can just choose the matrix $\mathbf{A}$ of this form during key generation, without affecting the security of the scheme. This makes the protocol more efficient because multiplying by a matrix of this form requires only $(n-m)*m$ multiplications instead of $n*m$ multiplications for a general matrix multiplication.

Optimizing key generation. It is of course not very efficient to include in the public key the matrix $\mathbf{A} = \left[\mathbf{c_i^A},\ \mathbf{i} \in \{\mathbf{1}, \cdots, \mathbf{n}\}\right]$, where c_i^A is the i-th column of $\mathcal{A}$. The first idea is to just pick a random seed, and use a PRG to expand this seed to obtain the matrix $\mathbf{A}$. The public key then consists of a random seed, and the vector $\mathbf{v}$ of length n. However, we can do slightly better than this. We can use a seed to generate $\mathbf{A^*}$ which is formed by the first $n-1$ columns $c_1^A, \cdots, c_{n-1}^A$ of $\mathbf{A}$ and the vector v. Then we pick a random permutation π, and we solve for the last column c_n^A of $\mathbf{A}$ such that $\mathbf{v}_\pi$ is in the right kernel of $\mathbf{A}$. Now the public key only consists of a seed and a vector of length m (instead of a vector of length n). Another important advantage of this approach is that we do not need to do Gaussian elimination this way (and if fact this was the motivation behind this optimization). The optimized key generation procedure is given in Alg. 3.

Algorithm 3 KeyGen(n,m,p)

$sk.seed \leftarrow$ Randomly sample λ *bits*
$(seed_\pi, pk.seed) \leftarrow \mathsf{PRG}_0(sk.seed)$
$\pi \leftarrow \mathsf{PRG}_1(seed_\pi)$
$(\mathbf{A}^*, \mathbf{v}) \leftarrow \mathsf{PRG}_2(pk.seed)$
Compute $\mathbf{c_n^A}$ from A^* and $\mathbf{v}_\pi$
$\mathsf{sk} \leftarrow \mathbf{sk.seed}$
$\mathsf{pk} \leftarrow (\mathbf{pk.seed}, \mathbf{c_n^A})$
Return $(\mathsf{pk}, \mathsf{sk})$

Sending seeds instead of permutations. Because of the second optimization, we can send a λ-bit seed instead of σ, if the challenge bit $b = 0$. However, in the case $b = 1$, we still need to send the permutation $\pi\sigma$, because we cannot generate both σ and $\pi\sigma$ with a PRG. However, this problem can be solved. We can generate $\mathbf{r}_\sigma$ with a PRG, and then we can send this seed instead of $\pi\sigma$. This seed can be used to compute $\pi\sigma$, because if the verifier knows $\mathbf{z}$ and $\mathbf{r}_\sigma$, then he can compute $\mathbf{z} - \mathbf{r}_\sigma = c\mathbf{v}_{\pi\sigma}$. And since $\mathbf{v}$ and c are known, it is easy to recover $\pi\sigma$ from $c\mathbf{v}_{\pi\sigma}$ (we choose the parameters such that the entries of v are all distinct, so there is a unique permutation that maps $\mathbf{v}$ to $\mathbf{v}_{\pi\sigma}$). Moreover, sending the seed

for $\mathbf{r}_\sigma$ does not reveal more information than sending $\pi\sigma$, because given $\mathbf{z}$ and $\pi\sigma$ it is trivial to compute $\mathbf{r}_\sigma$, so this optimization does not affect the security of the scheme. However, there is a problem: If $c = 0$, then the $c\mathbf{v}_{\pi\sigma} = 0$, and so the verifier cannot recover $\pi\sigma$. To solve this problem we just restrict the challenge space to $\mathbb{F}_p \setminus \{0\}$. This increases the soundness error to $\frac{p}{2p-2}$ (instead of $\frac{p+1}{2p}$), but this is not a big problem. An important advantage of this optimization is that the signature size is now constant. Without this optimization, a response to the challenge $b = 0$ would be smaller than a response to $b = 1$. But with the optimization, the second response is always a random seed, regardless of the value of b. We summarize the one round of the optimized IDS modified version in Algorithm 4.

Algorithm 4 The modified 5-pass of PKP-IDS

$\mathcal{P}(\mathsf{sk}, \mathsf{pk})$		$\mathcal{V}(\mathsf{pk})$
$\mathsf{seed}_0, \mathsf{seed}_1 \xleftarrow{\$} \{0,1\}^\lambda$ $\sigma \leftarrow \mathsf{PRG}_1(seed_\sigma)$ $\mathbf{r}_\sigma \leftarrow \mathsf{PRG}_2(\mathbf{r}_\sigma.seed)$ $\mathsf{C}_0 \leftarrow \mathrm{Com}(\sigma, \mathbf{Ar})$ $\mathsf{C}_1 \leftarrow \mathrm{Com}(\pi\sigma, \mathbf{r}_\sigma)$		
	$\xrightarrow{\mathsf{C}_0, \mathsf{C}_1}$	
		$c \xleftarrow{\$} \mathbb{F}_p \setminus \{0\}$
	$\xleftarrow{c}$	
$\mathbf{z} \leftarrow \mathbf{r}_\sigma + c\mathbf{v}_{\pi\sigma}$		
	$\xrightarrow{\mathbf{z}}$	
		$b \xleftarrow{\$} \{0,1\}$
	$\xleftarrow{b}$	
$\mathsf{rsp} \leftarrow \mathsf{seed}_b$		
	$\xrightarrow{\mathsf{rsp}}$	

		if $b = 0$ **then**
		$\sigma \leftarrow \mathsf{PRG}_1(\mathsf{rsp})$
		accept if $\mathsf{C}_0 = \mathrm{Com}(\sigma, A\mathbf{z}_{\sigma^{-1}})$
		else
		$\mathbf{r}_\sigma \leftarrow \mathsf{PRG}_2(\mathsf{rsp})$
		if $\mathbf{z} - \mathbf{r}_\sigma$ is not a permutation of $c\mathbf{v}$
		then
		Return reject
		else
		Let $\rho \in S_n$ such that $c\mathbf{v}_\rho = \mathbf{z} - \mathbf{r}_\sigma$.
		end if
		accept if $\mathsf{C}_1 = \mathrm{Com}(\rho, \mathbf{r}_\sigma)$
		end if

3.3 Security proof of the optimized Scheme

Theorem 3. – *The modified version of PKP-IDS is complete.*
– *If the commitment scheme is computationally binding, then the scheme is sound with soundness error* $\kappa = \frac{p}{2p-2}$.
– *If the used commitment scheme is computationally hiding and the output of* PRG_1 *and* PRG_2 *is indistinguishable from uniform randomness, then the scheme is computationally honest-verifier zero-knowledge.*

Proof. **Completeness.** In the case $b = 0$, if the prover acts honestly, then the commitment check will succeed if $\mathbf{Ar} = \mathbf{Az}_{\sigma}^{-1} = \mathbf{A}(\mathbf{r} + \mathbf{v}_{\pi\sigma\sigma^{-1}})$, which holds if and only if $\mathbf{Av}_{\pi} = 0$. Therefore, if π is a solution to the PKP problem, then the verifier will accept the transcript. In an honest execution with $b = 1$ the verifier will always accept, regardless of whether π was a solution to the PKP problem or not.

Soundness. First, we prove that the scheme has a $q2$-extractor [24]. That is, we show that, given four accepted transcripts $(\mathsf{C}_0, \mathsf{C}_1, c^{(i)}, \mathbf{z}^{(i)}, b^{(i)}, \mathsf{rsp}^{(i)})$ for i from 1 to 4, such that

$$\begin{aligned} c^{(1)} = c^{(2)} \neq c^{(3)} = c^{(4)} \\ \mathbf{z}^{(1)} = \mathbf{z}^{(2)} \quad \mathbf{z}^{(3)} = \mathbf{z}^{(4)} \\ b^{(1)} = b^{(3)} \neq b^{(2)} = b^{(4)} \end{aligned}$$

one can efficiently extract a solution for the PKP problem.

By relabeling the transcripts if necessary, we can assume that $b^{(1)} = b^{(3)} = 0$ and $b^{(2)} = b^{(4)} = 1$. Let us first look at transcripts 1 and 3. Let $\sigma = \mathsf{PRG}_1(\mathsf{rsp}^{(1)})$ and $\sigma' = \mathsf{PRG}_1(\mathsf{rsp}^{(3)})$, and let $\mathbf{x} = \mathbf{Az}^{(1)}_{\sigma^{-1}}$ and $\mathbf{x}' = \mathbf{Az}^{(3)}_{\sigma'^{-1}}$. Then, because both transcripts are accepted, we have

$$\mathsf{C}_0 = \mathsf{Com}(\sigma, \mathbf{x}) = \mathsf{Com}(\sigma', \mathbf{x}').$$

Therefore, the computationally binding property of Com implies that with overwhelming probability we have $\sigma = \sigma'$ and $\mathbf{x} = \mathbf{x}'$.

Now, lets look at transcripts 2 and 4. Let $\mathbf{y} = \mathsf{PRG}_2(\mathsf{rsp}^{(2)})$ and $\mathbf{y}' = \mathsf{PRG}_2(\mathsf{rsp}^{(4)})$. Since both transcripts are accepted, we know that $\mathbf{z}^{(2)} - \mathbf{y}$ and $\mathbf{z}^{(4)} - \mathbf{y}'$ are permutations of $c^{(2)}\mathbf{v}$ and $c^{(4)}\mathbf{v}$ respectively. Let ρ and ρ' be the permutations such that $c^{(2)}\mathbf{v}_{\rho} = \mathbf{z}^{(2)} - \mathbf{y}$ and $c^{(4)}\mathbf{v}'_{\rho} = \mathbf{z}^{(4)} - \mathbf{y}'$. Since both transcripts are accepted, we have

$$\mathsf{C}_1 = \mathsf{Com}(\rho, y) = \mathsf{Com}(\rho', y')\,,$$

so the computationally binding property of Com implies that with overwhelming probability we have $\rho = \rho'$ and $\mathbf{y} = \mathbf{y}'$. Now, we put everything together to get

$$\begin{aligned} 0 &= \mathbf{A}(\mathbf{z}^{(1)}_{\sigma^{-1}} - \mathbf{z}^{(3)}_{\sigma^{-1}}) \\ &= \mathbf{A}(\mathbf{z}^{(2)}_{\sigma^{-1}} - \mathbf{z}^{(4)}_{\sigma^{-1}}) \\ &= \mathbf{A}(c^{(2)}\mathbf{v}_{\rho\sigma^{-1}} - \mathbf{y}_{\sigma^{-1}} - c^{(4)}\mathbf{v}_{\rho\sigma^{-1}} + \mathbf{y}_{\sigma^{-1}}) \\ &= (c^{(2)} - c^{(4)})\mathbf{A}\mathbf{v}_{\rho\sigma^{-1}} \, . \end{aligned}$$

Since $c^{(2)} - c^{(4)}$ is nonzero, this means that $\rho\sigma^{-1}$ is a solution to the permuted kernel problem. Moreover the extractor can efficiently extract this solution, because he can extract ρ from either transcript 2 or 4, and he can extract σ from either transcript 1 or 3.

It is known that $2q$-extractability implies soundness with error $\frac{q+1}{2q}$, where q is the size of the first challenge space [22,24]. In our case, the first challenge space has $p-1$ elements, so the optimized IDS has soundness error $\frac{p}{2p-2}$.

Honest-Verifier Zero-knowledge. To prove Honest-Verifier Zero-Knowledge we construct a simulator that outputs transcripts that are computationally indistinguishable from transcripts of honest executions of the identification scheme. First, the simulator picks a uniformly random value $c \in \mathbb{F}_p \setminus \{0\}$ and a uniformly random bit b. We treat the cases $b = 0$ and $b = 1$ separately.

Case $b = 0$ **:** The simulator picks a random seed seed_0, a uniformly random vector $\mathbf{z}$, and computes $\sigma = \mathsf{PRG}_1(\mathsf{seed}_0)$ and $\mathsf{C}_0 = \mathsf{Com}(\sigma, \mathbf{Az})$. The simulator also commits to a dummy value to get C_1. Now the simulator outputs $(\mathsf{C}_0, \mathsf{C}_1, c, \mathbf{z}, b, \mathsf{seed}_\sigma)$.

This distribution is indistinguishable from honestly generated transcripts with $b = 0$. Indeed, the values $c, \mathbf{z}, \mathsf{seed}_0$ are indistinguishable from uniformly random in both the simulated transcripts and the honest transcripts (here we use the assumption that the output of PRG_2 is indistinguishable from the uniform distribution). The first commitment $\mathsf{C}_0 = \mathsf{Com}(\sigma, \mathbf{Az}_{\sigma^{-1}})$ is a function of seed_0 and $\mathbf{z}$, so it also has the same distribution in the simulated and the honest transcripts. Finally, the commitment C_1 is never opened, so the computationally hiding property of Com guarantees that C_1 in the simulated transcript is computationally indistinguishable from the C_1 in an honest transcript.

Case $b = 1$ **:** The simulator picks a uniformly random seed seed_1 and a uniformly random permutation ρ and computes $\mathbf{r}_\sigma = \mathsf{PRG}_2(\mathsf{seed}_1)$, $z = c\mathbf{v}_\rho + \mathbf{r}_\sigma$ and $\mathsf{C}_1 = \mathsf{Com}()$. The simulator also commits to a dummy value to produce a commitment C_0, then the simulator outputs the transcript $(\mathsf{C}_0, \mathsf{C}_1, c, \mathbf{z}, b, \mathsf{seed}_1)$.

We now show that the simulated transcripts are indistinguishable from honestly generated transcripts with $b = 1$. It is clear that c and seed_1 are uniformly random in both the simulated transcripts and the honestly generated

transcripts. Moreover, in both the simulated and the real transcripts, $\mathbf{z}$ is equal to $\mathsf{PRG}_2(\mathsf{seed}_1) + c\mathbf{v}_\rho$, and $\mathsf{C}_1 = \mathsf{Com}(\rho, \mathsf{PRG}_2(\mathsf{seed}_1))$ where ρ is indistinguishable from a uniformly random permutation (here we need the assumption that the output of PRG_1 is indistinguishable from a uniformly random permutation). Therefore $\mathbf{z}$ and C_1 have the same distribution in the simulated and the honest transcripts. Finally, the computationally hiding properties of Com guarantee that the value of C_0 in the simulated transcripts is indistinguishable from that of C_0 in honestly generated transcripts.

3.4 Communication cost

We can now provide the communication complexity of N rounds of the modified IDS, of which the soundness error is $\left(\frac{p}{2p-2}\right)^N$. The commitment consists of a single hash value, which is only 2λ bits. The first response consists of N vectors of length n over $\mathbb{F}_p$, so this costs $Nn\lceil \log_2 p \rceil$ bits of communication. Lastly, the second responses consist of N random λ-bit seeds, N commitments (which consist of 2λ bits each) and N commitment random strings (which consist of λ bits each), so this costs $4N\lambda$ bits of communication. In total, the communication cost (ignoring the challenges) is

$$2\lambda + N\left(n\lceil \log_2 p \rceil + 4\lambda\right) .$$

4 Digital signature scheme (DSS) based on PKP

We present here the main contribution of this work which is to construct a digital signature scheme, based on the PKP problem, from the optimized IDS defined in Section 3. This is simply a direct application of the well-known Fiat Shamir transformation [9].

The key generation algorithm is identical to the key generation algorithm for the identification scheme. To sign a message m, the signer executes the first phase of the commitment scheme to get a commitment com. Then he derives the first challenge $\mathbf{c} = (c_1, \cdots, c_N)$ from m and com by evaluating a hash function $\mathcal{H}_1(m||\mathsf{com})$. Then he does the next phase of the identification protocol to get the N response vectors $\mathsf{rsp}_1 = (\mathbf{z}^{(1)}, \cdots, \mathbf{z}^{(N)})$. Then he uses a second hash function to derive $\mathbf{b} = (b_1, \ldots, b_N)$ from m, com and rsp_1 as $\mathcal{H}_2(m||\mathsf{com}, \mathsf{rsp}_1)$. Then he finishes the identification protocol to obtain the vector of second responses $\mathsf{rsp}_2 = (\mathsf{rsp}^{(1)}, \cdots, \mathsf{rsp}^{(N)})$. Then, the signature is simply $(\mathsf{com}, \mathsf{rsp}_1, \mathsf{rsp}_2)$.

To verify a signature $(\mathsf{com}, \mathsf{rsp}_1, \mathsf{rsp}_2)$ for a message m, the verifier simply uses the hash function $\mathcal{H}_1$ and $\mathcal{H}_2$ to obtain $\mathbf{c}$ and $\mathbf{b}$ respectively. Then, he verifies that $(\mathsf{com}, \mathbf{c}, \mathsf{rsp}_1, \mathbf{b}, \mathsf{rsp}_2)$ is a valid transcript of the identification protocol.

Algorithm 5 Sign(sk, m)

derive $\mathbf{A}, \mathbf{v}$ and π from sk.
for i from 1 to N **do**
  pick λ-bit seeds $\mathsf{seed}_0^{(i)}$ and $\mathsf{seed}_1^{(i)}$ uniformly at random
  $\sigma^{(i)} \leftarrow \mathsf{PRG}_1(\mathsf{seed}_0^{(i)})$
  $\mathbf{r}_\sigma^{(i)} \leftarrow \mathsf{PRG}_2(\mathsf{seed}_1^{(i)})$
  $\mathsf{C}_0^{(i)} = \mathrm{Com}\big(\sigma^{(i)}, \mathbf{A}\mathbf{r}^{(i)}\big)$,
  $\mathsf{C}_1^{(i)} = \mathrm{Com}\big(\pi\sigma^{(i)}, \mathbf{r}_\sigma^{(i)}\big)$.
end for
$\mathsf{com} := \mathcal{H}_{\mathsf{com}}\big(\mathsf{C}_0^{(1)}, \mathsf{C}_1^{(1)}, \cdots, \mathsf{C}_0^{(N)}, \mathsf{C}_1^{(N)}\big)$
$c^{(1)}, \cdots, c^{(N)} \leftarrow \mathcal{H}_1\big(m||\mathsf{com}\big)$. $c^i \in \mathbb{F}_p \setminus \{0\}$
for i from 1 to N **do**
  $\mathbf{z}^{(i)} \leftarrow \mathbf{r}_\sigma^{(i)} + c^{(i)}\mathbf{v}_{\pi\sigma^{(i)}}$
end for
$\mathsf{rsp}_1 \leftarrow (\mathbf{z}^{(1)}, \cdots, \mathbf{z}^{(N)})$
$b^{(1)}, \cdots, b^{(N)} \leftarrow \mathcal{H}_2(m||\mathsf{com}||\mathsf{rsp}_1)$
for i from 1 to N **do**
  $\mathsf{rsp}_2^{(i)} \leftarrow (\mathsf{seed}_{b^{(i)}}^{(i)} || \mathsf{C}_{1-b^{(i)}}^{(i)})$
end for
$\mathsf{rsp}_2 \leftarrow (\mathsf{rsp}_2{}^{(1)}, \cdots, \mathsf{rsp}_2{}^{(N)})$
Return $(\mathsf{com}, \mathsf{rsp}_1, \mathsf{rsp}_2)$

Algorithm 6 Verify($m, \mathsf{pk}, \sigma = (\mathsf{com}, \mathsf{rsp}_1, \mathsf{rsp}_2)$)

$c^{(1)}, \cdots, c^{(N)} \leftarrow \mathcal{H}_1\big(m||\mathsf{com}\big)$.
$b^{(1)}, \cdots, b^{(N)} \leftarrow \mathcal{H}_2(m||\mathsf{com}||\mathsf{rsp}_1)$
Parse rsp_1 as $\mathbf{z}^{(1)}, \cdots, \mathbf{z}^{(N)}$
Parse rsp_2 as $\mathsf{seed}^{(1)}, \cdots, \mathsf{seed}^{(N)}, \mathsf{C}_{1-b^{(1)}}^{(1)}, \cdots, \mathsf{C}_{1-b^{(N)}}^{(N)}$
for i from 1 to N **do**
  if $b^{(i)} = 0$ **then**
    $\sigma^{(i)} \leftarrow \mathsf{PRG}_1(\mathsf{seed}^{(i)})$
    $\mathsf{C}_0^{(i)} \leftarrow \mathrm{Com}(\sigma^{(i)}, \mathbf{A}\mathbf{z}_{\sigma^{(i)}-1})$
  else
    $\mathbf{r}_\sigma^{(i)} \leftarrow \mathsf{PRG}_2(\mathsf{seed}^{(i)})$
    if $\mathbf{z}^{(i)} - \mathbf{r}_\sigma$ is not a permutation of $c\mathbf{v}$ **then**
      Return reject
    else
      $\pi\sigma^{(i)} \leftarrow$ the permutation that maps $c\mathbf{v}$ to $\mathbf{z}^{(i)} - \mathbf{r}_\sigma$.
    end if
    $\mathsf{C}_1^{(i)} \leftarrow \mathrm{Com}(\pi\sigma^{(i)}, \mathbf{r}_\sigma^{(i)})$
  end if
end for
$\mathsf{com}' := \mathcal{H}_{\mathsf{com}}\big(\mathsf{C}_0^{(1)}, \mathsf{C}_1^{(1)}, \cdots, \mathsf{C}_0^{(N)}, \mathsf{C}_1^{(N)}\big)$
Return accept if and only if $\mathsf{com} = \mathsf{com}'$

The signing and verification algorithms are displayed in Algorithm 5 and 6 in more detail.

We then get the same security result as Th. 5.1 in [7].

Theorem 4. *PKP-DSS is Existential-Unforgeable under Chosen Adaptive Message Attacks (EU-CMA) in the random oracle model, if*

- *the search version of the Permuted Kernel problem is intractable,*
- *the hash functions and pseudo-random generators are modeled as random oracles,*
- *the commitment functions are computationally binding, computationally hiding, and the probability that their output takes a given value is negligible in the security parameter.*

The proof is the same as in [7].

4.1 Generic attack

If the number of iterations N is chosen such that $(\frac{p}{2p-2})^N \leq 2^{-\lambda}$, then the cheating probability of the identification protocol is bounded by $2^{-\lambda}$. However, a recent attack by Kales and Zaverucha on MQDSS reveals that this does not meant that the Fiat-Shamir signature scheme has λ bits of security [16]. They give a generic attack that also applies to PKP-DSS. The attack exploits the fact that if an attacker can guess the first challenge **or** the second challenge, he can produce responses that the verifier will accept. The idea is to split up the attack in two phases. In the first phase, the attacker guesses the values of the N first challenges, and uses this guess to produce commitments. Then, he derives the challenges from the commitment and he hopes that at least k of his N guesses are correct. This requires on average

$$\mathsf{Cost}_1(N,k) = \sum_{i=k}^{N} \left(\frac{1}{p-1}\right)^k \left(\frac{p-2}{p-1}\right)^{N-k} \binom{N}{k}$$

trials. In the second phase, the attacker guesses the values of second challenges, and uses these guesses to generate a response. Then he derives the second challenges with a hash function and he hopes that his guess was correct for the $N-k$ rounds of the identification protocol where he did not guess the first challenge correctly. This requires on average 2^{N-k} tries. Therefore, the total cost of the attack is

$$\min_{0\leq k\leq N} \mathsf{Cost}_1(N,k) + 2^{N-k}\,.$$

5 Parameter choice and Implementation

5.1 Parameter choice

The PKP-DSS is mainly affected by the following set of parameters: (p, n, m). We now explicitly detail the choice of these parameters. Recall that firstly the IDS [23] was designed to suit small devices. Thus, Shamir proposed $p = 251$. To have an efficient implementation we choose p to be a prime number close to a power of 2, such as 251, 509 and 4093. A solution of a random instance of PKP is to find a kernel n-vector $(\mathbf{v}_\pi)$ with distinct coordinates in $\mathbb{F}_p$. Hence, the probability to find such a vector shouldn't be too small. The probability of an arbitrary vector to be in the kernel of the matrix $A \in \mathcal{M}_{m\times n}$ whose rank is equal to m, is p^{-m}. Moreover, if the n-vector v has no repeated entries, its orbit under the possible permutations π contains $n!$ vectors. Thus, to get on average one solution, we have the following constraint: $n! \approx p^m$. And finally, using the complexity of Poupard's algorithm [18] combined with Patarin-Chauvaud's method (See Section 2.1), triplets of (p, n, m) were selected matching the security requirements and optimizing the size of the signature. With these parameter choices, the scheme is secure against all the attacks described in [17]. We pick the value of N just large enough such that

$$\min_{0\leq k\leq N} \mathsf{Cost}_1(N, k) + 2^{N-k} \geq 2^\lambda \,,$$

such that the scheme is secure against the generic attack of Kales and Zaverucha [16]. The chosen parameter sets for three different security levels are shown in Table 1.

Parameter Set	Security level	p	n	m	Iterations N	Attack cost
PKP-DSS-128	128	251	69	41	157	2^{130}
PKP-DSS-192	192	509	94	54	229	2^{193}
PKP-DSS-256	256	4093	106	47	289	2^{257}

Table 1. PKP-DSS Parameters sets

5.2 Key and signature sizes

Public key. A public key consists of the last column $\mathbf{c_n^A}$ of $\mathbf{A}$ and a random seed **pk.seed**, which is used to generate $\mathbf{A^*}$ which is formed by all but the last column of $\mathbf{A}$ and the vector $\mathbf{v}$. Therefore, the public key consist of $\lambda + m\lfloor \log_2(p) \rfloor$ bits.

Secret key. A secret key is just a random seed **pk.seed** that was used to seed the key generation algorithm, therefore it consists of only λ bits.

Signature. Finally, a signature consists of a transcript of the identification protocol (excluding the challenges, because they are computed with a hash function). In Sect 3.4 we calculated that a transcript can be represented with $2\lambda + N\left(n\lceil\log_2 p\rceil + 4\lambda\right)$ bits, so this is also the signature size.

In Table 2 we summarize the key and signature sizes for the parameter sets proposed in the previous section.

Security level	Parameters (p, n, m, N)	$\lvert\mathsf{sk}\rvert$ Bytes	$\lvert\mathsf{pk}\rvert$ Bytes	$\lvert\mathsf{sig}\rvert$ Kilobytes
128	$(251, 69, 41, 157)$	16	57	20.4
192	$(509, 94, 54, 229)$	24	85	43.4
256	$(4093, 106, 47, 289)$	32	103	76.5

Table 2. Key and signature sizes for PKP-DSS with the three proposed parameter sets.

5.3 Implementation

To showcase the efficiency of PKP-DSS and to compare the performance to existing Fiat-Shamir signatures we made a proof-of-concept implementation in plain C. The code of our implementation is available on GitHub at [4]. We have used SHA-3 as hash function and commitment scheme, and we have used SHAKE128 as extendable output function. The running time of the signing and verification algorithms is dominated by expanding seeds into random vectors and random permutations. This can be sped up by using a vectorized implementation of SHAKE128, and using vector instructions to convert the random bitstring into a vector over $\mathbb{F}_p$ or a permutation in S_n. We leave this task for future work.

Making the implementation constant time. Most of the key generation and signing algorithms is inherently constant time (signing branches on the value of the challenge bits b, but this does not leak information because b is public). The only problem was that applying a secret permutation to the entries of a vector, when implemented naively, involves accessing data at secret indices. To prevent this potential timing leak we used the "djbsort" constant time sorting code [3]. More specifically, (see Alg. 7) we combine the permutation and the vector into a single list of n integers, where the permutation is stored in the most significant bits, and the entries of the vector are stored in the least significant bits. Then we sort this list of integers in constant time and we extract the permuted vector from the low order bits. Relative to the naive implementation this slows down signing by only 11%. There is no significant slowdown for key generation.

Algorithm 7 Constant time computation of $v' = v_\sigma$

1: Initialize a list of integers $L \leftarrow \emptyset$
2: $L := \big[\sigma[1] * B + v[1], \cdots, \sigma[n] * B + v[n]\big]$, where $B > n$ is a constant
3: sort L in constant time
4: $v' := \big[L[1] \ mod \ B, \cdots, L[n] \ mod \ B\big]$
5: Return v'

5.4 Performance results

To measure the performance of our implementation we ran experiments on a laptop with a i5-8250U CPU running at 1.8 GHz. The C code was compiled with gcc version 7.4.0 with the compile option `-O3`. The cycle counts in Table 3 are averages of 10000 key generations, signings, and verifications.

Security level	Parameters (p, n, m, N)	KeyGen 10^3 cycles	Sign 10^3 cycles	Verify 10^3 cycles
128	$(251, 69, 41, 157)$	72	2518	896
192	$(509, 94, 54, 229)$	121	5486	2088
256	$(4093, 106, 47, 289)$	151	7411	3491

Table 3. Average cycle counts for key generation, signing and verification, for our implementation of PKP-DSS with the three proposed parameter sets.

5.5 Comparison with existing FS signatures

In Table 4, we compare PKP-DSS to MQDSS, Picnic, and Picnic2. We can see that for all the schemes the public and secret keys are all very small. The main differences are signature size and speed. When compared to MQDSS, the signature sizes of PKP-DSS are roughly 30% smaller, while being a factor 14 and 30 faster for signing and verification respectively. Compared to Picnic, the PKP-DSS signatures are roughly 40% smaller, and signing and verification are 4 and 9 times faster respectively. Compared to Picinc2 our scheme is 153 and 170 times faster for signing and verification, but this comes at the cost of 50% larger signatures. Finally, compared to SUSHSYFISH [12], a different scheme based on the Permuted Kernel Problem, our scheme is 3.4 and 6.6 times faster, but at the cost of 45% larger signatures.

6 Conclusion

We introduce a new post-quantum secure signature scheme PKP-DSS, which is based on a PKP Zero-knowledge identification scheme [23]. We optimized this

Security level	Scheme	Secret key (Bytes)	Public key (Bytes)	Signature (KBytes)	Sign 10^6 cycles	Verify 10^6 cycles
128	PKP-DSS-128	16	57	20.4	2.5	0.9
	MQDSS-31-48	16	46	28.0	36	27
	Picnic-L1-FS	16	32	33.2	10	8.4
	Picnic2-L1-FS	16	32	13.5	384	153
	SUSHSYFISH-1	16	72	14.0	8.6	6
192	PKP-DSS-192	24	85	43.4	5.5	2.1
	MQDSS-31-64	24	64	58.6	116	85
	Picnic-L3-FS	24	48	74.9	24	20
	Picnic2-L3-FS	24	48	29.1	1183	357
	SUSHSYFISH-3	24	108	30.8	22.7	16.5
256	PKP-DSS-256	32	103	76.5	7.4	3.5
	Picnic-L5-FS	32	64	129.7	44	38
	Picnic2-L5-FS	32	64	53.5	2551	643
	SUSHSYFISH-5	32	142	54.9	25.7	18

Table 4. Comparison of different post-quantum Fiat-Shamir schemes

identification scheme, and to make it non-interactive, we used the well-known Fiat-Shamir transform.

We developed a constant-time implementation of PKP-DSS and we conclude that our scheme is competitive with other Post-Quantum Fiat-Shamir signature schemes such as MQDSS, Picnic/Picnic2, and SUSHSYFISH. The main advantages of our scheme are that signing and verification are much faster than existing Fiat-Shamir signatures and that the scheme is very simple to implement. Our implementation takes only 440 lines of C code.

Acknowledgments. This work was supported by the European Commission through the Horizon 2020 research and innovation program under grant agreement H2020-DS-LEIT-2017-780108 FENTEC, by the Flemish Government through FWO SBO project SNIPPET S007619N and by the IF/C1 on Cryptanalysis of post-quantum cryptography and by the French Programme d'Investissement d'Avenir under national project RISQ P141580. Ward Beullens is funded by an FWO fellowship.

References

1. Baritaud, T., Campana, M., Chauvaud, P., Gilbert, H. On the security of the permuted kernel identification scheme. In Annual International Cryptology Conference (pp. 305-311). (1992, August), Springer, Berlin, Heidelberg.
2. Bennett, C. H., Bernstein, E., Brassard, G., Vazirani, U. (1997). Strengths and weaknesses of quantum computing. SIAM journal on Computing, 26(5), 1510-1523.
3. The djbsort software library for sorting arrays of integers or floating-point numbers in constant time. `https://sorting.cr.yp.to/`

4. Beullens, Ward. PKPDSS, (2019) Public GitHub repository. https://github.com/WardBeullens/PKPDSS
5. Beullens, Ward. On sigma protocols with helper for MQ and PKP, fishy signature schemes and more. Cryptology ePrint Archive, Report 2019/490, 2019. https://eprint.iacr.org/2019/490.
6. Chase, M., Derler, D., Goldfeder, S., Orlandi, C., Ramacher, S., Rechberger, C., ... Zaverucha, G. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (pp. 1825-1842) (2017, October). ACM.
7. Chen, M. S., Hülsing, A., Rijneveld, J., Samardjiska, S., Schwabe, P. MQDSS specifications. (2018).
8. Damgård, I. Commitment schemes and zero-knowledge protocols. In School organized by the European Educational Forum (pp. 63-86). (1998, June). Springer, Berlin, Heidelberg.
9. Fiat, A., Shamir, A. (1986, August). How to prove yourself: Practical solutions to identification and signature problems. In Advances in Cryptology—CRYPTO'86 (pp. 186-194). Springer, Berlin, Heidelberg.
10. Gary, M., Johnson, D. (1979). Computers and Intractability: A Guide to NP-Completeness. New York: W H.
11. Georgiades, J. (1992). Some remarks on the security of the identification scheme based on permuted kernels. Journal of Cryptology, 5(2), 133-137.
12. W. Beullens. On sigma protocols with helper for MQ and PKP, fishy signature schemes and more IACR Cryptology ePrint Archive 2019 (2019): https://eprint.iacr.org/2019/490.
13. Don, J., Fehr, S., Majenz, C., Schaffner, C. Security of the Fiat-Shamir transformation in the Quantum Random-Oracle Model. Cryptology ePrint Archive, Report 2019/190 (2019), https://eprint.iacr.org/2019/190
14. Haitner, I., Nguyen, M. H., Ong, S. J., Reingold, O., Vadhan, S. Statistically hiding commitments and statistical zero-knowledge arguments from any one-way function. SIAM Journal on Computing, 39(3), pp. 1153-1218.
15. Jaulmes, É., Joux, A. Cryptanalysis of PKP: a new approach. In International Workshop on Public Key Cryptography (pp. 165-172). (2001, February) Springer, Berlin, Heidelberg.
16. Daniel Kales, Greg Zaverucha. "Forgery attacks against MQDSSv2.0" Note postes on the NIST PQC forum https://groups.google.com/a/list.nist.gov/forum/?utm_medium=email&utm_source=footer#!msg/pqc-forum/LlHhfwg73eQ/omM6TWwlEwAJ
17. Koussa, Eliane, Gilles Macario-Rat, and Jacques Patarin. "On the complexity of the Permuted Kernel Problem." IACR Cryptology ePrint Archive 2019 (2019): 412.
18. Poupard, G., (1997). A realistic security analysis of identification schemes based on combinatorial problems. European Transactions on Telecommunications.
19. Lampe, R., Patarin, J. Analysis of Some Natural Variants of the PKP Algorithm. IACR Cryptology ePrint Archive, 2011, 686.
20. NIST categories: Security strength categories. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf
21. Patarin, J., Chauvaud, P. Improved algorithms for the permuted kernel problem. In Annual International Cryptology Conference (pp. 391-402). (1993, August) Springer, Berlin, Heidelberg.

22. Sakumoto, K., Shirai, T., Hiwatari, H. (2011, August). Public-key identification schemes based on multivariate quadratic polynomials. In Annual Cryptology Conference (pp. 706-723). Springer, Berlin, Heidelberg.
23. Shamir, A. (1989, August). An efficient identification scheme based on permuted kernels. In Conference on the Theory and Application of Cryptology (pp. 606-609). Springer, New York, NY.
24. Chen, Ming-Shing, et al. From 5-Pass $\mathcal{MQ}$-Based Identification to $\mathcal{MQ}$-Based Signatures. International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, 2016.
25. Kiltz, E., Lyubashevsky, V., Schaffner, C. A new identification scheme based on syndrome decoding. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 552-586). (2018) Springer, Cham.
26. Unruh, D. Post-quantum security of Fiat-Shamir. In International Conference on the Theory and Application of Cryptology and Information Security. pp. 65-95. (2017, December) Springer, Cham.
27. Unruh, D. Non-interactive zero-knowledge proofs in the quantum random oracle model. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 755-784). (2015, April). Springer, Berlin, Heidelberg.
28. Gaëtan Leurent and Phong Q. Nguyen, How Risky is the Random-Oracle Model?
29. Shai Halevi and Hugo Krawczyk, Strengthening Digital Signatures Via Randomized Hashing, In CRYPTO,2006,41–59