post-quantum PFDH Signature Scheme

immediate

December 31, 2020

Abstract

The Full Domain Hash designed by Bellare and Rogaway is a signature scheme provably secure against chosen message attack in the random oracle model but with a security reduction which is not tight. It is enhanced by Coron into PFDH (Probabilistic Full Domain Hash) which offers better security. In this paper, we propose a post-quantum version of this latter, called pqPFDH (post-quantum Probabilistic Full Domain Hash). Our scheme has better security and remains secure even in the presence of an adversary that has access to quantum computers. For each signature generation, a random salt r is generated allowing to compute the RSA exponent. The security level achieved here is set as follows $\varepsilon_{\mathcal{R}} = (1 - \frac{1}{2^{|r|}})\varepsilon_{\mathcal{A}}$, where $\varepsilon_{\mathcal{R}}$ is the success probability of a reduction algorithm $\mathcal{R}$ that is able to invert RSA by using an attacker $\mathcal{A}$ that breaks pqPFDH with probability $\varepsilon_{\mathcal{A}}$.

keywords: *post-quantum RSA, signature scheme, PFDH, security reduction, CMA*

1 Introduction

C'est ici la modification.
The invention of Shor's algorithm [1] and the hope that quantum computer will be built making a qubit operation as inexpensive as a bit operation, devote the researches to post-quantum cryptography. It is clear that when quantum computers will be built, they will have the potential to break RSA, Diffie-Hellman and elliptic curve cryptography, roughly speaking public key cryptography.
To insure against this risk, cryptography researchers have deployed alternative public key cryptography algorithms called post-quantum algorithms that would resist quantum computers. In 2015, the NSA called for transition to quantum resistant algorithms designed to be safe even in the face of quantum computers, and in 2017 the US standardization agency NIST began a process that will eventually standardize post quantum algorithms. The main goal is to design practical and provably secure schemes which can resist to the threat of quantum computers.
In the digital signature cases, it is important to adjust the parameters such that signature and verification remain practicable for our classical computer and always being secure against an attacker that has access to quantum computers. This is the

adopted technical for post-quantum RSA ([**?**]) in which the modulus is the product of many primes in order to stop Shor's algorithm. Even though some operation like key generation take much times for classical computer, for quantum computers this remains reasonable.
To evaluate the security of a signature scheme, we give the attacker the power to obtain signatures of his choice. This security notion was defined by Goldwasser, Micali and Rivest in [11], as existential unforgeability under an adaptive chosen message attack.
The random oracle model, introduced by Bellare and Rogaway in [2], is a theoretical framework allowing to prove the security of hashed-and-sign signature schemes. In this model hash functions is seen as an oracle which outputs a random value for each new query.
To prove post-quantum security one needs to prove in quantum-accessible random oracle model where the adversary can query the random oracle with quantum state. There exists schemes that are secure when the adversary is given classical access to the random oracle, but which are insecure when the adversary can make quantum oracle queries. So, it is ingenious to develop generic conditions under which a classical random proof implies security in the quantum-accessible random oracle model. The concept of history-free reduction, proposed by Dan Boneh et al. in [17], is a category of classical random oracle reductions that allows to reach this security level. In this paper, they proved that the PFDH proposed by Coron in [5] is history-free.

Related works: The Full Domain Hash signature scheme was introduced by Bellare and Rogaway [3] by using the random oracle model. If f is a trapdoor permutation and $\mathcal{RO}$ is a random function from $\{0,1\}^*$ to the domain of f, these authors prove that signing a message m via $f^{-1}(\mathcal{RO}(m))$ is secure. But what happen for the tightness?
In 1996, Bellare and Rogaway in [3], propose a security reduction for RSA-FDH where the reduction algorithm $\mathcal{R}$ will provide a perfect simulation and $(\varepsilon_{\mathcal{R}}, \tau_{\mathcal{R}})$-solves RSA trapdoor permutation with success probability $\varepsilon_{\mathcal{R}} \geq \frac{\varepsilon_{\mathcal{A}}}{q_h+q_{sig}+1}$ and time bound $\tau_{\mathcal{R}} \leq \tau_{\mathcal{A}} + (q_h + q_{sig} + 1)polynom(k)$ where q_h (respectively q_{sig}) is the number of queries made by the attacker to the oracle hash (respectively to the oracle signature).
To improve the tightness of RSA-FDH, in 2002, Coron in [5], proposes a security reduction where the reduction algorithm $\mathcal{R}$ will provide a perfect simulation and $(\varepsilon_{\mathcal{R}}, \tau_{\mathcal{R}})$-solves RSA trapdoor permutation with success probability $\varepsilon_{\mathcal{R}} = \frac{\varepsilon_{\mathcal{A}}}{exp(1)q_{sig}}$ and time bound $\tau_{\mathcal{R}} = \tau_{\mathcal{A}} + (q_h + q_{sig} + 1)\mathcal{O}(k^3)$ where q_h (respectively q_{sig}) is the number of queries made by the attacker to the oracle hash (respectively to the oracle signature).
In [6], Coron proves that the reduction of RSA-FDH cannot be tight and this fact is intrinsic to RSA-FDH and conclude that one should have to look after a better and more efficient design than RSA-FDH.

Coron studies also the tightness of probabilistic FDH (PFDH briefly) in [6], and proposes a tight security reduction where the reduction algorithm $\mathcal{R}$ will provide a perfect simulation and $(\varepsilon_{\mathcal{R}}, \tau_{\mathcal{R}})$-solves RSA trapdoor permutation with success probability $\varepsilon_{\mathcal{R}} = \frac{\varepsilon_{\mathcal{A}}}{(1+6\frac{q_{sig}}{2^{k_0}})}$ and time bound $\tau_{\mathcal{R}} = \tau_{\mathcal{A}} + (q_h + q_{sig})\mathcal{O}(k^3)$ where q_h (respectively q_{sig}) is the number of queries made by the attacker to the oracle hash (respectively to the oracle signature).
In 2002, Dodis and Reyzin in [9], generalizing Coron's work, show that a similar result holds for any trapdoor permutation induced by a family of claw-free permutations. They show that a tight security reduction is impossible for RSA-FDH, RSA-PFDH and PSS-R with a small random. Moreover they show that, this cases, for any signature scheme outputs $sign(m) = \left(f^{-1}(\mathcal{RO}(m))\right)$ or $sign_r(m) = \left(f^{-1}(\mathcal{RO}(m,r)), r\right)$ [where f^{-1} is the inverse of the trapdoor permutation, m is the message and r is a random], if the scheme is to be analysed with a general "black-box" trapdoor permutation f.
In 2005, Dodis, Reyzin and Pietrzak in [10], using previous work of Dodis and al., prove that one can't hope to prove $sign(m) = \left(f^{-1}(\mathcal{RO}(m))\right)$ secure under any assumption which is satisfied by random permutations. Pietrzak tells that there work does not mean that RSA-FDH with SHA1 is insecure, but it is just impossible to prove it (with a tight security reduction).
In [17], Boneh at al. prove that FDH is claw-free permutation. This means that its security for classical computer implies its security for quantum computers. They also prove the quantum security of a varant of FDH due to Katz and Wang [18] which has tight security reduction.

Contributions: In this paper, we propose the post quantum version of the Probabilistic Full Domain Hash signature scheme (PFDH), called pqPFDH . A random salt is generated to compute the exponentiation in RSA trapdoor permutation at each signature.
The reduction algorithm of pqPFDH have: success probability $\varepsilon_{\mathcal{R}} = \varepsilon_{\mathcal{A}}(1 - \frac{1}{2^{|a|}})$ and time bound $\tau_{\mathcal{R}} \leq \tau_{\mathcal{A}} + q_G\mathcal{O}(1) + (q_h + q_{sig} + 2)\mathcal{O}(k^3)$, where $2^{|a|}$ is the number of random allowed to use for the exponentiation in pqPFDH signature scheme. To have this good success probability we use signature with random such that the trapdoor permutation are randomly chosen and inverted at each signature; namely, the general form of our signature is $sign_r(m) = \left(f_r^{-1}(\mathcal{RO}(m,r)), r\right)$ or $sign_f(m) = \left(f^{-1}(\mathcal{RO}(m)), f\right)$.
We remark that our security reduction for pqPFDH signature is tight (which means that it is equivalent to those of RSA). In this cases, we see that with a random of 5 bits, we have $\varepsilon_{\mathcal{R}} = 0.96875\ \varepsilon_{\mathcal{A}}$, where $\varepsilon_{\mathcal{R}}$ is the success probability of an algorithm $\mathcal{R}$ that is able to break RSA by using an attacker $\mathcal{A}$ that breaks pqPFDH with probability $\varepsilon_{\mathcal{A}}$. But for RSA-PFDH, it is necessary to use random with size greater than $\log_2 q_{sig} + 8$ (≥ 38), where q_{sig} is the number of signatures queries, in order to have $\varepsilon_{\mathcal{R}} = 1.04\ \varepsilon_{\mathcal{A}}$.
We remark also, that the success probability of the simulation is independent from

the number of signing and hashing oracles queries. Our new signature scheme is more secure than RSA-PFDH relatively to all known reductions, but it is also more slower.

Outline: This pape is organised as follows:

- In section 2, we recall some preliminaries on security model 2.1, on signature scheme 2.2 and on post-quantum RSA problem 2.3.
- In section 3, we propose pqPFDH which is a post-quantum version of the PFDH signature scheme with a random exponent for RSA's trapdoor in subsection 3.1 and show its security proof in subsection 3.2.

2 Preliminaries

In this section, we recall some definitions and known results about signatures. Definitions, basic notations and classical results are followed in "Introduction to Modern Cryptography" [18] of Katz and Lindell, and in "Provable Security for Public Key Schemes" of Pointcheval [14].

2.1 Security model

The Random Oracle Model: For any constant k, a random oracle is a function F_{rand} selected randomly in the set $\mathcal{F}_k$ of functions from $\{0,1\}^*$ to $\{0,1\}^k$.

Proof in the Random Oracle Model: In random oracle model (see [2] and [4]):

- We assume that the hash function is a random function i.e in the simulation process, the hash function is replaced by a random oracle which outputs a random value for each new input;
- The only way to compute the hash function, is to query the oracle hash;
- The reduction algorithm $\mathcal{R}$ must simulate the environment of the attacker $\mathcal{A}$ with her public key only;
- When the attacker $\mathcal{A}$ requests the oracle hash, the reduction algorithm $\mathcal{R}$ can choose the random to return as digest; hence $\mathcal{R}$ is able to embed the challenge (any information which able $\mathcal{R}$ to to invert the related one way function at the end of the game) of his choice in the answer of the oracle hash to the requests of $\mathcal{A}$.
- At the end of the simulation, if the attacker $\mathcal{A}$ output valid forgery (which be never returned by the oracle signature) then $\mathcal{R}$ must able to invert the related one way function with a good tightness.

Significance of a Proof in the Random Oracle Model: It is known that a proof in the random oracle model does not imply that the scheme is secure in the real world (Canetti, Goldreich and Halevi, 98) in [4], it is widely believed to be an acceptable engineering principle to design provably secure schemes in random oracle model.

2.2 Signature and security model

Randomized Signature Schemes: A Randomized signature scheme is a tuple of the following algorithms: Key Generation (Genkey): with input a security parameter k, the key generation algorithm outputs a pair of keys (p_{key}, s_{key}).

Signature algorithm (Sig):

- with input a security parameter k, the signing algorithm outputs a random r;
- with input (s_{key}, m, r), the signing algorithm produces a signature σ.

Verification (Ver): with input (m, σ, p_{key}), the verification algorithm returns 1 if the signature is valid and 0 otherwise.

Security of Randomized Signature Schemes: Goldwasser, Micali and Rivest (in 1988) in [11], introduce the basic security notion for signatures called "existential unforgeability with respect to adaptive chosen-message attacks".

For this, a reduction algorithm $\mathcal{R}$ and an attacker $\mathcal{A}$, simulate the following game

Setup: $\mathcal{R}$ runs the algorithm Genkey with a security parameter k as input, to obtain the public key p_{key} and the secret key s_{key}, and gives p_{key} to the adversary.

Queries: Proceeding adaptively, $\mathcal{A}$ may request a signature on any message $m \in \mathcal{M}$ (multiple requests of the same message are allowed) and $\mathcal{R}$ will respond with (m, r, σ) where $\sigma = Sig(s_{key}, m, r)$ and r is a random. Let $Hist(\mathcal{S})$ be the signing data base (=set of signatures already outputted by the oracle signature to the queries of the $\mathcal{A}$).

Output: Eventually, $\mathcal{A}$ will output a pair (m, r, σ) and is said to win the game if $Ver(p_{key}, m, r, \sigma) = 1$ and if $(m, r, \sigma) \notin Hist(\mathcal{S})$ (this last condition force the attacker $\mathcal{A}$ to output his own forgery).

The probability that $\mathcal{A}$ wins in the above game is denoted $Adv\mathcal{A}$.

Unforgeability against Adaptive Chosen Message Attacks (EUF-CMA): A signature scheme (Genkey, Sig,Ver) is existentially unforgeable with respect to adaptive chosen message attacks if for all probabilistic polynomial time attacker $\mathcal{A}$, $Adv\mathcal{A}$ is negligible in the security parameter k.

BR-CMA: This adversary model is CMA where the number of random used by the probabilistic signature algorithm is fixed, says D. Hence the signer cannot sign (and outputs distinct values) the same message more than D times.

Definition 2.1. *An attacker $\mathcal{A}$ is said to $(t, q_h, q_{sig}, \varepsilon)$-break the signature scheme $(Genkey, Sig, Ver)$ if after at most $q_H(k)$ queries to the hash oracle, $q_{sig}(k)$ signature queries and $\tau(k)$ processing time, it outputs a valid forgery with probability at least $\varepsilon(k)$ for all $k \in \mathbb{N}$.*

Definition 2.2. *A signature scheme $(Genkey, Sig, Ver)$ is $(\tau, q_h, q_{sig}, \varepsilon)$-secure if there is no attacker who $(\tau, q_h, q_{sig}, \varepsilon)$-breaks the scheme.*

2.3 Post-quantum RSA problem

Post-quantum RSA consists of adjusting RSA parameters in order to make extremely large keys which will resist power of quantum computers. The idea is to use many small primes which constitute the secret key and their product gives the public key. Clearly, a user generates K primes $q_1, q_2, ..., q_K$ of the form $q_i = 2p_i^{\beta_i} + 1$, where $1 \leq i \leq K$ and establishes $n = \prod_{i=0}^{K} q_i$.

Definition 2.3. *Now for a given pqRSA modulus n, e an integer coprime with $\varphi(n) = (q_1 - 1)(q_2 - 1)...(q_K - 1)$ and $z \in (\frac{\mathbb{Z}}{n\mathbb{Z}})^*$, find x such that $x^e = z \mod n$ is the RSA problem. Then, an algorithm $\mathcal{R}$ is said to $(\tau_{\mathcal{R}}, \varepsilon_{\mathcal{R}})$-solve the pqRSA problem, if in at most $\tau_{\mathcal{R}}$ operations, $Pr\{(n, e) \leftarrow RSA(1^k), z \leftarrow (\frac{\mathbb{Z}}{n\mathbb{Z}})^*, x \leftarrow \mathcal{R}(n, e, z), x^e = z \mod n\} \geq \varepsilon_{\mathcal{R}}$, where the probability is taken over the distribution of (n, z) and over $\mathcal{R}$'s random tapes.*

3 On pqPFDH signature with a tight reduction

In this section, we propose a new signature pqPFDH and prove that it is EUF-BR-CMA secure under RSA assumption in the random oracle model even assuming highly scalable quantum computers. It is similar to Probabilistic Full Domain Hash RSA Sign (RSA-PFDH), but with a random exponent generated for each signature.

Our main objective is to prove that the previous reduction technical can be adapted to RSA-PFDH in order to have the same success probability as above: $\varepsilon_{\mathcal{R}} = \varepsilon_{\mathcal{A}}(1 - \frac{1}{2^{|a|}})$ where $|a|$ is the size of the random.

3.1 Signature process

Key generation $G_{key}(1^k)$:

1. with input 1^k, G_{key} generates many primes $q_1, q_2, ..., q_K$ of the form $q_i = 2p_i^{\beta_i} + 1$ for $1 \leq i \leq K$ and computes the modulus $n = q_1 \times q_2 \times ... \times q_K$.
2. It also uses a function G which on input a random $a \in \{0,1\}^{k_a}$ outputs an odd integer e which is used to compute d such that $e.d = 1 \; mod \; \varphi(n)$.
   $G : \{0,1\}^{k_a} \longmapsto \{0,1\}^{k_e} \cap \mathbb{N}_{odd}$, where $\{0,1\}^{k_e}$ the bit representation of the integer e such that $e.d = 1 \; mod \; \varphi(n)$.

3. After concatenate the message m with the random a and the exponent e, use the hash function
$H : \{0,1\}^{k_a+k_m+k_e} \longmapsto \mathbb{Z}_n^*$, where k_m is the length of the message m.
Then the public key is $p_{key} = n$ and the secret key is $s_{key} = (q_1, q_2, ..., q_K)$.

Remark 3.1. *To build G, one can proceed as follows:*
Let $G' : \{0,1\}^{k_a} \to \{0,1\}^{k_e-2}$ be a hash function where k_e is a security parameter, define the hash function G by $G(x) = 1||G'(x)||1 \in [0, n-1] \cap [2^{k_e}, 2^{k_e+1}[\cap \mathbb{N}_{odd}$ where $\mathbb{N}_{odd}$ is the set off odd integers.

Remark 3.2. *Since $G(x)$ is odd then, with a hight probability, $G(x)$ is coprime with $\varphi(n) = 2^K p_1^{\beta_1} p_2^{\beta_2} ... p_K^{\beta_K}$ because finding an odd integer which is not coprime with $\varphi(n)$ is equivalent to factor n (which is known to be difficult).*

Signature algorithm: To sign a message m, the signer Bob should do the following steps:

(1) pick the private key $(q_1, q_2, ..., q_N)$ and the two hash functions G and H;

(2) select a random salt $a \in \{0,1\}^{k_a}$, compute $e = G(a)$ and $d = e^{-1} \, mod \, \varphi(n)$;

(4) compute $y \longleftarrow H(e||m||a)$ and $\sigma \longleftarrow y^d \mod n$;

(5) The signature of m is (m, a, σ).

Verification algorithm: To verify the signature (m, a, σ) of Bob on m, the verifier Alice should do the following steps:

(1) pick n, Bob's public key and the the signature (m, a, σ);

(2) compute $H(e||m||a) = y$;

(3) if $y = \sigma^e \mod n$ then return 1 else return 0.

Why verication works? It is obvious!

3.2 Security proof: Reduction in the random oracle model

Theorem 3.3. *If there exists an attacker $\mathcal{A}$ that $(q_H, q_G, q_{sig}, \tau_{\mathcal{A}}, \varepsilon_{\mathcal{A}})$-solves EUF-BR-CMA the pqPFDH signature scheme, then there exists a reduction $\mathcal{R}$ simulating the environment of $\mathcal{A}$ in the random oracle model that $(\tau_{\mathcal{R}}, \varepsilon_{\mathcal{R}})$-solves RSA with success probability $\varepsilon_{\mathcal{R}} = \varepsilon_{\mathcal{A}}(1 - \frac{1}{2^{|a|}})$ and time bound $\tau_{\mathcal{R}} \leq \tau_{\mathcal{A}} + q_G \mathcal{O}(1) + (q_H + q_{sig} + 2)\mathcal{O}(k^3)$ where $|a|$ is the size of random used for exponentiation, $\mathcal{R}$ receives q_{sig} signature queries, q_H and q_G queries respectively for the hash oracles H and G from $\mathcal{A}$ and k is a security parameter.*

Proof. Our reduction $\mathcal{R}$ behaves as follows:

1. $\mathcal{R}$ is given ($n \leftarrow RSA(1^k)$, generates at random $t \leftarrow [0, n-1] \cap \mathbb{N}_{odd}$ and $y \leftarrow \frac{\mathbb{Z}}{n\mathbb{Z}}^*$, as well as an attacker $\mathcal{A}$ that $(q_H, q_G, q_{sig}, \tau_{\mathcal{A}}, \varepsilon_{\mathcal{A}})$-solves EUF-BR- CMA(pqPFDH);
2. $\mathcal{R}$ simulates G_{key} and transmits some public key $p_{key} = n$ to $\mathcal{A}$;
3. $\mathcal{R}$ receives queries for G from $\mathcal{A}$: it will have to simulate G at most q_G times;
4. $\mathcal{R}$ receives queries for H from $\mathcal{A}$: it will have to simulate H at most q_H times;
5. $\mathcal{R}$ receives signature queries from $\mathcal{A}$: it will have to simulate a signing oracle at most q_{sig} times;
6. $\mathcal{A}$ outputs a forgery (m, a, S) for pqPFDH;
7. $\mathcal{R}$ simulates a verification of the forgery which is valid with probability $\varepsilon_{\mathcal{A}}$;
8. $\mathcal{R}$ outputs x such that $x^t = y \mod n$.

Simulation of oracle key generation $\mathbf{G}_{key}$: The reduction $\mathcal{R}$

- sets $Hist(S) = \emptyset$ (Signing oracle database);
- sets $Hist[G] = \emptyset$ (Oracle $\mathbf{G}$ database);
- sets $Hist[H] = \emptyset$ (Oracle $\mathbf{H}$ database);
- sends the pqPFDH public key n to $\mathcal{A}$;
- selects M (with $M < 2^{|a|}$) random integers $i_1, ..., i_M \in [1; 2^{|a|}]$ where $|a|$ is the size of random used for exponentiation in signature process and sets $L = \{i_1, ..., i_M\}$.

Simulation of oracle hash $\mathbf{G}$: when $\mathcal{A}$ queries G with a random a_j, $1 \leq j \leq L$.

- $\mathcal{R}$ checks in $Hist[G]$, if a_j was queried in the past. If $e_j = G(a_j)$, is already defined to a value $e_j = G_{a_j}$, returned this value.
- if $j \notin L$ $\mathcal{R}$ picks at random $g_{a_j} \leftarrow [0, n-1] \cap \mathbb{N}_{odd}$ and defines $e_j = g_{a_j}$
- if $j \in L$: $\mathcal{R}$ picks at random $g'_{a_j} \leftarrow [0, n-1] \cap \mathbb{N}_{odd}$ and defines $e_j = tg'_{a_j}$
- memorizes $\big(a_j, e_j\big)$ in $Hist[G]$

Simulation of oracle hash H: when $\mathcal{A}$ queries H with message and a random (m, a_j),

- $\mathcal{R}$ invokes its own simulation to compute $e_j = G(a_j)$,
- checks in $Hist[H]$, if $e_j||m||a_j$ was queried in the past. If $H(e_j||m||a_j)$ is already defined as $h_{e_j||m||a_j}$, returned $h_{e_j||m||a_j}$;
- if $j \notin L$, $\mathcal{R}$;
  - picks λ_{a_j} at random;
  - defines and returns $H(e_j||m||a_j) = \lambda_{a_j}^{e_j} \mod n$ to $\mathcal{A}$;
  - memorizes $(m, a_j, e_j, \lambda_{a_j}, \lambda_{a_j}^{e_j})$ in $Hist[H]$
- if $j \in L$, $\mathcal{R}$;
  - picks λ_{a_j} at random;
  - defines and returns $H(e_j||m||a_j) = \lambda_{a_j}^{e_j} y \mod n$ to $\mathcal{A}$;
  - memorizes $(m, a_j, e_j, \perp, y)$ in $Hist[H]$;

Simulation of oracle signature $\mathbf{S^{G,H}}$: when $\mathcal{A}$ requests the signature of some message m, $\mathcal{R}$,

- selects randomly j in $[1, 2^{|a|}] \setminus L$, after $\mathcal{R}$;
- invokes its own simulation of G and H to compute $e_j = G(a_j)$ and $H(e_j||m||a_j)$;
- search the unique $(m, a_j, e_j, \alpha, \beta)$ in $Hist[H]$ and returns (m, a_j, α);
- store (m, a_j, α) in oracle database $Hist[S]$.

Simulation of $\mathbf{V^{G,H}}$: Given (m, a, S), $\mathcal{R}$

- invokes its own simulation of H and G to get $e = G(a)$ and $H(e||m||a)$;
- outputs 1 if $H(e||m||a) = S^e(\mod n)$ and $(m, a, S) \notin Hist(S)$ or 0 otherwise.

Final Outcome: assume that at the end of the game, $\mathcal{A}$ outputs (m^*, a_d, S^*) as a forgery. Then,

- $\mathcal{R}$ simulates $V^{G,H}$ to verify if (m^*, a_d, S^*) is a valid forgery which means that $H(e_d||m||a_d) = S^{e_d}(\mod n)$ and $(m^*, a_d, S^*) \notin Hist(S)$ (this last condition allows to force the attacker to output his own forgery instead of using a signature of the real signer as forgery);

- if (m^*, a_d, S^*) is invalid, $\mathcal{R}$ aborts;
- if $d \notin L$, $\mathcal{R}$ aborts;
- $\mathcal{R}$ sets $x = \left(\frac{S^*}{\lambda_{a_d}}\right)^{e_d/t}$;
- $\mathcal{R}$ outputs x.

Tightness of this reduction:

- $\mathcal{R}$ perfectly simulates the scheme (oracles key generation, hash, signature and verification) with probability 1.
- $\mathcal{A}$ then outputs (m^*, a_d, S^*) with probability at least $\varepsilon_{\mathcal{A}}$ after time $\tau_{\mathcal{A}}$,
- since L is independent from $\mathcal{A}$, the event $d \in L$ occurs with probability $\frac{M}{2^{|a|}}$.
- Hence $\mathcal{R}$ then outputs a solution $(x; y)$ for $RSA[n; e; y]$ with probability one.

Summing up, $\mathcal{R}$ succeeds with probability $\varepsilon_{\mathcal{R}} = \varepsilon_{\mathcal{A}} \frac{M}{2^{|a|}}$ and time bound is given by $\tau_{\mathcal{R}} \leq \tau_{\mathcal{A}} + q_G\mathcal{O}(1) + (q_H + q_{sig} + 2)\mathcal{O}(k^3)$ where $|a|$ is the size of random used for exponentiation in pqPFDH signature scheme.

To have $\varepsilon_{\mathcal{R}} \geq \varepsilon_{\mathcal{A}}(1 - \frac{1}{2^{|a|}})$ it will suffice to choose the maximal value of $M = 2^{|a|} - 1$.

□

4 Conclusion

We have described a new signature scheme whose security can be proven in the random oracle model for both quantum and classical computers. This is a variant of PFDH designed by Coron but with a random salt generated to compute the RSA exponentiation for each signature. Our security reduction is a better security proof for PFDH, in which a much shorter random salt (a random of size 5) is sufficient to achieve the same security.

References

[1] P. Shor, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, SIAM J. Comput., 26 (5), 1997, pp.1484-1509.

[2] M. Bellare and P. Rogaway, *Random oracles are practical: a paradigm for designing efficient protocols*. Proceedings of the First Annual Conference on Computer and Communications Security, ACM, 1993.

[3] M. Bellare and P. Rogaway, *The Exact security of digital signatures: How to sign with rsa and Rabin*, Proceedings of Eurocrypt 1996, lncs, vol. 1070, Springer-Verlag, 1996, pp. 399-416.

[4] R. Canetti, O. Goldreich and S. Halevi, *The random oracle methodology*, revisited, STOC' 98, ACM, 1998.

[5] J. S. Coron *On the exact security of Full Domain Hash* Proccedin of Crypto's 2000, LNCS vol 1880, Springer Verlag 2000, pp. 229-235.

[6] J.S. Coron, *Optimal security proofs for pss and other signature schemes*, Proceed- ings of Eurocrypt'02, lncs, vol. 2332, Springer-Verlag, 2002, pp. 272-287.

[7] J.S. Coron, A. Joux, D. Naccache and P. Paillier, *Fault Attacks on Randomized RSA Signatures*. To appear at CHES 2009. Available at http://www.jscoron.fr/publications.html.

[8] J.S. Coron and A. Mandal, *PSS is Secure against Random Fault Attacks* vavilable at Conference: Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings.

[9] Y. Dodis and L. Reyzin, *On the power of claw-free permutations*. In Security in Communication Networks (SCN 2002), volume 2576 of Lecture Notes in Computer Science, pages 55-73. Springer, 2003.

[10] Y. Dodis, R.Oliveira and K. Pietrzak, *On the Generic Insecurity of Full-Domain Hash*. Advances in Cryptology-CRYPTO 2005. 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005. Proceedings.

[11] S. Goldwasser, S. Micali and R. Rivest, *A digital signature scheme secure against adaptive chosen-message attacks*, SIAM Journal of computing, 17(2), pp. 281-308, April 1988.

[12] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996.

[13] D. Pointcheval, *Provable Security for Public Key Schemes* Advanced Course on Contemporary Cryptology, pages 133-189, June 2005. http://www.di.ens.fr/ pointche/pub.php?reference=Po04.

[14] D. Pointcheval and J. Stern, *Security Proofs for Signature Schemes* Advances in Cryptology Proceedings of EUROCRYPT '96 (may 12-16, 1996, Zaragoza, Spain) U. Maurer, Ed. Springer-Verlag, LNCS 1070, pages 387-398.

[15] R. Rivest, A. Shamir and L. Adleman, *A method for obtaining digital signatures and public key cryptosystems*, CACM 21, 1978.

[16] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Chapman & Hall/CRC Cryp-tography and Network Security. Chapman & Hall/CRC, Boca Raton, FL, 2008

[17] Dan Boneh and al., *Random Oracles in a Quantum World*. International conference on the theory and Application of Cryptology and Information Security. ASIACRYPT 2011. Springer, pages 41-69.

[18] Jonathan Katz and Nan Wang. *Efficiency improvements for signature schemes with tight security reductions*, Proceedings of the 10th ACM conference on Computer and Communication security- CCS “03, page 155, 2003.