Post-quantum verifiable random functions from ring signatures

Endre Abraham

December 22, 2018

Abstract

One of the greatest challenges on exchanging seemingly random nonces or data either on a trusted or untrusted channel is the hardness of verifying the correctness of such output. If one of the parties or an eavesdropper can gain game-theoretic advantage of manipulating this seed, others cannot efficiently notice modifications nor accuse the oracle in some way. Decentralized applications where an oracle can go unnoticed with biased outputs are highly vulnerable to attacks of this kind, limiting applicability of these parties even though they can introduce great scalability to such systems. Verifiable random functions[1] presented by Micali can be viewed as keyed hash funcions where the key(s) used are asymmetric. They allow the oracle to prove correctness of a defined pseudorandom function on seed s without actually making it public, thus not compromising the unpredictability of the function. Our contribution here is to provide a variant of this scheme and proving it's security against known quantum attacks and quantum oracles.

1 Introduction

1.1 VRFs

Verifiable random functions are a collection of polynomial-time algorithms G, E, P, V where:

- $G(\lambda) := (sk, pk)$ a key generator
- $E(sk, m)$ an evaluator of the pseudorandom function at subject
- $P(sk, m) := \alpha$ a proof generator
- $V(pk, m, \alpha)$ a verifier

This protocol ensures that everyone can verify the output of the pseudorandom function on a given input but only the holder of the secret key can generate the same output.

1.2 Primer to quantumcomputing

A quantum system A is associated to a (finite-dimensional) complex Hilbert space H_a with an inner product $\langle\cdot|\cdot\rangle$. The state of the system is described by a vector $|\varphi\rangle \in H_a$ such that the Euclidean norm $\||\varphi\rangle\| = \sqrt{\langle\varphi|\varphi\rangle}$ is 1. Given quantum systems A and B over spaces H_a and H_B, respectively, we define the joint or composite quantum system through the tensor product $H_A \otimes H_B$. The product state of $|\varphi_A\rangle \in H_A$ and $|\varphi_B\rangle \in H_B$ is denoted by $|\varphi_A\rangle \otimes |\varphi_B\rangle$ or simply $|\varphi_A\rangle|\varphi_B\rangle$. An n-qubit system lives in the joint quantum system of n two-dimensional Hilbert spaces. The standard orthonormal computational basis $|x\rangle$ for such system is given by $|x_1\rangle \otimes \cdots \otimes |x_n\rangle$ for $x = x_1 \dots x_n$. Any (classical) bit string x is encoded into a quantum state as $|x\rangle$ An arbitrary pure n-qubit state $|\varphi\rangle$ can be expressed in the computational basis as $|\varphi\rangle = \Sigma_{x\in\{0,1\}}\alpha_x|x\rangle$ where α_x are complex amplitudes obeying $\Sigma_{x\in\{0,1\}^n}|\alpha_x|^2 = 1$.

2 Notation

In this paper we denote the finite field of integers modulo p with $\mathbb{Z}_p$ where p is a prime. We use $\mathbb{Z}_p[x]$ to denote polynomials with coefficients from this field. Let $\Phi(x)$ be an irreducible polynomial and $\mathbb{Z}_p[x]/\Phi(x)$ be a quotient ring with n nonzero coefficients. From now on, we assume n to be a power of 2 thus $\Phi(x) = x^n + 1$. We indicate uniform random sampling from a ring with $\xleftarrow{\$}$ and discrete gaussian sampling with $\xleftarrow{X}$.

2.1 RLWE and connection with SVP

RLWE[3] is an algebraic variant of the LWE[4] problem, both known to be reducible to the worst-case shortest vector lattice problem (thus implying resistance to polynomial-time quantum algorithms), with RLWE allowing much more efficient operations. The RLWE problems consist of public polynomials $a_i(x)$ with uniform distribution, private $b_i(x)$ and $s(x)$ where the coefficients of all b_i and s are relative to a public agreed bound either using uniform or discrete gaussian distribution such that $||b_i(x)||_\infty < b$. We advise using the later, for it's easier to prove correctness of parameter selection with gaussian distributions. We see that if the bound is selected correctly, $c_i(x) \approx a_i(x) \cdot s(x)$. Using the above polynomials we construct $c_i(x) = a_i(x) \cdot s(x) + b_i(x)$. We call the pair $a_i(x), c_i(x)$ a sample. The Decisional RLWE problem states that with access to many RLWE samples, it's impossible to differentiate whether a $c_i(x)$ was constructed as above, or randomly selected from $\mathbb{Z}_p[x]/\Phi(x)$, except with negligible probability.

Formally, $Pr[\, c(x) \,|\, q(x) \xleftarrow{\$} \mathbb{Z}_p[x]/\Phi(x) \,] < \frac{1}{2^\lambda}$

The Computational RLWE problem states that given many RLWE samples, it's impossible to find the single $s(x)$ used in all samples, except with negligible probability.

Peikert showed[3] that the search version is reducible to the problem of finding the approximate shortest vectors of ideal lattices in $\mathbb{Z}_p[x]/\Phi(x)$. We also know that at the time of writing there is no algorithm that uses the structural difference between ideal and regular lattices so for now it's safe to assume that no efficient algorithm can solve it for regular lattices either.

3 The Franklin-Zhang signature scheme

The starting VRF is a generalization of a ring signature scheme based on the Chaum-Pedersen proof of logarithm equalty[5]:
$s = log_g(h) \in \mathbb{Z}_p$ is known to the prover and $g, h \neq 1, y = g^s, y = h^s$ are public inputs. Prover wants to prove that $log_g(h) = log_m(z)$ where m is the message mapped into $\mathbb{Z}_p$ and $z = m^s$.

1. Prover sets a nonce $r \xleftarrow{\$} \in \mathbb{Z}_p$ and sends $a = g^r, b = g^r$
2. Verifier sends back a challenge $c \xleftarrow{\$} \in \mathbb{Z}_p$
3. Prover responds with $p = r - cx \mod p$
4. Verifier accepts the proof if $a = g^t y_1^c$ and $b = h^t y_2^c$

We see that this (naive) Chaum Pedersen HVZK can be transformed to a NIZK with the Fiat-Shamir heuristic[6] using a pseudorandom hash of the state after step 1. Let H_1 be our random oracle, a function from $\{0,1\}^* \to \mathbb{Z}_p$, H_2 a lenght-regular function of $\frac{\lambda}{2}$ bit output where λ is the security parameter, and H_3 a general purpose cryptographic hash, for eg SHA256. Let x be the input to H_1, and y be it's output. For now the only way to provide a witness for the PRF and proving it's correctness is to send x what we want to avoid. Instead we raise set h in the Chaum-Pedersen scheme to y^s where s is our private key, and apply the Fiat-Shamir transform to c with H_3. The actual output of this VRF function is the proof (y^s, c, p) and the output of $H_2(y^s)$. Now everyone can compute the VRF hash based on the proof only.

4 RLWE masking and reconcilliation

Starting with the scheme in the previous section we need to grasp the DDH assumption with RLWE as shown in [7]:
Let $dbl(x) : \mathbb{Z}_p \to \mathbb{Z}_{2p}; x \mapsto dbl(x) = 2x - e$ where e is sampled from $\{-1, 0, 1\}$ with distribution: $p_{-1} = p_1 = \frac{1}{4}, p_0 = \frac{1}{2}$. We will use the rounding and cross-rounding functions from [8]:

1. modular rounding: $\lfloor x \rceil_{p,2} : \mathbb{Z}_p \to \mathbb{Z}_2, x \mapsto \lfloor x \rceil_{p,2} = \lfloor \frac{p}{2} x \rceil \mod 2$
2. cross rounding: $\langle x \rangle_{p,2} : \mathbb{Z}_p \to \mathbb{Z}_2, x \mapsto \langle x \rangle_{p,2} = \lfloor \frac{4}{p} x \rfloor \mod 2$

It is proven that if x is uniformly random then the double rounding of $dbl(x)$ is also uniform random[8].

We also need a reconciliation function to get back $\lfloor x \rceil_{p,2}$ from a ring element.
Let $I_0 = \{0, 1, \ldots, \lfloor \frac{p}{2} \rceil - 1\}, I_1 = \{-\lfloor \frac{p}{2} \rfloor, \ldots, -1\}$ and $E = [-\frac{p}{4}, \frac{p}{4})$. Then $rec : \mathbb{Z}_{2p} \times \mathbb{Z}_2$ is:

$$rec(w, b) = \begin{cases} 0 & \text{if } w \in I_b + E \mod 2p \\ 1 & \text{otherwise} \end{cases}$$

Given $a \xleftarrow{\$} \mathbb{Z}_p[x]/\Phi(x)$, s_1, s_2, e_1, e_2, $e_3 \xleftarrow{X} \mathbb{Z}_p[x]/\Phi(x)$, $b_1 = as_1 + e_1$, $b_2 = as_2 + e_2$, $v = dbl(b_1 s_2 + e_3)$, $c = \langle v \rangle_{2p,2}$, $k_1 = \lfloor v \rceil_{2p,2}$ and $k_2 \xleftarrow{\$} \{0,1\}^n$, a probabilistic polynomial adversary $\mathcal{A}$ has negligible advantage differentiating k_1 from k_2 with access to RLWE samples:
$Adv(\mathcal{A}) = |Pr(\mathcal{A}(a, b_1, b_2, c, k_1) = 1) - Pr(\mathcal{A}(a, b_1, b_2, c, k_2) = 1| < \frac{1}{2} + \frac{1}{\lambda}$
We see that if $v = w + e \in \mathbb{Z}_p | 2e \pm 1 \in E$ then $rec(2w, \langle v \rangle_{2p,2}) = \lfloor v \rceil_{2p,2}$ Cancelling out error coefficients in a polynomial is done with repeated reconcillation on all coefficients.

5 Quantum random oracles

Proving correctness of a traditional random oracle against quantum adversaries is much harder then replacing our trapdoors with a quantum-resistant variant. For example traditional RO security is relying on the fact that the adversary gets the same output for a query with the exact same input, thus with forcing polynomially many k random queries in our protocol our adversary have $\frac{1}{k}$ chance of forging. This is however not the case with quantum oracles where the adversary can input polynomially many (quantum)states into our oracle with one query.
One way of proving security in the quantum oracle model happens with reducing a classical RO scheme to a so-called history-free version[9].
A signature scheme with access to a classical oracle Q_c is history free if one can prove it's classical RO security with the following 5 algorithms:

1. $GEN(x) \to (pk, z)$: on a problem instance x returns a public key pk and a private state z
2. $INST(pk)$ outputs an instance x such that $GEN(x) = (pk, z)$
3. $RAND(r, z)$ used every time when the adversary queries $Q_c(r)$
4. $SIGN(m, z)$ used when the adversary ask for signature on pk, m
5. $FINISH(m, \sigma, z)$ used on forgery candidate input (m, σ)

We see that shortly a history-free design "forgets" our previous queries to the oracle by using an instance-specific private quantum state z and appends it everywhere we need to model security with a classical oracle.

6 Our scheme

6.1 Prover side

With all the tools ready, we now present our post-quantum VRF.
Prover has access to $H_1, H_2, H_3, s, a(x), b_i(x)$ as defined above and wants to prove that $y = H_1(\alpha)$ is correct without distributing α. First it obtains $h = H_1(\alpha)$. Then it maps h into $h(x) \in \mathbb{Z}_p[x]/\Phi(x)$ and masks it with $\gamma = h(x) \cdot s(x) + b(x)$. It's trivial that H_1 should be pseudorandom for here we differ from the initial sampling requirement of RLWE. Next it selects $k \xleftarrow{\$} \mathbb{Z}_p[x]/\Phi(x)$ and compute $c = H_3(a(x), h, a(x) \cdot s(x) + b_1(x), \gamma, a(x) \cdot k(x) + b_2(x), h(x) \cdot k(x) + b_3(x))$.
To let the verifiers recover $a(x) \cdot k(x)$ and $h(x) \cdot k(x)$ we provide the rounding informations needed for the reconcilliation:

1. $r_1(x) = k(x) - (a(x) \cdot s(x) \cdot c(x) + b_1(x))$
2. $r_2(x) = \langle dbl(a(x) \cdot k(x) + b_2(x)) \rangle_{2p,2}$
3. $r_3(x) = \langle dbl(h(x) \cdot k(x) + b_3(x)) \rangle_{2p,2}$

Finally, it sends $\pi = (\gamma, c, r_1, r_2, r_3, H_2(\gamma))$ to the verifier

6.2 Verifier side

On the verifier end we have the prover's public key $pk = (a(x) \cdot s(x) + b(x))$, α and π from above. First we compute the output of H_1 same as the prover: $h = H_1(\alpha)$ and maps it into our ring.
Then we recover informations masked by the k nonce polynomial as:
$u = r_1 \cdot \sum_{i=1}^{n} rec(2pk \cdot c(x), r_2) x^i$.
$v = r_1 \cdot \sum_{i=1}^{n} rec(2h(x) \cdot c(x), r_3) x^i$
We see that if everything went correctly the verifiers u and v should equal to the prover's $a(x) \cdot k(x) + b_2(x)$ and $h(x) \cdot k(x) + b_3(x)$ respectively since the coefficient-wise reconcilliation should cancel out the errors.
Then the verifier accepts the proof if $c = H_3(a(x), h, pk, \gamma, u, v)$ and $H_2(\gamma)$ is the same provided in π.

6.3 History-free reduction

In the reduction we use two definitions, Full Domain Hashes and Preimage Sampleable Functions. More precisely we instantiate a FDH using a PSF as seen in [10].
PSFs are efficiently computable functions $f : X \to Y$ where the pair $x, y = f(x)$ has the same joint distribution regarding a negligible error e, and given y one can

sample x using f^{-1}. Note that this is contradictory one-wayness and collision resistance!

Full domain hashes: Let G, f, f^{-1} be a trapdoor permutation and O a surjective hash into f. An FDH is a scheme where signature of m using secret key sk and O is defined as $f^{-1}(sk, O(m))$ and verification of pk, m, r (where r is a public coin) is true if and only if $O(m) = f(pk, r)$.

To achieve a history-free reduction we instantiate a FDH-PSF using the strategy seen in section 5.

1. $pk := a(x) \cdot s(x) + b(x)$
2. $r \xleftarrow{X} \mathbb{Z}_p[x]/\Phi(x)$
3. $GEN(pk) := (pk, pk)$
4. $INSTANCE(pk) := pk$
5. On an adversarial query for $O(r)$, $RAND(r, pk) := f(pk, Sample(1^n; O(r)))$
6. $SIGN(sk, m) := Sample(1^n; O(m))$
7. $FINISH^O * (m, pk, r) := (Sample(1^n; O(m)), r)$

Proof of history-free security of this scheme is in [10]

7 Conclusion

We showed how to transform a linkable ring signature scheme into a verifiable random function using RLWE and proved it's security against quantum oracles with the assumptions that history-free reductions are generally safe. Our scheme can be used to detect adversarial behaviour of trusted parties, what can be useful in mixnets and distributed networks. It's trivial to see how this scheme can be transformed to a designated verifier setup since the reconcilliation functions is originally used in key-exchange protocols. An open question about this construction is whether it's possible to make a history-free reduction on top of other existing error-cancellations methods of RLWE polynomials, and whether it undermines current usecases (for e.g. designated verifiers) or scalability.

References

[1] S. Micali, S. Vadhan, and M. Rabin, "Verifiable random functions," in *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, FOCS '99, (Washington, DC, USA), pp. 120–, IEEE Computer Society, 1999.

[2] M. Franklin and H. Zhang, "Unique ring signatures: A practical construction," in *Financial Cryptography and Data Security* (A.-R. Sadeghi, ed.), (Berlin, Heidelberg), pp. 162–170, Springer Berlin Heidelberg, 2013.

[3] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings." Cryptology ePrint Archive, Report 2012/230, 2012. https://eprint.iacr.org/2012/230.

[4] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05, (New York, NY, USA), pp. 84–93, ACM, 2005.

[5] D. Chaum and T. P. Pedersen, "Wallet databases with observers," in *Advances in Cryptology — CRYPTO' 92* (E. F. Brickell, ed.), (Berlin, Heidelberg), pp. 89–105, Springer Berlin Heidelberg, 1993.

[6] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Proceedings on Advances in cryptology—CRYPTO '86*, (London, UK, UK), pp. 186–194, Springer-Verlag, 1987.

[7] J. W. Bos, C. Costello, M. Naehrig, and D. Stebila, "Post-quantum key exchange for the tls protocol from the ring learning with errors problem." Cryptology ePrint Archive, Report 2014/599, 2014. https://eprint.iacr.org/2014/599.

[8] C. Peikert, "Lattice cryptography for the internet." Cryptology ePrint Archive, Report 2014/070, 2014. https://eprint.iacr.org/2014/070.

[9] D. B. andOzur Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry, "Random oracles in a quantum world." Cryptology ePrint Archive, Report 2010/428, 2010. https://eprint.iacr.org/2010/428.

[10] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, (New York, NY, USA), pp. 197–206, ACM, 2008.