Post-quantum Cryptography

(September 2022)

Dr. Ratna Dutta
Associate Professor, IIT Kharagpur

Department of Mathematics
IIT Kharagpur

Why Post Quantum Crypto?

- Quantum computers will break the most popular public-key cryptosystems:
  - RSA.
  - DSA.
  - ECDSA.
  - ECC.
  - HECC, $\cdots$

  can be attacked in polynomial time using Shor's algorithm.

- Post Quantum Crypto brings solution to this threat.

Post Quantum Crypto

Post-quantum cryptography deals with cryptosystems that

- run on conventional computers
- are secure against attacks by quantum computers.

Examples:

- Hash-based cryptography.
- Code-based cryptography.
- Lattice-based cryptography.
- Multivariate-quadratic-equations cryptography.
- Isogeny-based cryptography.

Lattice-based hash function with two inputs

Let n, q, k, m positive integers where: n is the security parameter; $q = O(n)$; $k = \lfloor \log q \rfloor$, $m = 2nk$ and $\mathbb{Z}_q = \{0, \ldots, q-1\}$. The "powers-of-2" matrix is defined by

$$G = \begin{bmatrix} 1\ 2\ 4 \ldots\ 2^{k-1} & & & \\ & 1\ 2\ 4 \ldots\ 2^{k-1} & & \\ & & \ldots & \\ & & & 1\ 2\ 4 \ldots\ 2^{k-1} \end{bmatrix}$$

A Family of Lattice-Based Collision-Resistant Hash Function

The function family $\mathcal{H}$ mapping from $\{0,1\}^{nk} \times \{0,1\}^{nk}$ to $\{0,1\}^{nk}$ is defined as $\mathcal{H} = \{h_A | A \in \mathbb{Z}_q^{n \times m}\}$, where for $A = [A_0 | A_1]$ with $A_0, A_1 \in \mathbb{Z}_q^{n \times nk}$, and for any $(\mathbf{u}_0, \mathbf{u}_1) \in \{0,1\}^{nk} \times \{0,1\}^{nk}$, we have

$$h_A(\mathbf{u}_0, \mathbf{u}_1) = \mathrm{bin}(A_0 \cdot \mathbf{u}_0 + A_1 \cdot \mathbf{u}_1 \pmod{q})$$

Note that, $h_A(\mathbf{u}_0, \mathbf{u}_1) = \mathbf{u} \iff A_0 \cdot \mathbf{u}_0 + A_1 \cdot \mathbf{u}_1 = G \cdot \mathbf{u} \pmod{q}$

Lattice-based Hardness Assumption

Short integer solution(SIS) problem

The $SIS^{\infty}_{n,m,q,\beta}$ problem is as follows: Given uniformly random matrix $A \in \mathbb{Z}_q^{n \times m}$, find a non-zero vector $x \in \mathbb{Z}^m$ such that $||x||_\infty \leq \beta$ and $A \cdot x = 0 \pmod{q}$.

dRLWE problem

Let $q, m, n, \sigma > 0$ depend on security parameter (q, m, n are integers). The decision-RLWE problem ($dRLWE_{q,n,m,\sigma}$) is to distinguish between: $(a_i, a_i \cdot s + e_i)_{i \in [m]} \in (R_q)^2$ and $(a_i, u_i)_{i \in [m]} \in (R_q)^2$ for $a_i, u_i \leftarrow R_q$ and $s, e_i \leftarrow R(\chi_\sigma)$.

Merkle-Tree Accumulator

Acc.Setup(λ) $\longrightarrow$ $\mathsf{param}_{\mathsf{Acc}}$

Sample $A \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and set $\mathsf{param}_{\mathsf{Acc}} = A$. All the following algorithm takes input $\mathsf{param}_{\mathsf{Acc}}$ implicitly.

Acc.Accumulate($\mathcal{R} = \{\mathbf{d}_0 \in \{0,1\}^{nk}, \ldots, \mathbf{d}_{N-1} \in \{0,1\}^{nk}\}$) $\rightarrow \mathbf{u}$

(i) For $j = 1, 2, \ldots, N-1$, represent $\mathbf{d}_j \in \{0,1\}^n$ as $\mathbf{u}_{j_1, j_2, \ldots, j_l}$ for the binary representation $(j_1, j_2, \ldots, j_l) \in \{0,1\}^l$ of j where $l = \lceil \log N \rceil$.

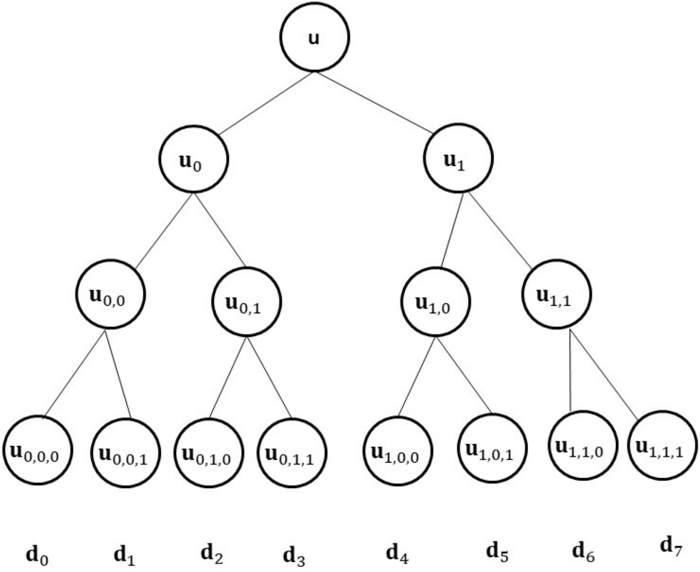
(ii) Build a binary tree with $N = 2^l$ leaves $\mathbf{u}_{0,0,\ldots,0}$, $\ldots$, $\mathbf{u}_{1,1,\ldots,1}$ and compute the accumulated value

$$\mathbf{u}_{a_1, a_2, \ldots, a_i} = h_A(\mathbf{u}_{a_1, a_2, \ldots, a_i, 0}, \mathbf{u}_{a_1, a_2, \ldots, a_i, 1})$$

for the nodes with values $\mathbf{u}_{a_1, a_2, \ldots, a_i, 0}, \mathbf{u}_{a_1, a_2, \ldots, a_i, 1} \in \{0,1\}^n$ and for all $(a_1, a_2 \ldots, a_i) \in \{0,1\}^i$ at depth $i = 1, 2, \ldots, l-1$.

(iii) At depth 0, compute the accumulated value $\mathbf{u} = h_A(\mathbf{u}_0, \mathbf{u}_1)$ at root for the node values at $\mathbf{u}_0, \mathbf{u}_1 \in \{0,1\}^n$.

Merkle-Tree Accumulator: Acc.Accumulate



Merkle-Tree Accumulator

$\mathsf{Acc.WitGen}(\mathsf{param}_{\mathsf{Acc}}, \mathcal{R}, \mathbf{d}) \longrightarrow \mathsf{wit}$

(i) If $\mathbf{d} \notin \mathcal{R}$, output $\perp$.

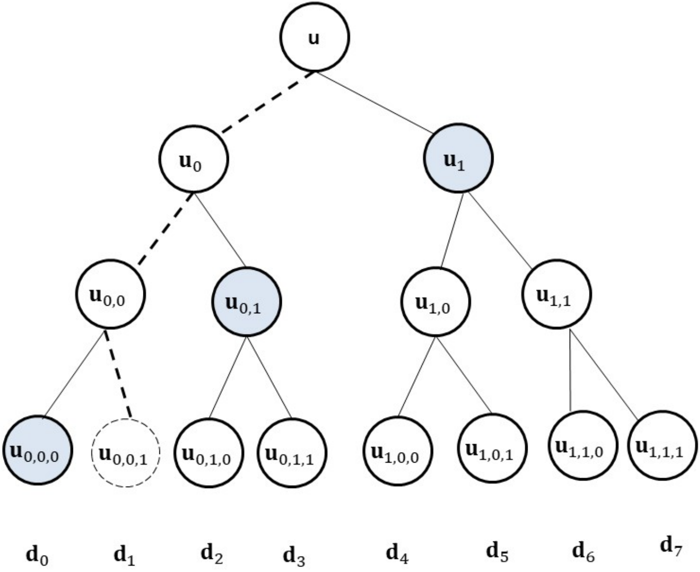
(ii) If $\mathbf{d} \in \mathcal{R}$, then $\mathbf{d}_j = \mathbf{d}$ for some j.

(iii) Represent $\mathbf{d}_j$ as $\mathbf{u}_{j_1,j_2,\ldots,j_l}$ where $(j_1, j_2, \ldots, j_l)$ is the binary representation of j.

(iv) Set the witness

$$\mathsf{wit} = ((j_1, j_2, \ldots, j_l), (\mathbf{w}_l, \mathbf{w}_{l-1}, \ldots, \mathbf{w}_1)) \in \{0,1\}^l \times (\{0,1\}^n)^l$$

Merkle-Tree Accumulator: Acc.WitGen



Merkle-Tree Accumulator

Acc.Verify($\mathbf{u}, \mathbf{d}, \text{wit} = ((j_1, j_2, \ldots, j_l), (\mathbf{w}_l, \mathbf{w}_{l-1}, \ldots, \mathbf{w}_1))) \longrightarrow 0/1$

Find the values $\mathbf{v}_l, \mathbf{v}_{l-1} \ldots, \mathbf{v}_0$ by setting $\mathbf{v}_l = \mathbf{d}$ and computing

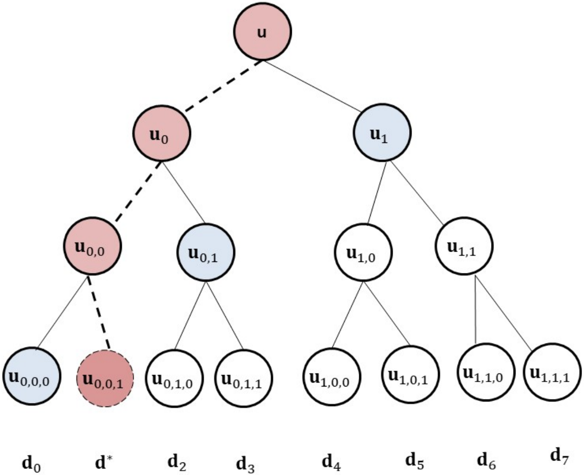
$$\mathbf{v}_i = \begin{cases} h_A(\mathbf{v}_{i+1}, \mathbf{w}_{i+1}) & \text{if } j_{i+1} = 0 \\ h_A(\mathbf{w}_{i+1}, \mathbf{v}_{i+1}) & \text{if } j_{i+1} = 1 \end{cases}$$

for $i = l - 1, \ldots, 1, 0$.
If $\mathbf{v}_0 = \mathbf{u}$, output 1, otherwise return 0.

Lemma

The given accumulator scheme is secure assuming the hardness of the $SIS^{\infty}_{n,m,q,\beta}$ problem.

Merkle-Tree Accumulator: Acc.Verify



Pseudo-random Function (PRF)

Setup $(\lambda) \longrightarrow (\mathbf{b}_0, \mathbf{b}_1)$

Let $R = \frac{\mathbb{Z}[X]}{<X^n+1>}$ and $R_q := R/qR$ and $l = \lceil \log_2 q \rceil$. Fix some $\mathbf{b}_0, \mathbf{b}_1 \in R_q^{1\times l}$. Ten set *params* $= (\mathbf{b}_0, \mathbf{b}_1)$. All the following algorithm takes *params* implicitly.

KeyGen$(\lambda) \longrightarrow k$

The key generation algorithm samples k from $R_q(\chi_\sigma)$, where χ_σ discrete Gaussian distribution over R_q with parameter σ .

Evaluation $(\mathbf{x}) \longrightarrow F_k(\mathbf{x})$

For input $\mathbf{x} = (x_1, \ldots, x_L) \in \{0,1\}^L$. Define

$$\mathbf{b_x} := \mathbf{b}_{x_1}.G^{-1}(\mathbf{b}_{x_2}.G^{-1}(\mathbf{b}_{x_3}.G^{-1} \ldots (\mathbf{b}_{x_{L-1}}.G^{-1}(b_{x_L})))) \in R_q^{1\times l}$$

PRF is defined as

$$F_k(\mathbf{x}) = \lceil \mathbf{b_x} \cdot k \rfloor_p = \lceil \frac{p}{q} \cdot \mathbf{b_x} \cdot k \rfloor$$

where $p|q$.

Pseudo-random Function (PRF)

Theorem

Sample $k \leftarrow R(\chi_\sigma)$. If $q >> p \cdot \sigma \cdot \sqrt{L} \cdot n \cdot l$, then the function $F_k(\mathbf{x}) = \lceil \mathbf{b}_\mathbf{x} \cdot k \rfloor_p$ is a PRF under the $dRLWE_{q,n,\sigma}$ assumption.

Signature Scheme by B. Libert et. al

Let λ be the security parameter. Let $m_1, m_2, m_3, \sigma, q, L$ be positive integers that are polynomial in λ . Let $k = \lfloor \log q \rfloor$. The signature scheme works as follows:

KeyGen$(\lambda) \rightarrow (sk = \mathbf{T},\, pk = (\mathbf{B}, \{\mathbf{B}_i\}_{i \in [0, m_3]}, \mathbf{u}, \tilde{\mathbf{B}}, \tilde{\mathbf{B}}_0, \tilde{\mathbf{B}}_1))$

On input the security parameter λ, the key generation algorithm first samples a random matrix $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{m_1 \times m_2}$ together with its trapdoor $\mathbf{T}$. Then it samples $\mathbf{B}_i \xleftarrow{\$} \mathbb{Z}_q^{m_1 \times m_2}$ for $i \in [0, m_3]$ and samples $\tilde{\mathbf{B}} \xleftarrow{\$} \mathbb{Z}_q^{m_1 \times m_2}$, $\tilde{\mathbf{B}}_0 \xleftarrow{\$} \mathbb{Z}_q^{m_1 \times 2m_2}$, $\tilde{\mathbf{B}}_1 \xleftarrow{\$} \mathbb{Z}_q^{m_1 \times L}$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^{m_1}$. Finally, it outputs $sk = \mathbf{T}$ and $pk = (\mathbf{B}, \{\mathbf{B}_i\}_{i \in [0, m_3]}, \mathbf{u}, \tilde{\mathbf{B}}, \tilde{\mathbf{B}}_0, \tilde{\mathbf{B}}_1)$.

Signature Scheme by B. Libert et. al

Sign $(sk = \mathbf{T}) \to (\boldsymbol{\tau}, \mathbf{v}, \mathbf{r})$

On input the secret key $\mathbf{T}$ and a message $\mathbf{m} \in \{0,1\}^L$, the sign algorithm first samples $\boldsymbol{\tau} \xleftarrow{\$} \{0,1\}^{m_3}$. Then it computes the matrix $\mathbf{B}_{\boldsymbol{\tau}} = (\mathbf{B}||\mathbf{B}_0 + \Sigma_{i=1}^{m_3}(\boldsymbol{\tau}[i] \cdot \mathbf{B}_i))$. Next, it samples $\mathbf{r} \xleftarrow{\$} D_\sigma^{2m_2}$ and computes $\mathbf{c} = \tilde{\mathbf{B}}_0 \cdot \mathbf{r} + \tilde{\mathbf{B}}_1 \cdot \mathbf{m}$. Then it uses the secret key $\mathbf{T}$ to samples a vector $\mathbf{v} \in D_\sigma^{2m_2}$ that satisfies $\mathbf{B}_{\boldsymbol{\tau}} \cdot \mathbf{v} = \mathbf{u} + \tilde{\mathbf{B}} \cdot \text{bin}(\mathbf{c})$. The signature is $(\boldsymbol{\tau}, \mathbf{v}, \mathbf{r})$.

Verify $(pk = (\mathbf{B}, \{\mathbf{B}_i\}_{i\in[0,m_3]}, \mathbf{u}, \tilde{\mathbf{B}}, \tilde{\mathbf{B}}_0, \tilde{\mathbf{B}}_1)) \to \{0,1\}$

On input the public key $(\mathbf{B}, \{\mathbf{B}_i\}_{i\in[0,m_3]}, \mathbf{u}, \tilde{\mathbf{B}}, \tilde{\mathbf{B}}_0, \tilde{\mathbf{B}}_1)$, a message $\mathbf{m}$ and a signature $(\boldsymbol{\tau}, \mathbf{v}, \mathbf{r}) \in \{0,1\}^{m_3} \times \mathbb{Z}^{2m_2} \times \mathbb{Z}^{2m_2}$, the verification algorithm first computes $\mathbf{B}_{\boldsymbol{\tau}} = (\mathbf{B}||\mathbf{B}_0 + \Sigma_{i=1}^{m_3}(\boldsymbol{\tau}[i] \cdot \mathbf{B}_i))$. Then it checks if $\mathbf{B}_{\boldsymbol{\tau}} \cdot \mathbf{v} = \mathbf{u} + \tilde{\mathbf{B}} \cdot \text{bin}(\tilde{\mathbf{B}}_0 \cdot r + \tilde{\mathbf{B}}_1 \cdot \mathbf{m})$ and $\mathbf{v}, \mathbf{r}$ are short vectors.


Thank you