Post-Quantum Signatures

Lise Millerjord

30.01.2020

Main topic

- Integrated PhD in cryptography
  - Started november 2019
  - Master thesis submission in 2020
- Lightweight cryptography for future smart networks
  - 5G
  - IoT
  - ...

PQ Signatures and Key Exchange

Preliminary focus: PQ signatures and key exchange

Today:

- Picnic - Round 2 NIST PQ

Goal:

- Understand how Picnic works
- Study design choices
- Do more research!

Picnic Design

- PQ signature system
- Not number theory
- Symmetric key primitive used to generate key pair
- NIZK-proof from Σ-protocol
  - Hash $\rightarrow$ Challenge $\rightarrow$ Response system
  - Unruh's transform applied to make it non-interactive
  - Challenge generated by hashing the provers commitment with the message

Security of Picnic:
Reduction to symmetric key primitive

Design Choices

- One-way function
  - Public key is the image of the secret key under a one-way function
  - Practical to use symmetric ciphers in the place of this one-way function
  - Choice: **LowMC** due to it's low multiplicative complexity

- Σ-protocol
  - **ZKBoo** $\rightarrow$ **ZKB++**

- Non-interactive transform
  - Two possibilities to make the proof noninteractive are considered:
  - Fiat-Shamir: Secure in ROM, smaller signatures
  - **Unruh's transform:** Secure in QROM, large signatures. Optimisations reduce overhead to 1.6x compared to Fiat-Shamir

ZKBoo and ZKB++

The zero knowledge proofs used are based on a construction called **ZKBoo**:

- Based on MPC-in-the-head, implementing the relation $y = H(x)$
- Prover decomposes witness to 3 shares, with corresponding output shares and commits. Sends output share and commits to verifier
- Verifier chooses 2 of the outputs, asks for the corresponding witness shares and checks these.
- Constructed such that knowing any 2 shares gives no information about witness

ZKB++: The protocol used in Picnic is ZKBoo combined with several optimisations to reduce overhead.

How does Picnic work?

Key Generation:

- Generate a random plaintext block p and random secret key sk
- Compute $C = LowMC(sk, p)$, where sk must be hard to recover
- Picnic public key is $pk = (C, p)$, secret key is sk

Sign(sk, pk, m):

- Prove knowledge of sk such that $C = LowMC(sk, p)$
- Message m is bound to the proof when computing the challenge
- Picnic signature is the proof, which must be zero knowledge

Now what?

- Are these the ideal design choices?
- Can we use Fiat Shamir transform?
  - ROM vs QROM

Thank you for your attention! :-)