Benchmarking Quantum-Resistant Authentification in IoT

Clémentine Gritti
~~Postdoc~~ Research Fellow at Eurecom

Journées C2

19 October 2023

Who I am

- Currently a research fellow at Eurecom in France:
  - Fault-tolerant and asynchronous Secure Aggregation for privacy-preserving Federated Learning
- Previously a senior lecturer (maîtresse de conférences) at the University of Canterbury in New Zealand:
  - Research project between NZ and Australia on Post-Quantum Cryptography (PQC)

Today presentation

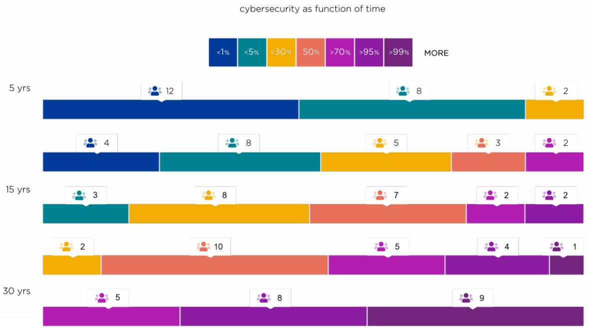
Ongoing research from my NZ-based PhD student:

- Personal context:
  - Research on PQC initiated in NZ from trans-Tasman project
  - Still the main supervisor
- International context:
  - PQC has attracted attention over the past few years
  - NIST standardisation

Plan

Quantum computing: a real threat?

- In 2 or 3 decades?
- IBM's 433-qubit Osprey Quantum Computer
- IBM has promised a 1,121-qubit processor in a near future



entrust.com/resources/certificate-solutions/learn/post-quantum-cryptography

Challenges

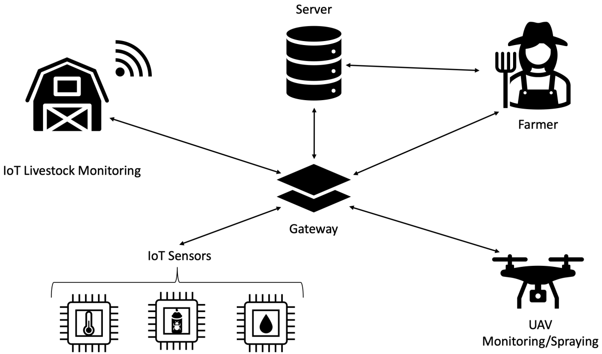
- Cryptographic algorithms built from maths problems seen as hard to solve:
  - Integer factorisation problem
  - Discrete log problem
- Those problems would be solved by a quantum computer:
  - Shor's algorithm
- Need for cryptographic algorithms considered as quantum resistant:
  - NIST standards: CRYSTALS-Kyber and -Dilithium, FALCON and SPHINCS+

PQC vs IoT

- Internet of Things:
  - Constrained resources (computation, communication and storage)
  - Low security
  - Simple design and heterogeneity
  - More and more devices and more and more manufacturers
- Post-Quantum Cryptography:
  - Bigger component sizes
  - Heavier computations
- Could we deploy PQC in IoT straightforwardly?

IoT use case

- Sensors sign their collected data
- The gateway verifies sensors' signatures
- The server manages the framework (e.g. key management)



Plan

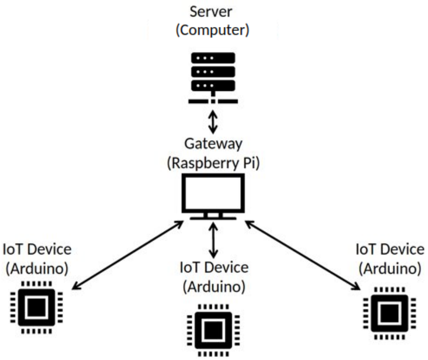
Our choice: CRYSTALS-Dilithium

- Based on lattices
- Being standardized
- Small components
- Good performance
- Why not FALCON?
  - Smaller parameter sizes but complex (floating-point) calculations
  - Error occurence and floating-point arithmetic implemented in software

Model of interaction

3-layer model:

- Device–gateway communication
- Gateway–server (cloud) communication



Implementation details

- Device and machine specification:
  - Device: Arduino Due
  - Gateway: Raspberry Pi 4 Model B
  - Server: computer Apple MacBook Pro
- Optimization specification:
  - Arduino Due: cortex-M
  - Raspberry Pi: Neon
  - Computer: Advanced Vector eXtensions 2 (AVX2)

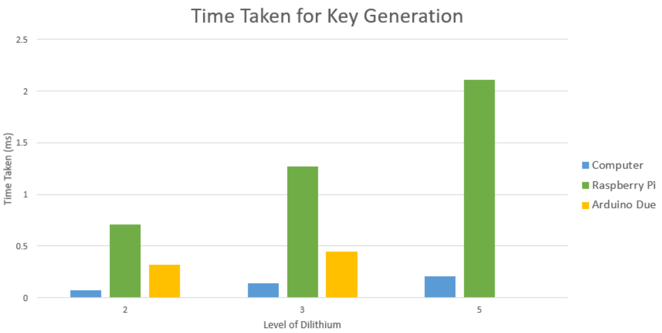
https://github.com/dilithium-cortexm/dilithium-cortexm
https://github.com/neon-ntt/neon-ntt
https://github.com/pq-crystals/dilithium.git

Experiment details

- Raspberry Pi and computer:
  - Optimizations + reference implementation
  - Running time and RAM usage
  - All security levels (i.e. 2, 3 and 5)
  - 100 times
- Arduino Due:
  - Only optimization
  - Running time and RAM usage
  - Security levels 2 and 3 (level 5 is too resource-intensive)
  - 1000 times

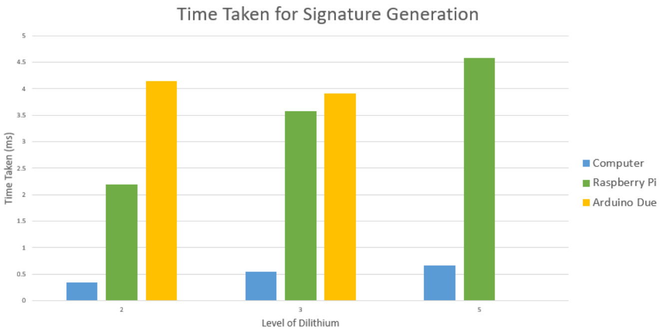
Plan

Running time: key generation



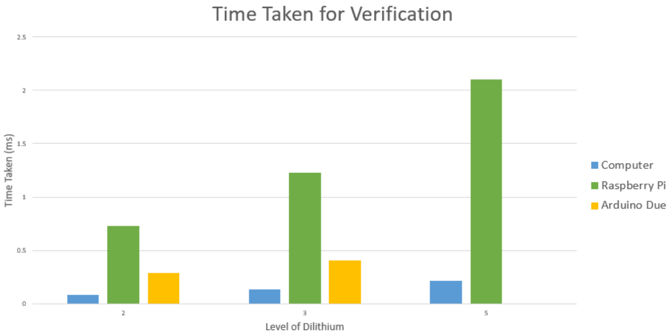
- Higher security level → longer running times
  - In particular for the Raspberry Pi

Running time: signature generation



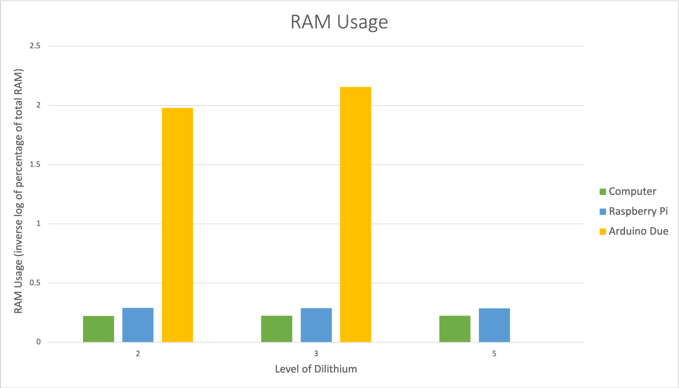
- Signing takes more time with the Arduino Due than other device/machine
- But still under 5 ms

Running time: signature verification



- Timing results similar to key generation

RAM usage



- Limited RAM on Arduino Due (96 KB)
- Optimization stack size at about 1/3 of the RAM

Summary

- Running times and RAM usages increase with security levels and depend on type of device/machine *as expected*
- Optimizations offer better results than reference implementation *as expected*
- CRYSTALS-Dilithium can be run on Arduino Due but not great yet?
  - Since GPU et CPU double every 3-4 years, focusing on the Raspberry Pi instead?
  - Expecting better optimizations?

Thank you!
Questions?

Summarizing CPU and GPU design trends with product data, Y. Sun *et al.*, arXiv:1911.11313, 2019