PQ-DPoL
: An Efficient Post-Quantum Blockchain Consensus Algorithm

Won-Woong Kim, Yea-Jun Kang, Hyun-Ji Kim, Kyung-Bae Jang, Hwa-Jeong Seo
Hansung University

한성대학교 HANSUNG UNIVERSITY

CryptoCraft LAB

Contents

Introduction

- The **ECC-based signature scheme** is commonly adopted in blockchains for **transaction signing and verification.** (e.g. secp256-k1 in Bitcoin/Ethereum)

- However, the following two factors prompt us to consider **replacing ECC with PQC**

  1. Shor’s algorithm
  2. The emergence of quantum computer

- In this paper, We proposed **“PQ-DPoL(Post-Quantum Delegated Poof of Luck)”**

Introduction || Contribution

- Our main Contributions

1. **Post-quantum blockchain** with PQC scheme.
   - **CRYSTALS-Dilithium**

2. New consensus algorithm for an efficient blockchain.
   - Delegate + **Proof-of-Luck**

3. Diverse Configurations and Benchmarks.
   - The number of nodes
   - **Security parameter** of Dilithium

Background || Consensus Algorithm

- **The consensus algorithm** is used by nodes on the blockchain **to ensure data integrity** through specific procedures and **to make the same decision.**

- Representative consensus algorithms
  - Proof-of-Work (PoW)
  - Proof-of-Stake (PoS)
  - Delegated-Proof-of-Stake (DPoS)
  - Proof-of-Elapsed-Time (PoET)
  - Proof-of-Luck (PoL)

Background || TEE (Trusted Execution Environment)

- **The security area of the main processor.**

- **There are various unique techniques for security.**
  - Trusted Time
  - Monotonic Counter
  - Attestation
  - Random Number Generator

Background || CRYSTALS-Dilithium

- **Post-quantum cryptography** selected by NIST as an algorithm to be standardized.

- **Lattice-based cipher** based on the **Shortest Vector Problem (SVP).**

Scheme	NIST level	Public key	Private Key	Signature
Dilithium-2	**2**	**1,312**	**2,528**	**2,420**
Dilithium-3	**3**	**1,952**	**4,000**	**3,293**
Dilithium-5	**5**	**2,592**	**4,864**	**4,595**

Background || Evaluation Metric

- **TPS**
- The number of **transactions that can be confirmed in one second.**
- Bitcoin - 7TPS / Ethereum - 20TPS / EOS - 3000TPS

- **Latency**
- The time it takes from the time a **transaction appears on the network until it is verified.**
- Bitcoin - 10minutes / Ethereum - 0.22 minutes / Ripple - 4seconds

- **In addition, there are other metrics**
  : verification time, decetralized level, and power consumpotion

Before Proposed Method

- **There are 5 considerations**
  - **Key Size**
  - **Signature Size**
  - Execution Speed
  - Computational Complexity
  - Power Consumption

Before Proposed Method

- **Reasons for applying Dilithium**
  - The Dilithium scheme **faster than PQC signature schemes**
  - The devices targeted for our blockchain are **CPUs that support TEE**
  - This indicates that our blockchain is **capable of accommodating PQC with large key sizes.**

- **However, several problems exist…**
  - We will present how **to increase the lowered TPS** using the proposed consensus algorithm, DPoL.

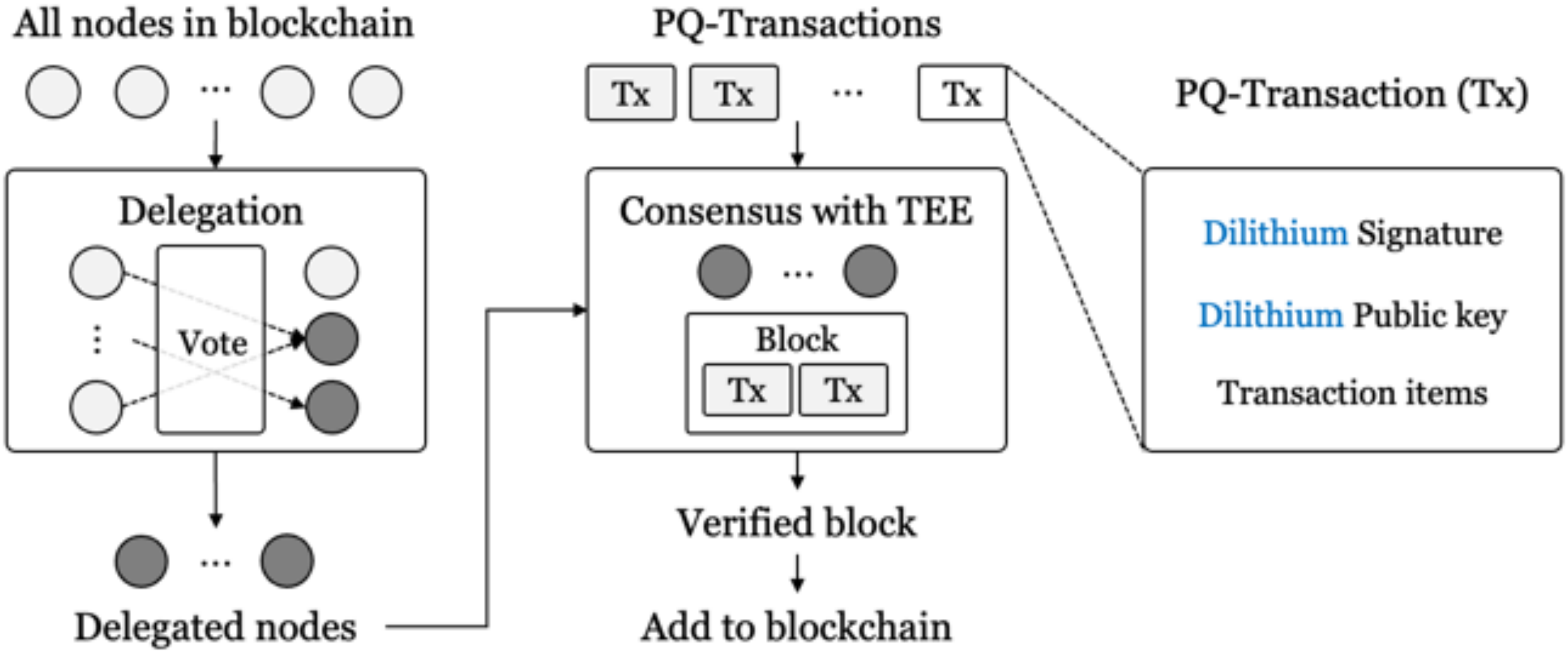
Before Proposed Method

- **How to use TEE in our method**
  - Trusted Time
    - **Give the delay** of schedule as much as round time
  - Monotonic Counter
    - **Prevent concurrent invocation**
  - Remote Attestation
    - **Generate proof** to verify with each other
  - Random Number Generation
    - **Use as the luck value** in the algorithm

Proposed Method || System Overview

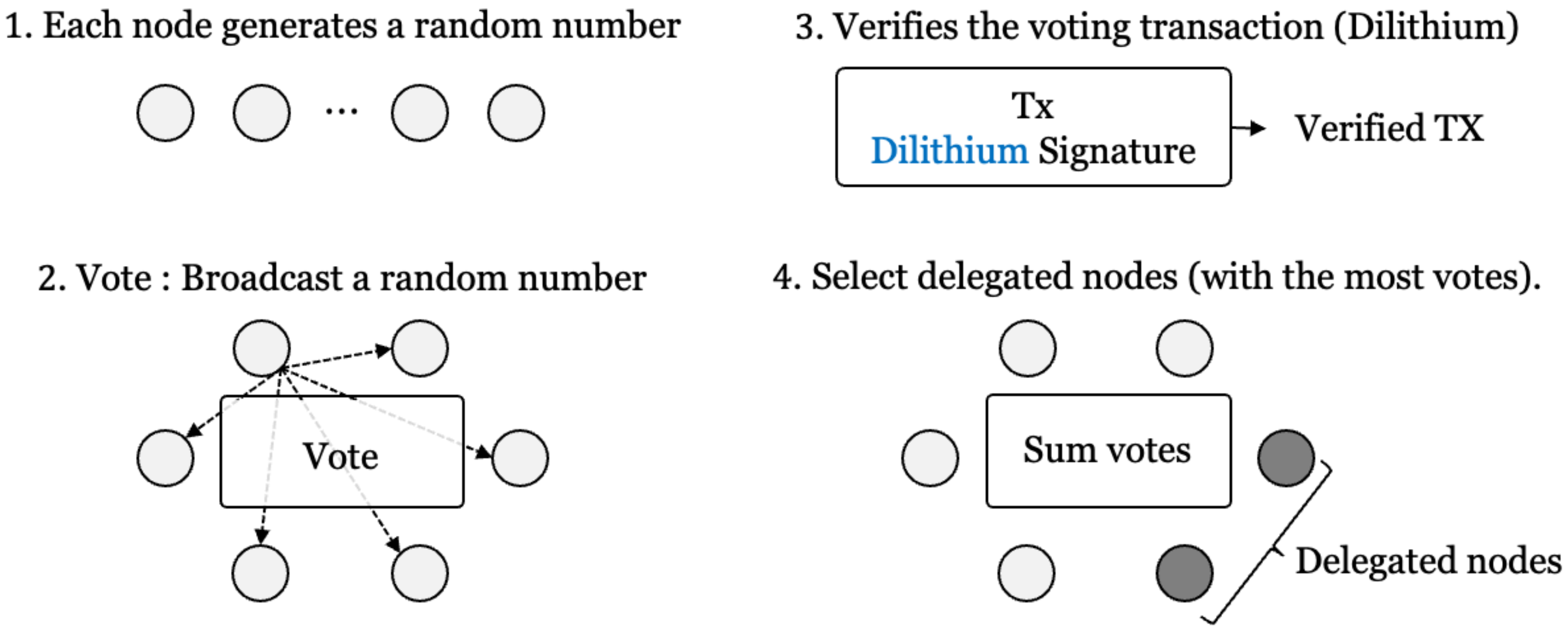
- **PQ-DPoL** is divided **Delegation** phase / **Consensus** phase

- **System Overview**



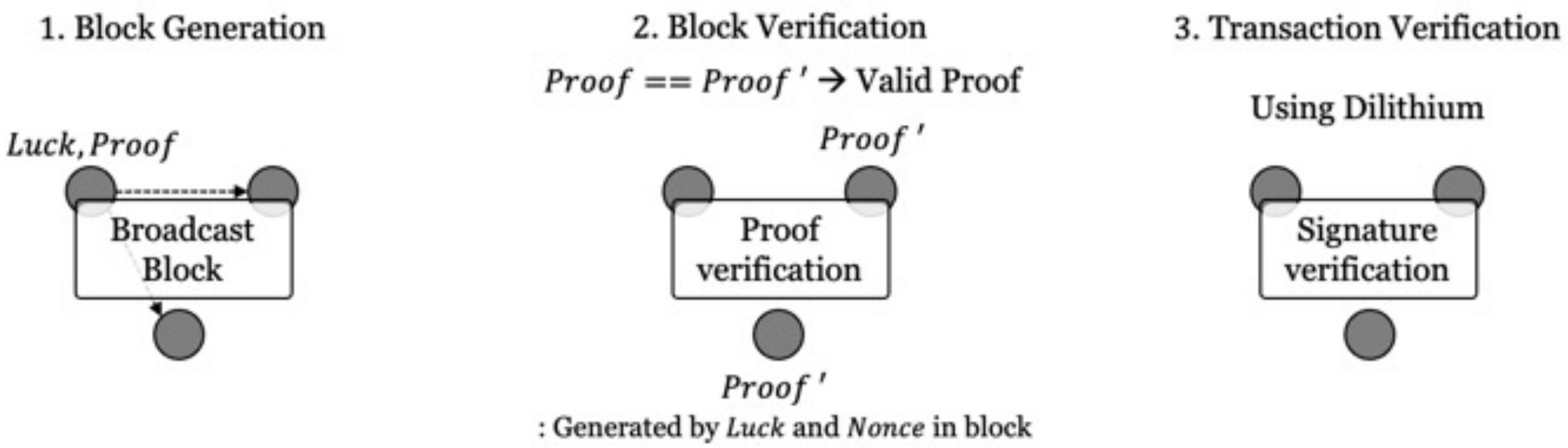
Proposed Method || Delegation

- **Delegation phase**
  1. Random Number Generation
  2. Vote
  3. Verification
  4. Delegation



Proposed Method || Consensus

- **Consensus Phase**
  1. Block Generation
  2. Block Verification
  3. Transaction Verification



Benchmark || Envrionment

- **Target Processor**: Intel i5-8295U CPU with 16GB RAM

- **Framework**: Visual Studio Code

- **Language**: C++

- To build an environment close to a real blockchain network, we use **NS-3**, an open-source network simulator.

Benchmark || Performance

- **Execution speed comparison of ECDSA (P-256) and Dilithium**

	Key Gen	Sig Gen	Sig Verify
ECDSA (P-256)	**0.000400**	**0.000670**	**0.000350**
Dilithium-2	**0.000027**	**0.000100**	**0.000026**
Dilithium-3	**0.000047**	**0.000160**	**0.000043**
Dilithium-5	**0.000073**	**0.000200**	**0.000065**

Benchmark || Performance

- **TPS of PQ-DPoL**

n	1	2	3	4	5	6	7
ECDSA	713.5797	122.5398	29.8184	6.9522	0.9468	0.2341	0.1030
Dilithium-2	88.8721	21.0373	5.1244	1.2515	0.3302	0.0776	0.0171
Dilithium-3	48.4901	11.7897	2.8879	0.7128	0.1815	0.0446	0.0104
Dilithium-5	30.8097	7.4470	1.8600	0.4526	0.1203	0.0291	0.0066

Benchmark || Performance

- **Latency of PQ-DPoL**

n	1	2	3	4	5	6	7
ECDSA	0.1401	0.8160	3.3536	14.3838	105.6131	427.1180	970.0098
Dilithium-2	0.0675	0.2852	1.1708	4.7940	18.1667	77.2573	350.8609
Dilithium-3	0.0824	0.3392	1.3850	5.6116	22.0333	89.5175	383.3946
Dilithium-5	0.0973	0.4028	1.1628	6.6277	24.9267	103.0436	454.1221

Conclusion

- The TPS of DPoL applied with Dillithium is **lower than that of ECDSA**

- To improve the degraded performance, **we propose DPoL**, a new consensus algorithm that **combines PoL and delegation approch.**

- Certainly, when employing Dillithium as the signature scheme in DPoL, although its performance may be lower than that of ECDSA, it still **provides reasonable performance.**

- Finding various methods to improve TPS can be considered for the future works.

Q & A