Communication Protocol Impact on Energy Efficiency of Blockchain Application

Analysis of Energy Efficiency of Android Blockchain Application Using UDP or QUIC

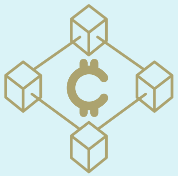
TU Delft
EEMCS Faculty (EWI)
Author: Tomasz Puczel, t.puczel@student.tudelft.nl
Supervisors: Johan Pouwelse, Bulat Nasrulin

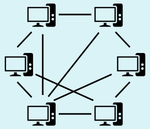
TU Delft CSE3000 Research Project Q4 2024/2025

1. Background

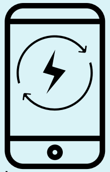
Blockchain is a technology capable of achieving decentralized data storage without any trust to a central authority. Most prominently adopted in cryptocurrencies - e.g., Bitcoin, Ethereum [1].

- Data is stored in **blocks**
- Blocks are **chained** together by a **hash** of the previous block in the chain
- The payload and block are **cryptographically** signed





Participants form a **peer-to-peer network**, where nodes are identified by public keys and communicate with each other.



At the same time, blockchain faces limitations [2] related to its scalability, or resource management on an individual node. Adoption of blockchain on resource-constrained devices, such as **smartphones**, requires the design to be **energy efficient**.

Smartphones could benefit from blockchain adoption but are used in our daily lives, and should not run out of battery too quickly or overheat.

Blockchain based systems can differ in a few design or implementation aspects, e.g., fraud prevention mechanism or the communication protocol used for the messaging between the nodes. One developed architecture is **TrustChain** [3].

In this work we analyze the energy efficiency of a simplified blockchain application for Android smartphones with focus on comparison of: **UDP** and **QUIC** communication protocols.

2. Research Questions

- **RQ1**: How energy efficient is a TrustChain-inspired blockchain implementation on Android mobile devices?
- **RQ2**: How do UDP and QUIC protocols affect the energy efficiency of such a blockchain application?
- **RQ3**: What factors contribute to the energy efficiency of such implementations the most?

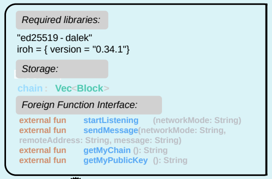
3. Methodology

High-level implementation

Inspired by the TrustChain design:

- blocks include signatures and previous hashes of both the sender and receiver, intertwining both chains
- communication happens in sender-receiver pairs
- nodes only store the chain with the blocks they are involved in

Kotlin UI component



Rust logic component

Communication protocols

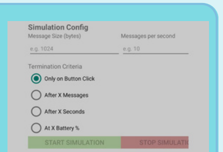
The application supports:

- **UDP**: via UdpSocket struct in Rust
- **QUIC**: abstracted away via Iroh[1] crate from Rust

Simulation

Automated simulations with:

- Target transmission rate
- Payload size
- Termination criteria



Measurements

Data collected:

- Battery voltage, temperature, current draw and capacity level queried using BatteryManager[2]
- Battery summary from `dumpsys batterystats`[3] providing breakdown per process
- System traces[4] recorded using native tracing utility

Recognized but not used:

- PowerMetric (only supported on newer models),
- Hardware measurement tools (unavailable),
- BatteryHistorian (no longer supported)

4. Results

1. Long-term battery metrics (RQ1 & RQ2)

Eight long-duration simulation runs were executed on a Samsung A50 smartphone.

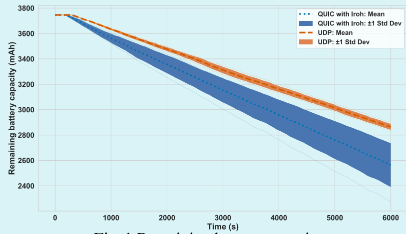


Fig. 1 Remaining battery capacity

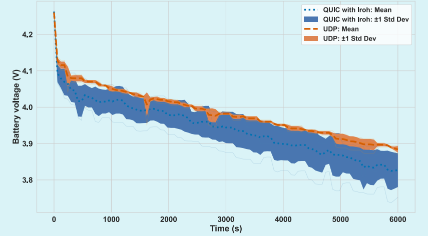


Fig. 2 Battery output voltage

The metrics collected from BatteryManager at 60-second intervals:

Metric	UDP	QUIC with Iroh
Voltage,V	3.89	3.82
Capacity,mAh	2865	2563
Avg. current,mA	-450 (±20)	-450 to -725 (±100)
Temp. increase,°C	+6.7	+9.3

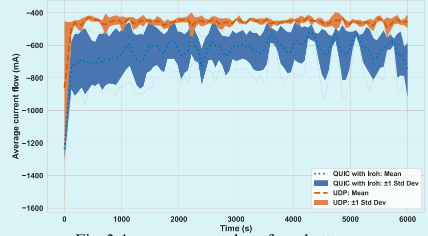


Fig. 3 Average current draw from battery

2. Components analysis (RQ2 & RQ3) - towards understanding contributing factors

`dumpsys batterystats` breakdown per system components

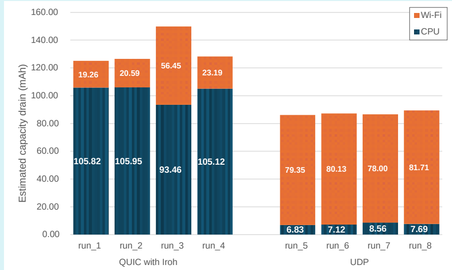


Fig. 4 Estimated capacity drain per CPU and Wi-Fi

Average number of packets per block

UDP	QUIC, new connection per block	QUIC with connection reuse
1	~14	~4

Observations

- Higher proportion of energy drain attributed to the CPU for QUIC-based implementation (usually 82-85%). Partially explained by spawning more threads in the implementation.
- In contrast, higher proportion (~90%) attributed to the Wi-Fi for UDP-based implementation, <u>despite</u> much fewer packets sent.

Further work is required to understand underlying causes.

5. Future Work

- Expand blockchain implementation, integrating consensus mechanism or trustworthiness calculations, full chain verification, and malicious parties tolerance - to evaluate their impact on energy consumption.
- Investigate direct causes of the energy consumption, narrowing it down to concrete components of the application or the phone's activity and isolating the network communication aspect further.
- Explore other alternatives for the communication protocol (e.g., TFTP, TCP, uTP, DCCP) and for the other aspects of the whole blockchain architecture.
- Evaluate energy efficiency on various mobile platforms (iOS), smartphone models, and other resource constrained devices.
- Analyze more extensively the impact of concrete properties of the examined protocols and the trade-offs related to employing different ones as the underlying communication implementations.

References

[1] - https://docs.rs/iroh/latest/iroh/, https://www.iroh.computer/
[2] - https://developer.android.com/reference/android/os/BatteryManager
[3] - https://developer.android.com/tools/dumpsys
[4] - https://developer.android.com/topic/performance/tracing

[1] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Cryptography Mailing list at https://metzdowd.com, 03 2009.
[2] Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed Kosba, Andrew Miller, Pra teek Saxena, Elaine Shi, Emin Gün Sirer, et al. On scaling decentralized blockchains: (a position paper). In International conference on financial cryptography and data security, pages 106–125. Springer, 2016.
[3] Pim Otte, Martijn de Vos, and Johan Pouwelse. Trustchain: A sybil-resistant scalable blockchain. Future Generation Computer Systems, 107:770–780, 2020.