Resilient Offline Payment With Wireless Mesh Network

Quang Huy Do[†‡], Sara Tucci-Piergiovanni[†], Sami Souihi[†§]
[†]LISSI-TincNET Research Team, University of Paris-Est Creteil, France
[‡] Ejara Lab, France
[§] Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
Email: huy.quangdo@ejara.africa, sara.tucci@cea.fr, sami.souihi@u-pec.fr

***Abstract*—Mobile payment systems have become an essential part of modern digital services, supporting applications such as money transfer, mobile banking, and digital wallets. Despite their widespread adoption, these systems rely heavily on stable Internet connectivity, which remains unavailable or unreliable in many parts of the world. To address these challenges, we propose a decentralized offline payment framework that integrates wireless mesh networking with a blockchain-based settlement layer, enabling secure, real-time transaction settlement without Internet access. Blockchain technology provides a transparent and tamper-resistant ledger, reduces transaction costs and long process of validity by eliminating intermediaries. The proposed system is implemented and evaluated using Mininet-WiFi, demonstrating its ability to achieve transaction latencies below 1.3 seconds and maintain operational stability under 33% node failure in networks exceeding 250 nodes. These results highlight the feasibility of mesh-based offline payment systems as a robust alternative for connectivity-constrained environments.**

***Index Terms*—Wireless Mesh Networks, Offline Payment, Blockchain**

I. Introduction

According to [1], the global mobile payment market projected to reach 7.58 trillion USD by 2027, at a CAGR of 29.0% during the forecast period. This remarkable expansion is particularly significant in developing countries, where mobile wallets have become the dominant payment method. The study [2], found that transactions via mobile wallets and phones were the equivalent of 87% of GDP in Kenya and 82% in Ghana.

However, internet infrastructure is one the main reasons that challenges mobile payment for wide adoption. The internet infrastructure deployment in developing countries—particularly across Africa—remains severely constrained. Less than a third(27%) of adults in Africa can have access mobile Internet services [3]. In bustling open-air markets, traders and shoppers often have internet capable phones in their pockets yet still pay with cash because the network is down or too slow to authorize transactions in real time. Compounding these challenges, internet shutdowns happened 21 times in 2024 in African countries, the highest number in a single year [4]. During shutdowns, mobile money agents and end users lose access to their wallets, resulting in the suspension of peer-to-peer transfers and merchant transactions.

Traditional mobile money systems further worsens these limitations, as they depend on centralized banking infrastructure for transaction settlement. These systems become inaccessible during network disruptions, with studies showing that internet shutdowns could have an estimated GDP impact of approximately $3 million per day of disruption, amounting to 0.4% of daily GDP [5]. Additionally, the complex security validation processes require extensive Know Your Customer (KYC) procedures that systematically exclude populations, particularly affecting the 850 million adults globally who do not have a form of ID, while mobile money providers typically charge significant fees ranging from 4.64% to 4.80% for cross border remittances [6]. In contrast, a blockchain based settlement layer enables peer-to-peer value transfer without relying on any single institution. It remains available even during network disruptions, reduces transaction costs by removing intermediaries, and provides a transparent, tamper resistant ledger that can serve users outside the conventional financial system [7].

Thus, in this paper, we propose a method that leverages wireless mesh networking and blockchain technology, enabling devices to form decentralized, peer-to-peer communication networks. In this architecture, mobile devices communicate directly with each other, relaying payment information across multiple hops until reaching the destination node. To address the challenges of trust and transaction validation in such a decentralized environment, we introduce a quorum mechanism to validate the payment. In this system, authority nodes (such as mobile money agents, merchants, or trusted community members) validate the transfer orders from users then send back signed signatures. When a quorum of signatures is reached, users can be sure that the payment is finalized and will be eventually reconciled on chain. This quorum protocol provides robust consensus and reducing the risk of fraud or double spending.

Contributions. We make the following contributions:

- Propose a novel, infrastructure independent mobile payment architecture that uses wireless mesh networking to enable resilient, peer-to-peer transactions in environments with unreliable or limited internet connectivity.
- Introduce a decentralized weighted voting to ensure secure, transaction validation within decentralized networks, mitigating risks of fraud and double spending.
- Provide an evaluation of the proposed system under realistic connectivity constraints and user mobility.

Outline. The remainder of this paper is structured as follows. Section II provides background and related work. Section III shows the overview of the whole proposed system. Section IV details the proposed offline payment methodology. Section V presents the experimental results and analysis. Finally, Section VI concludes the paper with a discussion of key findings and future research directions.

II. Related Work

LIO-Pay [8] currently represents the state of the art in offline payment research. The protocol tackles the offline payment trilemma by combining blockchain anchoring with cryptographically secure tokens executed inside trusted execution environments (TEEs). While this hardware design prevents double spending and facilitates loss recovery without continuous connectivity, it imposes three practical barriers to large scale deployment in low resource contexts such as rural Africa: (i) it relies on specialized secure elements or TEEs that are absent from most entry level smartphones, (ii) it introduces a centralized trust dependency on hardware vendors and authorities, and (iii) the cost of having millions of devices with certified TEEs is economically limited.

In fact, there are already several solutions that reuses commodity WiFi and Bluetooth radios to enable a peer-to-peer mesh networking. Among existing mesh based solutions, LNMesh [9] demonstrates that Bitcoin Lightning Network channels can be relayed over WiFi/BLE mesh network, achieving sub-second confirmations and onchain interoperability. While the work looks promising, it requires conditions that may limit its practical deployment in real world scenarios. For instance, LNMesh requires an active internet connection to establish and settle Lightning channels before and after a payment session. This requirement cannot be promised in environments prone to sudden or prolonged internet outages, where users may not have the opportunity to open a channel in advance and close them after transfer is done. In contrast, the protocol proposed in our work requires only a one time token deposit prior to offline usage, eliminates the need for an active internet connection during the transaction process.

It is also worth mentioning FastPay [10], which serves as an inspiration for our system design. FastPay employs a committee of authorities that pre-sign account certificates, enabling clients to gather a quorum of signatures within a single network round. This architecture achieves remarkable performance, delivering sub-100 ms settlement latency and throughput exceeding 80,000 transactions per second (TPS), while preserving strong Byzantine fault tolerance guarantees. We extend FastPay by replacing its static weighted quorum with a dynamic, decentralized weighted voting scheme inspried by Cabinet [11]. Specifically, authority nodes are selected through a hybrid admission process that combines stake commitment with performance-based evaluation: candidate nodes must lock a minimum stake to get admitted into the authority set. Once admitted, each authority's voting power is updated according to an audit score that reflects both performance (recent transaction throughput, uptime) and trust metrics (peer reputation, history of misconduct). Moreover, we remove infrastructure dependencies, we embed this enhanced FastPay atop a mesh network formed directly among mobile nodes. By employing decentralized weighted voting with a mesh overlay, our system sustains secure payments across multi hop paths during prolonged internet or cellular outages, overcoming the deployment barriers faced by prior offline payment solutions.

III. System Overview

A. Participants

Our protocol involves three types of participants: (i) users, (ii) Authorities and (iii) Smart Contract.

Users are ordinary nodes that hold cryptographic key pairs and initiate payments. Users deposit funds into the smart contract and generate off-chain transfer requests. Users hold following information

- Key pair (sk_u, pk_u).
- Committee configuration $\mathcal{C}$ (authority public keys, epoch).
- Local sequence counter $seq_u \in \mathbb{N}$ for off-chain orders.
- Last certified transfers Certs_u; Pending orders $\mathsf{Pending}_u$.

Authorities is a fixed committee of validator nodes (typically $3f + 1$ in total, tolerating up to f Byzantine faults) that maintains the off-chain ledger. To become authority, you must set a pre-defined fixed amount of token on-chain via a smart contract. Each authority has a key pair and holds the committee configuration (membership and weights). Authorities independently validate user orders and add their signatures. A quorum (e.g., $2f+1$ signatures) suffices to certify a transaction. Honest authorities strictly follow the protocol; faulty ones may behave arbitrarily. The smart contract on the primary ledger is aware of the authority set and enforces payments only when enough committee signatures are presented. The state of an authority consists of the following information:

- Key pair (sk_a, pk_a); committee configuration $\mathcal{C}$; quorum threshold q (e.g., $2f+1$).
- Off-chain account table $\mathsf{Ledger}[u] = (\mathsf{bal}_u, \mathsf{next}_u)$ with current balance and next valid sequence of all User.
- Pending orders $\mathsf{Pending}[u, s]$: validated user orders (sender u, sequence s) and local signature of all Users.
- Certified transfer $\mathsf{Certs}[u, s]$: quorum-certified transfers of all Users.
- Redeem log $\mathsf{Redeemed}[u]$: set or prefix of sequences already settled on the primary ledger.
- Weighted Voting Power

Smart Contract, the smart contract serves as the immutable settlement point. Users deposit (fund) and later redeem (settle) their balances via this contract. The contract verifies authority-signed certificates during withdrawals and ensures every off-chain balance is backed 1:1 by locked funds on-chain. The storage of the smart contract contains following information:

- Authority set $\mathcal{C}$ (identities, public keys, epoch, locked funds) and quorum threshold q.

- Locked funds mapping $\mathsf{Locked}[u] \in \mathbb{N}$ for user deposits.
- Redeem log $\mathsf{Redeemed}[u]$ (e.g., highest redeemed sequence or a membership set) to prevent double-withdrawals.

B. Protocol Messages

The protocol distinguishes between on-chain transactions and off-chain orders exchanged over the mesh. Key message types include:

1) **Deposit Transaction:** An user funds the system by sending an on-chain deposit transaction T to the smart contract, locking a specified amount in a primary account. The contract records this and emits a signed deposit receipt binding the amount to the user's public key and epoch.
2) **Funding Order:** The user forwards the deposit receipt to nearby authorities as proof of collateral. Each authority verifies the receipt and, once a quorum attests, credits the user's balance in the local ledger.
3) **Transfer Order:** When a sender wishes to pay a recipient, they create a transfer order O with:
   - $sender(O)$: Sender's account
   - $recipient(O)$: Recipient's account
   - $amount(O)$: Transfer amount
   - $sequence(O)$: Monotonically increasing sequence number
   - Optional memo or metadata
   - σ_{sender}: Sender's signature over the fields

   The order O is broadcast to all authorities, which validate balances and sequence numbers.
4) **Authority Response:** Each honest authority α that accepts O adds its signature σ_α and replies. These partial signatures form the basis of a certificate.
5) **Transfer Certificate:** Once the sender collects at least $2f + 1$ signatures on O, she assembles them into a transfer certificate C. This certificate is proof that the payment is valid. The sender delivers C to the recipient.
6) **Confirmation Broadcast:** Either sender or recipient broadcasts C to all authorities. Upon receiving C, each authority debits the sender and credits the recipient. At this point, the transfer is final.
7) **Settlement Transaction:** Periodically, or when a user withdraws, a settlement transaction is sent on-chain including the relevant certificates. The smart contract verifies these certificates and updates the on-chain balances accordingly.

C. Wireless Mesh Network

The off-chain protocol operates over a **wireless mesh network** (WMN). Each node (user or authority) connects to peers via wireless links (e.g., Wi-Fi, Bluetooth), forming a multi-hop mesh.

- **Decentralized Connectivity:** Each node relays packets for others; no central server is required. Nodes only need to transmit as far as the next hop. The mesh can be full (all nodes interconnected) or partial (local neighbors only).
- **Multi-Hop Routing:** Messages propagate hop-by-hop until they reach all authorities. Routing protocols OLSR [12] ensures finding the most optimal path to destination. Multiple redundant paths make the mesh resilient to link failures.
- **Asynchronous Delivery:** Wireless links may introduce arbitrary delays, reordering, or temporary losses, but eventual delivery is assumed for honest nodes. As long as $2f+1$ authorities eventually receive and sign an order, progress is guaranteed.
- **Dynamic Participation:** Nodes may move or disconnect. The protocol tolerates partitions as long as a quorum remains connected with the sender and receiver. Disconnected users can re-sync later.
- **Limited Bandwidth:** Wireless meshes often have low throughput. The protocol minimizes overhead: a payment requires only one broadcast of an order and one broadcast of a certificate. Certificates include only essential fields and signatures.

IV. Methodology

Our protocol uses a **side-chain** approach: users first lock funds on a main blockchain (e.g., via a smart contract on Ethereum [13]) and then transact in a local mesh network using pre-funded balances. This ensures every off-chain token is fully collateralized on the primary chain. When connectivity returns, the mesh submits batched updates to the main chain so balances remain consistent.

A. Depositing Funds

As illustrated in Figure 1, an user wants to use the protocol and become authorities can initiate a deposit by invoking the on chain smart contract on the main blockchain. This transaction is called a funding transaction, and includes the recipient address for the funds and the amount of value to transfer. The contract locks the specified amount and returns a signed deposit receipt that binds the value to the user's public key and the current mesh-committee epoch. The user forwards this receipt to a nearby authority, which verifies the on chain signature and records the corresponding balance in the local mesh ledger. Once at least a quorum of Authorities attest to the receipt, the funds become immediately spendable inside the offline mesh, guaranteeing that every token circulating off chain is fully collateralized on the main chain ledger.

B. Transferring Funds

Figure 2 depicts an offline payment between two Users. The Sender creates a transfer order O, sign and broadcasts it to the neighbor nodes. The Sender also stores the transfer order O to the $Pending_{sender}$ list. If Authority receives the order, it independently checks (i) that the signature is valid, (ii) that no previous transfer is pending, (iii) that the amount is positive, (iv) that the sequence number matches the expected next one, and (iv) the balance is sufficient. And if valid, authorities adds

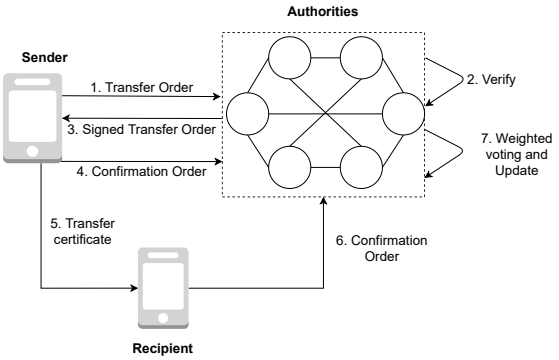

Fig. 1: Transfer of funds between Users

its signature to the order, mark it as pending order and send back a signed transfer order. The algorithm to handle transfer order is presented in Algorithm 1.

After collecting enough signed signatures from a quorum, the Sender can form a transfer certificate. With the certificate, the Sender can forward it to the Receiver as proof of payment. To conclude the order, the Sender or Recipient broadcast the transfer certificate (called confirmation order) to all authorities. Upon reception of a confirmation order for the current sequence number, each authority (i) checks that a quorum of signatures was reached, (ii) decreases the balance of the sender, (iii) increments the sequence number to ensure 'deliver once' semantics, and (iv) sets the pending order to None. Each authority also (v) adds the certificate to the list confirmed, (vi) increases the balance of the recipient account asynchronously (i.e. without sequencing this write in relation to any specific payments from this account across authorities) and finally (vii) updates its current voting power. The authority algorithm to handle confirmation orders is presented in Algorithm 2. If there is still pending orders inside $Pending_{sender}$, the sender periodically broadcasts the orders to all other neighbor nodes.

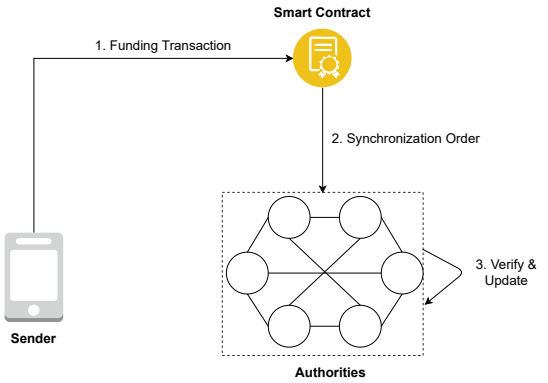

Fig. 2: Transfer of funds between Users

C. Withdraw Funds

When Internet connectivity resumes, each authority participates in a batched reconciliation: authorities aggregate their off-chain ledger view and submit a quorum-certified update

Algorithm 1 Authority: Handle Transfer Order

function HANDLE_TRANSFER_ORDER(α, O)
  ensure O.has_valid_signature()
  $account \leftarrow$ Ledger[O.sender]
  if $account$ is None **then**
    return REJECT("account-not-found")
  end if
  if Pending[O.sender, $next_u$] exists **then**
    if Pending.transfer $== O$ **then**
      return ACCEPT
    else
      return REJECT("conflict-order")
    end if
  end if
  ensure $next_u == O$.sequence
  ensure $account$.balance $\geq O$.amount
  ensure O.amount > 0
  $\sigma_\alpha \leftarrow$ Sign(sk_α, O)
  Pending[O.sender, O.sequence] $\leftarrow (O, \sigma_\alpha)$
  return ACCEPT(O, σ_α)
end function

Algorithm 2 Authority: Handle Confirmation Order

function HANDLE_CONFIRMATION_ORDER(α, C)
  ensure C.is_valid($\mathcal{C}, q$)
  $O \leftarrow$ value(C)
  $sender\_account \leftarrow$ Ledger[O.sender]
  if $sender\_account$ is None **then**
    $sender\_account \leftarrow (bal = 0, next = 0)$
  end if
  if $sender\_account.next > O$.sequence **then**
    return ACCEPT("already-confirmed")
  end if
  ensure $sender\_account.next == O$.sequence
  ensure $sender\_account.bal \geq O$.amount
  $sender\_account.bal \leftarrow sender\_account.bal - O$.amount
  $sender\_account.next \leftarrow sender\_account.next + 1$
  Pending[O.sender, O.sequence] $\leftarrow None$
  Certs[O.sender, O.sequence] $\leftarrow C$
  $recipient\_account \leftarrow$ Ledger[O.recipient]
  if $recipient\_account$ is None **then**
    $recipient\_account \leftarrow (bal = 0, next = 0)$
  end if
  $recipient\_account.bal \leftarrow recipient\_account.bal + O$.amount
  return ACCEPT("confirmed")
end function

to the smart contract. The contract verifies the committee signatures and updates its state, ensuring that every off-chain balance remains 1:1 backed by locked funds on the primary chain.

As illustrated in Figure 3, an user withdraws by issuing a direct on-chain request to the contract, specifying a destination and amount. The contract checks admissibility against its reconciled state: $0 \leq \mathit{amount} \leq \mathsf{Locked}[u] - \mathsf{Redeemed}[u]$, updates the redeem log to prevent double-withdrawals, and transfers the requested value to the user's address.

Post-withdrawal, authorities continuously fetch the smart contract to update per-account state and detect divergence. In detail, each authority periodically fetches the on-chain balance and the authoritative per-account sequence counter $seq_{\text{chain}}[u]$ and cross-checks it against its local next-sequence $seq_{\text{off}}[u]$. Authorities enforce $bal_{\text{off}}[u] = bal_{\text{chain}}[u]$ and that the locally available balance matches the on-chain collateral.

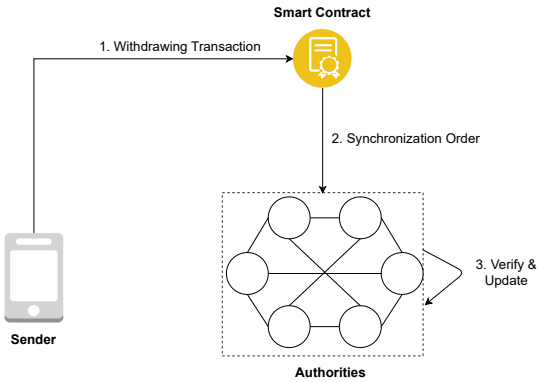

Fig. 3: Transfer of funds from Protocol to Blockchain

V. Evaluation

A. Experimental Setup

We perform the emulations with **Mininet-WiFi** [14] over an area of 1000 × 1000 m^2. All the emulations are averaged over 10 runs with each emulation running for 3000 s. Emulations are performed with 5 User and Authorities from 5 to 45. Each User sends one transfer order at 10 s interval time. Propagation loss model `logDistance` is used to limit the transmission ranges of the nodes. The transmission range of the nodes is set as 5m for evaluation. The mobility model used is a realistic version of Random Direction [15] to mimic the movement of mobile user in real world. A summary of all emulation parameters is shown in Table I. The implementation, bench marking scripts, and measurements data to enable reproducible results are here. [1]

B. Results And Analysis

1) Latency and Number of Authorities: Figure 4 shows a clear U-shaped trend, highlighting the tradeoff between network coverage and coordination overhead in our payment system. When there are only a few authorities (around 5–10 nodes), latency is very high—especially in the 5-meter transmission range scenario. This happens because the network is too sparse, forcing transactions to travel through many hops to reach their destination. In such cases, some areas may even become temporarily disconnected, which is consistent with the known minimum node density requirements for maintaining a connected ad-hoc network.

As the number of authorities increases, latency decreases and reaches its lowest point around 30 authorities for all transmission ranges. This point represents a balance: there

[1]https://github.com/HuyPHD2024-2027/smart-pay

Parameter	Value
Area	1000 × 1000 m
Number of runs	100
Emulation time	3000 s
Number of user nodes	200
Number of authorities	50
Transmission range	5 m
Transfer order generating time	10 s
Propagation loss model	logDistance
Mobility model	Random Direction
Node speed	0−20 m/s

TABLE I: Emulation Parameters

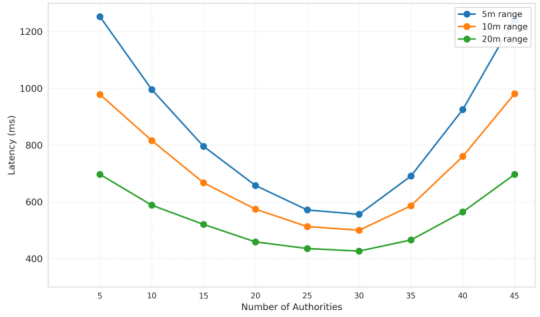

Fig. 4: Variation of the latency of transfer orders with the number of authorities, for various locations of the client.

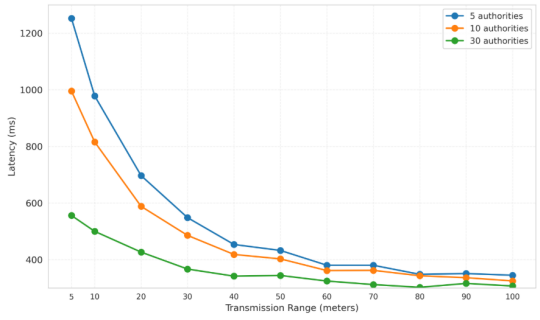

Fig. 5: Variation of the latency of transfer orders with the tranmission range of authorities, for various locations of the client.

are enough nodes to provide good coverage and short paths, but not so many that coordination between them becomes too costly. Beyond this optimal point, latency starts to rise again because the communication overhead grows quickly. In Byzantine quorum protocols, the number of inter-authority messages increases roughly quadratically with the number of nodes, and more nodes also mean more competition for the wireless channel.

The transmission range also plays a key role in shaping this relationship. Larger ranges, such as 20 meters, flatten the U-curve significantly. This means that with longer transmission ranges, even a smaller number of authorities can achieve near-optimal latency. When nodes can reach more peers directly, the network becomes more robust and less dependent on having many authorities. Practically, this suggests that improving the transmission capability of authorities nodes could reduce drastically latency of the transfer order.

2) Latency and Number of Authorities: Figure 5 demonstrates a clear inverse relationship, matching what we'd expect from multi-hop routing theory. As the radio range increases, latency drops because packets need fewer hops to reach their destination authorities. The biggest improvements happen between 5 and 30 meters, where each small increase in range noticeably shortens the average path through the mesh. In this region, the network transitions from being fragmented—where messages might need to pass through 15 or more hops—to a more connected state where many authorities can communicate directly. This shift eliminates much of the delay caused by the repeated store-and-forward process typical of multi-hop routing. Beyond about 50 meters, the latency curve flattens out. At that point, most authorities can already reach each other directly, so increasing the range further offers only minor benefits for nodes that are far apart.

Comparing different numbers of authorities gives additional insight into how network structure interacts with consensus overhead. When the range is limited (around 5–10 meters), configurations with only five authorities show much higher latency than denser setups. This happens because sparse deployments force transactions to travel across long multi-hop routes before they can reach enough authorities to form a quorum. However, once the transmission range exceeds 50 meters, these differences almost disappear: the 5-, 10-, and 30-authority setups perform within about 15% of each other. This convergence occurs because, at larger ranges, even a small number of authorities can directly communicate with one another. The network effectively shifts from a multi-hop mesh into a single-hop broadcast environment.

This observation is encouraging for real-world deployments. In well-connected settings—such as open outdoor areas or systems equipped with directional antennas—a relatively small number of authorities (10–15) can perform nearly as well as much denser networks. This means lower deployment costs, reduced energy use, and less communication overhead, all while maintaining the Byzantine fault tolerance required for secure decentralized payments.

VI. Conclusion

Our early results show that an offline payment protocol over a mesh network can clear retail payments in around 1.3 second even when internet connectivity is unavailable. These results, however, reflect best-case laboratory conditions: traffic is evenly distributed across nodes and authority load is still modest. Future work will therefore (i) validate the protocol on real mobile devices, (ii) integrate privacy primitives such as Coconut [16] credentials and confidential balances, and (iii) design a fee model that discourages abuse while incentivizing honest authorities. Addressing these challenges will transform the current protocol into a deployable, privacy-preserving and economically sustainable offline payment system.

References

[1] M. K. Regragui, "The african mobile wallets: an empirical analysis of the services and the anticipated trends," 2022.

[2] T. Creemers, T. Murugavel, F. Boutet, O. Omary, and T. Oikawa, "Five strategies for mobile-payment banking in africa," 2020.

[3] O. Mothobi and K. Kebotsamang, "The impact of network coverage on adoption of fintech and financial inclusion in sub-saharan africa," *Journal of Economic Structures*, vol. 13, no. 1, p. 5, 2024.

[4] digWatch, "Internet shutdowns surge in africa despite human rights concerns," 2025.

[5] M. Patnam and W. Yao, "The real effects of mobile money: Evidence from a large-scale fintech expansion," 2020.

[6] C. Lowe, "The commercial sustainability of mobile money providers in interoperability initiatives," 2024, accessed: 2024-02.

[7] F. d. S. Momo and A. Behr, "Blockchain: Effects in transactions costs from information governance," *BAR-Brazilian Administration Review*, vol. 18, no. spe, p. e200047, 2021.

[8] S. Sravan, S. Mandal, and P. Alphonse, "Lio-pay: Sustainable low-cost offline payment solution," *Electronic Commerce Research and Applications*, vol. 67, p. 101440, 2024.

[9] A. Kurt, A. Sahin, R. Harrilal-Parchment, and K. Akkaya, "Lnmesh: Who said you need internet to send bitcoin? offline lightning network payments using community wireless mesh networks," in *2023 IEEE 24th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 2023, pp. 261–270.

[10] M. Baudet, G. Danezis, and A. Sonnino, "Fastpay: High-performance byzantine fault tolerant settlement," in *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, 2020, pp. 163–177.

[11] G. Zhang, S. Zhang, M. Bachras, Y. Zhang, and H.-A. Jacobsen, "Cabinet: Dynamically weighted consensus made fast," *arXiv preprint arXiv:2503.08914*, 2025.

[12] T. Clausen and P. Jacquet, "Optimized link state routing protocol (olsr)," Tech. Rep., 2003.

[13] H. Arslanian, "Ethereum," in *The book of crypto: The complete guide to understanding bitcoin, cryptocurrencies and digital assets*. Springer, 2022, pp. 91–98.

[14] R. R. Fontes, S. Afzal, S. H. Brito, M. A. Santos, and C. E. Rothenberg, "Mininet-wifi: Emulating software-defined wireless networks," in *2015 11th International conference on network and service management (CNSM)*. IEEE, 2015, pp. 384–389.

[15] B. Gloss, M. Scharf, and D. Neubauer, "A more realistic random direction mobility model," *TD (05)*, vol. 52, pp. 13–14, 2005.

[16] A. Sonnino, M. Al-Bassam, S. Bano, S. Meiklejohn, and G. Danezis, "Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers. arxiv 2018," *arXiv preprint arXiv:1802.07344*, 2018.