Deployment of Threshold Signatures for Securing Bitcoin Transactions

1st Minh Tran
Department of Telecommunications
Brno University of Technology
Brno, Czech Republic
xtranm00@vutbr.cz

2nd Petr Dzurenda
Department of Telecommunications
Brno University of Technology
Brno, Czech Republic
0000-0002-4366-3950

***Abstract*—Blockchain technology, especially Bitcoin, has revolutionized how we think about and manage financial transactions. However, with the increasing demand and usage of blockchain technology, the security of cryptocurrency wallets has become a critical concern. Threshold signatures offer a promising solution to this problem, allowing multiple parties to sign a transaction without revealing their private keys. This article presents an Android mobile Bitcoin wallet application that uses Schnorr-based threshold signatures. The application also deploys smartwatch integration for enhanced security and usability. This integration provides an additional layer of security by requiring physical confirmation from the user before approving any transaction. Our implementation provides a secure and efficient platform for managing Bitcoin assets using threshold signatures while also providing an intuitive and easy-to-use interface for interacting with the application.**

***Index Terms*—Threshold Signature, Secret Sharing, Security of Transactions, Blockchain, Bitcoin, Android, Smartwatch**

I. Introduction

Blockchain technology is a digital ledger that enables transactions to be recorded and verified without the need for a centralized authority or intermediary. This means that transactions can occur directly between individuals or entities without the need for a trusted third party to oversee or validate the transaction. One of the most well-known applications of blockchain technology is Bitcoin [1], a decentralized digital currency operating on a blockchain network. While blockchain offers benefits such as transparency, reduced transaction costs, and enhanced security, it also poses unique security challenges. One such challenge is the use of single-key transactions, which can make them vulnerable to theft or loss.

To address this issue, multiparty cryptographic schemes and threshold signatures can enhance the security and privacy of blockchain-based systems by allowing multiple parties to sign a transaction without revealing their private keys. However, the use of threshold signatures in blockchain-based systems also poses some challenges. For example, there is a need to establish a secure and reliable protocol for distributing and managing the threshold keys. The usage of distributed signature is not a new discovery. Several blockchains, such as Bitcoin, Cardano, and Algorand, already support multi-signature transactions through multi-signature wallets, but this approach has privacy and scalability concerns. All owners and signers are publicly known, and the transaction size increases with the number of signers.

A. Contribution and Paper Structure

This paper presents several contributions: 1) development of a user-friendly Android Bitcoin wallet application that supports both single-key and threshold signatures, ensuring secure storing of private keys and sending of Bitcoins, 2) optimization of the threshold signature scheme introduced by [2], 3) integration of smartwatches into the threshold signature scheme to provide an additional layer of security by acting as hardware wallets, 4) experimenting and investigating the viability of threshold signatures on the Bitcoin blockchain, and 5) conducting extensive testing of the proposed solution.

The paper is organized as follows. Section II outlines the used terminology. Section III presents our design and development of a threshold signature wallet. Section IV presents our experimental results. In the last section, we conclude this work.

II. PRELIMINARIES

In this section, we introduce blockchain and threshold signatures technologies as they are fundamental building blocks of our implementation.

A. Bitcoin blockchain

The purpose of the Bitcoin blockchain is to provide a secure and decentralized way for individuals and organizations to conduct transactions without the need for a central authority or intermediary. The ledger is maintained by a network of nodes that work together to validate transactions and add them to the blockchain. This means that no single entity has control over the network, and all transactions are transparent and visible to anyone who wishes to view them. Transactions are secured using cryptographic algorithms and a consensus mechanism known as proof of work. This ensures that transactions cannot be tampered with or reversed, and that the network remains secure and resistant to attacks. The Bitcoin blockchain is also designed to be immutable. This ensures the integrity of the network and provides a high degree of trust in the system.

The security of Bitcoin transactions relies on the cryptography of private and public keys. Private keys are used to prove

ownership of the cryptocurrency. They are needed for signing transactions. They are kept secret by the owner of a Bitcoin wallet. Public keys, on the other hand, are used to verify transactions. They are derived from the private keys using child key derivation and are publicly visible on the blockchain.

From the Taproot hard fork [3], Bitcoin uses the Schnorr signature algorithm for securing transactions. This scheme allows more efficient and secure transactions. Unlike the traditional Elliptic Curve Digital Signature Algorithm (ECDSA) used in Bitcoin, Schnorr signatures allow for multiple inputs to be signed together, reducing the size and complexity of transactions on the blockchain. Accordingly, this permits the implementation of multi-party computation schemes like threshold signature algorithms.

B. Threshold Signatures

Threshold signatures allow multiple parties to sign a document or any other data with a shared secret key without revealing the key itself. This is achieved by dividing the secret key into several parts and distributing them among the parties. A predetermined number of parties must collaborate to generate a threshold signature. This method significantly reduces the risk of the key being lost or stolen and can adapt to changing requirements over time.

Our proposal is based on Ricci et all [2] scheme that involves 3 cryptographic primitives: Schnorr signature, Shamir secret sharing scheme and Paillier cryptosystem. The scheme consists of 3 stages: setup, signing and verification. Signing entities can be divided into main and secondary signers. The main signer initiates the signing process and has the final signature. The secondary signer responds to a request from the main device and has only a partial signature. In Algorithm 1, the Setup stage is described. Security of transmitted data and the ability to compute over encrypted data are achieved by the Paillier cryptosystem and its homomorphic properties.

Algorithm 1 `Setup`

Phase 1 - generation of public and private values

1: generate at random $d_1^{(j)}, \ldots, d_{t-1}^{(j)}$
2: generate the Paillier's key pair $(pk_{p,j}, sk_{p,j})$
3: generate at random k_j in $\mathbb{Z}_{q_{EC}}$
4: compute $pk_j = g^{k_j}$

Phase 2 - computation of Shamir secret sharing values

1: D_h generates random value $r_{j,h}$ and computes $e_{j,h} = Enc_{pk_{p,j}}(\alpha_j, r_h)$
2: D_h generates random value $v_{j,h}$ and computes $c_h = e_{j,h}^{\alpha_j^{t-2} * d_{t-1}^h} * e_{j,h}^{\alpha_j^{t-3} * d_{t-2}^h} * \cdots * e_{j,h}^{d_1^h} * Enc_{pk_{p,j}}(k_j, v_{j,h})$
3: if $h = j$, then D_j computes $f(\alpha_j) = Dec_{sk_{p,j}}(c_j - 1) + d_{t-1}^j * \alpha_j^{t-1} + \cdots + d_1^j * \alpha_j + k_j$

Figure 1 depicts a sketch of the scheme. In the Signing stage, the first thing is to specify who will join the signature. The Signing stage also has two phases. In the first phase, every signer who was specified to join the signature computes a session key. This is done through interpolation. In the second phase of this stage, every signer computes a commitment and sends it to the main signer, who adds all the commitments together and sends the aggregated commitments to secondary devices. After that, every signer computes their partial signature with the Schnorr signature algorithm and sends it to the main signer. The main signer finally aggregates all of the partial signatures into a final signature.

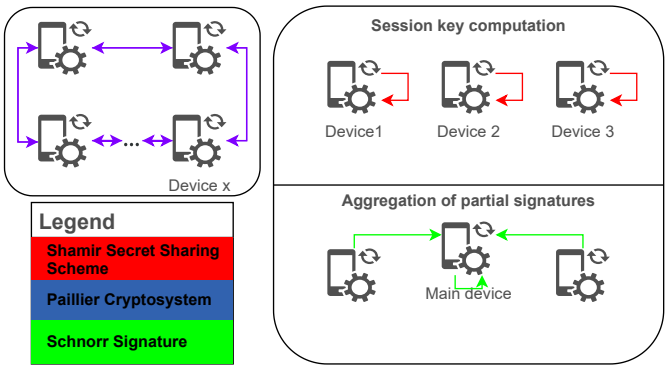

Fig. 1. Stages of the base threshold scheme.

The final Verification stage is done via the Bitcoin blockchain itself. This stage is described in Algorithm 2.

Algorithm 2 `Verification`

1: compute $e = taggedHash("BIP0340/challenge", r_{sig} + pk + msg)\%n$
2: compute $R = G * s_{sig} + pk * (n - e)$
3: **if** $(R == r)$ **return true** else **return false**

III. Design and Development of Threshold Signature Wallet

In this section, we introduce the system architecture of our threshold signature wallet application. Then, we discuss our choice of Bitcoin blockchain over other blockchains, present our modification of the base threshold signature scheme, and finally, describe our wallet's implementation details.

A. System Architecture

The architecture of the wallet application consists of four main components: 1) the wallet back end with communication with the Bitcoin blockchain, 2) the threshold signature module, 3) communication between devices via Bluetooth, and 4) the Graphical User Interface (GUI). The wallet back end is responsible for managing the user's Bitcoin funds and communicating with the Bitcoin blockchain. It consists of several sub-components, including the transaction manager, the key manager, and the blockchain client. The transaction manager is responsible for creating and signing Bitcoin transactions, while the key manager is responsible for generating, storing, and managing the user's private keys. The blockchain client communicates with the Bitcoin network to retrieve transaction

data and submit new transactions. The threshold signature module consists of several functions for generating private and public values, computation and aggregation of partial signatures, and verifying the final signature. The validity of the threshold signature is always checked before it is submitted to the Bitcoin network.

Communication channels between our system's components are depicted in Figure 2. Communication between devices is achieved by using Bluetooth technology. The devices must be paired before they can communicate with each other. Once paired, the devices can exchange data using Bluetooth to generate threshold signatures.

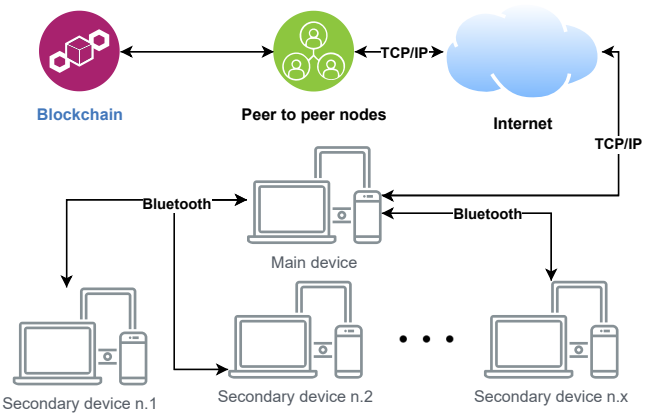

Fig. 2. Communication channels in our system.

The graphical user interface is the front end of the application, which provides a user-friendly interface for the user to interact with the application. It includes several screens, including the wallet screen, the transaction screen, and the settings screen. The wallet screen displays the user's Bitcoin balance, while the transaction screen allows the user to create and sign new transactions. The settings screen enables the user to configure the threshold signature module.

B. SELECTION OF BLOCKCHAIN TECHNOLOGY

There are several factors that lead us to choose the Bitcoin blockchain as the target platform for our threshold signature implementation. One of the most important factors is the fact that the Bitcoin blockchain supports the Schnorr signature scheme, which is a key component of the threshold signature scheme our work is based on. In addition to its support for the Schnorr signature scheme, the Bitcoin blockchain is the most popular today. This means that a wealth of documentation and resources is available for developers working with the platform, making it easier to build and maintain a high-quality implementation. Furthermore, the Bitcoin blockchain is standardized, which means a clear set of rules and specifications must be followed to ensure compatibility with the network. This makes developing and maintaining a secure and reliable implementation easier, as the rules and specifications provide a clear roadmap for developers to follow. To compare available blockchains, some of the most popular blockchains are listed and sorted by their market cap in Table I.

TABLE I
BLOCKCHAINS AND THEIR SIGNING PROTOCOLS

Blockchain	Signing protocol
Bitcoin	Schnorr signature
Ethereum	ECDSA
Binance chain	ECDSA
XRP	ECDSA
Cardano	EdDSA
Polygon	ECDSA
Solana	EdDSA
Polkadot	Schnorr signature

C. THRESHOLD SIGNATURE SCHEME MODIFICATION

The base threshold signature scheme we used in our implementation was initially developed as a standalone protocol and did not consider the specific requirements and characteristics of the Bitcoin blockchain. As a result, we needed to modify the scheme to ensure it was optimized for use in the Bitcoin blockchain. More precisely, the very last stage of the scheme, where partial Schnorr signatures are computed, had to be modified.

In the former scheme, the commitment c is computed as g^r, where r is a random number. In Bitcoin, the value r is not entirely random. Firstly value t is computed as byte-wise xor of the user's private key and a tagged hash of a random number. The value c is then computed as a tagged hash of a concatenation of t, the user's public key and the message. The challenge e is also computed differently as a tagged hash of a concatenation of c, the user's public key and the message. The prove s is computed the same way in both schemes. The final signature is a pair of c and s.

Another problem of the former scheme is the rogue-key attack. This attack can be mitigated by requiring that users prove possession of the secret key, e.g., by attaching a zero-knowledge proof of knowledge to their public keys. We used the solution from MuSig protocol [4] to exercise this approach. The challenge e is computed as $e = H_{agg}(L, X_i) * H_{sig}(\tilde{X}, R, m)$, where $\tilde{X}$ is the aggregate public key corresponding to the multiset of public keys $L = \{X_1, \ldots, X_n\}$.

In Algorithm 3, the signature generation of the base threshold signature is described. The parts highlighted in red are parts that we modified.

D. IMPLEMENTATION DETAILS

The application was developed in Kotlin using Android Studio, with a minimal version of applications programming interface (API) 21: Android 5.1 Lollipop and target API 31: Android 12 Snow Cone. This satisfies about 99.2 % of all Android devices and satisfies the condition for releasing the application on Google Play. The application supports both mobile phones and smartwatches.

The wallet back end is powered by the bitcoin-kpm library [5], which provides a comprehensive set of functions for managing the derivation of Bitcoin keys and creating and signing transactions. We use traditional mnemonic codes with a base of 24 words for the seed. But the users can choose

Algorithm 3 SignatureGen($\{s_j\}_{j\in\mathcal{J}_t}, m$)

for $j \in \mathcal{J}_t$ **do**: ▷ run privately by each D_j (i.e., MD and SDs)
  $t_j = k_j$ xor $taggedHash(\text{"}BIP0340/aux\text{"}, aux)$
  $c_j = taggedHash(t_j||pk_j||m) \bmod q_{EC}$ ▷ SDs send c_j to MD
end for
$c = \prod_{j\in\mathcal{J}_t} c_j$ ▷ run by MD, c and m sent to SDs
for $j \in \mathcal{J}_t$ **do**: ▷ run privately by each D_j (i.e., MD and SDs)
  $e = taggedHash(\text{"}BIP0340/challenge\text{"}, c||pk||m)$
  $z_j = r_j - e * s_j \bmod q_{EC}$ ▷ SDs send z_j to MD
end for
$z \leftarrow \sum_{j\in\mathcal{J}_t} z_j \bmod q_{EC}$ ▷ run by MD
return $\sigma = (c, z)$

their own 25th word to enhance the security even more. All the data is stored in MODE_PRIVATE, so no one except the application has the right to access it. For the blockchain client, we created a library that uses the Blockstream API [6] and communicates with their public node. This library provides a simple and efficient way to interact with the Bitcoin blockchain via posting and getting requests, allowing users to view their balances and perform transactions. The threshold signature module is a critical component of our application, and we developed two libraries to support it. The first library is the Paillier Cryptosystem, which is used to encrypt the secret values before they are distributed to the participants. The second library is the Bitcoin elliptic curve (secp256k1) library, which is used for every computation with the elliptic curve. These two libraries are combined to create the threshold signature module. Furthermore, we implemented Bluetooth connectivity using Wearable Data Layer API to facilitate communication between devices. This allows users to securely communicate and exchange information between devices without needing a network connection or other external infrastructure. Finally, the graphical user interface of our application uses standard .xml and activity architecture. For the main layout, the constraint layout was chosen.

IV. EXPERIMENTAL RESULTS

For our testing scenario, we used a combination of a mobile phone (Samsung Galaxy S9+) and a smartwatch (Samsung Galaxy Watch5 Pro) since they are the most common usage for potential everyday users of our application. This makes our testing scenario a 2-of-2 threshold signature. In order to evaluate the performance of our proposal, we conduct tests on both devices, specifically focusing on the execution time of the setup and signing phases. The findings about the execution time of the scheme and data transfer of the phases for each device are presented in Table II.

Our application successfully builds, signs and transmits transactions which are accepted by the Bitcoin blockchain i.e., the final verification part was also a success. One of those transactions can be viewed in Figure 3, showing a screenshot

TABLE II
BENCHMARKS IN SECONDS OF THE SETUP AND SIGNING PHASES

	Setup		Signing	
	Scheme	**Data transfer**	**Scheme**	**Data transfer**
Phone	0.13	1.04	0.12	0.71
Smartwatch	0.28		0.22	
Total	**1.45**		**1.05**	

from the Bitcoin Core application. Even though our threshold signature is constructed using multiple partial signatures, it has the same length and appearance as a normal simple single-key Schnorr signature, making it indistinguishable from any other Schnorr signature. This ensures efficient use of memory space and grants users a high level of anonymity.

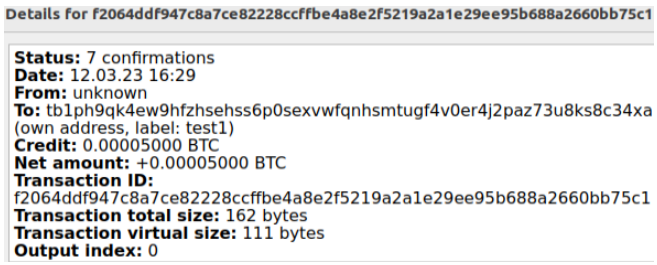

Fig. 3. Transaction made by our application.

V. CONCLUSION

In this paper, we design and implement an Android Bitcoin wallet that supports threshold signatures. The wallet application shows the practicality of using threshold signatures and smartwatches to enhance the security of Bitcoin transactions. The integration of smartwatches as an additional security layer provides users with physical authentication for transaction authorization, making transactions more secure. We have tested the application extensively, and it is capable of handling both single-signature and threshold-signature transactions efficiently and securely. Further improvements can be made to optimize the threshold signature protocol, enhance the user interface, and support other blockchain networks.

REFERENCES

[1] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008 https://bitcoin.org/bitcoin.pdf
[2] Ricci, S., Dzurenda, P., Casanova-Marqués, R., Cika, P. (2022). Threshold Signature for Privacy-Preserving Blockchain. In: , et al. Business Process Management: Blockchain, Robotic Process Automation, and Central and Eastern Europe Forum. BPM 2022. Lecture Notes in Business Information Processing, vol 459. Springer, Cham. https://doi.org/10.1007/978-3-031-16168-1_7
[3] Pieter Wuille, Jonas Nick and Tim Ruffing. Schnorr Signatures for secp256k1. 2020. https://github.com/bitcoin/bips/blob/master/bip-0340.mediawiki
[4] Gregory Maxwell, Andrew Poelstra, Yannick Seurin and Pieter Wuille. Simple Schnorr Multi-Signatures with Applications to Bitcoin. 2018. https://eprint.iacr.org/2018/068
[5] Fabrice Drouin, Bastien Teinturier, Salomon Brys, Richard Myers and Romain Boisselle. Kotlin Multiplatform Bitcoin Library. 2022. https://github.com/ACINQ/bitcoin-kmp/
[6] Nadav Ivgi, Matthew Haywood, Dimitris Tsapakidis, Benthe Carman and Hyunhum Cho. Esplora HTTP API. 2022. https://github.com/Blockstream/esplora/blob/master/API.md