Title
Decentralizing Custodial Wallets with MFKDF

Authors
Nair, Vivek
Song, Dawn

Publication Date
2023-05-01

DOI
10.1109/icbc56567.2023.10174998

Peer reviewed

Decentralizing Custodial Wallets with MFKDF

Vivek Nair
UC Berkeley
Berkeley, CA, USA
vcn@berkeley.edu

Dawn Song
UC Berkeley
Berkeley, CA, USA
dawnsong@berkeley.edu

***Abstract*—The average cryptocurrency user today faces a difficult choice between centralized custodial wallets, which are notoriously prone to spontaneous collapse, or cumbersome self-custody solutions, which if not managed properly can cause a total loss of funds. In this paper, we present a "best of both worlds" cryptocurrency wallet design that looks like, and inherits the user experience of, a centralized custodial solution, while in fact being entirely decentralized in design and implementation. In our design, private keys are not stored on any device, but are instead derived directly from a user's authentication factors, such as passwords, soft tokens (e.g., Google Authenticator), hard tokens (e.g., YubiKey), or out-of-band authentication (e.g., SMS). Public parameters (salts, one-time pads, etc.) needed to access the wallet can be safely stored in public view, such as on a public blockchain, thereby providing strong availability guarantees. Users can then simply "log in" to their decentralized wallet on any device using standard credentials and even recover from lost credentials, thereby providing the usability of a custodial wallet with the trust and security of a decentralized approach.**

***Index Terms*—mfkdf, custodial wallet, key management, mfa, applied cryptography, cryptocurrency, blockchain, sovereignty**

I. Introduction

The recent collapse of several major cryptocurrency exchange platforms offering custodial wallet services [1] has highlighted a critical vulnerability in the decentralized finance ecosystem: an over-reliance on centralized custodians of supposedly decentralized assets. By controlling large portions of cryptocurrency market capitalization [2], custodial wallets introduce concentrated failure modes into otherwise resilient decentralized platforms [3], thereby unduly reducing public trust in blockchain and crypto technologies as a whole [4].

Still, the appeal of custodial wallet services is undeniable. Users, who are notoriously bad at understanding and using public-key cryptography [5], can simply log in with familiar authentication factors like passwords while enjoying the security advantages of multi-factor authentication (MFA). Should any of these factors be lost, platforms typically provide native support for account recovery using alternative channels like email or SMS. Users benefit from the platforms' fault-tolerant and highly-available architectures, and enjoy the portability of being able to easily access their wallet from any device.

Today, self-custody wallets offer few of these benefits. Users are forced to securely store and manage cryptographic keys, and transact using pseudorandom addresses rather than human-readable identifiers. Their lack of native recovery mechanisms has led to countless notorious cases of millions of dollars of cryptocurrency being lost due to misplaced hardware [6] or forgotten passwords [7], [8]. Cross-device portability is virtually nonexistent, with no way to access funds on a new system without manually transferring keys from an old device.

It is increasingly evident that cryptocurrency users must reduce their reliance on centralized custodians in order to dampen the impact of large failure events and regain public trust in DeFi. However, it is also clear that many are unwilling to move to self-custody solutions until they offer usability on par with custodial wallets. Therefore, the goal of this paper is to provide a trustless, decentralized wallet design that looks and feels like a centralized custodial solution. In doing so, it provides the usability, portability, and recoverability of custodial wallets along with the strong security and privacy advantages of eliminating all trusted parties and committees.

Instead of storing cryptographic keys using hardware or software, our solution utilizes the multi-factor key derivation function (MFKDF) [9] to allow users to derive keys as needed using only familiar authentication factors like passwords, HMAC-based one-time password (HOTP) [10] and time-based one-time password (TOTP) [11] codes, out-of-band authentication (OOBA) such as email and SMS, and hardware tokens such as YubiKeys [12]. Users can thus simply "log in" to their wallet on any device using a human-readable username and some combination of these factors, as if it was hosted on a centralized exchange. Alternative factor combinations can be established for account recovery in case a primary authentication factor is lost. All public material necessary to facilitate this functionality, such as cryptographic salts and one-time pads, can be stored safely in the open (e.g., on a public blockchain) with no loss in security or privacy.

Contributions

To summarize, our proposed wallet design provides several key advantages over existing self-custody solutions:

1) We present the first known trustless self-custody wallet design that derives keys from common authentication factors like passwords, HOTP, TOTP, and YubiKeys (§IV-A).
2) Our approach provides strong availability guarantees with seamless cross-device portability (§IV-B).
3) Users can authenticate and transact using human-readable identifiers rather than unintelligible addresses (§IV-C).
4) Our system provides built-in support for account recovery using secondary factors like email and SMS (§IV-D).

978-8-3503-1019-1/23/$31.00 ©2023 IEEE

II. Background & Related Work

Between hardware, software, and custodial solutions, there are hundreds of cryptocurrency wallets for users to choose from today [13]. The purpose of this section is to outline the current state of cryptocurrency wallet design according to literature reviews of the field [14], [15], so as to clearly differentiate the proposed approach from known techniques.

A. Custodial Wallets

Custodial cryptocurrency wallets, whereby a centralized third-party service provider is trusted to store and manage private keys on behalf of their users, remain one of the most popular ways to enter the cryptocurrency space. In addition to the aforementioned advantages of usability, portability, and recoverability, custodial wallets are usually offered in conjunction with a centralized exchange where fiat currency can be used to purchase cryptocurrency assets.

The three largest centralized platforms, Binance [16], Coinbase [17], and Kraken [18], together account for nearly $20 billion in daily trading volume [19], orders of magnitude larger than the largest decentralized exchanges [20].

Unfortunately, this centralization has also led custodial wallets to be overrepresented in their share of major security incidents and fraud. From the infamous collapse of Mt. Gox in 2014 [21] to the recent downfall of FTX [22], custodial services have proven notoriously prone to catastrophic failure. Using such services fundamentally alters the trust assumptions underlying decentralized systems, defeating many of their security and privacy benefits. We are thus motivated to explore self-custody solutions, which are the focus of this paper.

B. Hardware Wallets

Hardware wallets, such as those offered by Ledger [23] and Trezor [24], are considered the gold standard for the safe management of cryptocurrency due to their use of a secure element for storing private keys and the absence of a general computing device which may be susceptible to malware.

Unfortunately, these devices also lie at the opposite end of the usability spectrum due to their relatively high up-front cost, cumbersome physical interface, and intrinsic limitations on the number and types of cryptocurrencies which can be used [14].

Hardware wallets offer a complete lack of portability, with no easy way to access funds on any device that is not physically connected to the wallet. The absence of built-in redundancy means that the loss of a single device can result in a total loss of funds unless additional work is done to establish backups. Accordingly, the community regularly hears stories of millions of dollars of cryptocurrency being lost due to misplaced hardware [6], [25]. Furthermore, most hardware wallets offer no native recovery mechanism other than the use of a BIP39 seed phrase [26], which must itself be stored and managed securely and is equally susceptible to loss.

Given these deficiencies, hardware wallets present an onerously high barrier to entry for most cryptocurrency users. We next turn to software-based solutions which aim to address these shortcomings but carry their own set of drawbacks.

C. Software Wallets

Non-custodial software-based wallets are a popular alternative to dedicated hardware wallets due to their lower entry cost and improved usability. While some literature reviews consider mobile wallets, such as Trust Wallet [27], and desktop wallets, such as MetaMask [28], to be entirely separate categories [14], software wallets operate using fundamentally similar mechanisms regardless of platform.

The conventional software wallet design involves storing a private key file directly on a user's file system, usually encrypted using a password. Portability is achieved by allowing this key file to be moved from one machine to another. Thus, the security of this setup is reduced to the ability to choose a secure password and manage private keys securely, both difficult tasks for the average user [5], [29]–[31].

Like with hardware wallets, a BIP39 seed phrase is often the only supported recovery mechanism, and cases of lost funds due to forgotten passwords are widespread [7], [8].

D. Multi-Factor Authentication

Within centralized applications, multi-factor authentication (MFA) has long been the go-to solution for both the insecurity of passwords as a sole authentication factor and the problem of account recovery. While a variety of MFA mechanisms are currently in use, one-time passwords (OTPs), such as HOTP [10], TOTP [11], and OOBA [32], are amongst the most popular MFA methods in use today.

Centralized exchanges like Coinbase [17] support and encourage the use of all of these methods, yet the direct protection of non-custodial cryptocurrency wallets using OTP factors is rarely seen in practice. Doing so in a cryptographically-secure way would require private keys to be locally derived from all authentication factors instead of just passwords, a feat which has not, until recently, been thought possible [9].

Unlike passwords, which are expected to remain fairly constant over time, OTPs are, by definition, intended for one-time use, and are thus expected to change upon each login. It is not immediately clear how a key can be deterministically derived from the OTP corresponding to any given login request. Instead, current attempts to construct MFA-based wallets rely either on new, purpose-built authentication factors, or on secret sharing keys across semi-trusted committees.

E. MFA Wallets

The idea of using multi-factor authentication to secure a cryptocurrency wallet is not entirely new. In fact, several works have proposed new MFA protocols specifically for the purpose of securing cryptocurrency wallets [33], [34]. However, the usability advantages of deriving private keys from familiar factors are somewhat diminished if users are required to learn and adopt a new MFA method in order to use a wallet.

Alternatively, several cryptocurrency wallet solutions have been proposed which rely upon secret-sharing a private key across a committee of nodes, at least some threshold of which are presumed to be honest [35], [36]. Users can then authenticate with these nodes using standard authentication

factors to recover the shares of their key, providing a semi-custodial experience. While more resilient than relying on a single trusted entity, these solutions do not provide the security properties of a fully-decentralized approach, in part because the nodes constituting a trusted committee are often homogeneous in design and thus subject to common vulnerabilities. By contrast, the wallet design of this paper relies neither on trusted third parties nor on partially honest committees, while also potentially allowing users to choose their own trust models through support for flexible policies (see §VI).

Deriving cryptocurrency wallet keys from popular, unmodified authentication factors like HOTP and TOTP without relying on a trusted third party or committee has long been considered a hard problem due to the dynamic nature of OTPs not being readily conducive to the derivation of a static key. However, MFKDF is designed to solve this exact problem, and by leveraging this technique, we are the first to present a cryptocurrency wallet design that is both completely trustless and fully backward-compatible with popular MFA factors.

F. MFKDF

The Multi-Factor Key Derivation Function (MFKDF) [9] is a recent improvement over password-based key derivation that incorporates multiple authentication factors into the key derivation process. Its construction provides the fundamental building block for the creation of a decentralized multi-factor authenticated wallet with support for standard, unmodified authentication factors such as HOTP and TOTP. Through a unique "key-feedback mechanism," MFKDF allows for the secure derivation of a static key from dynamic OTP factors.

MFKDF takes as input dynamic *factor witnesses* (W) and public parameters (α) and produces static key material (σ). *Factor witnesses* (W) refer to the exact values provided by a user to authenticate (e.g., a password and a 6-digit OTP). The public parameters (α), which include values like cryptographic salts and one-time pads, require no security assumptions and can safely be stored in the public, such as on a public blockchain. In most cases, these parameters must be updated upon each login ($\alpha_i \mapsto \alpha_{i+1}$). Therefore, a significant focus of this paper will be on storing and updating the public parameters (α) in a secure, portable, resilient, and highly-available manner. The resulting output (σ) can then be used to derive one or more cryptocurrency wallet private keys.

In addition to providing secure key derivation from multiple authentication factors, MFKDF also elegantly addresses the key recovery problem through a threshold mechanism that allows n authentication factors to be established, only some of which ($t < n$) are actually required to derive the key. As such, some number of authentication factors can be lost without causing the key to be lost entirely. In doing so, it allows primary authentication factors (such as a password and TOTP code) to be used for normal key derivation, and secondary authentication factors (such as email or SMS) to be used to recover the key when a primary factor, providing a very similar user experience to centralized applications.

G. Summary

In conclusion, cryptocurrency users today are still stuck with a difficult choice between convenient but fallible custodial wallets and secure but cumbersome self-custody solutions. MFKDF allows us, for the first time, to provide a "best of both worlds" non-custodial solution with the interface and user experience of a centralized service. In the following section, we more precisely describe the security and privacy goals of our solution before describing the specification in §IV.

	Custodial Wallets	Non-Custodial Wallets	MFKDF Wallet
Decentralized	✗	✓	✓
Trustless	✗	✓	✓
Portable	✓	✗	✓
Resilient	✓	✗	✓
MFA	✓	✗	✓
Familiar Factors	✓	✗	✓
Recoverable	✓	✗	✓

TABLE I
PROPERTIES OF CUSTODIAL, NON-CUSTODIAL, AND MFKDF WALLETS.

III. Problem Statement

For simplicity, this paper assumes that a cryptocurrency wallet is defined by a single private key (σ) and belongs to a single authorized user. That user is the only party in posession of the authentication factors used to constitute their wallet, and is thus the only party able to efficiently produce *factor witnesses* (W) for those factors. No assumptions are made of the entities storing public parameters (α) for a wallet. The basic security requirement of our system is that only the authorized user is able to obtain the private key (σ) of the wallet. Further desired properties of our system are as follows:

1) *Decentralized*: No external third party or committee shall have custodial access to a user's private wallet keys.
2) *Trustless*: The correct operation of the wallet shall not rely on the honest behavior of any third party or committee.[1]
3) *Portable*: Users can access their wallet on new devices regardless of the availability of previously-used devices.
4) *Resilient*: The availability a wallet does not depend on accessing a small number of specific physical devices.
5) *Multi-Factor Authenticated*: Wallet security (entropy) is a direct product of all authentication factors used.
6) *Compatible*: Familiar, unmodified authentication factors (e.g., TOTP, OOBA) can be used to access the wallet.
7) *Recoverable*: Alternative factors (e.g., SMS, email) can be used to recover the wallet with equivalent availability.

As shown in Table I, most cryptocurrency wallets in existence can offer only a subset of the desired properties. Existing attempts at building multi-factor authenticated wallets are either committee-based (not trustless) or require new, purpose-built authentication methods (not compatible with familiar

[1] In some cases, malicious colluding parties could perform an eclipse attack, leading to an unavoidable temporary denial of service for a legitimate user.

factors). In the next section, we will describe an MFKDF-based wallet architecture that meets the stated goals.

IV. System Overview

A. Multi-Factor Key Derivation

The fundamental technique of this paper is to use the multi-factor key derivation function (MFKDF) [9] to derive wallet keys dynamically from standard authentication factors rather than storing private keys in any location. The MFKDF specification (simplified here for comprehensibility) is split into setup and derive functions, as shown in Fig. 1. The setup function takes as input several authentication factors (W) and produces public parameters (α) and a secret key (σ). The public parameters must be used in the subsequent derive function along with the same factors in order to produce the same secret key, and will also be updated in the process.

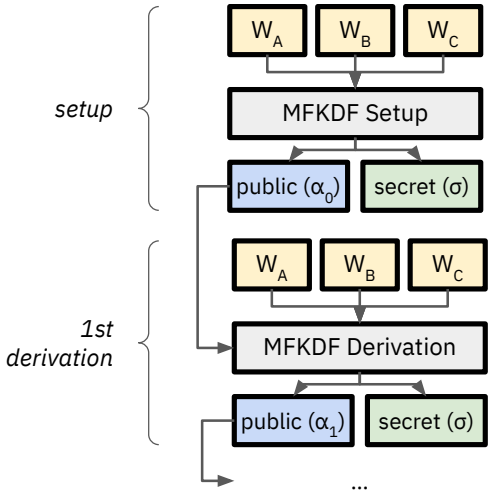

Fig. 1. Basic usage pattern of MFKDF setup and derive functions.

This parameter feed-forward mechanism of MFKDF is an important innovation that allows standard authentication factors like HOTP, TOTP, YubiKey, and OOBA to be used as part of the key derivation process. As such, a user can "sign up" using standard authentication factors, producing a private wallet key and public parameters, which can be safely stored. Later, those public parameters can be used along with the same authentication factors to "log in" to the wallet, causing the same private key to be derived if and only if correct authentication factors are provided. Thus, the key management aspect of the wallet is obscured, with the user experience matching that of a custodial service. Simultaneously, the user enjoys the security advantages of MFA, with the entropy of the key being jointly derived from all used authentication factors.

B. Availability & Portability

Although we have thus far succeeded at producing a self-custody cryptocurrency wallet that is secured by multi-factor authentication, the use of locally-stored public parameters in the derivation process limits the portability of this approach. In this section, we focus on ensuring the public parameters are always available for login on any device when desired. Because no trust assumptions are required of the party storing public material, even a public blockchain could be used to store these values. However, due to the prohibitively high cost of doing so, we instead suggest the use of a simple peer-to-peer storage approach, as shown in Fig. 2.

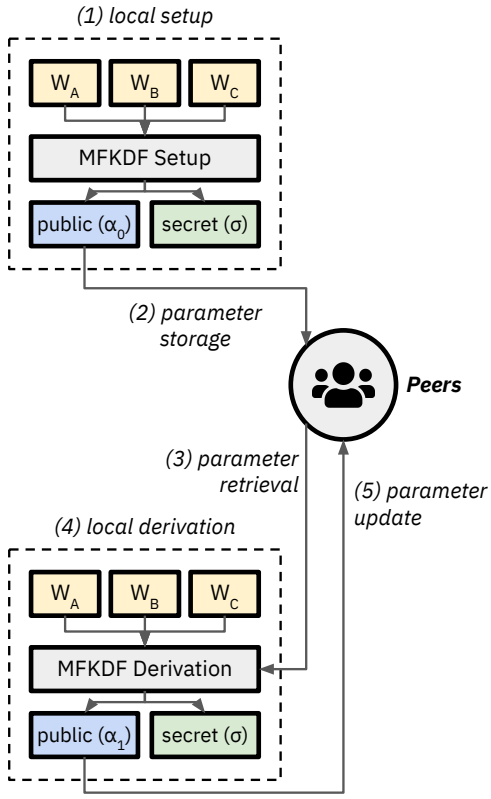

Fig. 2. Distributed system for storing public MFKDF values.

Per the suggested approach, the wallet application forms a network with its peers, and stores a copy of all legitimate MFKDF public parameters. During a local setup process, (1) an MFKDF key is established and (2) the public parameters are broadcast to the network. Later, when the wallet is accessed on another device, (3) the parameters are retrieved from the network and (4) combined with the authentication factors to re-derive the wallet key. Finally, (5) the parameters are updated and the new parameters are stored on the network. Several existing networks, such as IPFS [37], can serve this purpose.

The relatively small size ($\leq$ 10 kb) of MFKDF public parameters corresponding to each wallet makes this approach concretely practical, and it can be improved further through techniques like sharding as discussed in §VI. However, a significant drawback of the method as it stands is the potential ability for adversaries to flood the network with worthless (empty) wallets, thus exhausting available storage space and causing a denial of service for legitimate users. Therefore, we also suggest the use of an attestation mechanism that links wallet addresses to stored public material, as shown in Fig. 3.

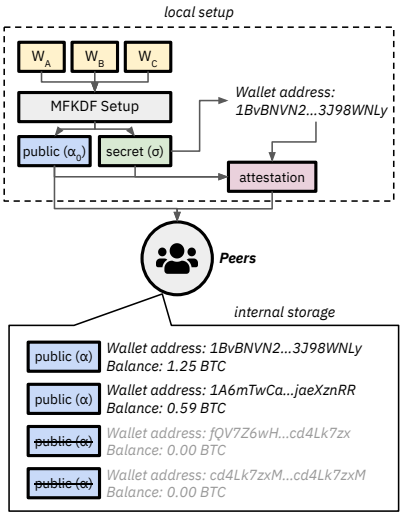

Fig. 3. Attestation mechanism for linking wallet addresses to public material.

Per the example of Fig. 3, MFKDF public material documents are signed using cryptocurrency wallet keys, allowing participants to identify the wallet associated with each stored public material object as well as its on-chain balance. If space is limited, material corresponding to addresses with zero on-chain asset value can be discarded without destruction of value, thus preventing denial-of-service attacks while ensuring high availability for legitimate (valuable) wallets.

C. Human-Readable Identifiers

While the above system and network architecture succeeds at offering high availability and preventing denial of service, it still requires users to remember long, pseudorandom wallet addresses in order to retrieve their public material and access their wallet. By contrast, a major usability advantage of centralized platforms is the ability to access accounts using a human-readable identifier, usually an email address. Thankfully, email authentication is a form of OOBA natively supported by MFKDF [9], thereby allowing for email addresses to be used to securely index stored public parameters.

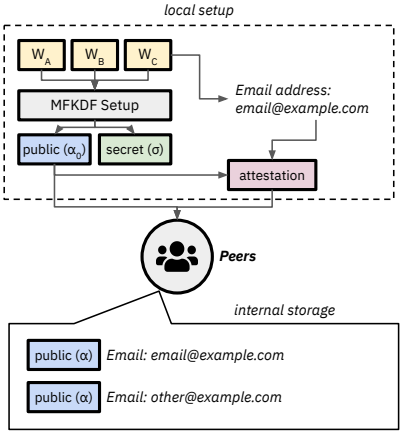

Fig. 4. Use of email OOBA for human-readable identifiers.

As illustrated in Fig. 4, email OOBA is established as one of the factors constituting the MFKDF-derived wallet key. At the end of the setup process, the email factor material is used to produce an attestation of the stored public material, allowing participants to securely store and retrieve email addresses and public material as key-value peers. Users can then "log in" to their wallet using an email address, rather than a wallet address, along with their password and other authentication factors, faithfully replicating the centralized wallet experience.

D. Account Recovery

Thus far, our proposal has centered around the simplest form of multi-factor key derivation, whereby n separate authentication factors are established, and all n are required to later derive the key. However, the MFKDF specification also provides for a threshold variant of MFKDF [9], whereby n factors are established, only t of which are later required to derive the key ($0 < t < n$). Behind the scenes, Shamir's secret sharing [38] is used to ensure that the key can be derived if and only if any t of the established factors are correctly provided.

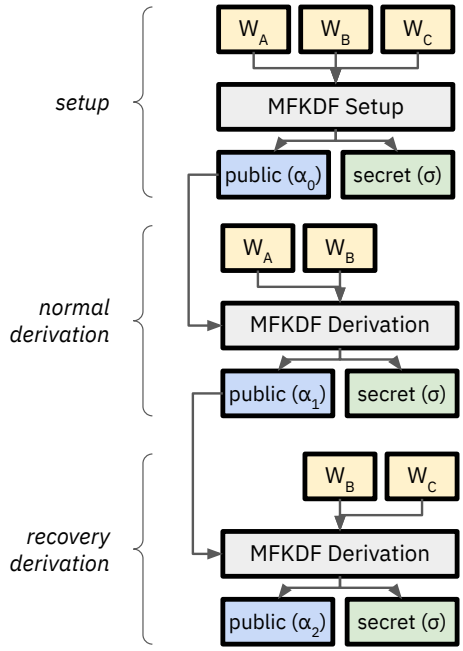

Fig. 5. Account recovery using threshold MFKDF.

The threshold version of MFKDF can be used to seamlessly facilitate account recovery in the event of a lost factor, as shown in Fig. 5. For instance, password, TOTP, and email OOBA factors can be used to establish an MFKDF-derived key according to a 2-of-3 threshold setup. During normal wallet accesses, a password and TOTP code can be supplied to derive the wallet key. However, if either the password or TOTP code is lost, the email OOBA factor can be used together with the remaining factor to recover the wallet. This setup is completely analogous to the typical account recovery process in centralized platforms, providing a familiar user experience and resilience to lost factors without weakening security.

V. Proof-of-Concept Implementation

To demonstrate the immediate practical utility of our MFKDF-based wallet architecture and provide a blueprint for its deployment, we implemented a functional MFKDF-based Ethereum web wallet application using the existing JavaScript library for MFKDF [39] and the eth-hot-wallet project [40]. The implementation uses a React.js front-end and operates as a light wallet, and must be connected to a full node to send and receive transactions. As such, no web back-end is required whatsoever, and the entire application can be delivered serverlessly, such as via IPFS.

A. Authentication

As described in §IV-A, the main advantage of using MFKDF in a decentralized wallet implementation is that users can "log in" to their wallet with traditional authentication factors as if it were an account on a centralized custodial service, without relying on committees or trust assumptions. For our implementation, we chose to use a 2-of-3 threshold MFKDF setup based on a password, YubiKey, and UUIDv4 recovery code, though any of the other authentication factors supported by MFKDF (e.g., HOTP, TOTP, OOBA) could have just as easily been used in place of the chosen factors.

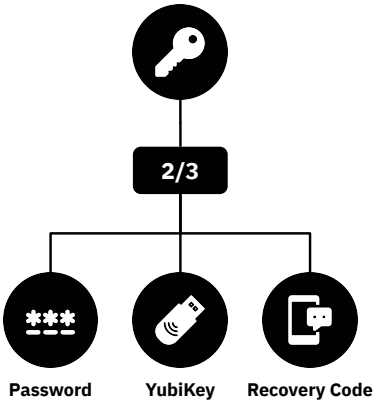

Fig. 6. Threshold MFKDF setup used in proof-of-concept implementation

As shown in Fig. 6, the 2-of-3 threshold key derivation approach allows any of the three factors to be forgotten without losing access to the underlying wallet key. We illustrate how this recovery process works in §V-D. Given that the average password provides 40 bits of entropy [30], UUIDv4 recovery codes provide 122 bits, and YubiKey via HMAC-SHA1 provides 160 bits, this key derivation policy provides at least 162 bits of security in the weakest configuration. In combination with the chosen KDF (Argon2 [41]), the wallet should provide very robust resistance to brute-force attacks.

Upon creating an "account" for the wallet, a password is selected, and a recovery code and HMAC secret for YubiKey are randomly generated. The MFKDF setup function is invoked to produce public parameters and a derived key, which is then used to display an Ethereum wallet address. The same wallet can be accessed using the parameters, password, and YubiKey.

B. Functionality

Fig. 7 shows the user interface of our proof-of-concept application after the user has successfully "signed in" (derived a key). Using their MFKDF-derived wallet key, the user is able to send and receive Ether as well as a number of ERC-20 tokens. We verified that this functionality was working correctly on both the Ropsten test network and the Ethereum main network. We do not expect difficulty adding other cryptocurrencies like Bitcoin to this wallet in the future.

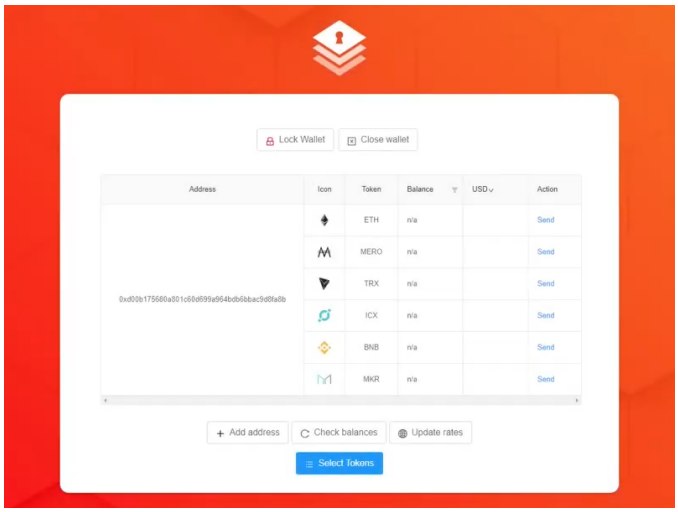

Fig. 7. User interface of proof-of-concept wallet implementation

C. Networking

Fig. 8 illustrates the network and distributed system setup used in our proof-of-concept demo. Due to the current lack of a persistent user base, we bootstrapped our system using the InterPlanetary File System (IPFS) [37] and InterPlanetary Name System (IPNS) [42] rather than using the peer-to-peer approach of §IV-B and the human-readable usernames of §IV-C. While slightly less usable than the proposed approach, this was a necessary short-term concession to avoid requiring other MFKDF wallet users to be online for availability.

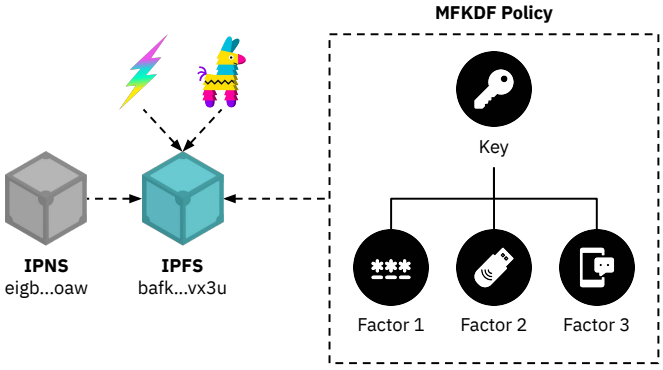

Fig. 8. Network architecture of proof-of-concept wallet implementation

Upon creating a new wallet, the public material is uploaded to IPFS, and a corresponding IPNS record is created, the address of which becomes the "username." Because other MFKDF wallet users are not currently available to store the public material, we invoke IPNS pinning services such as Pinata [43] and Fleek [44] to ensure the persistence of the public material. While such services do reintroduce an element of centralization in the short term, they do not require additional trust and only exist to ensure high availability.

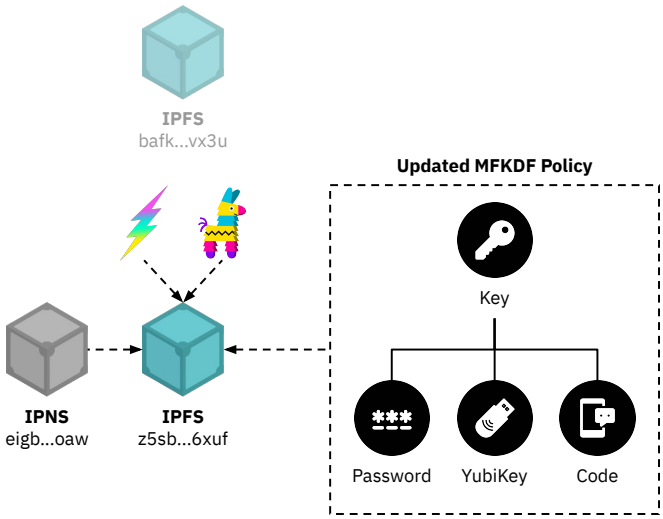

Fig. 9. Network updates upon change to MFKDF public parameters

When "logging in" to their wallet, a user provides their "username" (IPNS address, for this demo), along with at least two of their three authentication factors, such as their password and YubiKey. The MFKDF policy document is then updated, and the new policy is uploaded to IPFS. The IPNS record and pins will be updated accordingly, as shown in Fig. 9, such that the latest parameters are always used to access the wallet. The outdated version of the parameters will then go abandoned, and will stop being stored by other parties shortly thereafter.

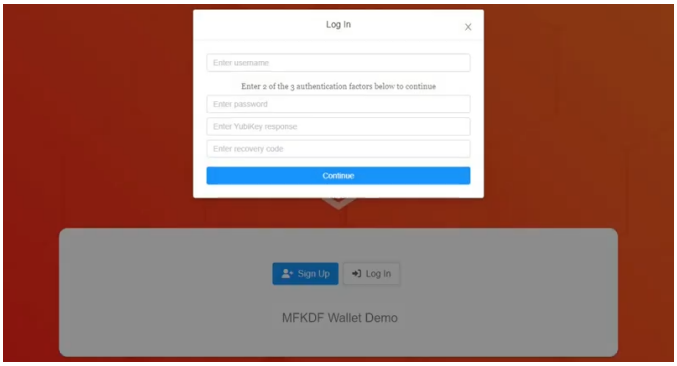

Fig. 10. Authentication interface of proof-of-concept wallet implementation

D. Recovery

Fig. 10 shows the login interface of the wallet demo application. As described in §V-A, a 2-of-3 threshold MFKDF setup was used to facilitate account recovery in case of a lost factor. For simplicity, we implemented a streamlined interface for login and recovery, whereby any two factors can be used to authenticate at any time. However, for a more traditional user experience, a login page can be configured to request only primary authentication factors, with a separate recovery flow used in case a primary factor is lost.

E. Discussion

We present the fully-featured web wallet application demo of this section to illustrate that the proposed MFKDF-based wallet solution is concretely practical and suitable for real-world deployment. While some concessions were made due to the current lack of a persistent install base, the application already largely looks and feels like a centralized custodial wallet, with "sign up" and "log in" options rather than needing manual key management, while in fact being completely decentralized in design and implementation.

We further demonstrated the backward compatibility of our solution with existing popular MFA methods such as YubiKey, with other methods supported by MFKDF like HOTP, TOTP, and OOBA easily being able to take its place. The chosen factor combination provides excellent brute-force resistance with at least 162 bits of entropy on average, while facilitating factor recovery using a 2-of-3 threshold approach.

Overall, while already practical and advantageous over existing wallets, the usability of our solution is likely to only improve over time with increased adoption and further technological advancements. Features like human-readable identifiers (§IV-C) that rely on a degree of existing adoption would serve to only further improve the experience of wallet users.

Given the comfort and familiarity of users with the interface and experience of custodial wallets, we hope to see MFKDF adopted as a primary key management approach for cryptocurrency wallets and thus drive adoption of self-custody solutions.

Limitations. It is expected that the MFKDF-based wallet constructions presented in this paper are only as secure as the underlying factors. For example, if SMS OOBA is used as a factor, but the underlying device is vulnerable to a SIM-swapping attack, then the wallet would be equally vulnerable. This is equivalent to centralized exchanges, where accounts can be compromised by adversaries knowing the credentials.

While the attestation mechanism of Fig. 3 serves to increase the cost of a brute-force attack, it does not entirely eliminate its possibility. For example, an attacker who wishes to consume 100 GB of network storage space could create 500,000 wallets, each consuming 200 KB, at a cost of about $2.5 million USD at current ETH and gas prices. The cost of such an attack could be increased by having a nominal minimum wallet value (e.g., 0.05 ETH), or by using sharded storage of wallet parameters.

Regrettably, we were not able to conduct a full usability study as part of this work, instead focusing on the security and privacy aspects along with a proof-of-concept implementation. We hope to see future work that compares the usability of the proposed wallet design to conventional self-custody wallets.

VI. Future Work

A. Security

In addition to the factors used in our implementation (e.g., YubiKey) and the other factors currently supported by MFKDF (e.g., HOTP, TOTP, OOBA), the MFKDF paper [9] also suggests that secure multi-party computation (MPC) could be used to construct additional MFKDF factors, including factors corresponding to geolocation, device identifiers, behavioral authentication, OAuth/OIDC, U2F, and more. In theory, the policy-based framework of MFKDF would then allow these factors to be combined in arbitrary ways, allowing for expressive policies such as "require 2 factors if a user is on a familiar device, and 3 factors otherwise." When used in combination with existing risk management frameworks [15], this could allow for highly flexible customization of the factors (and combinations thereof) used for authentication while strictly managing and quantifying the implied risk.

B. Privacy

Next, there are several improvements that can be made to enhance the privacy of the wallet system. Firstly, while the present proposal uses decentralized email verification as a means of providing human-readable identifiers, linking wallets to email addresses may be disadvantageous from a privacy perspective despite the existence of anonymous email services. A suitable alternative may be to use a decentralized namespace such as Ethereum Name Service (ENS) [45] to provide human-readable but anonymous usernames, though doing so in a way that is not costly to users may require further research.

Additionally, the suggested method uses a "proof of value" approach to ensure high availability of legitimate wallets while avoiding denial-of-service attacks that may result from storing an excessive volume of worthless public material. While this strategy is sound in practice, its current implementation may compromise user privacy by causing cryptocurrency wallet addresses to be linkable to the corresponding MFKDF public material. Thus, alternative approaches, such as using Zero Knowledge (ZK) proofs to validate the legitimate holdings of a wallet without revealing the underlying cryptocurrency address, could provide the same security with increased privacy.

Lastly, while moving away from centralized custodial wallets is inherently advantageous from a privacy standpoint, the current system may still reveal the identity or IP address of a wallet owner when public material is requested from the network to perform a "login" operation. As such, techniques from the fields of oblivious memory or private information retrieval may be used to obfuscate wallet access and thus enhance user privacy. Private networking protocols such as Tor [46] or Dandelion [47] can also be used to anonymize network requests, such as when broadcasting a transaction.

C. Scalability

Lastly, there are further optimizations that could be developed to improve the scalability of the proposed system. While the size of public parameters stored for an individual wallet is usually quite small ($\leq$ 10 kb), the storage space required for all users to store all valid wallets could become prohibitive if the system is adopted by millions. Thus, distributed storage techniques, such as sharding, may become necessary at scale. Existing technologies purpose-built for this use case, such as Filecoin [48], can also be used as the underlying storage solution. Lastly, some factors, such as TOTP, may require 200 kb or more of storage per user depending on configuration parameters, and the use of compression or other techniques to reduce the amount of data storage may be necessary.

VII. Conclusion

In this paper, we have presented an initial design for a cryptocurrency wallet based on multi-factor key derivation [9]. Our work is motivated by the observation that existing custodial and non-custodial wallet designs each have significant drawbacks, which we sought to rectify through a secure and user-friendly "best of both worlds" approach.

By using MFKDF to derive a wallet key on the fly from standard, unmodified authentication factors (such as passwords and software or hardware-based OTPs), we obviate the need to store private keys at all in any location. Users can simply "log in" to their wallet using their normal authentication factors and re-derive their wallet key as needed, providing the look and feel of a centralized experience with the security of multi-factor authentication. Threshold-based MFKDF allows the wallet key to be recovered even if a subset of the initially established authentication factors are forgotten. Any public material that requires persistence can be safely stored by peers without requiring a trusted committee, providing fault tolerance, redundancy, and high availability while ensuring that wallets can be accessed from any device. Thus, the proposed approach succeeds at the stated goals of §III, inheriting the decentralization and trustlessness of a self-custody solution while providing the portability, resilience, recoverability, and multi-factor authentication (with existing, familiar authentication factors) of a custodial wallet.

The wallet design of this paper is a quintessential example of a setting in which one would never consider using password-based key derivation (e.g., PBKDF2 [49]) alone. Passwords are known to be a poor solitary authentication factor in most cases [29], [30], with the risk of attacks such as credential stuffing [31] being far too high when the consequences include the theft of stored cryptocurrencies. Furthermore, the lack of a secure recovery when using passwords alone means that a forgotten password could entail the total loss of funds, which has unfortunately already occurred in several known instances [7], [8]. Multi-factor key derivation (MFKDF) is the critical improvement that provides significantly stronger security and brute-force attack resistance than password-based key derivation while also natively supporting secure key recovery in case of forgotten factors, allowing for a true custodial-like experience to be achieved in a decentralized way. Thus, we hope the current proposal serves as a turning point in the adoption of self-custody solutions amongst users who presently cling to custodial wallets for usability reasons.

References

[1] K. Huang, "Why Did FTX Collapse? Here's What to Know.," *The New York Times*, Nov. 2022. https://www.nytimes.com/2022/11/10/technology/ftx-binance-crypto-explained.html.

[2] M. Young, "Coinbase custodies 11% of entire crypto capitalization." https://cointelegraph.com/news/coinbase-custodies-11-of-entire-crypto-capitalization.

[3] J. Wood, "Custodial Wallets vs. Non-Custodial Crypto Wallets," Mar. 2022. https://www.coindesk.com/learn/custodial-wallets-vs-non-custodial-crypto-wallets/.

[4] N. Acheson, "After FTX: Rebuilding Trust in Crypto's Founding Mission," Nov. 2022. https://www.coindesk.com/layer2/2022/11/14/after-ftx-rebuilding-trust-in-cryptos-founding-mission/.

[5] A. Whitten and J. D. Tygar, "Why johnny can't encrypt: A usability evaluation of PGP 5.0," in *8th USENIX Security Symposium (USENIX Security 99)*, (Washington, D.C.), USENIX Association, Aug. 1999.

[6] N. Hartley, "Bitcoin: Missing hard drive could fund Newport crypto hub," *BBC News*, Aug. 2022. https://www.bbc.com/news/uk-wales-62381682.

[7] N. Popper, "Lost Passwords Lock Millionaires Out of Their Bitcoin Fortunes," *The New York Times*, Jan. 2021. https://www.nytimes.com/2021/01/12/technology/bitcoin-passwords-wallets-fortunes.html.

[8] C. B. C. Radio ·, "This man owns $321M in bitcoin — but he can't access it because he lost his password | CBC Radio," Jan. 2021. https://www.cbc.ca/radio/asithappens/as-it-happens-friday-edition-1.5875363/this-man-owns-321m-in-bitcoin-but-he-can-t-access-it-because-he-lost-his-password-1.5875366.

[9] V. Nair and D. Song, "Multi-factor key derivation function (mfkdf)," 2022. https://arxiv.org/abs/2208.05586.

[10] M. View, D. M'Raihi, F. Hoornaert, D. Naccache, M. Bellare, and O. Ranen, "HOTP: An HMAC-Based One-Time Password Algorithm," Request for Comments RFC 4226, Internet Engineering Task Force, Dec. 2005. https://datatracker.ietf.org/doc/rfc4226.

[11] M. View, J. Rydell, M. Pei, and S. Machani, "TOTP: Time-Based One-Time Password Algorithm," Tech. Rep. RFC 6238, Internet Engineering Task Force, May 2011. https://datatracker.ietf.org/doc/rfc6238.

[12] "Yubikey: Strong two-factor authentication." https://www.yubico.com/.

[13] F. Corva, "Cryptocurrency Wallets." https://www.finder.com/cryptocurrency/wallets.

[14] S. Suratkar, M. Shirole, and S. Bhirud, "Cryptocurrency wallet: A review," in *2020 4th International Conference on Computer, Communication and Signal Processing (ICCCSP)*, pp. 1–7, 2020.

[15] I. Eyal, "On cryptocurrency wallet design," in *3rd International Conference on Blockchain Economics, Security and Protocols (Tokenomics 2021)*, Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[16] "Binance.US | Buy, Sell, & Trade Crypto & Altcoins In The US." https://www.binance.us/.

[17] "Coinbase - Buy and Sell Bitcoin, Ethereum, and more with trust." https://www.coinbase.com/.

[18] "Kraken Cryptocurrency Exchange." https://www.kraken.com/.

[19] "Top Cryptocurrency Exchanges Ranked By Volume." https://coinmarketcap.com/rankings/exchanges/.

[20] "Top Cryptocurrency Decentralized Exchanges Ranked." https://coinmarketcap.com/rankings/exchanges/dex/.

[21] R. McMillan, "The Inside Story of Mt. Gox, Bitcoin's $460 Million Disaster | WIRED." https://www.wired.com/2014/03/bitcoin-exchange/.

[22] D. Yaffe-Bellany, "How Sam Bankman-Fried's FTX Crypto Empire Collapsed - The New York Times." https://www.nytimes.com/2022/11/14/technology/ftx-sam-bankman-fried-crypto-bankruptcy.html.

[23] "Hardware Wallet - State-of-the-art security for crypto assets." https://www.ledger.com.

[24] Trezor, "Trezor Hardware Wallet (Official) | Bitcoin & Crypto Security." https://trezor.io/.

[25] R. Browne, "Man makes last-ditch effort to recover $280 million in bitcoin he accidentally threw out." https://www.cnbc.com/2021/01/15/uk-man-makes-last-ditch-effort-to-recover-lost-bitcoin-hard-drive.html.

[26] M. Palatinus, P. Rusnak, A. Voisine, and S. Bowe, "Mnemonic code for generating deterministic keys." https://github.com/bitcoin/bips/blob/master/bip-0039.mediawiki.

[27] "Best Cryptocurrency Wallet | Ethereum Wallet | ERC20 Wallet." https://trustwallet.com/.

[28] MetaMask, "The crypto wallet for Defi, Web3 Dapps and NFTs | MetaMask." https://metamask.io/.

[29] A. Hanamsagar, S. S. Woo, C. Kanich, and J. Mirkovic, "How Users Choose and Reuse Passwords," 2016.

[30] D. Florencio and C. Herley, "A Large Scale Study of Web Password Habits," Tech. Rep. MSR-TR-2006-166, Microsoft, Nov. 2006. https://www.microsoft.com/en-us/research/publication/a-large-scale-study-of-web-password-habits/.

[31] "2020 state of the internet." https://www.akamai.com/site/en/documents/state-of-the-internet/soti-security-credential-stuffing-in-the-media-industry-report-2020.pdf.

[32] D. Handa, "What is Out-of-Band Authentication (OOBA)?." https://www.pingidentity.com/en/resources/blog/post/what-is-out-of-band-authentication-ooba.html.

[33] K. A. Taher, T. Nahar, and S. A. Hossain, "Enhanced cryptocurrency security by time-based token multi-factor authentication algorithm," in *2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)*, pp. 308–312, 2019.

[34] I. Homoliak, D. Breitenbacher, A. Binder, and P. Szalachowski, "An air-gapped 2-factor authentication for smart-contract wallets," *arXiv preprint arXiv:1812.03598*, 2018.

[35] S. He, Q. Wu, X. Luo, Z. Liang, D. Li, H. Feng, H. Zheng, and Y. Li, "A social-network-based cryptocurrency wallet-management scheme," *IEEE Access*, vol. 6, pp. 7654–7663, 2018.

[36] F. Zhu, W. Chen, Y. Wang, P. Lin, T. Li, X. Cao, and L. Yuan, "Trust your wallet: A new online wallet architecture for bitcoin," in *2017 International Conference on Progress in Informatics and Computing (PIC)*, pp. 307–311, IEEE, 2017.

[37] J. Benet, "Ipfs-content addressed, versioned, p2p file system," *arXiv preprint arXiv:1407.3561*, 2014.

[38] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, p. 612–613, nov 1979. https://doi.org/10.1145/359168.359176.

[39] V. Nair, "Javascript Implementation of a Multi-Factor Key Derivation Function (mfkdf)." https://github.com/multifactor/mfkdf.

[40] P. Laux, "Ethereum wallet." https://github.com/PaulLaux/eth-hot-wallet.

[41] A. Biryukov, D. Dinu, and D. Khovratovich, "Argon2: New generation of memory-hard functions for password hashing and other applications," in *IEEE EuroS&P*, pp. 292–302, 2016.

[42] N. Fotiou, V. A. Siris, and G. C. Polyzos, "Enabling self-verifiable mutable content items in ipfs using decentralized identifiers," in *2021 IFIP Networking Conference (IFIP Networking)*, pp. 1–6, IEEE, 2021.

[43] "Pinata | Your home for NFT media." https://www.pinata.cloud/.

[44] "Fleek: Build on the New Internet." https://fleek.co/.

[45] "Ethereum Name Service (ENS)." ens.domains.

[46] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-Generation Onion Router:," tech. rep., Defense Technical Information Center, Jan. 2004. http://www.dtic.mil/docs/citations/ADA465464.

[47] G. Fanti, S. B. Venkatakrishnan, S. Bakshi, B. Denby, S. Bhargava, A. Miller, and P. Viswanath, "Dandelion++: Lightweight Cryptocurrency Networking with Formal Anonymity Guarantees," May 2018. http://arxiv.org/abs/1805.11060.

[48] "Decentralized Storage Network." https://filecoin.io/filecoin.pdf.

[49] B. Kaliski, "PKCS #5: Password-Based Cryptography Specification Version 2.0," Request for Comments RFC 2898, Internet Engineering Task Force, Sept. 2000. https://datatracker.ietf.org/doc/rfc2898.

Acknowledgments

We appreciate the advice and feedback of Deevashwer Rathee, Xiaoyuan Liu, and Julien Piet. This work was supported in part by the National Science Foundation (NSF), by the National Physical Science Consortium (NPSC), by the Fannie and John Hertz Foundation, and by the Berkeley Center for Responsible, Decentralized Intelligence (RDI). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors alone, and do not necessarily reflect the views of the supporting entities.

Availability

We invite readers to try the MFKDF wallet demo at https://wallet.mfkdf.com. The source code for the demo is available at https://github.com/multifactor/mfkdf-wallet-demo.