SoK: Cryptocurrency Wallets – A Security Review and Classification based on Authentication Factors

***Abstract*—In this work, we review existing cryptocurrency wallet solutions with regard to authentication methods and factors from the user's point of view. In particular, we distinguish between authentication factors that are verified against the blockchain and the ones verified locally (or against a centralized party). With this in mind, we define notions for $k-factor$ authentication against the blockchain and $k-factor$ authentication against the authentication factors. Based on these notions, we propose a classification of authentication schemes. We extend our classification to accommodate the threshold signatures and signing transactions by centralized parties (such as exchanges or co-signing services). Finally, we apply our classification to existing wallet solutions, which we compare based on various security and key-management features.**

I. Introduction

Cryptocurrencies promise to revolutionize many fields and businesses, and indeed, they have been successful beyond expectations. Decentralized cryptocurrency platforms allow users to conduct monetary transfers, write smart contracts, make loans, and participate in predictive markets while benefiting from features that centralized platforms could not guarantee, such as decentralization, censorship resistance, integrity, transparency, 100% availability, etc. Cryptocurrencies utilize native crypto-tokens (a.k.a., coins), which can be transferred in transactions authenticated by private keys that belong to crypto-token owners. The users owning crypto-tokens interact with the cryptocurrency through a wallet software that manages their private keys.

Unfortunately, there are many cases of stolen keys and attacks on e.g., brain wallets [1], [2], cryptocurrency exchanges [3], [4], [5], [6], and even hardware wallets [7] with Cortex M3/M4 micro-controllers (such as Trezor, Ledger, KeepKey, etc.). Such cases have brought the research community's attention to the security issues related to key management in cryptocurrencies [8], [9], [10], [11].

A. Existing Surveys

The works of Eskandri et al. [8] and Bonneau et al. [10] proposed a categorization of cryptocurrency wallets with regard to key management. The works of Huoy et al. [12] and Erinle et al. [11] reviewed broad vulnerabilities and defenses related to wallets while they specified their categorizations. Suratkar et al. [13] focused on a review of wallets with regard to supported coins, anonymity, cost, platform support, key management, and recovery methods. Karantias [14] reviewed a few categories and instances of wallets with regard to the verification of transactions, privacy, communication complexity, and censorship.

B. Authentication Schemes of Wallets & Security Issues

We base on the works of Eskandri et al. [8] and Bonneau et al. [10], who focus on the key management approaches, and we review and extend their categorizations. Private keys are encrypted with selected passwords in *password-protected wallets*. Unfortunately, users often choose weak passwords that can be brute-forced if stolen by malware [15]; optionally, such malware may use a keylogger [10], [16]. Another similar option is to use *password-derived wallets* that generate keys based on the provided password. However, they also suffer from the possibility of weak passwords [2]. *Hardware wallets* enable only the signing of transactions, without revealing the private keys stored on the device. However, these wallets do not provide protection from an attacker with full access to the device [17], [18], [19], and more importantly, some wallets do not have a secure channel for informing the user about the details of a transaction being signed (e.g., [20]) may be exploited by malware targeting IPC mechanisms [21].

A popular option for storing private keys is to deposit them into *server-side hosted* (i.e., custodial) *wallets* and currency-exchange services [22], [23], [24], [25], [26], [27]. In contrast to the previous categories, server-side wallets imply trust in a provider (storing the private keys of the wallets), which is a potential risk of this category. Due to many cases of compromising server-side wallets [28], [29], [30], [31], [3] or fraudulent currency-exchange operators [32], *client-side hosted wallets* [33], [34], [35], [36], [37] (and their sub-category of *embedded wallets* [38], [39]) have started to proliferate. In such wallets, the main functionality, including the storage of private keys, has moved to the user side; hence, trust in the provider is reduced, but the users still depend on the provider's infrastructure.

To increase the security of former wallet categories, multi-factor authentication (MFA) is often used, which enables spending crypto-tokens only when a number of secrets are used together. Wallets from a split control category [8] provide MFA against the blockchain. This can be achieved by *threshold cryptography wallets* [9], [40], [41], *multi-signature wallets* [42], [43], [44], [45], and *state-aware smart-contract wallets* [46], [47], [48]. Nevertheless, these schemes impose additional usability implications, performance overhead, or cost of wallet devices.

In sum, while there exist studies categorizing cryptocurrency wallets with regards to security [10], [8], [12], [11], [14], there has not been any study that would deal with the classification of wallets based on locally-verified and blockchain-verified

authentication factors and their interconnection, which is the motivation for our work.

C. Contributions

With the existing key management approaches in mind, we aim to distinguish the type of wallet authentication principle(s) and procedure with regard to the wallet factors and their centralized and/or decentralized verification.

1) In particular, we propose a classification scheme for cryptocurrency wallets based on the authentication factors used for centralized and decentralized authentication.
2) To define the classification of authentication schemes of wallets, we introduce two new notions: k-factor authentication against the blockchain and k-factor authentication against the authentication factors.
3) We further extend our classification to accommodate threshold-signature approaches and centralized services that sign or co-sign transactions.
4) We extend the categorization of wallets from the previous works [8], [10], while we also apply our proposed classification to reviewed wallets and their types.

II. Classification of Authentication Schemes

In this section, we introduce our classification scheme for cryptocurrency wallets. We denote the user by $\mathbb{U}$.

A. Classification

We introduce the notion of k-factor authentication against the blockchain and k-factor authentication against the authentication factors. Using these notions, we propose a classification of authentication schemes, and we apply it to examples of existing key management solutions (see Sec. III and Tab. I).

In the context of the blockchain, we distinguish between k-factor authentication *against the blockchain* and k-factor authentication *against the authentication factors* themselves from $\mathbb{U}$'s point-of-view. For example, an authentication method may require $\mathbb{U}$ to perform 2-of-2 multi-signature in order to execute a transfer, while $\mathbb{U}$ may keep each private key stored in a dedicated device – each requiring a different password. In this case, 2FA is performed against the blockchain since all blockchain miners verify both signatures. Additionally, one-factor authentication is performed once in each device of $\mathbb{U}$ by entering a password in each of them. For clarity, we classify authentication schemes by the following:

$$\Big(Z + X_1 / \ldots / X_Z\Big), \tag{1}$$

where $Z \in \{0, 1, \ldots\}$, the first operand of "+", represents the number of authentication factors against the blockchain and $X_i \in \{0, 1, \ldots\} \mid i \in [1, \ldots, Z]$, the second operand of "+", represents the number of authentication factors against the i-th factor of Z (i.e., local authentication to access a particular factor). Hence, the "+" operator represents the connector between blockchain-verified factors and locally-verified factors. With this in mind, we remark that the previous example provides $(2 + 1/1)$-factor authentication: twice against the blockchain (i.e., two signatures), once for accessing the first device (i.e., the first password), and once for accessing the second device (i.e., the second password).

*1) **Extension for Threshold-Signature Co-Signing**:* Since the previous notation is insufficient for authentication schemes that use secret sharing [49], we extend it as follows:

$$\left(Z^{(W_1,\ldots,W_Z)} + \left(X_1^1, \ldots, X_1^{W_1}\right) / \ldots / \left(X_Z^1, \ldots, X_Z^{W_Z}\right)\right), \tag{2}$$

where Z has the same meaning as in the previous case, $W_i \in \{0, 1, \ldots\} \mid i \in [1, \ldots, Z]$ denotes the minimum number of secret shares required to use the complete i-th secret X_i. With this in mind, we remark that the aforementioned example provides $\big(2^{(1,1)} + (1)/(1)\big)$-factor authentication: twice against the blockchain (i.e., two signatures), once for accessing the first device (i.e., the first password), and once for accessing the second device (i.e., the second password). We consider an implicit value of $W_i = 1$; hence, the classification $(2 + 1/1)$ represents the same as the previous one (the first notation suffices). If one of the private keys were additionally split into two shares, each encrypted by a password, then such an approach would provide $\big(2^{(2,1)} + (1, 1)/(1)\big)$-factor authentication.

*2) **Extension for Factors Provisioned by Centralized Service(s)**:* Since the previous scheme is not sufficient to express whether some factor verified at the blockchain was signed/co-signed/produced by a centralized service such as exchange or other (e.g., upon some off-chain authentication of $\mathbb{U}$), we extend the previous notation as follows:

$$\left((Z - Y)^{(W_1,\ldots,W_{Z-Y})}\right. \tag{3}$$

$$+ \quad \left(X_1^1, \ldots, X_1^{W_1}\right) / \ldots / \left(X_{Z-Y}^1, \ldots, X_{Z-Y}^{W_{Z-Y}}\right) \tag{4}$$

$$\left. + \quad V_1 / \ldots / V_Y\right), \tag{5}$$

where the first two lines have the same meaning as Eq. 2 with the only difference that Z is "decreased" by Y – the number of blockchain-verified factors that were produced by a centralized party or more such parties (i.e., 3rd party wallet providers). Therefore, the "-" operator puts blockchain-verified factors produced by $\mathbb{U}$ versus a centralized party(-ies) into a relation. Thus, the 3rd line of Eq. 3 expresses the number of factors V_i that $\mathbb{U}$ have to present to a centralized party i for successful authentication, "authorizing" it to use a blockchain-verified secret for $\mathbb{U}$-requested operation (e.g., a signature on $\mathbb{U}$'s transaction).

For example, if $\mathbb{U}$ logins to a centralized exchange by login/password and provides OTP for the execution of an external transaction, while the centralized exchange owns the private key used for signing a transaction, then such a scheme would provide $((1 - 1) + 2)$-factor authentication. In another example, if $\mathbb{U}$ owns one private key in his local wallet (protected by a password) and requires a centralized party to

make a multi-signature on her transaction (upon authentication by login/password + OTP), such a scheme would provide $((2-1)+1+2)$-factor authentication.

III. Review of Wallet Types

We extend the previous work of Eskandari et al. [8] and Bonneau et al. [10], by categorizing and reviewing a few examples of key management solutions, while demonstrating the application of our classification (see Sec. II) to each wallet. We remark that the categories are not necessarily disjoint and one wallet may thus belong to more than one category.

A. Keys in Local Storage

In this category of wallets, the private keys are stored in plaintext form on the local storage of a machine, thus providing $(1+0)$-factor authentication. Examples that have historically enabled the use of unencrypted private key files are Bitcoin Core (until version 0.3) [50][1], Electrum (before version 1.9) [43] and MyEtherWallet (until 2018) [51][2] wallets. However, MyEtherWallet discouraged $\mathbb{U}$s from the local storage of private keys within the browsers since 2018 and mainly focused on integration with hardware wallets, while providing only a user interface for interaction with the hardware wallets. Unencrypted private keys in Bitcoin Core (which comes only as a standalone application) were possible until version 0.3, however, later the wallet required password protection and enabled integration with hardware wallets through an external bridge called Hardware Wallet Interface [52]. Electrum wallet (which comes only as a standalone application) has followed Bitcoin Core and also enabled integration with hardware wallet and two-factor authentication.

B. Password-Protected Wallets

These wallets require $\mathbb{U}$-specified password to encrypt a private key stored on the local storage, thus providing $(1+1)$-factor authentication. Examples that support this functionality are Armory Secure Wallet [42], Electrum Wallet [43], MyEtherWallet [51], Bitcoin Core [50], and Bitcoin Wallet [53]. This category addresses physical theft, yet enables the brute force of passwords and digital theft (e.g., keylogger).

C. Password-Derived Seed-Derived Wallets

Password-derived and seed-derived wallets (a.k.a., brain wallets and hierarchical deterministic wallets [54], [55]) can deterministically compute a sequence of private keys from a single password and or high-entropy seed, respectively. This approach takes advantage of the key creation in the ECDSA signature scheme that many blockchain platforms use. Examples of early password-derived wallets (for one of the configuration options) are Electrum [43], Armory Secure Wallet [42], Metamask [56], and Daedalus Wallet [57].[3] The wallets in this category provide $(1+X_1)$-factor authentication (usually $X_1 = 1$). While hierarchical deterministic wallets with high enough seed entropy provide enough resistance to brute-forcing, password-derived wallets might suffer from weak passwords [2], [58]. Vasek et al. [58] found that most of the brain wallets with weak passwords in Bitcoin were drained within 24 hours from creation, but more likely in a few minutes.

D. Hardware Storage Wallets

In general, wallets of this category include devices that can only sign transactions with private keys stored inside sealed storage, while the keys never leave the device. To sign a transaction, $\mathbb{U}$ connects the device to a machine and enters his passphrase. When signing a transaction, the device displays the transaction's data to $\mathbb{U}$, who may verify the details. Thus, wallets of this category usually provide $(1+1)$-factor authentication. Popular USB (or Bluetooth) hardware wallets containing displays are offered by Trezor [59], Ledger [60], KeepKey [61], and BitLox [62]. An example of a USB wallet that is not resistant against tampering with $\mathbb{C}$ (e.g., keyloggers) is Ledger Nano [20] – it does not have a display, hence $\mathbb{U}$ cannot verify the details of transactions being signed. An air-gapped transfer of transactions using QR codes is provided by ELLIPAL wallet [63]. In ELLIPAL, both $\mathbb{C}$ (e.g., smartphone App) and the hardware wallet must be equipped with cameras and display. $(1+0)$-factor authentication is provided by a credit-card-shaped hardware wallet from CoolBitX [64]. A hybrid approach that relies on a server providing a relay for 2FA is offered by BitBox [65]. Although a BitBox device does not have a display, after connecting to a machine, it communicates with $\mathbb{C}$ running on the machine, and at the same time, it communicates with a smartphone App through BitBox's server; each requested transaction is displayed and confirmed by $\mathbb{U}$ on the smartphone. One limitation of this solution is the lack of self-sovereignty.

E. Split Control – Threshold Cryptography

In threshold cryptography [49], [66], [67], [68], a key is split into several parties which enables the spending of crypto-tokens only when n-of-m parties collaborate. Threshold cryptography wallets provide $\left(1^{(W_1,\ldots,W_n)}+(X_1,\ldots,X_n)\right)$-factor authentication, as only a single signature verification is made on a blockchain, but n verifications are made by parties that compute a signature. Therefore, all the computations for co-signing a transaction are performed off-chain, which provides anonymity of access control policies (i.e., a transaction has a single signature) in contrast to the multi-signature scheme that is publicly visible on the blockchain. An example of this category is presented by Goldfeder et al. [9]. One limitation of this solution is a computational overhead directly proportional to the number of involved parties m (e.g.,

[1]Note that since Bitcoin Core enables also password-protected private keys, it also belongs to the next category.

[2]Note that MyEtherWallet also enables to use password-protection of private keys, and thus it also belongs to the category of password-protected wallets. At the same time, this wallet belongs to the category of client-side hosted wallets since, besides browser extension (or locally ran DAPP), it can run from the server (which is the most common option).

[3]Note that the password-based key derivation has been possible for one of the options or some versions.

for $m = 2$ it takes 13.26s). Another example of this category is a USB dongle called Mycelium Entropy [40], which, when connected to a printer, generates triplets of paper wallets using 2-of-3 Shamir's secret sharing; providing $(1^{(2)} + (0,\ 0))$-factor authentication. A hybrid example from this category (and client-side hosted wallets) is Zengo Wallet [41], which uses 2-of-2 co-signing, where the $\mathbb{U}$ owns one key (protected by PIN) and Zengo server owns another key (protected by email/password and 3D face lock). At the same time, $\mathbb{U}$ generates an encryption key that is used to backup the co-signing key at Zengo server, while the encryption key is stored at $\mathbb{U}$'s cloud provider, enabling $\mathbb{U}$ to recover the co-signing key. This example provides thus $(1^{(2)} + (1,\ 2))$-factor authentication.

F. Split Control – Multi-Signature Wallets.

In the case of multi-signature wallets, n-of-m owners of the wallet must co-sign the transaction made from the multi-owned address. Thus, the wallets of this category provide $(n\ +\ X_1/\ldots/X_n)$-factor authentication. One example of a multi-owned address approach is Bitcoin's Pay to Script Hash (P2SH).[4] Examples supporting multi-owned addresses are Lockboxes of Armory Secure Wallet [42] and Electrum Wallet [43]. A property of a multi-owned address is that each transaction with such an address requires off-chain communication. A hybrid instance of this category and client-side hosted wallets category is Trusted Coin's cosigning service [44], which provides a 2-of-3 multi-signature scheme – $\mathbb{U}$ owns a primary and a backup key, while TrustedCoin owns the third key. Each transaction is signed first by $\mathbb{U}$'s primary key and then, based on the correctness of the OTP from Google Authenticator, by TrustedCoin's key. Therefore, this approach provides $(2-1)+1+1$-factor authentication. Another hybrid instance of this category and client-side hosted wallets is Bitpay Wallet (former Copay) [45]. With Bitpay, $\mathbb{U}$ can create a multi-owned Bitpay wallet (for Bitcoin), where $\mathbb{U}$ has all keys in his machines and n-of-m keys co-sign each transaction. Transactions are resent across $\mathbb{U}$'s machines during multi-signing through Bitpay.

G. Split-Control – State-Aware Smart Contracts

State-aware smart contracts provide "rules" for how owners can spend crypto-tokens of a contract, while they keep the current setting of the rules on the blockchain. The most common example of state-aware smart contracts is the 2-of-3 multi-signature scheme that provides $(2 + X_1/X_2)$-factor authentication. An example of the 2-of-3 multi-signature approach that only supports Trezor hardware wallets is *Trezor-Multisig2of3* from Unchained Capital [46]. One disadvantage of this solution is that $\mathbb{U}$ has to own three Trezor devices, which may be an expensive solution that, moreover, relies only on a single vendor. Another example of this category, but using the n-of-m multi-signature scheme, is Parity Wallet [47]. However, two critical bugs [69], [70] have caused the multi-signature scheme to be currently disabled. The n-of-m multi-signature scheme is also used in *Gnosis Wallet* from ConsenSys [48]. The Gnosis multi-sig smart contract is also utilized in Bitpay wallet [45] (for Ethereum). A hybrid example of this category is Argent wallet [71], which runs as a smart contract that optionally enables to switch into 2-of-2 multi-signature mode, where one key is held in $\mathbb{U}$'s browser or smartphone App (protected by a password) and another key is stored at Argent's server, which co-signs $\mathbb{U}$ transaction upon successful email-based OTP verification, $((2-1)+1+1)$-factor authentication. In the case of a forgotten password, the Argent contract enables $\mathbb{U}$ (with only one private key) to switch off this feature after 7 days of inactivity. Another example of this category are SmartOTPs [72] that require the blockchain to verify signature in the first stage and One Time Password (OTP) in the second stage. The signature is provided by a hardware wallet (protected by PIN), and OTP is provided by the authenticator device or the smartphone App (protected by a password/fingerprint). Therefore, this solution also provides 2 + (1/1) authentication. Moreover, SmartOTPs enable recovery of lost secrets by last resort address that can receive all funds of the wallets without any authentication upon elapsing a certain last resort timeout of inactivity (e.g., in months).

H. Hosted Wallets

Common features of hosted wallets are that they provide an online interface for interaction with the blockchain, managing crypto-tokens, and viewing transaction history. At the same time, they also store private keys on the server side. If a hosted wallet has full control over private keys, it is referred to as a *server-side wallet*. A server-side wallet acts like a bank – the trust is centralized. Due to several cases of compromising such server-side wallets [3], [4], [5], [6], [28], [29], [30], [31], the hosted wallets that provide only an interface for interaction with the blockchain (or store only user-encrypted private keys) have started to proliferate. In such wallets, the functionality, including the storage of private keys, has moved to $\mathbb{U}$'s browser (i.e., client). We refer to these kinds of wallets as *client-side wallets* (a.k.a., hybrid wallets [8] and embedded wallets).

Server-Side Wallets. Coinbase [22] is an early example of a server-side hosted wallet, which also provides exchange services. Whenever $\mathbb{U}$ logs in or performs an operation, he authenticates himself against Coinbase's server using a password and obtains a code from Google Authenticator/Authy app/SMS/utilize passkeys.[5] Other examples of server-side wallets having similar security level to Coinbase are OKX [73] and Bitfinex [74]. Another example is Binance [23], which, on top of the login/password, requires $\mathbb{U}$ to provide 2 OTPs to perform the external operation with the wallet – one from the Google Authenticator and another one from the email on top of login and password. The wallets in this category usually provide $((1-1)+2)$-factor authentication when 2FA is enabled or $((1-1)+3)$-factor authentication when 3FA is enabled.

[4]We refer to the term *multi-owned address of P2SH* for clarity, although it can be viewed as Turing-incomplete smart contract.

[5]When making an external transaction $\mathbb{U}$ does not have to provide OTP/passkeys and password again.

	Authentication Scheme		Air-Gapped Property	Resilience to Tampering w. Client	Post-Quantum Resilience	No Off-Chain Communication	Malware Resistance	Secrets Kept Offline	Independence of Trusted Third Party	Resilience to Physical Theft	Resilience to Loss of Secrets	Comments
	Classification	**Details**										
Keys in Local Storage	$1+(0)$	Private key										
Bitcoin Core [50]	$1+(0)$	For one of the options	N	N	N	Y	N	N	Y	N	N/A	
Electrum Wallet [43]	$1+(0)$	For one of the options	N	N	N	Y	N	N	Y	Y	N/A	
MyEtherWallet [51]	$1+(0)$	For one of the options	N	N	N	Y	N	N	Y	N	N/A	
Password-Protected Wallets	$1+(1)$	Private key + encryption										
Armory Secure Wallet [42]	$1+(1)$		N	N	N	Y	N	N	Y	Y	N	
Electrum Wallet [43]	$1+(1)$		N	N	N	Y	N	N	Y	Y	N	
MyEtherWallet [51]	$1+(1)$		N	N	N	Y	N	N	Y	Y	N	
Bitcoin Core [50]	$1+(1)$		N	N	N	Y	N	N	Y	Y	N	
Bitcoin Wallet [53]	$1+(1)$		N	N	N	Y	N	N	Y	Y	N	
Password-(/Seed-)Derived Wallets	$1+(X_1)$											
Armory Secure Wallet [42]	$1+(1)$		N	N	N	Y	N	N	Y	Y	Y	
Electrum Wallet [43]	$1+(1)$		N	N	N	Y	N	N	Y	Y	Y	
Metamask [56]	$1+(1)$		N	N	N	Y	N	N	Y	Y	Y	
Daedalus Wallet [57]	$1+(2)$	2 passwords	N	N	N	Y	N	N	Y	Y	Y	
Hardware Storage Wallets	$1+(X_1)$											
Trezor [59]	$1+(1)$		N	Y	N	Y	Y	Y	Y	Y	Y	
Ledger [60]	$1+(1)$		N	Y	N	Y	Y	Y	Y	Y	Y	
KeepKey [61]	$1+(1)$		N	Y	N	Y	Y	Y	Y	Y	Y	
BitLox [62]	$1+(2)$	2 passwords*	N	Y	N	Y	Y	Y	Y	Y	Y	*Additionally, protection against the evil maid attack
CoolWallet S [64]	$1+(0)$		N	Y	N	Y	Y	Y	Y	P†	N/A	†Depending on the mode
Ledger Nano [20]	$1+(2)$	Password + GRID card	N	N	N	Y	N	Y	Y	Y	Y	
ELLIPAL wallet [63]	$1+(1)$		Y	Y	N	Y	Y	Y	Y	Y	Y	
BitBox USB Wallet [65]	$1+(2)$	1 password and App	N	Y	N	Y	Y	Y	P‡	Y	Y	‡Requires a relay server
Split Control – Threshold Cryptography	$1^{(W_1)}+(X_1^1,\ldots,X_1^{W_1})$											
Goldfeder et al. [9]	$1^{(2)}+(1,1)$	Assuming 2 devices, each protected by a password	N	Y	N	N	Y	N/A	N/A	N/A	N/A	
Mycelium Entropy [40]	$1^{(2)}+(0,0)$		N	Y	N	N	Y	Y	Y	Y	N/A	
Zengo Wallet. [41]	$1^{(2)}+(1,2)$	PIN at a device and password + 3D face lock on the server	N	N	N	N	N	N	N	Y	Y	A hybrid client-side wallet. An encrypted backup on Zengo. Encryption key in cloud backup.
Split Control – Multi-Signature Wallets	$Z+(X_1/\ldots/X_z)$											
Lockboxes of Armory Secure Wallet [42]	$Z+(X_1/\ldots/X_z)$	Z up to 7, X_i = 1	N	Y	N	N	Y	N	Y	Y	N	
Electrum Wallet [43]	$Z+(X_1/\ldots/X_z)$	Z up to 15, X_i = 1	N	Y	N	N	Y	N	Y	Y	Y	
Trusted Coin's Cosigning Service [44]	$(2-1)+1+1$	2 private keys (one co-signed) + 2 passwords and Google Auth.	N	Y	N	N	Y	N	N	Y	Y	A hybrid client-side wallet
Bitpay Wallet (former Copay) [45]	$2+(1/1)$	For one of the options (Bitcoin)	N	Y	N	N	Y	N	P	Y	Y	A hybrid client-side wallet
Split-Control – State-Aware Smart Contracts	$Z+(X_1/\ldots/X_z)$											
TrezorMultisig2of3 [46]	$2+(1/1)$	Assuming that each device is protected by a password	N	Y	N	N	Y	Y	Y	Y	Y	
Parity Wallet [47]	$Z+(X_1/\ldots/X_z)$	Z is unlimited, X_i = 1	N	Y	N	Y	Y	N	Y	Y	Y	
Gnosis Wallet [48]	$Z+(X_1/\ldots/X_z)$	Z up to 50, X_i = 1	N	Y	N	Y	Y	N	Y	N/A	Y	
Bitpay Wallet (former Copay) [45]	$Z+(X_1/\ldots/X_z)$	For one of the options (Ethereum); Z up to 50, X_i = 1	N	Y	N	Y	Y	N	Y	N/A	Y	A hybrid client-side wallet. Uses Gnosis smart contract.
Argent Wallet[71]	$(2-1)+1+1$	For one of the options. Cosigning based on OTP.	N	N	N	N	Y	N	N	Y	Y	A hybrid client-side wallet. Uses Gnosis smart contract.
SmartOTPs [72]	$2+(1/1)$	Private key and OTPs + passwords	Y°	Y	Y	Y	Y	Y	Y	Y	Y	°Fully air-gapped, if combined with ELLIPAL Wallet
Server-Side Wallets	$(1-1)+(X_1)$											
Coinbase [22], OKX [73], Bitfinex [74]	$(1-1)+(2)$	Password, Google Auth./SMS/passkeys	N	N	N	Y	N	N	N	Y	Y	
Binance [23]	$(1-1)+(3)$	Password, OTP from Google Auth./SMS/ + email	N	N	N	Y	N	N	N	Y	Y	
Client-Side Wallets	$Z+(X_1)$											
BTC Wallet [75]	$1+(2)$	Password + Biometrics or PIN	N	N	N	Y	N	N	N	Y	Y	Password-encrypted cloud backup.
Blockchain Wallet [76]	$1+(2)$	Password + Biometrics or PIN	N	N	N	Y	N	N	N	Y	Y	Password-encrypted cloud backup.
MyEtherWallet [51]	$1+(1)$		N	N	N	Y	N	N	N	N	Y	
Mycelium Wallet [33]	$1+(1)$		N	N	N	Y	N	N	N	Y	Y	
CarbonWallet [34]	$2+(2)$	2 private keys stored in browser and smartphone	N	Y	N	N	N	N	N	Y	Y	
Citowise Wallet [35]	$1+(2)$		N	Y¶	N	Y	N	P¶	N	Y	Y	¶If combined with Trezor or Ledger
Coinomi Wallet [36]	$1+(1)$		N	N	N	Y	N	N	N	Y	Y	
Infinito Wallet [37]	$1+(1)$		N	N	N	Y	N	N	N	Y	Y	
Harmony One [77]	$1+(2)$	For one of the options. OTP from Google Auth. (verified at the blockchain) + password	Y	N	N	Y	N	N	N	Y	Y	A hybrid smart contract wallet.
Embedded Wallets	$1+(X_1)$											
Thirdwallet [38]	$1+(1)$	OAuth w. Google or OTP by email	N	N	N	Y	N	N	N	N	N	The key is stored in local storage of the browser in the plain text
Beam Wallet [39]	$1+(2)$	Password and OAuth w. X or OTP by email	N	N	N	Y	N	N	N	Y	N	The encrypted key is stored in local storage of the browser

Tab. I: Comparison of state-of-the-art cryptocurrency wallets using our classification (see Sec. II) and other security features.

Client-Side Wallets. An example of a client-side hosted wallet is Bitcoin (BTC) Wallet [75], which requires (on top of encryption password) PIN or biometrics to access the wallet, and it provides 1-factor authentication against the blockchain. Moreover, it enables cloud backup of private keys, encrypted by the user-specified (unrecoverable) password. Equivalent functionality and security level as in BTC Wallet is offered by Blockchain DeFi Wallet [76], which is an independent

part of a combined client-side and server-side wallet.[6] Other examples of this category are password-encrypted wallets, like Mycelium Wallet [33], CarbonWallet [34], Citowise Wallet [35], Coinomi Wallet [36], and Infinito Wallet [37], which, in contrast to the previous examples, do not store backups of encrypted keys at the server. A 2FA against the blockchain is provided in addition to password-based authentication, in the case of CarbonWallet. In detail, the 2-of-2 multi-sig scheme uses the PC's browser and the smartphone's browser (or the app) to co-sign transactions. The wallets in this category usually provide $(1 + X_1)$-factor authentication, where X_1 is usually equal to 1 (in the cases of 2FA, it might be equal to 2). Argent wallet [71] is a wallet for Starknet (the L2 blockchain under Ethereum), which by default protects a private key stored in a browser by a password. Nevertheless, it optionally provides even more flexible features thanks to smart contracts (see state-aware smart contract below). Another example from this category is Harmony One wallet [77], which focuses on usability of small amount transfers. It requires $\mathbb{U}$ to login by a password, and then for each operation with the wallet she provides OTP from Google Authenticator only. In contrast to many other solutions, OTP is verified at the blockchain as the only factor required to execute the operation.[7] Since $\mathbb{U}$ has no private key, the relaying services of Harmony make the signing of the transaction (to pay the fees).

Embedded Wallets. Embedded wallets can be viewed as a subcategory of client-side hosted wallets since they contain most of their features. The only difference is that they do not need to run as a dedicated DAPP (on a new URL or locally as a browser extension) but they can run directly from the website of any service provider, and thus $\mathbb{U}$ does not need to leave its website nor interact with the browser extension. An example of this category is Thirdweb embedded wallet [38], which enables $\mathbb{U}$ to login to the wallet (and create its local private key) either by 3rd party authentication method OAuth [78] (through Google) or by sending OTP to $\mathbb{U}$'s email address. In both cases, $\mathbb{U}$ needs to have access to the email address of her account, which usually requires at least the password to access the email (i.e., $(1 + 1)$-factor authentication). However, since the key is stored at $\mathbb{U}$ device in plain text (in the case of browser within local storage), it is subject to extraction by malware, and moreover lacks the means for the recovery. Another similar example is Beam wallet (with Join integrated) [39], [79], which in contrast to Thirdweb wallet, provides OAuth login through X or email-based OTP verification. Nevertheless, it moreover enables encryption of locally stored private keys by a password (thus providing $(1 + 2)$-factor authentication).

IV. Security Features of Wallets

We present a comparison of wallets and approaches from Sec. III in Tab. I. In detail, we apply our proposed classification on authentication schemes, while we also survey a few selected security properties of the wallets that also contain some features from the work of Eskandri et al. [8]. In the following, we briefly describe each property.

*1) **Air-Gapped Property**:* We attribute this property (Y) to approaches that involve at least one hardware device storing secret information, which does not need a connection to a machine in order to operate.

*2) **Resilience to Tampering with the Client**:* We attribute this property (Y) to all hardware wallets that sign transactions within a device, while they require $\mathbb{U}$ to confirm transaction's details at the device (based on displayed information). Then, we attribute this property to wallets containing multiple clients that collaborate in several steps to co-sign transactions (the chance that all of them are tampered with is low).

*3) **Post-Quantum Resilience**:* We attribute this property (Y) to approaches that utilize hash-based cryptography that is known to be resilient against quantum computing attacks [80].

*4) **No Need for Off-Chain Communication**:* We attribute this property (Y) to approaches that do not require an off-chain communication/transfer of transactions among parties/devices to build a final (co-)signed transaction, before submitting it to a blockchain (applicable only for $Z \geq 2$ or $W_i \geq 2$).

*5) **Malware Resistance (e.g., Key-Loggers)**:* We attribute this property (Y) to approaches that either enable signing transactions inside of a sealed device or split signing control over secrets across multiple devices.

*6) **Secret(s) Kept Offline**:* We attribute this property (Y) to approaches that keep secrets inside their sealed storage, while they expose only signing functionality. Next, we attribute this property to paper wallets and fully air-gapped devices.

*7) **Independence of Trusted Third Party**:* We attribute this property (Y) to approaches that do not require a trusted party for operation, while we do not attribute this property to all client-side and server-side hosted wallets. We partially (P) attribute this property to approaches requiring an external relay server for their operation.

*8) **Resilience to Physical Theft**:* We attribute this property (Y) to approaches that are protected by an encryption password or PIN. We partially (P) attribute this property to approaches that do not provide password and PIN protection but have a specific feature to enforce the uniqueness of an environment in which they are used (e.g., Bluetooth pairing).

*9) **Resilience to Loss of Secrets**:* We attribute this property (Y) to approaches that provide means for the recovery of secrets (e.g., a seed of hierarchical deterministic wallets).

V. Conclusion

In sum, we proposed a classification of cryptocurrency wallets based on authentication methods and their factors. In the classification, we distinguished between centralized (or local) authentication and decentralized authentication against the blockchain. We demonstrated the application of our classification scheme in various categories and instances of the wallets that we moreover reviewed and cross-compared based on several security features from the literature.

[6]Note that server-side wallet uses different key.

[7]Note that client stores a salt for OTP, increasing the security of OTP.

REFERENCES

[1] J.-P. Buntinx, "Brain wallets are not secure and 'no one should use them,' says study," 2016. [Online]. Available: https://news.bitcoin.com/brain-wallets-not-secure-no-one-use-says-study/

[2] N. Courtois, G. Song, and R. Castellucci, "Speed optimizations in bitcoin key recovery attacks," *Tatra Mountains Mathematical Publications*, vol. 67(1), pp. 55–68, 2016.

[3] Binance, "Binance Security Breach Update," 2019. [Online]. Available: https://binance.zendesk.com/hc/en-us/articles/360028031711-Binance-Security-Breach-Update

[4] CoinDesk, "Crypto Exchange BitMart Hacked With Losses Estimated at $196M," 2021. [Online]. Available: https://www.coindesk.com/business/2021/12/05/crypto-exchange-bitmart-hacked-with-losses-estimated-at-196-million/

[5] ——, "The aftermath of Axie Infinitys '$650M Ronin Bridge hack," 2022. [Online]. Available: https://cointelegraph.com/news/the-aftermath-of-axie-infinity-s-650m-ronin-bridge-hack

[6] ——, "CoinEx hack: Compromised private keys led to $70M theft," 2023. [Online]. Available: https://cointelegraph.com/news/coinex-compromised-private-keys-behind-70-million-hack

[7] Kraken, "Kraken Identifies Critical Flaw in Trezor Hardware Wallets," 2019. [Online]. Available: https://blog.kraken.com/product/security/kraken-identifies-critical-flaw-in-trezor-hardware-wallets

[8] S. Eskandari, J. Clark, D. Barrera, and E. Stobert, "A first look at the usability of bitcoin key management," *preprint arXiv:1802.04351*, 2018.

[9] S. Goldfeder, R. Gennaro, H. Kalodner, J. Bonneau, J. A. Kroll, E. W. Felten, and A. Narayanan, "Securing bitcoin wallets via a new dsa/ecdsa threshold signature scheme," 2015.

[10] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "Sok: Research perspectives and challenges for bitcoin and cryptocurrencies," in *S&P*. IEEE, 2015, pp. 104–121.

[11] Y. Erinle, Y. Kethepalli, Y. Feng, and J. Xu, "Sok: Design, vulnerabilities, and security measures of cryptocurrency wallets," 2023.

[12] S. Houy, P. Schmid, and A. Bartel, "Security aspects of cryptocurrency wallets—a systematic literature review," *ACM Comput. Surv.*, vol. 56, no. 1, aug 2023. [Online]. Available: https://doi.org/10.1145/3596906

[13] S. Suratkar, M. Shirole, and S. Bhirud, "Cryptocurrency wallet: A review," in *2020 4th international conference on computer, communication and signal processing (ICCCSP)*. IEEE, 2020, pp. 1–7.

[14] K. Karantias, "Sok: A taxonomy of cryptocurrency wallets," Cryptology ePrint Archive, Paper 2020/868, 2020, https://eprint.iacr.org/2020/868. [Online]. Available: https://eprint.iacr.org/2020/868

[15] "Cryptocurrency-stealing malware landscape," Dell SecureWorks, 2015. [Online]. Available: http://www.opensource.im/cryptocurrency/cryptocurrency-stealing-malware-landscape-dell-secureworks.php

[16] A. Peyton, "Cyren sounds siren over bitcoin siphon scam," FinTech Futures, 2017. [Online]. Available: https://www.bankingtech.com/2017/01/cyren-sounds-siren-over-bitcoin-siphon-scam/

[17] Kraken, "Kraken Identifies Critical Flaw in Trezor Hardware Wallets," 2020. [Online]. Available: https://blog.kraken.com/post/3662/kraken-identifies-critical-flaw-in-trezor-hardware-wallets/

[18] ——, "Inside Kraken Security Labs: Flaw Found in Keepkey Crypto Hardware Wallet," 2019. [Online]. Available: https://blog.kraken.com/post/3245/flaw-found-in-keepkey-crypto-hardware-wallet/

[19] Donjon Team, "Extracting seed from Ellipal wallet," 2019. [Online]. Available: https://donjon.ledger.com/Ellipal-Security/

[20] Ledger, "Ledger Nano," 2018. [Online]. Available: https://www.ledgerwallet.com/products/1-ledger-nano

[21] T. Bui, S. P. Rao, M. Antikainen, V. M. Bojan, and T. Aura, "Man-in-the-machine: exploiting ill-secured communication inside the computer," in *27th USENIX Security Symposium (USENIX Security 18)*. USENIX Association, 2018, pp. 1511–1525.

[22] coinbase, "Coinbase," 2020. [Online]. Available: https://www.coinbase.com/

[23] Binance.com, "Binance," 2020. [Online]. Available: https://www.binance.com/

[24] Polo Digital Assets, Ltd., "Poloniex," 2020. [Online]. Available: https://poloniex.com/

[25] Payward, Inc, "Kraken," 2020. [Online]. Available: https://www.kraken.com/

[26] Luno, "Luno wallet," 2019. [Online]. Available: https://www.luno.com/wallet/

[27] Paxful, Inc., "Paxful," 2020. [Online]. Available: https://paxful.com/wallet

[28] W. Zhao, "Bithumb $31 Million Crypto Exchange Hack: What We Know (And Don't)," 2018. [Online]. Available: https://www.coindesk.com/bithumb-exchanges-31-million-hack-know-dont-know/

[29] R. Abrams and N. Popper, "Trading Site Failure Stirs Ire and Hope for Bitcoin," 2014. [Online]. Available: https://dealbook.nytimes.com/2014/02/25/trading-site-failure-stirs-ire-and-hope-for-bitcoin/

[30] Reuters, "Bitcoin Worth $72M Was Stolen in Bitfinex Exchange Hack in Hong Kong," 2016. [Online]. Available: http://fortune.com/2016/08/03/bitcoin-stolen-bitfinex-hack-hong-kong/

[31] T. Moore and N. Christin, "Beware the middleman: Empirical analysis of bitcoin-exchange risk," in *International Conference on Financial Cryptography and Data Security*. Springer, 2013, pp. 25–33.

[32] M. Vasek and T. Moore, "There's no free lunch, even using bitcoin: Tracking the popularity and profits of virtual currency scams," in *Financial Cryptography*. Springer, 2015, pp. 44–61.

[33] Mycelium LTD, "Mycelium wallet," 2019. [Online]. Available: https://wallet.mycelium.com/

[34] CarbonWallet.com, "Multi Signature Online Cryptocurrency Wallet," 2019. [Online]. Available: https://carbonwallet.com/

[35] Citowise Developments, "Citowise wallet," 2019. [Online]. Available: https://citowise.com/wallet

[36] Coinomi Ltd, "Coinomi Wallet," 2019. [Online]. Available: https://coinomi.com/

[37] Infinity Blockchain Labs Europe, "Infinito wallet," 2019. [Online]. Available: https://www.infinitowallet.io/

[38] thirdweb, "Embedded Wallet," 2024. [Online]. Available: https://thirdweb.com/dashboard/wallets/embedded

[39] Beam.eco, "Beam – Amazon Checkout," 2024. [Online]. Available: https://beam.eco/shop

[40] Mycelium Holding LTD, "Mycelium Entropy," 2019. [Online]. Available: https://mycelium.com/mycelium-entropy.html

[41] Zengo, "Zengo Wallet Security," 2024. [Online]. Available: https://zengo.com/security

[42] Armory Technologies, Inc, "Bitcoin Armory," 2016. [Online]. Available: https://www.bitcoinarmory.com

[43] Electrum Technologies GmbH, "Electrum Bitcoin wallet," 2019. [Online]. Available: https://electrum.org/

[44] TrustedCoin, LLC, "TrustedCoin cosigning service," 2019. [Online]. Available: https://trustedcoin.com

[45] Bitpay, "Bitpay Wallet (formerly Copay)," 2024. [Online]. Available: https://github.com/bitpay/wallet

[46] Unchained Capital, "TrezorMultisig2of3," 2019. [Online]. Available: https://github.com/unchained-capital/ethereum-multisig

[47] P. Technologies, "Parity Wallet," 2019. [Online]. Available: https://www.parity.io/

[48] ConsenSys, "Gnosis Wallet," 2019. [Online]. Available: https://github.com/Gnosis/MultiSigWallet

[49] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[50] Bitcoin Project, "Bitcoin Core," 2019. [Online]. Available: https://bitcoin.org/en/download

[51] MyEtherWallet, Inc, "MyEtherWallet," 2019. [Online]. Available: https://www.myetherwallet.com/

[52] Andrew Chow, "Bitcoin Hardware Wallet Interface," 2024. [Online]. Available: https://github.com/bitcoin-core/HWI

[53] Bitcoin Wallet developers, "Bitcoin Wallet," 2019. [Online]. Available: https://github.com/bitcoin-wallet/bitcoin-wallet

[54] G. Maxwell, "Deterministic wallets," 2011. [Online]. Available: https://bitcointalk.org/index.php?topic=19137

[55] Pieter Wuille, "BIP 0032 – Hierarchical deterministic wallets," 2012. [Online]. Available: https://en.bitcoin.it/wiki/BIP_0032

[56] MetaMask team, "MetaMask," 2019. [Online]. Available: https://metamask.io/

[57] Daedalus Team, "Daedalus Wallet," 2019. [Online]. Available: https://daedaluswallet.io/

[58] M. Vasek, J. Bonneau, R. Castellucci, C. Keith, and T. Moore, "The bitcoin brain drain: Examining the use and abuse of bitcoin brain wallets," in *Financial Cryptography and Data Security: 20th International Conference, FC 2016, Christ Church, Barbados, February 22–26, 2016, Revised Selected Papers 20*. Springer, 2017, pp. 609–618.

[59] Trezor, "Trezor," 2019. [Online]. Available: https://trezor.io/

[60] Ledger, "Ledger Nano S," 2019. [Online]. Available: https://www.ledger.com/products/ledger-nano-s
[61] KeepKey, "The Simple Cryptocurrency Hardware Wallet," 2019. [Online]. Available: https://www.keepkey.com/
[62] BitLox, "BitLox wallet," 2019. [Online]. Available: https://www.bitlox.com
[63] ELLIPAL, "ELLIPAL Hardware Wallet 2.0," 2019. [Online]. Available: https://www.ellipal.com/
[64] CoolBitX, "The CoolWallet S," 2019. [Online]. Available: https://coolwallet.io/
[65] SHIFT Cryptosecurity, "BitBox hardware wallet," 2019. [Online]. Available: https://shiftcrypto.ch/
[66] P. MacKenzie and M. Reiter, "Two-party Generation of DSA Signatures," in *Annual International Cryptology Conference*. Springer, 2001, pp. 137–154.
[67] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems," *Journal of Cryptology*, vol. 20, no. 1, pp. 51–83, 2007.
[68] G. R. Blakley *et al.*, "Safeguarding cryptographic keys," in *Proceedings of the national computer conference*, vol. 48, 1979, pp. 313–317.
[69] Parity Technologies, "The Multi-sig Hack: A Postmortem," 2017. [Online]. Available: https://paritytech.io/the-multi-sig-hack-a-postmortem/
[70] ——, "A Postmortem on the Parity Multi-Sig Library Self-Destruct," 2017. [Online]. Available: https://paritytech.io/a-postmortem-on-the-parity-multi-sig-library-self-destruct/
[71] Argent wallet, "Argent wallet," 2024. [Online]. Available: https://www.argent.xyz/
[72] I. Homoliak, D. Breitenbacher, O. Hujnak, P. Hartel, A. Binder, and P. Szalachowski, "Smartotps: An air-gapped 2-factor authentication for smart-contract wallets," in *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, 2020, pp. 145–162.
[73] L. OKX, "Okx wallet," 2017. [Online]. Available: https://www.okx.com/web3
[74] I. iFinex, "Bitfinex wallet," 2013. [Online]. Available: https://www.bitfinex.com
[75] Bitcoin.com, "Bitcoin (BTC) Wallet," 2024. [Online]. Available: https://wallet.bitcoin.com/bitcoin/
[76] Blockchain Luxembourg S.A., "Blockchain DeFi Wallet," 2024. [Online]. Available: https://www.blockchain.com/en/wallet#keys
[77] Harmony, "Bitfinex wallet," 2024. [Online]. Available: https://docs.harmony.one/home/general/ecosystem/wallets/1wallet
[78] M. Jones and D. Hardt, "The OAuth 2.0 Authorization Framework: Bearer Token Usage," Internet Requests for Comments, RFC Editor, RFC 6750, October 2012. [Online]. Available: https://www.rfc-editor.org/rfc/rfc6750.html
[79] CoinJurnal, "Beam wallet brings Amazon and Shopify purchases to users," 2024. [Online]. Available: https://coinjournal.net/news/beam-wallet-brings-amazon-and-shopify-purchases-to-users/
[80] M. Amy, O. Di Matteo, V. Gheorghiu, M. Mosca, A. Parent, and J. Schanck, "Estimating the cost of generic quantum pre-image attacks on sha-2 and sha-3," in *International Conference on Selected Areas in Cryptography*. Springer, 2016, pp. 317–337.