cryptography

Article

On the Proof of Ownership of Digital Wallets

Chen Wang [1], Zi-Yuan Liu [1] and Masahiro Mambo [2,*]

[1] Graduate School of Natural Science and Technology, Kanazawa University, Kanazawa 920-1192, Japan; wang880093@stu.kanazawa-u.ac.jp (C.W.); ziyaunliu@stu.kanazawa-u.ac.jp (Z.-Y.L.)
[2] Institute of Science and Engineering, Kanazawa University, Kanazawa 920-1192, Japan
* Correspondence: mambo@ec.t.kanazawa-u.ac.jp

Abstract: With the widespread adoption and increasing application of blockchain technology, cryptocurrency wallets used in Bitcoin and Ethereum play a crucial role in facilitating decentralized asset management and secure transactions. However, wallet security relies heavily on private keys, with insufficient attention to the risks of theft and exposure. To address this issue, Chaum et al. (ACNS'21) proposed a "proof of ownership" method using a "backup key" to prove ownership of private keys even when exposed. However, their interactive proof approach is inefficient in large-scale systems and vulnerable to side-channel attacks due to the long key generation time. Other related schemes also suffer from low efficiency and complex key management, increasing the difficulty of securely storing backup keys. In this paper, we present an efficient, non-interactive proof generation approach for ownership of secret keys using a single backup key. Our approach leverages non-interactive zero-knowledge proofs and symmetric encryption, allowing users to generate multiple proofs with one fixed backup key, simplifying key management. Additionally, our scheme resists quantum attacks and provides a fallback signature. Our new scheme can be proved to capture unforgeability under the computational indistinguishability from the Uniformly Random Distribution property of a proper hash function and soundness in the quantum random oracle model. Experimental results indicate that our approach achieves a short key generation time and enables an efficient proof generation scheme in large-scale decentralized systems. Compared with state-of-the-art schemes, our approach is applicable to a broader range of scenarios due to its non-interactive nature, short key generation time, high efficiency, and simplified key management system.

Keywords: digital wallet; digital signature; zero-knowledge proof; proof of ownership; symmetric encryption

check for updates

Citation: Wang, C.; Liu, Z.-Y.; Mambo, M. On the Proof of Ownership of Digital Wallets. *Cryptography* **2024**, *8*, 57. https://doi.org/10.3390/cryptography8040057

Academic Editor: Josef Pieprzyk

Received: 20 November 2024
Revised: 13 December 2024
Accepted: 14 December 2024
Published: 18 December 2024

1. Introduction

Nowadays, with the popularization of personal computers and smartphones, digital wallets have been widely applied in financial systems. Protecting these wallets and the corresponding key pairs is becoming increasingly crucial as adversary hackers constantly attempt to exploit security vulnerabilities to steal funds controlled by such wallets. In practice, users rely on a few approaches to counter such attacks. The most straightforward technique is to resort to secure hardware, i.e., hardware wallets [1]. Another widespread practice is hot–cold wallets [2]. A hot wallet is permanently connected to the network, initiating with a public key and address for received funds. On the other hand, cold wallets store secrets without a network connection. Both approaches aim to protect wallets at the network layer by isolating secrets from network connections, reducing the risk of adversary access. However, if attackers succeed at the digital signature layer, neither hardware nor hot–cold wallets remain secure.

Many digital wallets rely heavily on asymmetric key cryptographic systems as digital signatures, such as (i) the Digital Signature Algorithm (DSA), which is based on discrete logarithm cryptography first introduced by Diffie and Hellman [3], (ii) the Rivest–Shamir–Adleman (RSA) [4], and (iii) the Elliptic Curve Digital Signature Algorithm (ECDSA), which is based on Elliptic Curve Cryptography (ECC) [5]. These are collectively referred to as classical digital signature algorithms. With the development of large, fault-tolerant quantum computers, such classical digital signature schemes face significant threats. Shor's algorithm [6] can solve the discrete logarithm and integer factorization problems in $O(\log N)$ polynomial time N. This implies that any adversary possessing a large enough quantum computer can compute a user's private key within a feasible time once the public key is given. As a result, the adversary can generate valid signatures of the victim user. Recent studies, such as those in [7–9], have explored applying post-quantum cryptographic schemes. Although such schemes enable users to protect their digital assets against quantum attacks, users may fail or simply forget to migrate their classical digital signature-protected assets to systems secured by post-quantum cryptographic schemes. By the time they realize the issue, their secret key might already have been compromised. In such cases, post-quantum signatures and their implementations are not sufficient to address the problem.

In the scenario described above, proving the ownership of digital wallets becomes crucial for users. To prove ownership, simply employing zero-knowledge proofs, such as zero-knowledge proof of knowledge (ZKPK) [10] protocols, is not suitable because an adversary with knowledge of the stolen secret key can also generate a valid ZKPK proof. To address this issue, Chaum et al. (ACNS'21) [11] introduced the concepts of "proof of ownership" and "fallback signature," allowing users to prove ownership of secret keys and generate fallback signatures even when their secret keys are publicly exposed in a decentralized environment. To realize these concepts, they proposed the scheme S_{leeve}, a combination of the ECDSA signature with a hash-based eW-OTS$^+$ digital signature (an extension of the W-OTS$^+$ signature proposed by Chaum). The security definition of a proof of ownership framework is that the probability of an adversary without knowledge of the secret witness successfully forging a valid proof is negligible, which means that the proof scheme is sound. This implies that adversaries cannot forge a valid proof even if they successfully break the classical digital signature, such as ECDSA. In the S_{leeve} scheme, the proof system enables a user to generate a proof using a nested eW-OTS$^+$ digital signature scheme, which signs an unpredictable challenge generated by the verifier. Accordingly, if this post-quantum hash-based signature is unforgeable, it is infeasible for adversaries to forge the signature, thereby ensuring the soundness of the proof scheme. In their paper, the unforgeability is proved under the assumption of the second-preimage resistance of a hash function. However, their proof does not consider the quantum random oracle model (QROM). Unlike them, we directly prove the soundness of our proof scheme under QROM, allowing adversaries to perform quantum queries on random oracles.

In addition to providing a QROM-based security proof, we identify three major issues in the S_{leeve} scheme that can be improved. First, the S_{leeve} scheme involves an interactive proof protocol, requiring the prover to continuously generate proofs in response to challenges for each verification request. This requirement limits application scenarios, such as protecting long-term non-repudiation data, and undermines efficiency in a large-scale decentralized system. Second, to enable proof of ownership in classical digital signature schemes, the S_{leeve} scheme modifies the key generation approach to generate an additional backup key. Consequently, the key generation time is longer than that of a classical digital signature. The long key generation time can cause potential side channel attacks [12], especially when a user needs to generate multiple secret and public key pairs through repeated invocation of the key generation function, such as coin mixing [13] and HD wallets [14]. Third, as mentioned in [11], the key storage involves enhanced security measures utilizing cold–hot wallet technology to prevent any possible network access to the backup key. In scenarios where a prover holds multiple secret keys, the S_{leeve} scheme requires generating a corresponding backup key for each secret key. In our case, the proofs for multiple secret keys can be generated using a single fixed backup key. This means that regardless of how many secret keys users possess, they only need to store one backup key, significantly saving storage space. Additionally, when the backup key must be stored in an offline environment, our scheme provides easier and more efficient storage and retrieval compared with existing schemes.

Regarding the first issue we mentioned above, a natural idea to improve the S_{leeve} scheme is to transmit the involved interactive proof scheme into a non-interactive one employing the Fiat–Shamir transformation [15]. However, such a transformation does not address the key generation time and backup key storage issues. Considering all these three issues, following the concept of proof of ownership, we propose an approach utilizing zero-knowledge proof and symmetric encryption techniques. Specifically, our construction generates a non-interactive, post-quantum zero-knowledge proof of the relationship between the nested backup key and the secret key as the proof of ownership, enabling the users to non-interactively generate proofs of ownership of the secret keys without requiring inputs from other verifiers. In such non-interactive schemes, an adversary observing a valid proof could replicate it. Then, with cloned proof and public parameters, the adversary could pretend to other verifiers that he generated the valid proof. This potential clone attack could be addressed using Certification Authority (CA) systems and additional specific authentication algorithms. As this paper focuses solely on the proof of ownership scheme, concerns related to clone attacks are only mentioned here and not further explored.

Furthermore, our scheme requires only one fixed backup key to generate proofs of ownership for multiple secret keys, enabling a simplified key management system, especially when large-scale proofs of ownership for the secret keys are needed. Our protocol can also be nested within classical digital signature schemes in the real world and offers a more efficient key generation process than the S_{leeve} scheme. Our contributions are as follows.

- We propose a new efficient non-interactive proof of ownership scheme, featuring a simplified key management system while supporting a post-quantum fallback digital signature mechanism.
- We prove that the signing scheme in our approach achieves unforgeability under the computational indistinguishability from the Uniformly Random Distribution property of a proper hash function and that the proof scheme is sound in the quantum random oracle model.
- With evaluation, compared with state-of-the-art schemes, our scheme has a relatively efficient key generation phase and the most efficient proof generation phase in a large-scale decentralized system.

2. Background

2.1. Digital Signature

We describe a simple scenario involving two users, Alice and Bob, where Alice wants to send a message m to Bob securely. Alice wants to ensure that the message received by Bob is unchanged and that Bob can verify that the message was sent by Alice. Bob wants to confirm that the message received is intact and was sent from Alice, and he also wants to be able to prove this to a third party. In this case, Alice calls Gen to generate $\{\mathsf{sk}, \mathsf{pk}\}$. pk is public to all users, and sk is held by Alice secretly. Alice then executes Sign with the private key sk and message m, generating a signature σ. Alice transmits $\{m, \sigma\}$ to Bob, and then Bob executes Verify with $\{m, \sigma, \mathsf{pk}\}$. If Verify returns accept, then Bob has successfully received an unchanged message m with the signature proof σ from Alice.

Definition 1 (Classical Digital Signature)**.** *A classical digital signature scheme as a tuple* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$*:*

- $\mathsf{Gen}(1^n) \rightarrow (\mathsf{sk}, \mathsf{pk})$ *Given a security parameter of* 1^n *and the cryptographic key strength of* n *as input, outputs a private and public key pair* $(\mathsf{sk}, \mathsf{pk})$.
- $\mathsf{Sign}(m, \mathsf{sk}) \rightarrow \sigma$ *Given a message* m *and private key* sk *as input, outputs a signature* σ.
- $\mathsf{Verify}(m, \mathsf{pk}, \sigma) \rightarrow$ *result Given a message* m*, the public key* pk*, and signature* σ *as input, outputs 1 only if* σ *is a valid signature generated by* $\mathsf{Sign}(m, \mathsf{sk})$.

The correctness requires that, for any security parameter n and any message m,

$$\Pr[\mathsf{Verify}(m, \mathsf{pk}, \sigma) = 0 \mid (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^n), \sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)] \leq \mathsf{negl}(n) \tag{1}$$

For a classical digital signature, like ECDSA, unforgeability is a crucial security property, typically demonstrated through the EUF-CMA security game. We present the definition of EUF-CMA 1.

Definition 2 (EUF-CMA1 Security)**.** *Given a signature scheme* $\mathsf{sig} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *and the security parameter n, we say that* sig *is EUF-CMA1-secure if any PPT algorithm $\mathcal{A}$ has negligible advantage in the EUF-CMA1 game, defined as*

$$\begin{aligned}\mathsf{Adv}_{\mathcal{A}}^{\mathsf{EUF-CMA1}} = &\Pr[\mathsf{Verify}(\mathsf{pk}, m^*, \sigma^*) = 1 \wedge m^* \notin Q : (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^n),\\ &(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk}, \cdot)}(\mathsf{pk})]\end{aligned} \tag{2}$$

where $\mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot)}$ denotes the adversary's access to the signing oracle with private key sk *for each fresh message, and Q denotes the set of messages m queried by $\mathcal{A}$.*

To prove that our proof of ownership signature scheme achieves unforgeability, we assume an extra property for the hash function in our scheme.

Definition 3 (Indistinguishability)**.** *For security parameter n and a hash function H we introduced, we say that the hash function has computational indistinguishability from Uniformly Random Distribution if, for every PPT distinguisher D and arbitrary choices of parameter M, the following probability is negligible:*

$$|\Pr[x \leftarrow H(M), D(x) = 1] - \Pr[x \leftarrow \mathcal{U}, D(x) = 1]| \tag{3}$$

for the uniform distribution $\mathcal{U}$.

2.2. MPCitH-Based Signature

Multi-party computation in-the-head (MPCitH) is a paradigm to construct a zero-knowledge proof (ZKP) system from a multi-party computation (MPC) protocol, first introduced by Ishai et al. [16]. This technique can be utilized to generate signature schemes like ZKBoo [17], Picnic signature [18], and BBQ signature [19]. Given a one-way function f and an input–output pair (x, y) such that $f(x) = y$, one can construct a digital signature scheme with secret key x, public key y, and a non-interactive zero-knowledge proof of knowledge (NIZKPoK) of the secret x as a signature. Unlike other NIZK schemes relying on hardness assumptions for security, the security of these schemes solely depends on the security of the one-way function, typically a symmetric encryption algorithm like a hash function or AES function, thus enhancing the reliance of MPCitH-based signatures against post-quantum attacks. Regarding MPCitH zero-knowledge proofs, a prover creates a boolean computational circuit of d branches and commits to them, revealing $d - 1$ views to the verifier as proof of knowledge. To achieve non-interactive properties, the prover utilizes the Fiat–Shamir heuristic [20] to deterministically, yet unpredictably, decide which $d - 1$ views to send to the verifier. To clearly demonstrate the Fiat–Shamir transformation from a zero-knowledge interactive proof (ZKIP) to a NIZKPoK, as we mentioned above, following [21], we first demonstrate a definition of a multi-round general public coin interactive proof (PCIP) system (including the ZKIP system), which is also called an identification scheme.

Definition 4 (Public coin interactive proof system (PCIP))**.** *A $(2n + 1)-$ move public coin interactive proof system(PCIP) $\Pi = (\mathcal{P}, \mathcal{V})$ for a language $\mathcal{L}$ defined by means of a witness relation $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{W}$ such that $(x, w) \in \mathcal{R}$, is a $(2n + 1)-$ move two-party interactive protocol of the form, with C as a finite non-empty set, w a secret input of prover $\mathcal{P}$, and V a verification predicate of Π:*

$$\begin{array}{ccc}
\textit{Prover } \mathcal{P}(x) & & \textit{Verifier } \mathcal{V}(x) \\
 & \xrightarrow{\quad a_1 \quad} & \\
 & \xleftarrow{\quad c_1 \quad} & c_1 \overset{\$}{\leftarrow} C \\
 & \vdots & \\
 & \xrightarrow{\quad a_n \quad} & \\
 & \xleftarrow{\quad c_n \quad} & c_n \overset{\$}{\leftarrow} C \\
 & \xrightarrow{\quad z \quad} & \\
\end{array}$$

Verifier $\mathcal{V}$ accept if $V(x, a_1, \ldots, a_n, c_n, z) = 1$ is satisfied

where $a_1 \ldots a_n$ are commitments generated by the prover and $c_1 \ldots c_n$ are challenges randomly sampled from C by the verifier, z is the final output from the prover, and predicate $V(x, a_1, \ldots, a_n, c_n, z)$ outputs 1 only if the proof $(x, a_1, \ldots, a_n, c_n, z)$ is valid.

Then, the definition of a general Fiat–Shamir transformation for a PCIP system (including the ZKIP system) is demonstrated as

Definition 5 (Fiat–Shamir transformation for general PCIP (mFS))**.** *Given an $(2n+1)$-move PCIP $\Pi = (\mathcal{P}, \mathcal{V})$, a hash function H with appropriate domain and range in C, and a verification predicate V_{FS}, we define the non-interactive proof system $\mathsf{FS}[\Pi] = (\mathcal{P}^H_{\mathsf{FS}}, \mathcal{V}^H_{\mathsf{FS}})$ as follows. The prover $\mathcal{P}^H_{\mathsf{FS}}$ outputs*

$$(x, a_1, ..., a_n, z) \leftarrow \mathcal{P}^H_{\mathsf{FS}}$$

where a_i for $i = 1, ..., n$ are commitments, together with z computed by $\mathcal{P}^H_{\mathsf{FS}}$. The challenges are computed as follows:

$$c_1 = H(x, a_1) \textit{ and } c_i = H(c_{i-1}, a_i) \textit{ for } i = 2, \ldots, n,$$

The result output from the verifier $\mathcal{V}^H_{\mathsf{FS}}$ is determined by the predicate V_{FS} with $(x, a_1, c_1, \ldots, a_n, c_n, z)$ as input. $V_{\mathsf{FS}}(x, a_1, c_1, \ldots, a_n, c_n, z) = 1$ if $V(x, a_1, c_1, \ldots, a_n, c_n, z) = 1$, where $c_1 = H(x, a_1)$, $c_i = H(c_{i-1}, a_i)$, $i = 2, \ldots, n$.

In our design, we aim to prove the ownership of a secret key by embedding a backup key value and generating a NIZKPoK of the backup key. Additionally, we seek to enable a fallback signature scheme that is resilient to post-quantum attacks. To achieve this, we integrate the Banquet protocol [22] as our fallback signature scheme and modify it to produce a NIZKPoK for the backup key value within our proof generation algorithm. We employ Banquet for the following reasons:

- Although the Picnic scheme [18] can efficiently sign a message by performing a proof under the MPC-friendly block cipher LowMC [23] as one-way functions, the security of LowMC encryption suffers from various attacks, such as those described in [24,25]. We have mentioned that the security of an MPCitH-based signature scheme heavily depends on the security of symmetric encryption. Thus, we did not employ Picnic for our scheme.
- RAINer [26] and AIMer [27] are designed to achieve smaller signature sizes and improve efficiency using RAIN and AIM as one-way functions. However, research in [28,29] shows that the RAIN and AIM functions are vulnerable to algebraic attacks. Considering the crucial importance of security for our scheme, we have decided not to employ these two approaches.
- BBQ [19] and Banquet [22] aim to improve security using well-studied AES as the one-way function to address potential attacks. Considering that Banquet is more efficient and has a smaller signature size, we have utilized Banquet for our scheme.

Based on MPC-in-the-Head (MPCitH) techniques, the Banquet protocol constructs a 7-move interactive identification scheme for proving knowledge. This scheme can be converted into a non-interactive proof using the Fiat–Shamir transform, a method believed to be secure against quantum attacks. The simplified scheme operates as follows: given a key pair consisting of a random secret key $\mathsf{sk} = k$ and a public key $\mathsf{pk} = (x, y)$ such that $\mathsf{AES}_k(x) = y$, signing a message m involves a proof generation algorithm Prove, generating a non-interactive zero-knowledge proof of knowledge of k in a way that binds m to the proof. The verifier's challenges are derived from random oracles, with the message and public key as initial input. This ensures that these parameters are tied to the signature. Given an AES function $\mathsf{AES}_k(x)$: $\mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, which presents as a binary circuit, the Banquet scheme is described by three algorithms:

- $\mathsf{Gen}_b(1^n)$: given security parameter n as input, sample $x \in \mathcal{X} = \{0,1\}^n$ and $k \in \mathcal{K} = \{0,1\}^n$, and compute $y = \mathsf{AES}_k(x)$, outputs public key $\mathsf{pk} \leftarrow (x, y)$ and secret key $\mathsf{sk} \leftarrow k$.
- $\mathsf{Sign}_b(\mathsf{sk}, m)$: given a message m and secret key sk as input and compute $\sigma \leftarrow \mathsf{Prove}(\mathsf{sk}, m)$, output σ.
- $\mathsf{Verify}_b(\mathsf{pk}, m, \sigma)$: given signature σ and public key pk as input, compute the verification algorithm $\mathsf{Verify}(\mathsf{pk}, \sigma, m)$, which outputs 1 if the result of Verify is 1; otherwise, output 0.

2.3. Proof of Ownership of Digital Signature Scheme

In classical digital signature schemes, security relies on the secrecy of the private key. An adversary can forge signatures on arbitrary messages if the secret key is revealed. As discussed in Section 1, an attacker can apply a quantum attack to compromise the secret key of the classical asymmetric signature scheme. In this scenario, the classical digital signature is no longer secure, and any attacker can steal personal funds owned by the user by forging signatures with stolen secret keys. To solve such an issue, the notion of proof of ownership, first introduced by Chaum et al. [11], can be applied to construct a secure and extended digital signature scheme:

- $\mathsf{Gen}(1^n) \to (\mathsf{bk}, \mathsf{sk}, \mathsf{pk})$: given a security parameter 1^n, which defines the cryptographic key strength of n as input, it outputs a backup key bk, a secret key sk, and the corresponding public key pk.
- $\mathsf{Sign}(m, \mathsf{sk}) \to \sigma$: given a message m and a private key sk as input, it outputs a signature σ.
- $\mathsf{Verify}(m, \mathsf{pk}, \sigma) \to \mathit{result}$: Given message m, public key pk, and signature σ as input, outputs *accept* only if σ is a valid signature generated by $\mathsf{Sign}(m, \mathsf{sk})$.
- $\mathsf{Proof}(\mathsf{bk}, c) \to \pi$: given backup key bk and a challenge as input, compute a NIZKP with $\pi = \mathsf{Prove}(\mathsf{bk}, c)$, output π.
- $\mathsf{Verify–Proof}(\pi, m, \mathsf{pk}, \sigma) \to \mathit{result}$: given secret key sk, proof π, message m, public key pk, and signature σ as input, outputs *accept* only if proof verification $\mathsf{Verify}(\pi, c) = 1$, and σ is a valid signature generated by $\mathsf{Sign}(m, \mathsf{sk})$.

Compared with a classical digital signature scheme, a proof of ownership scheme adds an additional security layer by introducing a backup key for the secret key and generating a proof π with a backup key bk and challenge c as input. In the event of a secret key leakage, provers can use the Proof algorithm to generate a proof. Verifiers can use the Verify–Proof algorithm to verify both the validity of a digital signature and the ownership of the secret key. The security definition of proof of ownership of digital signature schemes is as follows.

Definition 6 (Proof of ownership)**.** *For any probabilistic polynomial time (PPT) algorithm $\mathcal{A}$, it holds*

$$\Pr[(\mathsf{pk}, \mathsf{sk}, \mathsf{bk}) \leftarrow \mathsf{Gen}(1^n) : (\pi^*) \leftarrow \mathcal{A}(\mathsf{sk}, c) \wedge \mathsf{Verify} - \mathsf{Proof}(\mathsf{pk}, \mathsf{sk}, m, \sigma, \pi^*, c^*) = 1] < \mathsf{negl}(\lambda) \quad (4)$$

for all probabilities are computed over the random coins of the generation and proof verification algorithm and adversary. c^* is the challenge generated by the adversary to generate π^*.

3. Related Work

In this paper, we address the scenario of a stolen secret key from classical digital signatures and propose a method to generate a post-quantum proof. The instinctive approach to proving the ownership of a secret in the digital signature is to use a replacement or additional secure algorithm. At this point of writing, many researchers have proposed a "drop-in replacement" in the form of a software library or hardware module. But since these algorithms have unique resources, performance, and platform considerations [30,31], coupled with insufficient proof of knowledge [32], they still need more extensions to be designed and carried out. Although fail-stop signature schemes [9] provide a forgery-proof mechanism, they neither prove ownership of the secret key nor provide fallback mechanisms, highlighting the need for additional improvements.

Another approach is to use a backup key that overrides the regular signing key when the secret key is leaked. In Chaum et al. [11], the researchers propose the use of a hash-based W-OTS$^+$ scheme to generate a proof for the regular secret key, which can be used as a backup signature scheme when the regular key is compromised or lost. However, the S_{leeve} scheme limits the scenario of longer-term non-interactive protection of data because the proof generation of a S_{leeve} scheme requires a challenge as a common input to compute the proof, which can also be treated as an interactive identification scheme. Moreover, the key generation time of the S_{leeve} scheme is 5–10 times slower than ECDSA, and it is hard to apply the S_{leeve} scheme as a classical digital signature scheme due to the low efficiency.

In [33], researchers enhance the security of digital signature schemes by randomly choosing a pre-image value related to the secret key in the key generation step and generating two zero-knowledge proofs of the public key generation. Although the authors aim to create a post-quantum extension scheme for the classical digital signature schemes, their scheme can also be utilized to prove the ownership of a revealed secret key. Compared with our scheme, the authors attempt to compute the secret key of an ECDSA encryption scheme with a corresponding pre-image value utilizing the proper hash function and generating two zero-knowledge proofs for the hash computation and elliptic curve computation. However, the extended signature scheme needs to be simplified. We argue that the zero-knowledge proof for the public generation is necessary because, under the quantum computing attack, one adversary can compute the secret key of a classical ECDSA scheme with a given public key. The ZKBoo scheme they used as a zero-knowledge proof scheme is not as competitive in execution time as other schemes like Picnic [34], BBQ [19], Banquet [22], and AIMer [27], which have been evaluated in Section 2.2.

4. Our Approach

We describe a scenario for digital wallets using classical digital signatures (like ECDSA) under the attack of secret key leakage. Users need to prove ownership of their digital funds, a requirement that can be effectively addressed through our proposed approach. While the S_{leeve} scheme proposed by Chaum et al. [11] offers a solution, it suffers from a long key generation time and requires an interactive proof scheme, significantly restricting its application, as discussed in Section 1.

To address these challenges, we propose a proof of ownership scheme that utilizes non-interactive zero knowledge (NIZK) to prove the knowledge of the nested backup key. Unlike the original scheme, our design efficiently generates ECDSA secret keys using AES encryption and a hash function, with a single, randomly selected backup key as input. Meanwhile, the challenge in our scheme is generated solely by the prover, employing a proper hash function with the public parameters. With these innovations, our scheme enables a prover to non-interactively generate multiple proofs with one query to the backup key while also reducing the key generation compared with the S_{leeve} scheme. The required building blocks and our proposed scheme are introduced as follows:

- ECDSA: ECDSA is commonly implemented in digital wallets as a signing scheme due to its use of an unpredictable random number generator for private key generation. This generator produces outputs that perfectly match the output space of a one-way hash function, which is also widely used in digital wallets.
- Hash function: In our approach, the hash function's output space is a finite field that aligns well with the curve in ECDSA.
- AES function: AES is a 128-bit block cipher based on a substitution–permutation network. The key length is allowed to be 128, 192, or 256 bits and can be considered as a 4×4 matrix of elements in $\mathbb{F}_{2^8}$. The AES S-box computes a multiplicative inverse in the field $\mathbb{F}_{2^8}$. In our case, the key k of the AES function should be properly selected so the key generation repeats the AES computation $y = \mathsf{AES}_k(x)$ until there are no S-boxes in the computation of y with input 0.
- Zero-knowledge proof: Our approach uses a non-interactive and post-quantum secure zero-knowledge proof generated from the Banquet protocol [22]. We modify Banquet's signing and verification scheme by removing the message input and generating a NIZKPoK for an AES encryption key. In Banquet signature, signing a message m involves adding the message into a hash function of the challenge and verifying it. Without message input, the generated NIZKPoK does not suffer from security or efficiency issues. The zero-knowledge proof demonstrates that for given (x, y), the prover possesses an AES encryption key k satisfying $\mathsf{AES}_k(x) = y$. The parameters and functions used in the proof are the following: the security parameter n, the total number of parties N in the underlying MPC protocol, the total number of S-boxes L, the number of parallel repetitions τ, the commitment function Commit, challenge computing hash functions (H_1, H_2, H_3), and the hash expand function Expand and random tape sample function tape.

With the setting definition mentioned above, our protocol can be evaluated as five algorithms: Gen, Sign, Verify, Proof, and Verify–Proof, which will be demonstrated sequentially.

- Realization of Gen: The function Gen, shown in Algorithm 1, is an extended function of the Gen algorithm of the ECDSA. Additional steps are taken to generate a backup key bk and use it to compute the secret key sk and the public key pk. It samples $\mathsf{bk} \leftarrow \{0,1\}^n$, $x \leftarrow \{0,1\}^n$, and computes $\mathsf{AES}_{\mathsf{bk}}(x) = y$. Here, during the AES process, checking whether 0 is the input of S-boxes is required. This confirms that a prover can correctly generate a NIZK of the knowledge of the backup key in Algorithm 4. The sampled value x will be stored by the prover and will be the input value in Algorithm 4.
- Realization of Sign: the function Sign, shown in Algorithm 2, is the same as the signature phase of the ECDSA.
- Realization of Verify: The function Verify is shown in Algorithm 3. In our approach, Verify only verifies the validity of a signed message.
- Realization of Proof: The function shown in Algorithm 4 takes (bk, x) as an input pair; the value x is the AES input in Algorithm 1, corresponding to the secret key for which we want to generate proof of ownership. It computes the AES function as $y = \mathsf{AES}_{\mathsf{bk}}(x)$ and a NIZKPoK of backup key bk with settled language $l \leftarrow \{(x, y) \mid \exists \mathsf{bk} \text{ s.t. } \mathsf{AES}_{\mathsf{bk}}(x) = y\}$, obtaining proof π_n, and outputs the proof $\pi = (\pi_n, x, y)$.
- Realization of Verify–Proof: Algorithm 5 takes a secret and public key pair $(\mathsf{sk}, \mathsf{pk})$ and proof π as input and verifies the correctness of the zero-knowledge proof generated by Algorithm 4 as well as the correctness of the secret and public key pair $(\mathsf{sk}, \mathsf{pk})$.

Algorithm 1 Proof of ownership ECDSA: key generation Gen

```
Input: ECC base point $G$; ECC order $P$; security parameter $n$
Output: $(\mathsf{bk}, \mathsf{sk}, \mathsf{pk})$
  Sample a backup key $\mathsf{bk} \leftarrow \{0,1\}^n$
  Sample a random value $x \leftarrow \{0,1\}^n$
  Compute the result of AES function $y = \mathsf{AES}_{\mathsf{bk}}(x)$
  for There are zero input S-boxes in AES computation do
    Resample $x, \mathsf{bk} \leftarrow \{0,1\}^n$
    Recompute $y = \mathsf{AES}_{\mathsf{bk}}(x)$
  end for
  Compute secret key $\mathsf{sk} = H(x, y)$
  Compute public key $\mathsf{pk} = \mathsf{sk} \cdot G \bmod P$
  return $(\mathsf{bk}, \mathsf{sk}, \mathsf{pk})$
```

Algorithm 2 Proof of ownership ECDSA: signing Sign

```
Input: ECC base point $G$; ECC order $P$; secret key $\mathsf{sk}$; message $m$
Output: $\sigma$
  Generate signature random value $r$
  Compute $r^{-1}$ where $r * r^{-1} = 1 \bmod P$
  Compute $R = (r_x, r_y)$ where $R = r \cdot G \bmod P$
  Compute the hash of the message: $h = H(m)$
  Compute $s$ where $s = r^{-1} * (h + r_x * \mathsf{sk}) \bmod P$
  Destroy $r, r^{-1}$
  return $\sigma = (r_x, s)$
```

Algorithm 3 Proof of ownership ECDSA: verification Verify

```
Input: ECC base point $G$; ECC order $P$; secret key $\mathsf{sk}$; message $m$; public key $\mathsf{pk}$;
  signature $\sigma$
Output: $result$
  Compute $s^{-1}$ where $s * s^{-1} = 1 \bmod P$
  Compute the hash of the message $h = H(m)$
  Compute $u_1 = s^{-1} * h \bmod P$
  Compute $u_2 = s^{-1} * r_x \bmod P$
  Compute $V = (v_x, v_y)$ where $V = u_1 \cdot G + u_2 \cdot \mathsf{pk}$
  if $v_x \neq r_x$ then
    return 0
  end if
  return 1
```

Algorithm 4 Proof of ownership ECDSA: proof generation Proof

```
Input: backup key $\mathsf{bk}$; sampled value $x$ in Algorithm 1
Output: $\pi$
  Compute the AES function $y = \mathsf{AES}_{\mathsf{bk}}(x)$
  Set language $l \leftarrow \{(x, y) \mid \exists \mathsf{bk} \text{ s.t. } \mathsf{AES}_{\mathsf{bk}}(x) = y\}$
  Compute NIZKPoK of $\mathsf{bk}$ from $l$, obtain proof $\pi_n$
  Output $\pi = (\pi_n, x, y)$
  return $\pi$
```

Algorithm 5 Proof of ownership ECDSA: proof verification Verify–Proof

```
Input: ECC base point $G$; ECC order $P$; secret key $\mathsf{sk}$; public key $\mathsf{pk}$; proof $\pi$
Output: $result$
  Parse $\pi = (\pi_n, x, y)$
  Check $\mathsf{Verify}(\pi_n) = result$
  Compute $\mathsf{sk}_v = H(x, y)$
  Compute $\mathsf{pk}_v = \mathsf{sk}_v \cdot G \bmod P$
  if $result = 1$, $\mathsf{sk}_v = \mathsf{sk}$, and $\mathsf{pk}_v = \mathsf{pk}$ then
    return 1
  else
    return 0
  end if
```

5. Security of Our Design

5.1. EUF-CMA1 Security of Signing Scheme

Firstly, similar to the proof in [35], we aim to prove that our signature scheme, whose main signature algorithms are Algorithms 1–3, is EUF-CMA1 secure. The security of the ECDSA scheme is given by [36]. Although many signature schemes are proven unforgeable under the random oracle model, our proof framework requires only the indistinguishability property. Specifically, instead of assuming that the hash function behaves precisely as a random oracle, we assume a hash function H that upholds the security property of indistinguishability defined in Definition 3 to generate the secret key.

We perform a security game to prove the EUF-CMA1 security of our scheme. The following lemma aims to show that the EUF-CMA1 security of our scheme is achieved with a suitable hash function and ECDSA.

Lemma 1. *Assume that ECDSA is EUF-CMA1 secure and the generation algorithm Gen from Algorithm 1 is constructed with a hash function H, which is indistinguishable from the uniform distribution as defined in Definition 3. Then, our scheme is EUF-CMA1 secure as given by the security game of Algorithm 6:*

Algorithm 6 EUF-CMA1 game of our scheme

```
  Procedure $\mathsf{Init}(n)$
    $\mathcal{L} \leftarrow \emptyset$
    $(\mathsf{bk}, \mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^n)$
    return $\mathsf{pk}$

  Procedure $\mathsf{Sign}(m)$
    If $m \in \mathcal{L}$: Abort
    $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$
    If $\sigma = \perp$: Return $\perp$
    $\mathcal{L} \leftarrow \mathcal{L} \cup m$
    return $\sigma$

  Procedure $\mathsf{Fin}(m^*, \sigma^*)$
    If $m^* \in \mathcal{L}$: Abort
    If $\mathsf{Verify}\ (\mathsf{pk}, m^*, \sigma^*) = 0$: Abort
    return 1
```

Note that the security game in Algorithm 6 is the One-Message Existential Unforgeability with Chosen Message Attack game, i.e., (EUF-CMA1). For the general form, i.e., standard (EUF-CMA), the sign procedure does not abort when the message is in the list $\mathcal{L}$:

- Procedure Init: the Init procedure takes a security parameter n as input, generates $(\mathsf{bk}, \mathsf{sk}, \mathsf{pk})$ as Algorithm 1 with security parameter n, generates an empty list $\mathcal{L}$ to store relevant information, and publishes the public key pk as output.
- Procedure Sign: A forger chooses a message m and sends it to the Sign procedure as input. Sign checks if the message m is already in the list $\mathcal{L}$. If the message is in the list, Sign will abort. Otherwise, Sign will sign the message using the secret key sk generated in Init, add the message m to the list $\mathcal{L}$, and send the signature σ back to the forger as output.
- Procedure Fin: The forger attempts to generate a valid message and signature pair (m^*, σ^*) and sends it to the procedure Fin as input. Fin first checks if the message m^* has already been on the list $\mathcal{L}$. If the message is in the list, Fin will abort. Otherwise, Fin verifies the message and signature pair by checking if $\mathsf{Verify}(\mathsf{pk}, m^*, \sigma^*) = 1$. If verification fails, Fin will abort the procedure. If verification is successful, Fin returns 1 as a result.

Proof. Assume the existence of a forger $\mathcal{F}$ of our signature scheme, which wins the game from Algorithm 6 by outputting a forgery (m^*, σ^*) with non-negligible probability. Then, we construct a PPT distinguisher algorithm D, which breaks the indistinguishability property of H with high probability. We construct D as follows:

- D receives the string x of Definition 3 as input. The string x is generated in one of two ways: either $x \leftarrow H(M)$, where H is a hash function applied to the arbitrarily chosen message M, or by $x \leftarrow \mathcal{U}$, where $\mathcal{U}$ represents a uniform distribution.
- In Algorithm 1, users compute the secret key using a hash function. With the input string x, D can modify the key generation algorithm by setting the secret key sk' as the string x and computing the public key as $\mathsf{pk}' = \mathsf{sk}' \cdot G \mod P$.
- Considering that the backup key bk is not used in the game procedure Sign and Fin, D can simulate the EUF-CMA1 security game in Algorithm 6 using such a pair of $(\mathsf{sk}', \mathsf{pk}')$. To simulate the procedure Init, D keeps the secret key sk' and publishes the public key pk' as output. To simulate the Sign procedure, D receives a message m from a forger and uses sk' to sign the message by $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}', m)$ and send the signature σ back to the forger as output. With non-negligible probability, the forger $\mathcal{F}$ outputs (m^*, σ^*).
- Then, D determines its output using a verification algorithm, Verify, as follows: If $\mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}') = 1$, then output 1; otherwise, output 0.

Then, the success probability of signature forgery by $\mathcal{F}$ in Algorithm 6 can be estimated with the following two considerations:

- Following the EUF-CMA1 security of ECDSA mentioned in Definition 2, we have that it is negligible for an ECDSA forger winning the EUF-CMA1 game by outputting a valid message and signature pair (m^*, σ^*) such that $\mathsf{Verify}(\mathsf{pk}', m^*, \sigma^*) = 1$. When the distinguisher D receives x generated by uniform distribution $x \leftarrow \mathcal{U}$, then D can generate an ECDSA key pair $(\mathsf{sk}', \mathsf{pk}')$ with x. Note that when D simulates the EUF-CMA1 game, the Sign and Fin procedures are the same between ECDSA and our scheme. Then, the distinguisher D can simulate an ECDSA forger playing the EUF-CMA1 game with received $x \leftarrow \mathcal{U}$. Considering that the probability of the ECDSA forger winning the EUF-CMA1 game is negligible, the probability $\Pr[x \leftarrow \mathcal{U}, \mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}') = 1]$ is negligible for the uniform random distribution $\mathcal{U}$.
- Distinguisher D determines its output using a verification algorithm Verify, which also determines whether a forger wins the simulated EUF-CMA1 security game or not. If $\mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}') = 1$, the output of D is 1, and the forger wins the simulated EUF-CMA1 security game. Then, we have that $\mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}') = 1$ and $D(x) = 1$ have the same probability with the same x received either from $x \leftarrow H(M)$ or $x \leftarrow \mathcal{U}$.

Then, the probability definition of indistinguishability $|\Pr[x \leftarrow H(M), D(x) = 1] - \Pr[x \leftarrow \mathcal{U}, D(x) = 1]|$ with the distinguisher D we constructed can be transformed as

$$\begin{aligned} &|\Pr[x \leftarrow H(M), D(x) = 1] - \Pr[x \leftarrow \mathcal{U}, D(x) = 1]| \\ =&|\Pr[x \leftarrow H(M), \mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}') = 1] - \Pr[x \leftarrow \mathcal{U}, \mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}') = 1]| \\ =&|\Pr[x \leftarrow H(M), \mathsf{Verify}(m^*, \sigma^* \mathsf{pk}') = 1] - \mathsf{negl}(n)|. \end{aligned}$$

If there exists a forger $\mathcal{F}$ of our signature scheme, it wins the EUF-CMA1 game from Algorithm 6 by outputting a forgery (m^*, σ^*) with non-negligible probability. Then, we have that $\Pr[x \leftarrow H(M), \mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}') = 1]$ is non-negligible. Therefore, $|\Pr[x \leftarrow H(M), \mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}') = 1] - \mathsf{negl}(n)|$ is non-negligible, and $|\Pr[x \leftarrow H(M), D(x) = 1] - \Pr[x \leftarrow \mathcal{U}, D(x) = 1]|$ is non-negligible, which is contrary to the assumption that the hash function in our scheme holds indistinguishability. □

5.2. Security of Proof Scheme

In the proof generation function, recall that the input is (bk, x) from the key generation phase. Compute $\mathsf{AES}_{\mathsf{bk}}(x) = y$, then generate the NIZKPoK proof π_n and output $\pi = (x, y, \pi_n)$ (Algorithms 1, 4 and 5). The soundness of the NIZKPoK proof system relies on the Fiat–Shamir transform [20,37] and obtains a better concrete-security reduction than [38]. To ensure that no information is revealed in the NIZKPoK computation, the backup key bk is selected such that for the AES computation $y = \mathsf{AES}_{\mathsf{bk}}(x)$, there are no S-boxes in the computation of y with input 0, as analyzed in [19]. The analysis also shows that $|\mathcal{K}'|/|\mathcal{K}| \approx 1/3$ under the assumption that S-box inputs are independent. An experiment confirms that the bit security of the inverting one-way function is reduced by about $\log_2 3 \approx 1.4$ bits for the AES-256 function. We argue that choosing bk from subset $\mathcal{K}'$ fits our statement that all n-bits keys decrease the security level of our scheme within a negligible amount. We prove the security of proof generation in our scheme under the security properties that AES is a one-way function with a security game in Algorithm 7. If AES is a one-way function, for all polynomial time adversaries A

$$\Pr[x \leftarrow \{0,1\}^n, k \leftarrow \mathcal{K}, y = \mathsf{AES}_k(x), k' = A(x, y) : y = \mathsf{AES}_{k'}(x)] \le \mathsf{negl}(n) \tag{5}$$

Algorithm 7 One-way function game

```
Challenger sample a random value $x$ and a key $k$
Challenger compute $y = f_k(x)$
Challenger send $(x, y)$ to the an attacker $A$
Attacker invert a key $k' = A(x, y)$
if $y = \mathsf{AES}_{k'}(x)$ then
    return 1
end if
return 0
```

In our scheme, we use the key set $\mathcal{K}'$, and we assume that adversary A has a high success probability p of generating a key k' and winning the security game in Algorithm 7. We can construct a one-way function attack algorithm $B(x, y)$ that returns $A(x, y)$. We assume that when k has no zero S-box input, A is guaranteed to succeed, and when k has one or more zero S-box inputs, A fails. Then, when $k \in \mathcal{K}'$, A succeeds with probability p. For our construction, we invoke attack algorithm B, and then B succeeds with probability $p \cdot |\mathcal{K}'|/|\mathcal{K}|$. Notice that p is the success probability of the adversary A, and it is assumed to be non-negligible. If $|\mathcal{K}'|/|\mathcal{K}|$ is non-negligible, the success probability of the one-way function attack B is non-negligible for a PPT adversary A. Therefore, our analysis relies on the non-negligible $|\mathcal{K}'|/|\mathcal{K}|$. Considering that $|\mathcal{K}'|/|\mathcal{K}|$ stands for the probability of S-box input being 0 for an AES function and it is non-negligible, we claim that if the AES function is a one-way function, then the AES function used in our scheme is also a one-way function, and for all polynomial time adversaries A:

$$\Pr[x \leftarrow \{0,1\}^n, k \leftarrow \mathcal{K}', y = \mathsf{AES}_k(x), k' = A(x,y) : y = \mathsf{AES}_{k'}(x)] \leq \mathsf{negl}(n). \tag{6}$$

5.3. Strong Unforgeability

In our design, we have analyzed the signing scheme using the EUF-CMA1 definition and, therefore, do not rule out the case that an attacker can find a new signature for a previously signed message. For instance, an attacker may manipulate a signature generated by the signer and, using some method, pass the verification with the same message. For strong unforgeability, we modify our signing scheme by combining Algorithms 2 and 4, as well as Algorithms 3 and 5. Then, we have that even if the ECDSA signature generated by Algorithm 3 is manipulated and passes the pre-verification of the ECDSA scheme in Algorithm 5, an attacker can not pass the verification because the verification of NIZKPoK is required. Regarding the NIZKPoK scheme, similar to the Banquet scheme, it requires the following:

- It is difficult to find a set of intermediate seeds in the tree that derive the same leaf seeds used in a signature, which means that more security concerns should be needed in seed sampling.
- It is difficult to find a different seed that produces the same outputs when input to the Expand function utilized as a seed generation function. In our scheme, Expand is instantiated by hash functions, and we assume that the hash function is a random oracle to meet the requirement, like the assumption in Definition 3.

With the above assumptions, our scheme is supposed to have strong unforgeability.

5.4. Post-Quantum Security

Our scheme aims to protect classical digital signature schemes against key-revealing attacks while ensuring post-quantum security. This requires that our proof of ownership scheme retain its security assurances even in the advent of quantum computing technologies. In the proof phase, as detailed in Algorithm 4, a NIZKPoK for the backup key bk is generated, and the secret key sk is computed with a hash function H. The security levels for the NIZKPoK, categorized as L1, L3, and L5 from the standards defined by NIST [39], correspond to AES key sizes of 128, 192, and 256 bits, respectively. In our scheme, we utilize AES-256 to ensure that the proof of ownership of our scheme maintains a high-security level under post-quantum attack (equivalent to the security of AES-128 in classical computing).

To demonstrate that our scheme is post-quantum secure, derived from the NIZKPoK, the security in the Quantum Random Oracle Model (QROM) should naturally be considered. The QROM, a more robust security version of the Random Oracle Model (ROM), allows attackers to make superposition queries to the random oracle, thus modeling attacks at the post-quantum level. Following [21], Corollary 13, we provide a reduction for the Fiat–Shamir transformation by simulating a prover in the underlying ZKIP as Definition 4 to prove the security in QROM of NIZKPoK in our scheme. The following lemma demonstrates the reduction.

Lemma 2. *There exists a black-box quantum polynomial time four-stage quantum algorithm $\mathcal{S}$ such that for any adaptive adversary $\mathcal{A}$ against our multi-round Fiat–Shamir transformed NIZKPoK of underlying seven-move ZKIP making q queries to a uniformly random function H with appropriate domain and range equal to C, and for any $x^\circ \in \mathcal{X}$:*

$$\begin{aligned} &\Pr[x = x^\circ \wedge v = accept : (x,v) \leftarrow \langle \mathcal{S}^{\mathcal{A}}, \mathcal{V} \rangle] \geq \\ &\frac{3!}{(2q+4)^6} \Pr_H\Big[x = x^\circ \wedge V^H_{\mathsf{FS}}(x,\pi) = 1 : (x,\pi) \leftarrow \mathcal{A}^H\Big] - \epsilon_{x^\circ} \end{aligned} \tag{7}$$

where the additive error term ϵ_{x° is equal to $\frac{3!}{|\mathsf{C}|}$ when summed over all x°.

Remark 1. *x° holds that $(x^\circ, w^\circ) \in R$ for the relation R of our NIZKPoK and underlying ZKIP. $\mathcal{S}^{\mathcal{A}}$ simulates a prover in the underlying ZKIP system, and $\mathcal{V}$ is a verifier outputting the verification result v. In our NIZKPoK, V_{FS}^{H} is the verification predicate, and π is composed of $(a_1, c_1, \ldots, a_n, c_n, z)$ according to Definition 5.*

Proof. Compared with the Banquet, our NIZKPoK removes the message m as hash input during the computation of the first challenge. Accordingly, the Fiat–Shamir transformation of our NIZKPoK, as defined in Definition 5, aligns with the Fiat–Shamir transformation for general PCIP (mFS). To generate the reduction in mFS, the challenges of Fiat–Shamir transformation are required to use distinct inputs to H with probability 1 for the application of the simulator from [21], Theorem 7. In our case, this distinctness is ensured by the input of commitment. Similarly to the construction in [22], the commitments of our NIZKPoK are generated separately and have different sizes, ensuring distinct inputs for different challenge computations. □

Using the reduction from Lemma 2, we can prove that our NIZKPoK system holds soundness in QROM with the following lemma.

Lemma 3. *Assume that the underlying ZKIP system has (statistical or computational) soundness. Then, in QROM, our NIZKPoK system also has (statistical or computational) soundness.*

Proof. From Lemma 2, given that q is polynomial in the security parameter, the success probability of any dishonest prover $A_{NIZKPoK}$ against our NIZKPoK and the success probability of a dishonest prover A_{ZKIP} challenging the underlying ZKIP are polynomially related. If our NIZKPoK does not have soundness in QROM, the success probability of $A_{NIZKPoK}$ is non-negligible. Then, the polynomially related success probability of A_{ZKIP} is also non-negligible, which is contrary to the assumption that the underlying ZKIP system has (statistical or computational) soundness. □

Considering that the underlying ZKIP system of our NIZKPoK is identical to the underlying zero-knowledge identification scheme [22] of the Banquet scheme, we can conclude that the underlying ZKIP system holds soundness if the underlying zero-knowledge identification scheme holds. Given that the soundness of the zero-knowledge identification scheme has been proved in [22] using the Schwarz–Zippel Lemma, using Lemma 3, we demonstrate that our NIZKPoK system has soundness in QROM.

5.5. Security of Proof of Ownership

Now we argue the security of proof of ownership in our design (Gen, Sign, Verify, Proof, and $\mathsf{Verify\text{–}Proof}$). We claim that our scheme generates a single proof of ownership π as per Definition 6. In Section 5.2, we have proved that the parameter-limited AES function we use in our scheme can also be considered a one-way function, and the NIZKPoK used in our scheme is sound. Furthermore, in Section 5.4, we have shown that the NIZKPoK in our scheme also holds soundness in QROM. Therefore, a proof π generated by our scheme satisfies Definition 6.

6. Extensions

Besides the proof of ownership of the digital signature scheme, we also provide a post-quantum fallback signature scheme when adversaries with quantum computing abilities attack the secret key of the classical digital signature scheme.

Fallback Signature Scheme

The fallback signature scheme is generated from the Banquet signature. In this case, the backup key bk is converted to a secret key, and the value (x, y) generated in Algorithm 1 serves as the public key of the fallback signature scheme. In our design, the fallback digital signature scheme, which produces post-quantum security guarantees, enables a digital wallet user to sign secret values even under a secret key revealing attack. Compared with the Banquet scheme, our scheme modifies the key generation phase with the input (x, k) generated from Algorithm 1, which satisfies the security requirement. We briefly describe it with three algorithms: $\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}$:

- Gen_f: given (bk, x, y) from Algorithm 1 and as input, set secret key $\mathsf{sk} \leftarrow \mathsf{bk}$, public key $\mathsf{pk} \leftarrow (x, y)$, output sk, pk.
- $\mathsf{Sign}_f(\mathsf{sk}, m)$: given a message m and secret key sk as input, compute $\sigma \leftarrow \mathsf{Prove}(\mathsf{sk}, m)$, output σ.
- $\mathsf{Verify}_f(\mathsf{pk}, m, \sigma)$: given signature σ and public key pk as input, compute a verification algorithm $\mathsf{Verify}(\mathsf{pk}, \sigma, m)$, that outputs 1 if the result of Verify is 1; otherwise, output 0.

The security of our fallback scheme derives from the Banquet signature scheme of Baum et al. [22]. They have proven that assuming that the functions used in the Signature Generation phase $\mathsf{Commit}, H_1, H_2$, and H_3 are modeled as random oracles, Expand is a pseudorandom generator (PRG) with output computationally ε_{PRG}-*close* to uniform; the seed tree construction is computationally hiding, the execution and security parameters are appropriately chosen, and the key generation function $f_x : \mathsf{sk} \mapsto \mathsf{pk}$ is a one-way function; the Banquet signature is EUF-CMA secure. Sign_f and Verify_f in our fallback scheme are the same as Sign_b and Verify_b, and Gen_f receives bk, x, y from Algorithm 1, satisfying that $f_x : \mathsf{bk} \mapsto (x, y)$ is a one-way function. Then, our fallback scheme is also EUF-CMA secure, like the Banquet signature. To prove the EUF-CMA security of our fallback scheme in QROM, the most promising approach seems to be the general result for multi-round Fiat–Shamir type signatures in [21,40]. However, in order to apply the QROM EUF-CMA result [21], Theorem 23, it would require a σ-protocol formulated by our fallback scheme, which is beyond our scope.

7. Evaluation

Our scheme and other state-of-the-art works only modify the key generation phase of the ECDSA scheme, keeping the signing and verification phases unchanged. Since the key generation phase is executed in real life, the runtime of the modified key generation algorithm should be as close as possible to the key generation time of ECDSA while simultaneously ensuring that both the signing and verification times of ECDSA remain unchanged.

To validate our results, we implemented the key generation phase of our scheme, S_{leeve}, and [33] using Python, given that our and related works do not modify the signing and verification phases of the ECDSA scheme. In our implementation, we follow the key generation of the AES function in Banquet [22], ensuring that there is non-zero input for the S-box computation, which slows down the key generation phase. For the hash function, we use SHA-256 and ensure that the output fits well with the ECDSA curve. The ECDSA curve we chose for our scheme is secp256r1 [5]. We first compare the operations of the key generation phases from different schemes and observe that our scheme requires additional AES operations, while the S_{leeve} scheme involves 1329 hash operations, which significantly slow down its key generation phase. While random value generation and index conversion are not included in our comparison of key generation operations, they do affect the key generation time in practical implementations of all schemes, especially in the S_{leeve} scheme, which involves more rounds of random value generation and index conversion. Then, we run our experiment on a 3.4 GHz Intel Core i7-3770 CPU with 16 GB of RAM, running 64-bit Windows 10 22H2. The performance of our scheme, demonstrated in Table 1, indicates that our key generation time is approximately 30% slower than the ECDSA scheme. Considering that our scheme enables a user to generate secret keys with one fixed backup key. Then, with a chosen backup key, our scheme needs only one AES operation, reducing the key generation time to within 10% slower than ECDSA, which is comparable with the scheme of Tan and Zhou [33]. Additionally, we implemented the S_{leeve} scheme, reporting that the key generation time is 5 to 10 times slower than that of a normal

ECDSA key generation algorithm, which is significantly slower than our scheme and the scheme of Tan and Zhou.

Table 1. Key generation phase.

Methods	Operations	Time (ms)
Our Scheme	$exp + 3AES + hash$	1.09
Our Scheme (with fixed bk)	$exp + AES + hash$	0.97
Tan and Zhou scheme	$exp + hash$	0.91
S_{leeve}	$exp + 1329hash$	4.48
ECDSA	exp	0.84

Let exp represent a Elliptic Curve Scalar Multiplication (ECSM) operation of secp256r1, let AES represent a AES function operation of AES-128, and $hash$ represent a hash function operation of SHA-256.

We also evaluate the proof scheme of our scheme, S_{leeve}, and the Tan and Zhou scheme [33] with a theoretical analysis in Table 2. Our analysis reveals that our scheme significantly reduces the proof size by half and approximately doubles the computational efficiency compared with the scheme presented in [33]. Given that t_H, t_E, and t_A constitute only a minor portion of the overall computational costs, the computational cost and the proof size of all schemes are primarily determined by the methods employed in each scheme. Thus, the comparative analysis between our scheme and S_{leeve} can be effectively reduced to a comparison of their respective methodologies. In our scheme, the proof generation and verification times are similar to the signing and verification times of the Banquet signature scheme. According to [22], Banquet has a signing time of 6.36 ms and a verification time of 4.68 ms for the L1 security level. In contrast, the eWOTS$^+$ signature scheme, analyzed in [11], has signing and verification times of 0.024 ms and 1.472 ms, respectively. Given that eWOTS$^+$ is a one-time signature and S_{leeve} is an interactive proof scheme, our scheme is more efficient in the scenario of a large-scale decentralized system like Bitcoin but less efficient in a centralized system like the Central Bank Digital Currency (CBDC) [41].

Table 2. Computation time and proof size comparison of the proof schemes.

Schemes	Method	Proof Size	Proof Generation Time	Verification Time
S_{leeve}	eWOTS$^+$	S_W	t_{PW}	$t_{VM} + t_E$
Tan and Zhou scheme	NIZK	$2S_N$	$t_{PN} + t_H$	$2t_{VN} + 2t_H + t_E$
Our scheme	NIZK	$S_N + 2n$	$t_{PN} + t_A$	$t_{VN} + t_H + t_E$

Let S_W and S_N represent the signature size of eWOTS$^+$ and proof size of NIZK in our scheme. Let t_{PW} and t_{VM} represent the signing and verification time of the eWOTS$^+$ signature; t_{PN} and t_{VN} represent the proof generation and verification time of the NIZK used in our scheme; let t_H, t_E, and t_A represent the computation time of hash function, elliptic curve, and AES encryption. We compare the proof size and the computational time of three proof schemes.

In the S_{leeve} and Tan and Zhou scheme, each secret key requires a corresponding backup key or pre-image during the key generation phase. In contrast, our scheme allows a single backup key mapping to multiple secret keys, which enables a fixed backup key in key generation processes and requires only a single query of the backup key to generate all required proofs. Specifically, Algorithm 1 demonstrates that with a single bk as input, multiple valid public and secret key pairs can be generated by varying randomly chosen parameter x. In the context of a digital wallet, especially in the scenario of an HD wallet [14], this means that our scheme can store only one backup key, thus saving the amount of storage space and providing a simpler key management system compared with other schemes.

8. Applications

In our design, we address the issue of secret key revelation in digital signature schemes and propose a method to prove the ownership of digital funds. Our approach allows digital wallet users to generate a statement of ownership of the secret key using a single backup key. This can solve security problems such as hard forks and revocation of wallet addresses. Compared with the proof of ownership scheme S_{leeve}, our design provides non-trivial long-term data protection and data migration capabilities. This ensures enhanced security and resilience in maintaining the integrity and ownership of digital assets over time. Furthermore, our scheme prevents backup key leakage when a user needs to prove the ownership of multiple secret keys.

Digital wallets: If there is evidence of a quantum attack against a digital wallet environment, such as a blockchain system, users can utilize our construction to prove the ownership of potentially compromised tokens by generating and publishing a NIZKPoK for the secret key without any challenge from other verifiers and redeem tokens in an alternative manner while the token is not locked up. Our scheme can also be utilized when a digital wallet user suffers a quantum attack and needs to revoke the ownership of an address the adversary has stolen. Suppose a user's ECDSA key is stolen; in that case, they can revoke the wallet address by creating proof of ownership with a backup key, informing the network, and non-interactively being verified by other users. The user can then introduce a new wallet address generated with the same backup key containing the funds associated with the old wallet. We also address scenarios where a user manages multiple secret keys, such as in coin mixing applications [13]. Additional security considerations of the backup key storage from our scheme offer a more secure solution compared with existing state-of-the-art approaches.

Beyond digital wallets: Our scheme can also be applied to sign long-term data requiring non-repudiation protection, such as providing extra security for quantum-resistant, time-stamped signed PDFs in Adobe Acrobat Reader DC. Our scheme can also facilitate efficient encryption scheme migration for classical digital signature schemes. Given that our proof of ownership digital signature scheme and fallback scheme can generate two signatures with the same secret value, input sampling, and message, the migration process only modifies the verification phase. It retains the old encryption without additional inputs. This migration can be carried out in three steps, as defined below:

1. The signing parties sign a document using Algorithm 2, similar to a classical ECDSA scheme, noticing that the document is vulnerable in a post-quantum environment. The verifying parties receive and verify the signed document using Algorithm 3, ensuring that the ECDSA signature scheme correctly signs the document. They then inform the signing parties to initiate the migration process.
2. The signing parties generate a proof of ownership for the secret key using Algorithm 4 and then publish this proof. Verifying parties first ignore the published new signatures and continue verifying old ones. Once the first verification is complete and the proof of ownership is received from the signing parties, the verifying parties check the computation correctness of the ECDSA signature key and verify the proof of knowledge of the secret key generation using Algorithm 5.
3. The signing parties migrate the signing strategy by setting the inputted secret key as the backup key generated in Algorithm 1 and signing the same document signed in step 1. After receiving the new signatures, the verifying parties update their verification strategy to verify the new ones while maintaining a concrete relationship with the old ones.

9. Conclusions

This paper introduced a new proof of ownership digital signature scheme incorporating a quantum-secure fallback within an ECDSA digital signature framework. The central concept involves embedding a backup key within the ECDSA secret key and generating a zero-knowledge proof for the backup key. We formally proved that our new signature scheme achieves unforgeability under the computational indistinguishability from the Uniformly Random Distribution property of a proper hash function and that the proof system is sound under QROM. Furthermore, our extension scheme enables users to generate post-quantum fallback signatures. Our non-interactive approach is suitable for longer-term digital data protection, such as PDF files for important financial contacts and valuable digital arts in blockchain systems. The simplified key management system, short key generation time, and efficient proof scheme make our approach well suited for scenarios in which a user manages multiple secret keys, such as in hierarchical deterministic (HD) wallets.

Our future work is as follows:

- A new zero-knowledge proof scheme: The size and computational time of the proof scheme are linked to the circuit computation in our design. VOLE-in-the-head presents a promising potential solution [42].
- QROM security: In this paper, we proved the soundness of our proof scheme under QROM. However, our work does not provide a concrete proof of the fallback (Banquet signature) scheme under QROM. The QROM security of the Banquet signature could potentially be proven if we can modify the NIZKPoK in the Banquet signature into a $\sum$-protocol [22]. Another possible method to prove the QROM secure Banquet signature is to follow the proof framework in [43], where they provide a QROM security proof for the Helium proof system [44] with online extractability [45].
- Post-quantum signatures: Our signature is designed to hide any post-quantum signature inside a classical digital signature scheme. So, in addition to NIZK-based signatures, we will implement other types of post-quantum signature schemes, such as lattice-based signatures.

Author Contributions: Conceptualization, C.W., Z.-Y.L. and M.M.; methodology, C.W. and Z.-Y.L.; software, C.W.; validation, C.W., Z.-Y.L. and M.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by JSPS KAKENHI Grant Number JP23K28070 (formerly, JP23H03380).

Data Availability Statement: The data presented in this study are openly available in https://github.com/wangbabb/poo (accessed on 13 December 2024).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Arapinis, M.; Gkaniatsou, A.; Karakostas, D.; Kiayias, A. A formal treatment of hardware wallets. In Proceedings of the Financial Cryptography and Data Security: 23rd International Conference, FC 2019, Frigate Bay, St. Kitts and Nevis, 18–22 February 2019; Revised Selected Papers 23; Springer: Berlin/Heidelberg, Germany, 2019; pp. 426–445.
2. Das, P.; Faust, S.; Loss, J. A formal treatment of deterministic wallets. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, New York, NY, USA, 31 October 2019; pp. 651–668.
3. Hellman, M. New directions in cryptography. *IEEE Trans. Inf. Theory* **1976**, *22*, 644–654.
4. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
5. Brown, D.R. *Sec 2: Recommended Elliptic Curve Domain Parameters*; Standards for Efficient Cryptography Group: Mississauga, ON, Canada, 2010.
6. Shor, P.W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.* **1999**, *41*, 303–332. [CrossRef]
7. Li, C.; Dong, M.; Li, J.; Xu, G.; Chen, X.B.; Liu, W.; Ota, K. Efficient medical big data management with keyword-searchable encryption in healthchain. *IEEE Syst. J.* **2022**, *16*, 5521–5532. [CrossRef]
8. Li, C.; Jiang, B.; Dong, M.; Chen, Y.; Zhang, Z.; Xin, X.; Ota, K. Efficient Designated Verifier Signature for Secure Cross-Chain Health Data Sharing in BIoMT. *IEEE Internet Things J.* **2024**, *11*, 19838–19851. [CrossRef]
9. Boschini, C.; Dahari, H.; Naor, M.; Ronen, E. That's not my signature! Fail-stop signatures for a post-quantum world. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2024; Springer: Berlin/Heidelberg, Germany, 2024; pp. 107–140.
10. Rackoff, C.; Simon, D.R. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 11–15 August 1991; Springer: Berlin/Heidelberg, Germany, 1991; pp. 433–444.

11. Chaum, D.; Larangeira, M.; Yaksetig, M.; Carter, W. W-OTS+ up my sleeve! A hidden secure fallback for cryptocurrency wallets. In Proceedings of the International Conference on Applied Cryptography and Network Security, Virtual, 21–24 June 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 195–219.
12. Standaert, F.X. Introduction to side-channel attacks. In *Secure Integrated Circuits and Systems*; Springer: Boston, MA, USA, 2010; pp. 27–42.
13. Ruffing, T.; Moreno-Sanchez, P.; Kate, A. Coinshuffle: Practical decentralized coin mixing for bitcoin. In Proceedings of the Computer Security-ESORICS 2014: 19th European Symposium on Research in Computer Security, Wroclaw, Poland, 7–11 September 2014; Proceedings, Part II 19; Springer: Berlin/Heidelberg, Germany, 2014; pp. 345–364.
14. Gutoski, G.; Stebila, D. Hierarchical deterministic bitcoin wallets that tolerate key leakage. In Proceedings of the International Conference on Financial Cryptography and Data Security, San Juan, Puerto Rico, 30 January 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 497–504.
15. Fiat, A.; Shamir, A. How to prove yourself: Practical solutions to identification and signature problems. In Proceedings of the Conference on the Theory and Application of Cryptographic Techniques, Linz, Austria, 1 January 1986; Springer: Berlin/Heidelberg, Germany, 1986; pp. 186–194.
16. Ishai, Y.; Kushilevitz, E.; Ostrovsky, R.; Sahai, A. Zero-knowledge from secure multiparty computation. In Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing, San Diego, CA, USA, 11–13 June 2007; pp. 21–30.
17. Giacomelli, I.; Madsen, J.; Orlandi, C. {ZKBoo}: Faster {Zero-Knowledge} for Boolean Circuits. In Proceedings of the 25th USENIX Security Symposium (USENIX Security 16), Austin, TX, USA, 10–12 August 2016; pp. 1069–1083.
18. Chase, M.; Derler, D.; Goldfeder, S.; Katz, J.; Kolesnikov, V.; Orlandi, C.; Ramacher, S.; Rechberger, C.; Slamanig, D.; Wang, X.; et al. The Picnic Signature Scheme. Submission to NIST Post-Quantum Cryptography Project. 2020. Available online: https://raw.githubusercontent.com/microsoft/Picnic/master/spec/design-v2.2.pdf (accessed on 13 December 2024).
19. de Saint Guilhem, C.D.; De Meyer, L.; Orsini, E.; Smart, N.P. BBQ: Using AES in picnic signatures. In Proceedings of the International Conference on Selected Areas in Cryptography, Waterloo, ON, Canada, 14–16 August 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 669–692.
20. Liu, Q.; Zhandry, M. Revisiting post-quantum fiat-shamir. In Proceedings of the Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2019; Proceedings, Part II 39; Springer: Berlin/Heidelberg, Germany, 2019; pp. 326–355.
21. Don, J.; Fehr, S.; Majenz, C. The measure-and-reprogram technique 2.0: Multi-round fiat-shamir and more. In Proceedings of the Annual International Cryptology Conference, Virtual, 17–21 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 602–631.
22. Baum, C.; de Saint Guilhem, C.D.; Kales, D.; Orsini, E.; Scholl, P.; Zaverucha, G. Banquet: Short and fast signatures from AES. In Proceedings of the IACR International Conference on Public-Key Cryptography, Virtual, 10–13 May 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 266–297.
23. Albrecht, M.R.; Rechberger, C.; Schneider, T.; Tiessen, T.; Zohner, M. Ciphers for MPC and FHE. In Proceedings of the Advances in Cryptology–EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 April 2015; Proceedings, Part I 34; Springer: Berlin/Heidelberg, Germany, 2015; pp. 430–454.
24. Banik, S.; Barooti, K.; Vaudenay, S.; Durak, F.B. Cryptanalysis of LowMC instances using single plaintext/ciphertext pair. *IACR Trans. Symmetric Cryptol.* **2020**, *2020*, 130–146. [CrossRef]
25. Banik, S.; Barooti, K.; Vaudenay, S.; Yan, H. New attacks on LowMC instances with a single plaintext/ciphertext pair. In Proceedings of the Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, 6–10 December 2021; Proceedings, Part I 27; Springer: Berlin/Heidelberg, Germany, 2021; pp. 303–331.
26. Dobraunig, C.; Kales, D.; Rechberger, C.; Schofnegger, M.; Zaverucha, G. Shorter signatures based on tailor-made minimalist symmetric-key crypto. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, Los Angeles, CA, USA, 7–11 November 2022; pp. 843–857.
27. Kim, S.; Ha, J.; Son, M.; Lee, B.; Moon, D.; Lee, J.; Lee, S.; Kwon, J.; Cho, J.; Yoon, H.; et al. AIM: Symmetric primitive for shorter signatures with stronger security. In Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, Copenhagen, Denmark, 26–30 November 2023; pp. 401–415.
28. Liu, F.; Mahzoun, M.; Øygarden, M.; Meier, W. Algebraic attacks on RAIN and AIM using equivalent representations. *Cryptol. Eprint Arch.* **2023**. Available online: https://eprint.iacr.org/2023/1133 (accessed on 13 December 2024).
29. Zhang, K.; Wang, Q.; Yu, Y.; Guo, C.; Cui, H. Algebraic Attacks on Round-Reduced Rain and Full AIM-III. In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, Guangzhou, China, 4–8 December 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 285–310.
30. Raavi, M.; Wuthier, S.; Chandramouli, P.; Balytskyi, Y.; Zhou, X.; Chang, S.Y. Security comparisons and performance analyses of post-quantum signature algorithms. In Proceedings of the International Conference on Applied Cryptography and Network Security, Virtual, 21–24 June 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 424–447.
31. Sikeridis, D.; Kampanakis, P.; Devetsikiotis, M. Post-quantum authentication in TLS 1.3: A performance study. *Cryptol. Eprint Arch.* **2020**. Available online: https://eprint.iacr.org/2020/071 (accessed on 13 December 2024).

32. Au, M.H.; Susilo, W.; Mu, Y. Proof-of-knowledge of representation of committed value and its applications. In Proceedings of the Australasian Conference on Information Security and Privacy, Sydney, Australia, 5–7 July 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 352–369.
33. Tan, T.G.; Zhou, J. Layering quantum-resistance into classical digital signature algorithms. In Proceedings of the Information Security: 24th International Conference, ISC 2021, Virtual Event, 10–12 November 2021; Proceedings 24; Springer: Berlin/Heidelberg, Germany, 2021; pp. 26–41.
34. Chase, M.; Derler, D.; Goldfeder, S.; Orlandi, C.; Ramacher, S.; Rechberger, C.; Slamanig, D.; Zaverucha, G. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In Proceedings of the 2017 ACM Sigsac Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 1825–1842.
35. Chaum, D.; Larangeira, M.; Yaksetig, M. Tweakable Sleeve: A Novel Sleeve Construction Based on Tweakable Hash Functions. In Proceedings of the the International Conference on Mathematical Research for Blockchain Economy, Vilamoura, Portugal, 12–14 July 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 169–186.
36. Fersch, M.; Kiltz, E.; Poettering, B. On the one-per-message unforgeability of (EC) DSA and its variants. In Proceedings of the Theory of Cryptography: 15th International Conference, TCC 2017, Baltimore, MD, USA, 12–15 November 2017; Proceedings, Part II 15; Springer: Berlin/Heidelberg, Germany, 2017; pp. 519–534.
37. Pointcheval, D.; Stern, J. Security arguments for digital signatures and blind signatures. *J. Cryptol.* **2000**, *13*, 361–396. [CrossRef]
38. Katz, J.; Kolesnikov, V.; Wang, X. Improved non-interactive zero knowledge with applications to post-quantum signatures. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, New York, NY, USA, 3 October 2018; pp. 525–537.
39. Alagic, G.; Alagic, G.; Apon, D.; Cooper, D.; Dang, Q.; Dang, T.; Kelsey, J.; Lichtinger, J.; Liu, Y.K.; Miller, C.; et al. Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process. 2022. Available online: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=934458 (accessed on 13 December 2024).
40. Don, J.; Fehr, S.; Majenz, C.; Schaffner, C. Security of the Fiat-Shamir transformation in the quantum random-oracle model. In Proceedings of the Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2019; Proceedings, Part II 39; Springer: Berlin/Heidelberg, Germany, 2019; pp. 356–383.
41. Bindseil, U. Tiered CBDC and the financial system. *Available Ssrn 3513422*. 2020. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3513422 (accessed on 16 December 2024)
42. Baum, C.; Braun, L.; de Saint Guilhem, C.D.; Klooß, M.; Orsini, E.; Roy, L.; Scholl, P. Publicly verifiable zero-knowledge and post-quantum signatures from vole-in-the-head. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–24 August 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 581–615.
43. Balumuri, S.; Eaton, E.; Lamontagne, P. Quantum-Safe Public Key Blinding from MPC-in-the-Head Signature Schemes. *Cryptol. Eprint Arch.* **2024**. Available online: https://eprint.iacr.org/2024/945 (accessed on 13 December 2024).
44. Kales, D.; Zaverucha, G. Efficient lifting for shorter zero-knowledge proofs and post-quantum signatures. *Cryptol. Eprint Arch.* **2022**. Available online: https://eprint.iacr.org/2022/588 (accessed on 13 December 2024).
45. Don, J.; Fehr, S.; Majenz, C.; Schaffner, C. Online-extractability in the quantum random-oracle model. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, 30 May–3 June 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 677–706.