Post Quantum Fuzzy Stealth Signatures and Applications

Sihang Pu[1], Sri AravindaKrishnan Thyagarajan[2], Nico Döttling[1], and Lucjan Hanzlik[1]

[1]CISPA Helmholtz Center for Information Security
[2]NTT Research

July 25, 2023

Abstract

Private payments in blockchain-based cryptocurrencies have been a topic of research, both academic and industrial, ever since the advent of Bitcoin. Stealth address payments were proposed as a solution to improve payment privacy for users and are, in fact, deployed in several major cryptocurrencies today. The mechanism lets users receive payments so that none of these payments are linkable to each other or the recipient. Currently known stealth address mechanisms either (1) are insecure in certain reasonable adversarial models, (2) are inefficient in practice or (3) are incompatible with many existing currencies.

In this work, we formalize the underlying cryptographic abstraction of this mechanism, namely, *stealth signatures* with formal game-based definitions. We show a surprising application of our notions to passwordless authentication defined in the Fast IDentity Online (FIDO) standard. We then present SPIRIT, the first efficient post-quantum secure stealth signature construction based on the NIST standardized signature and key-encapsulation schemes, Dilithium and Kyber. The basic form of SPIRIT is only secure in a *weak* security model, but we provide an efficiency-preserving and generic transform, which *boosts* the security of SPIRIT to guarantee the strongest security notion defined in this work. Compared to state-of-the-art, there is an approximately 800x improvement on the signature size while keeping signing and verification as efficient as 0.2 ms.

We extend SPIRIT with a *fuzzy tracking* functionality where recipients can outsource the tracking of incoming transactions to a tracking server, satisfying an anonymity notion similar to that of fuzzy message detection (FMD) recently introduced in [CCS 2021]. We also extend SPIRIT with a new fuzzy tracking framework called *scalable fuzzy tracking* that we introduce in this work. This new framework can be considered as a dual of FMD, in that it reduces the tracking server's computational workload to *sublinear* in the number of users, as opposed to linear in FMD. Experimental results show that, for millions of users, the server only needs 3.4 ms to filter each incoming message which is a significant improvement upon the state-of-the-art.

Contents

1 Introduction

Cryptocurrencies provide support for trustless and publicly verifiable payments. The sender of a payment posts a transaction onto a public ledger called *blockchain*. In the most basic form, the transaction specifies the sender and receiver's respective public keys (or addresses), and the transaction is authorized by the sender via a digital signature wrt. their public key. E-commerce [eco], donation platforms [donc, dona, donb], gaming platforms [ega], etc., are just some of the popular use cases that are enabled by cryptocurrencies and their trustless payments. For example, donation platforms accept donations in the form of cryptocurrency payments, and to do this, a donation platform announces its addresses and users can make transactions paying to these addresses without requiring permission from any authority.

A critical weakness of the above paradigm is that it lacks reliable anonymity guarantees in its basic form. Several de-anonymisation techniques [OKH13, SO13, RH13, RS13, MSH+17] for law enforcement purposes have been demonstrated that link addresses on the blockchain to the real-world entities that own them. However, it has also led to questionable forms of censorship [fre] of users and their payments.

A mechanism known as *stealth addresses* [ste, vS, Tod, CM17] was developed to address these anonymity issues. For instance, the donation platform publishes a single master address, a so-called *stealth address*, and any user can send donations to the platform, by using a *locally re-randomized* version of the stealth address called one-time address. Such a one-time address is *unlinkable* to the stealth address for any outside observer, consequently, transactions to such a stealth address look as if they are going to random recipients (and not necessarily the donation platform). Also, with access to its master secret, the donation platform can link such a one-time address to its stealth address and further generate the corresponding one-time secret locally, on the fly. Using this one-time secret, the coins associated with the one-time address can be spent. In this case, the recipient only needs to publish its master address, and *does not* need to give out fresh unlinkable addresses for each potential sender. As the number of senders could well be in the hundreds or thousands (as is the case with e-commerce, donations, etc.), this mechanism leads to a *scalable solution.*

The stealth address scheme proposed in [vS] has in fact been deployed in many of the major currencies like Bitcoin [ste], Ethereum [umb], and Monero [vS]. The mechanism has further found direct application in privacy enhancement of payment protocols like Blitz [AMKM21]. As Monero implements stealth addresses via signature schemes, we will refer to the cryptographic abstraction of the mechanism from [vS] as *stealth signatures.* Thus we will henceforth use the terms addresses and public keys interchangeably.

Recent academic works [LYW+19, LLN+20] initiated the formal treatment of stealth signatures and observed that the construction of [vS] does not satisfy security under so-called *key-exposures.* Roughly, this means that if an adversary learns the corresponding one-time secret key for the one-time public keys that he generated, then he can learn all one-time secret keys of all one-time public keys that he generates for this particular master address.

More recent proposals of stealth signature schemes [LYW+19, LLN+20] were designed to be secure against such key-exposure attacks, with the downside that their schemes use heavy tools such as pairings [BF01] or lattice-basis-delegation [ABB10]. These are currently not compatible with any of the major cryptocurrencies that exist today. Furthermore, with the threat of quantum computers looming large, cryptocurrency payments including the *pre-quantum* stealth signature mechanisms of [vS, LYW+19] remain vulnerable. While a lattice-based (and thus plausibly post-quantum) construction of stealth signatures was proposed in [LLN+20], this construction relies on the aforementioned lattice basis delegation. Consequently, their scheme is most likely too inefficient for practical use[1]. We compare our constructions and related works in Table 1. Please refer to Appendix B for more discussion.

This work is motivated by the following two questions:

- *Can we have an efficient stealth signature scheme with security against unbounded key-exposures,*

[1] As the authors of [LLN+20] point out in Section 1.1, their "public key and signature sizes are too large for practical use".

Table 1: Comparison with Prior Works about Stealth Signatures

Works	w/KE[1]	Security	Post-quantum	opk Size	Signature Size
Monero's SS [vS]	∘	sEUFCMA	∘	64 B	64 B
Paring-based SS [LYW+19]	•	EUFCMA	∘	231 B	115 B
ABB10-based SS [LLN+20]	•	EUFCMA	•	3.35 GB	3.26 MB
[LLN+20] + NTRU (potential optimization)	•	EUFCMA	•	13.82 KB	13.82 KB
Appendix D	◒[2]	sEUFCMA	∘	96 B	64 B
Section 6.1	∘	EUFCMA	•	2.08 KB	2.54 KB
Section 6.1+Dilithium (compiler from Section 5)	•	sEUFCMA	•	2.08 KB	6.40 KB
Section 6.1+Falcon (compiler from Section 5)	•	sEUFCMA	•	2.08 KB	4.09 KB

[1] Secure against key-exposures. Our construction presented in Section 6.1 can be upgraded to w/KE according to Section 5.
[2] Secure against bounded key-exposures.

Table 2: Comparison with Prior Works about Fuzzy/Private Tracking

Works	Privacy	Assumptions	Post-quantum	Server's Work	Latency/msg[2]	Receiver's Time
FMD_2 [BLMG21]	ρN-anonymity[1]	Random Oracle	∘	$O(N)$	933 sec	37.5 ms
Π_{TEE} [MSS+21]	Full Privacy	Trusted Execute Environment	∘	$O(N)$	228 sec	12 ms
Π_{GC} [MSS+21]	Full Privacy	Two Non-colluding Servers	◒	$O(N)$	81.1 hour	1 ms
$\mathsf{OMR}_{\mathsf{p2}}$ [LT21]	Full Privacy	Fully Homomorphic Encryption	•	$O(N)$	43.1 hour	63 ms
Section 6.2	ρN-anonymity	Standard Model	•	$O(N)$	11.70 sec	37.5 ms
Section 6.3	ρN-anonymity	Random Oracle	•	$O(\rho N)$	3.42 ms	37.5 ms

[1] ρ denotes the false-positive rate and N the number of clients.
[2] Calculated in a setting with $N = 2^{20}$ users and $M = 500,000$ messages based on the numbers from their papers. Latency per message induced by the server. See more discussion in Appendix B.

that is compatible with Schnorr, ECDSA and other group based signature schemes predominantly used in currencies today?

- *Can we have an efficient stealth signature scheme secure with unbounded key exposure that is post-quantum secure?*

A caveat of the stealth address mechanism is that a recipient (online or offline) has to parse through a large number (hundreds of thousands per day) of transactions to identify those that send coins to one-time addresses corresponding to his master address. A workaround was proposed in [vS], where a recipient can *delegate* identification of incoming payments to a semi-trusted third party server called the *tracking server*. To do so, the recipient can generate a so-called *tracking key* from his secret key and provide it to the tracking server. The tracking key allows the tracking server to identify or *track* all incoming payments to the recipient using the tracking key, and later notify the recipient of these exact payments. On the other hand, such a tracking key should not enable the tracking server to generate one-time secrets for the concerned one-time addresses. Prior works [ADE+20, LYW+19, LLN+20] omit this important tracking functionality in their formalization of stealth signatures.

A downside of the above tracking method is that we fully give up anonymity/unlinkability with regards to the tracking server who learns *exactly* which payments are addressed to the recipient. While there is a natural and obvious tension between the anonymity goal of unlinkability and functional goal of trackability, a recent work of Beck et al. [BLMG21] attempts to strike a balance between these notions. They introduce the concept of *fuzzy message detection (FMD)*, where a tracking server can *approximately* detect messages meant for a recipient with adjustable degree of uncertainty. More specifically, their notion of detection is *fuzzy* in the sense that messages meant for the recipient are always correctly identified, but there is recipient-controlled false positive rate (baked into the *fuzzy* tracking key) which causes messages meant for other users to be misclassified as being meant for the recipient. Thus, the tracking server cannot decide with certainty if a detected message is actually intended for the recipient or not. This mechanism makes it necessary for the sender of the message to include additional *fuzzy tracking information* and the tracking server possesses a fuzzy tracking key. In principal, applying their technique to enable fuzzy tracking of one-time addresses in stealth signatures is straightforward. However, relying on their schemes comes with considerable drawbacks. While their first scheme (FMD_2) is efficient, it relies on the pre-quantum DDH assumption. Their second scheme ($\mathsf{FMD}_{\mathsf{frac}}$) relies on heavy tools like garbled circuits that lead to an unacceptable size-blowup of the sender's message. On the other hand, there are signalling detection or retrieval schemes [MSS+21, LT21] for fully private tracking instead of fuzzy tracking, but all of them require linear work at the server side which doesn't scale to thousands or millions of users. We discuss their schemes and ours in Appendix B and present a comparison in Table 2. This leads us to ask:

- *Can we have a stealth signature scheme with efficient fuzzy tracking in the post-quantum setting and scalable to hundreds of thousands (or even millions) of users?*

1.1 Our Contributions

We summarize our contributions below.

Modular Framework. We introduce Spirit (in Section 6.1), the first practically efficient post-quantum stealth signature scheme secure without key-exposure[2]. Towards this goal, we consider the lattice-based Dilithium [LDK+20] signature scheme, which is the winner of the NIST standardization competition and most likely candidate to be adopted into cryptocurrencies. Without changing the signature scheme in any way, we augment Dilithium with additional algorithms to obtain Spirit so that it now supports one-time key derivations and tracking.

Next, we show how one can generically transform (in Section 5) a stealth signature scheme that is secure *without* key-exposure into a scheme that is secure *with* unbounded key-exposure. Thus we can upgrade Spirit into one that is practically efficient and secure with unbounded key-exposure. Both Spirit and its upgrade are compatible with cryptocurrencies that would support Dilithium signature verification and no additional scripting is required.

Furthermore, we construct a stealth signature scheme (in Appendix D) that is compatible with group-based schemes like Schnorr, and ECDSA which are used in most of the currencies today. However, it only guarantees security with *bounded* key-exposure: It tolerates an a-priori number of one-time secret key leakage.

Fuzzy Constructions. We then present two fuzzy stealth signature schemes (using Spirit), both of which are the first efficient and post-quantum candidates.

In the first construction (in Section 6.2), we take a similar approach as FMD from [BLMG21]. But we reduce its overhead from $O(\lambda)$ to 1 bit per signal by making novel use of ciphertext compression techniques [BDGM19]. Additionally, we show how to allow *finer* false-positive rates without requiring heavy tools like garbled circuits as in [BLMG21].

We then present a new *scalable* framework for fuzzy tracking (in Section 4.4) followed by an efficient construction (in Section 6.3) in the random oracle model. This framework can be viewed as a 'dual' version of FMD mechanism from [BLMG21]. Intuitively, it is a trade-off between efficiency

[2] A recent work by [ADE+20] presents the construction of a re-randomized signature, which bears resemblance to the concept of stealth addresses. However, it is important to note that their proposed functionality does not offer public tracking support and is not secure against key-exposure attacks.

and usability: By limiting the users' ability to choose false-positive rates, we are able to reduce the tracking server's computational work to an amount which is *sublinear* in the total number of users. This compares very favourably with prior works, where the server needs to take a linear scan of each user's tracking key [BLMG21, MSS+21, LT21].

Implementation. We implemented SPIRIT, post-quantum FMD, and scalable fuzzy tracking based on Dilithium, Kyber, and Falcon with anonymized open-source code [imp]. We test them with different parameter sets on an ordinary laptop as presented in Table 3 and Table 4 (in Appendix B). Experiment results show that our stealth signature with strongest security only yields a 4.09 KB signature, while the verification time is less than 0.2 ms. Similarly, our scalable fuzzy tracking mechanism only takes 3.42 ms to filter each incoming message in the setting with millions of users.

Application to FIDO. As our final contribution, we surprisingly apply our stealth address notion to FIDO2 standard passwordless authentication schemes (formally defined in [BBCW21]). We show how manufacturers can implement device authenticators that not only provide post-quantum security but require limited secure memory (one master secret key), support global revocation (as defined in [HLW22]), multi-device credentials [All22], and can be used to implement asynchronous remote key generation ([FGK+20]).

2 Technical Overview

Let us first recall the group-based stealth signature scheme of [vS]: Given a cryptographic group $\mathbb{G} = \langle g \rangle$ of prime order p, the master public key is $\mathsf{mpk} := (g, h_0 := g^a, h_1 := g^b) \in \mathbb{G}^3$, where $\mathsf{msk} := (a \leftarrow\!\$\, \mathbb{Z}_p, b \leftarrow\!\$\, \mathbb{Z}_p)$ is the master secret key, and $\mathsf{mtk} := a$ is the tracking key. To re-randomize mpk to a one-time address (i.e., one-time public key), the sender samples a uniformly random $r \leftarrow\!\$\, \mathbb{Z}_p$ and computes $\mathsf{opk} := g^{\mathsf{H}(h_0^r)} \cdot h_1 \in \mathbb{G}$ where $\mathsf{H} : \mathbb{G} \mapsto \mathbb{Z}_p$ is a hash function modelled as a random oracle. Additionally, the sender attaches tracking information $\mathsf{tki} := g^r \in \mathbb{G}$ to the opk. To derive the corresponding one-time secret key osk from msk, the receiver computes $\mathsf{osk} := \mathsf{H}(\mathsf{tki}^a) + b \in \mathbb{Z}_p$ with the help of tki. Now, opk and osk satisfy the discrete-log relation $\mathsf{opk} = g^{\mathsf{osk}}$, hence the receiver can sign (for e.g., Schnorr or ECDSA) any message with osk to output a signature which can be verified with corresponding opk. An additional mechanism is that $\mathsf{mtk} := a$ can be given to a tracking server for tracking: By comparing whether $\mathsf{opk} \stackrel{?}{=} g^{\mathsf{H}(\mathsf{tki}^{\mathsf{mtk}})} \cdot h_1$, the tracking server can determine whether opk links to the issuer of mtk.

Taking a closer look, this approach to build stealth signature apparently can be generically decomposed to a linearly homomorphic one-way function $\mathsf{f} : \mathcal{D} \mapsto \mathcal{M}$ where $\mathsf{f}(x + y) = \mathsf{f}(x) + \mathsf{f}(y)$, and a key-exchange protocol $(\mathsf{KE}_1, \mathsf{KE}_2, \mathsf{KE}_3)$, where KE_i denotes the i-th message function:

$$\begin{aligned} \mathsf{ct}_1 \in \mathcal{C}_1 &\leftarrow \mathsf{KE}_1(r_1), \\ (\mathsf{ct}_2 \in \mathcal{C}_2, K \in \mathcal{K}) &\leftarrow \mathsf{KE}_2(r_2, \mathsf{ct}_1), \\ K \in \mathcal{K} &\leftarrow \mathsf{KE}_3(r_1, \mathsf{ct}_2), \end{aligned}$$

where r_1, r_2 are two user's secrets, and K is the agreed-upon key. Here, $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{K}$ represent the first message space, second message space, and the key space, respectively. Now, let $\mathsf{mpk} := \big(\mathsf{ct}_1 := \mathsf{KE}_1(r_1), B := \mathsf{f}(b)\big)$ and $\mathsf{msk} := (r_1, b), \mathsf{mtk} := r_1$. To publish a one-time address, the sender can just compute $(\mathsf{ct}_2, K) \leftarrow \mathsf{KE}_2(r_2, \mathsf{ct}_1)$ and publish

$$\mathsf{opk} := B + \mathsf{f}\big(\mathsf{H}(K)\big), \mathsf{tki} := \mathsf{ct}_2$$

where $\mathsf{H} : \mathcal{K} \mapsto \mathcal{D}$. Correspondingly,

$$\mathsf{osk} := b + \mathsf{H}\big(\mathsf{KE}_3(r_1, \mathsf{tki})\big).$$

Since they obey the relation $\mathsf{f}(\mathsf{osk}) = \mathsf{opk}$, we can leverage this to sign and verify. The tracking mechanism still works by checking if

$$\mathsf{opk} \stackrel{?}{=} \mathsf{f}\Big(\mathsf{H}\big(\mathsf{KE}_3(\mathsf{mtk}, \mathsf{tki})\big)\Big) + B.$$

We will now adapt this blueprint to construct a stealth signature in lattice setting.

2.1 Spirit: Lattice-based Stealth Signature

To make our protocol both *efficient* and *practical*, we would like to use optimized NIST winners as our building blocks. In this work, we choose Dilithium as the underlying digital signature considering that it is one of most popular signature schemes in NIST [LDK+20]. We call the resulting stealth signature scheme Spirit. Basically, it follows the above approach: In Dilithium, the public key is a Module Learning With Errors (MLWE) [BGV12] sample $\mathbf{t} := \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$ where its secret-error pair $(\mathbf{s}_1, \mathbf{s}_2)$ (both chosen from a suitable *short* distribution) acts as the secret key. Since MLWE involves only linear operations, we have that

$$\mathbf{t} + \mathbf{t}' = \mathbf{A}(\mathbf{s}_1 + \mathbf{s}_1') + \mathbf{s}_2 + \mathbf{s}_2'.$$

Yet, even though adding samples is approximately linearly homomorphic, this addition will increase *error rates* or lengths for both $\mathbf{s}_1$ and $\mathbf{s}_2$. Typically, the $\mathbf{s}_1$ and $\mathbf{s}_2$ are generated by sampling their coefficients uniformly with absolute value at most η (for some small parameter η). The increased norm of the new secrets $(\mathbf{s}_1 + \mathbf{s}_1', \mathbf{s}_2 + \mathbf{s}_2')$ will incur additional running time during signing due to the so-called "Fiat-Shamir with Abort" mechanism of Dilithium. To alleviate this issue, we only prove Spirit to be *existential unforgeable*. This will give us better parameters to balance between security and efficiency. Looking ahead, we point out that Spirit can be transformed to a *strongly existentially unforgeable* scheme using a generic compiler which we will introduce later.

Apart from a linearly homomorphic one-way functions, we still need a key-exchange protocol. However, this key-exchange needs some additional properties. Specifically, we need a non-interactive key-exchange (NIKE) protocol which is *substantially* stronger than KE we depicted above. The starting point is that it needs to be *anonymous* under chosen plaintext attacks (CPA), which means given the message ct_2, the adversary cannot link it to the ct_1 used to generate ct_2. This is for stealth signatures as we don't want our one-time address to be linkable to the original master public address. This security notion is formalized as *unlinkability*.

But anonymity under chosen plaintext attacks will not even suffice yet for our applications. We will require a stronger notion of anonymity under plaintext checking attacks (PCA). Here, the adversary is given an additional oracle which allows him to check whether a ciphertext-plaintext pair is valid or not. To see why this is necessary, consider an adversary who is trying to link some $(\mathsf{opk}, \mathsf{tki})$ to mpk. Such an adversary will be *able* to sample $\mathsf{ct}_2 \leftarrow\!\$\, \mathcal{C}_2, K \leftarrow\!\$\, \mathcal{K}$ to generate $(\mathsf{opk}', \mathsf{tki}')$, which can then be published to see if the tracking check passes. It turns out that anonymity under plaintext checking attacks is sufficient for this setting. However, we currently don't have a simple construction satisfying anonymity under plaintext checking attacks. As a consequence, we use an even stronger key-exchange protocol which is anonymous under chosen ciphertext attacks (CCA), namely, it is ANOCCA-secure (formalized in Definition C.7). Fortunately, the recent standardized KEM by NIST, Kyber [SAB+20], can be slightly modified to be ANOCCA-secure [GMP22] and we use Kyber in the concrete instantiation. There are multiple technical details not covered in this outline, for instance, besides tki, the one-time address opk itself also needs to be anonymous. We refer to Section 6.1 for detailed construction and analysis.

So far, we briefly mentioned two important security notions for stealth signatures, namely unforgeability and unlinkability (Section 4 for formalization). However, we note that we only formalize these two notions as unforgeability *without* key-exposure and unlinkability *without* key-exposure, respectively. It turns out the above approach to build stealth signatures (as well as in SPIRIT) is *no longer secure* if a one-time secret key osk leaks: Suppose the sender learns osk somehow, he can instantly recover msk as

$$b := \mathsf{osk} - \mathsf{H}\big(\mathsf{KE}_3(r_1, \mathsf{tki})\big),$$

if he knows r_1 which is used to generate corresponding opk.

2.2 Generic Transformation: Security with Key-exposure

As mentioned above and noticed in prior works [LYW+19, NMRL16], leaking one-time secret keys is almost as bad as leaking the master secret key. This is a potential issue in current practical stealth signature schemes [vS] and it is costly to avoid. For instance, if we are willing to use techniques implying hierarchical identity based encryption (HIBE), we could have a stealth signature scheme secure *with* key-exposure attacks by using pairing [BF01, LYW+19], lattice basis delegation [ABB10, LLN+20], or non-black box tools [DG17]. All of above techniques are several orders of magnitude slower in computational time, or orders of magnitude larger in signature or one-time public key size.

The reason we don't have a simple solution to this issue is that one-time secret keys are usually a linear function of the msk as mentioned in [LRR+19]. Apparently we can achieve security with *bounded* key-exposure by adding more secrets in msk where bounded key-exposure means msk remains secure if the number of leaked osk is smaller than some 'a priori bound' and we show a candidate construction in Appendix D. However, any generic-group based techniques to prevent unbounded key-exposure should imply IBE which is known to be impossible using only black-box techniques [PRV12, SGS21].

In this work, we provide a conceptually simple, generic, and powerful black-box compiler to tackle this problem in the context of stealth signatures (in Section 5): We use a short chain of signatures [Mer90] to compile any stealth signature $\mathsf{SS}_{\mathsf{w/o}}$ secure *without* key-exposure into a strong stealth signature SS_{w} secure *with* unbounded key-exposure. The high level idea is to break this 'linear' relation between osk and msk. Specifically, instead of generating osk directly, with the help of an additional digital signature DS, we generate

$$\mathsf{osk} := (\sigma_1, \mathsf{sk}, \mathsf{vk}),$$

where $\sigma_1 \leftarrow \mathsf{SS}_{\mathsf{w/o}}.\mathsf{Sign}(\mathsf{osk}', \mathsf{vk})$ and $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{DS}.\mathsf{Gen}(\lambda)$. Note that osk' is the one-time secret key in the scheme $\mathsf{SS}_{\mathsf{w/o}}$. Intuitively, since osk has a non-linear relation with msk, the adversary cannot recover msk from osk as $\mathsf{SS}_{\mathsf{w/o}}$ is unforgeable. To sign a message m, it runs $\sigma_2 \leftarrow \mathsf{DS}.\mathsf{Sign}(\mathsf{sk}, m)$ and outputs the final signature $\sigma := (\sigma_1, \sigma_2, \mathsf{vk})$. Similarly, to verify σ just use opk to verify the signature σ_1 on vk and use vk to verify the signature σ_2 on m. Compared to original stealth signature $\mathsf{SS}_{\mathsf{w/o}}$, our compiled one SS_{w} incurs slightly larger signature size and longer verification time, but in turn is far more efficient than above HIBE-related techniques.

Additionally, we show this compiler can also leverage $\mathsf{SS}_{\mathsf{w/o}}$ with existential unforgeability to SS_{w} with strong unforgeability via a small tweak: Instead of signing on m, we sign as $\sigma_2 \leftarrow \mathsf{DS}.\mathsf{Sign}(\mathsf{sk}, m||\sigma_1)$. This prevents strong unforgeability attacks of SS_{w} because: Assuming vk in σ is not altered, a different $\sigma_1' \neq \sigma_1$ will lead to a forgery $(m||\sigma_1', \sigma_2)$ of DS in SS_{w}. Therefore, SPIRIT can also be leveraged in this way to be strongly unforgeable with key-exposure. This gives us the first practical post-quantum SS_{w} secure with key-exposure.

2.3 Fuzzy Tracking

We will now turn to the issue that in the above constructions, the tracking mechanism will leak the users' metadata to the tracking servers, i.e., the tracking server will know exactly which mtk belongs to which specific $(\mathsf{opk}, \mathsf{tki})$. As discussed above, to address this problem, Beck et al. [BLMG21] proposed a mechanism named fuzzy message detection (FMD): The server is given a fuzzy tracking key ftk instead of mtk to *filter* incoming fuzzy tracking information ftki for its users. Here, ftki is attached with $(\mathsf{opk}, \mathsf{tki})$. Specifically, for unmatched ftki and ftk, they will be linked with probability roughly ρ.

Transforming their scheme to post-quantum world is non-trivial as there are still two potential obstacles in the lattice setting: First, it is not practically efficient since its ftki is as large as $O(n \cdot |\mathsf{ct}|)$-bit where $|\mathsf{ct}| = \mathsf{poly}(\lambda)$. This is highly undesirable in practise as our expectation is something like $O(\lambda) + n$. The other problem is the uniformly-ambiguous (recalled in Appendix C) encryption, as it is unclear how to extend the random oracle based approach in [BLMG21], to the lattice setting due to the presence of noise. We show that these two obstacles are related and can be resolved simultaneously. For simplicity, assume $n = 1$ for the moment. Recall that in Regev encryption with modulus q, the ciphertext is composed of two parts, a vector $\mathbf{c}_1 \in \mathbb{Z}_q^\ell$ and a scalar $c_2 \in \mathbb{Z}_q$. The secret key is $\mathbf{s} \in \mathbb{Z}_q^\ell$ and decryption consists of rounding after a linear operation:

$$\lceil \mathbf{s}^T \mathbf{c}_1 - c_2 \rfloor_2 = \lceil \frac{q}{2} \cdot m + e \rfloor_2,$$

where $e < B < \frac{q}{4}$ is a bounded error. This is not just bad for efficiency (as we need additional $n \log q$ bits to encrypt n more bits), but also for security: With the correct secret key $\mathbf{s}$, $\mathbf{s}^T \mathbf{c}_1 - c_2$ is distributed as a Gaussian around $\frac{q}{2}$ or 0; With a wrong key $\mathbf{s}_*$, $\mathbf{s}_*^T \mathbf{c}_1 - c_2$ is distributed uniformly random over the entire domain $\mathbb{Z}_q$. These two cases are clearly distinguishable by an adversary.

Our solution will be to compress c_2 into a single bit, which doesn't convey enough information about the distribution. Hence this idea will solve both of the above problems simultaneously. Brakerski et al. [BDGM19] introduced *rate-1* packed Regev encryption which can compress each c_2 to just one bit but require an additional offset scalar $z \in \mathbb{Z}_q$ in the header. Thus to to encrypt n bits, the ciphertext after compression is $(\mathbf{c}_1, z, w_1, \ldots, w_n)$ where $w_i \in \{0, 1\}$. To make the offset z statistically close to uniformly random (in our setting pseudorandom doesn't suffice because the adversary gets the secret key), we require *super-polynomial* noise-modulus ratio of Learning With Errors (LWE) [Reg05] which makes the scheme slightly less efficient. This gives us a lattice-based fuzzy tracking scheme (and ambiguous encryption), and surprisingly, it doesn't rely on heuristic assumptions like random oracles which are necessary in [BLMG21].

2.4 Scalable Fuzzy Tracking

We observe that in the above FMD style tracking, the server's computational work is $O(N)$ with N users and is not scalable when thousands (or millions) of users are using the service of the server. We provide a framework for *scalable* fuzzy tracking which we view as a dual version of FMD [BLMG21], where the server's work is *sublinear*. In this framework, we weaken the requirement that the false-positive rate can be adaptively changed by users. Instead, it is fixed in advance in this setting. This weakening is reasonable as it was shown in [SPB21] that an adversary can mount statistical attacks if users have varying false positive rates. To circumvent such attacks it was suggested that all users have high enough false positivity rates as even a small subset of low rate users can affect unlinkability for the entire pool of users. Therefore we can fix the false positivity rate to be a high enough value for everyone. For example, as calculated in [SPB21], the false-positive rate ρ is better to be as large as $\frac{1}{\sqrt{N}}$[3]. In this case, we can make the server's overhead $O(\rho N)$ for each incoming message which was at least $O(N)$ in prior works [BLMG21, MSS+21, LT21].

We let the tracking server run FTKGen in the beginning to publish fuzzy public key fpk and secretly hold the fuzzy tracking key ftk. For each ftki received from senders, the tracking server will expand ftki to a list of size t composed of potential users' master public keys to which ftki may belong to. The tracking server can then store $(\mathsf{opk}, \mathsf{tki})$ to the mailbox of each candidate in this list. Crucially, the master public keys of other potential candidates should remain uncontrollable to either the sender or the server. Otherwise the sender might manipulate the chance of each key appearing in the list. This additional property is named *unbiasedness*. This rules out the trivial solution, where for instance the sender just sends directly a range of master public keys including the targeted mpk.

Since mpk of each user can be large, in our construction we hash $\mathsf{mpk} \in \mathcal{K}$ to some small $\mathsf{hint} \in \mathcal{T}$ (while making $|\mathcal{T}| \geq N$) and use the hint to locate each user's mailbox. Our scheme is based on the underlying INDCPA encryption of Kyber, except that we use non-prime modulus. For instance, assuming the hint contains $n = \lceil \log N \rceil$ bits, i.e., $b := \mathsf{hint} \in \{0,1\}^n$, to generate ftki, the sender modifies the Kyber512's ciphertext $\mathsf{ct} := (\mathbf{c}_1, c_2)$ to $\mathsf{ct}' := (\mathbf{c}_1', c_2')$ as follows:

$$\mathbf{c}_1' := \mathbf{c}_1 + \frac{q}{2}\begin{bmatrix} x_i \\ 0 \end{bmatrix} \quad c_2' := c_2 + \frac{q}{2} y_i,$$

where ct (and ct') encrypts hint_i as the plaintext, $x_i \leftarrow \mathsf{encode}_{R_q}(\mathbf{x}_i)$ is a polynomial mapped from the vector $\mathbf{x}_i$, and $\mathbf{x}_i, \mathbf{y}_i \in \{0,1\}^m \leftarrow \mathsf{H}(\delta, i)$ are outputs of a hash function H with the seed δ. Here $i \in [t]$ denotes the i-th target mpk as the intended recipient.

For $\mathsf{ftki} := \mathsf{ct}'$, the tracking server decrypts ct' using the key $\mathsf{sk} = \mathbf{s}$ as follows: for $\forall j \in [t]$,

$$\mathsf{hint}_j \leftarrow \mathsf{decode}_{R_q}(\lceil \mathbf{s}^T(\mathbf{c}_1' - \frac{q}{2}\begin{bmatrix} x_j \\ 0 \end{bmatrix}) - c_2')\rfloor_2 \oplus y_j),$$

to get t potential hints. To argue privacy, intuitively, since $\mathbf{s}$ remains random to the sender, the decrypted hint for $j \neq i$ would also be random to the sender as

$$\mathsf{hint}_j = \mathsf{hint}_i \oplus (y_j \oplus y_i) \oplus \mathsf{decode}_{R_q}(\lceil \frac{q}{2}\mathbf{s}^T \begin{bmatrix} x_j \\ 0 \end{bmatrix} \rfloor_2).$$

However, to prove unbiasedness we mentioned above, we need to be careful because standard regularity lemma seems hard to apply with such small noise parameter and modulus in ideal lattices. Our solution is to rely on the specific structure of the corresponding cyclotomic polynomial and show that even $\mathbf{s}^T \begin{bmatrix} x_j \\ 0 \end{bmatrix}$ is not close to a uniformly random polynomial but there's enough entropy to make hint_j uniformly random over $\{0,1\}^n$ as long as n is much smaller than the degree of the polynomial.

2.5 From Stealth Addresses to FIDO

We will now describe how stealth addresses fit into FIDO-based passwordless authentication. The FIDO standard specifies ways of using device-based tokens (authenticators) for online authentication.

[3] According to [SPB21], "concerning recipient unlinkability and temporal detection ambiguity, the false-positive rate needs to be high and there must be a large number of users in the system." This can be achieved in current blockchain systems with millions of users by setting the false-positive rate to be $1/\sqrt{N}$. Although the same work points out that relationship anonymity only holds when senders are hidden from the server, this is typically addressed using other techniques such as ring signatures. Furthermore, as shown in [SPB21], "...users do not employ any cover traffic due to their selfishness...", which has inspired us to propose scalable fuzzy tracking as a solution to overcome this issue.

The protocol is a simple challenge-response where the authenticator creates a signature under the server's challenge. The server verifies the signature against a public key stored during the registration phase. Thus, the token uses different public keys per server to ensure privacy, which poses a challenge for memory-constrained devices. The common practice is to use key wrapping or a key derivation function to generate the signing key ad-hoc during the authentication process, where the server provides additional information (e.g., the ciphertext wrapping the signing keys or random value for the derivation function) for the re-computation.

Applying stealth addresses to this setting would use the same idea. The authenticator only needs to store the master secret key msk and the tracking key mtk. During the registration phase, the server would receive a one-time public key opk and send it to the token during authentication. The authenticator can then reconstruct the corresponding one-time secret key osk and respond to the server's challenge. Interestingly, due to our strong key-exposure notions, one-time keys can be leaked without compromising the unforgeability of non-leaked keys. Thus, we can use the same keys on multiple devices owned by the user, implementing the concept of FIDO multi-device credentials [All22].

Contrary to existing solutions, the public keys do not have to be generated on-token. The user platform (e.g., browser, second token) can generate the one-time public key opk without msk by just using the master public key mpk. What we just described is also called asynchronous remote key generation [FGK+20], a solution that cleverly uses two devices to solve the token loss problem. After an initialization phase, one of the authenticators is put into cold storage while registration (in the name of both) is done using the primary token. In case of loss, the authenticator in storage can create a valid response to the server's challenge. Using stealth addresses provides the same feature without a complicated initialization step.

One problem with lost devices is that the public keys on the server side are still bound to the user's account. A simple but bothersome solution is for the user to contact all servers it used in the past and revoke the keys. A more flexible solution was proposed by Hanzlik, Loss, and Wagner [HLW22]. With the help of a revocation key published by the user, servers can identify public keys corresponding to lost tokens. Stealth addresses provide the same feature and even improve it a bit. One can identify one-time public keys generated using the same master public key mpk with the help of the tracking key mtk. In the FIDO scenario, a published mtk can be used to globally revoke a lost/stolen token while at the same time not allowing to create forger signature as required by the notions in [HLW22].

Releasing the tracking key mtk will allow everyone to link the one-time public keys, which is a privacy concern. Fuzzy tracking allows for more fine-tuned protection in the case of lost/stolen tokens. Instead of revoking public keys, servers could employ an additional policy-based mechanism to challenge authentications against public keys identified by fuzzy tracking. In other words, the user can hide the actual public key in a set of potential keys, and the server requires some additional authentication factors for those keys. Authenticating using the lost/stolen token will be impossible in such a scenario while at the same time providing extra privacy for the user that lost the device.

3 Preliminaries

We denote by $\lambda \in \mathbb{N}$ the security parameter and by $x \leftarrow \mathcal{A}(\mathsf{in}; r)$ the output of the algorithm $\mathcal{A}$ on input in using $r \leftarrow \{0,1\}^*$ as its randomness. We often omit this randomness and only mention it explicitly when required. The notation $[n]$ denotes a set $\{1,\ldots,n\}$ and $\mathbf{x}[: n]$ denotes the sub-vector of $\mathbf{x}$ with first n elements. We consider *probabilistic polynomial time* (PPT) machines as efficient algorithms. Also, we use $\approx_c$ and $\approx_s$ to denote computational closeness and statistical closeness, respectively. We defer the reader to Appendix C for assumptions and analysis tools we use in this work. Apart from this, we make use of the following cryptographic primitives.

Digital Signatures. A digital signature scheme DS, formally, has a key generation algorithm $\mathsf{KGen}(\lambda)$ that takes the security parameter λ and outputs the verification/signing key pair $(\mathsf{vk}, \mathsf{sk})$, a signing algorithm $\mathsf{Sign}(\mathsf{sk}, m)$ inputs a signing key and a message $m \in \{0,1\}^*$ and outputs a signature σ, and a verification algorithm $\mathsf{Vf}(\mathsf{vk}, m, \sigma)$ outputs 1 if σ is a valid signature on m under the verification key vk, and outputs 0 otherwise. We require unforgeability, which guarantees that a PPT adversary cannot forge a fresh signature on a fresh message of its choice under a given verification key while having access to a signing oracle (that returns a valid signatures on the queried messages). Formally the notion can be captured in an experiment denoted by EUFCMA. Strong unforgeability refers to the case where the adversary is required to forge a fresh signature on not necessarily a fresh message. Formally the notion can be captured in an experiment denoted by $\mathsf{sEUFCMA}$.

Key Encapsulation Mechanism. A key encapsulation mechanism KEM, formally, has a key generation algorithm $\mathsf{KGen}(\lambda)$ that takes the security parameter λ and outputs a encaps key ek and a decaps key dk. An encapsulation algorithm $\mathsf{Encaps}(\mathsf{ek})$ inputs an encaps key and outputs a ciphertext C and agreed key K. Finally, we have a decapsulation algorithm $\mathsf{Decaps}(\mathsf{dk})$ inputs a decaps key and a ciphertext and outputs an agreed key K. Apart from INDCCA security, we additionally require its anonymous property which can be formally captured in Definition C.7 denoted by ANOCCA and it means the adversary cannot link any ciphertext C to its encaps key ek even being able to access a decaps oracle. Concretely, we use Kyber [SAB+20] with the modification shown in Figure 6 of [GMP22].

4 Definitions of (Fuzzy) Stealth Signatures

In this section we first present our formal definitions for a stealth signature scheme, followed by how we can add-on fuzziness to the scheme. Note that stealth signatures were formalized in prior works [LYW+19, LLN+20], however our formalization of security is strictly stronger than theirs, and moreover we are the first to formalize tracking and fuzzy tracking for a stealth signature scheme. We will point out the exact differences[4] in the formalism as we introduce the security notions formally.

Below we present the definition of stealth signatures, that formalizes the tracking of keys which was absent in prior works. This formalization allows for tracking to be outsourced to third-party servers.

Definition 4.1. A *stealth signature* (SS) scheme consists of the PPT algorithms (MKGen, OPKGen, OSKGen, Track, Sign, Vf) that are defined as follows.

$\underline{(\mathsf{mpk}, \mathsf{msk}, \mathsf{mtk}) \leftarrow \mathsf{MKGen}(\lambda)}$: the master key generation algorithm takes as input the security parameter λ and outputs the master public key mpk, the master secret key msk, and the master tracking key mtk.

$\underline{(\mathsf{opk}, \mathsf{tki}) \leftarrow \mathsf{OPKGen}(\mathsf{mpk})}$: the one-time public key generation algorithm takes as input the master public key mpk, and outputs the one-time public key opk and a tracking information tki.

$\underline{\mathsf{osk}/\perp \leftarrow \mathsf{OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})}$: the one-time secret key generation algorithm takes as input the master secret key msk, the one-time public key opk, and the tracking information tki, and outputs a one-time secret key osk or a special symbol $\perp$.

$\underline{\mathsf{true}/\mathsf{false} \leftarrow \mathsf{Track}(\mathsf{mtk}, \mathsf{opk}, \mathsf{tki})}$: the tracking algorithm takes as input the master tracking key mtk, the one-time public key opk, and the tracking information tki, and outputs true or false.

[4]Please refer to Definition 4.6 for details.

$\sigma/\perp\leftarrow \mathsf{Sign}(\mathsf{osk}, m)$: the signing algorithm takes as input the one-time secret key osk, and a message m, and outputs a signature σ or a special symbol $\perp$.

$\mathsf{true}/\mathsf{false} \leftarrow \mathsf{Vf}(\mathsf{opk}, m, \sigma)$: the verification algorithm takes as input the one-time public key opk, a message m, and a signature σ, and outputs true or false.

The notion of correctness is formalized below.

Definition 4.2 (Correctness)**.** A SS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf) is said to be *correct* if for all $\lambda \in \mathbb{N}$, all $(\mathsf{mpk}, \mathsf{msk}, \mathsf{mtk}) \leftarrow \mathsf{MKGen}(\lambda)$, all $(\mathsf{opk}, \mathsf{tki}) \leftarrow \mathsf{OPKGen}(\mathsf{mpk})$, all $\mathsf{osk} \leftarrow \mathsf{OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})$, we have the following that hold simultaneously:

- we have $\Pr[\mathsf{Track}(\mathsf{mtk}, \mathsf{opk}, \mathsf{tki}) = \mathsf{true}] = 1$
- we have $\Pr[\mathsf{Vf}(\mathsf{opk}, m, \mathsf{Sign}(\mathsf{osk}, m)) = \mathsf{true}] = 1$,

note that sometimes we don't require *perfect* correctness and having correctness probability $1-\mathsf{negl}(\lambda)$ instead would suffice.

4.1 Security of SS Without Key Exposure

In terms of security, we first want unforgeability, which guarantees that it is infeasible for an adversary to forge a signature on a (fresh) message wrt. some one-time public key opk^* for a master public key mpk. The adversary is given access to a one-time secret key generation oracle $\mathsf{OSKGen}\mathcal{O}$ using which the adversary can generate a fresh one-time secret key. However, the adversary does not get to learn the generated one-time secret keys, therefore the notion is said to be *without key exposure*. The adversary also has access to a signing oracle, to which it can query a signature on any message of its choice wrt. any one-time secret key that has been generated with a query to $\mathsf{OSKGen}\mathcal{O}$. The formal definition is presented below.

Definition 4.3 (Unforgeability without key-exposure)**.** A SS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf) is said to be *unforgeable without key exposure* if there exists a negligible function negl for all $\lambda \in \mathbb{N}$, and for all PPT adversaries $\mathcal{A}$ the following holds:

$$\Pr\Big[\mathsf{sEUFCMA}^{\mathcal{A}}_{\mathsf{w/o-ke}}(\lambda) = 1\Big] \leq \mathsf{negl}(\lambda)$$

where $\mathsf{sEUFCMA}_{\mathsf{w/o-ke}}$ is defined in Figure 1.

We then want unlinkability, which guarantees that it is infeasible for an adversary to associate a one-time public key to the master public key wrt. which it was generated. The adversary is given two master public keys mpk_0 and mpk_1, while also given a challenge one-time public key opk_b and the corresponding tracking information tki_b (for $b \in \{0, 1\}$) generated wrt. mpk_b. The adversary is given access to the $\mathsf{OSKGen}\mathcal{O}$ as before, and a signing oracle. The adversary is not given access to any of the one-time secret keys and therefore the notion is said to be *without key exposure*. The formal definition is presented below.

Definition 4.4 (Unlinkability without key-exposure)**.** A SS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf) is said to be *unlinkability without key exposure* if there exists a negligible function negl for all $\lambda \in \mathbb{N}$, and for all PPT adversaries $\mathcal{A}$ the following holds:

$$\Pr\Big[\mathsf{UNLNK}^{\mathcal{A}}_{\mathsf{w/o-ke}}(\lambda) = 1\Big] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$ is defined in Figure 2.

$\mathsf{EUFCMA}^{\mathcal{A}}_{\mathsf{w/o-ke}}(\lambda)$

$(\mathsf{mpk}, \mathsf{msk}, \mathsf{mtk}) \leftarrow \mathsf{MKGen}(\lambda)$
$\mathsf{OK} := [], Q := \emptyset$
(m^*, σ^*, i^*)
$\quad \leftarrow \mathcal{A}^{\mathsf{OSKGen}\mathcal{O}, \mathsf{Sign}\mathcal{O}}(\mathsf{mpk}, \mathsf{mtk})$
$(\mathsf{opk}^*, \mathsf{osk}^*) := \mathsf{OK}[i^*]$
$b_0 := (m^*, \cdot, i^*) \notin Q$
$\quad /\!/ \ (m^*, \sigma^*, i^*) \notin Q$ for $\mathsf{sEUFCMA}_{\mathsf{w/o-ke}}$
$b_1 := \mathsf{Vf}(\mathsf{opk}^*, m^*, \sigma^*) \stackrel{?}{=} \mathsf{true}$
return $b_0 \wedge b_1$

$\mathsf{OSKGen}\mathcal{O}(\mathsf{opk}, \mathsf{tki})$

$\mathsf{osk} \leftarrow \mathsf{OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})$
$\mathsf{OK} := \mathsf{OK} || (\mathsf{opk}, \mathsf{osk})$
return 1

$\mathsf{Sign}\mathcal{O}(i, m)$

$(\mathsf{opk}, \mathsf{osk}) \leftarrow \mathsf{OK}[i]$
$\sigma \leftarrow \mathsf{Sign}(\mathsf{osk}, m)$
$Q := Q \cup (m, \sigma, i)$
return σ

Figure 1: Experiment for unforgeability without key exposure.

$\mathsf{UNLNK}^{\mathcal{A}}_{\mathsf{w/o-ke}}(\lambda)$

$(\mathsf{mpk}_0, \mathsf{msk}_0, \mathsf{mtk}_0) \leftarrow \mathsf{MKGen}(\lambda)$
$(\mathsf{mpk}_1, \mathsf{msk}_1, \mathsf{mtk}_1) \leftarrow \mathsf{MKGen}(\lambda)$
$\mathsf{OK}_0 := \mathsf{OK}_1 := []$
$b \leftarrow \{0, 1\}$
$(\mathsf{opk}_b, \mathsf{tki}_b) \leftarrow \mathsf{OPKGen}(\mathsf{mpk}_b)$
$\mathsf{osk}_b \leftarrow \mathsf{OSKGen}(\mathsf{msk}_b, \mathsf{opk}_b, \mathsf{tki}_b)$
$b' \leftarrow \mathcal{A}^{\mathsf{OSKGen}\mathcal{O}, \mathsf{Sign}\mathcal{O}}(X, \mathsf{opk}_b, \mathsf{tki}_b)$
$\quad /\!/$ where $X := (\mathsf{mpk}_0, \mathsf{mpk}_1)$
$b_0 := (b = b')$
return b_0

$\mathsf{OSKGen}\mathcal{O}(b^*, \mathsf{opk}, \mathsf{tki})$

$\mathsf{osk} \leftarrow \mathsf{OSKGen}(\mathsf{msk}_{b^*}, \mathsf{opk}, \mathsf{tki})$
$\mathsf{OK}_{b^*} := \mathsf{OK}_{b^*} || (\mathsf{opk}, \mathsf{osk})$
return 1

$\mathsf{Sign}\mathcal{O}(b^*, i, m)$

if $i = -1$ **then**
$\quad \sigma \leftarrow \mathsf{Sign}(\mathsf{osk}_b, m)$
else
$\quad (\mathsf{opk}, \mathsf{osk}) \leftarrow \mathsf{OK}_{b^*}[i]$
$\quad \sigma \leftarrow \mathsf{Sign}(\mathsf{osk}, m)$
return σ

Figure 2: Experiment for unlinkability without key exposure.

4.2 Security of SS With Key Exposure

Prior works [LYW+19, LLN+20] formalized security with additionally giving adversary the one-time secret keys, i.e., the $\mathsf{OSKGen}\mathcal{O}$ returns the generated osk to the adversary.

The unforgeability notion *with key exposure* is formalized below. Notice that our formalization exposes the one-time secret keys osk to the adversary except the key wrt. which the adversary forges the signature.

Definition 4.5 (Unforgeability with key-exposure)**.** A SS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf) is said to be *unforgeable with key exposure* if there exists a negligible function negl for all $\lambda \in \mathbb{N}$, and for all PPT adversaries $\mathcal{A}$ the following holds:

$$\Pr\left[\mathsf{sEUFCMA}^{\mathcal{A}}_{\mathsf{w-ke}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda)$$

where $\mathsf{sEUFCMA}_{\mathsf{w/o-ke}}$ is defined in Figure 3.

The notion of unlinkability *with key exposure* is formalized below. Similar to the case above, the $\mathsf{OSKGen}\mathcal{O}$ returns the generated osk.

Definition 4.6 (Unlinkability with key-exposure)**.** A SS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf) is said to be *unlinkability with key exposure* if there exists a negligible function negl for all

$\mathsf{sEUFCMA}^{\mathcal{A}}_{\mathsf{w-ke}}(\lambda)$

$(\mathsf{mpk}, \mathsf{msk}, \mathsf{mtk}) \leftarrow \mathsf{MKGen}(\lambda)$
$\mathsf{OK} := [], Q := \emptyset$
(m^*, σ^*, i^*)
$\quad \leftarrow \mathcal{A}^{\mathsf{OSKGen}\mathcal{O}, \mathsf{Sign}\mathcal{O}}(\mathsf{mpk}, \mathsf{mtk})$
$(\mathsf{opk}^*, \mathsf{osk}^*, \cdot) := \mathsf{OK}[i^*]$
$b_0 := (m^*, \sigma^*, i^*) \notin Q$
$b_1 := \mathsf{Vf}(\mathsf{opk}^*, m^*, \sigma^*) = 1$
$b_2 := (\mathsf{OK}[i^*] \neq (\cdot, \cdot, \mathsf{true}))$
return $b_0 \wedge b_1 \wedge b_2$

$\mathsf{OSKGen}\mathcal{O}(i, \mathsf{opk}, \mathsf{tki}, \mathsf{flag})$

if $\mathsf{OK}[i] = (\mathsf{opk}, \cdot, \cdot) \wedge \mathsf{flag} = \mathsf{true}$
return $\mathsf{OK}[i].\mathsf{osk}$
$\mathsf{osk} \leftarrow \mathsf{OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})$
$\mathsf{OK} := \mathsf{OK} || (\mathsf{opk}, \mathsf{osk}, \mathsf{flag})$
if $\mathsf{flag} = \mathsf{true}$ **then return** osk
else return 1

$\mathsf{Sign}\mathcal{O}(i, m)$

$(\mathsf{opk}, \mathsf{osk}, \mathsf{flag}) \leftarrow \mathsf{OK}[i]$
$\sigma \leftarrow \mathsf{Sign}(\mathsf{osk}, m)$
$Q := Q \cup (m, \sigma, i)$
return σ

Figure 3: Experiment for unforgeability with key exposure.

$\mathsf{UNLNK}^{\mathcal{A}}_{\mathsf{w-ke}}(\lambda)$

$(\mathsf{mpk}_0, \mathsf{msk}_0, \mathsf{mtk}_0) \leftarrow \mathsf{MKGen}(\lambda)$
$(\mathsf{mpk}_1, \mathsf{msk}_1, \mathsf{mtk}_1) \leftarrow \mathsf{MKGen}(\lambda)$
$\mathsf{OK}_0 := \mathsf{OK}_1 := []$
$b \leftarrow \{0, 1\}$
$(\mathsf{opk}_b, \mathsf{tki}_b) \leftarrow \mathsf{OPKGen}(\mathsf{mpk}_b)$
$\mathsf{osk}_b \leftarrow \mathsf{OSKGen}(\mathsf{msk}_b, \mathsf{opk}_b, \mathsf{tki}_b)$
$b' \leftarrow \mathcal{A}^{\mathsf{OSKGen}\mathcal{O}}(X, \mathsf{opk}_b, \mathsf{tki}_b, \mathsf{osk}_b)$
$\quad$ // where $X := (\mathsf{mpk}_0, \mathsf{mpk}_1)$
$b_0 := (b = b')$
return b_0

$\mathsf{OSKGen}\mathcal{O}(b^*, \mathsf{opk}, \mathsf{tki})$

$\mathsf{osk} \leftarrow \mathsf{OSKGen}(\mathsf{msk}_{b^*}, \mathsf{opk}, \mathsf{tki})$
$\mathsf{OK}_{b^*} := \mathsf{OK}_{b^*} || (\mathsf{opk}, \mathsf{osk})$
return osk

Figure 4: Experiment for unlinkability with key exposure.

$\lambda \in \mathbb{N}$, and for all PPT adversaries $\mathcal{A}$ the following holds:

$$\Pr\Big[\mathsf{UNLNK}^{\mathcal{A}}_{\mathsf{w-ke}}(\lambda) = 1\Big] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$ is defined in Figure 4.

Remark. It is worth noting that our formalization apart from the tracking functionality, is *stronger* than prior works in that the adversary is even given the challenge one-time secret key osk_b.

4.3 Fuzzy Stealth Signatures

We now formally incorporate the fuzzy tracking functionality into the definition of stealth signing.

Definition 4.7 (Fuzzy Stealth Signatures)**.** A *fuzzy stealth signatures* (F-SS) scheme is a SS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf) with additional interfaces (FTKGen, FTrack) defined below.

$\underline{(\mathsf{opk}, \mathsf{tki}, \mathsf{ftki}) \leftarrow \mathsf{OPKGen}(\mathsf{mpk})}$: overloading the interface OPKGen to output the fuzzy tracking information ftki.

$\mathsf{ftk} \leftarrow \mathsf{FTKGen}(\mathsf{mtk}, \rho)$: the fuzzy tracking key generation algorithm takes as input the master tracking key mtk, and a false positivity rate ρ, and outputs a fuzzy tracking key ftk.

$\mathsf{true}/\mathsf{false} \leftarrow \mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki})$: the fuzzy tracking algorithm takes as input the fuzzy tracking key ftk, the fuzzy tracking information ftki, and outputs true or false.

We define the notion of correctness below. We borrow the notion of fuzziness from [BLMG21] and adapt the same for the stealth signature setting. Intuitively, the correctness of fuzzy tracking says that with a probability ρ, the fuzzy tracking algorithm returns true for a mismatched fuzzy tracking key and a one-time public key. For a correctly matched fuzzy tracking key and a one-time public key, the tracking algorithm always returns true.

Definition 4.8 (Correctness for fuzzy tracking)**.** A F-SS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf, FTKGen, FTrack) is said to be *correct* if the original SS scheme is correct and if for all $\lambda \in \mathbb{N}$, all $\rho \in (0, 1]$ such that $\log_2 \rho \in \mathbb{Z}$, all $(\mathsf{mpk}, \mathsf{msk}, \mathsf{mtk}) \leftarrow \mathsf{MKGen}(\lambda)$, all $(\mathsf{opk}, \mathsf{tki}, \mathsf{ftki}) \leftarrow \mathsf{OPKGen}(\mathsf{mpk})$, all $\mathsf{osk} \leftarrow \mathsf{OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})$, all $\mathsf{ftk} \leftarrow \mathsf{FTKGen}(\mathsf{mtk}, \rho)$, we have the following that holds simultaneously:

- $\Pr[\mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki}) = \mathsf{true}] = 1$
- and for any $\mathsf{ftki}' \notin \mathsf{SUPP}(\mathsf{OPKGen}(\mathsf{mpk}))$, we have

$$\Pr\left[\mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki}') = \mathsf{true}\right] = \rho.$$

The unforgeability notion remains the same as in Figure 3, as the adversary in this notion already has access to the master tracking key.

Unlinkability with fuzzy tracking ensures that it is computationally infeasible for an adversary, given two fuzzy tracking keys that both return either true or false when tracking a challenge one-time public key $(\mathsf{opk}_b, \mathsf{ftki}_b)$ simultaneously, to associate $(\mathsf{opk}_b, \mathsf{ftki}_b)$ with the correct tracking key (either ftk_0 or ftk_1). The adversary is said to violate the notion if it can guess correctly the association non-negligibly more than $1/2$.

Definition 4.9 (Unlinkability with key-exposure and fuzzy tracking)**.** A F-SS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf, FTKGen, FTrack) is said to be *unlinkable with key-exposure and fuzzy tracking* if there exists a negligible function negl for all $\lambda \in \mathbb{N}$, all $\rho \in (0, 1]$ such that $\log_2 \rho \in \mathbb{Z}$, and for all PPT adversaries $\mathcal{A}$ the following holds:

$$\Pr\left[\mathsf{UNLNK}^{\mathcal{A}}_{\mathsf{fw-ke}}(\lambda, \rho) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathsf{UNLNK}_{\mathsf{fw-ke}}$ is defined in Figure 5.

4.4 Scalable Fuzzy Tracking

We now formalize the functionality, correctness, and security of fuzzy scalable stealth signatures as follows.

Definition 4.10 (Fuzzy Scalable Stealth Signatures)**.** A *fuzzy scalable stealth signature* (F-SSS) is a SS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf) with additional interfaces ($\mathsf{FTKGen}, \mathsf{FTrack}$) and a modified OPKGen defined below.

$(\mathsf{fpk}, \mathsf{ftk}) \leftarrow \mathsf{FTKGen}(\rho, N)$: the fuzzy tracking key generation algorithm takes as input a false positivity rate ρ, and the number of total users N, and outputs a fuzzy tracking key ftk and fuzzy public key fpk. The algorithm is run by the tracking server ahead of time.

$\mathsf{UNLNK}_{\mathsf{fw-ke}}(\lambda)$	$\mathsf{OSKGen}\mathcal{O}(b^*, \mathsf{opk}, \mathsf{tki})$
$\mathsf{OK}_0 := \mathsf{OK}_1 := []$ $(\mathsf{mpk}_0, \mathsf{msk}_0, \mathsf{mtk}_0) \leftarrow \mathsf{MKGen}(\lambda)$ $(\mathsf{mpk}_1, \mathsf{msk}_1, \mathsf{mtk}_1) \leftarrow \mathsf{MKGen}(\lambda)$ $b \leftarrow \{0,1\}$ $(\mathsf{opk}_b, \mathsf{tki}_b, \mathsf{ftki}_b) \leftarrow \mathsf{OPKGen}(\mathsf{mpk}_b)$ $\mathsf{osk}_b \leftarrow \mathsf{OSKGen}(\mathsf{msk}_b, \mathsf{opk}_b, \mathsf{tki}_b)$ $(\mathsf{st}_{\mathcal{A}}, \rho) \leftarrow \mathcal{A}_1(\mathsf{mpk}_0, \mathsf{mpk}_1, \mathsf{opk}_b,$ $\quad \mathsf{tki}_b, \mathsf{ftki}_b, \mathsf{osk}_b)$ $\mathsf{ftk}_0 \leftarrow \mathsf{FTKGen}(\mathsf{mtk}_0, \rho)$ $\mathsf{ftk}_1 \leftarrow \mathsf{FTKGen}(\mathsf{mtk}_1, \rho)$ $b_1 \leftarrow \mathsf{FTrack}(\mathsf{ftk}_0, \mathsf{ftki}_b)$ $b_2 \leftarrow \mathsf{FTrack}(\mathsf{ftk}_1, \mathsf{ftki}_b)$ **if** $b_1 = b_2$ $\quad b' \leftarrow \mathcal{A}_2^{\mathsf{OSKGen}\mathcal{O}}(\mathsf{st}_{\mathcal{A}}, \mathsf{ftk}_0, \mathsf{ftk}_1)$ **else** $\quad b' \leftarrow\$ \{0,1\}$ **return** $(b = b')$	$\mathsf{osk} \leftarrow \mathsf{OSKGen}(\mathsf{msk}_{b^*}, \mathsf{opk}, \mathsf{tki})$ $\mathsf{OK}_{b^*} := \mathsf{OK}_{b^*} \| (\mathsf{opk}, \mathsf{osk})$ **return** osk

Figure 5: Experiment for unlinkability of F-SS with key-exposure.

$(\mathsf{opk}, \mathsf{tki}, \mathsf{ftki}) \leftarrow \mathsf{OPKGen}(\mathsf{mpk}, \mathsf{fpk})$: overloading the interface OPKGen to additionally take input fpk and output fuzzy tracking information ftki.

$\mathsf{list} \leftarrow \mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki})$: the fuzzy tracking algorithm takes as input the fuzzy tracking key ftk, the fuzzy tracking information ftki, and outputs a list consisting of master public keys.

Definition 4.11 (Correctness for fuzzy scalable stealth signatures)**.** A F-SSS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf, FTKGen, FTrack) is said to be *correct* if the original SS scheme is correct and if for all $\lambda \in \mathbb{N}$, any integer N, all $\rho \in (0,1]$ such that $\frac{1}{2^{\lceil \log_2 N \rceil}} \mid \rho$, all $(\mathsf{mpk}, \mathsf{msk}, \mathsf{mtk}) \leftarrow \mathsf{MKGen}(\lambda)$, all $(\mathsf{opk}, \mathsf{tki}, \mathsf{ftki}) \leftarrow \mathsf{OPKGen}(\mathsf{mpk}, \mathsf{fpk})$, all $\mathsf{osk} \leftarrow \mathsf{OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})$, all $(\mathsf{fpk}, \mathsf{ftk}) \leftarrow \mathsf{FTKGen}(\rho, N)$, we have the following that holds simultaneously:

- $\Pr[\mathsf{mpk} \in \mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki})] = 1$
- and for any $\mathsf{mpk}' \neq \mathsf{mpk}$, we have

$$\Pr\big[\mathsf{mpk}' \in \mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki})\big] = \rho.$$

Crucially, we omit opk in FTrack as ftki is already associated with opk and we still have the regular Track algorithm that works with tk, opk and tki for tracking. The correctness definition above 'ties' together the keys ftk, mpk and mtk, and $(\mathsf{opk}, \mathsf{tki}, \mathsf{ftki}) \leftarrow \mathsf{OPKGen}(\mathsf{mpk})$ by requiring that $\mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki})$ always returns 1.

Definition 4.12 (Unlinkability with key-exposure and fuzzy scalable tracking)**.** A F-SSS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf, FTKGen, FTrack) is said to be *unlinkable with key-exposure and fuzzy scalable tracking* if there exists a negligible function negl for all $\lambda \in \mathbb{N}$, any integer N, all $\rho \in (0,1]$ such that $\frac{1}{2^{\lceil \log_2 N \rceil}} \mid \rho$, and for all PPT adversaries $\mathcal{A}$ the following holds:

$$\Pr\Big[\mathsf{UNLNK}^{\mathcal{A}}_{\mathsf{f^sw-ke}}(\lambda, \rho, N) = 1\Big] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

$\mathsf{UNLNK}_{\mathsf{fs}_{\mathsf{w-ke}}}(\lambda, \rho, N)$	$\mathsf{UNIUBS}_{\mathsf{fs}}(\lambda, \rho, N)$
$\mathsf{OK}_0 := \mathsf{OK}_1 := []$ $(\mathsf{mpk}_0, \mathsf{msk}_0, \mathsf{mtk}_0) \leftarrow \mathsf{MKGen}(\lambda)$ $(\mathsf{mpk}_1, \mathsf{msk}_1, \mathsf{mtk}_1) \leftarrow \mathsf{MKGen}(\lambda)$ $(\mathsf{fpk}, \mathsf{ftk}) \leftarrow \mathsf{FTKGen}(\rho, N)$ $b \leftarrow \{0, 1\}$ $(\mathsf{opk}_b, \mathsf{tki}_b, \mathsf{ftki}_b) \leftarrow \mathsf{OPKGen}(\mathsf{mpk}_b, \mathsf{fpk})$ $\mathsf{osk}_b \leftarrow \mathsf{OSKGen}(\mathsf{msk}_b, \mathsf{opk}_b, \mathsf{tki}_b)$ $\mathsf{list} \leftarrow \mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki}_b)$ **if** $\mathsf{mpk}_0 \in \mathsf{list} \wedge \mathsf{mpk}_1 \in \mathsf{list}$ $\quad b' \leftarrow \mathcal{A}_2^{\mathsf{OSKGen}\mathcal{O}}(\mathsf{ftk}, \mathsf{mpk}_0, \mathsf{mpk}_1, \mathsf{opk}_b, \mathsf{tki}_b, \mathsf{ftki}_b, \mathsf{osk}_b)$ **else** $\quad b' \leftarrow\$ \{0, 1\}$ $b_0 := (b = b')$ **return** b_0	$(\mathsf{fpk}, \mathsf{ftk}) \leftarrow \mathsf{FTKGen}(\rho, N)$ $(\mathsf{st}_{\mathcal{A}}, \mathsf{ftki}, i, j, \mathsf{mpk}) \leftarrow \mathcal{A}_1(\mathsf{fpk})$ $\mathsf{list} \leftarrow \mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki})$ $b \leftarrow\$ \{0, 1\}$ **if** $\mathsf{list}[i] \neq \mathsf{mpk} \vee i = j \vee \mathsf{mpk} \notin \mathcal{K}$ $\quad b' \leftarrow\$ \{0, 1\}$ **else** $\quad \mathbf{v}^0 := \mathsf{list}[j], \mathbf{v}^1 \leftarrow\$ \mathcal{K}$ $\quad b' \leftarrow \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathbf{v}^b)$ **return** $b \stackrel{?}{=} b'$ $\mathsf{OSKGen}\mathcal{O}(b^*, \mathsf{opk}, \mathsf{tki})$ $\mathsf{osk} \leftarrow \mathsf{OSKGen}(\mathsf{msk}_{b^*}, \mathsf{opk}, \mathsf{tki})$ $\mathsf{OK}_{b^*} := \mathsf{OK}_{b^*} \| (\mathsf{opk}, \mathsf{osk})$ **return** osk

Figure 6: Experiments for unlinkability and uniformly unbiasedness of F-SSS with Key-Exposure.

where $\mathsf{UNLNK}_{\mathsf{fs}_{\mathsf{w-ke}}}$ is defined in Figure 6. Note that, similar to prior works, we only consider the semi-honest server in the definition.

Definition 4.13 (Unbiasedness for fuzzy scalable tracking)**.** A F-SSS scheme (MKGen, OPKGen, OSKGen, Track, Sign, Vf, FTKGen, FTrack) is said to be *unbiased by senders* if there exists a negligible function negl, for all $\lambda \in \mathbb{N}$, and for all PPT adversaries $\mathcal{A}$ the following holds:

$$\Pr\left[\mathsf{UNIUBS}_{\mathsf{fs}}^{\mathcal{A}}(\lambda, \rho, N) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{UNIUBS}_{\mathsf{fs}}$ is defined in Figure 6 where $\mathsf{list}[i]$ denotes the i-th item of the list and $\mathcal{K}$ denotes the master public key space.

5 Generic Transformation To Get Security With Key Exposure

We provide our black-box compiler below to upgrade an $\mathsf{SS}_{\mathsf{w/o}}$ without key-exposure to an SS_{w} with key-exposure.

Suppose we have a digital signature scheme DS which is strongly unforgeable sEUFCMA. Then we have a black-box compiler leveraging SS to stronger version as shown in Figure 7. Basically, the compiler transforms any $\mathsf{SS}_{\mathsf{w/o}}$ with $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$ and $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$ security (without key-exposure) into an SS_{w} with $\mathsf{sEUFCMA}_{\mathsf{w-ke}}$ and $\mathsf{UNLNK}_{\mathsf{w-ke}}$ security (with key-exposure).

It is easy to see that correctness always holds as long as $\mathsf{SS}_{\mathsf{w/o}}$ and DS are correct. The security of unforgeability and unlinkability for SS_{w} are captured informally in the following theorem. The formal theorem and security proofs are deferred to Appendix E.

Theorem 5.1 (informal). The stealth signature SS_{w} constructed in Section 5 is secure in $\mathsf{sEUFCMA}_{\mathsf{w-ke}}$ and $\mathsf{UNLNK}_{\mathsf{w-ke}}$ experiments if $\mathsf{SS}_{\mathsf{w/o}}$ is $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$ secure, $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$ secure, and DS is sEUFCMA secure.

$\underline{\mathsf{MKGen}(\lambda)}$

$\textbf{return } \mathsf{SS_{w/o}.MKGen}(\lambda)$

$\underline{\mathsf{OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})}$

$(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{DS.Gen}(\lambda)$
$\mathsf{epk} \leftarrow \mathsf{SS_{w/o}.OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})$
$\textbf{return } \perp \textbf{ if } \mathsf{epk} = \perp$
$\sigma_1 \leftarrow \mathsf{SS_{w/o}.Sign}(\mathsf{epk}, \mathsf{vk})$
$\textbf{return } \mathsf{osk} := (\sigma_1, \mathsf{sk}, \mathsf{vk})$

$\underline{\mathsf{Track}(\mathsf{mtk}, \mathsf{opk}, \mathsf{tki})}$

$\textbf{return } \mathsf{SS_{w/o}.Track}(\mathsf{mtk}, \mathsf{opk}, \mathsf{tki})$

$\underline{\mathsf{OPKGen}(\mathsf{mpk})}$

$\textbf{return } \mathsf{SS_{w/o}.OPKGen}(\mathsf{mpk})$

$\underline{\mathsf{Sign}(\mathsf{osk}, m)}$

$\textbf{return } \perp \textbf{ if } \mathsf{osk} = \perp$
$(\sigma_1, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{osk}$
$\sigma_2 \leftarrow \mathsf{DS.Sign}(\mathsf{sk}, m || \sigma_1)$
$\textbf{return } \sigma := (\sigma_1, \sigma_2, \mathsf{vk})$

$\underline{\mathsf{Vf}(\mathsf{opk}, \sigma, m)}$

$(\sigma_1, \sigma_2, \mathsf{vk}) := \sigma$
$\textbf{if } \mathsf{SS_{w/o}.Vf}(\mathsf{opk}, \sigma_1, \mathsf{vk}) \wedge$
$\quad \mathsf{DS.Vf}(\mathsf{vk}, \sigma_2, m || \sigma_1)$
$\quad \textbf{return } 1$
$\textbf{else return } 0$

Figure 7: A generic transformation to lift $\mathsf{SS_{w/o}}$ to $\mathsf{SS_w}$.

$\underline{\mathsf{MKGen}(\lambda)}$

$\mathbf{A} \in R_q^{k \times \ell} \leftarrow \mathsf{Dil.ExpandA}(\mathsf{crs})$
$(\mathbf{s}_1, \mathbf{s}_2) \leftarrow\!\$\; S_\eta^\ell \times S_\eta^k$
$\mathbf{t} := \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$
$(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KEM.Gen}(\lambda)$
$\mathsf{mpk} := (\mathbf{t}, \mathsf{ek}),$
$\mathsf{msk} := (\mathbf{s}_1, \mathbf{s}_2, \mathsf{dk}, \mathbf{t})$
$\mathsf{mtk} := (\mathsf{dk}, \mathbf{t})$
$\textbf{return } (\mathsf{mpk}, \mathsf{msk}, \mathsf{mtk})$

$\underline{\mathsf{OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})}$

$(\mathbf{s}_1, \mathbf{s}_2, \mathsf{dk}, \mathbf{t}) := \mathsf{msk}$
$\textbf{if } \mathsf{false} \leftarrow \mathsf{Track}((\mathsf{dk}, \mathbf{t}), \mathsf{opk}, \mathsf{tki})$
$\quad \textbf{return } \perp$
$K \leftarrow \mathsf{KEM.Decaps}(\mathsf{dk}, \mathsf{tki})$
$(\mathbf{s}_1', \mathbf{s}_2') \leftarrow \mathsf{Dil.ExpandS}(K)$
$\textbf{return } \mathsf{osk} := (\mathbf{s}_1 + \mathbf{s}_1', \mathbf{s}_2 + \mathbf{s}_2')$

$\underline{\mathsf{Vf}(\mathsf{opk}, \sigma, m)}$

$\textbf{return } \mathsf{Dil.Vf}(\mathsf{opk}, \sigma, m)$

$\underline{\mathsf{OPKGen}(\mathsf{mpk})}$

$(\mathbf{t}, \mathsf{ek}) := \mathsf{mpk}$
$\mathbf{A} \leftarrow \mathsf{Dil.ExpandA}(\mathsf{crs})$
$(C, K) \leftarrow \mathsf{KEM.Encaps}(\mathsf{ek})$
$(\mathbf{s}_1', \mathbf{s}_2') \in S_\eta^\ell \times S_\eta^k \leftarrow \mathsf{Dil.ExpandS}(K)$
$\mathbf{t}' := \mathbf{t} + \mathbf{A}\mathbf{s}_1' + \mathbf{s}_2'$
$(\mathbf{t}_1', \cdot) \leftarrow \mathsf{Dil.power2Round}(\mathbf{t}', d)$
$\textbf{return } (\mathsf{opk} := \mathbf{t}_1', \mathsf{tki} := C)$

$\underline{\mathsf{Sign}(\mathsf{osk}, m)}$

$\textbf{return } \perp \textbf{ if } \mathsf{osk} = \perp$
$\textbf{return } \sigma := \mathsf{Dil.Sign}(\mathsf{osk}, m)$

$\underline{\mathsf{Track}(\mathsf{mtk}, \mathsf{opk}, \mathsf{tki})}$

$(\mathsf{dk}, \mathbf{t}) := \mathsf{mtk}$
$\mathbf{A} \leftarrow \mathsf{Dil.ExpandA}(\mathsf{crs})$
$K \leftarrow \mathsf{KEM.Decaps}(\mathsf{dk}, \mathsf{tki})$
$(\mathbf{s}_1', \mathbf{s}_2') \leftarrow \mathsf{Dil.ExpandS}(K)$
$\tilde{\mathbf{t}} := \mathbf{t} + \mathbf{A}\mathbf{s}_1' + \mathbf{s}_2'$
$(\tilde{\mathbf{t}}_1, \cdot) \leftarrow \mathsf{Dil.power2Round}(\tilde{\mathbf{t}}, d)$
$\textbf{return } \mathsf{opk} \stackrel{?}{=} \tilde{\mathbf{t}}_1$

Figure 8: Construction of SPIRIT with $\mathsf{EUFCMA_{w/o-ke}}$ and $\mathsf{UNLNK_{w/o-ke}}$ security

6 Spirit: Lattice based (Fuzzy) Stealth Signature

We first describe SPIRIT and later show we can make it fuzzy.

6.1 Lattice-based Stealth Signature

We use an ANOCCA-secure key exchange KEM (Kyber) [SAB+20] and an EUFCMA-secure signature (Dilithium) to construct an SS scheme with *existential unforgeability without key-exposure* and *unlinkability without key-exposure* in random oracle model. We require a common reference string $\mathsf{crs} \leftarrow\!\$\, \{0,1\}^{256}$, but for conciseness, we omit the explicit mention of crs in interfaces. We provide the detailed construction in Figure 8.

Intuitively, we use KEM to re-randomize the underlying master secret key msk to obtain osk each time and it needs to be actively anonymous which can be instantiated by Kyber with slight modification as shown in [GMP22]. Also, we only require Dilithium to be EUFCMA secure which gives us larger space to choose parameters. We recall Dilithium as follows.

Definition 6.1 (Dilithium [LDK+20])**.** Dilithium denoted by Dil is a post-quantum digital signature DS scheme based on the "Fiat-Shamir with Aborts" approach [Lyu09, Lyu12]. It is based on MLWE, MSIS and SelfTargetMSIS assumptions with ring $R_q := \mathbb{Z}_q[X]/(X^m + 1)$. Moreover, for secrets $\mathbf{s} \leftarrow\!\$\, S_\eta^\ell$, its each coefficient of the vector is an element of R_q with small coefficients of size at most η. In its optimized construction, there are some useful supporting algorithms which we described as follows:

- $\mathsf{ExpandA}(\mathsf{crs})$: The function maps a uniform seed crs to a matrix $\mathbf{A} \in R_q^{k\times\ell}$.
- $\mathsf{ExpandS}(K)$: The function used for generating the secret vectors in key generation, maps a seed K to $(\mathbf{s}_1, \mathbf{s}_2) \in S_\eta^\ell \times S_\eta^k$.
- $\mathsf{power2Round}(r, d)$: The function is the straightforward bit-wise way to break up an element $r := r_1 \cdot 2^d + r_0$ where $r_0 = r \mod 2^d$ and $r_1 = (r - r_0)/2^d$.
- $\mathsf{HighBits}_q(r, \alpha)$: The function select an α that is a divisor of $q-1$ and write $r = r_1 \cdot \alpha + r_0$ in the same way as before then returns r_1.
- $\mathsf{MakeHint}_q(z, r, \alpha)$: The function runs $r_1 \leftarrow \mathsf{HighBits}(r, \alpha)$ and $v_1 \leftarrow \mathsf{HighBits}(r + z, \alpha)$, then returns $r_1 \neq v_1$.

Correctness. Since

$$\mathbf{t}' = \mathbf{t} + \mathbf{A}\mathbf{s}_1' + \mathbf{s}_2' = \mathbf{A}(\mathbf{s}_1' + \mathbf{s}_1) + (\mathbf{s}_2 + \mathbf{s}_2'),$$

it is easy to see we have $1 - \mathsf{negl}(\lambda)$ correctness as long as underlying KEM and Dil have $1 - \mathsf{negl}(\lambda)$ correctness.

Notably, $\mathbf{s}_1' + \mathbf{s}_1$ and $\mathbf{s}_2 + \mathbf{s}_2'$ have approximately doubled norms, which results in doubled β in signatures. This will require additional iterations in the Sign algorithm, as the number of repetitions is roughly $2^{-256\cdot\beta(\frac{\ell}{\gamma_1}+\frac{k}{\gamma_2})}$, where $\gamma_1 \approx 2\gamma_2$ [LDK+20]. However, besides having doubled β, we can also increase γ_1 and γ_2 to $2\gamma_1$ and $2\gamma_2$, respectively. This adjustment slightly lowers the SelfTargetMSIS hardness but won't harm the running time. To see this, in Dil's proof, the reduction's advantage is mainly dominated by $\mathsf{MSIS}_{k,\ell,4\gamma_2}$ for sEUFCMA security, but $\mathsf{SelfTargetMSIS}_{k,\ell+1,2\gamma_2}$ for EUFCMA security. Without using the forking lemma (since it is not tight and not applicable in the quantum setting), the hardness of SelfTargetMSIS mainly comes from finding short vectors ($\|\cdot\|_\infty \leq 2\gamma_2$) $\mathbf{z}, \mathbf{u}'$ such that $\mathbf{Az} + \mathbf{u}' = \mathbf{t}'$ and amounts to the MSIS problem (refer to Section 6.2.1 and Appendix C.3 in [LDK+20] for details). Therefore, doubling γ_2 in our Spirit construction provides the reduction of $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$ with roughly the same advantage as that of sEUFCMA in Dil. We present the concrete security levels in Table 3.

$\mathsf{MKGen}(\lambda, n)$

$\mathsf{mpk}', \mathsf{mtk}', \mathsf{msk}' \leftarrow \mathsf{SS.MKGen}(\lambda)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{pRgv.Gen}(\lambda, n)$
$\mathsf{mpk} := (\mathsf{mpk}', \mathsf{pk}), \mathsf{msk} := \mathsf{msk}'$
$\mathsf{mtk} := (\mathsf{mtk}', \mathsf{sk})$
return $(\mathsf{mpk}, \mathsf{mtk}, \mathsf{msk})$

$\mathsf{FTKGen}(\mathsf{mtk}, \rho)$

parse $(\mathsf{mtk}', \mathsf{sk}) := \mathsf{mtk}$
parse $(\mathbf{s}_1, \ldots, \mathbf{s}_n) := \mathsf{sk}$
$t \leftarrow \log_2(\frac{1}{\rho})$
return $\mathsf{ftk} := (\mathbf{s}_1, \ldots, \mathbf{s}_t)$

$\mathsf{Sign}(\mathsf{osk}, m)$

return $\mathsf{SS.Sign}(\mathsf{osk}, m)$

$\mathsf{Vf}(\mathsf{opk}, \sigma, m)$

return $\mathsf{SS.Vf}(\mathsf{opk}, \sigma, m)$

$\mathsf{OPKGen}(\mathsf{mpk})$

parse $(\mathsf{mpk}', \mathsf{pk}) := \mathsf{mpk}$
$\mathsf{ftki} \leftarrow \mathsf{pRgv.Enc}(\mathsf{pk}, \mathbf{1})$
return $(\mathsf{SS.OPKGen}(\mathsf{mpk}'), \mathsf{ftki})$

$\mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki})$

$[m_1, \ldots, m_{|\mathsf{ftk}|}] \leftarrow \mathsf{pRgv.Dec}(\mathsf{ftk}, \mathsf{ftki})$
$b := \bigwedge_{i=1}^{|\mathsf{ftk}|} m_i$
return $b \stackrel{?}{=} 1$

$\mathsf{Track}(\mathsf{mtk}, \mathsf{opk}, \mathsf{tki})$

parse $(\mathsf{mtk}', \mathsf{sk}) := \mathsf{mtk}$
return $\mathsf{SS.Track}(\mathsf{mtk}', \mathsf{opk}, \mathsf{tki})$

$\mathsf{OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})$

return $\mathsf{SS.OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})$

Figure 9: Post-quantum FMD fuzzy tracking

Security Analysis. We prove the construction of Spirit in Figure 8 is existential unforgeable and unlinkable *without* key exposure, and is secure in $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$ and $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$ experiment, respectively. For security of $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$, we prove this in two steps. First, we show it is unforgeable without key exposure under no-message attacks (NMA), i.e., the adversary cannot query $\mathsf{Sign}\mathcal{O}(\cdot)$, and we refer the corresponding experiment to $\mathsf{UFNMA}_{\mathsf{w/o-ke}}$; Next, we show a reduction from $\mathsf{UFNMA}_{\mathsf{w/o-ke}}$ to $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$. Since Dil does not rely on the lower parts of public key $\mathbf{t}_0$ to be secret, so for simplicity, we assume the one-time public key opk is $\mathbf{t}'$ instead of $\mathbf{t}'_1$. Also, we assume $\mathsf{crs} := \mathbf{A}$ directly and is publicly known.

Lemma 6.1 (informal). Spirit in Figure 8 is unforgeable without key exposure under no-message attacks if $\mathsf{SelfTargetMSIS}$ and MLWE assumptions hold.

Then we have the following theorems to show the construction is unforgeable and unlinkable. The formal statement and analysis of the above lemma and the following theorem is deferred to Appendix F.

Theorem 6.2 (informal). Spirit in Figure 8 is existential unforgeable and unlinkable without key exposures if it is $\mathsf{UFNMA}_{\mathsf{w/o-ke}}$ and the KEM used is ANOCCA secure.

6.2 Lattice-based Fuzzy Stealth Signature

We provide a lattice-based construction for fuzzy tracking in standard model. Basically, it is packed Regev encryption (denoted as pRgv) with ciphertext compression [BDGM19]. And this gives us the first post-quantumly ambiguous encryption without relying on random oracles.

Packed Regev (compressed). For a more detailed description and analysis, please refer to Appendix G. In short terms, the packed Regev scheme pRgv is a lattice-based linearly homomorphic encryption that has an additional property that allows for *ciphertext compression*. This unique feature enables the representation of a ciphertext encrypting n bits with a size of only $n + O(\lambda)$ bits. This reduced ciphertext size contributes to the both asymptotic and concrete efficiency of the scheme.

Since INDCPA and IKCPA security (recalled in Appendix C) of pRgv are discussed in prior works already, we focus on its ambiguous security and we show it is actually Uniformly-Ambiguous (recalled in Appendix C) with super-poly noise-modulus ratio. The formal statement and proof of the lemma below is deferred to Appendix G.

Lemma 6.3. Packed Regev encryption pRgv with ciphertext compression shown in Figure 16 satisfies Definition C.8 and is uniformly-ambiguous UNIAMB-secure when $\frac{4Bn}{q}$ is $\mathsf{negl}(\lambda)$.

The modulus. To argue uniformly-ambiguous security, we need super-polynomial noise-to-modulus ratio (e.g., 60-bit modulus in our case) which is usually assumed in homomorphic encryption related works. This is a somewhat stronger assumption since it assumes the lattice problem BDD or GapSVP is hard even with super-polynomial approximation factor [Reg05].

Construction. We then provide a lattice-based fuzzy stealth signature in Figure 9, which is composed of a standard stealth signature SS and a compressed packed Regev encryption pRgv shown above. Basically, it use the same framework as FMD_1 presented in [BLMG21].

Correctness. We provide the correctness analysis in Appendix G.

Now we consider the false-positive rate ρ when using different fuzzy tracking key. Since $\mathbf{c}_1$ looks uniformly random due to LWE assumption, $\mathbf{s}_i^T\mathbf{c}_1$ is uniformly random over $\mathbb{Z}_q$ by Leftover Hash Lemma as inner product is a strong randomness extractor[5]. This implies $\lceil \mathbf{s}_i^T\mathbf{c}_1 + z\rfloor$ is uniformly random over $\{0,1\}$ and FTrack returns true with probability $2^{-t} = \rho$.

Security Analysis. Formal statement and corresponding proof of the following theorem are deferred to Appendix G.

Theorem 6.4 (informal). The fuzzy stealth signature constructed in Figure 9 is *unlinkable with key-exposure and fuzzy tracking* if the underlying stealth signature is $\mathsf{UNLNK}_{\mathsf{w-ke}}$ and pRgv is UNIAMB and IKCPA secure.

We also provide an approach to extend it to finer false-positive rate as shown in Appendix G.

6.3 Scalable Lattice-based Fuzzy Tracking

As discussed in Section 2.4, we limit the user's ability to choose false-positive rate and provide a new framework of fuzzy tracking which is substantially more scalable than prior works[BLMG21, MSS+21, LT21]. Please refer to Section 4.4 for functionality and security definitions.

Construction. We describe the detailed construction in Figure 10, where $\{0,1\}^{2m} \leftarrow \mathsf{H}(k \in \{0,1\}^\lambda, i \in [t])$ is a hash function with the seed k and $\mathsf{H}_n : \{0,1\}^{|\mathsf{mpk}|} \mapsto \{0,1\}^n$ is another hash function mapping mpk to a hint which is used to locate mpk's mailbox in server's storage. Since it is based on Module-LWE assumption, R_q denotes the ring $\mathbb{Z}_q[X]/(X^m+1)$, and $\mathsf{encode}_{R_q} : \{0,1\}^m \mapsto \mathbb{Z}_q[X]/(X^m+1)$ is a function mapping binary strings to the ring elements with binary coefficients; Similarly, decode_{R_q} is the reverse operation to map back to binary string. Basically, it is a variant of the underlying INDCPA encryption of Kyber with non-prime modulus because we need $\mathbb{Z}_2$ to be a subgroup of $\mathbb{Z}_q$ in correctness and security analysis. Though we lose the advantage of NTT multiplications, we can still mitigate this by using Karatsuba and Toom-Cook algorithms.

Correctness. It is clear to see that the targeted mpk must have $\mathsf{hint} := \mathsf{hint}^i = \mathsf{H}_n(\mathsf{mpk})$ appears in list with probability 1: For the targeted index $i \in [t]$, we have $\mathbf{c}_1^i = \mathbf{A}^T\mathbf{r} + \mathbf{e}_1$ which is the same as standard ciphertext header. The decryption will output hint directly as long as $q > 4B$. Now we focus on the other case where $\mathsf{mpk}' \neq \mathsf{mpk}$. Firstly, considering $\mathsf{hint}_j \in \mathsf{list}$, it is decrypted as

$$\lceil \mathbf{s}^T\mathbf{c}_1^j - c_2\rfloor_2 \oplus y^j = \lceil e' + \frac{q}{2}(w + s_1(x^i - x^j) + y^i)\rfloor_2 \oplus y^j,$$

[5] We only use it for correctness (or fuzziness), not for security.

$\mathsf{MKGen}(\lambda)$

return $\mathsf{SS.MKGen}(\lambda)$

$\mathsf{FTKGen}(\rho, N)$

$n := \lceil \log_2 N \rceil$
$t := \rho \cdot 2^n$
$\mathbf{A} \in R_q^{\ell \times \ell} \leftarrow\!\$\ \mathsf{crs}$
$(\mathbf{s}, \mathbf{e}) \leftarrow\!\$\ (B_\eta^\ell)^2$
$\mathbf{b} := \mathbf{A}\mathbf{s} + \mathbf{e}$
return $\mathsf{ftk} := (\mathbf{s}, t), \mathsf{fpk} := (\mathbf{b}, \mathsf{H}_n, t)$

$\mathsf{FTrack}(\mathsf{ftk}, \mathsf{ftki})$

parse $(\mathbf{s}, t) := \mathsf{ftk}, (\mathbf{c}_1, c_2, \delta) := \mathsf{ftki}$
$\forall i \in [t]:$
  $(\mathbf{x}^i, \mathbf{y}^i) \leftarrow \mathsf{H}(\delta, i)$
  $x^i, y^i \in R_q \leftarrow \mathsf{encode}_{R_q}(\mathbf{x}^i, \mathbf{y}^i)$
  $\mathbf{c}_1^i := \mathbf{c}_1 - \frac{q}{2} \cdot (\begin{bmatrix} x^i \\ 0 \end{bmatrix})$
  $w^i := \lceil \mathbf{s}^T \mathbf{c}_1^i - c_2 \rfloor_2 \oplus y^i$
$\mathsf{hint}^i := \mathsf{decode}_{R_q}(w^i)[: n]$
return $\mathsf{list} := \{\mathsf{hint}^1, \ldots, \mathsf{hint}^t\}$

$\mathsf{Sign}(\mathsf{osk}, m)$

return $\mathsf{SS.Sign}(\mathsf{osk}, m)$

$\mathsf{OPKGen}(\mathsf{mpk}, \mathsf{fpk})$

parse $(\mathbf{b} \in R_q^\ell, \mathsf{H}_n, t) := \mathsf{fpk}$
$\mathsf{hint} \in \{0,1\}^n \leftarrow \mathsf{H}_n(\mathsf{mpk})$
$\mathbf{z} \leftarrow\!\$\ \{0,1\}^{m-n}$
$\mathbf{w}^T := [\mathsf{hint}^T \| \mathbf{z}^T]$
$i \leftarrow\!\$\ [t]$
$\delta \leftarrow\!\$\ \{0,1\}^\lambda$
$(\mathbf{r}, \mathbf{e}_1) \leftarrow\!\$\ (B_\eta^\ell)^2, e_2 \leftarrow\!\$\ B_\eta$
$(\mathbf{x}, \mathbf{y} \in \{0,1\}^m) \leftarrow \mathsf{H}(\delta, i)$
$x, y, w \in R_q \leftarrow \mathsf{encode}_{R_q}(\mathbf{x}, \mathbf{y}, \mathbf{w})$
$\mathbf{c}_1 := \mathbf{A}^T \mathbf{r} + \mathbf{e}_1 + \frac{q}{2} \cdot \begin{bmatrix} x \\ 0 \end{bmatrix}$
$c_2 := \mathbf{b}^T \mathbf{r} + e_2 + \frac{q}{2} \cdot (w + y)$
$\mathsf{ftki} := (\mathbf{c}_1, c_2, \delta)$
return $\mathsf{SS.OPKGen}(\mathsf{mpk}), \mathsf{ftki}$

$\mathsf{OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})$

return $\mathsf{SS.OSKGen}(\mathsf{msk}, \mathsf{opk}, \mathsf{tki})$

$\mathsf{Track}(\mathsf{mtk}, \mathsf{opk}, \mathsf{tki})$

return $\mathsf{SS.Track}(\mathsf{mtk}, \mathsf{opk}, \mathsf{tki})$

$\mathsf{Vf}(\mathsf{opk}, \sigma, m)$

return $\mathsf{SS.Vf}(\mathsf{opk}, \sigma, m)$

Figure 10: Scalable lattice-based fuzzy tracking

where s_1 is the first ring element of $\mathbf{s}$. hint_j is uniformly random over $\{0,1\}^n$ after rounding $\lceil \cdot \rfloor_2$ as $y^j \oplus y^i$ are outputs of the random oracle H. Then, for any $\mathsf{mpk}' \neq \mathsf{mpk}$, $\Pr\big[\mathsf{H}_n(\mathsf{mpk}') = \mathsf{hint}_j\big] = \frac{1}{2^n}$ since H_n is a random oracle, and

$$\begin{aligned} \Pr\big[\mathsf{H}_n(\mathsf{mpk}') \in \mathsf{list}\big] &= \sum_{j=1}^{t} \Pr\big[\mathsf{H}_n(\mathsf{mpk}') = \mathsf{hint}_j\big] \\ &= \frac{t}{2^n} = \rho. \end{aligned}$$

Security Analysis. The formal theorem statements and proof of the following theorems are deferred to Appendix H.

Theorem 6.5 (informal). The fuzzy scalable stealth signature constructed in Figure 10 is unlinkable with key-exposure and fuzzy tracking if the underlying stealth signature is $\mathsf{UNLNK}_{\mathsf{w-ke}}$ and MLWE holds. It is also *unbiased* and satisfying $\mathsf{UNIUBS}_{\mathsf{fs}}$ defined in Definition 4.13 if $n \leq \frac{m}{2}$ where m is a power of 2 and B_η is a centered binomial distribution.

7 Conclusion

In this work, we have presented a novel and practical approach to address post-quantum secure stealth addresses. Along the way, we demonstrate its potential applications, such as privacy-preserving payments and passwordless authentication schemes like FIDO. We have also introduced a generic method to transform standard security without key-exposure resistance into a robust security solution capable of withstanding key-exposure attacks. Additionally, we have explored post-quantum fuzzy message detection (fuzzy tracking) and proposed two potential constructions. Future work and open problems include exploring the integration of our approach into RingCT-like frameworks [EZS+19, ESZ22] and investigating methods to reduce the signature size based on our current results.

Acknowledgement

Nico Döttling is funded by the European Union (ERC, LACONIC, 101041207). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

[ABB10] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 553–572, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany.

[ADE+20] Nabil Alkeilani Alkadri, Poulami Das, Andreas Erwig, Sebastian Faust, Juliane Krämer, Siavash Riahi, and Patrick Struck. Deterministic wallets in a quantum world. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020: 27th Conference on Computer and Communications Security*, pages 1017–1031, Virtual Event, USA, November 9–13, 2020. ACM Press.

[AFLT12] Michel Abdalla, Pierre-Alain Fouque, Vadim Lyubashevsky, and Mehdi Tibouchi. Tightly-secure signatures from lossy identification schemes. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 572–590, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany.

[Ajt98] Miklós Ajtai. The shortest vector problem in L2 is NP-hard for randomized reductions (extended abstract). In *30th Annual ACM Symposium on Theory of Computing*, pages 10–19, Dallas, TX, USA, May 23–26, 1998. ACM Press.

[All22] FIDO Alliance. White Paper: Multi-Device FIDO credentials, Mar 2022.

[AMKM21] Lukas Aumayr, Pedro Moreno-Sanchez, Aniket Kate, and Matteo Maffei. Blitz: Secure multi-hop payments without two-phase commits. In Michael Bailey and Rachel Greenstadt, editors, *USENIX Security 2021: 30th USENIX Security Symposium*, pages 4043–4060. USENIX Association, August 11–13, 2021.

[BBCW21] Manuel Barbosa, Alexandra Boldyreva, Shan Chen, and Bogdan Warinschi. Provable security analysis of FIDO2. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 125–156, Virtual Event, August 16–20, 2021. Springer, Heidelberg, Germany.

[BBDP01] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582, Gold Coast, Australia, December 9–13, 2001. Springer, Heidelberg, Germany.

[BDGM19] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019: 17th Theory of Cryptography Conference, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 407–437, Nuremberg, Germany, December 1–5, 2019. Springer, Heidelberg, Germany.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany.

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS 2012: 3rd Innovations in Theoretical Computer Science*, pages 309–325, Cambridge, MA, USA, January 8–10, 2012. Association for Computing Machinery.

[BLMG21] Gabrielle Beck, Julia Len, Ian Miers, and Matthew Green. Fuzzy message detection. In Giovanni Vigna and Elaine Shi, editors, *ACM CCS 2021: 28th Conference on Computer and Communications Security*, pages 1507–1528, Virtual Event, Republic of Korea, November 15–19, 2021. ACM Press.

[CM17] Nicolas T Courtois and Rebekah Mercer. Stealth address and key management techniques in blockchain systems. *ICISSP*, 2017:559–566, 2017.

[DG17] Nico Döttling and Sanjam Garg. Identity-based encryption from the Diffie-Hellman assumption. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 537–569, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

[dona] Cryptocurrency solutions for institutional philanthropy.

[donb] How to donate crypto.

[donc] Why donate bitcoin, ethereum, nfts and other cryptocurrencies to charity.

[eco] Cryptocurrency and e-commerce.

[ega] Cryptocurrency and online gaming.

[ESZ22] Muhammed F. Esgin, Ron Steinfeld, and Raymond K. Zhao. MatRiCT$^+$: More efficient post-quantum private blockchain payments. In *2022 IEEE Symposium on Security and Privacy*, pages 1281–1298, San Francisco, CA, USA, May 22–26, 2022. IEEE Computer Society Press.

[EZS+19] Muhammed F. Esgin, Raymond K. Zhao, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. MatRiCT: Efficient, scalable and post-quantum blockchain confidential transactions protocol. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 567–584, London, UK, November 11–15, 2019. ACM Press.

[FGK+20] Nick Frymann, Daniel Gardham, Franziskus Kiefer, Emil Lundberg, Mark Manulis, and Dain Nilsson. Asynchronous remote key generation: An analysis of yubico's proposal for W3C WebAuthn. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020: 27th Conference on Computer and Communications Security*, pages 939–954, Virtual Event, USA, November 9–13, 2020. ACM Press.

[fre] Digital currency donations for freedom convoy evading seizure by authorities.

[GMP22] Paul Grubbs, Varun Maram, and Kenneth G. Paterson. Anonymous, robust post-quantum public key encryption. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 402–432, Trondheim, Norway, May 30 – June 3, 2022. Springer, Heidelberg, Germany.

[HLW22] Lucjan Hanzlik, Julian Loss, and Benedikt Wagner. Token meets wallet: Formalizing privacy and revocation for FIDO2. Cryptology ePrint Archive, Report 2022/084, 2022. https://eprint.iacr.org/2022/084.

[ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstracts). In *21st Annual ACM Symposium on Theory of Computing*, pages 12–24, Seattle, WA, USA, May 15–17, 1989. ACM Press.

[imp] https://github.com/sihangpu/SPIRIT.

[Ing56] A. W. Ingleton. The rank of circulant matrices. *Journal of the London Mathematical Society*, s1-31(4):445–460, 1956.

[KLS18] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 552–586, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.

[LDK+20] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.

[LLN+20] Wenling Liu, Zhen Liu, Khoa Nguyen, Guomin Yang, and Yu Yu. A lattice-based key-insulated and privacy-preserving signature scheme with publicly derived public key. In Liqun Chen, Ninghui Li, Kaitai Liang, and Steve A. Schneider, editors, *ESORICS 2020: 25th European Symposium on Research in Computer Security, Part II*, volume 12309 of *Lecture Notes in Computer Science*, pages 357–377, Guildford, UK, September 14–18, 2020. Springer, Heidelberg, Germany.

[LRR+19] Russell W. F. Lai, Viktoria Ronge, Tim Ruffing, Dominique Schröder, Sri Aravinda Krishnan Thyagarajan, and Jiafan Wang. Omniring: Scaling private payments without trusted setup. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 31–48, London, UK, November 11–15, 2019. ACM Press.

[LT21] Zeyu Liu and Eran Tromer. Oblivious message retrieval. Cryptology ePrint Archive, Report 2021/1256, 2021. https://eprint.iacr.org/2021/1256.

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 598–616, Tokyo, Japan, December 6–10, 2009. Springer, Heidelberg, Germany.

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 738–755, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany.

[LYW+19] Zhen Liu, Guomin Yang, Duncan S. Wong, Khoa Nguyen, and Huaxiong Wang. Key-Insulated and Privacy-Preserving signature scheme with publicly derived public key. In *2019 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 215–230, 2019.

[Mer90] Ralph C. Merkle. A certified digital signature. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238, Santa Barbara, CA, USA, August 20–24, 1990. Springer, Heidelberg, Germany.

[MSH+17] Malte Möser, Kyle Soska, Ethan Heilman, Kevin Lee, Henry Heffan, Shashvat Srivastava, Kyle Hogan, Jason Hennessey, Andrew Miller, Arvind Narayanan, et al. An empirical analysis of traceability in the monero blockchain. *arXiv preprint arXiv:1704.04299*, 2017.

[MSS+21] Varun Madathil, Alessandra Scafuro, István András Seres, Omer Shlomovits, and Denis Varlakov. Private signaling. Cryptology ePrint Archive, Report 2021/853, 2021. https://eprint.iacr.org/2021/853.

[NMRL16] Shen Noether, Adam Mackenzie, and the Monero Research Lab. Ring confidential transactions. *Ledger*, 1:1–18, Dec. 2016.

[OKH13] Micha Ober, Stefan Katzenbeisser, and Kay Hamacher. Structure and anonymity of the bitcoin transaction graph. *Future internet*, 5(2):237–250, 2013.

[PFH+22] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2022. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

[PRV12] Periklis A. Papakonstantinou, Charles W. Rackoff, and Yevgeniy Vahlis. How powerful are the DDH hard groups? Cryptology ePrint Archive, Report 2012/653, 2012. https://eprint.iacr.org/2012/653.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 84–93, Baltimore, MA, USA, May 22–24, 2005. ACM Press.

[RH13] Fergal Reid and Martin Harrigan. An analysis of anonymity in the bitcoin system. In *Security and privacy in social networks*, pages 197–223. Springer, 2013.

[RS13] Dorit Ron and Adi Shamir. Quantitative analysis of the full bitcoin transaction graph. In *International Conference on Financial Cryptography and Data Security*, pages 6–24. Springer, 2013.

[SAB+20] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, and Damien Stehlé. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.

[SAB+22] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, Damien Stehlé, and Jintai Ding. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2022. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

[SGS21] Gili Schul-Ganz and Gil Segev. Generic-group identity-based encryption: A tight impossibility result. Cryptology ePrint Archive, Report 2021/745, 2021. https://eprint.iacr.org/2021/745.

[SO13] Marc Santamaria Ortega. The bitcoin transaction graph anonymity. 2013.

[SPB21] István András Seres, Balázs Pejó, and Péter Burcsi. The effect of false positives: Why fuzzy message detection leads to fuzzy privacy guarantees? Cryptology ePrint Archive, Report 2021/1180, 2021. https://eprint.iacr.org/2021/1180.

[ste] Untraceable transactions which can contain a secure message are inevitable. 2011.

[Tod] Peter Todd. Stealth addresses, 2014.

[umb] Umbra: Privacy preserving stealth payments.

[use] How many people own and use bitcoin?

[vS] Nicolas van Saberhagen. Cryponote v 2.0. 2013.

[YZ21] Takashi Yamakawa and Mark Zhandry. Classical vs quantum random oracles. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part II*, volume 12697 of *Lecture Notes in Computer Science*, pages 568–597, Zagreb, Croatia, October 17–21, 2021. Springer, Heidelberg, Germany.

[ZMS+21] Raymond K. Zhao, Sarah McCarthy, Ron Steinfeld, Amin Sakzad, and Máire O'Neill. Quantum-safe HIBE: does it cost a latte? Cryptology ePrint Archive, Report 2021/222, 2021. https://eprint.iacr.org/2021/222.

A Discussions about Quantum Random Oracles

Since our goal is a *practically efficient* construction, we mainly focus on post-quantum security in the classical random oracle setting which is in the same spirit as in related works such as [LLN+20, EZS+19, ESZ22], etc. However, our protocols are highly likely to be secure even in the QROM setting. In more detail, for unforgeability, we can either follow the same strategy in Section 4.5 of [KLS18] to argue the EUF-CMA security in the QROM setting or apply the lifting theorem 1.1 and 1.2 from [YZ21] to establish the reduction from EUF-NMA in classical random oracle model to EUF-CMA in the QROM setting; for unlinkability, according to [GMP22], our adapted Kyber is already ANO-CCA secure in the QROM setting, and we only program the random oracle in a non-adaptive way as in the unforgeability game, thus the security in QROM is supposed to be preserved. We will add this discussion to the paper.

B Performance Analysis

We present the performance result in Table 3 and Table 4.

Implementation. We implement the Spirit, post-quantum FMD, and scalable fuzzy tracking schemes in C, and the open-source code of our proof-of-concept implementation can be found at [imp]. Specifically, we choose the anonymized variant of Kyber [SAB+22] to instantiate the KEM for building Spirit: We replace the original FO transform of Kyber with the one suggested in [GMP22], which makes Kyber ANOCCA-secure.

To instantiate the generic transformation in Section 5 to make Spirit secure against key-exposure attacks, we consider using Dilithium or Falcon [PFH+22] as the additional digital signature. Combining Spirit with Dilithium results in better efficiency but a slightly larger signature size; on the other hand, combining Spirit with Falcon leads to the most compact signatures but significantly longer key generation time.

For Spirit in Section 6.1, similar to Dilithium, we denote the scheme with three security levels as $\textsc{Spirit}_2$, $\textsc{Spirit}_3$, and $\textsc{Spirit}_5$. Parameters are the same as Dilithium's, except that our $\beta, \gamma_1, \gamma_2$ are doubled.

For post-quantum FMD in Section 6.2, to achieve 104-bit computational security and 40-bit statistical security, we choose $q = 2^{60}$, $\ell = 2304$, and $\chi = B_\eta$ is a binomial distribution with parameter $\eta = 3$.

For Scalable Fuzzy Tracking in Section 6.3, to attain 115-bit security and negligible failure probability, we choose $q = 4096$; other parameters are the same as Kyber512, specifically, we have $m = 256, \eta = 3, \ell = 2$.

Environment. We run the implementation on a standard laptop: Macbook Air (M1 2020) with 8GB RAM and a 2.1 GHz CPU (Turbo 3.2 GHz). It is important to note that our implementation is based on the reference implementation of Dilithium, Kyber, and Falcon, without using AES or AVX optimization. We perform each test 10,000 times to calculate the average running time. For post-quantum FMD, we run tests 100 times to obtain the average running time.

Experimental results demonstrate that Falcon512+$\textsc{Spirit}_2$ provides the smallest signature size (4.09 KB) for security against key-exposures with a decent hardness level (114-bit security). Additionally, Scalable Fuzzy Tracking offers the smallest communication cost (800 Bytes) and server's computational overhead (3.4 ms) for millions of clients.

Prior Works. We provide comparison tables with prior works in Table 1 and Table 2.

In Table 1, we compare our group-based stealth signature (Appendix D), $\textsc{Spirit}_2$ (Section 6.1), $\textsc{Spirit}_2$ +Dilithium2, and $\textsc{Spirit}_2$ +Falcon512 with previous works. It is important to note

Table 3: Performance Result of Constructions in Section 5 and Section 6.1

Scheme[1]	w/KE	sec	sec_q[2]	opk	Signature	MKGen	OPKGen	Track	OSKGen	Sign	Vf
$\textsc{Spirit}_2$	○	114	104	**2.08** KB	**2.54** KB	0.068 ms	0.074 ms	0.076 ms	**0.078** ms	0.208 ms	**0.053** ms
$\textsc{Spirit}_3$	○	171	155	3.04 KB	3.45 KB	0.131 ms	0.137 ms	0.136 ms	0.138 ms	0.377 ms	0.089 ms
$\textsc{Spirit}_5$	○	245	223	4.16 KB	4.81 KB	0.191 ms	0.198 ms	0.202 ms	0.215 ms	0.443 ms	0.145 ms
Dilithium2+$\textsc{Spirit}_2$	•	114	104	**2.08** KB	6.40 KB	0.070 ms	0.076 ms	0.078 ms	**0.358** ms	0.222 ms	0.114 ms
Dilithium3+$\textsc{Spirit}_3$	•	171	155	3.04 KB	8.85 KB	0.129 ms	0.136 ms	0.132 ms	0.597 ms	0.369 ms	0.182 ms
Dilithium5+$\textsc{Spirit}_5$	•	245	223	4.16 KB	12.2 KB	0.186 ms	0.193 ms	0.197 ms	0.762 ms	0.428 ms	0.291 ms
Falcon512+$\textsc{Spirit}_2$	•	114	104	**2.08** KB	**4.09** KB	0.069 ms	0.074 ms	0.075 ms	5.458 ms	0.226 ms	**0.074** ms
Falcon1024+$\textsc{Spirit}_3$	•	171	155	3.04 KB	6.51 KB	0.133 ms	0.133 ms	0.133 ms	17.7 ms	0.444 ms	0.130 ms
Falcon1024+$\textsc{Spirit}_5$	•	245	223	4.16 KB	7.88 KB	0.194 ms	0.198 ms	0.201 ms	17.5 ms	0.441 ms	0.185 ms

[1] $\textsc{Spirit}_2$ is based on Dilithium2 and anonymized Kyber512, $\textsc{Spirit}_3$ on Dilithium3 and Kyber768, and $\textsc{Spirit}_5$ on Dilithium5 and Kyber1024. Falcon+Spirit indicates the use of an additional Falcon as DS in the generic transformation, while Dilithium+Spirit signifies the use of an additional Dilithium as DS in the generic transformation.
[2] sec_q represents the hardness of Quantum Core-SVP, while sec denotes the hardness of Classical Core-SVP.

Table 4: Performance Result of Constructions in Section 6.2 and Section 6.3

Scheme	sec	sec_q	N Clients	ρ	Public Key	Fuzzy Tracking Info	Setup Time	OPKGen[1]	FTrack[2]
Post-quantum FMD	104	94	2^{20}	2^{-10}	345.6 KB	17.2 KB	108.8 ms	75.13 ms	11.74 sec
Post-quantum FMD	104	94	2^{30}	2^{-15}	518.4 KB	17.2 KB	124.3 ms	74.64 ms	4.772 hour
Scalable Fuzzy Tracking	115	104	2^{20}	2^{-10}	**800** B	**800** B	0.011 ms	0.0148 ms	**3.424** ms
Scalable Fuzzy Tracking	115	104	2^{30}	2^{-15}	800 B	800 B	0.011 ms	0.0149 ms	108.77 ms

[1] Only consider the fuzzy part, i.e., the time to generate the fuzzy tracking information, ftki.
[2] The server's running time for each incoming ftki. For Post-quantum FMD, we calculate the time to run FTrack for all of clients (recipients).

that[LLN+20] is a theoretical work without concrete parameters. We estimate the parameters based on the information provided in the paper. For a more in-depth analysis of the estimated parameters, please refer to the original text.

If we aim to improve their work with the recent advancements in NTRU, it is worth mentioning that the techniques used in [LLN+20] are derived from [ABB10], which implies HIBE. Combining it with NTRU could potentially enhance its efficiency. However, it is likely to have parameters similar to the state-of-the-art NTRU-based HIBE [ZMS+21]. Thus, we estimate the parameters here based on [ZMS+21] for 80-bit security, as they only provide two levels of security (80-bit or 160-bit).

In Table 2, we compare our Post-quantum FMD (Section 6.2) and Scalable Fuzzy Tracking (Section 6.3) with previous works on message detection or retrieval. All these works assume a semi-honest server, except for Π_{TEE}, which also considers a malicious server. Note that for security, ρ needs to be as large as $\frac{1}{\sqrt{N}}$ as calculated in [SPB21]. Additionally, some prior works consider fuzzy schemes ([BLMG21] and ours) as ρM-anonymity, where M is the total number of messages. However, this is not accurate due to statistical attacks as shown in [SPB21]: even with only one message ($M = 1$), some extent of anonymity is maintained if N is large.

Regarding the server's workload, we compare the results for a single server with a single thread, as all works (except for Π_{GC}) support distributed servers or parallelized threads. [BLMG21] requires running their test functionality for each recipient's detection key for each incoming message. Other schemes with full privacy inherently demand $O(N)$ work from the server; otherwise, information leakage will occur.

Latency per message is dominated by the server's computational time. Assuming there are $N = 2^{20}$ clients (a reasonable assumption for cryptocurrencies [use]) and setting the false-positive rate $\rho = 2^{-10}$ for [BLMG21] and ours. The numbers of other works are taken directly from their papers. With an assumption of $10 - 20$ messages per second (e.g., Bitcoin or Ethereum), only our scheme is practical with many users.

To compute each recipient's computational time, we assume a total of $M = 500,000$ messages

C Additional Preliminaries

C.1 Assumptions

Definition C.1 (Cyclotomic Polynomial)**.** We denote by R the ring $\mathbb{Z}[X]/(X^n + 1)$ and by R_q the ring $\mathbb{Z}_q[X]/(X^m + 1)$, where $m = 2^{m'-1}$ such that $X^m + 1$ is the $2m'$-th cyclotomic polynomial $\Phi_{2m'}(X)$. Moreover, we have

$$\prod_{d|m} \Phi_d(X) = X^m - 1.$$

Definition C.2 (Learning with Errors (LWE)[Reg05])**.** For a vector $\mathbf{s} \in \mathbb{Z}_q^n$ called the secret, the LWE distribution $A_{\mathbf{s},\chi}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ is sampled by choosing $a \in \mathbb{Z}_q^n$ uniformly at random, choosing $e \leftarrow\!\$\, \chi$, and outputting $(\mathbf{a}, b = \langle \mathbf{a}, \mathbf{s} \rangle + e \mod q)$. Moreover, decisional-$\mathsf{LWE}_{n,m,q,\chi}$ is

$$\begin{aligned}\mathsf{Adv}^{\mathsf{lwe}}_{n,m,q,\chi}(\mathcal{A}) = \big| \Pr[&b = 1 | \mathbf{A} \leftarrow\!\$\, \mathbb{Z}_q^{m\times n}, \mathbf{t} \leftarrow\!\$\, \mathbb{Z}_q^m; \\ &b \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{t})] \\ - \Pr[b = 1 | \mathbf{A} \leftarrow \mathbb{Z}_q^{m\times n}, &\mathbf{s} \leftarrow\!\$\, \mathbb{Z}_q^n, \mathbf{e} \leftarrow\!\$\, \chi^m; \\ &b \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e})] \big|.\end{aligned}$$

Definition C.3 (Module Learning With Errors MLWE [BGV12])**.** For integers m, k, and a probability distribution $D : R_q \to [0, 1]$, we say that the advantage of algorithm $\mathcal{A}$ in solving the decisional $\mathsf{MLWE}_{m,k,D}$ problem over the ring R_q is

$$\begin{aligned}\mathsf{Adv}^{\mathsf{mlwe}}_{m,k,D}(\mathcal{A}) = \big| \Pr[&b = 1 | \mathbf{A} \leftarrow R_q^{m\times k}, \mathbf{t} \leftarrow R_q^m; \\ &b \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{t})] \\ - \Pr[b = 1 | \mathbf{A} \leftarrow R_q^{m\times k}, &\mathbf{s}_1 \leftarrow D^k, \mathbf{s}_2 \leftarrow D^m; \\ &b \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2)] \big|\end{aligned}$$

Definition C.4 (Module Short Integer Solution MSIS[Ajt98])**.**

$$\begin{aligned}\mathsf{Adv}^{\mathsf{msis}}_{m,k,\gamma}(\mathcal{A}) = \Pr\Big[&0 < \|\mathbf{y}\|_\infty < \gamma \wedge [\mathbf{I} \mid \mathbf{A}] \cdot \mathbf{y} = \mathbf{0} \,\Big| \\ &\mathbf{A} \leftarrow R_q^{m\times k}; \mathbf{y} \leftarrow \mathcal{A}(\mathbf{A})\Big]\end{aligned}$$

Definition C.5 (The $\mathsf{SelfTargetMSIS}$ Problem in [LDK+20])**.** Suppose that $\mathsf{H} : \{0, 1\}^* \to B_\tau$ is a cryptographic hash function. To an algorithm $\mathcal{A}$ we associate the advantage function

$$\begin{aligned}&\mathsf{Adv}^{\mathsf{selftargetmsis}}_{\mathsf{H},m,k,\gamma}(\mathcal{A}) = \\ &\Pr\left[\begin{array}{l} 0 < \|\mathbf{y}\|_\infty < \gamma \\ \wedge\, H(\mu \,\|\, [\mathbf{I}|\mathbf{A}] \cdot \mathbf{y}) = c \end{array} \middle| \begin{array}{l} \mathbf{A} \leftarrow R_q^{m\times k}; \\ (\mathbf{y} := \begin{bmatrix} \mathbf{r} \\ c \end{bmatrix}, \mu) \leftarrow \mathcal{A}^{|H(\cdot)\rangle}(\mathbf{A}) \end{array}\right]\end{aligned}$$

$\mathsf{ANOCCA}^{\mathcal{A}}_{\mathsf{KEM}}(\lambda)$	$\mathsf{Decaps}\mathcal{O}(b', C')$
$(\mathsf{ek}_0, \mathsf{dk}_0) \leftarrow \mathsf{KEM.Gen}(\lambda)$ $(\mathsf{ek}_1, \mathsf{dk}_1) \leftarrow \mathsf{KEM.Gen}(\lambda)$ $b \leftarrow\$ \{0,1\}$ $(C^*, K^*) \leftarrow \mathsf{KEM.Encaps}(\mathsf{ek}_b)$ $b' \leftarrow \mathcal{A}^{\mathsf{Decaps}\mathcal{O}(\cdot,\cdot)}(\mathsf{ek}_0, \mathsf{ek}_1, C^*, K^*)$ $b_0 := (b = b')$ **return** b_0	$K' := \mathsf{KEM.Decaps}(\mathsf{dk}_{b'}, C')$ **return** K'

Figure 11: Experiment for $\mathsf{ANOCCA}^{\mathcal{A}}_{\mathsf{KEM}}(\lambda)$

C.2 Cryptographic Tools

Definition C.6 (Binomial Distribution[SAB+20])**.** We define the binomial distribution B_η as follows:

$$(a_1, \ldots, a_\eta, b_1, \ldots, b_\eta) \leftarrow\$ \{0,1\}^{2\eta},$$

and then output $\sum_i^\eta a_i - b_i$. If we write some polynomial $f \leftarrow\$ B_\eta$, then each coefficient of f is sampled from B_η.

Definition C.7 (Anonymous KEM[GMP22])**.** A KEM is said to be *anonymous under chosen-ciphertext attacks* if there exists a negligible function $\mathsf{negl}(\lambda)$ for all $\lambda \in \mathbb{N}$, and for all adversaries $\mathcal{A}$ the following holds:

$$\Pr\left[\mathsf{ANOCCA}^{\mathcal{A}}(\lambda) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where ANOCCA is defined in Figure 11. Similarly, we also define IKCPA experiment for PKE in Figure 12, which just removes access to the decryption oracle[BBDP01].

Definition C.8 (Uniformly-Ambiguous Encryption[BLMG21])**.** Let $\mathsf{PKE} := (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a public-key encryption scheme for the message space $\{0,1\}^n$. For any $\lambda \in \mathbb{N}$, uniformly sampled message $\mathbf{m} \leftarrow\$ \{0,1\}^n$, we say PKE is UNIAMB-secure if

$$\begin{aligned}\mathsf{Adv}^{\mathsf{uniamb}}_{\lambda}(\mathcal{A}) := \Big| &\Pr[\mathsf{UNIAMB}^{\mathcal{A}}_{\mathsf{PKE}}(\lambda) \Rightarrow 0] - \\ &\Pr[\mathsf{UNIAMB}^{\mathcal{A}}_{\mathsf{PKE}}(\lambda) \Rightarrow 1]\Big| \leq \mathsf{negl}(\lambda),\end{aligned}$$

where the experiment $\mathsf{UNIAMB}^{\mathcal{A}}_{\mathsf{PKE}}(\lambda)$ is defined in Figure 12.

C.2.1 Statistical Tools

Definition C.9 (Statistical Distance)**.** The statistical distance between two probability distributions A and B is

$$\mathsf{SD}(A, B) = \frac{1}{2}\sum_v \big|\Pr[A = v] - \Pr[B = v]\big|.$$

Recall min-entropy of a random variable A is

$$H_\infty(A) := -\log(\max_a \Pr[A = a]),$$

then we have the following lemma.

$\mathsf{UNIAMB}^{\mathcal{A}}_{\mathsf{PKE}}(\lambda)$	$\mathsf{IKCPA}^{\mathcal{A}}_{\mathsf{PKE}}(\lambda)$
$(\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow \mathsf{PKE.Gen}(\lambda)$	$(\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow \mathsf{PKE.Gen}(\lambda)$
$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{PKE.Gen}(\lambda)$	$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{PKE.Gen}(\lambda)$
$b \leftarrow\!\$\ \{0,1\}$	$(\mathsf{st}_{\mathcal{A}}, m) \leftarrow \mathcal{A}_1(\mathsf{pk}_0, \mathsf{pk}_1)$
$\mathbf{m} \leftarrow\!\$\ \{0,1\}^n$	$b \leftarrow\!\$\ \{0,1\}$
$\mathbf{c}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_b, \mathbf{m})$	$c^* \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_b, m)$
$b' \leftarrow \mathcal{A}(\mathsf{pk}_0, \mathsf{pk}_1, \mathsf{sk}_0, \mathsf{sk}_1, \mathbf{c}^*)$	$b' \leftarrow \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{pk}_0, \mathsf{pk}_1, c^*)$
return $b \stackrel{?}{=} b'$	**return** $b \stackrel{?}{=} b'$

Figure 12: Experiment for $\mathsf{UNIAMB}^{\mathcal{A}}_{\mathsf{PKE}}(\lambda)$ and $\mathsf{IKCPA}^{\mathcal{A}}_{\mathsf{PKE}}(\lambda)$

Lemma C.1 (Leftover Hash Lemma[ILL89]). Assume a family of functions $\{\mathsf{H}_x : \{0,1\}^n \mapsto \{0,1\}^m\}_{x\in\mathcal{X}}$ is universal: $\forall\, a \neq b \in \{0,1\}^n$, $\Pr_{x\in\mathcal{X}}[\mathsf{H}_x(a) = \mathsf{H}_x(b)] = 2^{-m}$. Then, for any random variable W,

$$\mathsf{SD}((\mathsf{H}_X(W), X), (U_m, X)) \leq \epsilon,$$

whenever $m \leq k - 2\log(\frac{1}{\epsilon}) + 2$ and $k = H_\infty(W)$.

Lemma C.2 (Rank of the Circulant Matrix[Ing56]). The rank of a circulant matrix C of order m is $m-d$, where d is the degree of the greatest common divisors of $X^m - 1$ and the associated polynomial of C.

D Group-based Construction against Bounded Leakage

We provide an SS which is unforgeable and unlinkable with *bounded* key-exposure. The construction is shown in Figure 13, where $\mathbb{G}$ is a group of primer order p, g is a generator, and H is a random oracle mapping from $\mathbb{G}$ to $\mathbb{Z}_p$. Additionally, DS is an efficient group-based signature scheme such as ECDSA, Schnorr and others whose verification key and signing key has discrete logarithm relation, i.e., $\mathsf{vk} = g^{\mathsf{sk}}$.

Correctness. It is clear that $\mathsf{opk} = g^{\mathsf{osk}}$ as

$$\mathsf{opk} = \prod_{i=1}^{n} h_i^{\mathsf{H}(h_i^r)} = g^{\sum_{i=1}^{n} x_i \cdot \mathsf{H}(g^{r\cdot x_i})} = g^{\mathsf{osk}}.$$

Tracking mechanism also works since

$$R_0 = \mathsf{opk}^{\mathsf{H}(h_0^r)} = \mathsf{opk}^{\mathsf{H}(g^{r\cdot x_0})}.$$

Security Analysis. Now we analyze the security of above construction.

Theorem D.1. The construction in Figure 13 is (strongly) unforgeable and unlinkable with $(n-1)$-*bounded* key exposures.

Proof. (sketch) For unlinkability, without knowing x_i or r, by DDH assumption, the triple $g^r, g^{x_i}, g^{r\cdot x_i}$ remains uniformly random over $\mathbb{G}$. With random oracle H, $\mathsf{H}(g^{r\cdot x_i})$ is also uniformly random over $\mathbb{Z}_p$. Therefore, it is clear that opk, R, R_0 are uniformly random.

For unforgeability, as long as DS is (strongly) unforgeable, then SS is also (strongly) unforgeable.

MKGen(λ)

Sample $\mathbb{G} = \langle g \rangle$
$(x_0, x_1, \ldots, x_n) \leftarrow\$ \mathbb{Z}_p$
$\mathsf{mpk} := (g, h_0 := g^{x_0}, \ldots, h_n := g^{x_n})$
$\mathsf{msk} := (x_0, \ldots, x_n)$
$\mathsf{mtk} := x_0$
return mpk, msk, mtk

OSKGen(msk, opk, tki)

parse $(x_0, \ldots, x_n) := \mathsf{msk}$
parse $(R, R_0) := \mathsf{tki}$
return $\perp$ **if**
$\mathsf{false} \leftarrow \mathsf{Track}(x_0, , \mathsf{opk}, \mathsf{tki})$
$\mathsf{osk} := \sum_{i=1}^{n} x_i \cdot \mathsf{H}(R^{x_i})$
return osk

Vf(opk, σ, m)

return $\mathsf{DS.Vf}(\mathsf{opk}, \sigma, m)$

OPKGen(mpk)

parse $(g, h_0, \ldots, h_n) := \mathsf{mpk}$
$r \leftarrow\$ \mathbb{Z}_p$
$\mathsf{opk} := \prod_{i=1}^{n} h_i^{\mathsf{H}(h_i^r)}$
$\mathsf{tki} := (R := g^r, R_0 := \mathsf{opk}^{\mathsf{H}(h_0^r)})$
return opk, tki

Sign(osk, m)

return $\sigma := \mathsf{DS.Sign}(\mathsf{osk}, m)$

Track(mtk, opk, tki)

parse $(R, R_0) := \mathsf{tki}$
parse $x_0 := \mathsf{mtk}$
return $\mathsf{opk}^{\mathsf{H}(R^{x_0})} \stackrel{?}{=} R_0$

Figure 13: Construction of group-based SS secure with $(n-1)$-bounded key-exposure

Now we consider key-exposures. Since

$$\mathsf{osk} = \sum_{i=1}^{n} x_i \cdot \mathsf{H}(h_i^r),$$

this is an equation with n variables (x_i) for adversaries. If the adversary learns at most $n-1$ equations, then this linear system is undetermined and has at least p solutions which is exponentially large. Thus msk is hiding when there are at most $(n-1)$ key-exposures. $\square$

E Security Analysis of Generic Transform

Proof of Theorem 5.1 We restate the theorem here more formally for the case of unforgeability.

Theorem E.1. The stealth signature $\mathsf{SS_w}$ constructed in Section 5 is secure in $\mathsf{sEUFCMA_{w-ke}}$ experiment if $\mathsf{SS_{w/o}}$ is $\mathsf{EUFCMA_{w/o-ke}}$ secure and DS is sEUFCMA secure. Specifically, for any $\lambda \in \mathbb{N}$, and for any PPT adversary $\mathcal{A}$, if it succeeds in the experiment $\mathsf{sEUFCMA_{w-ke}}$, then there are other adversaries $\mathcal{B}_1, \mathcal{B}_2$ running in roughly same time such that

$$\mathsf{Adv}_\lambda^{\mathsf{seufcma_{w-ke}}}(\mathcal{A}) \leq \mathsf{Adv}_\lambda^{\mathsf{eufcma_{w/o-ke}}}(\mathcal{B}_1) + \mathsf{Adv}_\lambda^{\mathsf{seufcma}}(\mathcal{B}_2).$$

Proof. We prove the theorem by reduction. Suppose there's an adversary $\mathcal{A}$ has non-negligible advantage in $\mathsf{sEUFCMA_{w-ke}}$, then we can construct another adversary $\mathcal{B}$ to win the experiment $\mathsf{EUFCMA_{w/o-ke}}$ of $\mathsf{SS_{w/o}}$ or the experiment sEUFCMA (strong unforgeability) of DS as follows. $\mathcal{B}$ forwards mpk, mtk from the challenger in $\mathsf{EUFCMA_{w/o-ke}}$ to $\mathcal{A}$.

To simulate $\mathsf{OSKGen}\mathcal{O}(i, \mathsf{opk}^i, \mathsf{tki}^i, \mathsf{flag}^i)$, if $\mathsf{flag}^i = \mathsf{true}$, $\mathcal{B}$ runs $(\mathsf{vk}^i, \mathsf{sk}^i) \leftarrow \mathsf{DS.Gen}$, then queries $\sigma_1^i \leftarrow \mathsf{Sign}\mathcal{O}(\mathsf{vk}^i)$ in $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$ and returns $\mathsf{osk}^i := (\sigma_1^i, \mathsf{vk}^i, \mathsf{sk}^i)$ to $\mathcal{A}$; If $\mathsf{flag}^i = \mathsf{false}$, $\mathcal{B}$ asks a challenger $\mathcal{C}^i$ in $\mathsf{sEUFCMA}$ of DS to send a challenge verification key vk^i, then queries $\sigma_1^i \leftarrow \mathsf{Sign}\mathcal{O}(\mathsf{vk}^i)$ in $\mathsf{sEUFCMA}_{\mathsf{w/o-ke}}$ of $\mathsf{SS}_{\mathsf{w/o}}$ and stores σ_1^i; If $\mathsf{OK}[i] = (\mathsf{opk}^i, \cdot, \cdot) \wedge \mathsf{flag}^i = \mathsf{true}$, $\mathcal{B}$ signals $\mathcal{C}^i$ to terminal the experiment and asks for its osk^i then forwards that to $\mathcal{A}$. To simulate $\mathsf{Sign}\mathcal{O}(i, m^j)$, $\mathcal{B}$ queries $\sigma_2^j \leftarrow \mathsf{Sign}\mathcal{O}(m^j || \sigma_1^i)$ and returns $\sigma^j := (\sigma_1^i, \sigma_2^j, \mathsf{vk}^i)$ to $\mathcal{A}$.

Once $\mathcal{A}$ submits some valid forgery $\sigma' := (\sigma_1', \sigma_2', \mathsf{vk}'), m', i'$ as shown in Figure 3, $\mathcal{B}$ behaves in following cases:

- If m' is not appeared in Q (recall that Q is the set to record signing queries), $\mathcal{B}$ forwards $\sigma_2', m' || \sigma_1'$ to i'-th challenger in $\mathsf{sEUFCMA}$ of DS;
- If vk' is not appeared in Q, $\mathcal{B}$ forwards σ_1', vk' to the challenger in $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$ of $\mathsf{SS}_{\mathsf{w/o}}$;
- If both m', vk' are in Q, then the only case that σ' is a valid forgery is either σ_1' or σ_2' not appeared in Q. In either case, $\mathcal{B}$ just forwards $\sigma_2', m' || \sigma_1'$ to the challenger in $\mathsf{sEUFCMA}$ of DS.

This completes the proof.

□

We restate the theorem here for unlinkability.

Theorem E.2. The stealth signature $\mathsf{SS_w}$ constructed in Section 5 is secure in $\mathsf{UNLNK}_{\mathsf{w-ke}}$ experiment if $\mathsf{SS}_{\mathsf{w/o}}$ is $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$ secure. Specifically, for any $\lambda \in \mathbb{N}$, and for any PPT adversary $\mathcal{A}$, if it succeeds in the experiment $\mathsf{UNLNK}_{\mathsf{w-ke}}$, then there are other adversaries $\mathcal{B}$ running in roughly same time such that

$$\mathsf{Adv}_{\lambda}^{\mathsf{unlnk}_{\mathsf{w-ke}}}(\mathcal{A}) \leq \mathsf{Adv}_{\lambda}^{\mathsf{unlnk}_{\mathsf{w/o-ke}}}(\mathcal{B}).$$

Proof. Similarly, we can also prove this theorem easily by reduction. Suppose there's an adversary $\mathcal{A}$ has non-negligible advantage in $\mathsf{UNLNK}_{\mathsf{w-ke}}$, then we can construct another adversary $\mathcal{B}$ to win the experiment $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$ of $\mathsf{SS}_{\mathsf{w/o}}$ as follows. $\mathcal{B}$ forwards $\mathsf{mpk}_0, \mathsf{mpk}_1, \mathsf{opk}_b, \mathsf{tki}_b$ from the challenger in $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$ to $\mathcal{A}$. To simulate osk_b, $\mathcal{B}$ runs $\mathsf{DS.Gen}$ to get $(\mathsf{vk}, \mathsf{sk})$, then queries the signing oracle via $\mathsf{Sign}\mathcal{O}(\cdot, -1, \mathsf{vk})$ from $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$ to learn a signature σ_1 of vk, then returns $\mathsf{osk}_b := (\sigma_1, \mathsf{vk}, \mathsf{sk})$ to $\mathcal{A}$. To simulate $\mathsf{OSKGen}\mathcal{O}$, $\mathcal{B}$ queries $\mathsf{Sign}\mathcal{O}$ and runs $\mathsf{DS.Gen}$ as above to generate osk. Once $\mathcal{A}$ submits b', $\mathcal{B}$ simply forwards b' as its final guess. This completes the proof.

□

F Security Analysis of Stealth Signature Without Fuzzy Tracking

Proof of Lemma 6.1 We restate the lemma formally here.

Lemma F.1. SPIRIT in Figure 8 is unforgeable without key exposure under no-message attacks. Specifically, in random oracle model, for any $\lambda \in \mathbb{N}$, for any adversary $\mathcal{A}$, if Dil has parameters $\beta, \gamma_1, \gamma_2$, and we denote H' as a random oracle can be accessed by $\mathcal{A}$ and $\mathcal{B}_2$, then the advantage to win the game $\mathsf{UFNMA}_{\mathsf{w/o-ke}}^{\mathcal{A}}(\lambda)$ is

$$\mathsf{Adv}_{\lambda,\mathsf{H}',\gamma_1,\gamma_2,\beta}^{\mathsf{ufnma}_{\mathsf{w/o-ke}}}(\mathcal{A}) \leq \mathsf{Adv}_{k,\ell,D}^{\mathsf{mlwe}}(\mathcal{B}_1) \qquad +\mathsf{Adv}_{\mathsf{H}',k,\ell+1,\zeta}^{\mathsf{selftargetmsis}}(\mathcal{B}_2).$$

Proof. Consider the experiment $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$ in Figure 1 where the $\mathsf{Sign}\mathcal{O}$ is forbidden to access. Suppose $\mathcal{A}$ forges $\sigma*$, then we have the following claim.

Claim 1. If an adversary $\mathcal{A}$ can forge σ^* without accessing $\mathsf{Sign}\mathcal{O}$ and assuming $\mathsf{MLWE}_{k,\ell,D}$ assumption holds, then there is another adversary $\mathcal{B}_2$ who solves $\mathsf{SelfTargetMSIS}_{\mathsf{H}',k,\ell+1,\zeta}$ in roughly same time with non-negligible probability.

Proof. After receiving uniformly random samples $(\mathbf{A}, \mathbf{t}) \in R^{k\times\ell} \times R^k$ and random oracle access $\mathsf{H}'(\cdot)$ from the challenger in
$\mathsf{SelfTargetMSIS}_{\mathsf{H}',k,\ell+1,\zeta'+\beta}$, $\mathcal{B}_2$ computes $\mathsf{mpk} := (\mathbf{A}, \mathbf{t}, \mathsf{ek})$, $\mathsf{mtk} := (\mathsf{dk}, \mathbf{t})$ and forwards $\mathsf{mpk}, \mathsf{mtk}, \mathsf{H}'$ to $\mathcal{A}$. As long as the $\mathsf{MLWE}_{k,\ell,D}$ assumption holds, mpk looks indistinguishable from real public key for $\mathcal{A}$. For i-th query in $\mathsf{OSKGen}\mathcal{O}$, $\mathcal{B}_2$ computes and stores $\mathbf{s}_1^i, \mathbf{s}_2^i$. Once $\mathcal{A}$ submits some valid forgery σ^* with i^*, meaning it finds some $(\mathbf{x}, \mathbf{z}, c)$ for $\mathsf{opk}^* := \mathbf{t}^*$ such that

$$\mathsf{H}'\left(\mu \,\|\, [\mathbf{I}_k|\mathbf{A}|\mathbf{t}^*] \cdot \begin{bmatrix}\mathbf{x}\\ \mathbf{z}\\ c\end{bmatrix}\right) = c,$$

where $\|\mathbf{x}\|_\infty \le 2\gamma_2 + 1 + 2^{d-1}\tau$, $\|\mathbf{z}\|_\infty \le \gamma_1 - 2\beta$ and $\|c\|_\infty = 1$[LDK+20]. Then $\mathcal{B}_2$ can retrieve $\mathbf{s}_1^*, \mathbf{s}_2^*$ from its storage and instantly return $\mathbf{y} := \begin{bmatrix}\mathbf{x}'\\ \mathbf{z}'\\ c\end{bmatrix}, \mu$ to the $\mathsf{SelfTargetMSIS}_{\mathsf{H}',k,\ell+1,\zeta}$ challenger, where $\mathbf{x}' := \mathbf{x} + c\mathbf{s}_2^*$ and $\mathbf{z}' := \mathbf{z} + c\mathbf{s}_1^*$. Note that $\|c\mathbf{s}_1^*\|, \|c\mathbf{s}_2^*\| \le \beta$. Since we can write $\mathbf{t}^* := \mathbf{t} + \mathbf{A}\mathbf{s}_1^* + \mathbf{s}_2^*$, it is easy to check that this is a valid solution

$$\mathsf{H}'\left(\mu \,\|\, [\mathbf{I}_k|\mathbf{A}|\mathbf{t}] \cdot \begin{bmatrix}\mathbf{x} + c\mathbf{s}_2^*\\ \mathbf{z} + c\mathbf{s}_1^*\\ c\end{bmatrix}\right) = c$$

where $\|\mathbf{y}\|_\infty \le \zeta$ and $\zeta := \max\{\gamma_1 - \beta, 2\gamma_2 + 1 + 2^{d-1}\tau + \beta\}$. ☐

This completes the proof to show it is secure in $\mathsf{UFNMA}_{\mathsf{w/o-ke}}$ experiment. ☐

Proof of Theorem 6.2 We restate the theorem for unforgeable without key exposures formally here.

Theorem F.2. SPIRIT in Figure 8 is existential unforgeable without key exposures. Specifically, for any adversary $\mathcal{A}$, if it succeeds in the experiment $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$, then there is another adversary $\mathcal{B}$ running in roughly same time such that

$$\mathsf{Adv}^{\mathsf{eufcma}_{\mathsf{w/o-ke}}}_{\lambda,\mathsf{H},\gamma_1,\gamma_2,\beta}(\mathcal{A}) \le \mathsf{Adv}^{\mathsf{ufnma}_{\mathsf{w/o-ke}}}_{\lambda,\mathsf{H}',\gamma_1,\gamma_2,\beta}(\mathcal{B}) + \mathsf{negl}(\lambda),$$

where we denote H', H as random oracles can be accessed by $\mathcal{B}_1$ and $\mathcal{A}$, respectively.

Proof. Intuitively, reduction from CMA to NMA usually needs "patching" random oracles [KLS18, AFLT12]. We prove this theorem in a sequence of hybrid games as follows.
$\mathbf{Hybrid}_0$**:** This is exactly the standard $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$ experiment. Thus we have

$$\Pr[\mathbf{Hybrid}_0 \Rightarrow 1] = \mathsf{Adv}^{\mathsf{eufcma}_{\mathsf{w/o-ke}}}_{\lambda,\mathsf{H},\gamma_1,\gamma_2,\beta}(\mathcal{A}).$$

$\mathbf{Hybrid}_1$**:** We modify $\mathbf{Hybrid}_0$ as follows. In $\mathsf{OSKGen}\mathcal{O}(\mathsf{opk}^i, \mathsf{tki}^i)$, for i-th query, if $\mathsf{true} \leftarrow \mathsf{Track}(\mathsf{mtk}, \mathsf{opk}^i, \mathsf{tki}^i)$ it only stores $\mathbf{s}_1^i, \mathbf{s}_2^i, \mathbf{t}^i := \mathbf{A}\mathbf{s}_1^i + \mathbf{s}_2^i + \mathbf{t}$, sets $\mathsf{osk}^i := \top$ and returns 1. In $\mathsf{Sign}\mathcal{O}(i, m^j)$, for j-th query, it generates and sets osk^i by msk if $\mathsf{osk}^i := \top$, then return a signature

$\mathsf{H}(\mathbf{w}_1 \| \mu)$	$\mathsf{EUFCMA}^{\mathcal{A}}_{\mathsf{w/o-ke}}(\lambda)$
// in Hyb_2 and Hyb_3	$(\mathsf{mpk}, \mathsf{msk}, \mathsf{mtk}) \leftarrow \mathsf{MKGen}(\lambda)$
Retrieve $\langle \mu : (c^{\mu}, \mathbf{w}_1^{\mu}\rangle$ for μ	$\mathsf{OK} := [], Q := \emptyset$
if $\mathbf{w}_1 = \mathbf{w}_1^{\mu}$	$(m^*, \sigma^*, i^*) \leftarrow \mathcal{A}^{\mathsf{OSKGen}\mathcal{O}, \mathsf{Sign}\mathcal{O}}(\mathsf{mpk}, \mathsf{mtk})$
then return $c := c^{\mu}$	$(\mathsf{opk}^* := \mathbf{t}^*, \mathsf{osk}^*, \cdot) := \mathsf{OK}[i^*]$
else return $c := \mathsf{H}'(\mathbf{w}_1 \| \mu)$	// Hyb_3 block begins
	$(\mathbf{z}^*, c^*, \mathbf{h}^*) := \sigma^*$
	$\mu^* \leftarrow \mathsf{G}(m^* \| \mathbf{t}^*)$
	$\mathbf{w}_1^* \leftarrow \mathsf{HighBits}_q(\mathbf{A}\mathbf{z}^* - c^*\mathbf{t}^*, 2\gamma_2)$
	if $\mathsf{H}'(\mathbf{w}_1^* \| \mu^*) \neq c^*$
	then return 0
	// Hyb_3 block ends
	$b_0 := (m^*, i^*) \notin Q$
	$b_1 := \mathsf{Vf}(\mathsf{opk}^*, m^*, \sigma^*) = 1$
	$b_2 := (\mathsf{OK}[i^*] \neq (\cdot, \cdot, \perp))$
	return $b_0 \wedge b_1 \wedge b_2$

Figure 14: Simulated H and $\mathsf{EUFCMA}^{\mathcal{A}}_{\mathsf{w/o-ke}}(\lambda)$ in $\mathbf{Hybrid}_2$ and $\mathbf{Hybrid}_3$

σ^j by using osk^i. This game only changes the time to generate osk^i, thus advantage remains the same:

$$|\Pr[\mathbf{Hybrid}_1 \Rightarrow 1] - \Pr[\mathbf{Hybrid}_0 \Rightarrow 1]| = 0.$$

$\mathbf{Hybrid}_2$**:** We update $\mathbf{Hybrid}_1$ by modifying $\mathsf{Sign}\mathcal{O}(i, m^j)$ in j-th query: Instead of generating σ^j with osk^i when needed, it just simulates σ^j by choosing uniformly random $(\mathbf{z}^j, c^j) \in S^{\ell}_{\gamma_1 - 2\beta - 1} \times B_{\tau}$ and stores a key-value pair $\langle \mu^j : (c^j, \mathbf{w}_1^j)\rangle$ where $\mu^j \leftarrow \mathsf{G}(m^j \| \mathbf{t}^i)$, $\mathbf{w}_1^{\mu^j} \leftarrow \mathsf{HighBits}_q(\mathbf{A}\mathbf{z}^j - c^j\mathbf{t}^i, 2\gamma_2)$, and G is a perfect random function. We also use a new random oracle $\mathsf{H}(\mathbf{w}_1 \| \mu)$ to simulate random oracle $\mathsf{H}'(\mathbf{w}_1 \| \mu)$ in above game as shown in left part of Figure 14. Now we analyze the advantage. In our construction, $\mathsf{Dil.Sign}$ remains unaltered, thus the resulting signature σ is still perfectly zero-knowledge (where the exact simulation is shown in Sign of Figure 15). Therefore the distribution of each σ is exactly the same as the one in $\mathbf{Hybrid}_1$, then we have

$$|\Pr[\mathbf{Hybrid}_2 \Rightarrow 1] - \Pr[\mathbf{Hybrid}_1 \Rightarrow 1]| = 0.$$

$\mathbf{Hybrid}_3$**:** We modify the above game by adding an additional block in $\mathsf{EUFCMA}_{\mathsf{w/o-ke}}$ as shown in right part of Figure 14. This game only differs from the $\mathbf{Hybrid}_2$ if $\mathbf{w}_1^* = \mathbf{w}_1^{\mu^*}$ and $((m^*, \cdot, i^*) \notin Q) \wedge b_1 \wedge b_2$ ($\mathbf{Hybrid}_3$ return 0 and $\mathbf{Hybrid}_2$ return 1). However, $\mathcal{A}$ didn't query $\mathsf{Sign}\mathcal{O}(i^*, m^*)$ before, thus $\mathbf{w}_1^{\mu^*}$ should remain hidden. And from [KLS18], it shows that Dil signature has enough min-entropy, thus the probability $\Pr[\mathbf{w}_1^* = \mathbf{w}_1^{\mu^*}]$ is negligible, i.e.,

$$|\Pr[\mathbf{Hybrid}_3 \Rightarrow 1] - \Pr[\mathbf{Hybrid}_2 \Rightarrow 1]| \leq \mathsf{negl}(\lambda).$$

This game can be fully simulated by $\mathcal{B}$ against $\mathsf{UFNMA}_{\mathsf{w/o-ke}}$ as follows. $\mathcal{B}_1$ simulates $\mathsf{OSKGen}\mathcal{O}, \mathsf{Sign}\mathcal{O}$ oracles without knowing msk, and it patches H' from $\mathsf{UFNMA}_{\mathsf{w/o-ke}}$ to H for generating σ^i. Once $\mathcal{A}$ submits a valid signature σ^* and if H' works well in σ^*, $\mathcal{B}$ directly forwards σ^* to the challenger of $\mathsf{UFNMA}_{\mathsf{w/o-ke}}$. Therefore

$$\Pr[\mathbf{Hybrid}_3 \Rightarrow 1] = \mathsf{Adv}^{\mathsf{ufnma}_{\mathsf{w/o-ke}}}_{\lambda, \mathsf{H}', \gamma_1, \gamma_2, \beta}(\mathcal{B}_1)$$

and we completes the proof. □

$$
\begin{array}{l}
\underline{\mathsf{Sign}(\mathsf{opk}^i, m^j)} \\
\quad /\!/ \text{ in } Hyb_1 \text{ and } Hyb_2 \\
\textbf{parse } \mathbf{t}^i := \mathsf{opk}^i \\
(\mathbf{z}^i, c^i) \leftarrow\!\!\$\; S^{\ell}_{\gamma_1 - 2\beta_1} \times B_\tau \\
\mu^i \leftarrow \mathsf{G}(m^j || \mathbf{t}^i) \\
\mathbf{w}_1^i \leftarrow \mathsf{HighBits}_q(\mathbf{A}\mathbf{z}^i - c^i\mathbf{t}^i, 2\gamma_2) \\
\text{Program } \mathsf{H} \text{ s.t. } \mathsf{H}(\mathbf{w}_1^i || \mu_1^i) := c^i \\
\mathbf{h}^i \leftarrow \mathsf{MakeHint}_q(-c^i\mathbf{t}_0^i, \mathbf{A}\mathbf{z}^i - c^i\mathbf{t}^i + c^i\mathbf{t}_0^i, 2\gamma_2) \\
\quad /\!/ \ \mathbf{t}_0^i \text{ are lower bits of } \mathbf{t}^i \\
\textbf{return } \sigma^i := (\mathbf{z}^i, c^i, \mathbf{h}^i)
\end{array}
$$

Figure 15: Simulation of Sign from $\mathbf{Hybrid}_1$ to $\mathbf{Hybrid}_2$

We state the theorem for unlinkable without key exposures formally here.

Theorem F.3. SPIRIT in Figure 8 is unlinkable without key exposures. Specifically, for any adversary $\mathcal{A}$, if it succeeds in the experiment $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$, then there are other adversaries $\mathcal{B}_1, \mathcal{B}_2$ running in roughly same time such that

$$\mathsf{Adv}^{\mathsf{unlnk}_{\mathsf{w/o-ke}}}_{\lambda,\mathsf{H},\gamma_1,\gamma_2,\beta}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{anocca}}_{\lambda}(\mathcal{B}_1) + \mathsf{Adv}^{\mathsf{mlwe}}_{k,\ell,D}(\mathcal{B}_2).$$

where we denote H as a random oracles can be accessed by $\mathcal{A}$ and $\gamma_1, \gamma_2, \beta$ are parameters of the underlying Dil scheme.

Proof. We prove the theorem in a sequence of hybrid games.
$\mathbf{Hybrid}_0$**:** This is the original $\mathsf{UNLNK}_{\mathsf{w-ke}}$ experiment, thus we have

$$\Pr[\mathbf{Hybrid}_0 \Rightarrow 1] = \mathsf{Adv}^{\mathsf{unlnk}_{\mathsf{w/o-ke}}}_{\lambda,\mathsf{H},\gamma_1,\gamma_2,\beta}(\mathcal{A}).$$

$\mathbf{Hybrid}_1$**:** We modify the above game by changing the function $\mathsf{Sign}(\mathsf{osk}, m^j)$ in $\mathsf{UNLNK}_{\mathsf{w/o-ke}}$ experiment to the $\mathsf{Sign}(\mathsf{opk}^i, m^j)$ without using osk in Figure 15. Specifically, it samples uniformly random $(\mathbf{z}^j, c^j)$, programs the random oracle such that $\mathsf{H}(\mu^j || \mathbf{w}_1^j) = c^j$ where μ^j is determined by m^j and $\mathbf{w}_1^j := \mathsf{HighBits}(\mathbf{A}\mathbf{z}^j - c^j\mathbf{t}^i, 2\gamma_2)$. Then set $\sigma^j := (\mathbf{z}^j, c^j, \mathbf{h}^j)$ where $\mathbf{h}^j$ can be determined by $c^j, \mathbf{t}^i, \mathbf{z}^j$. Because of the perfectly zero-knowledge of σ^j , the distribution of signatures in this hybrid is the same as the one in $\mathbf{Hybrid}_0$, i.e.,

$$|\Pr[\mathbf{Hybrid}_0 \Rightarrow 1] - \Pr[\mathbf{Hybrid}_1 \Rightarrow 1]| = 0.$$

$\mathbf{Hybrid}_2$**:** We modify the above game by follows. Parse $\mathsf{mpk}_0 := (\mathbf{t}_0, \mathsf{ek}_{1,0})$ and $\mathsf{mpk}_1 := (\mathbf{t}_1, \mathsf{ek}_{1,1})$, instead of generating $\mathbf{t}_0, \mathbf{t}_1$ from msk, we sample uniformly random $(\mathbf{t}_0, \mathbf{t}_1) \leftarrow\!\!\$\; R_q^k \times R_q^k$. By $\mathsf{MLWE}_{k,\ell,D}$ assumption, we know this hybrid only differs from $\mathbf{Hybrid}_1$ by:

$$|\Pr[\mathbf{Hybrid}_2 \Rightarrow 1] - \Pr[\mathbf{Hybrid}_1 \Rightarrow 1]| \leq \mathsf{Adv}^{\mathsf{mlwe}}_{k,\ell,D}(\mathcal{B}_2).$$

Besides, this hybrid can be fully simulated by an adversary $\mathcal{B}_1$ of ANOCCA experiment. $\mathcal{B}_1$ simulates the random oracle H for $\mathcal{A}$. Upon receiving $\mathsf{ek}_0, \mathsf{ek}_1$ and (C_b, K_b) from ANOCCA experiment, $\mathcal{B}_1$ sets $\mathsf{mpk}_0 := (\mathbf{t}_0, \mathsf{ek}_0)$ and $\mathsf{mpk}_1 := (\mathbf{t}_0, \mathsf{ek}_1)$ where $(\mathbf{t}_0, \mathbf{t}_1) \leftarrow\!\!\$\; R_q^k \times R_q^k$ are uniformly sampled.

$\mathsf{pRgv.Gen}(\lambda, n)$	$\mathsf{pRgv.Enc}(\mathsf{pk}, \mathbf{m} \in \{0,1\}^n)$
$\mathbf{A} \in \mathbb{Z}_q^{\ell \times \ell} \leftarrow\$ \,\mathsf{crs}$ $(\mathbf{S}, \mathbf{E}) \leftarrow\$ \,(\chi^{\ell \times n})^2$ $\mathbf{B} := \mathbf{AS} + \mathbf{E}$ **return** $\mathsf{pk} := \mathbf{B}, \mathsf{sk} := \mathbf{S}$ $\mathsf{pRgv.Dec}(\mathsf{sk}, \mathsf{ct})$ **parse** $(\mathbf{c}_1, z, w_1, \ldots, w_n) := \mathsf{ct}$ **parse** $(\mathbf{s}_1, \ldots, \mathbf{s}_t) := \mathsf{sk}$ $\forall i \in [t], m_i := \lceil \mathbf{s}_i^T \mathbf{c}_1 + z \rfloor_2 \oplus w_i$ **return** $\mathbf{m} := [m_1, \ldots, m_t]$	$(\mathbf{r}, \mathbf{e}_1) \leftarrow\$ \,(\chi^{\ell})^2, \mathbf{e}_2 \leftarrow\$ \,\chi^n$ $\mathbf{c}_1 := \mathbf{A}^T \mathbf{r} + \mathbf{e}_1$ $\mathbf{c}_2 := \mathbf{B}^T \mathbf{r} + \mathbf{e}_2 + \frac{q}{2} \cdot \mathbf{m}$ $z \leftarrow\$ \,\mathbb{Z}_q$ such that $\forall i \in [n]$: $z + c_{2,i} \notin [\frac{q}{4} - B, \frac{q}{4} + B] \cup$ $[\frac{3q}{4} - B, \frac{3q}{4} + B]$ $\forall i \in [n], w_i := \lceil z + c_{2,i} \rfloor_2$ **return** $\mathsf{ct} := (\mathbf{c}_1, z, w_1, \ldots, w_n)$

Figure 16: Packed Regev encryption pRgv with ciphertext compression

$\mathcal{B}_1$ sets $\mathsf{tki}_b := C_b, \mathsf{osk}_b := \top, \mathsf{opk}_b \leftarrow\$ \,R_q^k$, and sends $(\mathsf{mpk}_0, \mathsf{mpk}_1, \mathsf{tki}_b, \mathsf{opk}_b)$ to $\mathcal{A}$ of $\mathbf{Hybrid}_2$. For each query of $\mathsf{OSKGen}\mathcal{O}(b^*, \mathsf{opk}^i, \mathsf{tki}^i)$, $\mathcal{B}_1$ queries $K^i \leftarrow \mathsf{KEM.\ Decaps}\mathcal{O}(b^*, \mathsf{tki}^i)$ to check if $\mathbf{As}_1^i + \mathbf{s}_2^i + \mathbf{t}_{b^*} = \mathsf{opk}^i$ where $\mathbf{s}_1^i, \mathbf{s}_2^i \leftarrow \mathsf{Dil.ExpandS}(K^i)$. If the check doesn't pass, set $\mathsf{osk}_{b^*}^i :=\perp$; Otherwise set $\mathsf{osk}_{b^*}^i := \top$. For each query of $\mathsf{Sign}\mathcal{O}(b^*, i, m^j)$, $\mathcal{B}_1$ simulates the signature σ^j by using $\mathsf{opk}_{b^*}^i$ if the corresponding $\mathsf{osk}_{b^*}^i = \top$, otherwise return $\perp$. If $i = -1$, just simulates a signature using opk_b.

Then $\mathbf{Hybrid}_2$ can be simulated without knowing any $\mathsf{msk}, \mathsf{mtk}$ or b. Once $\mathcal{A}$ returns b', $\mathcal{B}_1$ simply forwards b' to the challenger of ANOCCA. Thus we have

$$\Pr[\mathbf{Hybrid}_2 \Rightarrow 1] = \mathsf{Adv}_\lambda^{\mathsf{anocca}}(\mathcal{B}_1),$$

and this completes the proof. □

G Analysis of Post-quantum FMD

We first recall the construction of packed Regev with ciphertext compression [BDGM19] in Figure 16, where χ is the error distribution and B is the error bound between $z + c_{2,i}$ and $z + \mathbf{s}_i^T \mathbf{c}_1$.

Note that apart from the header $(\mathbf{c}_1, z)$, the payload (w_i) are just n bits which is almost as succinct as DLog-based fuzzy message detection scheme FMD_2 in [BLMG21]. Specifically, the entire ciphertext is $(\ell + 1) \log q + n$-bit large.

Correctness. We show the scheme in Figure 9 satisfies Definition 4.8 as follows. For each $i \in [t]$, we have $\lceil \mathbf{s}_i^T \mathbf{c}_1 + z \rfloor_2 \oplus w_i = 1$. Since $c_{2,i} - \mathbf{s}_i^T \mathbf{c}_1 = \frac{q}{2} + e'$ where $e' \in [-B, B]$ is some short error, we have $c_{2,i} - e' = \mathbf{s}_i^T \mathbf{c}_1 + \frac{q}{2}$. Also, we choose

$$c_{2,i} + z \notin [\frac{q}{4} - B, \frac{q}{4} + B] \cup [\frac{3q}{4} - B, \frac{3q}{4} + B],$$

thus we have $\lceil c_{2,i} + z \rfloor_2 = \lceil c_{2,i} + z - e' \rfloor_2$, which implies $w_i = \lceil \mathbf{s}_i^T \mathbf{c}_1 + z + \frac{q}{2} \rfloor_2 = \lceil \mathbf{s}_i^T \mathbf{c}_1 + z \rfloor_2 \oplus 1$. Therefore, with correct ftk, FTrack always returns true. Note that for correctness, we require $q > 4Bn$.

Security Analysis. We show the scheme in Figure 9 is unlinkable with key-exposure and fuzzy tracking (Definition 4.9).

Proof of Lemma 6.3 and Theorem 6.4 We restate the formal lemma here.

Lemma G.1. Packed Regev encryption pRgv with ciphertext compression shown in Figure 16 satisfies Definition C.8 and is uniformly-ambiguous UNIAMB-secure when $\frac{4Bn}{q}$ is $\mathsf{negl}(\lambda)$. Specifically, we have

$$\mathsf{Adv}_{\lambda}^{\mathsf{uniamb}}(\mathcal{A}) \leq \mathsf{Adv}_{\ell,q}^{\mathsf{lwe}}(\mathcal{A}) + \frac{4Bn}{q},$$

where B is the bound such that $\left\|\mathbf{S}^T\mathbf{c}_1 - \mathbf{c}_2\right\|_\infty \mod \frac{q}{2} < B$.

Proof. To see it is uniformly-ambiguous, firstly note that $\mathbf{c}_1$ looks uniformly random due to LWE assumption, and w_i is uniformly random due to m_i is a uniformly random bit in the experiment in Figure 12. For z, the statistical distance between its distribution and uniformly random distribution over $\mathbb{Z}_q$ is $\frac{4Bn}{q}$. Thus as long as $\frac{4Bn}{q} \leq \mathsf{negl}(\lambda)$, we can simulate the entire ciphertext without knowing b or sk. □

We restate the theorem formally here.

Theorem G.2. The fuzzy stealth signature constructed in Figure 9 is *unlinkable with key-exposure and fuzzy tracking*. Specifically, for any λ, n, t where $n \geq t$, if there is a PPT adversary $\mathcal{A}$ has non-negligible advantage in experiment defined in Figure 5, then there exist other adversaries $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ running in roughly same time such that:

$$\begin{aligned}\mathsf{Adv}_{\lambda,n,t}^{\mathsf{unlnk}_{\mathsf{fw-ke}}}(\mathcal{A}) \leq 2\mathsf{Adv}_{\lambda}^{\mathsf{unlnk}_{\mathsf{w-ke}}}(\mathcal{B}_1) +& \\ p(\lambda) \cdot \big(4t\mathsf{Adv}_{\lambda}^{\mathsf{uniamb}}(\mathcal{B}_2) + (n-t)\mathsf{Adv}_{\lambda}^{\mathsf{ikcpa}}(\mathcal{B}_3)\big)&,\end{aligned}$$

where $p(\lambda)$ is some polynomial on security parameter λ.

Proof. Combined with Lemma G.1, recall Theorem 11 and Lemma 2 in [BLMG21] to prove this via the same approach. □

Extends to finer false-positive rates. We introduce an approach to achieve finer false-positive rates ($\rho \neq \frac{1}{2^t}$) in fuzzy tracking (and also FMD) schemes. As mentioned in [BLMG21], to achieve finer rates like $\frac{1}{3}, \frac{1}{5}$ is easy via switching the base. However, to achieve rates like $\frac{3}{4}$ is still challenging without garbled circuits. We show how to achieve rate like $\frac{\alpha}{2^k}$ where $1 \leq \alpha \leq 2^k - 1$ with a small tweak but α, k needs to be fixed in advance. The sender instead of computing $\mathsf{Enc}(\mathsf{pk}_i, 1)$ for each $i \in [n]$, it computes $c_i \leftarrow \mathsf{Enc}(\mathsf{pk}_i, \mathsf{msg}_i)$ where msg_i is uniformly sampled via $\mathsf{msg}_i \leftarrow\!\!\$\, \{0, 1, \ldots, \alpha\}$. The detector only accepts the ciphertext c_i if and only if $\mathsf{Dec}(\mathsf{sk}_i, c_i) \leq \alpha$. It is easy to see that this satisfies correctness, fuzziness and security simultaneously and is compatible to $\mathsf{FMD}_1, \mathsf{FMD}_2$ in [BLMG21] and our fuzzy tracking scheme in Figure 9. Essentially, the receiver is able to 'tune' the false-positive rate ρ via a finer step: Originally, ρ can only be decreased half by half (i.e., from ρ to $\frac{\rho}{2}$ each time); Now it can be decreased by a factor $\frac{\alpha}{2^k}$ (i.e., from ρ to $\frac{\rho\alpha}{2^k}$). For example, if we choose $k = 2, \alpha = 3$, then we have rates set like $\{\frac{3}{4}, \frac{3^2}{4^2}, \ldots, \frac{3^n}{4^n}\}$.

H Analysis of Scalable Fuzzy Tracking

Security Analysis. For adversaries without holding secret keys, arguments for security are the same as standard encryption. We consider the unlinkability defined in Definition 4.12, then we argue it also satisfies unbiased fuzziness defined in Definition 4.13. Intuitively, unlinkability is to make true-positive and false-positive indistinguishable from the tracking server; And unbiased fuzziness is to make the hint' of each potential mpk' uniformly random for the sender.

Proof of Theorem 6.5 We restate the theorem for unlinkability formally here.

Theorem H.1. The fuzzy scalable stealth signature constructed in Figure 10 is unlinkable with key-exposure and fuzzy tracking. Specifically, for any λ, N, ρ, if there is a PPT adversary $\mathcal{A}$ has non-negligible advantage in experiment defined in Figure 6, then there exist other adversaries $\mathcal{B}$ running in roughly same time such that:

$$\mathsf{Adv}_{\lambda,N,\rho}^{\mathsf{unlnk_{fs_{w-ke}}}}(\mathcal{A}) = \mathsf{Adv}_{\lambda}^{\mathsf{unlnk_{w-ke}}}(\mathcal{B}) + \mathsf{Adv}_{\ell,q,\eta}^{\mathsf{mlwe}}(\mathcal{C}).$$

Proof. First consider the two hybrids as follows:

Hybrid$_0$: This is the standard experiment.

Hybrid$_1$: This only changes ftki_b to ftki_{1-b} when $\mathsf{hint}_0 \in \mathsf{list} \wedge \mathsf{hint}_1 \in \mathsf{list}$ whereas $\mathsf{opk}_b, \mathsf{tki}_b$ remain unchanged.

Claim 2. **Hybrid$_0$** and **Hybrid$_1$** are computationally indistinguishable to the adversary if the decisional MLWE holds.

Proof. Since we maps each mpk to hint, we only need to consider the case where $\mathsf{hint}_0 \in \mathsf{list} \wedge \mathsf{hint}_1 \in \mathsf{list}$ as otherwise $b' \leftarrow\$ \{0,1\}$ and $\mathcal{A}_2$ will not be invoked. Without loss of generality, we assume $\mathsf{ftki}_b = \mathsf{ftki}_0$ and $\mathsf{hint}_0 = \mathsf{list}[i]$ which implies that, for list generated from ftki_0 and $\forall j \in [|\mathsf{list}|]$, there is

$$\begin{aligned}
w_0^j &= \lceil \mathbf{s}^T \mathbf{c}_1^j - c_2 \rfloor_2 \oplus y^j \\
&= \lceil \frac{q}{2}(s_1(x^i - x^j)) + e' - \frac{q}{2}(w_0 + y^i) \rfloor_2 \oplus y^j \\
&= \lceil \frac{q}{2}(s_1(x^i - x^j)) + e' - \frac{q}{2}(w_0) \rfloor_2 \oplus (y^i \oplus y^j) \\
&= \lceil \frac{q}{2}(w_0 + e' + s_1(x^i - x^j)) \rfloor_2 \oplus y^i \oplus y^j \\
&= w_0 \oplus (y^i \oplus y^j) \oplus \lceil \frac{q}{2}(s_1(x^i - x^j)) \rfloor_2,
\end{aligned}$$

where s_1 is the first ring element of $\mathbf{s}$ and $\mathsf{hint}_0 = \mathsf{decode}_{R_q}(w_0)[: n]$. On the other hand, if hint_1 (i.e., w_1) appears in the list with index k, i.e., $w_1 = \mathsf{list}[k] = w_0^k$, then the list can also be generated from ftki_1 because $\forall j \in [|\mathsf{list}|]$:

$$\begin{aligned}
w_1^j &= w_1 \oplus (y^k \oplus y^j) \oplus \lceil \frac{q}{2}(s_1(x^k - x^j)) \rfloor_2 \\
&= w_0^k \oplus (y^k \oplus y^j) \oplus \lceil \frac{q}{2}(s_1(x^k - x^j)) \rfloor_2 \\
&= w_0 \oplus (y^i \oplus y^k) \oplus \lceil \frac{q}{2}(s_1(x^i - x^k)) \rfloor_2 \\
&\quad \oplus (y^k \oplus y^j) \oplus \lceil \frac{q}{2}(s_1(x^k - x^j)) \rfloor_2 \\
&= w_0 \oplus (y^i \oplus y^j) \oplus \lceil \frac{q}{2}(s_1(x^i - x^j)) \rfloor_2 \\
&= w_0^j,
\end{aligned}$$

which means ftki_0 and ftki_1 will generate exactly the same list. Particularly, there is

$$\frac{q}{2}(w_0 + y^i + s_1 x^i) = \frac{q}{2}(w_1 + y^j + s_1 x^j).$$

Now consider the hybrids. We have $\mathsf{ftki}_0 := (\mathbf{c}_1, c_2)$ and $\mathsf{ftki}_1 := (\mathbf{c}_1', c_2')$, specifically,

$$\begin{aligned}
\mathbf{c}_1 &= \mathbf{A}^T \mathbf{r} + \mathbf{e}_1 + \frac{q}{2} \cdot \begin{bmatrix} x^i \\ 0 \end{bmatrix}, c_2 = \mathbf{b}^T \mathbf{r} + e_2 + \frac{q}{2} \cdot (w_0 + y^i) \\
\mathbf{c}_1' &= \mathbf{A}^T \mathbf{r} + \mathbf{e}_1 + \frac{q}{2} \cdot \begin{bmatrix} x^j \\ 0 \end{bmatrix}, c_2' = \mathbf{b}^T \mathbf{r} + e_2 + \frac{q}{2} \cdot (w_1 + y^j).
\end{aligned}$$

Thus, there is

$$\begin{aligned}(\mathbf{c}_1, c_2) &\approx_c (\mathbf{u} + \frac{q}{2} \cdot \begin{bmatrix} x^i \\ 0 \end{bmatrix}, \mathbf{s}^T\mathbf{u} + \frac{q}{2} \cdot (w_0 + y^i) + e') \\ &\approx_s (\mathbf{u}', \mathbf{s}^T\mathbf{u}' + \frac{q}{2} \cdot (w_0 + y^i - \mathbf{s}^T \begin{bmatrix} x^i \\ 0 \end{bmatrix}) + e') \\ &= (\mathbf{u}', \mathbf{s}^T\mathbf{u}' + \frac{q}{2} \cdot (w_0 + y^i + s_1 x^i) + e') \\ &= (\mathbf{u}', \mathbf{s}^T\mathbf{u}' + \frac{q}{2} \cdot (w_1 + y^j + s_1 x^j) + e') \\ &\approx_c (\mathbf{c}_1', c_2'),\end{aligned}$$

where $e' = e_2 + \mathbf{e}^T\mathbf{r} - \mathbf{s}^T\mathbf{e}_1$ is the small noise term and $\mathbf{u}' = \mathbf{u} - \frac{q}{2} \cdot \begin{bmatrix} x^i \\ 0 \end{bmatrix}$. Note that $\frac{q}{2}(+s_1x^i) = \frac{q}{2}(-s_1x^i)$ mod q. Therefore, we have shown that ftki_0 and ftki_1 are indistinguishable for the adversary. □

Since ftki_b and ftki_{1-b} are indistinguishable and exchangeable for $\mathcal{A}$ in $\mathsf{UNLNK}_{\mathsf{fs}\mathsf{w-ke}}$. Now we show that $\mathcal{B}$ can fully simulate the $\mathsf{UNLNK}_{\mathsf{fs}\mathsf{w-ke}}$ experiment as follows. Upon receiving $\mathsf{mpk}_0, \mathsf{mpk}_1, \mathsf{opk}_b, \mathsf{tki}_b, \mathsf{osk}_b$ from $\mathsf{UNLNK}_{\mathsf{w-ke}}$, $\mathcal{B}$ sample $\mathsf{ftk}, \mathsf{fpk}$ then computes corresponding $\mathsf{ftki}_{b'}$ and list for $b' \leftarrow\$ \{0,1\}$, such that $w_0 \in \mathsf{list} \wedge w_1 \in \mathsf{list}$ where $\mathsf{H}(\mathsf{mpk}_0) = \mathsf{decode}_{R_q}(w_0)[: n]$ and $\mathsf{H}(\mathsf{mpk}_1) = \mathsf{decode}_{R_q}(w_1)[: n]$. Then $\mathcal{B}$ forwards all of them to $\mathcal{A}$ of $\mathsf{UNLNK}_{\mathsf{fs}\mathsf{w-ke}}$ and $\mathcal{A}$ cannot distinguish between ftki_0 or ftki_1 due to Claim 2. If $\mathcal{A}$ has non-negligible advantage $u(\lambda)$ in $\mathsf{UNLNK}_{\mathsf{fs}\mathsf{w-ke}}$, then $\mathcal{B}$ has the same non-negligible advantage $u(\lambda)$ in $\mathsf{UNLNK}_{\mathsf{w-ke}}$. □

We restate the theorem for $\mathsf{UNIUBS}_{\mathsf{fs}}$ formally here.

Theorem H.2. If there is $n \leq \frac{m}{2}$ where m is a power of 2, and B_η is a centered binomial distribution, then the scalable fuzzy tracking constructed in Figure 10 is information theoretically *unbiased* and satisfies $\mathsf{UNIUBS}_{\mathsf{fs}}$ defined in Definition 4.13.

Proof. If $\mathcal{A}$ is able to output valid mpk^i (i.e., valid hint^i and w^i), then for him, there is

$$w^j = w^i \oplus y^i \oplus y^j \oplus \lceil \frac{q}{2}(s_1(x^i - x^j)) \rfloor_2.$$

Note that the coefficients of $\lceil \frac{q}{2} s_1 \rfloor_2$ are uniformly random over $\{0,1\}^m$ because $s_1 \leftarrow\$ B_\eta$ where B_η is a centered binomial distribution. Moreover, since polynomial multiplication can be written as circular convolution, $\lceil \frac{q}{2}(s_1(x^i - x^j)) \rfloor_2$ can be written as $\mathbf{X}\mathbf{s} \mod 2$ where $\mathbf{s} \leftarrow \mathsf{decode}_{R_q}(\lceil \frac{q}{2} s_1 \rfloor_2)$ and $\mathbf{X}$ is the circulant matrix represented by the polynomial $x \leftarrow \lceil \frac{q}{2}(x^i - x^j) \rfloor_2$. Specifically, the first column of $\mathbf{X}$ is $\mathsf{decode}_{R_q}(x)$ and other columns are rotational shift of the previous column. Since m is a power of 2, it only has divisors from 2^0 to $2^{\log m}$. According to Lemma C.2 and Definition C.1, the biggest divisor of $X^m - 1$ is the polynomial $\Phi_m(X) = X^{\frac{m}{2}} + 1$ with degree $\frac{m}{2}$. Thus the rank of $\mathbf{X}$ is at least $m - \frac{m}{2}$ and at least a half of elements in $\mathbf{X}\mathbf{s} \mod 2$ are uniformly randomly distributed. This means $\mathsf{hint}^j \leftarrow \mathsf{decode}_{R_q}(w^j)[: n]$ is uniformly random as long as $n \leq \frac{m}{2}$.

□