Post-Quantum Adaptor Signatures and Payment Channel Networks

Muhammed F. Esgin[1,2(✉)], Oğuzhan Ersoy[3], and Zekeriya Erkin[3]

[1] Faculty of Information Technology, Monash University, Melbourne, Australia
muhammed.esgin@monash.edu
[2] Data61, CSIRO, Melbourne, Australia
[3] Cyber Security Group,
Delft University of Technology, Delft, Netherlands
{o.ersoy,z.erkin}@tudelft.nl

Abstract. Adaptor signatures, also known as *scriptless scripts*, have recently become an important tool in addressing the scalability and interoperability issues of blockchain applications such as cryptocurrencies. An adaptor signature extends a digital signature in a way that a complete signature reveals a secret based on a cryptographic condition. It brings about various advantages such as (i) low on-chain cost, (ii) improved fungibility of transactions, and (iii) advanced functionality beyond the limitation of the blockchain's scripting language.

In this work, we introduce the *first post-quantum* adaptor signature, named LAS. Our construction relies on the standard lattice assumptions, namely Module-SIS and Module-LWE. There are certain challenges specific to the lattice setting, arising mainly from the so-called *knowledge gap* in lattice-based proof systems, that makes the realization of an adaptor signature and its applications difficult. We show how to overcome these technical difficulties without introducing additional on-chain costs. Our evaluation demonstrates that LAS is essentially as efficient as an ordinary lattice-based signature in terms of both communication and computation. We further show how to achieve *post-quantum* atomic swaps and payment channel networks using LAS.

Keywords: Post-quantum · Blockchain · Lattice · Adaptor signature · Scriptless script · Payment channel network.

1 Introduction

Blockchains are decentralized platforms run by miners, where each transaction on the blockchain can be seen as an application formed of some script(s). The scripting language of a blockchain defines potential functionalities that can be implemented on blockchain. Bitcoin, for example, consists of very few scripts, which restricts its use mainly into coin transactions. Ethereum, on the other hand, has a Turing-complete scripting language that enables users to run more advanced and complicated applications.

L. Chen et al. (Eds.): ESORICS 2020, LNCS 12309, pp. 378–397, 2020.
https://doi.org/10.1007/978-3-030-59013-0_19

A user who wants to deploy and execute a transaction needs to pay a fee to the miners. The fee is determined by the storage and computational costs of running each script of the transaction. Thus, it is beneficial to handle some operations off-chain to reduce the on-chain fee paid to the miners. In this manner, Poelstra introduced the notion of *scriptless scripts* [25], which is later named as *adaptor signatures* [3,15].

Adaptor signatures can be seen as an extension over a digital signature, where first a "pre-signature" is generated and its completion to a (full) signature reveals a secret based on a cryptographic condition. The conditions are defined over a hard relation such as the discrete log problem, and the complete signature reveals a witness matching with the statement embedded into the pre-signature. The verification of the signature is done in the same way as the original signature scheme. Thus, while the miners verify only the signature, parties involved in the signature generation can embed an additional condition.

The main advantages of adaptor signatures can be summarized as follows: (i) A significant reduction in on-chain costs, (ii) improved fungibility of transactions, and (iii) ability to incorporate complex conditions, which may otherwise be impossible to execute due to the limitation of the blockchain's scripting language. More specifically, if the condition is published on-chain separately, then it would incur additional storage and verification costs. At the same time, since the condition is embedded inside a signature, for the outsiders and miners the signature with a condition is indistinguishable from a regular one. This fungibility property is especially useful to hide payment channel network transactions among any other transactions [21]. Moreover, adaptor signatures enhance the functionality of blockchains with a limited scripting language. Since the condition embedded within the signature is not verified by miners, it is not limited by the blockchain's scripting language. These advantages have been utilized in payment channel networks [3,21], atomic swaps [24], and discrete log contracts [8].

None of these works, however, provide security against powerful quantum computers as they rely on discrete-log-related assumptions. As evident, e.g., from NIST's efforts for standardization of *post-quantum* (i.e., quantum-resistant) algorithms [22], there is a major need for designing quantum-secure alternatives of currently deployed schemes. In fact, in the blockchain community, there are already significant efforts and considerations towards migrating to post-quantum cryptography. For example, Ethereum 2.0 Serenity upgrade [5] is planned to have an option for a post-quantum signature, Zcash developers plan to update their protocol with post-quantum alternatives when they are mature enough [31], and Hcash is building a post-quantum privacy-preserving blockchain [17].

Lattice-based cryptography, studied extensively in the last decades, is a promising candidate for post-quantum security. For example, Dilithium [9], which is based on standard lattice assumptions, is among the 2nd round signature candidates in NIST's post-quantum standardization process. Beyond basic cryptographic schemes such as encryption and signature, lattice-based cryptography also supports advanced schemes such as zero-knowledge proofs (ZKP), which play a crucial role in blockchain applications. For example, advanced ZKPs have recently been studied in [12,13] and there are even recent efforts in constructing blockchain-specific applications based on lattice assumptions [14,30].

Our contributions. In this work, we introduce the *first post-quantum* adaptor signature, LAS, in support of the efforts towards migration to post-quantum cryptography. Our construction relies on standard lattice assumptions, namely Module-LWE and Module-SIS, and is essentially as efficient as an ordinary lattice-based signature scheme based on the same assumptions. In particular, the signature scheme underlying LAS is a simplified version of Dilithium [9].

We further show how to realize post-quantum payment channel networks and atomic swaps using LAS. Our results show that these applications can be realized in the post-quantum setting without incurring an additional on-chain cost. The on-chain cost is effectively the cost of an ordinary lattice-based signature.

The main technical difficulties in constructing lattice-based adaptor signatures, as well as atomic swaps and payment channel networks, stem from the following two related facts. First, hard-to-find pre-images of lattice-based one-way functions, and in general user's secret keys, are required to have *small* coefficients in comparison to the system modulus q. In this case, a common technique used to hide user's secrets is rejection sampling, which is applied depending on the secret. As a result, in the setting of a payment channel network where a multi-party interaction is required with each user having his/her own secret, the realization of a secure construction demands a more careful analysis.

Secondly, *efficient* lattice-based zero-knowledge proofs underlying the (ordinary) signature scheme we employ have an inherent *knowledge (soundness) gap* (see, for example, [12,19,20]). That is, a witness extracted from a protocol interaction satisfies an *extended* relation R' whereas an honest user's secret satisfies a *stronger* relation R such that $R \subseteq R'$. Therefore, we need to adjust the security model carefully and also show that the extended guarantees are still meaningful and sufficient for practical applications. To this end, we extend the formal model of adaptor signatures introduced recently in [3], and show how to overcome the technical difficulties in our applications.

Organization of the paper. In Sect. 2, we present our security assumptions, lattice-based signatures and the rejection sampling technique as well as our extended formal definition for adaptor signatures. We introduce LAS, our adaptor signature, in Sect. 3, where the security and performance analyses and the effect of the knowledge gap are also given. We discuss the application of LAS to atomic swaps and payment channel networks in Sect. 4.

2 Preliminaries

We define $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$ to be a cyclotomic ring of power-of-2 degree d for an odd modulus q. We denote by $\mathbb{S}_c$ the set of polynomials in $\mathcal{R}_q$ whose maximum absolute coefficient is at most $c \in \mathbb{Z}^+$. Similarly, $\mathcal{R} = \mathbb{Z}[X]/(X^d + 1)$.

We denote by $\boldsymbol{I}_n$ the n-dimensional identity matrix. Vectors and matrices over $\mathcal{R}$ are denoted by lower-case and capital bold letters such $\boldsymbol{a}$ and $\boldsymbol{A}$, respectively. For a polynomial $f = f_0 + f_1 X + \cdots + f_{d-1} X^{d-1} \in \mathcal{R}$, we define the norms in the typical way: $\|f\| = \sqrt{\sum_{i=0}^{d-1} f_i^2}$, $\|f\|_\infty = \max_i |f_i|$ and $\|f\|_1 = \sum_{i=0}^{d-1} |f_i|$.

For a vector $\boldsymbol{v} = (v_0, \ldots, v_{s-1}) \in \mathcal{R}^s$ of polynomials with $s \geq 1$, we further define $\|\boldsymbol{v}\| = \sqrt{\sum_{i=0}^{s-1} \|v_i\|^2}$, $\|\boldsymbol{v}\|_1 = \sum_{i=0}^{s-1} \|v_i\|_1$, $\|\boldsymbol{v}\|_\infty = \max_i \|v_i\|_\infty$.

2.1 Security Assumptions: Module-SIS and Module-LWE

The security assumptions on which our constructions rely are the two well-known lattice problems, namely Module-SIS (M-SIS) and Module-LWE (M-LWE) [18]. They are generalizations of SIS [2] and LWE [28] problems, respectively. These problems are widely believed to resist attacks against powerful quantum adversaries. As in [9,12,13], we define below M-SIS in "Hermite normal form", which is as hard as M-SIS with a completely random matrix $\boldsymbol{A}$.

Definition 1 ($\mathbf{M\text{-}SIS}_{n,m,q,\beta_{\mathrm{SIS}}}$). *Let $\boldsymbol{A}' \xleftarrow{\$} \mathcal{R}_q^{n\times(m-n)}$ and $\boldsymbol{A} = [\boldsymbol{I}_n \,\|\, \boldsymbol{A}']$. Given $\boldsymbol{A}$, M-SIS problem with parameters $m > n > 0$ and $0 < \beta_{\mathrm{SIS}} < q$ asks to find a* short non-zero *$\boldsymbol{v} \in \mathcal{R}_q^m$ such that $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0}$ over $\mathcal{R}_q$ and $\|\boldsymbol{v}\| \leq \beta_{\mathrm{SIS}}$.*

We use a standard variant of M-LWE where both the error and secret coefficients are sampled uniformly from $\{-1, 0, 1\}$. This variant is commonly used in many recent proposals such as [12–14].

Definition 2 ($\mathbf{M\text{-}LWE}_{\ell,m,q}$). *M-LWE problem with parameters $\ell, m > 0$ asks to distinguish between the following two cases: 1) $(\boldsymbol{A}, \boldsymbol{b}) \xleftarrow{\$} \mathcal{R}_q^{m\times\ell} \times \mathcal{R}_q^m$, and 2) $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{s}+\boldsymbol{e})$ for $\boldsymbol{A} \xleftarrow{\$} \mathcal{R}_q^{m\times\ell}$, a secret vector $\boldsymbol{s} \xleftarrow{\$} \mathbb{S}_1^\ell$ and an error vector $\boldsymbol{e} \xleftarrow{\$} \mathbb{S}_1^m$.*

It is well-known that if the error and the secret coefficients are sampled from $\mathbb{S}_\gamma$ for $\gamma > 1$, then M-LWE problem gets harder. Therefore, $\text{M-LWE}_{\ell,m,q}$ hardness assumption implies that $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e}$ is (computationally) indistinguishable from a uniformly random element of $\mathcal{R}_q^m$ when $\boldsymbol{s} \xleftarrow{\$} \mathbb{S}_\gamma^\ell$ and $\boldsymbol{e} \xleftarrow{\$} \mathbb{S}_\gamma^m$ for any $\gamma \geq 1$.

2.2 Lattice-Based Signature and Rejection Sampling

The (ordinary) signature part of our construction can be seen as a simplified version of Dilithium [9], which is a 2nd round signature candidate in NIST's post-quantum standardization process. This signature scheme itself is based on Lyubashevsky's signatures [19,20]. In our construction, we do not employ the optimizations in Dilithium in order to simplify the presentation.

To make sure that the signature does not leak information about the secret key, we employ the rejection sampling technique from [19] as also done in Dilithium. The idea for this works as follows. Let $\boldsymbol{s} \in \mathcal{R}_q^k$ be a secret-dependant vector with $\|\boldsymbol{s}\|_\infty \leq p \in \mathbb{Z}^+$. In order to tie the security to M-SIS, we require the masked vector $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{s}$ to be short relative to q. Therefore, $\boldsymbol{y}$ cannot be sampled uniformly at random from $\mathcal{R}_q^k$. Instead, we sample $\boldsymbol{y} \xleftarrow{\$} \mathbb{S}_\gamma^k$ for $\gamma \approx kd \cdot p$. Then, we restart signing (i.e., reject $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{s}$) if $\|\boldsymbol{z}\|_\infty > \gamma - p$. It is easy to see that conditioned on $\boldsymbol{z}$ being accepted, the distribution of $\boldsymbol{z}$ is identical to the uniform distribution on $S_{\gamma-p}^k$. That is, the distribution of $\boldsymbol{z}$ is forced to be uniform in a box, and thus is (perfectly) simulatable using public information.

2.3 Adaptor Signatures

In [3], an adaptor signature $\Pi_{R,\Sigma}$ is defined with respect to a hard relation R and a signature scheme $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$. A relation R with a language $L_R := \{Y \mid \exists y : (Y, y) \in R\}$ is said to be hard [6] if: (i) there exists a probabilistic polynomial time (PPT) generator $\mathsf{Gen}(1^n)$ that outputs $(Y, y) \in R$, (ii) for every PPT algorithm $\mathcal{A}$, given $Y \in L_R$, the probability of $\mathcal{A}$ outputting y is negligible. A signature scheme Σ is defined by three algorithms: (i) KeyGen generates a public-secret key pair $(\mathsf{pk}, \mathsf{sk})$, (ii) Sign produces a signature σ using the key $(\mathsf{pk}, \mathsf{sk})$ and message M, (iii) Verify verifies the correctness of a signature σ on a message M using a public key pk. Our underlying signature, Dilithium [9], is SUF-CMA (Strong existential unforgeability under chosen message attacks) secure.

In the lattice setting, we need to define two relations R, R' with $R \subseteq R'$. Here, R constitutes the relation for the statement-witness pairs output by Gen (i.e., those used by honest users) whereas R' is an *extended* relation that defines the relation for *extracted* witnesses. The reason for this extension is detailed in Section 3, and stems from the *knowledge/soundness gap* inherent in *efficient* lattice-based zero-knowledge proofs (see, e.g., the soundness definition in [13, Section 2.3]). We denote an adaptor signature scheme in this setting by $\Pi_{R,R',\Sigma}$, which extends the definition given in [3], and elaborate further below the reason why this extension is necessary.

Definition 3 (Adaptor Signature Scheme). *An adaptor signature scheme $\Pi_{R,R',\Sigma}$ consists of four algorithms* $(\mathsf{PreSign}, \mathsf{PreVerify}, \mathsf{Adapt}, \mathsf{Ext})$ *defined below.*

$\mathsf{PreSign}((\mathsf{pk}, \mathsf{sk}), Y, M)$**:** *on input a key pair* $(\mathsf{pk}, \mathsf{sk})$*, a statement $Y \in L_R$ and a message $M \in \{0,1\}^*$, outputs a pre-signature $\hat{\sigma}$.*

$\mathsf{PreVerify}(Y, \mathsf{pk}, \hat{\sigma}, M)$**:** *on input a statement $Y \in L_R$, a pre-signature $\hat{\sigma}$, a public key* pk *and a message $M \in \{0,1\}^*$, outputs a bit b.*

$\mathsf{Adapt}((Y, y), \mathsf{pk}, \hat{\sigma}, M)$**:** *on input a statement-witness pair $(\hat{\sigma}, y)$, a public key* pk*, a pre-signature $\hat{\sigma}$ and a message $M \in \{0,1\}^*$, outputs a signature σ.*

$\mathsf{Ext}(Y, \sigma, \hat{\sigma})$**:** *on input a statement $Y \in L_R$, a signature σ and a pre-signature $\hat{\sigma}$, outputs a witness y such that $(Y, y) \in R'$, or $\perp$.*

Note that an adaptor signature $\Pi_{R,R',\Sigma}$ also inherits KeyGen, Sign and Verify algorithms from the signature scheme Σ. The authors in [3] define the security properties for an adaptor signature: $\mathsf{aEUF\text{-}CMA}$ security, pre-signature adaptability and witness extractability. In addition, they extend the standard correctness definition of signature algorithms with pre-signature correctness, which states that an honestly generated pre-signature of a statement $Y \in L_R$ passes $\mathsf{PreVerify}$ and can be completed into a signature where the witness y can be extracted. We extend further the formal definitions of the security properties in [3], where $R = R'$ yields the setting in [3].

Definition 4 ($\mathsf{aEUF\text{-}CMA}$ security). *An adaptor signature scheme $\Pi_{R,R',\Sigma}$ is* $\mathsf{aEUF\text{-}CMA}$ *secure if for every* PPT *adversary $\mathcal{A}$ there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that* $\Pr[\mathsf{aSignForge}_{\mathcal{A},\Pi_{R,R',\Sigma}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$*, where the experiment* $\mathsf{aSignForge}_{\mathcal{A},\Pi_{R,R',\Sigma}}$ *is defined as follows:*

$\mathsf{aSignForge}_{\mathcal{A},\Pi_{R,R',\Sigma}}(\lambda)$

$\mathcal{Q} := \emptyset$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
$M^* \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{S}}(\cdot),\mathcal{O}_{\mathrm{pS}}(\cdot,\cdot)}(\mathsf{pk})$
$(Y, y) \leftarrow \mathsf{Gen}(1^\lambda)$
$\hat{\sigma} \leftarrow \mathsf{PreSign}((\mathsf{pk}, \mathsf{sk}), Y, M^*)$
$\sigma \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{S}}(\cdot),\mathcal{O}_{\mathrm{pS}}(\cdot,\cdot)}(\hat{\sigma}, Y)$
return $(M^* \notin \mathcal{Q} \wedge \mathsf{Verify}(\mathsf{pk}, \sigma, M^*))$

$\mathcal{O}_{\mathrm{S}}(M)$

$\sigma \leftarrow \mathsf{Sign}((\mathsf{pk}, \mathsf{sk}), M)$
$\mathcal{Q} := \mathcal{Q} \cup \{M\}$
return σ

$\mathcal{O}_{\mathrm{pS}}(M, Y)$

$\hat{\sigma} \leftarrow \mathsf{PreSign}((\mathsf{pk}, \mathsf{sk}), Y, M)$
$\mathcal{Q} := \mathcal{Q} \cup \{M\}$
return $\hat{\sigma}$

Definition 5 (Weak pre-signature adaptability). *An adaptor signature scheme* $\Pi_{R,R',\Sigma}$ *is* weak pre-signature adaptable *if for any message* $M \in \{0,1\}^*$*, any statement/witness pair* $(Y, y) \in R$*, any key pair* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and any pre-signature* $\hat{\sigma} \leftarrow \{0,1\}^*$ *with* $\mathsf{PreVerify}(Y, \mathsf{pk}, \hat{\sigma}, M) = 1$*, we have* $\Pr[\mathsf{Verify}(\mathsf{pk}, \mathsf{Adapt}((Y, y), \mathsf{pk}, \hat{\sigma}, M), M) = 1] = 1$.

We call our pre-signature adaptability definition *weak* because only statement-witness pairs satisfying R are guaranteed to be adaptable, and not those satisfying R'. This is similar to the *knowledge gap* of the ZKP underlying Dilithium, where the soundness only guarantees extraction of a witness from an *extended* relation. Therefore, pre-signature adaptability does not guarantee, for example, that an *extracted* witness can be used to adapt a pre-signature successfully (see Remark 1). This issue becomes effective in the applications of our adaptor signature, and we show how to overcome it in Section 4. Note that still the pre-signature $\hat{\sigma}$ in the above definition can be adversarially generated as in [3].

Definition 6 (Witness extractability). *An adaptor signature scheme* $\Pi_{R,R',\Sigma}$ *is* witness extractable *if for every* PPT *adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that the following holds:* $\Pr[\mathsf{aWitExt}_{\mathcal{A},\Pi_{R,R',\Sigma}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$*, where the experiment* $\mathsf{aWitExt}_{\mathcal{A},\Pi_{R,R',\Sigma}}$ *is defined as follows*

$\mathsf{aWitExt}_{\mathcal{A},\Pi_{R,R',\Sigma}}(\lambda)$

$\mathcal{Q} := \emptyset$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
$(M^*, Y) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{S}}(\cdot),\mathcal{O}_{\mathrm{pS}}(\cdot,\cdot)}(\mathsf{pk})$
$\hat{\sigma} \leftarrow \mathsf{PreSign}((\mathsf{pk}, \mathsf{sk}), Y, M^*)$
$\sigma \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{S}}(\cdot),\mathcal{O}_{\mathrm{pS}}(\cdot,\cdot)}(\hat{\sigma})$
$y' := \mathsf{Ext}(Y, \sigma, \hat{\sigma})$
return $(M^* \notin \mathcal{Q} \wedge (Y, y') \notin R'$
$\wedge\, \mathsf{Verify}(\mathsf{pk}, \sigma, M^*))$

$\mathcal{O}_{\mathrm{S}}(M)$

$\sigma \leftarrow \mathsf{Sign}((\mathsf{pk}, \mathsf{sk}), M)$
$\mathcal{Q} := \mathcal{Q} \cup \{M\}$
return σ

$\mathcal{O}_{\mathrm{pS}}(M, Y)$

$\hat{\sigma} \leftarrow \mathsf{PreSign}((\mathsf{pk}, \mathsf{sk}), Y, M)$
$\mathcal{Q} := \mathcal{Q} \cup \{M\}$
return $\hat{\sigma}$

Note that, in the above witness extractability definition, the adversary's winning condition is restricted to the extracted witness not being in R'.

Since $R \subseteq R'$, $(Y, y') \notin R'$ implies that $(Y, y') \notin R$. Therefore, it is sufficient to ensure that R' is a hard relation, which itself implies that R is also a hard relation. As a result, in our security assumptions, we make sure that R' is a hard relation.

3 LAS: An Efficient Adaptor Signature from Lattices

In this section, we describe our lattice-based adaptor signature, LAS. Let $\boldsymbol{A} = [\boldsymbol{I}_n \,\|\, \boldsymbol{A}'] \in R_q^{n\times(n+\ell)}$ for $\boldsymbol{A}' \xleftarrow{\$} R_q^{n\times\ell}$ and $\mathsf{H} : \{0,1\}^* \to \mathcal{C}$ be a hash function (modelled as a random oracle). We assume that the public parameters $pp = (\boldsymbol{A}, \mathsf{H})$ are publicly available and can be used by any algorithm. In practice, $\boldsymbol{A}'$ can be generated from a small seed using an extendable output function (modelled as a random oracle) as done in Dilithium [9]. The function $f_{\boldsymbol{A}}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}$ over $\mathcal{R}_q$ is Ajtai's hash function [2] defined over module lattices where the matrix $\boldsymbol{A}$ is in Hermite normal form (HNF). It is clear that the function is additively homomorphic, and Ajtai [2] showed that it is one-way in the setting of SIS. In our case, the security is based on M-SIS (in HNF). Collision-resistance is also clear as a collision $(\boldsymbol{x}, \boldsymbol{x}')$ yields an immediate M-SIS solution: $\boldsymbol{A}(\boldsymbol{x}-\boldsymbol{x}') = 0$.

In Table 1, we first summarize the identifiers used for LAS, where the hard relations R, R' are given by $\mathfrak{R}_{\boldsymbol{A}}, \mathfrak{R}'_{\boldsymbol{A}}$ with $\mathfrak{R}_{\boldsymbol{A}} \subseteq \mathfrak{R}'_{\boldsymbol{A}}$. The statement-witness generation Gen for $\mathfrak{R}_{\boldsymbol{A}}$ runs exactly as KeyGen. It is easy to see that if M-SIS$_{n,n+\ell+1,q,\beta}$ for $\beta = 2\gamma d(n+\ell)$ is hard, then $\mathfrak{R}_{\boldsymbol{A}}$ and $\mathfrak{R}'_{\boldsymbol{A}}$ are hard relations. This is because if one can find $\boldsymbol{r}$ such that $(\boldsymbol{t}, \boldsymbol{r}) \in \mathfrak{R}'_{\boldsymbol{A}}$ for a random $\boldsymbol{t}$, then $[\boldsymbol{A} \,\|\, \boldsymbol{t}] \cdot \begin{pmatrix} \boldsymbol{r} \\ -1 \end{pmatrix} = 0$. Hence, $\begin{pmatrix} \boldsymbol{r} \\ -1 \end{pmatrix}$ is a solution to M-SIS$_{n,n+\ell+1,q,\beta}$ for $\beta = 2\gamma d(n+\ell)$ since $\|\boldsymbol{r}\| \leq \beta$.

Table 1. Identifiers for LAS.

Notation	Explanation	Value
d	a power-of-2 ring dimension	256
$\mathcal{R}_q$	cyclotomic ring of degree d: $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d+1)$	$\log q \approx 24$
$\mathbb{S}_c$	the set of polynomials $f \in \mathcal{R}_q$ with $\|f\|_\infty \leq c$ for $c \in \mathbb{Z}^+$	
n	M-SIS rank	4
ℓ	M-LWE rank	4
$\mathcal{C}$	the challenge set and range of H: $\{c \in \mathcal{R} : \|c\|_1 = \kappa \ \wedge \ \|c\|_\infty = 1\}$	$\kappa = 60$
γ	maximum absolute coefficient of a masking randomness	$\kappa d(n+\ell)$
$(Y, y) \in \mathfrak{R}_{\boldsymbol{A}}$	the base relation with $[\boldsymbol{I}_n \,\|\, \boldsymbol{A}'] = \boldsymbol{A} \in \mathcal{R}_q^{n\times(n+\ell)}$: $(Y, y) = (\boldsymbol{t}, \boldsymbol{r}) \in \mathfrak{R}_{\boldsymbol{A}}$ if $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{r}$ and $\|\boldsymbol{r}\|_\infty \leq 1$	
$(Y, y) \in \mathfrak{R}'_{\boldsymbol{A}}$	the extended relation with $[\boldsymbol{I}_n \,\|\, \boldsymbol{A}'] = \boldsymbol{A} \in \mathcal{R}_q^{n\times(n+\ell)}$: $(Y, y) = (\boldsymbol{t}, \boldsymbol{r}) \in \mathfrak{R}_{\boldsymbol{A}}$ if $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{r}$ and $\|\boldsymbol{r}\|_\infty \leq 2(\gamma - \kappa)$	$\gamma > \kappa$

Algorithm 1. Lattice-Based Signature

procedure KeyGen(): ▷ same as Gen
  $\boldsymbol{r} \xleftarrow{\$} \mathbb{S}_1^{n+\ell}$
  $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{r}$
  return $(\mathsf{pk}, \mathsf{sk}) = (\boldsymbol{t}, \boldsymbol{r})$
end procedure

procedure Sign($(\mathsf{pk}, \mathsf{sk}), M$):
  $\boldsymbol{y} \xleftarrow{\$} \mathbb{S}_\gamma^{n+\ell}$
  $\boldsymbol{w} = \boldsymbol{A}\boldsymbol{y}$
  $c = \mathsf{H}(\mathsf{pk}, \boldsymbol{w}, M)$
  $\boldsymbol{z} = \boldsymbol{y} + c\boldsymbol{r}$ where $\boldsymbol{r} := \mathsf{sk}$
  if $\|\boldsymbol{z}\|_\infty > \gamma - \kappa$, **then** Restart
  return $\sigma = (c, \boldsymbol{z})$
end procedure

procedure Verify(pk, σ, M):
  Parse $(c, \boldsymbol{z}) := \sigma$
  if $\|\boldsymbol{z}\|_\infty > \gamma - \kappa$, **then return** 0
  $\boldsymbol{w}' = \boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t}$ where $\boldsymbol{t} := \mathsf{pk}$
  if $c \neq \mathsf{H}(\mathsf{pk}, \boldsymbol{w}', M)$, **then return** 0
  return 1
end procedure

We present the ordinary signature procedures in Algorithm 1, and then the procedures for the adaptor signature in Algorithm 2. The idea for the signature is similar to the Schnorr signature [29] with the main difference being the use of rejection sampling at Step 11. This is the so-called "Fiat-Shamir with Aborts" technique [19,20].

In the adaptor signature part in Algorithm 2, PreSign and PreVerify operate very similar to Sign and Verify, respectively. The main issue is that the signer may not know (at the time of running PreSign) the witness y to the statement Y, and yet for many applications in practice (such as payment channel networks), one would want to make sure that having access only to the signature (but not the pre-signature) does not reveal any information on the witness y.

To this end, we need to modify the rejection sampling step. Even though the signer does not know the witness y, he does know how it is supposed to be generated in an honest run. Therefore, he knows that the maximum absolute coefficient of any honestly-generated witness is at most 1 (recall that Gen runs exactly as KeyGen). Since we have $\boldsymbol{z} = \boldsymbol{y} + c\boldsymbol{r} + \boldsymbol{r}'$ for $\boldsymbol{r}' := y$ in an honestly-generated full signature, we know that the secret-dependant part $c\boldsymbol{r} + \boldsymbol{r}'$ has infinity norm at most $\kappa + 1$. Therefore, the signer artificially performs a stronger rejection sampling step in PreSign, where $\|\hat{\boldsymbol{z}}\|_\infty \leq \gamma - \kappa - 1$ is required. This ensures that even when the witness is added to the response in Adapt, the response $\boldsymbol{z}$ still satisfies the rejection sampling condition in Sign, and thus remains publicly simulatable, i.e., no secret information including the witness is revealed.

In fact, there are further reasons for this important modification. One is in regards to adaptability. If the rejection sampling in PreSign is done exactly as in Sign, then verification of an adapted pre-signature (i.e., output of Adapt) via Verify may not succeed as the infinity norm condition may be violated due to the addition of $\boldsymbol{r}' := y$. Another reason comes from the security analysis. In order to be able to simulate the outputs of both Sign and PreSign, this change to rejection sampling plays a crucial role.

Algorithm 2. LAS: Lattice-Based Adaptor Signature

procedure $\mathsf{PreSign}((\mathsf{pk},\mathsf{sk}), Y, M)$:
  $\boldsymbol{y} \xleftarrow{\$} \mathbb{S}_{\gamma}^{n+\ell}$
  $\boldsymbol{w} = \boldsymbol{A}\boldsymbol{y}$
  $c = \mathsf{H}(\mathsf{pk}, \boldsymbol{w} + \boldsymbol{t}', M)$ for $\boldsymbol{t}' := Y$
  $\hat{\boldsymbol{z}} = \boldsymbol{y} + c\boldsymbol{r}$ where $\boldsymbol{r} := \mathsf{sk}$
  if $\|\hat{\boldsymbol{z}}\|_\infty > \gamma - \kappa - 1$, **then** Restart
  return $\hat{\sigma} = (c, \hat{\boldsymbol{z}})$
end procedure

procedure $\mathsf{PreVerify}(Y, \mathsf{pk}, \hat{\sigma}, M)$:
  Parse $(c, \hat{\boldsymbol{z}}) := \hat{\sigma}$ and $\boldsymbol{t}' := Y$
  if $\|\hat{\boldsymbol{z}}\|_\infty > \gamma - \kappa - 1$ **then**
    return 0
  end if
  $\boldsymbol{w}' = \boldsymbol{A}\hat{\boldsymbol{z}} - c\boldsymbol{t}$ where $\boldsymbol{t} := \mathsf{pk}$
  if $c \neq \mathsf{H}(\mathsf{pk}, \boldsymbol{w}' + \boldsymbol{t}', M)$ **then**
    return 0
  end if
  return 1
end procedure

procedure $\mathsf{Adapt}((Y, y), \mathsf{pk}, \hat{\sigma}, M)$:
  if $\mathsf{PreVerify}(Y, \mathsf{pk}, \hat{\sigma}, M) = 0$ **then**
    return $\perp$
  end if
  Parse $(c, \hat{\boldsymbol{z}}) := \hat{\sigma}$ and $\boldsymbol{r}' := y$
  return $\sigma = (c, \hat{\boldsymbol{z}} + \boldsymbol{r}')$
end procedure

procedure $\mathsf{Ext}(Y, \sigma, \hat{\sigma})$:
  Parse $(c, \boldsymbol{z}) := \sigma$ and $(\hat{c}, \hat{\boldsymbol{z}}) := \hat{\sigma}$
  Parse $\boldsymbol{t}' := Y$
  $\boldsymbol{s} = \boldsymbol{z} - \hat{\boldsymbol{z}}$
  if $\boldsymbol{t}' \neq \boldsymbol{A}\boldsymbol{s}$, **then return** $\perp$
  return $\boldsymbol{s}$
end procedure

Let us summarize the following two facts as we will make use of them repeatedly in the security proofs.

Fact 1. *We can see that* $\|c\boldsymbol{r}\|_\infty \leq \kappa$ *since* $\|c\|_1 \leq \kappa$ *and* $\|\boldsymbol{r}\|_\infty \leq 1$. *Therefore, both* $\hat{\boldsymbol{z}}$ *in* PreSign *and* $\boldsymbol{z}$ *in* Sign *can be simulated publicly as they follow uniform distributions on* $\mathbb{S}_{\gamma-\kappa-1}^{n+\ell}$ *and* $\mathbb{S}_{\gamma-\kappa}^{n+\ell}$, *respectively, due to the rejection sampling.*

Fact 2. *Assuming the hardness of M-LWE*$_{\ell,n,q}$, *the result of* $\boldsymbol{A}\boldsymbol{x}$ *is (computationally) indistinguishable from a uniformly random element in* $\mathcal{R}_q^n$ *whenever* $\boldsymbol{x} \xleftarrow{\$} \mathbb{S}_c^{n+\ell}$ *for some* $c \geq 1$. *We can see this by realizing that* $\boldsymbol{A}\boldsymbol{x} = [\,\boldsymbol{I}_n \,\|\, \boldsymbol{A}'\,] \cdot \begin{pmatrix} \boldsymbol{x}_0 \\ \boldsymbol{x}_1 \end{pmatrix} = \boldsymbol{x}_0 + \boldsymbol{A}'\boldsymbol{x}_1$. *This is an M-LWE instance with the secret vector* $\boldsymbol{x}_1 \in \mathbb{S}_c^\ell$ *and the error vector* $\boldsymbol{x}_0 \in \mathbb{S}_c^n$.

Note that there is a *knowledge gap* between a witness used by an honest user and a witness extracted by Ext for a statement Y. In particular, an honest user's witness $y = \boldsymbol{r}$ satisfies $\|\boldsymbol{r}\|_\infty \leq 1$ (i.e., $(Y, y) \in \mathfrak{R}_{\boldsymbol{A}}$), whereas an extracted witness $y' = \boldsymbol{r}'$ is only guaranteed to satisfy $\|\boldsymbol{r}'\|_\infty \leq 2(\gamma - \kappa)$ (i.e., $(Y, y') \in \mathfrak{R}'_{\boldsymbol{A}}$). Such a knowledge gap is inherent in the existing *efficient* lattice-based zero-knowledge proofs such as the one underlying Dilithium. However, we emphasize that this knowledge gap does not raise a security concern as our hardness assumptions require that finding even a witness as big as an extracted witness is still hard, which itself implies that finding an honest user's witness is also hard. In the next section, we study the security aspects more rigorously.

3.1 Security Analysis

Pre-signature correctness follows via a straightforward investigation. In the following sequence of lemmas, we prove the security properties.

Lemma 1 (Weak pre-signature adaptability). LAS *satisfies weak pre-signature adaptability with respect to the relation* $\mathfrak{R}_{\boldsymbol{A}}$ *given in Table 1.*

Proof. Let $\hat{\sigma} = (c, \hat{\boldsymbol{z}})$ be a valid pre-signature with $\mathsf{PreVerify}(Y, \mathsf{pk}, \hat{\sigma}, M) = 1$ and $y = \boldsymbol{r}' \in \mathbb{S}_1^{n+\ell}$ be a witness corresponding to Y. Note that $\|\hat{\boldsymbol{z}}\|_\infty \leq \gamma - \kappa - 1$ since $\hat{\sigma}$ is valid. Then, $\mathsf{Adapt}((Y, y), \mathsf{pk}, \hat{\sigma}, M) = (c, \hat{\boldsymbol{z}} + \boldsymbol{r}') =: (c, \boldsymbol{z}) = \sigma$. Now, we have

$$\|\boldsymbol{z}\|_\infty = \|\hat{\boldsymbol{z}} + \boldsymbol{r}'\|_\infty \leq \|\hat{\boldsymbol{z}}\|_\infty + \|\boldsymbol{r}'\|_\infty = (\gamma - \kappa - 1) + 1 = \gamma - \kappa. \tag{1}$$

We further have

$$\begin{aligned} \mathsf{H}(\mathsf{pk}, \boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t}, M) &= \mathsf{H}(\mathsf{pk}, \boldsymbol{A}(\hat{\boldsymbol{z}} + \boldsymbol{r}') - c\boldsymbol{t}, M) = \mathsf{H}(\mathsf{pk}, \boldsymbol{A}\hat{\boldsymbol{z}} - c\boldsymbol{t} + \boldsymbol{A}\boldsymbol{r}', M) \\ &= \mathsf{H}(\mathsf{pk}, \boldsymbol{A}\hat{\boldsymbol{z}} - c\boldsymbol{t} + \boldsymbol{t}', M) = c. \end{aligned} \tag{2}$$

From (1) and (2), it follows that σ is valid, i.e., $\mathsf{Verify}(\mathsf{pk}, \sigma, M) = 1$. □

Remark 1. Observe in the proof of Lemma 1 that we crucially rely on the fact that for a witness $y = \boldsymbol{r}'$ in $\mathfrak{R}_{\boldsymbol{A}}$, we have $\|\boldsymbol{r}'\|_\infty \leq 1$. An *extracted* witness $\boldsymbol{s}$ does not necessarily obey this rule as the relation $\mathfrak{R}'_{\boldsymbol{A}}$ only requires $\|\boldsymbol{s}\|_\infty \leq 2(\gamma - \kappa)$. Therefore, extra care needs to be taken when dealing with the cases where an extracted witness is used to adapt a pre-signature.

Lemma 2 (Witness extractability). *If M-LWE*$_{\ell,n,q}$ *and M-SIS*$_{n,n+\ell+1,q,\beta}$ *for* $\beta = 2\gamma\sqrt{d(n+\ell)}$ *are hard, then* LAS *is witness extractable in the random oracle model.*

Proof. Here, we only investigate the case that the signature output by the adversary shares the same challenge with the pre-signature. The other case (where the two challenges are distinct) can be proven exactly as in **Case 2** of the proof of Lemma 3 because how Y is generated is irrelevant for that case.

For a given pair of public key and statement $(\mathsf{pk}, Y) = (\boldsymbol{t}, \boldsymbol{t}')$ and a message M, let $\hat{\sigma} = (c, \hat{\boldsymbol{z}})$ and $\sigma = (c, \boldsymbol{z})$ be a valid pre-signature and a valid signature, respectively. Then, from the corresponding verification algorithms (i.e., Verify and $\mathsf{PreVerify}$), we have $\mathsf{H}(\mathsf{pk}, \boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t}, M) = \mathsf{H}(\mathsf{pk}, \boldsymbol{A}\hat{\boldsymbol{z}} - c\boldsymbol{t} + \boldsymbol{t}', M)$. Since H is modelled as a random oracle, this holds only when $\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t} = \boldsymbol{A}\hat{\boldsymbol{z}} - c\boldsymbol{t} + \boldsymbol{t}'$, which implies that $\boldsymbol{A}\boldsymbol{z} - \boldsymbol{A}\hat{\boldsymbol{z}} = \boldsymbol{A}(\boldsymbol{z} - \hat{\boldsymbol{z}}) = \boldsymbol{t}'$. It is easy to see that $\|\boldsymbol{z} - \hat{\boldsymbol{z}}\|_\infty \leq 2(\gamma - \kappa)$. Therefore, for the output $\boldsymbol{s} = \boldsymbol{z} - \hat{\boldsymbol{z}}$ of $\mathsf{Ext}(Y, \sigma, \hat{\sigma})$, we have $(\boldsymbol{t}', \boldsymbol{s}) \in \mathfrak{R}'_{\boldsymbol{A}}$. Note also that $\boldsymbol{s}$ is non-zero since $\boldsymbol{t}'$ is non-zero except for a negligible probability. □

Lemma 3 (Unforgeability). *If M-SIS*$_{n,n+\ell+1,q,\beta}$ *for* $\beta = 2\gamma\sqrt{d(n+\ell)}$ *and M-LWE*$_{\ell,n,q}$ *are hard, then* LAS *is* aEUF-CMA *secure in the random oracle model.*

Proof. First, from the assumptions in the statement, we know that

1. both $\mathfrak{R}_{\boldsymbol{A}}$ and $\mathfrak{R}'_{\boldsymbol{A}}$ are hard relations,
2. any public key output by KeyGen and any statement output by Gen is indistinguishable from a uniformly random element in $\mathcal{R}_q^n$ due to Fact 2.

Let $\mathcal{F}$ be a PPT adversary who wins the aEUF-CMA security game with non-negligible probability. We will build an adversary $\mathcal{S}$ that solves M-SIS$_{n,n+\ell+1,q,\beta}$. Let $\beta = 2\gamma\sqrt{d(n+\ell)}$ and $\boldsymbol{B} = [\,\boldsymbol{I}_n \,\|\, \boldsymbol{A}' \,\|\, \boldsymbol{a}\,] \in \mathcal{R}_q^{n\times(n+\ell+1)}$ for $\boldsymbol{A}' \xleftarrow{\$} \mathcal{R}_q^{n\times\ell}$ and $\boldsymbol{a} \xleftarrow{\$} \mathcal{R}_q^n$. Assume that $\mathcal{S}$ wants to solve M-SIS w.r.t. $\boldsymbol{B}$. Let $\boldsymbol{A}$ denote $[\,\boldsymbol{I}_n \,\|\, \boldsymbol{A}'\,]$.

Setup. $\mathcal{S}$ sets $\boldsymbol{A}$ together with some hash function H as the public parameters. It is clear that $\boldsymbol{A}$ has the correct distribution. Then, it sets $\mathsf{pk} = \boldsymbol{t} = \boldsymbol{B}\boldsymbol{r}$ where $\boldsymbol{r} = \begin{pmatrix} \boldsymbol{r}' \\ 1 \end{pmatrix}$ for $\boldsymbol{r}' \xleftarrow{\$} \mathbb{S}_1^{n+\ell}$. $\mathcal{S}$ sends pk to $\mathcal{F}$. By M-LWE$_{\ell,n,q}$, pk is indistinguishable from a public key output by KeyGen since $\boldsymbol{B}\boldsymbol{r} = \boldsymbol{A}\boldsymbol{r}' + \boldsymbol{a}$ looks uniformly random as $\boldsymbol{A}\boldsymbol{r}'$ does. Note also that $\boldsymbol{t} = \mathsf{pk}$ is non-zero with overwhelming probability.

Oracle simulation. For $\mathcal{O}_{\mathrm{S}}(M)$, $\mathcal{S}$ picks $\boldsymbol{z} \xleftarrow{\$} \mathbb{S}_{\gamma-\kappa}^{n+\ell}$ and $c \xleftarrow{\$} \mathcal{C}$, and programs the random oracle such that $c = \mathsf{H}(\mathsf{pk}, \boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t}, M)$. If the input of H has been queried before, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ returns $\sigma = (c, \boldsymbol{z})$. The simulated output is indistinguishable from a real one due to Fact 1.

For $\mathcal{O}_{\mathrm{pS}}(M, Y)$, the simulator picks $\hat{\boldsymbol{z}} \xleftarrow{\$} \mathbb{S}_{\gamma-\kappa-1}^{n+\ell}$ and $c \xleftarrow{\$} \mathcal{C}$, and programs the random oracle such that $c = \mathsf{H}(\mathsf{pk}, \boldsymbol{A}\hat{\boldsymbol{z}} - c\boldsymbol{t} + \boldsymbol{t}', M)$ for $\boldsymbol{t}' := Y$. If the input of H has been queried before, $\mathcal{S}$ aborts. Otherwise, the simulator returns $\hat{\sigma} = (c, \hat{\boldsymbol{z}})$. The simulated output is indistinguishable from a real one due to Fact 1.

In both cases, the probability of an abort is negligible as $\mathcal{F}$ can make at most polynomially many queries to H.

Forgery. $\mathcal{F}$ returns the target message M^* to $\mathcal{S}$. $\mathcal{S}$ sets $Y = -\boldsymbol{a}$ and computes a pre-signature $\hat{\sigma}^* = (c^*, \hat{\boldsymbol{z}}^*)$ using the simulation method above. $\mathcal{S}$ sends $(Y, \hat{\sigma}^*)$ to $\mathcal{F}$. Again, note that Y is indistinguishable from a real output by Gen, and $\hat{\sigma}^*$ is indistinguishable from a real output of PreSign. Finally, $\mathcal{F}$ returns a forged signature $\sigma = (c, \boldsymbol{z})$ on M^*.

Case 1 ($c^* = c$) : If this is the case, then as shown in the proof of Lemma 2, $\mathcal{S}$ can extract a witness to $\mathfrak{R}'_{\boldsymbol{A}}$. That is, $\mathcal{S}$ gets $(Y, y) \in \mathfrak{R}'_{\boldsymbol{A}}$ with $\boldsymbol{s}' := y$, which implies that $\boldsymbol{A}\boldsymbol{s}' = -\boldsymbol{a}$ (since $Y = -\boldsymbol{a}$) and $\|\boldsymbol{s}'\|_\infty \leq 2(\gamma - \kappa)$. This is equivalent to $\boldsymbol{B}\boldsymbol{s} = \boldsymbol{0}$ for $\boldsymbol{s} = \begin{pmatrix} \boldsymbol{s}' \\ 1 \end{pmatrix}$. Note that $\|\boldsymbol{s}\| \leq \beta$. Hence, $\mathcal{S}$ finds a solution to M-SIS$_{n,n+\ell+1,q,\beta}$.

Case 2 ($c^* \neq c$) : In this case, we know that the forged signature's challenge comes from a random oracle query output (with overwhelming probability). Therefore, we can use a standard rewinding argument as in [26], where $\mathcal{S}$ rewinds $\mathcal{F}$ to get another forgery $\sigma' = (c', \boldsymbol{z}')$ such that $c' \neq c$ and $\mathsf{H}(\mathsf{pk}, \boldsymbol{A}\boldsymbol{z}' - c'\boldsymbol{t}, M^*) = \mathsf{H}(\mathsf{pk}, \boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t}, M^*)$. Therefore, we have

$$\boldsymbol{A}\boldsymbol{z}' - c'\boldsymbol{t} = \boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t} \quad \Longleftrightarrow \quad \boldsymbol{A}\,(\boldsymbol{z}' - \boldsymbol{z}) = (c' - c)\boldsymbol{t}. \tag{3}$$

Since $c' \neq c$, we have $\boldsymbol{z}' - \boldsymbol{z} \neq 0$. The above equation (3) can be equivalently written as

$$\boldsymbol{B}\begin{pmatrix} \boldsymbol{z}' - \boldsymbol{z} \\ 0 \end{pmatrix} = (c' - c)\boldsymbol{t}. \tag{4}$$

Now recalling that $\boldsymbol{t} = \boldsymbol{B}\boldsymbol{r}$, we also have

$$(c' - c)\boldsymbol{t} = \boldsymbol{B} \cdot (c' - c)\boldsymbol{r}. \tag{5}$$

Subtracting (3) from (5), we get

$$\boldsymbol{B}\left[(c' - c)\boldsymbol{r} - \begin{pmatrix} \boldsymbol{z}' - \boldsymbol{z} \\ 0 \end{pmatrix}\right] = \boldsymbol{0}. \tag{6}$$

Recalling that the last coordinate of $\boldsymbol{r}$ is 1, i.e., non-zero, the above gives a non-trivial solution to M-SIS$_{n,n+\ell+1,q,\beta}$. Here note that $\|\boldsymbol{z}' - \boldsymbol{z}\| \leq 2(\gamma - \kappa)\sqrt{d(n+\ell)} < \beta$ and $\|(c' - c)\boldsymbol{r}\| \leq 2\kappa\sqrt{d(n+\ell+1)}$. Since $\gamma \gg \kappa$, the total norm of the M-SIS solution remains below $\beta = 2\gamma\sqrt{d(n+\ell)}$. □

3.2 Parameter Setting and Performance Analysis

First, we set $\gamma = \kappa d(n+\ell)$ so that the average number of restarts in Sign and PreSign is about $e < 3$. Then, we set $d = 256$ and $\kappa = 60$, which ensures that the challenge set $\mathcal{C}$ has more than 2^{256} elements. Finally, in order to meet the M-SIS$_{n,n+\ell+1,q,\beta}$ and M-LWE$_{\ell,n,q}$ security requirements for $\beta = 2\gamma\sqrt{d(n+\ell)}$, we set $n = \ell = 4$ and $q \approx 2^{24}$. Only the size of the modulus q is important, and therefore the concrete value can be chosen to allow fast computation such as Number Theoretic Transformation (NTT).

In estimating the practical security of M-SIS and M-LWE, we follow the methodology outlined in [10, Section 3.2.4] and measure the practical hardness in terms of "root Hermite factor" δ. This parameter setting yields $\delta < 1.0045$ for both M-SIS and M-LWE. $\delta \approx 1.0045$ has been used in recent works, e.g., [12–14] for targeting 128-bit post-quantum security. From here, we can compute the concrete signature length as

$$|\sigma| = d(n+\ell)\log(2\gamma)/8 + 32 \text{ bytes} \approx 3210 \text{ bytes}. \tag{7}$$

This length is slightly larger than the size of Dilithium (2701 bytes) [9] with recommended parameters. The main reason is because we do not employ the optimizations for ease of presentation.

In terms of the computational efficiency, the operations performed in LAS are almost identical to those in Dilithium. Thus, hundreds of signing (and even more verification) can be done per second on a standard PC as shown in [9, Table 2].

4 Applications

In this section, we present two blockchain applications of our adaptor signature, namely atomic swaps and payment channel networks. To match with the existing adaptor signature applications, we assume an Unspent Transaction Output (UTXO)-based blockchain like Bitcoin where the signature algorithm is replaced with a lattice-based signature scheme given in Algorithm 1 . In the UTXO model, coins are kept in addresses where each address consists of the amount and the spending condition. The spending condition is defined by the scripting language and the most common ones are signature and hash preimage verifications, and timing conditions. For our applications, we also assume that the underlying blockchain supports these scripts.

4.1 Atomic Swaps

An atomic swap can be defined between two users u_1 and u_2 who want to exchange two different cryptocurrencies c_1 and c_2. The crucial point of the exchange is ensuring fairness, i.e., either both parties receive their expected output or none do. In [23], an atomic swap protocol is presented with the following steps.

Setup. First, u_1 shares a hash value $h_1 := \mathsf{H}(r_1)$ of a secret r_1 to u_2. Then, u_1 creates a transaction on the coins c_1 such that it can be spendable by u_2 only if the preimage of h_1 is presented. Similarly, u_2 also creates a transaction on the coins c_2 with the same preimage condition for u_1. Here, both transactions have timeouts t_i such that, once t_i elapses, u_i can redeem c_i if the counterparty does not continue to the exchange. Also, the timelock, t_2, on u_2's transaction is shorter (i.e., $t_2 < t_1$) to ensure that u_2 would have enough time to react. First, u_1 publishes her transaction on-chain, then u_2 does the same.

Swap. Once both transactions are on-chain, u_1 can obtain c_2 by revealing r_1, which yields to u_2 obtaining c_1. Note that this protocol requires both scripting languages of the cryptocurrencies to have preimage conditioned scripts. Later on, in [24], the scriptless version of the protocol is presented where the hash condition is embedded into the signature algorithm.

Let us explain how to achieve atomic swaps using LAS, which requires careful analysis because of the aforementioned knowledge gap. In the scenario below, an *extracted* witness, which satisfies an *extended* relation (i.e., $\mathfrak{R}'_{\boldsymbol{A}}$, but not necessarily $\mathfrak{R}_{\boldsymbol{A}}$), will constitute the opening condition to receive coins.

Let $(\mathsf{pk}_i, \mathsf{sk}_i)$ be the public-secret key pair for user u_i for $i = 1, 2$. First, u_1 generates a statement-witness pair $(Y, y) = (\boldsymbol{t}, \boldsymbol{r}) \in \mathfrak{R}_{\boldsymbol{A}}$ as in Section 3, and sends Y to u_2 along with a proof π of knowledge of a witness $\boldsymbol{r}$ such that $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{r}$ and $\|\boldsymbol{r}\|_\infty \leq 1$. Such a proof can be realized using the recent Esgin-Nguyen-Seiler proof system [11]. Then, u_1 also creates a pre-signature $\hat{\sigma}_1 \leftarrow \mathsf{PreSign}((\mathsf{pk}_1, \mathsf{sk}_1), Y, \mathsf{tx}_1)$ for tx_1 spending the coins c_1 to u_2. After verifying the proof π, u_2 similarly creates a pre-signature $\hat{\sigma}_2 \leftarrow \mathsf{PreSign}((\mathsf{pk}_2, \mathsf{sk}_2), Y, \mathsf{tx}_2)$ for tx_2 spending the coins c_2 to u_1. Then, the two pre-signatures are exchanged between the parties. Now u_1 adapts the pre-signature

$\hat{\sigma}_2$ as $\sigma_2 \leftarrow \mathsf{Adapt}((Y, y), \mathsf{pk}_2, \hat{\sigma}_2, \mathsf{tx}_2)$, and aborts if $\sigma_2 = \perp$. Otherwise, he publishes the full signature σ_2 on the second cryptocurrency's blockchain in order to receive the coins c_2. Then, seeing σ_2, u_2 runs $y' = \boldsymbol{s} \leftarrow \mathsf{Ext}(Y, \sigma_2, \hat{\sigma}_2)$ and $\sigma_1 \leftarrow \mathsf{Adapt}((Y, y'), \mathsf{pk}_1, \hat{\sigma}_1, \mathsf{tx}_1)$. If any of them returns $\perp$, u_2 aborts. Otherwise, u_2 publishes σ_1 on the first cryptocurrency's blockchain to receive the coins c_1. This interaction is depicted in Figure 1.

Let us now analyze whether u_1 receives c_2 if and only if u_2 receives c_1. If u_1 does not receive c_2, i.e., u_1 aborts, then u_2 clearly cannot receive c_1 due to the aEUF-CMA security of LAS as u_2 only has the pre-signature $\hat{\sigma}_1$ and the statement Y (without a witness to Y). On the other hand, if u_1 does receive c_2, this means that σ_2 is valid signature published on a blockchain, i.e., accessible by u_2. Therefore, by the witness extractability of LAS, u_2 can extract a witness $\boldsymbol{s}$ to $Y = \boldsymbol{t}$ such that $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{s}$. Recall that u_1 proved knowledge of a witness $\boldsymbol{r}$ to $Y = \boldsymbol{t}$ such that $\|\boldsymbol{r}\|_\infty \leq 1$. By the hardness of M-SIS, it must be the case that $\boldsymbol{s} = \boldsymbol{r}$ as otherwise $\boldsymbol{A}(\boldsymbol{s} - \boldsymbol{r}) = 0$ gives a solution to $\text{M-SIS}_{n,n+l,q,\beta}$ for $\beta = 2\gamma\sqrt{d(n+\ell)}$. As a result, we have that $\|\boldsymbol{s}\|_\infty = \|\boldsymbol{r}\|_\infty \leq 1$. Therefore, $\boldsymbol{s} \in \mathfrak{R}_{\boldsymbol{A}}$ and the pre-signature adaptability works, and hence the signature σ_1 adapted by u_2 passes the verification. Note that without the proof of knowledge π,

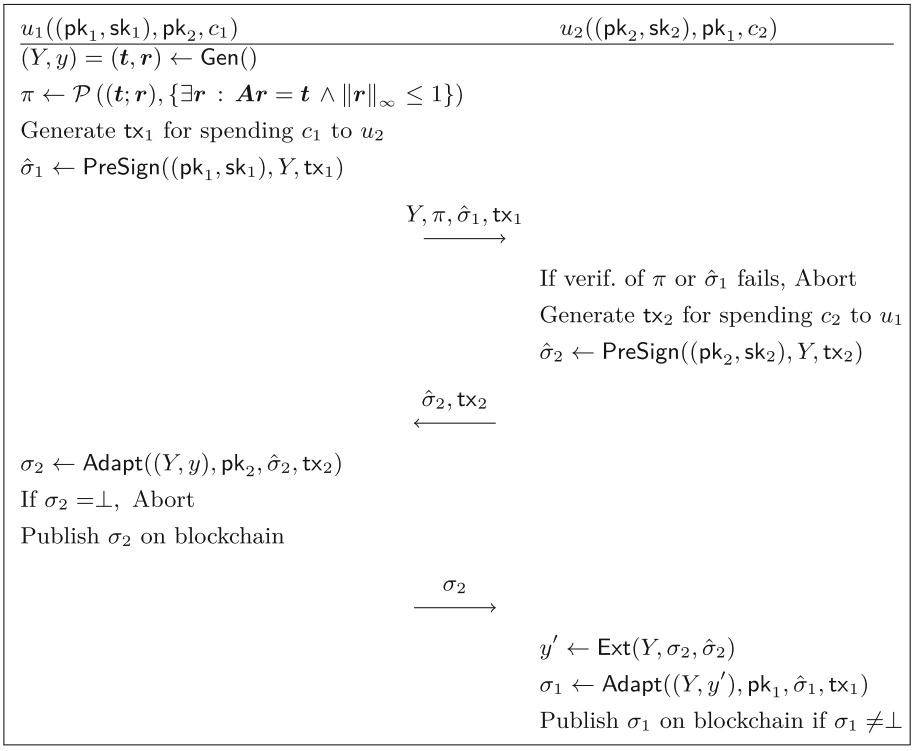

Fig. 1. Atomic swap protocol using LAS.

we cannot guarantee that the extracted witness $\boldsymbol{s}$ will satisfy $\|\boldsymbol{s}\|_\infty \leq 1$, and hence pre-signature adaptability would not have been guaranteed without π. In other words, π is essential to make sure that u_2 receives the coins c_1.

We also note that even though a lattice-based proof of knowledge, π, is relatively costly in terms of communication in practice (but very efficient in computation), this proof is only exchanged between the parties, and not published on blockchain. Therefore, it does not incur additional on-chain storage costs.

4.2 Payment Channel Networks

Payment channel networks (PCNs) [7,16,21,27] are one of the promising solutions to the scalability issues of blockchains. More specifically, many blockchains have poor transaction throughput compared to alternatives like credit card networks because of their consensus mechanisms, where every party (miner) approves and stores every transaction. PCNs improve the throughput by moving some transactions off-chain while relying on the security of the blockchain. In a PCN, two parties can lock coins into a channel where they can make instant and arbitrarily many transactions between each other so long as they have enough balances. One of the most popular PCNs built on Bitcoin is the Lightning network [27]. The overall structure of our post-quantum PCN resembles the Lightning network.

A payment channel consists of three steps: create, update, and close. In the creation phase, parties deposit some coins into the channel and create a funding transaction that spends the input addresses into a single output of the channel. The funding transaction is published on the blockchain and afterward, all of the updates are done off-chain until the closing part. The output condition of funding is spendable only if both parties sign it, which ensures an agreement by both parties. The condition can be implemented by a two-party multi-signature.

In realizing a two-party multi-signature, a straightforward option is to simply combine two individual signatures. Alternatively, there is a lattice-based multi-signature in [4], which can be used in the two-party setting. The underlying signature uses the same "Fiat-Shamir with Aborts" technique, and as stated in [4], the multi-signature can be realized over module lattices as in our work.

When parties want to send/receive coins in the channel, they make off-chain transactions and update the channel balances. In each update, parties create new commit transactions that spend the output of the funding transaction into the two new addresses of the parties with their corresponding balances. Also, parties revoke the previous commits by sharing the signing keys with each other. The revocation can be seen as a punishment mechanism to prevent a malicious party from publishing an old commitment. Once parties are done with the channel, they can close it and obtain their coins by publishing the latest commitment on-chain. A payment channel creation, update, and closing can be done in the same manner as the Lightning network. Now, we investigate how to achieve *multi-hop* payments with our adaptor signature scheme.

A *network* of channels allows parties to make multi-hop payments. More specifically, parties, who do not have a direct channel, can route a payment

using the channels of some intermediary nodes. In these multi-hop payments, it is crucial to synchronize each channel on the route so that either all of them update accordingly or no one does. The Lightning network achieves this by using HTLC (hash-time lock contract). However, in [21], the authors presented privacy concerns as well as the *wormhole attack* for the HTLC mechanism. In this manner, we adopt the AMHL (anonymous multi-hop lock) technique [21] for the multi-hop payments. Also, it is stated that AMHLs are sufficient to construct a payment channel network [21, Theorem 4]. In a scenario where sender S (or I_0) wants to send payment through the intermediary nodes $I_1, \ldots, I_{k-1}$ to the receiver R (or I_k), AMHL-based multi-hop payment works as follows (for simplicity, we omit the fees given to the intermediary nodes).

Setup. S chooses random strings $\ell_0, \ell_1, \ldots, \ell_{k-1}$, and computes $y_j := \sum_{i=0}^{j} \ell_i$ and $Y_j := \mathsf{G}(y_j)$ for $j = 0, \ldots, k-1$ where G is an additively homomorphic one-way function. Then, S shares (Y_{j-1}, Y_j, ℓ_j) with each intermediary I_j for $i = 1, \ldots, k-1$ and (Y_{k-1}, y_{k-1}) with R. Each intermediary party I_j validates the correctness of values by using the homomorphism, i.e., checking that $\mathsf{G}(\ell_j) \oplus Y_{j-1} = \mathsf{G}(y_j) = Y_j$, where $\oplus$ denotes the operation in the range of G.

Payment. S makes a conditional payment to I_1 requiring preimage of Y_0, while each intermediary party I_j, for $j = 1, \ldots, k-1$, makes a payment of the same amount to I_{j+1} with a condition on preimage of Y_j after they receive a similar payment from I_{j-1}. Once all conditional payments are placed, S reveals the preimage y_{k-1} to $R = I_k$ showing that she can redeem the payment. This creates a chain reaction as follows. When an intermediary party I_j receives y_j from I_{j+1}, he can compute $y_{j-1} = y_j - \ell_j$ and redeem the payment by revealing y_{j-1} to I_{j-1}. The procedure is completed once all the channels are updated accordingly.

We can realize AMHL in the post-quantum setting using LAS, but again a special care is required due to the knowledge gap and the use of rejection sampling. First of all, we assume that the length of the PCN is at most $K \ll q$ (i.e., $k \leq K \ll q$) and update the norm check at Steps 6 and 11 in Algorithm 2 by $\|\hat{\boldsymbol{z}}\|_\infty > \gamma - \kappa - K$. Now, S samples $\boldsymbol{r}_j \xleftarrow{\$} \mathbb{S}_1^{n+\ell}$, and computes $\boldsymbol{s}_j = \sum_{i=0}^{j} \boldsymbol{r}_i$ and $\boldsymbol{t}_j = \boldsymbol{A}\boldsymbol{s}_j$ for $j = 0, \ldots, k-1$. Observe that we have $\|\boldsymbol{s}_j\|_\infty \leq k \leq K$ for all $j = 0, \ldots, k-1$. Then, S treats $Y_j = \boldsymbol{t}_j$, $y_j = \boldsymbol{s}_j$ and $\ell_j = \boldsymbol{r}_j$ for $j = 0, \ldots, k-1$. The additively homomorphic function is $f_{\boldsymbol{A}}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}$ (over $\mathcal{R}_q$) mentioned in Section 3. Then, the Setup phase of AMHL described above is run. Additionally, for each $j = 0, \ldots, k-2$, S sends I_{j+1} a NIZK proof π_{j+1} that she knows a witness $y_j = \boldsymbol{s}_j$ to $Y_j = \boldsymbol{t}_j$ such that

$$\|\boldsymbol{s}_j\|_\infty \leq K. \tag{8}$$

After this setup, payment phase begins. Let $(\mathsf{pk}_j, \mathsf{sk}_j)$ be I_j's public-secret key pair used in his channel with I_{j+1}, and tx_j be the transaction transferring the relevant coins from I_j to I_{j+1}. S creates a pre-signature $\hat{\sigma}_0 \leftarrow \mathsf{PreSign}((\mathsf{pk}_0, \mathsf{sk}_0), Y_0, \mathsf{tx}_0)$ and sends it to I_1. Then, for $j = 1, \ldots, k-1$, each user I_j creates a pre-signature $\hat{\sigma}_j \leftarrow \mathsf{PreSign}((\mathsf{pk}_j, \mathsf{sk}_j), Y_j, \mathsf{tx}_j)$ after receiving the pre-signature $\hat{\sigma}_{j-1}$ from I_{j-1}. Once all pre-signatures are generated and transferred, S reveals y_{k-1}

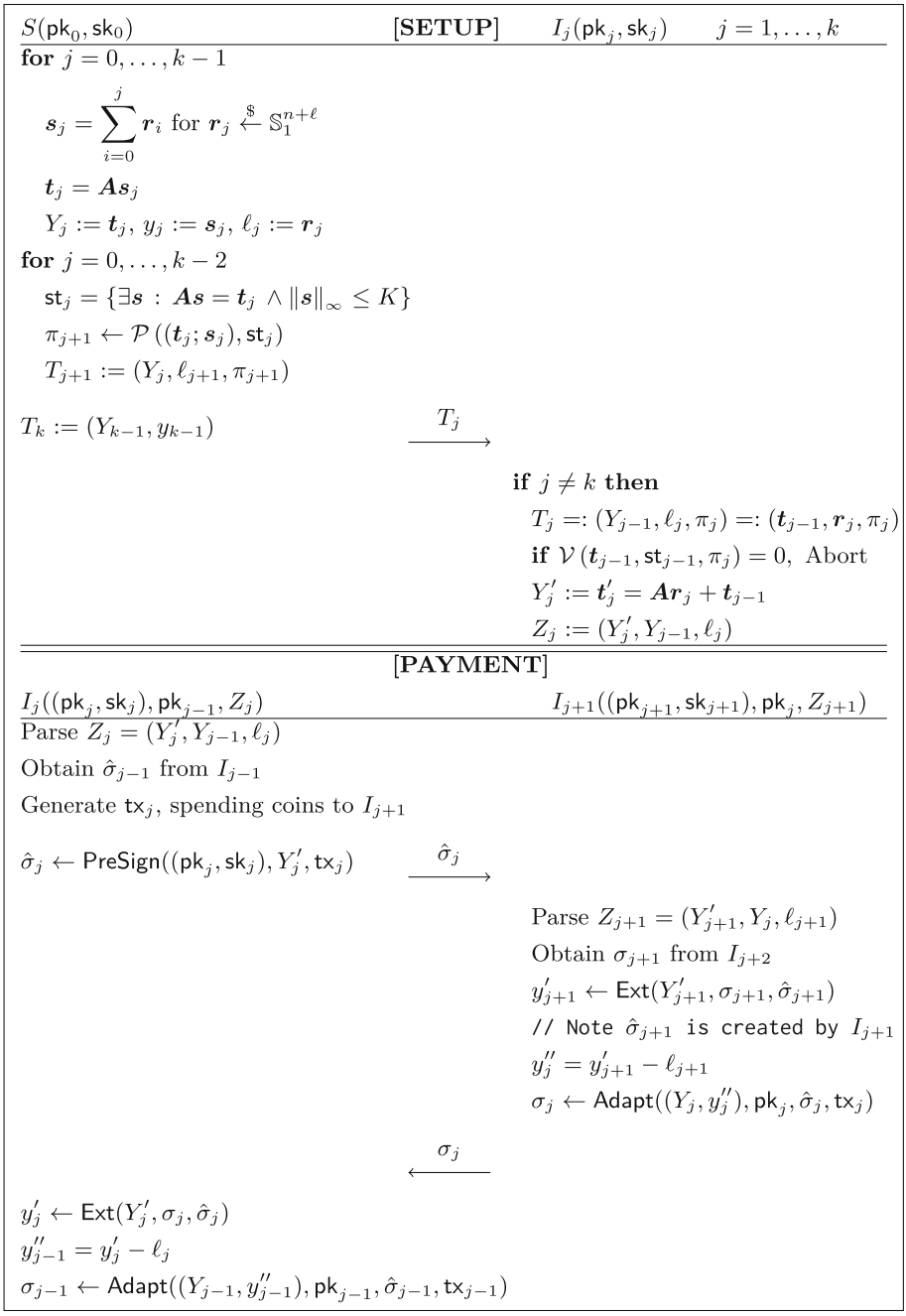

Fig. 2. Anonymous multi-hop payments using LAS. We assume that (i) T_j's are transmitted confidentially, (ii) pre-signature transmission from I_j to I_{j+1} happens only if that from I_{j-1} to I_j already happened, and (iii) signature transmission from I_{j+1} to I_j happens only if that from I_{j+2} to I_{j+1} already happened.

to R, which allows R to adapt the pre-signature $\hat{\sigma}_{k-1}$ to σ_{k-1} in order to receive the relevant coins from I_{k-1}. R sends σ_{k-1} to I_{k-1}. From here, I_{k-1} extracts a witness y'_{k-1} to Y_{k-1}. Then, she computes $y''_{k-2} = y'_{k-1} - \ell_{k-1}$ and uses it to complete the pre-signature $\hat{\sigma}_{k-2}$. Continuing this way, completion of a pre-signature by I_j enables I_{j-1} to obtain a witness to Y_{j-1} and then compute a witness to Y_{j-2} using ℓ_j. The process ends with S receiving σ_0. This anonymous multi-hop payment procedure is depicted in Figure 2.

Let us analyze the details now. First of all, each party I_j has a proof that S knows a witness $y_{j-1} = \boldsymbol{s}_{j-1}$ to Y_{j-1} satisfying (8). Due to the M-SIS hardness as before, no party I_j can obtain another witness to Y_{j-1}, but y_{j-1} generated by S. Therefore, each party I_j is ensured that the witness he extracts will have infinity norm at most K. As a result, each party I_j will be able to adapt the pre-signature $\hat{\sigma}_{j-1}$ successfully and claim his coins thanks to the aforementioned change to Steps 6 and 11 in Algorithm 2.

We emphasize again the importance of the proof π_j's that guarantee pre-signature adaptability. These proofs are only communicated off-chain and thus do not incur any additional on-chain cost, and can be realized using the techniques in [11]. Moreover, the change to Steps 6 and 11 in Algorithm 2 is also important as, in this setting, even honestly-generated witnesses have potentially absolute coefficients greater than 1, but still at most K. Note that this change does not affect the security assumptions as still the original conditions (and even stronger ones) in Algorithms 1 and 2 hold. The only effect is that PreSign may have more restarts, but for most practical settings of, say, $K \leq 50$ (i.e., the length of the PCN is at most 50), the effect will be minimal. In practice, for example, in Lightning Network, the route search algorithm typically stops after $K = 20$ [1].

5 Conclusion

In this work, we constructed the first post-quantum adaptor signature based on standard lattice assumptions. We also showed that our construction, LAS, leads to efficient atomic swaps and payment channel networks in the post-quantum world. In particular, our applications do not incur additional costs on the blockchain, other than the cost of an ordinary lattice-based signature.

References

1. Basis of lightning technology, available at: https://github.com/lightningnetwork/lightning-rfc/blob/master/00-introduction.md
2. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: STOC. pp. 99–108. ACM (1996)
3. Aumayr, L., Ersoy, O., Erwig, A., Faust, S., Hostakova, K., Maffei, M., Moreno-Sanchez, P., Riahi, S.: Generalized bitcoin-compatible channels. Cryptology ePrint Archive, Report 2020/476 (2020), https://eprint.iacr.org/2020/476
4. El Bansarkhani, R., Sturm, J.: An efficient lattice-based multisignature scheme with applications to bitcoins. In: Foresti, S., Persiano, G. (eds.) CANS 2016. LNCS, vol. 10052, pp. 140–155. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48965-0_9

5. Buterin, V.: Understanding serenity, part i: Abstraction (2015), https://blog.ethereum.org/2015/12/24/understanding-serenity-part-i-abstraction/, Accessed on 20 April 2020
6. Damgård, I.: On Σ-protocols. Lecture Notes, University of Aarhus, Department for Computer Science (2002), https://www.cs.au.dk/~ivan/Sigma.pdf
7. Decker, C., Wattenhofer, R.: A fast and scalable payment network with bitcoin duplex micropayment channels. In: Pelc, A., Schwarzmann, A.A. (eds.) SSS 2015. LNCS, vol. 9212, pp. 3–18. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21741-3_1
8. Dryja, T.: Discreet log contracts, https://adiabat.github.io/dlc.pdf
9. Ducas, L., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-Dilithium: Digital signatures from module lattices. In: CHES. vol. 2018–1 (2018), https://eprint.iacr.org/2017/633.pdf
10. Esgin, M.F.: Practice-Oriented Techniques in Lattice-Based Cryptography. Ph.D. thesis, Monash University (5 2020). https://doi.org/10.26180/5eb8f525b3562, https://bridges.monash.edu/articles/Practice-Oriented_Techniques_in_Lattice-Based_Cryptography/12279728
11. Esgin, M.F., Nguyen, N.K., Seiler, G.: Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. Cryptology ePrint Archive, Report 2020/518 (2020), https://eprint.iacr.org/2020/518
12. Esgin, M.F., Steinfeld, R., Liu, J.K., Liu, D.: Lattice-based zero-knowledge proofs: new techniques for shorter and faster constructions and applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 115–146. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26948-7_5
13. Esgin, M.F., Steinfeld, R., Sakzad, A., Liu, J.K., Liu, D.: Short lattice-based one-out-of-many proofs and applications to ring signatures. In: Deng, R.H., Gauthier-Umaña, V., Ochoa, M., Yung, M. (eds.) ACNS 2019. LNCS, vol. 11464, pp. 67–88. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-21568-2_4
14. Esgin, M.F., Zhao, R.K., Steinfeld, R., Liu, J.K., Liu, D.: MatRiCT: Efficient, scalable and post-quantum blockchain confidential transactions protocol. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. pp. 567–584. CCS '19, ACM (2019). https://doi.org/10.1145/3319535.3354200, (Full version at https://eprint.iacr.org/2019/1287)
15. Fournier, L.: One-time verifiably encrypted signatures a.k.a. adaptor signatures (2019), https://github.com/LLFourn/one-time-VES/blob/master/main.pdf
16. Gudgeon, L., Moreno-Sanchez, P., Roos, S., McCorry, P., Gervais, A.: Sok: off the chain transactions. IACR Cryptol. ePrint Arch. **2019**, 360 (2019)
17. Hcash: Hcash features, https://h.cash/#section4, Accessed on 20 April 2020
18. Langlois, A., Stehlé, D.: Worst-case to average-case reductions for module lattices. Design Code Cryptogr. **75**(3), 565–599 (2014). https://doi.org/10.1007/s10623-014-9938-4
19. Lyubashevsky, V.: Fiat-shamir with aborts: applications to lattice and factoring-based signatures. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 598–616. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10366-7_35
20. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_43

21. Malavolta, G., Moreno-Sanchez, P., Schneidewind, C., Kate, A., Maffei, M.: Anonymous multi-hop locks for blockchain scalability and interoperability. In: 26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24–27, 2019 (2019), https://www.ndss-symposium.org/ndss-paper/anonymous-multi-hop-locks-for-blockchain-scalability-and-interoperability/
22. NIST: Post-quantum cryptography - call for proposals (2017), https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization/Call-for-Proposals, Accessed on 20 April 2020
23. Nolan, T.: Alt chains and atomic transfers, https://bitcointalk.org/index.php?topic=193281.msg2224949#msg2224949
24. Poelstra, A.: Adaptor signatures and atomic swaps from scriptless scripts, https://github.com/ElementsProject/scriptless-scripts/blob/master/md/atomic-swap.md
25. Poelstra, A.: Scriptless scripts. Presentation Slides, https://lists.launchpad.net/mimblewimble/msg00086.html
26. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. J. Cryptol. **13**(3), 361–396 (2000)
27. Poon, J., Dryja, T.: The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments (2016), draft version 0.5.9.2, available at https://lightning.network/lightning-network-paper.pdf
28. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM **56**(6), 1–40 (2009)
29. Schnorr, C.P.: Efficient Identification and Signatures for Smart Cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_22
30. Alberto Torres, W., Kuchta, V., Steinfeld, R., Sakzad, A., Liu, J.K., Cheng, J.: Lattice RingCT V2.0 with multiple input and multiple output wallets. In: Jang-Jaccard, J., Guo, F. (eds.) ACISP 2019. LNCS, vol. 11547, pp. 156–175. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-21548-4_9
31. Zcash: Frequently asked questions, https://z.cash/support/faq/#quantum-computers, Accessed on 20 April 2020