Functional Adaptor Signatures: Beyond All-or-Nothing Blockchain-based Payments

Nikhil Vanjani
Carnegie Mellon University
Pittsburgh, USA
nvanjani@cmu.edu

Pratik Soni
University of Utah
Salt Lake City, USA
psoni@cs.utah.edu

Sri AravindaKrishnan Thyagarajan
University of Sydney
Sydney, Australia
t.srikrishnan@gmail.com

ABSTRACT

In scenarios where a seller holds sensitive data x, like employee / patient records or ecological data, and a buyer seeks to obtain an evaluation of specific function f on this data, solutions in trustless digital environments like blockchain-based Web3 systems typically fall into two categories: (1) Smart contract-powered solutions and (2) cryptographic solutions leveraging tools such as adaptor signatures. The former approach offers atomic transactions where the buyer learns the function evaluation $f(x)$ (and not x entirely) upon payment. However, this approach is often inefficient, costly, lacks privacy for the seller's data, and is incompatible with systems that do not support smart contracts with required functionalities. In contrast, the adaptor signature-based approach addresses all of the above issues but comes with an "all-or-nothing" guarantee, where the buyer fully extracts x and does not support functional extraction of the sensitive data. In this work, we aim to bridge the gap between these approaches, developing a solution that enables fair functional sales of information while offering improved efficiency, privacy, and compatibility similar to adaptor signatures.

Towards this, we propose *functional adaptor signatures (FAS)* a novel cryptographic primitive that achieves all the desired properties as listed above. Using FAS, the seller can publish an advertisement committing to x. The buyer can pre-sign the payment transaction w.r.t. a function f, and send it along with the transaction to the seller. The seller adapts the pre-signature into a valid (buyer's) signature and posts the payment and the adapted signature on the blockchain to get paid. Finally, using the pre-signature and the posted signature, the buyer efficiently extracts $f(x)$, and completes the sale. We formalize the security properties of FAS, among which is a new notion called *witness privacy* to capture seller's privacy, which ensures the buyer does not learn anything beyond $f(x)$. We present multiple variants of witness privacy, namely, *witness hiding*, *witness indistinguishability*, and *zero-knowledge*, to capture varying levels of leakage about x beyond $f(x)$ to a malicious buyer.

We introduce two efficient constructions of FAS supporting linear functions (like statistics/aggregates, kernels in machine learning, etc.), that satisfy the strongest notion of witness privacy. One construction is based on prime-order groups and compatible with Schnorr signatures for payments, and the other is based on lattices and compatible with a variant of Lyubashevsky's signature scheme. A central conceptual contribution of our work lies in revealing a surprising connection between functional encryption, a well-explored concept over the past decade, and adaptor signatures, a relatively new primitive in the cryptographic landscape. On a technical level, we avoid heavy cryptographic machinery and achieve improved efficiency, by making black-box use of building blocks like *inner product functional encryption (IPFE)* while relying on certain security-enhancing techniques for the IPFE in a non-black-box manner. We implement our FAS construction for Schnorr signatures and show that for reasonably sized seller witnesses, the different operations are quite efficient even for commodity hardware.

CCS CONCEPTS

• **Security and privacy** → **Cryptography**.

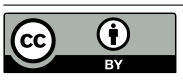
KEYWORDS

Adaptor Signatures; Functional Encryption

ACM Reference Format:
Nikhil Vanjani, Pratik Soni, and Sri AravindaKrishnan Thyagarajan. 2024. Functional Adaptor Signatures: Beyond All-or-Nothing Blockchain-based Payments. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS '24), October 14–18, 2024, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3658644.3690240

1 INTRODUCTION

The shift from centralized Web2 to decentralized blockchain-based Web3 solutions has transformed digital goods trading. Smart contracts have been pivotal in this transition, and they facilitate seamless transactions between sellers and buyers. Imagine a seller offering a solution to a computational puzzle or a problem for sale. Typical smart contracts let the seller specify the terms of their digital offering, referred to as an *advertisement*, creating a transparent and accessible mechanism for buyers. The buyers lock coins to the contract as an expression of interest to buy the solution. Now the seller publishes the solution in the form of a transaction on the blockchain that invokes the contract. The contract executes a validation procedure to verify the solution's correctness, and if successful, transfers the locked coins to the seller. If the seller fails to respond with a correct solution, the contract refunds the buyer after the expiry of a timeout. This template underpins various digital trading scenarios including Hash Timelock Contracts [35], Bridges [4], NFT sales [39], smart contract-based offline services [7], and e-commerce [31].



CCS '24, October 14–18, 2024, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-0636-3/24/10
https://doi.org/10.1145/3658644.3690240

Despite the advantages of smart contracts, their current implementation faces significant challenges:

- *Cost and Efficiency*: Sellers often incur hefty fees for posting advertisements on smart contracts and verifying secret solutions, resulting in higher transaction costs. Ethereum measures these costs in gas, and many popular smart contract applications have significantly higher gas costs compared to regular user-to-user payment transactions [5].
- *Privacy*: Openly disclosing the secret solution compromises privacy. Encrypting values on the contract to address this necessitates more expensive and intricate cryptographic tools, further exacerbating costs. Further, the approach affects the fungibility of the system as pointed out in many previous works [21, 29, 37, 38].
- *Compatibility*: Finally, the method relies on *complex* smart contracts, rendering itself incompatible with prominent blockchains like Bitcoin. This poor adaptability hampers scalability in the broader Web3 ecosystem.

Adaptor Signatures. As the community delves deeper into enhancing the efficiency and simplicity of smart contracts, cryptographic tools like adaptor signatures [12, 21, 22, 29, 33, 36, 37] have emerged as promising solutions. Adaptor signatures help model a blockchain-based fair exchange between a buyer for its signature σ (on a transaction) and an NP witness x that is known to the seller. More formally, the seller first samples an NP statement X along with its witness x. The statement X binds the seller to its witness which it publishes to advertise its intent to sell x.[1] Now, adaptor signatures offers four algorithms PreSign, PreVerify, Adapt, Ext, that work as follows. The buyer using the pre-sign algorithm PreSign first generates a pre-signature $\widetilde{\sigma}$ on the payment transaction m w.r.t. the signing key sk and seller's statement X. The seller verifies the validity of $\widetilde{\sigma}$ using PreVerify algorithm. Then, using the Adapt algorithm, the seller adapts $\widetilde{\sigma}$ into a full signature σ on m using its witness x. To redeem the transaction m, the seller posts σ on the blockchain. Central to the efficacy of adaptor signatures is their unique property that allows the buyer, with access to the pre-signature $\widetilde{\sigma}$ and the posted signature σ, to efficiently *extract* the secret solution x using the algorithm Ext, thus completing the fair exchange. This innovative approach addresses the drawbacks of traditional smart contracts:

- *Improved efficiency and privacy:* Adaptor signatures require only a single transaction and signature to be posted on the blockchain, reducing transaction costs and enhancing efficiency. Moreover, with buyers locally extracting secrets, transactions resemble regular payments, thereby improving the privacy and fungibility of coins in the system.
- *Enhanced compatibility:* Signature verification, a universally supported operation, ensures compatibility across all blockchain systems, making adaptor signatures-based digital sales compatible with the broader Web3 landscape.

Looking ahead: Granular and functional sale of digital information. However, while adaptor signatures offer a promising solution for digital transactions, they come with certain limitations. They operate on an all-or-nothing basis, revealing the entire secret upon successful transactions or learning nothing in case the seller aborts. This lack of granularity contrasts with the concept of *functional encryption (FE)* [16], where recipients can decrypt and learn specific functions applied to the encrypted message instead of the complete message. Exploring granularity not only enables more nuanced and functional sales of digital information but also ensures the privacy of the seller's data. For instance, the seller holding medical records might desire to safeguard specific functions of the records, adhering to medical data privacy regulations like the Health Insurance Portability and Accountability Act (HIPAA) in the United States. The privacy requirement here is that the buyer learns only the specific function for which the payment is made, ensuring compliance with legal frameworks. Below we expand on this intuition and give two illustrative examples of real-world scenarios where functional sales of digital information can be critical.

Application 1: Medical information. In this setting the seller holds a medical database and is authorized to sell functions of the database by the patients or appropriate authorities. For instance, the buyer who might be an insurance firm, can query some aggregate/statistics of the demography in the database. If it's a research organization, more complex functions like correlations of pre-existing medical conditions and cancer can be queried. The seller has the financial incentive to provide the information, which, in the long run, can contribute greatly to medical advancements.

Application 2: ML model training information. In this case, the seller owns a dataset, and the buyer, with an ML model, seeks to train the model on the seller's dataset. The buyer presents the model to the seller, who then trains the model over the dataset and returns the result for a price. Training may involve simple similarity measures like computing the kernel of two vectors or more complex functions with large datasets (e.g., regression analysis, clustering).

Therefore, we ask the following question,

Can we facilitate the functional sale of digital information using adaptor signatures?

More concretely, let the seller hold secret data x, and the buyer wants access to a function f. *Can we expand the functionality of adaptor signatures such that, at the end of the exchange, the seller obtains a payment transaction and a signature from the buyer, and the buyer learns $f(x)$, and not the entirety of x?*

While general functions are the ultimate goal, this work focuses on the restricted function class of linear functions (where f is a linear function), arguing that they are already widely applicable. We show that practical constructions using lightweight cryptographic tools can be achieved for linear functions, with some insights into the challenges of achieving a more expressive class of functions.

1.1 Our Contributions

Below we summarize the contributions of this work.

New cryptographic primitive - Functional Adaptor Signatures. To answer the above question affirmatively, we formally introduce a novel cryptographic primitive called *functional adaptor signatures (FAS)* (in Section 4). It is similar in functionality to standard adaptor signatures, except that it is additionally parameterized by a family of functions $\mathcal{F}$. This addition introduces new interfaces

[1] Typically, this step is not explicitly stated in the formalization of adaptor signatures. We adopt this extension to aid our functional adaptor signature formalization.

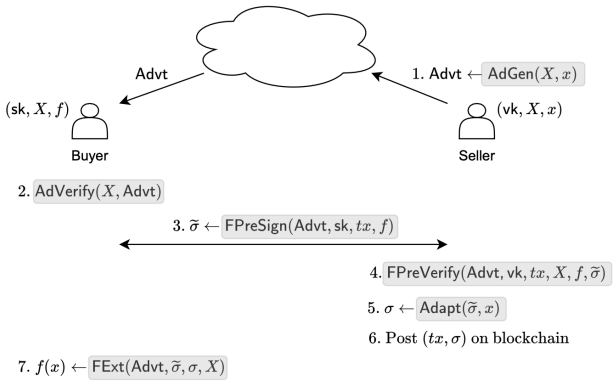


Figure 1: Functional payments via simplified FAS interface.

to FAS; we describe these below in the context of a digital trade for ease of understanding. A pictorial description is given in Figure 1.

The seller of the trade generates an advertisement advt using the FAS interface AdGen, that embeds the statement X and the corresponding secret witness x, where $(X, x) \in R$ for some NP relation R. Anyone can publicly verify the well-formedness of advt using the AdVerify algorithm. The buyer executes the functional pre-sign algorithm FPreSign to generate a pre-signature $\widetilde{\sigma}$ on the message m (the payment transaction, denoted as tx in Figure 1) for the statement X and a function f. FPreVerify algorithm helps the seller to verify if $\widetilde{\sigma}$ is valid with respect to the message m, statement X and function f. If the pre-signature is valid, the seller using the Adapt algorithm and the witness x can adapt $\widetilde{\sigma}$ into a signature σ on the message m. The seller may now post the transaction and the signature on the blockchain to get paid. Finally, the buyer can efficiently extract the value $y = f(x)$ using the functional extract algorithm FExt given access to advt, $\widetilde{\sigma}$, and σ. With the above outline of FAS, we can see that we gain the same advantages of adaptor signatures over smart contracts for digital trading, while also achieving the desired granularity.

Definitional framework. Recent works [19, 25] have highlighted the inherent complexities in designing the security framework for even standard adaptor signatures, with many subtle nuances that require a lot of care. In this work, we overcome these challenges and present a comprehensive definitional framework for the security of FAS (in Section 4). We establish the formal foundations of FAS, constituting a substantial contribution of this work.

Broadly speaking, for a secure FAS, we are required to upgrade the security definitions of standard adaptor signatures to handle additional constraints posed by the functional aspects of the primitive. Moreover, we introduce additional novel security properties for a secure FAS to satisfy, among which *witness privacy* is a prominent one. This property captures the leakage a malicious buyer learns about x beyond $f(x)$. To formalize this, we present three different notions of witness privacy for FAS, akin to the different privacy guarantees of cryptographic proofs. To be more specific, we present (1) *witness hiding*, where the malicious buyer after learning $f(x)$ cannot output the secret witness x, (2) *witness indistinguishable*, where the malicious buyer learns $f(x)$ and cannot distinguish between two witnesses $x = x_0$ and $x = x_1$ (for the statement X) as long as $f(x_0) = f(x_1)$, and (3) *zero-knowledge*, where the malicious buyer learns no other information about x, other than $f(x)$.

Efficient constructions. From a practical perspective, we give several efficient constructions of FAS for the class of linear functions (like aggregates, statistics, kernel computation in ML, etc.) that are compatible with standard signatures like Schnorr and ECDSA. Notably, these signature schemes are widely used in blockchain systems for transaction verification, thus making our constructions ready for deployment in current systems. Towards a post-quantum instantiation of FAS, when the buyer has linearly independent function queries, we also present a FAS construction compatible with a variant of the lattice-based signature scheme of Lyubashevsky [32]. Later in Section 2.4, we discuss how the linear-independency requirement can be relaxed with minor trade-offs in efficiency.

Most importantly, the key conceptual contribution of our work is the surprising connection between two seemingly unrelated tools, namely, functional encryption [16] that has been extensively studied for over a decade and a relatively new primitive, adaptor signatures. Concretely, we construct FAS for a family of inner-product functions $\mathcal{F}_{\mathsf{IP}}$ using three lightweight building blocks:

(i) a functional encryption scheme IPFE for $\mathcal{F}_{\mathsf{IP}}$,
(ii) an adaptor signature scheme AS w.r.t. the digital signature scheme DS and some hard relation R_{IPFE} (related to IPFE),
(iii) NIZK argument system for NP.

A nice feature of our construction is that we show it satisfies the zero-knowledge style witness privacy assuming only selective, IND-security of IPFE and zero-knowledge of NIZK.

We show two ways to instantiate the construction:

- **Prime-order groups (Section 6).** Using a varaint of the DDH-based IPFE scheme by Abdalla et al. [9], we get R_{IPFE} is the discrete log relation. So, we use Schnorr adaptor signature [12] for AS.
- **Lattices (Section 7).** Using a variant of the LWE-based IPFE scheme by Agrawal et al. [11], we get R_{IPFE} is the short integer solution (SIS) relation. So, we use a variant of the post-quantum adaptor signature [22] for AS.

Throughout the paper, for conceptual clarity, we abstract out the relation R to be any NP relation, and make black-box use of a NIZK [34] for relation R. However, we emphasize that in both cases, the NIZK can be efficiently instantiated depending on the exact application (i.e., for a concrete relation R) and plugged in without affecting other components. Lastly, for a weaker witness privacy notion of witness-indistinguishability, we present a round-optimal construction (details in the full version of the paper).

Performance evaluation. To assess the practicality of our FAS, we provide an open-source implementation [6] of our prime-order group-based instantiation satisfying zero-knowledge style witness privacy in Python. The details of our performance evaluation are given in Section 8. We also perform a series of benchmarks on our FAS scheme for a wide range of parameter settings. Our results show that our scheme is practical for a wide variety of real-world scenarios. For instance, on a MacBook Pro, for 300KB size witness, the time taken for pre-signing is 0.344 seconds, pre-verification is 0.424 seconds, adapt is 0.035 seconds, functional-extract is 1.025 seconds. We share more details of our findings in Section 8.

Applicability of Linear Functions. Linear functions although restricted, allow us to capture many scenarios including learning statistical information (like mean, average, weighted linear functions, select entries) of records in a medical database, employee records of an organization, weather data, or ecological records of endangered species. Linear functions also allow a buyer to measure the proximity of a vector he holds with the secret vector of the seller. Such proximity measures are quite fundamental in Machine Learning where they are referred to as computing the kernel of two vectors or graphs in the case of [13].

2 TECHNICAL OVERVIEW

In this overview, we present our approach to fair payments for learning linear functions of a seller's witness.

To describe our techniques, we use the following setup: let p and ℓ be integers where p is prime and L be an NP language containing statements X with witness $\mathbf{x} = (x_1, \ldots, x_\ell) \in \mathbb{Z}_p^\ell$. Our goal is to support linear functions over $\mathbf{x}$. Throughout this paper, we represent such linear functions by vectors $\mathbf{y} = (y_1, \ldots, y_\ell) \in \mathbb{Z}_p^\ell$ and the corresponding evaluation on $\mathbf{x}$ by the inner-product of the vectors $\mathbf{x}$ and $\mathbf{y}$ modulo p. That is, $f_{\mathbf{y}}(\mathbf{x}) = \langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{\ell} x_i \cdot y_i \mod p$. [2]

2.1 Functional Adaptor Signatures for Fair Functional Payments

We have a fair exchange scenario where the seller has a witness $\mathbf{x}$ which is a vector in $\mathbb{Z}_p^\ell$, and only desires to sell some linear functions $\mathbf{y}$ on input $\mathbf{x}$. To enable such an exchange FAS, like adaptor signatures, involves algorithms like FPreSign and FPreVerify, Adapt, and FExt. However, the key distinction lies in the fact that FPreSign and FPreVerify are defined w.r.t. a function $\mathbf{y}$ from the buyer, while the functional extraction algorithm FExt, returns $f_{\mathbf{y}}(\mathbf{x})$ instead of $\mathbf{x}$ itself, ensuring the buyer learns only the function's result.

2.1.1 **Security of FAS.** We consider two cases, where the adversary can corrupt either the seller or the buyer. Similar to standard adaptor signatures, when the seller is corrupt, FAS must mainly guarantee (1) *unforgeability*, where an adversary who doesn't know the witness cannot forge honest buyers' signatures, and (2) *witness extractability*, where the adversary cannot publish a buyer's signature such that when the buyer tries to extract using FExt, it gets $z' \neq f_{\mathbf{y}}(\mathbf{x})$. [3] Crucially, in the case of FAS, the adversary is allowed to get many functional pre-signatures for any choice of function $\mathbf{y}$.

On the other hand, when the buyer is corrupt, similar to standard adaptor signatures, FAS must guarantee *pre-signature adaptability*, where if the adversary outputs a valid functional pre-signature, then adapting it using a valid witness $\mathbf{x}$ should result in a valid signature under the buyer's key. Most importantly, FAS must guarantee protection of the seller's witness $\mathbf{x}$ which we capture in the notion called *witness privacy*.

In more detail, FAS satisfies *zero-knowledge* witness privacy if a malicious buyer learns no more information about $\mathbf{x}$ than $f_{\mathbf{y}}(\mathbf{x})$ after the interaction. To formally define this in Section 4, we carefully borrow formalism from the related privacy notion of zero-knowledge for cryptographic proof systems [26]. Informally, for every malicious buyer (that can sample its signing key), we require the existence of an efficient p.p.t. simulator Sim that can simulate the buyer's view only using $f_{\mathbf{y}}(\mathbf{x})$. Philosophically, the existence of such an efficient simulator ensures that whatever the buyer learned from interacting with the seller, it could have on its own from $f_{\mathbf{y}}(\mathbf{x})$ by running in polynomial time. While our notion shares similarities and is inspired by cryptographic proof systems, we need additional care to correctly model the fact that the buyer is allowed to ask for multiple functions which, moreover, can be adaptively chosen. To gain a comprehensive understanding of witness privacy, we also introduce two natural relaxed variants referred to as *witness hiding* and *witness indistinguishability* in the full version, and study their relation with zero-knowledge witness privacy . Interestingly, in Section 2.5, we show that the relaxation to witness indistinguishability enables a round optimal construction.

For the rest of this overview, we will focus on building an efficient FAS centered around the witness privacy guarantee given it is the unique property of FAS compared to standard adaptor signatures. Moreover, our focus will be on the strongest notion of zero-knowledge witness privacy.

2.1.2 **Strawman construction of FAS.** To build FAS, one can combine Schnorr adaptor signature (discussed above) along with a general-purpose non-interactive zero-knowledge (NIZK) proof. We present this strawman solution below and highlight a key efficiency challenge that serves as the starting point of our construction.

Recall the protocol flow in Figure 1. We describe how advt, $\widetilde{\sigma}, \sigma$ are computed and the functional extraction process. The seller publishes advt = (ct, π), where ct is a semantically-secure public-key encryption of the witness $\mathbf{x}$ and NIZK proof π that certifies ct encrypts a witness for the statement X. The pre-signature $\widetilde{\sigma}$ is computed interactively among the buyer and the seller as follows. (i) The buyer sends f to the seller. (ii) The seller computes a secret-key encryption ct_z of the requested evaluation $z = f(\mathbf{x})$ using a fresh secret-key k. Then, the seller encodes k in $\mathbb{Z}_p$ and *engages with the buyer in an adaptor execution to sell the witness for the discrete logarithm of group element K, where $K = g^k$ for some generator g of the group* $\mathbb{G}$. That is, the seller sends $(\mathsf{ct}_z, K, \pi_z)$ to the buyer, where NIZK proof π_z certifies ct_z encrypts $f(\mathbf{x})$ using the key k where $\mathbf{x}$ is encrypted in ct. The buyer treats K as the Schnorr adaptor statement and computes a pre-signature $\widetilde{\sigma}$ on the transaction message m if and only if the NIZK proof π_z verifies. On obtaining $\widetilde{\sigma}$, the seller adapts it into a valid signature σ on m using the key k as the witness. For functional extraction, the buyer runs Ext algorithm of the Schnorr adaptor signature to extract k from $(\widetilde{\sigma}, \sigma)$ and then decrypts ct_z using it to recover the function evaluation z.

The desired privacy of FAS is attained by relying on the zero-knowledge property of both NIZK proofs and the semantic security of ct. That is, the seller side of interaction can be efficiently simulated just from the evaluation $z = f(\mathbf{x})$. The security against a malicious seller relies on the security of the underlying adaptor signature scheme and the soundness of the NIZK protocol.

While this approach enables fair functional payments, it requires that the seller compute NIZK proofs for every FPreSign interaction

[2] While our lattice-based instantiation is for computing inner products $\mod p$, the group-based instantiation is for computing integer inner products with bounded outputs, i.e., $f_{\mathbf{y}}(\mathbf{x}) \in \{0, \ldots, B\}$ for some apriori fixed bound $B \ll p$.

[3] We also require *advertisement soundness* and *pre-signature validity* properties against a corrupt seller, and defer their discussion to the full version.

with a buyer. The seller's computation in computing these proofs (and their size) grows *polynomially* in the witness size $\mathbf{x}$ and is proportional to the complexity of the function f. While the proof sizes can be reduced by relying on general-purpose ZK-SNARKs [15], the proving cost growing super-linearly in $|\mathbf{x}|$ will become prohibitively expensive even for moderately sized databases $\mathbf{x}$. This cost also affects the scalability of the solution at the application layer. For instance, if the seller is dealing with several thousand buyers each with a different function, the seller will be computationally strained, leading to delays and a potential Denial of Service (DoS) attack vector via malicious buyers. A more efficient solution would allow (a) the seller's computation to grow linearly in the size of $\mathbf{x}$ and (b) the seller's communication to be proportional to the size of the evaluation z which would be significantly smaller than $\mathbf{x}$ itself. As we explain below, we achieve these efficiency targets without relying on NIZKs, which ensures we use underlying cryptographic primitives in a *black-box* manner.

2.2 Our Techniques: Functional Encryption + Adaptor Signatures

The blueprint of our FAS construction is to combine adaptor signatures with another cryptographic tool called functional encryption (FE). FE was introduced as an enrichment of any (public-key) encryption scheme that allows for fine-grained decryption. In particular, in addition to algorithms (Setup, Enc, Dec) which are typical to any encryption scheme, a FE scheme for a function class $\mathcal{F}$ provides an additional functional key-generation algorithm KGen. The algorithm takes as input the master secret-key msk and function $f \in \mathcal{F}$, and outputs a functional secret-key sk_f such that decrypting any encryption of $\mathbf{x}$ (w.r.t. msk) with sk_f reveals $f(\mathbf{x})$ and no more. As discussed earlier, this notion of strong privacy where no information about $\mathbf{x}$ is leaked other than $f(\mathbf{x})$ is formalized using the *simulation* paradigm. For simplicity, we will henceforth refer to FE schemes satisfying this privacy property as *simulation-secure*. Numerous works have built FE schemes for rich classes of functions from a variety of hardness assumptions. More specifically, for the case of linear functions, we know of group-based schemes for prime-order groups [9–11, 40] as well as for unknown-order groups [10, 11]; and post-quantum constructions [9–11] are known from the hardness of the LWE problem.

Our construction template in more detail. In the quest of removing NIZKs from FPreSign, we modify the above steps of the fair exchange as follows: (a) The AdGen remains identical except that ct is computed using the FE scheme's encryption algorithm, (b) In the FPreSign interaction, the seller instead computes the functional secret-key sk_f and engages with the buyer in an adaptor execution to sell sk_f, (c) the Adapt algorithm remains the same, and (d) the FExt algorithm extracts the functional secret-key sk_f and decrypts ct directly to return $f(\mathbf{x})$. If the FE scheme satisfies simulation-security and the NIZK (in the advertisement) is zero-knowledge, then seller's interaction can be simulated only via $f(\mathbf{x})$, which ensures the desired seller privacy.

An astute reader might ask why to use an encryption of $\mathbf{x}$ in the advertisement and not a one-way function of $\mathbf{x}$, similar to how statement X and witness x have a discrete log relation in standard adaptor signature constructions. A crucial advantage of our construction template is that since $\mathbf{x}$ is encrypted under semantically secure encryption, we can let witness $\mathbf{x}$ be of low-entropy like user databases, which are the main applications we are targeting for FAS. A similar setting would be insecure when using one-way functions instead of encryption since $\mathbf{x}$ is not a high-entropy witness. Moreover, achieving zero-knowledge witness privacy seems hopeless when using one-way functions for setting up X.

Challenges. Unfortunately the above approach doesn't quite work as-is and runs into two main challenges:

(1) *Correctness of the Adaptor Statement*: Suppose that one could encode sk_f as an integer x_f in $\mathbb{Z}_p$ (for an appropriate p), and then use the Schnorr adaptor signature for the statement $X_f = g^{x_f}$. Still, for fairness, the buyer needs to gain confidence in the validity of X_f (e.g., learning the incorrect functional key $\tilde{x}$ will disallow the buyer from learning the correct evaluation) during FPreSign. Augmenting the seller's message with a NIZK proof that attests to the correctness of X_f (i.e., discrete-logarithm of X_f is indeed the correct sk_f) would be sufficient. But the use of NIZKs was exactly what we were trying to avoid, and hence it seems we are back to square one.
(2) *Compatibility with Adaptor Signatures*: Recall that known adaptor signature schemes only facilitate the sale of witnesses for special algebraic languages. For the above approach to work, we need a *simulation-secure* FE scheme where the functional key-generation algorithm exhibits the required structural compatibility. For e.g., to rely on Schnorr adaptor signatures that allow selling discrete-logarithms $x \in \mathbb{Z}_p$ of prime-order group elements $X = g^x$, we need a FE scheme where the functional secret-key sk_f w.r.t. any linear function f is such that $\mathsf{sk}_f \in \mathbb{Z}_p$. In fact, [9] presents such an IPFE scheme, but the scheme satisfies only a weaker notion of IND-security (and not simulation security) which is insufficient for the template to work.

Our Technique 1: Augmenting IPFE with PubKGen **algorithm.** To avoid using NIZKs during FPreSign, we observe that the IPFE schemes for computing inner product of vectors of length ℓ are obtained by starting with ℓ instances of some PKE scheme and a functional secret key is essentially a linear combination of the secret keys of the ℓ PKE instances (as pointed out in [9]). Combining this observation with the fact that the public key and secret key of the underlying PKE satisfy some algebraic relation, one can envision that a similar linear combination of the public keys of the PKE schemes might give us a commitment to the functional secret key that can be directly used as the statement X_f. Crucially, if X_f can be deterministically computed using IPFE's master public key and the function description f, then such an algorithm would be public and the buyer can compute X_f without any interaction with the seller, thus avoiding NIZKs during FPreSign altogether. We formalize this vision by augmenting IPFE with a public deterministic algorithm PubKGen. The requirement that the output of PubKGen, i.e., X_f must be a commitment to the output of KGen, i.e., sk_f, is formalized via a *compliance property* (Definition 3.2). Informally, R_{IPFE}-compliance says that for a given relation R_{IPFE}, it must be the case that $(X_f, \mathsf{sk}_f) \in R_{\mathsf{IPFE}}$.

For instance, if we set the relation R_{IPFE} to be the discrete-log relation, then the key structure becomes compatible with the

hard relation of Schnorr adaptor signatures. As an example, the IND-secure IPFE scheme of [9] can be shown to have such a key structure. In the IND-secure IPFE scheme of [9], the master secret key is $\mathbf{s} = (s_1, \ldots, s_\ell) \in \mathbb{Z}_p^\ell$, the master public key is $g^{\mathbf{s}} = (g^{s_1}, \ldots, g^{s_\ell})$, the functional secret key for function $\mathbf{y} = (y_1, \ldots, y_\ell)$ is $\mathsf{sk}_{\mathbf{y}} = \sum_{i\in[\ell]} s_i y_i \bmod p$. Then, PubKGen can be defined to output $\mathsf{pk}_{\mathbf{y}} = \prod_{i\in[\ell]} (g^{s_i})^{y_i}$. Consequently, we can observe that $\mathsf{pk}_{\mathbf{y}} = g^{\mathsf{sk}_{\mathbf{y}}}$, thus $(\mathsf{pk}_{\mathbf{y}}, \mathsf{sk}_{\mathbf{y}})$ satisfy the discrete log relation. Defining PubKGen in this way makes it compliant with the Schnorr adaptor signatures: if the buyer wants to learn function $\mathbf{y}$ evaluated on the seller's witness, he can locally compute $\mathsf{pk}_{\mathbf{y}}$ and use it as the adaptor statement. Upon receiving a pre-signature from the buyer, the seller can locally compute $\mathsf{sk}_{\mathbf{y}}$ and use it to adapt the pre-signature. Eventually, the buyer can extract $\mathsf{sk}_{\mathbf{y}}$, thus enabling functional decryption. Crucially notice that because of deterministic nature of PubKGen, the pre-signing becomes non-interactive. But as noted earlier, IND-secure IPFE isn't sufficient for the FAS template to work, so we are not done yet.

Our Technique 2: Simulation-Secure Compatible IPFE. To resolve the compatibility with known adaptor signatures, we rely on the IND-secure IPFE to simulation-secure IPFE compiler of [10]. More specifically, their compiler lifts any IND-secure IPFE to achieve simulation-security without changing the structure of the functional secret keys. This is achieved by increasing the length of function vectors from ℓ to 2ℓ, where the first ℓ slots encode the function as before, and the extra ℓ slots encode some random coins. These random coins are used by the IPFE simulator to argue simulation security. To preserve correctness, the length of the message vector to be encrypted is also increased from ℓ to 2ℓ, where the first ℓ slots encode the message as before, and the extra ℓ slots are set to zero. A small caveat is that the compiler results in a stateful functional secret key generation algorithm. We make it stateless which allows us to make an optimization where we increase the vector lengths from ℓ to only $\ell + 1$. Instantiating their compiler with the DDH-based IND-secure IPFE scheme of [9] results in a simulation-secure FE that is compatible with Schnorr adaptor signatures. We also extend this to the post-quantum setting by using a simplified variant of the LWE-based simulation-secure IPFE scheme of [10] [4] that is compatible with the Dilithium adaptor signature [23] instantiated on unstructured lattices.

While the above technique addresses the compatibility w.r.t. known adaptor signatures, proving *zero-knowledge* witness privacy requires more care and we will introduce another technique for it. Let us first explain the challenge with proving *zero-knowledge*. Even when starting with a base IPFE scheme (that is IND-secure) with a deterministic functional key-generation algorithm, the IPFE compiler introduces randomness in the functional key-generation algorithm. [5] This randomization is crucial for the IPFE simulator's correctness and essential to work with the base FE scheme that is only IND-secure. In the context of FAS, during FPreSign, if the seller could share the randomness used as part of the generating sk_f, then the buyer can mimic the same computation using the seller's randomness and determine the validity of the sk_f embedded in X_f. While this approach makes pre-signing interactive yet it would certainly avoid the use of general-purpose NIZKs as seen in the template above because the check is *canonical* now. Unfortunately, exposing the seller's randomness would compromise simulation security of the compiled IPFE scheme as the IPFE simulator would then get stuck in the analysis. Consequently, we can't show *zero-knowledge* of FAS. Note that an apporach to determine validity of X_f without having the seller share the random coins could be to use NIZKs, but as said before, we want to avoid it.

Our Technique 3: Interactive pre-signing without NIZKs. To address the *zero-knowledge* of FAS, we open up the IPFE compiler of [10], that is, make non-black-box use of the IPFE compiler and reveal only a part of the randomness of the functional key-generation algorithm. By carefully choosing the randomness revealed by the seller, we can guarantee the validity of the adaptor statement X_f to the buyer (without expensive NIZKs) and also allow flexibility to the IPFE simulator to successfully finish the simulation (See Remarks 2.1 and 2.2 for more details).

We emphasize that the non-black-box use of the techniques from [10] is a significant technical part of this work. We elaborate on these aspects of our construction in more detail in Section 2.3.

2.3 Our Construction

Suppose that the relation R is such that for any statement X and witness $\mathbf{x}$ satisfying $(X, \mathbf{x}) \in R$, it is the case that $\mathbf{x} \in \mathbb{Z}_p^\ell$. Further, suppose that inner-product functions are of the form $\mathbf{y} \in \mathbb{Z}_p^\ell$. Then, our FAS construction is obtained by employing the abovementioned techniques. We remind the reader of the protocol flow in Figure 1.

- Setup: Run by a trusted third party, it samples common reference string crs for NIZK and public parameters pp′ for IPFE.
- AdGen: The seller samples $(\mathsf{mpk}, \mathsf{msk})$ corresponding to IND-secure IPFE and random coins $\mathbf{t}$ used by the non-black-box IPFE compiler to upgrade from IND-secure IPFE to simulation-secure IPFE. It computes the elongated vector $\widetilde{\mathbf{x}}$ used by the IPFE compiler, encrypts $\widetilde{\mathbf{x}}$ to obtain ct and computes a NIZK proof π certifying that ct encrypts a witness corresponding to X.
- AdVerify: The buyer verifies the NIZK proof π.
- Interactive pre-signing: For the sake of notational simplicity, we denote the 3-round interactive functional pre-signing via three algorithms: AuxGen, AuxVerify, FPreSign.
  - First round: the buyer sends the function $\mathbf{y}$ to the seller.
  - Second round: the seller runs the AuxGen algorithm and sends auxiliary value $\mathsf{aux}_{\mathbf{y}}$ along with a proof $\pi_{\mathbf{y}}$ validating authenticity of $\mathsf{aux}_{\mathbf{y}}$ to the buyer. Specifically, the seller uses the random coins $\mathbf{t}$ of the IPFE compiler to compute $f_{\mathbf{y}}(\mathbf{t})$. Then, it sets the elongated vector $\widetilde{\mathbf{y}} = (\mathbf{y}^T, f_{\mathbf{y}}(\mathbf{t}))^T$ and generates $\mathsf{pk}_{\mathbf{y}}$ for it. It sends $(\mathsf{aux}_{\mathbf{y}}, \pi_{\mathbf{y}}) := (\mathsf{pk}_{\mathbf{y}}, f_{\mathbf{y}}(\mathbf{t}))$.

[4] Making the functional secret key generation algorithm stateless restricts the functionality in the LWE setting. In particular, such IPFE can only handle linearly independent function queries. We show how to slightly modify our FAS construction to ensure that linearly dependent function queries from different buyers can be augmented with some extra slots that always guarantee linear independence of function queries to the underlying IPFE. See the full version for more details.

[5] DDH-based IPFE scheme of [11] has been shown to be simulation-secure in [10]. Interestingly, it does not introduce randomness in functional key-generation and for proving simulation-security, the simulator relies on the fact that there can be many master secret keys corresponding to a master public key. But unfortunately, it does not satisfy the IPFE compliance property as its master keys do not satisfy the discrete log relation. Hence, it is incompatible with Schnorr adaptor signatures. Despite many attempts, we could not make the two compatible.

  - Third round: the buyer verifies the auxiliary value via AuxVerify algorithm, i.e., it creates $\widetilde{\mathbf{y}} = (\mathbf{y}^T, \pi_{\mathsf{y}})^T$ and checks if $\mathsf{aux}_{\mathsf{y}}$ matches the output of $\mathsf{IPFE.PubKGen}(\mathsf{mpk}, \widetilde{\mathbf{y}})$. If it verifies, then, it computes pre-signature $\widetilde{\sigma}$ for the adaptor statement $\mathsf{aux}_{\mathsf{y}}$ and sends it to the seller.
- FPreVerify: The seller verfies that $\widetilde{\sigma}$ corresponds to $\mathbf{y}$, i.e., it verifies that $\widetilde{\sigma}$ corresponds to $\mathsf{aux}_{\mathsf{y}}$ and $\mathsf{aux}_{\mathsf{y}}$ corresponds to $\mathbf{y}$.
- Adapt: The seller computes IPFE functional key sk_{y} for function $\widetilde{\mathbf{y}}$. Since sk_{y} is a witness to pk_{y}, it uses sk_{y} to adapt $\widetilde{\sigma}$ into σ.
- FExt: The buyer extracts the IPFE functional key sk_{y} from $(\widetilde{\sigma}, \sigma)$ and uses it to decrypt ct and recover $f_{\mathbf{y}}(\mathbf{x})$.

Our formal construction is as in Figure 2. We defer the security theorem to Section 5. We alluded to in 'Our Technique 3' on how to avoid NIZKs in interactive pre-signing by making non-blackbox use of the IPFE compiler. Having described the full details of FAS construction, we now elucidate on it in Remarks 2.1 and 2.2.

Remark 2.1 (Simulatability with leakage $f_{\mathbf{y}}(\mathbf{t})$ on $\mathbf{t}$). *Note that while proving zero-knowledge of FAS, the FAS simulator would still have to reveal $f_{\mathbf{y}}(\mathbf{t})$. This helps us avoid NIZKs in the interactive pre-signing as discussed before. Also note that FAS simulator would use the IPFE simulator internally, and $f_{\mathbf{y}}(\mathbf{t})$ is a leakage on the random coins $\mathbf{t}$ of the IPFE (compiler's) simulator. Crucially, this does not break simulation security of IPFE. This is because the IPFE compiler of [10] — and hence the IPFE simulator too – inherently leaks $f_{\mathbf{y}}(\mathbf{t})$. The latter is because* IPFE *decryption implicitly takes $\widetilde{\mathbf{y}}$ as input and hence, the decrypter needs to know $f_{\mathbf{y}}(\mathbf{t})$ to ensure decryption correctness. Hence, from an honest seller's perspective, the* FAS *zero-knowledge simulation won't be affected by giving away $f_{\mathbf{y}}(\mathbf{t})$ as part of π_{y}.*

Remark 2.2 (Purpose of π_{y}). *Note that π_{y} enables verifying that $\mathsf{aux}_{\mathsf{y}}$ is functional public key corresponding to $\widetilde{\mathbf{y}} = (\mathbf{y}^T, \pi_{\mathsf{y}})^T$. Regarding $\widetilde{\mathbf{y}}$, the buyer knows $\mathbf{y}$ but it does not know whether the last slot of $\widetilde{\mathbf{y}}$ chosen by the seller is well-formed, i.e., is $\pi_{\mathsf{y}} = f_{\mathbf{y}}(\mathbf{t})$? In security against a malicious seller, we argue that we do not need to guarantee this. In particular, it does not affect an honest buyer whether $\pi_{\mathsf{y}} = f_{\mathbf{y}}(\mathbf{t})$ holds or not. Suppose that the malicous seller chooses arbitrary value* val *and sends $(\mathsf{aux}_{\mathsf{y}}, \pi_{\mathsf{y}}) = (\mathsf{pk}_{\mathsf{y}}, \mathsf{val})$, where $\mathsf{pk}_{\mathsf{y}} = \mathsf{IPFE.PubKGen}(\mathsf{mpk}, \widetilde{\mathbf{y}})$ and $\widetilde{\mathbf{y}} = (\mathbf{y}^T, \mathsf{val})^T$. This would certainly ensure that* AuxVerify *passes and the buyer continues the protocol with the seller. Crucially though, we argue that if the honest buyer does end up making the payment to the seller, then, the honest buyer indeed learns $f_{\mathbf{y}}(\mathbf{x})$ at the end of the protocol disregard of what* val *was chosen by the malicious seller. This is because* advertisement soundness *would guarantee the honest buyer that* ct *encrypts a vector $\widetilde{\mathbf{x}}$ of the form $\widetilde{\mathbf{x}} = (\mathbf{x}^T, 0)^T$, where $(X, \mathbf{x}) \in R$. Further, witness extractability would guarantee that the honest buyer extracts a functional secret key sk_{y} corresponding to $\widetilde{\mathbf{y}} = (\mathbf{y}^T, \mathsf{val})^T$. Thus, IPFE correctness would guarantee that the honest buyer recovers $f_{\widetilde{\mathbf{y}}}(\widetilde{\mathbf{x}})$ which is same as $f_{\mathbf{y}}(\mathbf{x})$ since the last slot of $\widetilde{\mathbf{x}}$ is zero.*

2.4 Instantiations

In Section 6, we provide an instantiation from prime order groups for the function class of inner products over integers with output values polynomially bounded by $B \ll p$. For example, if each entry of the witness/function vector is bounded by 10^3, and the vectors have 10^6 entries, then, the inner product value would be bounded by 10^{12}, so we can set $B = 10^{12}$. We explain below application scenarios where such restrictions are reasonable. We also present an instantiation from lattices in Section 7 that removes such restrictions. Specifically, it works for the function class of inner products modulo prime integer p. Further, it is post-quantum secure.

2.4.1 **Prime-order groups based Instantiation.** To instantiate our FAS construction in Figure 2 from prime order groups, we instantiate the AS as Schnorr adaptor signature scheme [21] w.r.t. the hard relation R_{DL}, where R_{DL} is the discrete-log relation in prime order groups. We then set $R_{\mathsf{IPFE}} = R_{\mathsf{DL}}$ and show that a varaiant of the selective, IND-secure IPFE scheme by Abdalla et al. [9] satisfies R_{DL}-compliance.

Small plaintext space: reason and application scope. This limitation is because the function output here is encoded in the exponent and functional extraction of FAS instantiation involves a discrete log computation step. For values in the exponent bounded by B, this incurs a running time of $O(\sqrt{B})$. This is efficient only if bound B is a polynomial. We argue that this is sufficient for several realistic application scenarios. We benchmarked computation for datasets having a maximum of 10^6 entries, resulting in a 32 MB dataset. For example, the breast cancer dataset on Kaggle [1] has 20000 entries, total size 50 KB, sum of entries $\approx 10^6$. Assuming each entry of a function is at most 10^4, we get $B = 10^{10}$. Thus, the benchmarks in Table 2 for $(\ell = 10^4, B = 10^{10})$ fit closest for this dataset. Other scenarios occur when the secret database is a company's employee record with their age, years of service with the employer, retirement contribution, etc. Such values while sensitive, are typically bounded (e.g., by $B = 10^{10}$) and the buyer may wish to learn statistical/aggregate information, like sum, weighted mean or average, etc. whose result is also in the bounded plaintext space. The buyer could be a research organization that is studying the workforce in companies or factories, or it could be a recruitment recommendation agency that tries to match clients with potential employers with appropriate aggregate requirements of the client.

2.4.2 **Lattice based Instantiation.** We instantiate the AS as the lattice-based adaptor signature scheme by Esgin et al. [22] w.r.t. the inhomogenous short integer solution relation R_{ISIS}. Our instantiation is over the ring of integers $\mathcal{R} = \mathbb{Z}$. Consequently, its security follows from plain SIS and LWE assumptions. Further, we show that a variant of the IND-secure IPFE scheme by Agrawal et al. [11, Section 4.2] satisfies R_{ISIS}-compliance.

For technical reasons related to IPFE, the resulting FAS construction can only support linearly independent function request across all buyers (See Remark 7.1 for details). This could be quite restrictive since the scheme cannot support same function requests from different buyers. In the full version, we show how to modify the FAS construction to remove this restriction.

2.5 Towards Non-Interactive Pre-Signing

One of the main features of adaptor signatures that enable scalability is the non-interactive nature of the pre-signature generation algorithm. More specifically, the buyer in an adaptor signature execution can generate a pre-signature just from the seller's advertisement and, more importantly, without further interaction with the seller. Since the exchange of pre-signatures is done off-chain

$\mathsf{Setup}(1^\lambda)$

1 : Sample $\mathsf{crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$
2 : Sample $\mathsf{pp}' \leftarrow \mathsf{IPFE.Gen}(1^\lambda)$
3 : **ret** $\mathsf{pp} := (\mathsf{crs}, \mathsf{pp}')$

$\mathsf{AdGen}(\mathsf{pp}, X, \mathbf{x})$:

1 : Sample random coins r_0, r_1
2 : Let $(\mathsf{mpk}, \mathsf{msk}) := \mathsf{IPFE.Setup}(\mathsf{pp}', 1^{\ell+1}; r_0)$
3 : Sample $\mathbf{t} \xleftarrow{\$} \mathbb{Z}_p^\ell$, let $\widetilde{\mathbf{x}} := (\mathbf{x}^T, 0)^T \in \mathbb{Z}_p^{\ell+1}$
4 : Let $\mathsf{ct} := \mathsf{IPFE.Enc}(\mathsf{mpk}, \widetilde{\mathbf{x}}; r_1)$
5 : Let $\mathsf{stmt} := (X, \mathsf{pp}', \mathsf{mpk}, \mathsf{ct})$, $\mathsf{wit} := (r_0, r_1, \mathbf{x})$
6 : Let $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit})$
7 : **ret** $\mathsf{advt} := (\mathsf{mpk}, \mathsf{ct}, \pi)$, $\mathsf{st} := (\mathsf{msk}, \mathbf{t})$

$\mathsf{AdVerify}(\mathsf{pp}, X, \mathsf{advt})$

1 : **ret** $\mathsf{NIZK.Vf}(\mathsf{crs}, (X, \mathsf{pp}', \mathsf{mpk}, \mathsf{ct}), \pi)$

$\mathsf{AuxGen}(\mathsf{advt}, \mathsf{st}, \mathbf{y})$

1 : Parse $\mathsf{advt} = (\mathsf{mpk}, \mathsf{ct}, \pi)$, $\mathsf{st} = (\mathsf{msk}, \mathbf{t})$
2 : Let $\widetilde{\mathbf{y}} := (\mathbf{y}^T, f_{\mathbf{y}}(\mathbf{t}))^T \in \mathbb{Z}_p^{\ell+1}$
3 : Let $\mathsf{pk}_{\mathbf{y}} := \mathsf{IPFE.PubKGen}(\mathsf{mpk}, \widetilde{\mathbf{y}})$
4 : **ret** $\mathsf{aux}_{\mathbf{y}} := \mathsf{pk}_{\mathbf{y}}$, $\pi_{\mathbf{y}} := f_{\mathbf{y}}(\mathbf{t})$

$\mathsf{AuxVerify}(\mathsf{advt}, \mathbf{y}, \mathsf{aux}_{\mathbf{y}}, \pi_{\mathbf{y}})$

1 : Parse $\mathsf{advt} = (\mathsf{mpk}, \mathsf{ct}, \pi)$, let $\widetilde{\mathbf{y}} := (\mathbf{y}^T, \pi_{\mathbf{y}})^T$
2 : **ret** 1 iff $\mathsf{aux}_{\mathbf{y}} = \mathsf{IPFE.PubKGen}(\mathsf{mpk}, \widetilde{\mathbf{y}})$

$\mathsf{FPreSign}(\mathsf{advt}, \mathsf{sk}, m, X, \mathbf{y}, \mathsf{aux}_{\mathbf{y}})$

1 : **ret** $\widetilde{\sigma} \leftarrow \mathsf{AS.PreSign}(\mathsf{sk}, m, \mathsf{aux}_{\mathbf{y}})$

$\mathsf{FPreVerify}(\mathsf{advt}, \mathsf{vk}, m, X, \mathbf{y}, \mathsf{aux}_{\mathbf{y}}, \pi_{\mathbf{y}}, \widetilde{\sigma})$

1 : **ret** $\mathsf{AuxVerify}(\mathsf{advt}, \mathbf{y}, \mathsf{aux}_{\mathbf{y}}, \pi_{\mathbf{y}}) \wedge$
$\mathsf{AS.PreVerify}(\mathsf{vk}, m, \mathsf{aux}_{\mathbf{y}}, \widetilde{\sigma})$

$\mathsf{Adapt}(\mathsf{advt}, \mathsf{st}, \mathsf{vk}, m, X, \mathbf{x}, \mathbf{y}, \mathsf{aux}_{\mathbf{y}}, \widetilde{\sigma})$

1 : Parse $\mathsf{advt} = (\mathsf{mpk}, \mathsf{ct}, \pi)$, $\mathsf{st} = (\mathsf{msk}, \mathbf{t})$
2 : Let $\widetilde{\mathbf{y}} := (\mathbf{y}^T, f_{\mathbf{y}}(\mathbf{t}))^T$
3 : Let $\mathsf{sk}_{\mathbf{y}} := \mathsf{IPFE.KGen}(\mathsf{msk}, \widetilde{\mathbf{y}})$
4 : **ret** $\sigma := \mathsf{AS.Adapt}(\mathsf{vk}, m, \mathsf{aux}_{\mathbf{y}}, \mathsf{sk}_{\mathbf{y}}, \widetilde{\sigma})$

$\mathsf{FExt}(\mathsf{advt}, \widetilde{\sigma}, \sigma, X, \mathbf{y}, \mathsf{aux}_{\mathbf{y}})$

1 : Parse $\mathsf{advt} = (\mathsf{mpk}, \mathsf{ct}, \pi)$.
2 : Let $z := \mathsf{AS.Ext}(\widetilde{\sigma}, \sigma, \mathsf{aux}_{\mathbf{y}})$
3 : **ret** $v := \mathsf{IPFE.Dec}(z, \mathsf{ct})$

Figure 2: Construction: Functional Adaptor Signatures

and hence the latency doesn't impact the blockchain eco-system, a non-interactive pre-signing phase is more preferable.

We relaxed pre-signing to be interactive to achieve strong witness privacy for FAS, namely, the malicious buyer learns no information about $\mathbf{x}$ other than $f(\mathbf{x})$. This relaxation is not new to our work and was already introduced in [36] to achieve the advanced feature of *blindness* for adaptor signatures. Despite numerous attempts, the interactive nature of pre-signing seems necessary for this notion of strong privacy. An immediate challenge towards non-interactive pre-signing would be to build a simulation-secure IPFE scheme where the buyer can itself generate $\mathsf{pk}_f \leftarrow \mathsf{IPFE.PubKGen}(\mathsf{mpk}, f)$ such that the seller can generate the corresponding functional secret-key $\mathsf{sk}_f \leftarrow \mathsf{IPFE.KGen}(\mathsf{msk}, f)$. Unfortunately, the randomized nature of the key-generation algorithm of simulation-secure scheme is a major hurdle to enabling this.

We explore whether a non-interactive pre-signing can be achieved for FAS for relaxed privacy definitions. Towards this, we study FAS that only satisfies *witness indistinguishability*. The construction of such a FAS follows from our construction template by replacing the simulation-secure IPFE with an appropriate IND-secure IPFE. To understand the intuition, we refer the reader to the IPFE compliance example discussed in our technique 1 in Section 2.2. We provide more details on this construction in the full version.

3 PRELIMINARIES

We briefly recall the cryptographic tools needed in this work.

Notations. We denote by $x \leftarrow S$ the experiment of sampling x from a probability distribution S. We denote by $[A(\cdot)]$ the range of an algorithm $A(\cdot)$. If $p(\cdot,\cdot)$ denotes a predicate, then $\Pr[p(y,z) : x \leftarrow S, (y,z) \leftarrow A(x)]$ is the probability that the predicate $p(y,z)$ is true after the ordered sequence of events $x \leftarrow S$ followed by $(y,z) \leftarrow A(x)$. We denote scalars by lower-case alphabets such as x, vectors by bold-face lower-case alphabets such as $\mathbf{x}$, matrices by bold-face upper-case alphabets such as $\mathbf{X}$. $\mathbf{x} = (x_1, x_2)^T$ denotes a column-vector with elements x_1 and x_2. $\mathbf{x}^T$ denotes a row-vector.

Linear Functions. We denote linear functions as inner product computation. Let $\mathcal{F}_{\mathsf{IP},\ell}$ denote the family of inner products of vectors of length ℓ. For a prime p, in this work we study two restrictions of inner product computations as follows:

- *Inner products modulo p.* The function class $\mathcal{F}_{\mathsf{IP},\ell,p} = \{f_{\mathbf{y}} : \mathbf{y} \in \mathbb{Z}_p^\ell\}$, where $f_{\mathbf{y}} : \mathbb{Z}_p^\ell \to \mathbb{Z}_p$ is defined as $f_{\mathbf{y}}(\mathbf{x}) = \mathbf{x}^T\mathbf{y} \bmod p$.
- *Inner products with output values polynomially bounded by $B \ll p$.* The function class $\mathcal{F}_{\mathsf{IP},\ell,p,B} = \{f_{\mathbf{y}} : \mathbf{y} \in \mathbb{Z}_p^\ell\}$, where $f_{\mathbf{y}} : \mathbb{Z}_p^\ell \to \{0,\ldots,B\}$ is defined as $f_{\mathbf{y}}(\mathbf{x}) = \mathbf{x}^T\mathbf{y} \in \{0,\ldots,B\}$.

Often we will use $\mathbf{y}$ instead of $f_{\mathbf{y}}$ to denote the function.

Digital Signature. A digital signature scheme $\mathsf{DS} := (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vf})$ has a key generation algorithm $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$ that outputs a verification-signing key pair. Using signing key sk we can compute signatures on a message m by running $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$, which can be publicly verified using the corresponding verification key vk by running $\mathsf{Vf}(\mathsf{vk}, m, \sigma)$. We require the digital signature scheme to satisfy strong existential unforgeability [27].

Hard Relations. We recall the notion of a hard relation R with statement/witness pairs (X, x). We denote by L_R the associated language defined as $L_R := \{ X \mid \exists\, x, (X,x) \in R \}$. We want the following properties from the hard relation: (1) Statement-witness pairs $(X,x) \in R$ can be efficiently sampled using a sampling algorithm $\mathsf{GenR}(1^\lambda)$, (2) for all PPT adversaries $\mathcal{A}$ the probability of $\mathcal{A}$ on input X outputting a witness x' such that $(X,x') \in R$ is negligible, and (3) for a given function class $\mathcal{F}$, for any PPT adversaries $\mathcal{A}$ that is given input X, and returns (f,z), it must be the case that $(f \in \mathcal{F}) \wedge (z \in \{f(x') : \exists\, x' \text{ such that } (X,x') \in R\})$ only with negligible probability. If a relation satisfies only the first two properties, we call it *hard*, and if it satisfies all three, it we call it *$\mathcal{F}$-hard*. The formal definitions are described in the full version. In this work, we use the discrete log language L_{DL} defined with respect to a group $\mathbb{G}$ with generator g and order p. The language is defined as $L_{\mathsf{DL}} := \{Y \mid \exists y \in \mathbb{Z}_p,\ Y = g^y\}$ with corresponding hard relation $R_{\mathsf{DL}} = \{(X,x) | X = g^x\}$.

Adaptor Signatures (AS). Adaptor Signatures were first formally defined in [12]. [22] generalized the syntax in [12] to enable constructing adaptor signatures from lattices. More concretely, [12] defined adaptor signatures w.r.t. a digital signature scheme and a hard

relation R. [22] generalized it to be w.r.t. two hard relations R and R' such that such that $R \subseteq R'$. For the group-based constructions, typically we have $R = R'$, but for the lattice based constructions, typically we have $R \neq R'$. In this work, one of our constructions will be from lattices, hence, we follow this generalized syntax.

In a different vein, [19, 25] have strengthened the security properties of adaptor signatures, with notions like *unforgeability*, *pre-signature extractability*, and *witness extractability*. [19] further added *unique extractability* and *unlinkability* as security properties and unified *unforgeability* and *witness extractability* under a single security property called *extractability*. [25] further added *pre-verify soundness* as a security property. We note that not all security properties are needed always. Within the scope of this work, we will use adaptor signatures as a building block for the construction of FAS and we only need *pre-signature adaptability* and *witness extractability* security properties of adaptor signatures. Intuitively, pre-signature adaptability ensures that given a valid pre-signature and a witness for the statement, one can always adapt the pre-signature into a valid signature. Witness extractability requires that given an honestly generated pre-signature and a valid signature, one should be able to extract a witness. The formal definitions of the properties are deferred to the full version.

Non-Interactive Zero-Knowledge (NIZK) Arguments. A NIZK for the NP language L_R in the *common reference string (CRS)* model consists of algorithms $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Vf})$. $\mathsf{Setup}(1^\lambda)$ outputs the common reference string crs. $\mathsf{Prove}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit})$ outputs a proof π that statement $\mathsf{stmt} \in L_R$ using a witness wit. The validity of the proof can be checked by $\mathsf{Vf}(\mathsf{crs}, \mathsf{stmt}, \pi)$. We require the NIZK argument system to be (1) *complete*, where honestly generated proofs for $(\mathsf{stmt}, \mathsf{wit}) \in R$ verify successfully, (2) *zero-knowledge*, where a malicious verifier does not learn more than the validity of the statement stmt, i.e., there exists a simulator Sim that without any information of the witness, can create an ideal-world view that the verifier can't distringuish from its real-world view, and (3) *adaptive soundness*, where it is hard for any p.p.t. prover to convince a verifier of an invalid statement that is chosen by the prover after receiving crs. For formal definitions, we refer the reader to the full version. Such NIZK arguments are known from factoring [24], and standard assumptions on pairings [18, 28] and lattices [17, 34].

Inner Product Functional Encryption (IPFE) We will need a functional encryption scheme for computing inner products of vectors of length ℓ. Let the message space be $\mathcal{M} \subseteq \mathbb{Z}^\ell$ and function space $\mathcal{F}_{\mathsf{IP},\ell}$. Messages are denoted by $\mathbf{x} \in \mathcal{M}$, functions are denoted by $\mathbf{y} \in \mathcal{F}_{\mathsf{IP},\ell}$ and the function evaluation is denoted by $f_{\mathbf{y}}(\mathbf{x})$. An IPFE scheme is a tuple of five algorithms $(\mathsf{Gen}, \mathsf{Setup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$: $\mathsf{Gen}(1^\lambda)$ algorithm samples public parameters pp. $\mathsf{Setup}(\mathsf{pp}, \ell)$ algorithm samples master public key mpk and master secret key msk. $\mathsf{KGen}(\mathsf{msk}, \mathbf{y})$ is a deterministic algorithm that outputs a functional secret key $\mathsf{sk}_{\mathbf{y}}$ for function vector $\mathbf{y}$. $\mathsf{Enc}(\mathsf{mpk}, \mathbf{x})$ algorithm outputs a ciphertext ct encrypting message vector $\mathbf{x}$. $\mathsf{Dec}(\mathsf{sk}_{\mathbf{y}}, \mathsf{ct})$ algorithm can be used to decrypt ciphertext ct to reveal a value v using functional secret key $\mathsf{sk}_{\mathbf{y}}$. We require IPFE to satisfy (1) *correctness*, where $v = f_{\mathbf{y}}(\mathbf{x})$ if everything was honestly computed, (2) *selective, IND-security*, where the adversary commits to challenge messages $(\mathbf{x}_0^*, \mathbf{x}_1^*)$ at the beginning of the game and obtains $\mathsf{mpk}, \mathsf{ct}^*$, where ct^* encrypts $\mathbf{x}_b^*$ for $b \in \{0, 1\}$ unknown to adversary. Then, the adversary can make polynomially many KGen queries for functions $\mathbf{y}$ satisfying $f_{\mathbf{y}}(\mathbf{x}_0^*) = f_{\mathbf{y}}(\mathbf{x}_1^*)$. At the end of the game, the adversary wins if it can correctly guess the bit b. We defer the formal definitions to the full version.

Further, we augment the IPFE scheme with an additional algorithm PubKGen defined as follows. $\mathsf{PubKGen}(\mathsf{mpk}, \mathbf{y} \in \mathcal{F}_{\mathsf{IP},\ell})$ is a *deterministic* algorithm that takes in the master public key mpk, and a vector $\mathbf{y} \in \mathcal{F}_{\mathsf{IP},\ell}$, and outputs a functional public key $\mathsf{pk}_{\mathbf{y}}$.

Remark 3.1. *We note that* $\mathsf{pk}_{\mathbf{y}}$ *does not enable decryption, so it does not change the correctness requirement. It also does not affect security of IPFE. To see this, note that* PubKGen *is deterministic. So, an adversary can always compute* $\mathsf{pk}_{\mathbf{y}}$ *on its own. Given that* KGen *is also deterministic, one way to think of* $\mathsf{pk}_{\mathbf{y}}$ *is as a commitment to* $\mathsf{sk}_{\mathbf{y}}$. *Typically,* mpk *also has this property w.r.t.* msk *and it is always implicit in the security definitions of IPFE. We formalize this compliance requirement for* $\mathsf{pk}_{\mathbf{y}}$ *and* $\mathsf{sk}_{\mathbf{y}}$ *below since FAS will use it.*

Definition 3.2 (R_{IPFE}-Compliant IPFE). *For a hard relation* R_{IPFE}*, we say that an IPFE scheme is* R_{IPFE} *compliant if for any* $\lambda \in \mathbb{N}$*, for any* $\mathsf{pp} \leftarrow \mathsf{IPFE.Gen}(1^\lambda)$*, for any* $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{IPFE.Setup}(\mathsf{pp}, 1^\ell)$*, for any* $\mathbf{y} \in \mathcal{F}_{\mathsf{IP},\ell}$*, let* $\mathsf{pk}_{\mathbf{y}} := \mathsf{PubKGen}(\mathsf{mpk}, \mathbf{y})$*, and* $\mathsf{sk}_{\mathbf{y}} := \mathsf{KGen}(\mathsf{msk}, \mathbf{y})$*. Then, it must be the case that* $(\mathsf{pk}_{\mathbf{y}}, \mathsf{sk}_{\mathbf{y}}) \in R_{\mathsf{IPFE}}$.

Next, we describe an additional correctness property of IPFE that is tailored to make IPFE compatible with adaptor signatures (AS) when the two are used in tandem in our FAS construction. Looking ahead, we need this property because in our FAS construction, we will use an AS to sell functional secret keys. Recall that AS are defined w.r.t. two relations R and R'. In this case, R will be R_{IPFE} as defined in Definition 3.2 and we denote the extended relation R' by R'_{IPFE}. Then, the AS.Ext algorithm will extract a functional secret key $\mathsf{sk}'_{\mathbf{y}}$ as the witness for $\mathsf{pk}_{\mathbf{y}}$. While $\mathsf{sk}'_{\mathbf{y}}$ will be a witness of R'_{IPFE}, it may not be a witness of the base relation R_{IPFE} since $R_{\mathsf{IPFE}} \subseteq R'_{\mathsf{IPFE}}$, and we still want that $\mathsf{sk}'_{\mathbf{y}}$ should enable meaningful decryption. We formalize this robustness requirement below.

Definition 3.3 (R'_{IPFE}-Robust IPFE). *Let* R_{IPFE} *be the hard relation as defined in Definition 3.2. Let* R'_{IPFE} *such that* $R_{\mathsf{IPFE}} \subseteq R'_{\mathsf{IPFE}}$ *be an an extended relation of* R_{IPFE}*. We say that an IPFE scheme is* R'_{IPFE} *robust if IPFE correctness holds even w.r.t. a functional secret key satisfying* R'_{IPFE}*. More formally, for any* $\lambda \in \mathbb{N}$*, let* $\mathsf{pp} \leftarrow \mathsf{IPFE.Gen}(1^\lambda)$*, for any* $\ell \in \mathbb{N}$*,* $\mathbf{x} \in \mathcal{M}, \mathbf{y} \in \mathcal{F}_{\mathsf{IP},\ell}$*, let* $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(\mathsf{pp}, \ell)$*,* $\mathsf{pk}_{\mathbf{y}} := \mathsf{PubKGen}(\mathsf{msk}, \mathbf{y})$*,* $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathbf{x})$*. For any* $\mathsf{sk}'_{\mathbf{y}}$*, if* $(\mathsf{pk}_{\mathbf{y}}, \mathsf{sk}'_{\mathbf{y}}) \in R'_{\mathsf{IPFE}}$*, then,* $\mathsf{Dec}(\mathsf{sk}'_{\mathbf{y}}, \mathsf{ct})$ *outputs* $f_{\mathbf{y}}(\mathbf{x})$.

4 FAS DEFINITION

A functional adaptor signature scheme shares similar syntax to an adaptor signature scheme with few additional interfaces and parameters. Specifically, a functional adaptor signature scheme $\mathsf{FAS} := (\mathsf{Setup}, \mathsf{AdGen}, \mathsf{AdVerify}, \mathsf{AuxGen}, \mathsf{AuxVerify}, \mathsf{FPreSign}, \mathsf{FPreVerify}, \mathsf{Adapt}, \mathsf{FExt})$ is defined with respect to a signature scheme $\mathsf{DS} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vf})$, a hard relation R, and a family of functions $\mathcal{F}$. We recall that $(\mathsf{AuxGen}, \mathsf{AuxVerify}, \mathsf{FPreSign})$ essentially denote the 3-round interactive pre-signing process. We defer formal description of syntax to the full version due to space constraints.

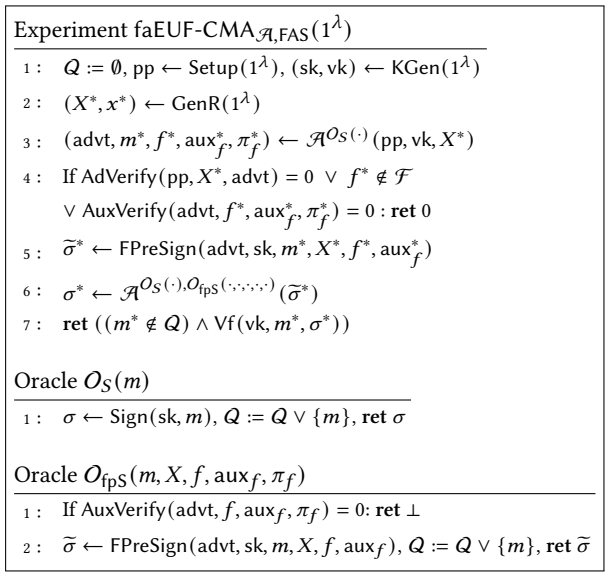


Figure 3: Unforgeability of functional adaptor signatures

Correctness. If everything is run honestly, then, FExt must return $f(x)$. We defer the formal details to the full version.

Security. Due to space constraints, we defer all properties (mentioned in Section 2.1.1) except unforgeability and zero-knowledge witness privacy to the full version.

Unforgeability. The property requires that even when the adversary (malicious seller) is given access to pre-signatures with respect to functions f of its choice, it should not be able to forge signatures if it does not know the witness x^* for the challenge statement X^* in the first place. To capture this essence, the experiment (See Figure 3) samples (X^*, x^*) at random. Beyond this, the adversary has all the powers of a (malicious) seller starting with generating advt. A curious reader might ask if the adversary doesn't know x^*, then, wouldn't it typically fail to pass the AdVerify check in step 4 in Figure 3 and thus rendering the experiment after that pointless? The answer is no, because *advertisement soundness* is for statements not in the language L_R, but here $X^* \in L_R$ as the experiment chooses it.

Definition 4.1 (Unforgeability). *A* FAS *scheme is* faEUF-CMA*-secure or simply unforgeable, if for every stateful p.p.t. adversary $\mathcal{A}$ there exists a negligible function* negl *such that for all $\lambda \in \mathbb{N}$,* $\Pr[\mathsf{faEUF\text{-}CMA}_{\mathcal{A},\mathsf{FAS}}(1^\lambda) = 1] \leq \mathsf{negl}(\lambda)$*, where the experiment* $\mathsf{faEUF\text{-}CMA}_{\mathcal{A},\mathsf{FAS}}$ *is defined as in Figure 3.*

Witness Privacy. We want that in an interaction between a malicious buyer wanting to learn function evaluation f on the witness and an honest seller using witness x to adapt the pre-signature into a valid signature, at the end the buyer learns no information beyond $f(x)$. The strongest way to capture this is via a simulation-style definition which we define as zero-knowledge. Informally, zero-knowledge requires that there exists a p.p.t. simulator Sim such that the adversary's can't guess whether it is interacting with a real-world challenger that uses witness x or an ideal-world challenger that only uses $f(x)$ via simulator Sim. Formal details follow.

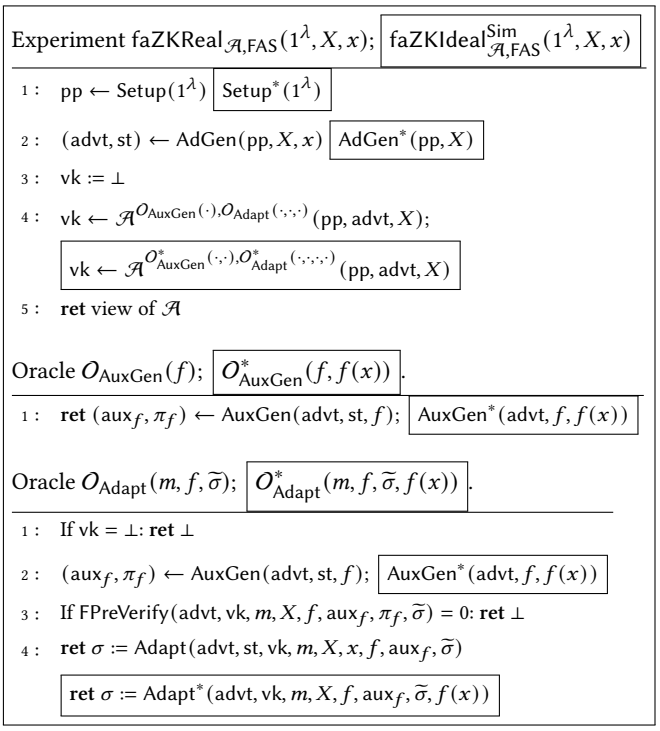


Figure 4: Zero-Knowledge experiments of FAS

Definition 4.2 (Zero-Knowledge). *A* FAS *scheme satisfies zero-knowledge if for every p.p.t. adversary $\mathcal{A}$, there exists a stateful p.p.t. simulator* $\mathsf{Sim} = (\mathsf{Setup}^*, \mathsf{AdGen}^*, \mathsf{AuxGen}^*, \mathsf{Adapt}^*)$*, there exists a negligible function* negl *s.t. for all p.p.t. distinguishers $\mathcal{D}$, for all $\lambda \in \mathbb{N}$, for all $(X, x) \in R$,* $|\Pr[\mathcal{D}(\mathsf{faZKReal}_{\mathcal{A},\mathsf{FAS}}(1^\lambda, X, x)) = 1] - \Pr[\mathcal{D}(\mathsf{faZKIdeal}^{\mathsf{Sim}}_{\mathcal{A},\mathsf{FAS}}(1^\lambda, X, x)) = 1]| \leq \mathsf{negl}(\lambda)$*, where experiments* $\mathsf{faZKReal}_{\mathcal{A},\mathsf{FAS}}$ *and* $\mathsf{faZKIdeal}^{\mathsf{Sim}}_{\mathcal{A},\mathsf{FAS}}$ *are defined in Figure 4.*

Definition 4.3 (Strongly-secure Functional Adaptor Signature Scheme). *A* FAS *scheme is strongly-secure if it is advertisement sound, pre-signature valid, witness extractable,* faEUF-CMA *secure, pre-signature adaptable and zero-knowledge.*

5 STRONGLY-SECURE FAS CONSTRUCTION

In this section, we build a generic construction of FAS w.r.t. digital signature scheme DS, any NP relation R with statement/witness pairs $(X, \mathbf{x}) \in R$ such that $\mathbf{x} \in \mathcal{M}$ for some set $\mathcal{M} \subseteq \mathbb{Z}^\ell$, and function family $\mathcal{F}_{\mathsf{IP},\ell}$ for computing inner products of vectors of length ℓ. Our generic construction of FAS is presented in Figure 2 [6] and it uses the following building blocks:

- A selective, IND-secure IPFE satisfying R_{IPFE}-compliance and R'_{IPFE}-robustness (Definitions 3.2 and 3.3) w.r.t. hard relations R_{IPFE} and R'_{IPFE} such that $R_{\mathsf{IPFE}} \subseteq R'_{\mathsf{IPFE}}$. The message space of IPFE is $\mathcal{M}' \subseteq \mathbb{Z}^{\ell+1}$ and the function family is $\mathcal{F}_{\mathsf{IP},\ell+1}$.

[6] Figure 2 is presented with $\mathcal{M} = \mathbb{Z}_p^\ell$, but the scheme generalizes to $\mathcal{M} \subseteq \mathbb{Z}^\ell$ as well.

- An adaptor signature scheme AS w.r.t. a digital signature scheme DS, and hard relations R_{IPFE} and R'_{IPFE} that satisfies witness extractability and weak pre-signature adaptability security properties.
- A non-interactive zero-knowledge argument system NIZK in the common reference string model for the NP language L_{NIZK}:

$$L_{\mathsf{NIZK}} := \left\{ \begin{array}{c} (X, \mathsf{pp}', \mathsf{mpk}, \mathsf{ct}) : \\ \exists (r_0, r_1, \mathbf{x}) \text{ such that } \mathsf{pp}' \in [\mathsf{IPFE.Gen}(1^\lambda)], \\ (\mathsf{mpk}, \mathsf{msk}) = \mathsf{IPFE.Setup}(\mathsf{pp}', 1^{\ell+1}; r_0), \\ (X, \mathbf{x}) \in R, \mathsf{ct} = \mathsf{IPFE.Enc}(\mathsf{mpk}, (\mathbf{x}^T, 0)^T; r_1) \end{array} \right\}.$$

We sketch the security proof of unforgeability and zero-knowledge witness privacy below. Correctness and formal proofs are deferred to the full version.

Lemma 5.1. *Suppose NIZK satisfies correctness,* AS *satisfies correctness, and* IPFE *satisfies R_{IPFE}-compliance and R'_{IPFE}-robustness. Then, the* FAS *construction in Figure 2 satisfies correctness.*

Theorem 5.2. *Let $\mathcal{F}_{\mathsf{IP},\ell}$ be the family of inner products functions of vectors of length ℓ. Let R be any NP relation with statement/witness pairs $(X, \mathbf{x})$ such that $\mathbf{x} \in \mathcal{M}$ for some set $\mathcal{M} \subseteq \mathbb{Z}^\ell$. Suppose that*

- *$\mathcal{M}$ is an additive group, R is $\mathcal{F}_{\mathsf{IP},\ell}$-hard,*
- NIZK *is a secure NIZK argument system,*
- AS *is an adaptor signature scheme w.r.t. digital signature scheme* DS *and hard relations $R_{\mathsf{IPFE}}, R'_{\mathsf{IPFE}}$ satisfying weak pre-signature adaptability, witness extractability,*
- IPFE *is a selective, IND-secure IPFE scheme for function family $\mathcal{F}_{\mathsf{IP},\ell+1}$ satisfying R_{IPFE}-compliance (Definition 3.2), R'_{IPFE}-robustness (Definition 3.3).*

Then, the construction in Figure 2 is a strongly-secure (Definition 4.3) functional adaptor signature scheme w.r.t. digital signature scheme DS*, NP relation R, and family of inner product functions $\mathcal{F}_{\mathsf{IP},\ell}$.*

Proof sketch of Unforgeability. A natural idea for proving unforgeability of FAS would be to somehow reduce it to unforgeability of AS. But, this requires non-blackbox use of the underlying AS, and thus results in a complex proof. We avoid that altogether and present a counter-intuitive yet elegant way of proving unforgeability of FAS by reducing it to witness extractability of AS. Our proof technique does not need to open up the blackbox of AS as evident in the sequence of games below.

In a little more detail, to show that $\Pr[G_0(1^\lambda) = 1] \leq \mathsf{negl}(\lambda)$, we consider a sequence of games G_0, G_1, G_2 as decribed in Figure 5.

- Game G_0 is the original game $\mathsf{faEUF\text{-}CMA}_{\mathcal{A},\mathsf{FAS}}$, where the adversary $\mathcal{A}$ has to come up with a valid forgery on a message m^* of his choice, while having access to functional pre-sign oracle O_{fpS} and sign oracle O_S. Here, variables $\mathsf{Bad}_1, \mathsf{Bad}_2$ do not affect the game and are used only to aide the analysis below.
- Game G_1 is same as G_0, except that before computing the pre-signature, the game checks if the NIZK statement $(X^*, \mathsf{pp}', \mathsf{mpk}, \mathsf{ct})$ is in the language L_{NIZK}. If no, the game sets the flag $\mathsf{Bad}_1 = \mathsf{true}$ and returns 0. Observe that G_0 and G_1 are identical until G_1 checks the condition for setting Bad_1. Hence,

$$|\Pr[G_0(1^\lambda) = 1] - \Pr[G_1(1^\lambda) = 1]| \leq \Pr[\mathsf{Bad}_1 \text{ in } G_1].$$

Thus, for proving $|\Pr[G_0(1^\lambda) = 1] - \Pr[G_1(1^\lambda) = 1]| \leq \mathsf{negl}(\lambda)$, it suffices to show that in game G_1, $\Pr[\mathsf{Bad}_1] \leq \mathsf{negl}(\lambda)$. $\mathsf{Bad}_1 =$ true implies stmt $\notin L_{NIZK}$ and $\mathsf{NIZK.Vf}(\mathsf{crs}, \mathsf{stmt}, \pi) = 1$. Thus, the probability bound follows from the adaptive soundness of NIZK argument system.

- Game G_2 is same as G_1, except that when $\mathcal{A}$ outputs the forgery σ^*, the game extracts the witness z of the underlying adaptor signature scheme for the statement $\mathsf{pk}_{\mathbf{y}^*}$ and checks if the NIZK statement $(\mathsf{aux}^*_{\mathbf{y}}, z)$ satisfies R'_{IPFE}. If no, the game sets the flag $\mathsf{Bad}_2 = \mathsf{true}$ and returns 0. Observe that G_0 and G_1 are identical until G_1 checks the condition for setting Bad_1. Hence,

$$|\Pr[G_1(1^\lambda) = 1] - \Pr[G_2(1^\lambda) = 1]| \leq \Pr[\mathsf{Bad}_2 \text{ in } G_2].$$

Thus, for proving $|\Pr[G_1(1^\lambda) = 1] - \Pr[G_2(1^\lambda) = 1]| \leq \mathsf{negl}(\lambda)$, it suffices to show that in game G_2, $\Pr[\mathsf{Bad}_2] \leq \mathsf{negl}(\lambda)$. This follows from the witness extractability of the underlying adaptor signature scheme AS. In a little more detail, we can show that if an adversary $\mathcal{A}$ successfully causes G_2 to set Bad_2, then, we can use it to come up with a reduction $\mathcal{B}$ that breaks the witness extractability of the underlying adaptor signature scheme. This is because $\mathsf{Bad}_2 = \mathsf{true}$ implies $(m^* \notin Q) \wedge ((\mathsf{aux}^*_{\mathbf{y}}, z) \notin R'_{\mathsf{IPFE}}) \wedge \mathsf{Vf}(\mathsf{vk}, m^*, \sigma^*)$, i.e., the winning condition for $\mathcal{B}$.
- Lastly, we can show that $\Pr[G_2(1^\lambda) = 1] \leq \mathsf{negl}(\lambda)$ assuming $\mathcal{F}_{\mathsf{IP},\ell}$-hardness of relation R and R'_{IPFE}-robustness of the IPFE scheme. Essentially, we show that if $\mathcal{A}$ wins G_2, then, we can

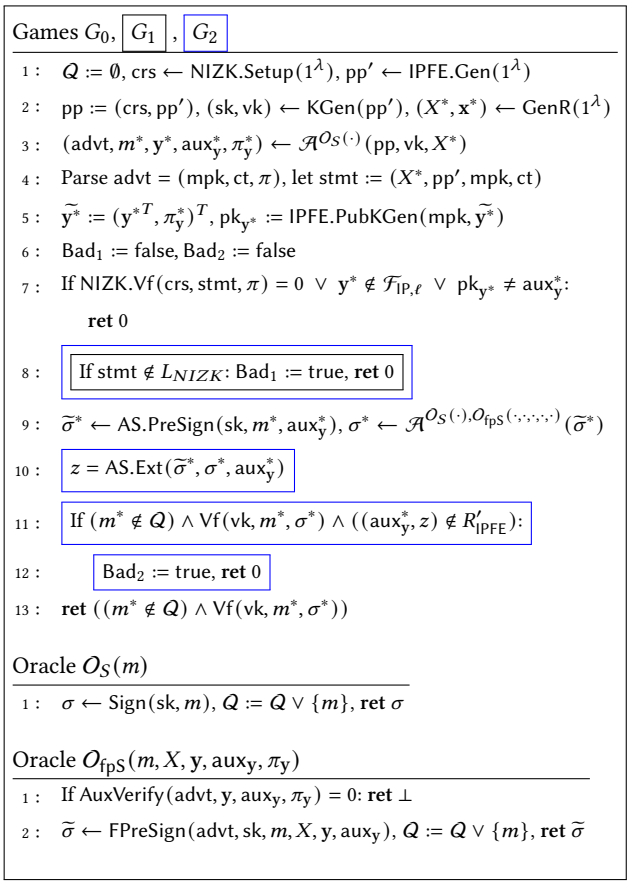


Figure 5: Unforgeability Proof: Games G_0, G_1, G_2

build a reduction $\mathcal{B}$ that breaks the $\mathcal{F}_{\mathsf{IP},\ell}$-hardness of relation R. $\mathcal{B}$ outputs $(\mathbf{y}^*, v)$, where $v = \mathsf{IPFE.Dec}(z, \mathsf{ct})$. To win, $\mathcal{B}$'s output must satisfy $(\mathbf{y}^* \in \mathcal{F}_{\mathsf{IP},\ell}) \wedge (v \in \{f_{\mathbf{y}^*}(\mathbf{x}) : \exists \mathbf{x}\ s.t.\ (X, \mathbf{x}) \in R\})$. If $\mathcal{A}$ wins, then $\mathbf{y}^* \in \mathcal{F}_{\mathsf{IP},\ell}$ and $\mathsf{pk}_{\mathbf{y}^*} = \mathsf{aux}^*_{\mathbf{y}}$. Next, $\mathsf{Bad}_1 = \text{false}$ implies that $\mathsf{stmt} \in L_{NIZK}$, where $\mathsf{stmt} = (X^*, \mathsf{pp}', \mathsf{mpk}, \mathsf{ct})$. Hence, it follows that ct encrypts some vector $\widetilde{\mathbf{x}} = (\mathbf{x}^T, 0)^T \in \mathcal{M}' \subseteq \mathbb{Z}^{\ell+1}_p$ under mpk such that $(X^*, \mathbf{x}) \in R$. Next, $\mathsf{Bad}_2 = \text{false}$ implies that $(\mathsf{aux}^*_{\mathbf{y}}, z) \in R'_{\mathsf{IPFE}}$. As $\mathsf{pk}_{\mathbf{y}^*} = \mathsf{aux}^*_{\mathbf{y}}$ and IPFE satisfies R'_{IPFE}-robustness, it follows that $v = f_{\widetilde{\mathbf{y}}^*}(\widetilde{\mathbf{x}})$. As $\widetilde{\mathbf{y}}^* = (\mathbf{y}^{*T}, \pi^*_{\mathbf{y}})^T$ and $\widetilde{\mathbf{x}} = (\mathbf{x}^T, 0)^T$, we get that $f_{\widetilde{\mathbf{y}}^*}(\widetilde{\mathbf{x}}) = f_{\mathbf{y}^*}(\mathbf{x})$. Hence, we conclude that $v \in \{f_{\mathbf{y}^*}(\mathbf{x}) : \exists \mathbf{x}\ s.t.\ (X, \mathbf{x}) \in R\}$. Thus, $\mathcal{B}$ wins its game. This completes the proof.

Proof sketch of Zero-Knowledge. We first describe the stateful simulator $\mathsf{Sim} = (\mathsf{Setup}^*, \mathsf{AdGen}^*, \mathsf{AuxGen}^*, \mathsf{Adapt}^*)$. Let the NIZK simulator be $\mathsf{NIZK.Sim} = (\mathsf{NIZK.Setup}^*, \mathsf{NIZK.Prove}^*)$. Then, the simulator Sim is as follows.

- $\mathsf{Setup}^*(1^\lambda)$: same as Setup except $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{NIZK.Setup}^*(1^\lambda)$ and trapdoor td is stored as internal state by Sim.
- $\mathsf{AdGen}^*(\mathsf{pp}, X)$: same as AdGen except $\widetilde{\mathbf{x}} := (-\mathbf{t}^T, 1)^T \in \mathbb{Z}^{\ell+1}_p$ and $\pi \leftarrow \mathsf{NIZK.Prove}^*(\mathsf{crs}, \mathsf{td}, (X, \mathsf{pp}', \mathsf{mpk}, \mathsf{ct}))$.
- $\mathsf{AuxGen}^*(\mathsf{advt}, \mathbf{y}, f_{\mathbf{y}}(\mathbf{x}))$: same as AuxGen except $\mathsf{pk}_{\mathbf{y}}$ is computed for $\widetilde{\mathbf{y}} := (\mathbf{y}^T, f_{\mathbf{y}}(\mathbf{t}) + f_{\mathbf{y}}(\mathbf{x}))^T \in \mathcal{F}_{\mathsf{IP},\ell+1}$ and thus $\pi_{\mathbf{y}} := f_{\mathbf{y}}(\mathbf{t}) + f_{\mathbf{y}}(\mathbf{x})$ as well.
- $\mathsf{Adapt}^*(\mathsf{advt}, \mathsf{vk}, m, X, \mathbf{y}, \mathsf{aux}_{\mathbf{y}}, \widetilde{\sigma}, f_{\mathbf{y}}(\mathbf{x}))$: same as Adapt except $\mathsf{sk}_{\mathbf{y}}$ for $\widetilde{\mathbf{y}} := (\mathbf{y}^T, f_{\mathbf{y}}(\mathbf{t}) + f_{\mathbf{y}}(\mathbf{x}))^T$ is used.

To see that adversary's views in real and ideal world (Figure 4) are indistinguishable, consider a sequence of games G_0, G_1, G_2, G_3.

- Game G_0 corresponds to the real-world experiment where Setup, AdGen, AuxGen, Adapt are used.
- Game G_1 is same as G_0, except the NIZK is switched to simulation mode, i.e., we change Setup to perform $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{NIZK.Setup}^*(1^\lambda)$ and change AdGen to perform $\pi \leftarrow \mathsf{NIZK.Prove}^*(\mathsf{crs}, \mathsf{td}, (X, \mathsf{pp}', \mathsf{mpk}, \mathsf{ct}))$. It follows from the zero-knowledge property of NIZK that $G_0 \approx_c G_1$.
- Game G_2 is same as G_1 except $\widetilde{\mathbf{y}}$ used in AuxGen and Adapt is switched from $\widetilde{\mathbf{y}} := (\mathbf{y}^T, f_{\mathbf{y}}(\mathbf{t}))^T$ to $\widetilde{\mathbf{y}} := (\mathbf{y}^T, f_{\mathbf{y}}(\mathbf{t}) + f_{\mathbf{y}}(\mathbf{x}))^T$. Since f is a linear function, it follows that $f_{\mathbf{y}}(\mathbf{t}) + f_{\mathbf{y}}(\mathbf{x}) = f_{\mathbf{y}}(\mathbf{t} + \mathbf{x})$, thus, one can view the transition to G_2 as a change of variables $\mathbf{t} \to \mathbf{t} + \mathbf{x}$. Since $\mathbf{t}$ is uniform random, it follows that G_1 and G_2 are identically distributed.
- Game G_3 is same as G_2 except that in AdGen, ct encrypts $\widetilde{\mathbf{x}} := (-\mathbf{t}^T, 1)^T$ instead of $\widetilde{\mathbf{x}} := (\mathbf{x}^T, 0)^T$. Observe that this does not change the inner product value $\widetilde{\mathbf{x}}^T\widetilde{\mathbf{y}}$. Thus, $G_2 \approx_c G_3$ follows from IND-security of the IPFE scheme.
- Lastly observe that adversary's view in G_3 is same as that in the ideal-world experiment. This complete the proof.

We emphasize that in G_3, even the underlying IPFE only uses the information $f_{\mathbf{y}}(\mathbf{x})$ about $\mathbf{x}$. Thus, going from G_1 to G_3 is essentially building an IND-security to simulation-security IPFE compiler. The random coins of this compiler are $\mathbf{t}$. Crucially, G_2 and G_3 explicitly use $f_{\mathbf{y}}(\mathbf{t})$, which is a leakage on these random coins and for this reason the IPFE compiler is used in a non-blackbox way. We refer the reader back to Remark 2.1 on why $f_{\mathbf{y}}(\mathbf{t})$ is an acceptable leakage on the IPFE simulator's random coins $\mathbf{t}$.

$\mathsf{IPFE.Gen}(1^\lambda)$
1 : $\mathsf{pp} := (\mathbb{G}, p, g) \leftarrow \mathsf{GGen}(1^\lambda)$
2 : **ret** pp

$\mathsf{IPFE.Setup}(\mathsf{pp}, 1^\ell)$
1 : $\mathbf{s} \xleftarrow{\$} \mathbb{Z}^\ell_p$, **ret** $\mathsf{msk} := \mathbf{s}$, $\mathsf{mpk} := g^{\mathbf{s}}$

$\mathsf{IPFE.Enc}(\mathsf{mpk}, \mathbf{x} \in \mathbb{Z}^\ell_p)$
1 : $r \xleftarrow{\$} \mathbb{Z}_p$, $\mathsf{ct}_0 := g^r$, $\mathsf{ct}_1 := g^{\mathbf{x}} \cdot \mathsf{mpk}^r$
2 : **ret** $\mathsf{ct} := (\mathsf{ct}_0, \mathsf{ct}_1)$

$\mathsf{IPFE.KGen}(\mathsf{msk}, \mathbf{y} \in \mathbb{Z}^\ell_p)$
1 : **ret** $\mathsf{sk}_{\mathbf{y}} := \mathbf{s}^T\mathbf{y} \in \mathbb{Z}_p$

$\mathsf{IPFE.PubKGen}(\mathsf{mpk}, \mathbf{y} \in \mathbb{Z}^\ell_p)$
1 : Parse $\mathsf{mpk} = (k_1, \ldots, k_\ell)$
2 : Parse $\mathbf{y} = (y_1, \ldots, y_\ell)$
3 : **ret** $\mathsf{pk}_{\mathbf{y}} := \prod_{i \in [\ell]} k_i^{y_i}$

$\mathsf{IPFE.Dec}(\mathsf{sk}_{\mathbf{y}}, \mathsf{ct})$
1 : Parse $\mathsf{ct}_1 = (c_1, \ldots, c_\ell)$
2 : Parse $\mathbf{y} = (y_1, \ldots, y_\ell)$
3 : $d := \prod_{i \in [\ell]} c_i^{y_i} / \mathsf{ct}_0^{\mathsf{sk}_{\mathbf{y}}}$
4 : **ret** $v := \mathsf{DLog}_g(d)$

Figure 6: Abdalla et al. [9] IPFE scheme augmented with PubKGen algorithm

6 FAS FROM PRIME-ORDER GROUPS

We provide instantiations of the FAS construction in Figure 2 from prime order groups. For this, it suffices to instantiate the building blocks IPFE and AS from prime order groups, while ensuring the two are compatible with each other. We describe these next. We can instantiate the NIZK for the language L_{NIZK} depending on the concrete relation R. That is, given R and the above instantiation of IPFE, we can pick the most efficient NIZK for L_{NIZK}. Since this is not the main contribution of this work, we will assume R to be a general NP relation and rely on the existence of NIZK for general NP [30, 34] and not delve into its details here.

More concretely, we instantiate the AS as Schnorr adaptor signature scheme [12] and thus, we have $R_{\mathsf{IPFE}} = R'_{\mathsf{IPFE}} = R_{\mathsf{DL}}$ as defined in the full version. Further, we show that the selective, IND-secure IPFE scheme by Abdalla et al. [9] satisfies R_{DL}-compliance and R_{DL}-robustness when appropriately augmented with a PubKGen algorithm. As the AS scheme is not part of our contributions, we defer its details to the full version and describe the IPFE scheme here.

IPFE from prime-order groups. We first recall the IPFE scheme by Abdalla et al. [9]. Then, we show that it satisfies the additional compliance and robustness properties needed by our FAS scheme.

Suppose p is a λ-bit prime number and suppose we want to compute inner products of vectors of length ℓ. The Abdalla et al. [9] IPFE scheme computes inner products with output values polynomially bounded by $B \ll p$. Specifically, the message space is $\mathcal{M} = \mathbb{Z}^\ell_p$ and the function class is $\mathcal{F}_{\mathsf{IP},\ell,p,B} = \{f_{\mathbf{y}} : \mathbf{y} \in \mathbb{Z}^\ell_p\} \subseteq \mathcal{F}_{\mathsf{IP},\ell}$, where $f_{\mathbf{y}} : \mathbb{Z}^\ell_p \to \{0, \ldots, B\}$ is defined as $f_{\mathbf{y}}(\mathbf{x}) = \mathbf{x}^T\mathbf{y} \in \{0, \ldots, B\}$. The scheme by Abdalla et al.[9] augmented with PubKGen algorithm is as in Figure 6. For $d \in \mathbb{G}$, we denote the algorithm for computing the discrete log of d with respect to base $g \in \mathbb{G}$ by $\mathsf{DLog}_g(d)$.

Lemma 6.1 ([9]). *Suppose the DDH assumption holds. Then, the IPFE scheme in Figure 6 is selective, IND-secure.*

Next, we show the IPFE scheme augmented with the above PubKGen algorithm satisfies R_{DL}-compliance and R_{DL}-robustness.

Lemma 6.2. *The IPFE scheme in Figure 6 is R_{DL}-compliant.*

Proof. For any $\lambda \in \mathbb{N}$, for any $\mathsf{pp} \leftarrow \mathsf{IPFE.Gen}(1^\lambda)$ as implemented in Figure 6, for any $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{IPFE.Setup}(\mathsf{pp}, 1^\ell)$ as implemented in Figure 6, for any $\mathbf{y} \in \mathcal{F}_{\mathsf{IP},\ell,p,B}$, let $\mathsf{pk}_\mathbf{y} := \mathsf{PubKGen}(\mathsf{mpk}, \mathbf{y})$ as implemented in Figure 6, and $\mathsf{sk}_\mathbf{y} := \mathsf{KGen}(\mathsf{msk}, \mathbf{y})$ as implemented in Figure 6. Suppose here $\mathsf{msk} = \mathbf{s} \in \mathbb{Z}_p^\ell$. Then, $\mathsf{sk}_\mathbf{y} = \mathbf{s}^T\mathbf{y} \in \mathbb{Z}_p$ and $\mathsf{pk}_\mathbf{y} = g^{\mathbf{s}^T\mathbf{y}} \in \mathbb{G}$. Thus, clearly, $(\mathsf{pk}_\mathbf{y}, \mathsf{sk}_\mathbf{y}) \in R_{\mathsf{DL}}$. □

Lemma 6.3. *The IPFE scheme in Figure 6 is R_{DL}-robust.*

Proof. Observe that in the IPFE scheme in Figure 6, the base relation R and the extended relation R' are the same: $R = R' = R_{\mathsf{DL}}$. Further, R_{DL} is a unique witness relation. Thus, R_{DL}-robustness follows trivially from the correctness of the IPFE scheme. □

FAS **construction.** Instantiating FAS in Section 5 with IPFE as in Figure 6 and AS as in the full version, we obtain the following informal corollary. Formal details are deferred to the full version.

Corollary 6.4 (Informal). *There exists a functional adaptor signature scheme w.r.t. Schnorr signature scheme* Sch*, NP relation R, and family of inner product functions $\mathcal{F}_{\mathsf{IP},\ell,p,B}$ that is strongly-secure.*

The proof of the corollary is immediate from that of Theorem 5.2 concerning the generic construction, and Lemmas 6.1 to 6.3 that show that the IPFE scheme in Figure 6 has the required properties.

7 FAS FROM LATTICES

In this section, we provide instantiations of the FAS construction in Figure 2 from lattices. For this, it suffices to instantiate the building blocks IPFE and AS from lattices, while ensuring the two are compatible with each other. We describe these next.

More specifically, we instantiate the AS as the lattice-based adaptor signature scheme by Esgin et al. [22]. This scheme was built using cyclotomic ring $\mathcal{R} = \mathbb{Z}[x]/(x^d + 1)$ of degree $d = 256$ under Module-SIS and Module-LWE assumptions. Here, we set the degree to be $d = 1$, thus giving us the ring of integers $\mathcal{R} = \mathbb{Z}$. Consequently, security follows from plain SIS and LWE assumptions. One can look at the resulting AS to be w.r.t. a digital scheme Lyu′ that is somewhere in between Lyubashevsky's signature scheme [32] and Dilithium [20] instantiated with unstructured lattices. Further, AS is w.r.t. hard relations R_{ISIS} and R'_{ISIS} such that $R_{\mathsf{ISIS}} \subset R'_{\mathsf{ISIS}}$.

Further, we show that the IND-secure IPFE scheme by Agrawal et al. [11, Section 4.2] satisfies R_{ISIS}-compliance and R'_{ISIS}-robustness when appropriately augmented with a PubKGen algorithm. We describe the instantiations of these two building blocks in the subsequent sections.

IPFE from lattices. We first recall the IPFE scheme by Agrawal et al. [11]. Then, we show that it satisfies the additional compliance and robustness properties needed by our FAS construction.

Suppose p is a prime number and suppose we want to compute inner products of vectors of length ℓ. Let the set of messages be $\mathcal{M} = \mathbb{Z}_p^\ell$ and the function class be $\mathcal{F}_{\mathsf{IP},\ell,p} = \{f_\mathbf{y} : \mathbf{y} \in \mathbb{Z}_p^\ell\}$, where $f_\mathbf{y} : \mathbb{Z}_p^\ell \to \mathbb{Z}_p$ is defined as $f_\mathbf{y}(\mathbf{x}) = \mathbf{x}^T\mathbf{y} \bmod p$. Our IPFE scheme augmented with PubKGen algorithm is as in Figure 7.

Remark 7.1. *We note that the original construction requires* KGen *to be stateful to support key generation queries that are linearly dependent modulo p, which is problematic since* PubKGen *would also need to be stateful, undermining the algorithm's public nature. To address this, we assume that all* KGen *queries to* IPFE *are linearly independent. While this restriction is acceptable for a single buyer, it limits support for multiple buyers. In the full version, we show how to assign unique IDs to each buyer and make the underlying* IPFE *key generation queries linearly independent.*

$\mathsf{IPFE.Gen}(1^\lambda, p)$

1: Let $n = \lambda$, integers m, k, real $\alpha \in (0,1)$ as explained below
2: Let $q = p^k$, **ret** $\mathsf{pp} = (n, m, p, q, k, \alpha)$

$\mathsf{IPFE.Setup}(\mathsf{pp}, 1^\ell)$

1: Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, $\mathbf{Z} \leftarrow \tau$ where distribution τ over $\mathbb{Z}^{\ell \times m}$ is as explained below
2: Let $\mathbf{U} := \mathbf{ZA} \in \mathbb{Z}_q^{\ell \times n}$, **ret** $\mathsf{mpk} := (\mathbf{A}, \mathbf{U})$, $\mathsf{msk} := \mathbf{Z}$

$\mathsf{IPFE.Enc}(\mathsf{mpk}, \mathbf{x} \in \mathbb{Z}_p^\ell)$

1: Sample $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}_0 \leftarrow D^m_{\mathbb{Z},\alpha q}$, $\mathbf{e}_1 \leftarrow D^\ell_{\mathbb{Z},\alpha q}$
2: Let $\mathsf{ct}_0 := \mathbf{As} + \mathbf{e}_0 \in \mathbb{Z}_q^m$, $\mathsf{ct}_1 := \mathbf{Us} + \mathbf{e}_1 + p^{k-1}\mathbf{x} \in \mathbb{Z}_q^\ell$
3: **ret** $\mathsf{ct} := (\mathsf{ct}_0, \mathsf{ct}_1)$

$\mathsf{IPFE.KGen}(\mathsf{msk}, \mathbf{y} \in \mathbb{Z}_p^\ell)$

1: **ret** $\mathsf{sk}_\mathbf{y} := \mathbf{Z}^T\mathbf{y} \in \mathbb{Z}^m$

$\mathsf{IPFE.PubKGen}(\mathsf{mpk}, \mathbf{y} \in \mathbb{Z}_p^\ell)$

1: **ret** $\mathsf{pk}_\mathbf{y} := \mathbf{U}^T\mathbf{y} \in \mathbb{Z}_q^n$

$\mathsf{IPFE.Dec}(\mathsf{sk}_\mathbf{y}, \mathsf{ct})$

1: Let $d := \mathsf{ct}_1^T\mathbf{y} - \mathsf{ct}_0^T\mathsf{sk}_\mathbf{y} \bmod q$
2: **ret** $v \in \mathbb{Z}_p$ that minimizes $| p^{k-1}v - d |$

Figure 7: Agrawal et al. [11] IPFE scheme augmented with PubKGen **algorithm. Note that** KGen **here is stateless.**

Parameter choices. Let B_τ be such that with probability at most $n^{-\omega(1)}$, each row of sample from τ has ℓ_2-norm at least B_τ. Then, the parameter constraints for correctness and security are as follows.

- $\alpha^{-1} \geq \ell^2 p^3 B_\tau \omega(\sqrt{\log n})$, $q \geq \alpha^{-1}\omega(\sqrt{\log n})$,
- $\tau = D^{\ell \times m/2}_{\mathbb{Z},\sigma_1} \times (D_{\mathbb{Z}^{m/2},\sigma_2,\delta_1} \times \ldots \times D_{\mathbb{Z}^{m/2},\sigma_2,\delta_\ell})$, where $\delta_i \in Z^\ell$ denotes the i-th canonical vector, and the standard deviation parameters satisfy $\sigma_1 = \Theta(\sqrt{n \log m}\max(\sqrt{m}, K'))$ and $\sigma_2 = \Theta(n^{7/2}m1/2\max(m, K'^2)\log^{5/2} m)$, with $K' = (\sqrt{\ell}p)^\ell$.

Further, R'_{ISIS}-robustness (Lemma 7.4) will require the constraint $\alpha^{-1} \geq 4p\omega(\sqrt{\log n})(\ell p + m\beta_1)$, where β_1 is as chosen below.

Lemma 7.2 ([11]). *Suppose $\ell \leq n^{O(1)}$, $m \geq 4n\log_2(q)$ and q, α, τ are as described above. Suppose* mheLWE$_{q,\alpha,m,\ell,\tau}$ *assumption holds. Then, the IPFE scheme in Figure 7 is selective, IND-secure.*

Next, we show the IPFE scheme augmented with the above PubKGen algorithm satisfies R_{ISIS}-compliance and R'_{ISIS}-robustness, where $R_{\mathsf{ISIS}} = \mathsf{ISIS}_{n,m,q,\beta_0}$ and $R'_{\mathsf{ISIS}} = \mathsf{ISIS}_{n,m,q,\beta_1}$, where $\beta_0 = \ell p B_\tau$ and $\beta_1 = 2(\gamma - \kappa) - \beta_0$ s.t. $\beta_1 > \beta_0$ and $\gamma - \kappa - \beta_0 > 0$.

Lemma 7.3. *The IPFE scheme in Figure 7 is R_{ISIS}-compliant, where $R_{\mathsf{ISIS}} = \mathsf{ISIS}_{n,m,q,\beta_0}$ and $\beta_0 = \ell p B_\tau$.*

Proof. For any $\lambda \in \mathbb{N}$, for any $\mathsf{pp} \leftarrow \mathsf{IPFE.Gen}(1^\lambda)$ as implemented in Figure 7, for any $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{IPFE.Setup}(\mathsf{pp}, 1^\ell)$ as implemented in Figure 7, for any $\mathbf{y} \in \mathcal{F}_{\mathsf{IP},\ell,p}$, let $\mathsf{pk}_\mathbf{y} := \mathsf{PubKGen}($

$\mathsf{mpk}, \mathbf{y})$ as implemented in Figure 7, and $\mathsf{sk}_\mathbf{y} := \mathsf{KGen}(\mathsf{msk}, \mathbf{y})$ as implemented in Figure 7. Suppose here $\mathsf{mpk} = (\mathbf{A}, \mathbf{U})$ and $\mathsf{msk} = \mathbf{Z}$. Then, $\mathsf{sk}_\mathbf{y} = \mathbf{Z}^T\mathbf{y} \in \mathbb{Z}^m$ and $\mathsf{pk}_\mathbf{y} = \mathbf{U}^T\mathbf{y} \bmod q = \mathbf{A}^T\mathsf{sk}_\mathbf{y} \bmod q$. Further, note that $||\mathsf{sk}_\mathbf{y}||_\infty \le \ell \cdot ||\mathbf{Z}||_\infty \cdot ||\mathbf{y}||_\infty \le \ell \cdot ||\mathbf{Z}||_2 \cdot ||\mathbf{y}||_\infty \le \ell \cdot B_\tau \cdot p = \beta_0$. Hence, $(\mathsf{pk}_\mathbf{y}, \mathsf{sk}_\mathbf{y}) \in R_{\mathsf{ISIS}}$. □

Lemma 7.4. *If $\alpha^{-1} \ge 4p\omega(\sqrt{\log n})(\ell p + m\beta_1)$, then, the IPFE scheme in Figure 7 is R'_{ISIS}-robust, where $R'_{\mathsf{ISIS}} = \mathsf{ISIS}_{n,m,q,\beta_1}$ and $\beta_1 = 2(\gamma - \kappa) - \beta_0$ such that $\beta_1 > \beta_0$.*

Proof. For any $\lambda \in \mathbb{N}$ let $\mathsf{pp} \leftarrow \mathsf{IPFE.Gen}(1^\lambda)$ as implemented in Figure 7, for any $\ell \in \mathbb{N}$, $\mathbf{x} \in \mathbb{Z}_p^\ell$, $\mathbf{y} \in \mathcal{F}_{\mathsf{IP},\ell,p}$, let $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(\mathsf{pp}, \ell)$ as implemented in Figure 7, let $\mathsf{pk}_\mathbf{y} = \mathsf{PubKGen}(\mathsf{msk}, \mathbf{y})$ as implemented in Figure 7, let $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathbf{x})$ as implemented in Figure 7. Then, for any $\mathsf{sk}'_\mathbf{y}$ such that $(\mathsf{pk}_\mathbf{y}, \mathsf{sk}'_\mathbf{y}) \in R'_{\mathsf{ISIS}}$, we want that $\mathsf{Dec}(\mathsf{sk}'_\mathbf{y}, \mathsf{ct})$ outputs $\mathbf{x}^T\mathbf{y} \bmod q$. Observe that the value d computed by Dec implemented in Figure 7 simplifies to $d = p^{k-1}\mathbf{x}^T\mathbf{y} + (\mathbf{e}_1^T\mathbf{y} - \mathbf{e}_0^T\mathsf{sk}'_\mathbf{y}) \bmod q$. Observe that

$$|\mathbf{e}_1^T\mathbf{y} - \mathbf{e}_0^T\mathsf{sk}_\mathbf{y}| \le \ell p\alpha q\omega(\sqrt{\log n}) + m\alpha q\omega(\sqrt{\log n})||\mathsf{sk}'_\mathbf{y}||_\infty$$
$$= \alpha q\omega(\sqrt{\log n})(\ell p + m\beta_1).$$

For decryption correctness to hold, we need that $|\mathbf{e}_1^T\mathbf{y} - \mathbf{e}_0^T\mathsf{sk}_\mathbf{y}| \le q/4p$. Hence it suffices to have $\alpha^{-1} \ge 4p\omega(\sqrt{\log n})(\ell p + m\beta_1)$. □

FAS **construction.** Instantiating FAS in Section 5 with IPFE as in Figure 7 and AS as in the full version, we obtain the following informal corollary. Formal details are deferred to the full version.

Corollary 7.5 (Informal). *There exists a functional adaptor signature scheme w.r.t. Lyubashevsky signature scheme* Lyu′*, NP relation R, and family of inner product functions $\mathcal{F}_{\mathsf{IP},\ell,p}$ that is strongly-secure.*

The proof of corollary is immediate from that of Theorem 5.2 concerning the generic construction, and Lemmas 7.2 to 7.4 that show that the IPFE scheme in Figure 7 has the required properties.

8 PERFORMANCE EVALUATION

Our implementation aims to (i) show that FAS can replace smart contracts for efficient functional sales, and (ii) benchmark the computational costs for each functional sale. We provide an open-source implementation [6] of our prime-order group-based strongly-secure FAS in Python. We also perform a series of benchmarks on our implementation for a wide range of parameters. Our results show that our scheme is practical for variety of real-world scenarios.

We measured the costs on a Apple MacBook Pro with M2 chip, 16GB memory, 8 cores (4 cores @3.49 GHz and 4 cores @2.42 GHz). In typical applications, a seller wants to sell multiple functions of the same witness. The (application dependent) advertisement – containing NIZK proof – is just a one-time cost, while other processes are done once per sale. So, we benchmark computation costs of all algorithms except AdGen and AdVerify.

We use the Secp256k1 elliptic curve that is used by Bitcoin [3] and other cryptocurrencies. We use the curve's python implementation from hanabi1224 [8]. For Schnorr signatures, we use a modified version of BIP-340 reference implementation [2]. We implement the Schnorr adaptor signatures [12] on top of it. For compliance purpose, we implement the IPFE scheme in Figure 6 on the Secp256k1 curve. Finally, using these adaptor signatures and IPFE schemes, we implement our functional adaptor signature construction. We do not implement the NIZKs as they are needed only for AdGen and AdVerify which we do not implement here.

Table 1: Communication efficiency of FAS for $\ell = 10^6$. Last column denotes if a secure channel for buyer/seller is used.

Variable	Size	On-chain?	Secure channel?
advt = (IPFE.mpk, IPFE.ct)	128 MB	no[7]	no
Function $\mathbf{y}$	32 MB	no	yes
$(\mathsf{aux}_\mathbf{y}, \pi_\mathbf{y})$	96 bytes	no	yes
Pre-signature	64 bytes	no	yes
Adapted-signature	64 bytes	yes	no

Table 2: Computational efficiency of FAS.

Params		Running Times (seconds) [8]					
ℓ	B	AuxG	AuxV	FPS	FPV	Adapt	FExt
1	10^6	0.003	0.003	0.008	0.008	0.013	0.082
10^2	10^6	0.003	0.300	0.013	0.418	0.021	0.142
10^2	10^8	0.003	0.332	0.014	0.477	0.023	0.722
10^4	10^8	0.011	0.323	0.010	0.424	0.035	1.025
10^4	10^{10}	0.011	0.369	0.009	0.352	0.022	9.952
10^5	10^{11}	0.092	2.067	0.009	2.101	0.111	30.38
10^6	10^{12}	0.879	19.08	0.008	18.41	0.871	95.07
10^7	10^{13}	9.191	223.4	0.011	217.9	10.22	317.8
$3{\cdot}10^7$	$3{\cdot}10^{13}$	32.03	766.6	0.011	740.4	40.27	452.2

Optimizations. We make following implementation optimizations.

- For vectors of length ℓ, we parallelize IPFE.Setup, IPFE.Enc.
- Recall that IPFE.Dec computes $\mathsf{ct}_1^T\mathbf{y} - \mathsf{ct}_0^T\mathsf{sk}_\mathbf{y}$ (See Figure 6). Here, ct_1 and $\mathbf{y}$ are vectors of length ℓ and $\mathsf{ct}_1^T\mathbf{y}$ is computed as $\prod_{i\in\ell} c_i^{y_i}$, where $\mathsf{ct}_1 = (c_1, \ldots, c_\ell)$ is a vector of group elements and $\mathbf{y} = (y_1, \ldots, y_\ell)$ is a vector of scalars. Thus, computing $\mathsf{ct}_1^T\mathbf{y}$ is expensive as it involves ℓ group exponentiation operations. But, this computation can be done in an offline stage by the buyer as it does not require knowledge of $\mathsf{sk}_\mathbf{y}$ from the seller. This helps us make IPFE.Dec and thus, FAS.FExt very efficient in practice.
- Note that IPFE.PubKGen and IPFE.Dec involve computing a product of powers such as $\prod_{i\in\ell} k_i^{y_i}$ and $\prod_{i\in\ell} c_i^{y_i}$ respectively. We compute these via FastMult algorithm of [14, Section 3.2].

Communication efficiency. In our implementation, each group element is 64 bytes elliptic curve point and each Z_p element is 32 bytes. In Table 1, we give bounds on communication cost for a witness x of dimension $\ell = 10^6$ (total size 32 MB) and also specify if the communication is done on- or off-chain.

Using standard compression techniques outlined in BIP-340 [2], advertisement size can be reduced to 66 MB by encoding 64 bytes elliptic curve points using 33 bytes. In practice, the size of advt = (IPFE.mpk, IPFE.ct) can be amortized to 33 MB by reusing the same

[7] for example, advt can be published on a website.
[7] AuxG, AuxV, FPS, FPV denote AuxGen, AuxVerify, FPreSign, FPreVerify.
[8] AuxG, AuxV, FPS, FPV denote AuxGen, AuxVerify, FPreSign, FPreVerify.

IPFE.mpk for all advertisements by a seller. The multi-message security of IPFE scheme should preserve FAS security and result in optimal amortized advertisement size of 1.03x the witness size.

Computational efficiency. We share the performance numbers in Table 2. Here, ℓ denotes the length of the witness vector. Each entry of the vector is 32 Bytes integer. Thus, $\ell = 10^6$ implies that the witness is of size 32MB. We note that the pre-dominant cost in the implementation is the group exponentiation operation which hasn't been optimized here. We note that for parameter ℓ, the number of group exponentiation operations in each algorithm are as follows: in AuxGen, $\ell+1$ in AuxVerify, 1 in FPreSign, $\ell+2$ in FPreVerify, 0 in Adapt, $\ell+2$ in FExt. Asymptotically, AuxVerify and FPreVerify run in time $O(\ell)$ and FExt runs in time $O(\sqrt{\ell})$. But, concretely FExt is the slowest for practical scenarios as highlighted in Table 2. The last two rows of Table 2 do not highlight practical scenarios as the costs of these algorithms are higher than 100 seconds each, but we benchmark them to understand at what point do the concrete running times of AuxVerify and FPreVerify start becoming a bottleneck.

ACKNOWLEDGEMENTS

This work was supported in part through a gift awarded to Pratik Soni and Sri AravindaKrishnan Thyagarajan under the Stellar Development Foundation Academic Research Grants program. This work was also sponsored by the Algorand Foundation, NSF grant under award number 2044679, and ONR grant under award number N000142212064. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

REFERENCES

[1] [n. d.]. https://www.kaggle.com/datasets/erdemtaha/cancer-data.
[2] [n. d.]. BIP 340: Schnorr Signatures for secp256k1. https://github.com/bitcoin/bips/blob/master/bip-0340/reference.py.
[3] [n. d.]. Bitcoin Wiki: Secp256k1. https://en.bitcoin.it/wiki/Secp256k1.
[4] [n. d.]. Blockchain Bridges. https://ethereum.org/en/bridges/.
[5] [n. d.]. Ethereum Gas Cost. https://milkroad.com/ethereum/gas/.
[6] [n. d.]. Functional Adaptor Signatures: Implementation. https://github.com/nikhilvanjani/fas-impl.
[7] [n. d.]. Oracle Network For Smart Contracts. https://shorturl.at/fpDEZ.
[8] [n. d.]. Secp256k1 Python. https://github.com/hanabi1224/Programming-Language-Benchmarks/blob/main/bench/algorithm/secp256k1/1.py.
[9] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. 2015. Simple functional encryption schemes for inner products. In *PKC*.
[10] Shweta Agrawal, Benoît Libert, Monosij Maitra, and Radu Titiu. 2020. Adaptive simulation security for inner product functional encryption. In *IACR International Conference on Public-Key Cryptography*. Springer, 34–64.
[11] Shweta Agrawal, Benoît Libert, and Damien Stehlé. 2016. Fully Secure Functional Encryption for Inner Products, from Standard Assumptions. In *CRYPTO*.
[12] Lukas Aumayr, Oguzhan Ersoy, Andreas Erwig, Sebastian Faust, Kristina Hostáková, Matteo Maffei, Pedro Moreno-Sanchez, and Siavash Riahi. 2021. Generalized channels from limited blockchain scripts and adaptor signatures. In *Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security*.
[13] Konstantin Avrachenkov, Pavel Chebotarev, and Dmytro Rubanov. 2017. Kernels on graphs as proximity measures. In *Algorithms and Models for the Web Graph: 14th International Workshop, WAW 2017*.
[14] Mihir Bellare, Juan A Garay, and Tal Rabin. 1998. Fast batch verification for modular exponentiation and digital signatures. In *Advances in Cryptology—EUROCRYPT'98: International Conference on the Theory and Application of Cryptographic Techniques*.
[15] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. 2013. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing (STOC '13)*. Association for Computing Machinery.
[16] Dan Boneh, Amit Sahai, and Brent Waters. 2011. Functional Encryption: Definitions and Challenges. In *TCC*.
[17] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N Rothblum, Ron D Rothblum, and Daniel Wichs. 2019. Fiat-Shamir: from practice to theory. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*.
[18] Ran Canetti, Shai Halevi, and Jonathan Katz. 2003. A forward-secure public-key encryption scheme. In *Advances in Cryptology—EUROCRYPT 2003: International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4–8, 2003 Proceedings 22*. Springer, 255–271.
[19] Wei Dai, Tatsuaki Okamoto, and Go Yamamoto. 2022. Stronger security and generic constructions for adaptor signatures. In *International Conference on Cryptology in India*. Springer, 52–77.
[20] Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. 2018. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems* (2018), 238–268.
[21] Andreas Erwig, Sebastian Faust, Kristina Hostáková, Monosij Maitra, and Siavash Riahi. 2021. Two-party adaptor signatures from identification schemes. In *IACR International Conference on Public-Key Cryptography*. Springer, 451–480.
[22] Muhammed F Esgin, Oğuzhan Ersoy, and Zekeriya Erkin. 2020. Post-quantum adaptor signatures and payment channel networks. In *European Symposium on Research in Computer Security*. Springer, 378–397.
[23] Muhammed F. Esgin, Oğuzhan Ersoy, and Zekeriya Erkin. 2020. Post-Quantum Adaptor Signatures and Payment Channel Networks. In *Computer Security – ESORICS 2020*, Liqun Chen, Ninghui Li, Kaitai Liang, and Steve Schneider (Eds.). Springer International Publishing, Cham, 378–397.
[24] Uriel Feige, Dror Lapidot, and Adi Shamir. 1990. Multiple non-interactive zero knowledge proofs based on a single random string. In *Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science*. IEEE, 308–317.
[25] Paul Gerhart, Dominique Schröder, Pratik Soni, and Sri AravindaKrishnan Thyagarajan. 2024. Foundations of Adaptor Signatures. In *Advances in Cryptology – EUROCRYPT 2024*, Marc Joye and Gregor Leander (Eds.).
[26] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. 1989. The Knowledge Complexity of Interactive Proof Systems. *SIAM J. Comput.* 18, 1 (1989), 186–208. https://doi.org/10.1137/0218012
[27] Shafi Goldwasser, Silvio Micali, and Ronald L Rivest. 1988. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on computing* 17, 2 (1988), 281–308.
[28] Jens Groth, Rafail Ostrovsky, and Amit Sahai. 2006. Non-interactive zaps and new techniques for NIZK. In *Advances in Cryptology-CRYPTO 2006: 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006. Proceedings 26*. Springer, 97–111.
[29] Lucjan Hanzlik, Julian Loss, Sri AravindaKrishnan Thyagarajan, and Benedikt Wagner. 2024. Sweep-UC: Swapping coins privately. In *2024 IEEE Symposium on Security and Privacy (SP)*.
[30] Abhishek Jain and Zhengzhong Jin. 2021. Non-interactive zero knowledge from sub-exponential DDH. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 3–32.
[31] Mohammad Monirujjaman Khan, Nesat Tasneem RoJa, Faris A Almalki, Maha Aljohani, et al. 2022. Revolutionizing E-Commerce Using Blockchain Technology and Implementing Smart Contract. *Security and Communication Networks* (2022).
[32] Vadim Lyubashevsky. 2012. Lattice signatures without trapdoors. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 738–755.
[33] Giulio Malavolta, Pedro Moreno-Sanchez, Clara Schneidewind, Aniket Kate, and Matteo Maffei. 2019. Anonymous Multi-Hop Locks for Blockchain Scalability and Interoperability. In *Proceedings 2019 Network and Distributed System Security Symposium*. Internet Society.
[34] Chris Peikert and Sina Shiehian. 2019. Noninteractive zero knowledge for NP from (plain) learning with errors. In *Annual International Cryptology Conference*.
[35] Joseph Poon and Thaddeus Dryja. [n. d.]. The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments.
[36] Xianrui Qin, Shimin Pan, Arash Mirzaei, Zhimei Sui, Oğuzhan Ersoy, Amin Sakzad, Muhammed F Esgin, Joseph K Liu, Jiangshan Yu, and Tsz Hon Yuen. 2023. Blindhub: Bitcoin-compatible privacy-preserving payment channel hubs supporting variable amounts. In *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2462–2480.
[37] Sri AravindaKrishnan Thyagarajan, Giulio Malavolta, and Pedro Moreno-Sanchez. 2022. Universal atomic swaps: Secure exchange of coins across all blockchains. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1299–1316.
[38] Sri Aravinda Krishnan Thyagarajan and Giulio Malavolta. 2021. Lockable signatures for blockchains: Scriptless scripts for all signatures. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 937–954.
[39] Gang Wang and Mark Nixon. 2022. SoK: tokenization on blockchain. In *Proceedings of the 14th IEEE/ACM International Conference on Utility and Cloud Computing Companion* (Leicester, United Kingdom) *(UCC '21)*. Association for Computing Machinery, New York, NY, USA.
[40] Hoeteck Wee. 2016. New techniques for attribute-hiding in prime-order bilinear groups. Manuscript.