Bicameral and Auditably Private Signatures

Khoa Nguyen [1], Partha Sarathi Roy [1], Willy Susilo [1], and Yanhong Xu [2]✉

[1] Institute of Cybersecurity and Cryptology,
School of Computing and Information Technology,
University of Wollongong, Northfields Avenue, Wollongong, NSW 2522, Australia
{khoa,partha,wsusilo}@uow.edu.au
[2] School of Electronic Information and Electrical Engineering,
Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai 200240, China
✉ yanhong.xu@sjtu.edu.cn

Abstract. This paper introduces Bicameral and Auditably Private Signatures (BAPS) – a new privacy-preserving signature system with several novel features. In a BAPS system, given a certified attribute $\mathbf{x}$ and a certified policy P, a signer can issue a publicly verifiable signature Σ on a message m as long as $(m, \mathbf{x})$ satisfies P. A noteworthy characteristic of BAPS is that both attribute $\mathbf{x}$ and policy P are kept hidden from the verifier, yet the latter is convinced that these objects were certified by an attribute-issuing authority and a policy-issuing authority, respectively. By considering bicameral certification authorities and requiring privacy for both attributes and policies, BAPS generalizes the spirit of existing advanced signature primitives with fine-grained controls on signing capabilities (e.g., attribute-based signatures, predicate signatures, policy-based signatures). Furthermore, BAPS provides an appealing feature named auditable privacy, allowing the signer of Σ to verifiably disclose various pieces of partial information about P and $\mathbf{x}$ when asked by auditor(s)/court(s) at later times. Auditable privacy is intrinsically different from and can be complementary to the notion of accountable privacy traditionally incorporated in traceable anonymous systems such as group signatures. Equipped with these distinguished features, BAPS can potentially address interesting application scenarios for which existing primitives do not offer a direct solution.
We provide rigorous security definitions for BAPS, following a "sim-ext" approach. We then demonstrate a generic construction based on commonly used cryptographic building blocks, which employs a sign-then-commit-then-prove design. Finally, we present a concrete instantiation of BAPS, that is proven secure in the random oracle model under lattice assumptions. The scheme can handle arbitrary policies represented by polynomial-size Boolean circuits and can address quadratic disclosing functions. In the construction process, we develop a new technical building block that could be of independent interest: a zero-knowledge argument system allowing to prove the satisfiability of a certified-and-hidden Boolean circuit on certified-and-committed inputs.

Keywords: new primitive, signatures, bicamerality, auditable privacy, fine-grained information disclosure, zero-knowledge for hidden circuits

1 Introduction

A prominent line of privacy-preserving cryptography research is dedicated to the development of advanced multi-user signature systems with fine-grained controls over the signability of messages. Those controls are often based on authorities' policies, and/or users' attributes. Examples of these advanced systems include attribute-based signatures (ABS) [45], policy-based signatures (PBS) [2], functional signatures (FS) [9], predicate signatures (PS) [1] and multimodal private signatures (MPS) [48]. In ABS, a user with a certified-and-private attribute $\mathbf{x}$ can sign *any* message with respect to a *public* policy P if $P(\mathbf{x}) = 1$. PS offers a setting dual to ABS, where policies are certified-and-private while attributes are public. In PBS and FS: (i) policies/functions are also certified-and-private; (ii) one can sign messages satisfying some policy or in the range of some function; yet (iii) the notion of attributes is not considered. MPS utilizes both policies and attributes, but policies have to be public. All these systems share a common feature: each of them employs a single certification authority: either an attribute-issuing authority as in ABS/MPS, or a policy-issuing authority as in PBS/FS/PS. Furthermore, the question of simultaneously protecting the privacy of both policies and attributes was not considered.

Another active body of work in privacy-preserving signatures focuses on developing methods for realizing signers' accountability. Let us name that desirable feature accountable privacy. Among the earliest and most well-known accountably private systems are group signatures [13], in which a designated authority can trace the signer of any valid signature. Subsequent works have refined the tracing function in various directions: "who can trace" [31,52], "whether to trace" [54,32], "when to trace" [11,22], and more recently, "what can be traced" [39,48]. Nevertheless, all these systems share a common characteristic: the signer has no control over which private information can be learned by others (either the public or the tracing authorities) *after* outputting signatures.

Motivations. This work aims to address the limitations of the advanced signature primitives mentioned above. Let us start with several motivating examples.

Consider a conference that implements a double-blind reviewing process and that allows authors to declare Conflicts of Interest (CoI) with reviewers according to some certified policies. For instance, the IACR has different policies to determine CoI [3], that can be used for IACR conferences. While such a CoI declaration system seems to work well over the years, there has not been implemented any privacy-preserving mechanism:

(i) For preventing false declarations by dishonest authors (who could attempt to avoid having their papers reviewed by some non-conflicting reviewers);
(ii) For the author to provide more information on a declared CoI, should the need arise at a later point while retaining the author's privacy.

Let us attempt to address the issue (i) by employing ABS/MPS for CoI declarations. Then, while authors' attributes (e.g., lists of advisors, advisees,

[3] `https://www.iacr.org/docs/conflicts.pdf`.

recent affiliations, recent co-authors, and family members) are protected, the underlying policies are not. It could seriously violate authors' privacy, e.g., the disclosure of an advisor-advisee relationship could likely reveal side-channel information about the author. On the other hand, if one employs PBS/FS/PS, then the policies can be kept private, but the attributes are not protected.

Let us also try to address the issue (ii) by employing a group-signature-like system. In this case, the traditional method is to force the author to encrypt the relevant attributes, so that an authority can recover via decryption. The problem here is the encrypted attributes can already be learned by the authority at the submission time, regardless of whether there will be a need to justify the declared CoIs during the reviewing process.

We can observe that existing systems cannot offer satisfactory solutions because they do not simultaneously protect policies and attributes, and the notion of accountable privacy typically requires some escrow of private information. This inspires us to investigate a new privacy-preserving signature primitive that overcomes these limitations. How about the following arrangements?

Assume that the policies are publicly certified by some authority (e.g., the IACR). Assume further that authors have their attributes certified by some other authority. Then, when authors submit a paper, they can check whether the paper has a CoI with a given reviewer and generate a publicly verifiable signature if that is the case. Here, if the signature does not reveal any information about the underlying policy and attribute, then we have a solution for issue (i).

Next, suppose that the PC Chair wants to know some information about the circumstances of a given CoI. Depending on the context, the required information could be about which policy was activated, the list of advisees, or a recent affiliation of the author. Here, a mechanism allowing the disclosure of the exact piece of information requested by the Chair would help resolve the issue (ii).

Let us consider a further example. Suppose an author would like to apply for a visa so that they can travel to the conference. In the application process, the author submits their certified attributes (which could include financial data, criminal records, health examinations, travel records, etc.) and the associated certificates to the visa department. Suppose that the latter has several confidential policies for visa acceptance/rejection, which are certified by some higher authority. Now, assume that there is some concern about the transparency of the decision-making process. In this situation, the privacy-preserving system we have just discussed can allow the visa department to verifiably disclose to a judge certain partial information about the underlying policy and attribute, e.g., whether the policy considers the applicant's race, or the criminal records of the applicant while retaining the privacy of non-disclosed information.

More generally, many decisions regarding crucial issues, such as welfare and financial aid, employment offers, scholarship/citizenship grants, and tax audits, are taken based on organizational policies that should not be known by outsiders. In these scenarios, the policies could be certified by some authority A, while the users' attributes could be certified by some other authority B. To show that a correct decision has been made [33], the decision-makers would need a mechanism

allowing them to sign some message m and demonstrate that a private attribute $\mathbf{x}$ certified by B does satisfy a private policy P certified by A. Furthermore, for auditability purposes, it would be highly desirable if the system also allows the decision-makers to verifiably disclose the precise pieces of information about $\mathbf{x}$ and P requested by the auditors.

Our Contributions. We introduce "Bicameral and Auditably Private Signatures" (BAPS) as a new privacy-preserving signature primitive aiming to address (i) the problem of simultaneously protecting policies and attributes and (ii) the problem of secure disclosures of private information after signing. Let us first highlight several key features of BAPS.

Bicamerality. The system is "bicameral" in the sense that it involves two certification authorities, namely, attribute-issuing and policy-issuing ones, that are responsible for certifying users' attributes and organizations' policies, respectively. As discussed above, having two separate certification authorities is a commonly seen situation in practice.

More concretely, in a BAPS system, the attribute-issuing authority, given its master secret key $\mathsf{msk}_{\mathsf{X}}$, can issue a signing key $\mathsf{sk}_{\mathbf{x}}$ for attribute $\mathbf{x}$. Similarly, the policy-issuing authority can use its master secret key $\mathsf{msk}_{\mathsf{P}}$ to generate a certificate Cert_P for policy P. Note that, although the certification procedures of $\mathbf{x}$ and P are analogous, the treatments of "signing key" $\mathsf{sk}_{\mathbf{x}}$ and "certificate" Cert_P could be largely different in practice. The former typically should be kept private, unless its owner would like to have some other party to sign on their behalf. The latter normally can be made publicly available, e.g., in a list $\{(P_i, \mathsf{Cert}_{P_i})\}_i$ on an organization's website, except when the policies of the organization must be kept confidential. In our example about CoI declarations, we consider private $(\mathbf{x}, \mathsf{sk}_{\mathbf{x}})$ and public $\{(P_i, \mathsf{Cert}_{P_i})\}_i$. Meanwhile, the example with visa applications assumes that the applicant submits their credentials $(\mathbf{x}, \mathsf{sk}_{\mathbf{x}})$ to the visa department – which keeps their policies private. In any application scenario, the signer needs to know both pairs $(\mathbf{x}, \mathsf{sk}_{\mathbf{x}})$ and (P, Cert_P).

Signability and Privacy. To sign a message m, in addition to the pairs $(\mathbf{x}, \mathsf{sk}_{\mathbf{x}})$ and (P, Cert_P), the signer needs to possess a witness w such that $P(m, \mathbf{x}, w) = 1$. Here, similar to [2,39,48], witness w can be viewed as a piece of context-dependent information which intuitively serves as evidence why m is signable with respect to $\mathbf{x}$ and P. Note that our notion of "signability" is more general and does capture that of attribute-based signatures [45] (where P takes only attribute $\mathbf{x}$ as its sole input) and policy-based signatures [2,14] (where users' attributes are not considered in the syntax and P depends on (m, w) only).

If signing is successful, the signer obtains a signature Σ that is publicly verifiable. In addition, the signer stores a private clue c associated with (m, Σ), which later can be utilized for auditing purposes – should the need arise.

We demand a strong privacy property for BAPS: a valid message-signature pair (m, Σ) must not leak any information about the underlying policy P and attribute $\mathbf{x}$ (apart from the fact that $P(m, \mathbf{x}, w) = 1$ for some w). This should hold even if both authorities are fully corrupted.

Disclosures and Auditable Privacy. When asked to disclose certain partial information of P and/or $\mathbf{x}$ according to a disclosing function F (which is chosen by a court or an auditor among a public list of admitted functions), the signer of a pair (m, Σ) uses the associated clue c to generate a publicly verifiable testimony-attestation pair (t, a). Here, the attestation intuitively demonstrates that the value of the testimony is exactly determined as $t = F(P, x)$, where $(P, \mathbf{x})$ is precisely the policy-attribute pair underlying the message-signature pair (m, Σ). Such a disclosure process can be done multiple times with respect to different disclosing functions.

Here, we demand a noteworthy property for BAPS: auditable privacy. It says that no additional information about P or $\mathbf{x}$ can be learned from the pair (t, a) beyond the fact that $F(P, \mathbf{x}) = t$. In other words, it guarantees the "residual" privacy of P and $\mathbf{x}$ after (potentially many) disclosures of their partial information. It should hold against corrupted authorities and can even be defined in the strong, statistical sense.

As a short summary, by considering bicameral authorities and requiring privacy for both attributes and policies, BAPS generalizes the spirit of existing advanced signature primitives with fine-grained controls on signing capabilities. Moreover, the property of auditable privacy is sharply different from and can be complementary to the notion of accountable privacy in group signatures [13] and variants [31,54,52,32,39,48], which demands that each signature contains a fixed piece of signer's information that can be recovered by a designated party. Equipped with these distinguished features, BAPS can potentially address application scenarios for which existing primitives do not offer a direct solution.

Formalizing Security Requirements for BAPS. Let us next discuss our formalizations of security for BAPS, which is a non-trivial process on its own.

We first tried to define privacy and auditable privacy for BAPS using an indistinguishability-based approach. However, with that approach, we could not manage to quantify the amount of information leaked by the disclosure processes. In fact, we need to ensure that no extra information about P and $\mathbf{x}$ is leaked, apart from the pieces of information carried by the testimonies outputted by the signer. We then observe that this requirement is quite similar to those in the contexts of zero-knowledge proofs [23], and functional encryption [6], where a simulation-based approach has been employed and proven successful. We thus adopt such an approach, which allows us to formalize privacy and auditable privacy via a unified notion of simulatability. It essentially says that the setup, signing, and disclosing processes of BAPS can be efficiently simulated in a way (statistically) indistinguishable from the real algorithms.

We next aim at formalizing other expected security properties of BAPS. On the one hand, we would like to ensure that, even if both authorities are corrupted, any valid signature Σ on any message m should be associated with some policy P, some attribute $\mathbf{x}$ and some witness w such that $P(m, \mathbf{x}, w) = 1$. Also, if $\mathsf{Judge}(m, \Sigma, F, t, a) = 1$, then it should hold that $t = F(P, \mathbf{x})$.

On the other hand, we also need to protect the security of each of the authorities, which can be divided into two orthogonal properties. Specifically, it should be infeasible to generate any valid signature, if

- One does not possess a legitimate signing key $\mathsf{sk}_{\mathbf{x}}$ for some attribute $\mathbf{x}$, even if one corrupted the policy-issuing authority;
- One does not possess a legitimate certificate Cert_P for some policy P, even if one corrupted the attribute-issuing authority.

The major technical challenge in formalizing these expected security properties is that BAPS does not readily provide a rigorous mechanism to determine whether a valid message-signature pair (m, Σ) is actually associated with some P, $\mathbf{x}$ and w such that $P(m, \mathbf{x}, w) = 1$. Therefore, for definitional purposes, we would need to introduce an extractable mode that allows us to extract additional information from (m, Σ) so that we can meaningfully explain whether and how a violation of security has occurred. As a result, we come up with a notion called intractability, which nicely captures and unifies the said properties. This is partially inspired by the "sim-ext" spirit [12], which was also employed in the context of PBS [2].

We would like to remark that our formalizations of BAPS yield a proper generalization of the PBS primitive [2]. Indeed, to obtain a PBS, one can simply remove from a BAPS the treatments of attributes and attribute-issuing authority, as well as the disclosure process. Simulatability and extractability of the resulting PBS, as defined by Bellare and Fuchsbauer [2], directly follow from those of the original BAPS. Recall that PBS is already an exceedingly powerful primitive on its own: it was shown to imply group signatures [3], attribute-based signatures [45], simulation-sound extractable NIZKs [24], and others.

Generic Constructions. Our next step is to demonstrate the feasibility of constructing secure BAPS systems from standard assumptions. Specifically, we present a generic construction that employs several commonly used cryptographic building blocks: two signature schemes, a commitment scheme, and two non-interactive zero-knowledge (NIZK) argument systems for some NP-relations. The correctness and security properties of the construction directly follow from those of the employed building blocks.

On the one hand, the two signature systems and the two NIZK systems are used in a relatively standard manner. The former systems are governed by the two authorities and are used for issuing attribute keys and policy certificates. The latter systems are utilized when generating and verifying BAPS signatures and attestations. On the other hand, our implementation of the commitment scheme is worth highlighting. We commit to policy P and attribute $\mathbf{x}$ as com_P and $\mathsf{com}_{\mathbf{x}}$, respectively, and treat them as *a bridge connecting the signing and disclosing phases*. The binding of the commitment scheme ensures that the pair $(P, \mathbf{x})$ involved in the disclosure(s) of a BAPS signature Σ is the same as the pair used for generating Σ. Furthermore, its hiding property guarantees that no additional information about $(P, \mathbf{x})$ can be learned from com_P and $\mathsf{com}_{\mathbf{x}}$. Note that in existing group-signature-like systems, a public-key encryption scheme is typically

used to disclose some private information of the signer to a designated opening authority. Due to the existence of a decryption key (which is owned by the opening authority, and recoverable by an unbounded adversary), the accountable privacy property in these systems is only achieved in the computational sense. Here, in contrast, auditable privacy in the statistical sense is achievable if the commitment is statistically hiding.

In more detail, in the construction, a signature Σ on message m contains the commitments com_P and $\mathsf{com}_{\mathbf{x}}$ as well as a NIZK argument Π demonstrating that the committed values $P, \mathbf{x}$ were properly certified by the authorities, and that $P(m, \mathbf{x}, w) = 1$ for some witness w known by the signer. Verification of Σ is simply the verification of Π. The clue – which will be used for later disclosures – consists of $\mathbf{x}, P$, and the two randomnesses.

When asked to disclose certain partial information of $\mathbf{x}$ and/or P according to a disclosing function F, the signer of a message-signature pair (m, Σ), who possesses the corresponding clue, first computes $t = F(P, \mathbf{x})$ and then proves that t is well-formed with respect to the values P and $\mathbf{x}$ committed in the signature.

We remark that our design approach, which we term *sign-then-commit-then-prove*, effectively differs from the sign-then-encrypt-then-prove paradigm traditionally used for achieving accountable privacy in group-signature-like systems. The advantage of our approach is that the signer can preserve full privacy of the committed values (especially if the underlying commitment scheme is statistically hiding) while maintaining the capability of proving additional relations about the committed values at later times.

Our construction can serve as a proof of concept for designing secure BAPS systems based on standard assumptions in a modular manner. In particular, it can be realized in the standard model from pairings and from lattices, using the techniques for obtaining NIZKs for NP by Groth-Ostrovsky-Sahai [25] and by Peikert-Shiehian [50] (in conjunction with a lattice-based compiler by Libert et al. [38]), respectively.

A Lattice-Based Instantiation for Arbitrary Policies. Although via techniques of [50,38] it is feasible to instantiate BAPS in the standard model under lattice-based (and hence, quantum-safe) assumptions, such construction would expectedly be extremely inefficient. Our goal here is to build a concrete lattice-based BAPS that has better efficiency than the generic approach, and that can handle expressive classes of policies, e.g., polynomial-size Boolean circuits. We stress that the lattice-based BAPS scheme we provide in this paper merely serves as an illustration of how to concretely instantiate the aforementioned generic construction. It is not practical, and moreover, we do not view it as the main contribution of the paper.

At a high level, our lattice-based BAPS scheme follows the sign-then-commit-then-prove design approach of the generic construction discussed above. However, when it comes to middle-level techniques, we do introduce several novel aspects regarding the evaluation of policies in zero-knowledge and the instantiation of disclosing functions via multivariate quadratic functions.

More specifically, for both authorities, we implement the Ducas-Micciancio signature [15], which has a short public key and has a companion zero-knowledge argument of a valid message-signature pair [42]. As for commitment schemes, we employ several adaptations of the ideal-lattice-based scheme by Kawachi, Tanaka and Xagawa [30], which are used not only to commit to P and $\mathbf{x}$, but also various different objects appearing in the scheme execution. Last but not least, we need expressive zero-knowledge argument systems that can handle relatively sophisticated statements, in particular, those that capture the satisfiability of a Boolean circuit whose description is hidden, yet certified via the Ducas-Micciancio signature. To this end, we adopt a framework proposed in [36] for interactive Stern-like argument systems [53] and employ several dedicated techniques for handling the required relations. We remark that our choice of Stern-like protocols as the zero-knowledge tools has its own advantages and disadvantages. On the upside, these tools work smoothly with the statements to be proven. First, they can be directly applied to interrelated equations involving two moduli $q_1 = 2$ (representing the evaluation processes of Boolean circuits as well as disclosing functions) and $q_2 = 3^k$ for some positive integer k (representing relations capturing the Ducas-Micciancio signature verification process). Second, Stern-like protocols normally provide statistical zero-knowledge, which is crucial in attaining the desirable feature of statistical privacy for the resulting lattice-based BAPS scheme. Therefore, these zero-knowledge protocols are quite suitable for our illustration purpose. On the downside, because of the need to repeat Stern-like protocols many times to make their soundness errors negligibly small, these tools are much less efficient than the state-of-the-art Schnorr-like zero-knowledge proof/argument systems, such as [55,8,18,16,44,17,7,27]. While we have not been able to provide an efficient lattice-based BAPS (see more elaborations in the discussions on open questions at the end of this section), we nevertheless expect that, in the near future, practically usable lattice-based BAPS systems, if any, would likely be developed based on these state-of-the-art tools.

The major technical challenge of the construction is to prove in zero-knowledge the satisfiability of a certified-and-hidden Boolean circuit P on some input $(m, \mathbf{x}, w)$, where $\mathbf{x}$ is also certified and hidden.

Evaluating Circuits in ZK with "Imaginary Buckets". To the best of our knowledge, there has not been much work related to proving circuit satisfiability where both the circuit and input are certified and private. Ling et al. [40] proposed a code-based protocol for the restricted class of symmetric Boolean functions. Libert et al. [35] suggested a lattice-based protocol for functions in $\mathsf{NC1}$, represented by branching programs. In Libert et al.'s protocol, the prover commits to the inputs, builds a Merkle hash tree on top of the commitments, and fetches the inputs to the branching program by following the tree paths corresponding to the program's binary representation. We observe that the ideas from [35] might potentially be extended to handle Boolean circuits. However, the expected complexity of each tree-based retrieval step would be

$\mathcal{O}(\log(K+N)\cdot\lambda\cdot\log\lambda)$, where λ is the security parameter, K is the bit-size of the inputs and N is the circuit size.

We, therefore, take a conceptually different approach with expected complexity $\mathcal{O}(\sqrt{K+N}+\lambda\cdot\log\lambda)$. This would yield some valuable improvement when $K+N$ is a small polynomial in λ, e.g., $K+N=o(\lambda^3)$.

Consider a circuit P whose topology is determined by two functions $g,h:[0,N-1]\to[0,K+N-2]$. Namely, if the inputs to P are $s_0,s_1,\ldots,s_{K-1}$ and the N gate outputs are ordered as $s_K,\ldots,s_{K+N-1}$, then we have

$$s_{K+i}=s_{g(i)}\ \mathsf{NAND}\ s_{h(i)},\ \forall i=0,\ldots,N-1.$$

At each step in the circuit evaluation process, we need to fetch the values of $s_{g(i)}$ and $s_{h(i)}$. The problem is that not only these values but also both the indices $g(i)$ and $h(i)$ are committed. Hence, we would need a mechanism to retrieve these values properly. Our approach employs a "bucket-based" retrieval process, the high-level ideas of which are as follows.

We divide the bits of $s_0,\ldots,s_{K+N-2}$ into ρ buckets $\mathbf{s}_0,\ldots,\mathbf{s}_{\rho-1}$, each of which is ρ-bit long. (Here, for simplicity, we assume $K+N-1=\rho^2$.) Next, we commit to each bucket, obtaining ρ commitments: $\mathsf{com}_0,\ldots,\mathsf{com}_{\rho-1}$. To fetch the correct inputs to the gates, we will examine the binary representations of g,h, and follow them to search for the buckets where the correct inputs are committed, then identify the exact locations of the inputs within the found buckets. Such a process only requires $\mathcal{O}(\rho)$ steps, and thus, yields complexity $\mathcal{O}(\sqrt{K+N})$. We provide more detailed explanations in Section 4.

Handling quadratic disclosing functions. For disclosures, we consider multivariate quadratic functions. In this setting, the testimony $t=F(P,\mathbf{x})$ is a multidimensional vector, each coordinate of which has the form

$$\sum_{i,j\in[1,\bar{k}]}\alpha_{i,j}\cdot(b_i\cdot b_j)+\sum_{\ell\in[1,\bar{k}]}\beta_\ell\cdot b_\ell \bmod 2,$$

where $\alpha_{i,j}$'s, β_ℓ's are public bits and b_i's the bits determining policy P and attribute $\mathbf{x}$. This definition is thus quite general and captures arbitrary linear and quadratic relations with respect to the bits of $(P,\mathbf{x})$. By setting the coefficients appropriately, one indeed can enforce the disclosures of any bits of $(P,\mathbf{x})$, or any bit-products, or any linear and/or quadratic combinations of the bits.

To prove the well-formedness of $t=F(P,\mathbf{x})$, we will need to demonstrate in zero-knowledge the correct evaluations of many equations of the above form. This sub-task also requires several non-trivial steps, since the bits b_i's are involved not only in these linear and quadratic relations, but they also simultaneously satisfy various other relations, e.g., they were hashed, signed, and committed.

Open Questions. As the first work that introduces BAPS, we do not (and cannot) attempt to address all the issues around this new primitive. We pay more attention to laying the foundations for BAPS, and we view the problem of constructing a scheme with practical efficiency and/or with additional features as fascinating open questions for future investigations. In the following, we will briefly discuss several questions that we are particularly interested in.

– **Developing practically usable lattice-based BAPS schemes.** While the state-of-the-art zero-knowledge proof/argument systems from lattice assumptions and for lattice relations, such as [18,16,44,17,7,27], have already come close to practicality, the problem of applying these protocols in an efficiency-preserving manner to mixed relations like "correct evaluation of a certified-and-hidden Boolean circuit on a certified-and-hidden input" (which are helpful for developing BAPS) is still less well-studied. Additionally, these proof/argument systems typically only satisfy computational zero-knowledge, which could be an obstacle if one insists on achieving statistically private BAPS. Nevertheless, we do hope that some reasonably practical lattice-based BAPS schemes can be developed based on these tools in the near future, especially if one only targets computational privacy and would like to consider only some restricted classes of policies, e.g., those defined by inner-product-like relations.
– **Designing efficient BAPS without NIZK.** As we have discussed, a major barrier to constructing efficient BAPS systems is the need for NIZK systems that can handle relatively sophisticated statements. While the use of NIZKs seems unavoidable, as a BAPS satisfying our stringent security definitions does imply a PBS [2] – which in turn implies NIZK, it might be possible to circumvent this barrier by considering some relaxed security requirements for BAPS. A similar line of research was conducted with respect to group signatures [28].
– **BAPS with additional functionalities.** One interesting question along this line of research is to enable efficient user revocations, preventing revoked users from generating valid signatures while ensuring small computation/communication overheads for non-revoked users. Another appealing question is to conceptualize and realize systems that simultaneously offer both accountable privacy and auditable privacy.

Organization. The rest of the paper is organized as follows. In Section 2, we present our definitions of BAPS, describe its syntax, and formalize its security requirements. Section 3 then provides a generic construction of BAPS satisfying our model, based on commonly used cryptographic building blocks. In Section 4, we describe our lattice-based construction of BAPS and the supporting techniques for evaluating hidden-and-certified Boolean circuits in zero-knowledge. The reminders on the cryptographic building blocks employed in our constructions, the detailed descriptions of the zero-knowledge protocols used in our lattice-based BAPS scheme, and most of the security analyses are deferred to the appendix.

2 Bicameral and Auditably Private Signatures

Any Bicameral and Auditably Private Signature (BAPS) system is associated with a message space $\mathcal{M}$, an attribute space $\mathcal{X}$, a witness space $\mathcal{W}$, a disclosing space

$\mathcal{DS}$, a family $\mathcal{P} := \{P : \mathcal{M} \times \mathcal{X} \times \mathcal{W} \longrightarrow \{0,1\}\}$ of policies, and a family $\mathcal{F} := \{F : \mathcal{P} \times \mathcal{X} \to \mathcal{DS}\}$ of disclosing functions.

A BAPS system is set up by a trusted party, whose jobs include generating public parameters and creating secret keys for the bicameral authorities, namely, the attribute-issuing authority and the policy-issuing authority. Policies (of organizations) and attributes (of users) are authorized and added to the system by the corresponding authorities. A signer, given a certified attribute $\mathbf{x} \in \mathcal{X}$ and a certified policy $P \in \mathcal{P}$, can issue a publicly verifiable signature Σ on a message $m \in \mathcal{M}$ if the signer possesses a witness w such that $P(m, \mathbf{x}, w) = 1$. Here, similar to [2,39,48], witness w can be viewed as a context-dependent string which serves as evidence why m is signable with respect to $\mathbf{x}$ and P. In addition, the signer stores a private clue c associated with (m, Σ), which later can be utilized for auditing purposes – should the need arise.

When asked (e.g., by a court or an auditor) to disclose certain partial information of P and/or $\mathbf{x}$ according to a disclosing function $F \in \mathcal{F}$, the signer of a pair (m, Σ) uses the associated clue c to generate a publicly verifiable testimony-attestation pair (t, a). Here, the attestation intuitively demonstrates that the value of the testimony is exactly determined as $t = F(P, \mathbf{x}) \in \mathcal{DS}$, where $(P, \mathbf{x})$ is precisely the policy-attribute pair underlying the message-signature pair (m, Σ). Such a disclosure process can be done multiple times with respect to different disclosing functions in $\mathcal{F}$.

2.1 Syntax of BAPS

A BAPS system associated with $(\mathcal{M}, \mathcal{X}, \mathcal{W}, \mathcal{DS}, \mathcal{P}, \mathcal{F})$ is a tuple of polynomial-time algorithms $(\mathsf{Setup}, \mathsf{Attribute\text{-}Iss}, \mathsf{Policy\text{-}Iss}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Disclose}, \mathsf{Judge})$, defined as follows.

$\mathsf{Setup}(1^\lambda)$**:** On input a security parameter λ, it outputs $(\mathsf{PP}, \mathsf{msk}_\mathsf{X}, \mathsf{msk}_\mathsf{P})$, where:
- PP denotes the public parameters which include, among others, the descriptions of $\mathcal{M}$, $\mathcal{X}$, $\mathcal{W}$, $\mathcal{DS}$, $\mathcal{P}$, $\mathcal{F}$;
- msk_X is the master attribute key and msk_P is the the master policy key.

All the subsequent algorithms take PP as an implicit input.

$\mathsf{Attribute\text{-}Iss}(\mathsf{msk}_\mathsf{X}, \mathbf{x})$**:** The attribute-issuing algorithm takes as inputs the key msk_X and an attribute $\mathbf{x} \in \mathcal{X}$. It outputs a signing key $\mathsf{sk}_\mathbf{x}$ for $\mathbf{x}$.

$\mathsf{Policy\text{-}Iss}(\mathsf{msk}_\mathsf{P}, P)$**:** The policy-issuing algorithm takes as inputs the key msk_P and a policy $P \in \mathcal{P}$. It outputs a certificate Cert_P for P.

$\mathsf{Sign}(\mathbf{x}, \mathsf{sk}_\mathbf{x}, P, \mathsf{Cert}_P, m, w)$**:** The signing algorithm takes as inputs a signing key $\mathsf{sk}_\mathbf{x}$ for an attribute $\mathbf{x}$, a certificate Cert_P for a policy P, a message $m \in \mathcal{M}$ and a witness w. It outputs either a signature Σ together with a clue c, or the symbol $\perp$ indicating failure.

$\mathsf{Verify}(m, \Sigma)$**:** On input a message-signature pair (m, Σ), the verification algorithm outputs 1 or 0, indicating the (in)validity of the signature Σ on m.

$\mathsf{Disclose}((m, \Sigma), c, F)$**:** On input a message-signature pair (m, Σ), a clue c and a disclosing function $F \in \mathcal{F}$, the disclosing algorithm outputs a testimony t together with an attestation a.

$\mathsf{Judge}((m, \Sigma), F, (t, a))$**:** On input a message-signature pair (m, Σ), a disclosing function F and a testimony-attestation pair (t, a), this judging algorithm outputs 1 or 0, indicating the (in)validity of the disclosure.

2.2 Correctness and Security of BAPS

Correctness. Intuitively, the correctness of BAPS guarantees that honestly generated message-signature pairs are accepted by Verify, that faithfully generated testimony-attestation pairs are accepted by Judge, and that the testimony associated with $F, P, \mathbf{x}$ must precisely be $t = F(P, x)$. Formally, a BAPS scheme is correct if for any $P \in \mathcal{P}$, $\mathbf{x} \in \mathcal{X}$, $m \in \mathcal{M}$ and $w \in \mathcal{W}$ such that $P(m, \mathbf{x}, w) = 1$, and for any $F \in \mathcal{F}$, it holds that

$$\Pr\left[\begin{array}{l|l} \mathsf{Verify}(m, \Sigma) = 1, & (\mathsf{PP}, \mathsf{msk}_\mathsf{X}, \mathsf{msk}_\mathsf{P}) \leftarrow \mathsf{Setup}(1^\lambda), \\ t = F(P, x), & \mathsf{sk}_\mathbf{x} \leftarrow \mathsf{Attribute\text{-}Iss}(\mathsf{msk}_\mathsf{X}, \mathbf{x}), \\ \mathsf{Judge}(m, \Sigma, F, t, a) = 1 & \mathsf{Cert}_P \leftarrow \mathsf{Policy\text{-}Iss}(\mathsf{msk}_\mathsf{P}, P), \\ & (\Sigma, c) \leftarrow \mathsf{Sign}(\mathbf{x}, \mathsf{sk}_\mathbf{x}, P, \mathsf{Cert}_P, m, w), \\ & (t, a) \leftarrow \mathsf{Disclose}((m, \Sigma), c, F) \end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

Security. We will first discuss the security features that any BAPS is expected to satisfy, and then present the formal definitions that capture these features.

We expect that a secure BAPS should provide the following guarantees.

- Justifiability of signatures: Any valid signature Σ on any message m should be associated with some policy P, some attribute $\mathbf{x}$ and some witness w such that $P(m, \mathbf{x}, w) = 1$. This must hold even if both authorities are corrupted.
- Necessity of possessing attribute keys: Without possessing a legitimate signing key $\mathsf{sk}_\mathbf{x}$ for some attribute $\mathbf{x}$, it should be infeasible to generate any valid signature. This property captures the security of the attribute-issuing authority, and it must hold even when the policy-issuing authority is corrupted.
- Necessity of possessing policy certificates: Without possessing a legitimate certificate Cert_P for some policy P, it would be infeasible to generate any valid signature. This property is orthogonal to the preceding one: it captures the security of the policy-issuing authority in the presence of a potentially corrupted attribute-issuing authority.
- Auditability: If $\mathsf{Judge}(m, \Sigma, F, t, a) = 1$, then it should hold that $t = F(P, \mathbf{x})$, where P and $\mathbf{x}$ are the policy and attribute underlying the message-signature pair (m, Σ). This property ensures the infeasibility of misleading disclosure results, and it must hold even if both authorities are corrupted.
- Privacy: This property guarantees that a valid message-signature pair (m, Σ) does not leak any information about the underlying policy P and attribute $\mathbf{x}$ (apart from the fact that $P(m, \mathbf{x}, w) = 1$ for some w). Privacy should

hold when both authorities are corrupted. Moreover, it can even be defined in the statistical sense (similar to ring signatures [4] and attribute-based signatures [45]).

– Auditable Privacy: This property says that no additional information about P or $\mathbf{x}$ can be learned from the disclosure result (t, a) beyond the fact that $F(P, \mathbf{x}) = t$. In other words, it guarantees the "residual" privacy of P and $\mathbf{x}$ after (potentially many) disclosures of their partial information. It should hold against corrupted authorities and can even be defined in the strong, statistical sense.

While other security features are somewhat reminiscent of similar properties in existing privacy-preserving signature primitives [13,51,31,54,52,9,32,39,48], auditable privacy is a distinguished property of BAPS. It allows the signer to flexibly and securely disclose selected pieces of private information when asked by different auditors. This property is sharply different from the notion of accountable privacy in group signatures [13] and variants [31,54,52,32,39,48], which demands that each signature contains a fixed piece of information about the signer that can be recovered by a designated party.

How to formalize Privacy and Auditable Privacy? In our first attempt to define privacy and auditable privacy for BAPS, we follow an indistinguishability-based approach. Specifically, the adversary provides a message m, two policies P_0, P_1, two attributes $\mathbf{x}_0, \mathbf{x}_1$, and two corresponding witnesses w_0, w_1 such that m is signable under both pairs $(P_0, \mathbf{x}_0, w_0)$ and $(P_1, \mathbf{x}_1, w_1)$, i.e., $P_0(m, \mathbf{x}_0, w_0) = 1$ and $P_1(m, \mathbf{x}_1, w_1) = 1$. The challenger then chooses a random bit b and generates a challenge signature Σ^* on m using $(P_b, \mathbf{x}_b, w_b, \mathsf{Cert}_{P_b}, \mathsf{sk}_{\mathbf{x}_b})$. Ideally, we would like to consider a strong adversary who can fully corrupt both authorities and can adaptively query the disclosing oracle with respect to the challenge message-signature pair (m, Σ^*) and any disclosing function F of its choice. However, if the adversary queries the disclosing algorithm for some F such that $F(P_0, \mathbf{x}_0) \neq F(P_1, \mathbf{x}_1)$, then it can easily guess the bit b. Hence, for the definition to be satisfiable, we must restrict the adversary's choice of $(P_0, \mathbf{x}_0, w_0)$ and $(P_1, \mathbf{x}_1, w_1)$ and require that $F(P_0, \mathbf{x}_0) = F(P_1, \mathbf{x}_1)$ for all F for which the adversary queries the disclosing algorithm.

Unfortunately, even with the above restriction, the indistinguishability-based approach might still be inadequate in capturing the expected notion of privacy. Suppose that the system only allows a single disclosing function, which is the identity function $F(P, \mathbf{x}) = (P, \mathbf{x})$. Then, to prevent the adversary from trivially winning, we must either demand that $(P_0, \mathbf{x}_0) = (P_1, \mathbf{x}_1)$ or totally prohibit disclosing queries with respect to the challenge message-signature pair.

We then take a step back and note that our major goal here is to ensure that, apart from the testimonies of the form $t = F(P, \mathbf{x})$, no additional knowledge about P and $\mathbf{x}$ is leaked. This requirement is quite similar to those in the contexts of zero-knowledge proofs [23], and functional encryption [6], where a simulation-based approach has been employed and widely accepted. We, therefore, adopt this approach, which allows us to formalize privacy and auditable privacy via a unified notion: simulatability.

Simulatability. We formalize this simulation-based notion by requiring the existence of three auxiliary algorithms $\mathsf{SimSetup}$, $\mathsf{SimSign}$, and $\mathsf{SimDisclose}$.

$\mathsf{SimSetup}(1^\lambda)$**:** On input λ, this algorithm outputs public parameters PP, keys $\mathsf{msk}_\mathsf{X}, \mathsf{msk}_\mathsf{P}$ for two authorities, together with a trapdoor tr.

$\mathsf{SimSign}(\mathsf{tr}, m, P, \mathbf{x}, w)$**:** This algorithm takes as inputs the trapdoor tr, and a message m, a policy P, an attribute $\mathbf{x}$, and a witness w. If $P(m, \mathbf{x}, w) = 0$, it returns 0. Otherwise, it returns a simulated signature. Note that a signing key $\mathsf{sk}_\mathbf{x}$ or a certificate Cert_P is not needed here.

$\mathsf{SimDisclose}(m, \Sigma, \mathsf{tr}, P, \mathbf{x}, F)$**:** This algorithm takes as inputs a valid message-signature pair (m, Σ), the trapdoor tr, a policy-attribute pair $(P, \mathbf{x})$, and a disclosing function F. It returns (t, a) as an output.

Intuitively, the simulatability of BAPS guarantees that the outputs of the above algorithms are indistinguishable from those of the real algorithms. This notion is modeled via experiment $\mathbf{Exp}^{\mathsf{sim}}_{\mathcal{A}}(\lambda)$ in Fig. 1 and is formally defined below.

$b \xleftarrow{\$} \{0,1\}$, $i \leftarrow 0$, $j \leftarrow 0$, $k \leftarrow 0$;
$(\mathsf{PP}^0, \mathsf{msk}^0_\mathsf{X}, \mathsf{msk}^0_\mathsf{P}, \mathsf{tr}) \leftarrow \mathsf{SimSetup}(1^\lambda)$; $(\mathsf{PP}^1, \mathsf{msk}^1_\mathsf{X}, \mathsf{msk}^1_\mathsf{P}) \leftarrow \mathsf{Setup}(1^\lambda)$;
$\mathsf{st} = (\mathsf{PP}^b, \mathsf{msk}^b_\mathsf{X}, \mathsf{msk}^b_\mathsf{P})$.
$b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sor}}_{\mathsf{PolicyKey}}, \mathcal{O}^{\mathsf{sor}}_{\mathsf{AttributeKey}}, \mathcal{O}^{\mathsf{sor}}_{\mathsf{Sign}}, \mathcal{O}^{\mathsf{sor}}_{\mathsf{Disclose}}}(\mathsf{st})$;
If $b = b'$, return 1; otherwise return 0.

$\underline{\mathcal{O}^{\mathsf{sor}}_{\mathsf{Sign}}(i^*, j^*, m, w)}$
If $i^* \notin [1, i]$ or $j^* \notin [1, j]$, return $\perp$;
Let $P = L_\mathcal{P}[i^*][0]$ and $\mathbf{x} = L_\mathcal{X}[j^*][0]$;
If $P(m, \mathbf{x}, w) = 0$, return $\perp$;
$k \leftarrow k + 1$;
$\Sigma^{*0} \leftarrow \mathsf{SimSign}(\mathsf{tr}, m, P, \mathbf{x}, w)$;
Let $\mathsf{Cert}_P = L_\mathcal{P}[i^*][1]$ and $\mathsf{sk}_\mathbf{x} = L_\mathcal{X}[j^*][1]$;
$(\Sigma^{*1}, c^1) \leftarrow \mathsf{Sign}(\mathbf{x}, \mathsf{sk}_\mathbf{x}, P, \mathsf{Cert}_P, m, w)$;
$L_S[k][0] = (P, \mathbf{x})$; $L_S[k][1] = c^1$;
$L_S[k][2] = (m, \Sigma^{*b})$;
Return Σ^{*b}.

$\underline{\mathcal{O}^{\mathsf{sor}}_{\mathsf{Disclose}}(m, \Sigma, F)}$
Check if $\exists\ k^\star$ such that $L_S[k^*][2] = (m, \Sigma)$;
If $k^\star$ does not exist, return $\perp$;
Let $(P, \mathbf{x}) = L_S[k^*][0]$; $c^1 = L_S[k^*][1]$;
$(t^0, a^0) \leftarrow \mathsf{SimDisclose}(m, \Sigma, \mathsf{tr}, (P, \mathbf{x}), F)$;
$(t^1, a^1) \leftarrow \mathsf{Disclose}(m, \Sigma, c^1, F)$;
Return (t^b, a^b).

$\underline{\mathcal{O}^{\mathsf{sor}}_{\mathsf{PolicyKey}}(P)}$
$i \leftarrow i + 1$;
$\mathsf{Cert}^0_P \leftarrow \mathsf{Policy\text{-}Iss}(\mathsf{msk}^0_\mathsf{P}, P)$;
$\mathsf{Cert}^1_P \leftarrow \mathsf{Policy\text{-}Iss}(\mathsf{msk}^1_\mathsf{P}, P)$;
$L_\mathcal{P}[i][0] = P$, $L_\mathcal{P}[i][1] = \mathsf{Cert}^1_P$;
Return Cert^b_P.

$\underline{\mathcal{O}^{\mathsf{sor}}_{\mathsf{AttributeKey}}(\mathbf{x})}$
$j \leftarrow j + 1$;
$\mathsf{sk}^0_\mathbf{x} \leftarrow \mathsf{Attribute\text{-}Iss}(\mathsf{msk}^0_\mathsf{X}, \mathbf{x})$;
$\mathsf{sk}^1_\mathbf{x} \leftarrow \mathsf{Attribute\text{-}Iss}(\mathsf{msk}^1_\mathsf{X}, \mathbf{x})$;
$L_\mathcal{X}[j][0] = \mathbf{x}$, $L_\mathcal{X}[j][1] = \mathsf{sk}^1_\mathbf{x}$;
Return $\mathsf{sk}^b_\mathbf{x}$.

Fig. 1: Experiment $\mathbf{Exp}^{\mathsf{sim}}_{\mathcal{A}}(\lambda)$

Definition 1 (Simulatability). *A BAPS scheme is said to satisfy simulatability if the advantage of $\mathcal{A}$ involved in experiment $\mathbf{Exp}^{\mathsf{sim}}_{\mathcal{A}}(\lambda)$, defined as $\mathbf{Adv}^{\mathsf{sim}}_{\mathcal{A}}(\lambda) = |\mathsf{Pr}[\mathbf{Exp}^{\mathsf{sim}}_{\mathcal{A}}(\lambda) = 1] - 1/2|$, is $\mathsf{negl}(\lambda)$. We say that the BAPS scheme is computationally simulatable if the advantage of any PPT algorithm $\mathcal{A}$ is negligible in λ. It is statistically simulatable if the advantage of any algorithm $\mathcal{A}$ is negligible and perfectly simulatable if the advantage of any algorithm $\mathcal{A}$ is zero.*

We next aim at formalizing other expected security properties of BAPS, i.e., justifiability of signatures, auditability, necessity of possessing attribute keys, and policy certificates. This turns out to be a non-trivial task. The major reason is that the syntax of BAPS does not provide a rigorous mechanism to determine whether a valid message-signature pair (m, Σ) is actually associated with some P, $\mathbf{x}$ and w such that $P(m, \mathbf{x}, w) = 1$. Therefore, for definitional purposes, we would need to introduce some auxiliary procedure that allows us to extract additional information from (m, Σ), e.g., some P, some $\mathbf{x}$, and some w, so that we can meaningfully explain whether and how a violation of security has occurred. In addition, such extraction should be possible even in the simulated setting. Thus, we further assume the existence of algorithm $\mathsf{Extract}$ defined below.

$\mathsf{Extract}(\mathsf{tr}, m, \Sigma)$**:** Given the trapdoor tr and a valid message-signature pair (m, Σ), it returns a tuple $(P, \mathbf{x}, w) \in \mathcal{P} \times \mathcal{X} \times \mathcal{W}$.

Equipped with such an extractable mode, we are now ready to formalize the expected properties. For the sake of simpler terminology, we will consider the following three notions:

- $\mathsf{Soundness}$ ensures that the extracted tuple $(P, \mathbf{x}, w)$ satisfies $P(m, \mathbf{x}, w) = 1$ and $t = F(P, \mathbf{x})$, for any testimony t outputted by a disclosing process involving function F. Note that soundness captures both "justifiability of signatures" and "auditability".
- $\mathsf{Unforgeability\text{-}I}$ addresses the "necessity of possessing attribute keys" and aims to protect the attribute-issuing authority.
- $\mathsf{Unforgeability\text{-}II}$ addresses the "necessity of possessing policy certificates" and aims to protect the policy-issuing authority.

We then define $\mathsf{extractability}$ as the notion unifying soundness, unforgeability-I, and unforgeability-II. This is a reminiscence of the "sim-ext" spirit [12], which was also employed in the context of policy-based signatures [2].

Extractability. We model the three requirements of extractability in Fig. 2 using three experiments $\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda)$, $\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathcal{A}}(\lambda)$, $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathcal{A}}(\lambda)$. All experiments are run between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Definition 2 (Extractability). *A BAPS scheme is said to satisfy the extractability property if there exists an additional algorithm* $\mathsf{Extract}$ *(as defined above) in the simulated setup, and for any PPT adversary $\mathcal{A}$ involved in the*

experiments $\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda)$, $\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathcal{A}}(\lambda)$, $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathcal{A}}(\lambda)$, *one has*

$$\mathbf{Adv}^{\mathsf{sound}}_{\mathcal{A}}(\lambda) = \mathsf{Pr}[\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda) = 1] \in \mathsf{negl}(\lambda);$$
$$\mathbf{Adv}^{\mathsf{Uf\text{-}I}}_{\mathcal{A}}(\lambda) = \mathsf{Pr}[\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathcal{A}}(\lambda) = 1] \in \mathsf{negl}(\lambda);$$
$$\mathbf{Adv}^{\mathsf{Uf\text{-}II}}_{\mathcal{A}}(\lambda) = \mathsf{Pr}[\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathcal{A}}(\lambda) = 1] \in \mathsf{negl}(\lambda).$$

1 $(\mathsf{PP}, \mathsf{msk}_{\mathsf{X}}, \mathsf{msk}_{\mathsf{P}}, \mathsf{tr}) \leftarrow \mathsf{SimSetup}(1^{\lambda})$; $k \leftarrow 0$;
2 $\mathsf{st} = (\mathsf{PP}, \mathsf{msk}_{\mathsf{X}}, \mathsf{msk}_{\mathsf{P}})$; $(m, \Sigma) \leftarrow \mathcal{A}(\mathsf{st})$;
3 $\mathsf{st} = (\mathsf{PP}, \mathsf{msk}_{\mathsf{P}})$; $(m, \Sigma) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{AttriKey}}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Dis}}}(\mathsf{st})$;
4 $\mathsf{st} = (\mathsf{PP}, \mathsf{msk}_{\mathsf{X}})$; $(m, \Sigma) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{PolicyKey}}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Dis}}}(\mathsf{st})$;
5 If $(m, \Sigma) = L_S[k^\star][1]$ for $k^\star \in [1, k]$ or $\mathsf{Verify}(\mathsf{PP}, m, \Sigma) = 0$, return 0.
6 $(P^*, \mathbf{x}^*, w^*) \leftarrow \mathsf{Extract}(\mathsf{tr}, m, \Sigma)$;
7 If $P^*(m, \mathbf{x}^*, w^*) = 0$, return 1.
8 If $\mathbf{x}^* \notin Q_X$, return 1. | If $P^* \notin Q_P$, return 1.
9 $(F, t, a) \leftarrow \mathcal{A}(\mathsf{st})$;
If $(m, \Sigma, F, t, a) \in Q_D$ or $\mathsf{Judge}(\mathsf{PP}, m, \Sigma, F, t, a) = 0$, return 0.
If $t \neq F(P^*, \mathbf{x}^*)$, return 1.
10 Return 0.

$\mathcal{O}_{\mathsf{AttriKey}}(\mathbf{x})$
$\mathsf{sk}_{\mathbf{x}} \leftarrow \mathsf{Attribute\text{-}Iss}(\mathsf{msk}_{\mathsf{X}}, \mathbf{x})$;
$Q_X \leftarrow Q_X \cup \{\mathbf{x}\}$
Return $\mathsf{sk}_{\mathbf{x}}$.

$\mathcal{O}_{\mathsf{Sign}}(m, P, \mathbf{x}, w)$
If $P(m, \mathbf{x}, w) = 0$, return $\perp$;
$k \leftarrow k + 1$;
$\Sigma \leftarrow \mathsf{SimSign}(\mathsf{tr}, m, P, \mathbf{x}, w)$;
$L_S[k][0] = (P, \mathbf{x}), L_S[k][1] = (m, \Sigma)$;
Return Σ.

$\mathcal{O}_{\mathsf{PolicyKey}}(P)$
$\mathsf{Cert}_P \leftarrow \mathsf{Policy\text{-}Iss}(\mathsf{msk}_{\mathsf{P}}, P)$;
$Q_P \leftarrow Q_P \cup \{P\}$;
Return Cert_P.

$\mathcal{O}_{\mathsf{Dis}}(m, \Sigma, F)$
Check if $\exists\ k^\star \in [1, k]$ such that $L_S[k^\star][1] = (m, \Sigma)$;
If $k^\star$ does not exist, return $\perp$;
Let $(P, \mathbf{x}) = L_S[k^\star][0]$;
$(t, a) \leftarrow \mathsf{SimDisclose}(m, \Sigma, \mathsf{tr}, P, \mathbf{x}, F)$;
$Q_D \leftarrow Q_D \cup \{(m, \Sigma, F, t, a)\}$;
Return (t, a).

Fig. 2: Experiments $\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda)$ (excluding dotted and double solid boxes), $\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathcal{A}}(\lambda)$ (excluding solid and double solid boxes), and $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathcal{A}}(\lambda)$ (excluding the solid and dotted boxes).

The experiment $\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda)$ operates in two stages which first defines an extractable mode of the scheme allowing to extract a policy P^*, an attribute $\mathbf{x}^*$, and a witness w^* from any valid message-signature pair (m, Σ). Such an extraction then enables evaluating the value $P^*(m, \mathbf{x}^*, w^*)$ a posteriori. The

adversary wins the experiment if the evaluated value is 0, indicating that m is not signable with respect to P^* and $\mathbf{x}^*$. It proceeds to the second stage only if $P^*(m, \mathbf{x}^*, w^*) = 1$, i.e., the adversary did not win in the first stage. The aim of the adversary in the second stage is to output a disclosing function F and a Judge-accepted testimony-attestation pair (t, a) corresponding to (m, Σ) outputted in the first stage such that $t \neq F(P^*, \mathbf{x}^*)$. To define soundness in the strongest sense, both authorities' keys are exposed to the adversary. A BAPS system satisfies soundness if the winning probability of the adversary in $\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda)$ is negligible in λ. Said otherwise, even with the help of both fully corrupted authorities, no signer can fool the system by producing valid signatures on non-signable messages or creating a testimony accompanied by an accepted attestation that does not respect the underlying disclosing value $F(P^*, \mathbf{x}^*)$.

Both experiments $\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathcal{A}}(\lambda)$, $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathcal{A}}(\lambda)$ function in the extractable setting.

- Unforgeability-I is similar to the unforgeability/type-1 unforgeability notion of ABS [45]/MPS [48]. It protects the security of the attribute-issuing authority. In the experiment, the adversary fully corrupts the policy-issuing authority and can learn their signing keys on attributes of its choices via $\mathcal{O}_{\mathsf{AttriKey}}$. The adversary also makes signing queries by submitting $(m, P, \mathbf{x}, w)$ to $\mathcal{O}_{\mathsf{Sign}}$. We stress that the signing key of $\mathbf{x}$ may not be revealed to the adversary. Its goal is to output a valid pair (m, Σ) such that the extraction points to an attribute of which it has not previously learned the signing key.
- Unforgeability-II captures the spirit of the unforgeability/extractability notion in FS [9]/PS [1]/PBS [2]. This notion is orthogonal to unforgeability-I and protects the security of the policy-issuing authority. In the experiment, the adversary has access to various oracles and intends to output a valid pair (m, Σ) that is extracted to a policy $P^* \notin Q_P$.

3 A Generic Construction for BAPS

In this section, we present a generic construction of BAPS for arbitrary policies and arbitrary disclosing functions. The construction satisfies the correctness and the security requirements defined Section 2.2. It employs several commonly used cryptographic building blocks: two signature schemes, a commitment scheme, and two non-interactive zero-knowledge (NIZK) argument systems for some NP-relations. The definitions and the required security properties of these primitives are recalled in Appendix A.

As a remark, we require that the employed NIZK systems satisfy the simulation-sound extractability property [24]. We note that it could be possible to replace these building blocks with a combination of ordinary NIZK systems, (lossy) public-key encryption, and one-time signatures (similar to a construction in [2]). The resulting construction, however, would be syntactically much more complicated while relying on essentially the same high-level ideas.

We will give a technical overview of the construction in Section 3.1, describe it in detail in Section 3.2, and provide its analyses in Section 3.3.

3.1 Technical Overview

The construction employs the following cryptographic building blocks.

– Two secure signature schemes

$$\mathcal{S}_{\mathsf{X}} = (\mathsf{S_X.Kg}, \mathsf{S_X.Sign}, \mathsf{S_X.Ver}), \quad \mathcal{S}_{\mathsf{P}} = (\mathsf{S_P.Kg}, \mathsf{S_P.Sign}, \mathsf{S_P.Ver});$$

– A secure commitment scheme $\mathcal{COM} = (\mathsf{C.Setup}, \mathsf{C.Com}, \mathsf{C.Open})$;
– Two simulation-sound extractable NIZK systems

$$\mathcal{NIZK}_{\mathsf{S}} = (\mathsf{ZK_S.Setup}, \mathsf{ZK_S.SimSetup}, \mathsf{ZK_S.Prove}, \mathsf{ZK_S.Ver}, \mathsf{ZK_S.Sim}, \mathsf{ZK_S.Extr}),$$
$$\mathcal{NIZK}_{\mathsf{D}} = (\mathsf{ZK_D.Setup}, \mathsf{ZK_D.SimSetup}, \mathsf{ZK_D.Prove}, \mathsf{ZK_D.Ver}, \mathsf{ZK_D.Sim}, \mathsf{ZK_D.Extr}),$$

for the NP-relations $\mathcal{R}_{\mathsf{S}}$ and $\mathcal{R}_{\mathsf{D}}$, respectively, defined below.

The attribute-issuing authority is associated with a signing-verification key-pair $(\mathsf{msk_X}, \mathsf{vk_X})$ for $\mathcal{S}_{\mathsf{X}}$. A signing key $\mathsf{sk}_{\mathbf{x}}$ for attribute $\mathbf{x}$ is defined as a signature of the authority on "message" $\mathbf{x}$.

Similarly, the policy-issuing authority is associated with a signing-verification key-pair $(\mathsf{msk_P}, \mathsf{vk_P})$ for $\mathcal{S}_{\mathsf{P}}$. A certificate Cert_P for policy P is then a signature of the authority on "message" P.

To sign a message m with respect to policy P, attribute $\mathbf{x}$ and witness w, the signer first checks whether $P(m, \mathbf{x}, w) = 1$, and aborts if this is not the case. Next, the signer commits to P and $\mathbf{x}$ as com_P and $\mathsf{com}_{\mathbf{x}}$, respectively. Then, it proves that (i) m is signable, i.e., $P(m, \mathbf{x}, w) = 1$; (ii) each of P and $\mathbf{x}$ is properly certified by the respective authority; and (iii) com_P and $\mathsf{com}_{\mathbf{x}}$ are valid commitments to P and $\mathbf{x}$, respectively. Specifically, the signer generates a NIZK argument Π for the following relation

$$\begin{aligned}\mathcal{R}_{\mathsf{S}} := \Big\{ \ \Big(\ (m, \mathsf{vk_X}, \mathsf{vk_P}, \mathsf{pp_C}, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_P), \ \ (\mathbf{x}, \mathsf{sk}_{\mathbf{x}}, P, \mathsf{Cert}_P, w, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},P}) \ \Big) : \\ \Big(P(m, \mathbf{x}, w) = 1\Big) \wedge \Big(\mathsf{S_X.Ver}(\mathsf{vk_X}, \mathbf{x}, \mathsf{sk}_{\mathbf{x}}) = 1\Big) \wedge \Big(\mathsf{S_P.Ver}(\mathsf{vk_P}, P, \mathsf{Cert}_P) = 1\Big) \\ \wedge \Big(\mathsf{C.Open}(\mathsf{pp_C}, \mathsf{com}_{\mathbf{x}}, \mathbf{x}, \mathbf{r}_{\mathsf{com},\mathbf{x}}) = 1\Big) \wedge \Big(\mathsf{C.Open}(\mathsf{pp_C}, \mathsf{com}_P, P, \mathbf{r}_{\mathsf{com},P}) = 1\Big) \Big\}.\end{aligned}$$

The signature Σ then contains the commitments com_P and $\mathsf{com}_{\mathbf{x}}$ as well as the argument Π. Verification of Σ is simply the verification of Π. The clue – which will be used for later disclosures – consists of $\mathbf{x}, P$ and the randomnesses $\mathbf{r}_{\mathsf{com},\mathbf{x}}$ and $\mathbf{r}_{\mathsf{com},P}$.

We remark that the design approach being used here, which we term *sign-then-commit-then-prove*, is effectively different from the sign-then-encrypt-then-prove paradigm traditionally used for achieving accountable privacy in group-signature-like systems. The advantage of our approach is that the signer can preserve full privacy of the committed values (especially if the underlying commitment scheme is statistically hiding) while maintaining the capability of proving additional relations about the committed values at later times.

When asked to disclose certain partial information of $\mathbf{x}$ and/or P according to a disclosing function $F \in \mathcal{F}$, the signer of a message-signature pair (m, Σ), who possesses the corresponding clue $c = (\mathbf{x}, P, \mathbf{r}_{\mathsf{com},P}, \mathbf{r}_{\mathsf{com},\mathbf{x}})$, first computes $t = F(P, \mathbf{x})$ and then proves that t is well-formed with respect to the values P and $\mathbf{x}$ committed in the signature. Specifically, it generates a NIZK argument a for the following relation:

$$\mathcal{R}_{\mathsf{D}} := \Big\{ \Big(F, t, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_P, \mathsf{pp}_{\mathsf{C}}), (\mathbf{x}, P, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},P}) \Big) : \Big(t = F(P, \mathbf{x})\Big) \\ \wedge \Big(\mathsf{C.Open}(\mathsf{pp}_{\mathsf{C}}, \mathsf{com}_{\mathbf{x}}, \mathbf{x}, \mathbf{r}_{\mathsf{com},\mathbf{x}}) = 1\Big) \wedge \Big(\mathsf{C.Open}(\mathsf{pp}_{\mathsf{C}}, \mathsf{com}_P, P, \mathbf{r}_{\mathsf{com},P}) = 1\Big) \Big\}.$$

To determine the validity of a testimony-attestation pair (t, a) outputted by the disclosing algorithm, one simply verifies the NIZK argument a.

The correctness and security properties of the construction tightly follow from those of the employed building blocks. In particular, the scheme is (statistically) simulatable as long as $\mathcal{NIZK}_{\mathsf{S}}$ and $\mathcal{NIZK}_{\mathsf{D}}$ are (statistically) zero-knowledge and $\mathcal{COM}$ is (statistically) hiding. Its extractability, on the other hand, relies on the unforgeability of $\mathcal{S}_{\mathsf{X}}$ and $\mathcal{S}_{\mathsf{P}}$, the simulation-sound extractability of the NIZK systems and the binding property of $\mathcal{COM}$.

3.2 Description

In the description of the generic construction, we do not specify the choice of system parameters $(\mathcal{M}, \mathcal{X}, \mathcal{W}, \mathcal{DS}, \mathcal{P}, \mathcal{F})$. We do not make any restriction on the policies in $\mathcal{P}$ nor the disclosing functions in $\mathcal{F}$. We, however, note that these system parameters should be compatible with those of the building blocks $\mathcal{S}_{\mathsf{X}}$ and $\mathcal{S}_{\mathsf{P}}$, $\mathcal{NIZK}_{\mathsf{S}}$ and $\mathcal{NIZK}_{\mathsf{D}}$, and $\mathcal{COM}$.

$\mathsf{Setup}(1^\lambda)$**:** On input the security parameter λ, this probabilistic algorithm performs the following steps:

1. Run $(\mathsf{msk}_{\mathsf{P}}, \mathsf{vk}_{\mathsf{P}}) \leftarrow \mathsf{S_P.Kg}(1^\lambda)$ and $(\mathsf{msk}_{\mathsf{X}}, \mathsf{vk}_{\mathsf{X}}) \leftarrow \mathsf{S_X.Kg}(1^\lambda)$ to obtain signing-verification key pairs for the policy-issuing authority and the attribute-issuing authority, respectively.
2. Run $\mathsf{ZK_S.Setup}(1^\lambda)$ and $\mathsf{ZK_D.Setup}(1^\lambda)$ to obtain $\mathsf{crs}_{\mathsf{S}}$ and $\mathsf{crs}_{\mathsf{D}}$ for the argument systems $\mathcal{NIZK}_{\mathsf{S}}$ and $\mathcal{NIZK}_{\mathsf{D}}$, respectively.
3. Generate public parameters $\mathsf{pp}_{\mathsf{C}} \leftarrow \mathsf{C.Setup}(1^\lambda)$ for $\mathcal{COM}$.

Let $\mathsf{PP} := (\mathsf{crs}_{\mathsf{S}}, \mathsf{crs}_{\mathsf{D}}, \mathsf{pp}_{\mathsf{C}}, \mathsf{vk}_{\mathsf{P}}, \mathsf{vk}_{\mathsf{X}})$ and output $(\mathsf{PP}, \mathsf{msk}_{\mathsf{P}}, \mathsf{msk}_{\mathsf{X}})$.

$\mathsf{Attribute\text{-}Iss}(\mathsf{msk}_{\mathsf{X}}, \mathbf{x})$**:** Generate a signing key $\mathsf{sk}_{\mathbf{x}}$ for attribute $\mathbf{x}$ as

$$\mathsf{sk}_{\mathbf{x}} \leftarrow \mathsf{S_X.Sign}(\mathsf{msk}_{\mathsf{X}}, \mathbf{x}).$$

$\mathsf{Policy\text{-}Iss}(\mathsf{msk}_{\mathsf{P}}, P)$**:** Generate a certificate Cert_P for policy P as

$$\mathsf{Cert}_P \leftarrow \mathsf{S_P.Sign}(\mathsf{msk}_{\mathsf{P}}, P).$$

$\mathsf{Sign}(\mathbf{x}, \mathsf{sk}_{\mathbf{x}}, P, \mathsf{Cert}_P, m, w)$**:** If $P(m, \mathbf{x}, w) = 0$, the signing algorithm returns $\perp$. Otherwise, it proceeds as follows.

1. Commit to $\mathbf{x}$ as $\mathsf{com}_{\mathbf{x}} = \mathsf{C.Com}(\mathsf{pp}_{\mathsf{C}}, \mathbf{x}, \mathbf{r}_{\mathsf{com},\mathbf{x}})$, and commit to P as $\mathsf{com}_P = \mathsf{Com}(\mathsf{pp}_{\mathsf{C}}, P, \mathbf{r}_{\mathsf{com},P})$. Here, $\mathbf{r}_{\mathsf{com},\mathbf{x}}$ and $\mathbf{r}_{\mathsf{com},P}$ are the commitment randomness.
2. Generate a NIZK argument Π to demonstrate the knowledge of a tuple $\eta = (\mathbf{x}, \mathsf{sk}_{\mathbf{x}}, P, \mathsf{Cert}_P, w, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},P})$ such that the following conditions hold.
   (a) The message m is signable with respect to policy P, attribute $\mathbf{x}$ and witness w, i.e., $P(m, \mathbf{x}, w) = 1$.
   (b) $\mathsf{com}_{\mathbf{x}}$ and com_P are valid commitments to $\mathbf{x}$ and P, respectively, i.e.,
   $$\mathsf{C.Open}(\mathsf{pp}_{\mathsf{C}}, \mathsf{com}_{\mathbf{x}}, \mathbf{x}, \mathbf{r}_{\mathsf{com},\mathbf{x}}) = 1; \quad \mathsf{C.Open}(\mathsf{pp}_{\mathsf{C}}, \mathsf{com}_P, P, \mathbf{r}_{\mathsf{com},P}) = 1.$$
   (c) $(\mathbf{x}, \mathsf{sk}_{\mathbf{x}})$ is a valid (attribute, signing key) pair, i.e.,
   $$\mathsf{S_X.Ver}(\mathsf{vk}_{\mathsf{X}}, \mathbf{x}, \mathsf{sk}_{\mathbf{x}}) = 1.$$
   (d) (P, Cert_P) is a valid (attribute, certificate) pair, i.e.,
   $$\mathsf{S_P.Ver}(\mathsf{vk}_{\mathsf{P}}, P, \mathsf{Cert}_P) = 1.$$
   This is done by running
   $$\Pi \leftarrow \mathsf{ZK_S.Prove}\big(\ \mathsf{crs}_{\mathsf{S}}, (m, \mathsf{vk}_{\mathsf{X}}, \mathsf{vk}_{\mathsf{P}}, \mathsf{pp}_{\mathsf{C}}, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_P), \eta\ \big)$$
   to prove that $\big(\ (m, \mathsf{vk}_{\mathsf{X}}, \mathsf{vk}_{\mathsf{P}}, \mathsf{pp}_{\mathsf{C}}, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_P), \eta\ \big) \in \mathcal{R}_{\mathsf{S}}$.
3. Let
   $$\Sigma = (\mathsf{com}_{\mathbf{x}}, \mathsf{com}_P, \Pi); \quad \mathsf{clue} = (\mathbf{x}, P, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},P}),$$
   and return (Σ, clue).

$\mathsf{Verify}(m, \Sigma)$**:** Parse $\Sigma = (\mathsf{com}_{\mathbf{x}}, \mathsf{com}_P, \Pi)$. Then return the bit
$$b' \leftarrow \mathsf{ZK_S.Ver}\big(\ \mathsf{crs}_{\mathsf{S}}, (m, \mathsf{vk}_{\mathsf{X}}, \mathsf{vk}_{\mathsf{P}}, \mathsf{pp}_{\mathsf{C}}, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_P), \Pi\ \big).$$

$\mathsf{Disclose}(m, \Sigma, \mathsf{clue}, F)$**:** On input a valid message-signature pair (m, Σ), where $\Sigma = (\mathsf{com}_{\mathbf{x}}, \mathsf{com}_P, \Pi)$, a clue $\mathsf{clue} = (\mathbf{x}, P, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},P})$, and a disclosing function $F \in \mathcal{F}$, this algorithm proceeds as follows.
1. Compute $t = F(P, \mathbf{x})$.
2. Generate a NIZK argument a to show the possession of the tuple $\mathsf{clue} = (\mathbf{x}, P, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},P})$ such that the following conditions hold.
   (i) The value t is honestly computed, i.e., $t = F(P, \mathbf{x})$.
   (ii) $\mathsf{com}_{\mathbf{x}}$ and com_P are valid commitments to $\mathbf{x}$ and P, respectively, i.e.,
   $$\mathsf{C.Open}(\mathsf{pp}_{\mathsf{C}}, \mathsf{com}_{\mathbf{x}}, \mathbf{x}, \mathbf{r}_{\mathsf{com},\mathbf{x}}) = 1; \quad \mathsf{C.Open}(\mathsf{pp}_{\mathsf{C}}, \mathsf{com}_P, P, \mathbf{r}_{\mathsf{com},P}) = 1.$$
   This is done by running
   $$a \leftarrow \mathsf{ZK_D.Prove}(\mathsf{crs}_{\mathsf{D}}, (F, t, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_P, \mathsf{pp}_{\mathsf{C}}), \mathsf{clue})$$
   to prove that $((F, t, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_P, \mathsf{pp}_{\mathsf{C}}), \mathsf{clue}) \in \mathcal{R}_{\mathsf{D}}$.
3. Return (t, a) as a testimony-attestation pair.

$\mathsf{Judge}(m, \Sigma, F, t, a)$**:** If $\mathsf{Verify}(m, \Sigma) = 0$, return 0. Otherwise, parse Σ as $\Sigma = (\mathsf{com}_{\mathbf{x}}, \mathsf{com}_P, \Pi)$ and return $b'' \leftarrow \mathsf{ZK_D.Ver}(\mathsf{crs}_{\mathsf{D}}, (F, t, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_P, \mathsf{pp}_{\mathsf{C}}), a)$.

3.3 Analyses

Correctness. The correctness of the presented BAPS scheme follows directly from the correctness/completeness of the employed ingredients.

The correctness of the commitment scheme $\mathcal{COM}$ ensures that

$$\mathsf{C.Open}(\mathsf{pp_C}, \mathsf{com_x}, \mathbf{x}, \mathbf{r}_{\mathsf{com},\mathbf{x}}) = 1, \quad \mathsf{C.Open}(\mathsf{pp_C}, \mathsf{com}_P, P, \mathbf{r}_{\mathsf{com},P}) = 1.$$

Also, it follows from the correctness of signature schemes $\mathcal{S}_\mathsf{X}$ and $\mathcal{S}_\mathsf{P}$ that

$$\mathsf{S_X.Ver}(\mathsf{vk_X}, \mathbf{x}, \mathsf{sk_x}) = 1, \quad \mathsf{S_P.Ver}(\mathsf{vk_P}, P, \mathsf{Cert}_P) = 1.$$

Therefore, an honest signer is able to obtain a valid witness

$$\eta = (\mathbf{x}, \mathsf{sk_x}, P, \mathsf{Cert}_P, w, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},P})$$

such that $\big(\ (m, \mathsf{vk_X}, \mathsf{vk_P}, \mathsf{pp_C}, \mathsf{com_x}, \mathsf{com}_P), \eta\ \big) \in \mathcal{R}_\mathsf{S}$. Then, thanks to the completeness of $\mathcal{NIZK}_\mathsf{S}$, an honestly generated proof Π will be accepted by $\mathsf{ZK_S.Ver}$. In other words, algorithm Verify returns 1 with overwhelming probability.

Next, if algorithm $\mathsf{Disclose}$ is run honestly, then one has that $t = F(P, \mathbf{x})$ and that $\mathsf{clue} = (\mathbf{x}, P, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},P})$ satisfies $((F, t, \mathsf{com_x}, \mathsf{com}_P, \mathsf{pp_C}), \mathsf{clue}) \in \mathcal{R}_\mathsf{D}$. As a result, algorithm $\mathsf{ZK_D.Ver}$ returns 1 with overwhelming probability, thanks to the completeness of $\mathcal{NIZK}_\mathsf{D}$, and so does algorithm Judge.

Security. The proposed generic construction satisfies simulatability and extractability as defined in Section 2.2. We summarize the security in Theorem 1 and Theorem 2, proofs of which are deferred to Appendix C.

Theorem 1. *Assume that the two NIZK systems for $\mathcal{R}_\mathsf{S}$ and $\mathcal{R}_\mathsf{D}$ are statistical zero-knowledge and the commitment scheme $\mathcal{COM}$ is statistically hiding. Then the proposed BAPS scheme is statistically simulatable.*

Theorem 2. *The proposed BAPS scheme is extractable if the two underlying NIZK systems satisfy simulation-sound extractability, $\mathcal{COM}$ is computationally binding, and the two signature schemes are* $\mathsf{EUF\text{-}CMA}$ *secure.*

4 A Lattice-Based BAPS Scheme

In this section, we present a concrete construction of BAPS, that is proven secure under lattice-based assumptions in the random oracle model. The scheme can address arbitrary policies, represented as polynomial-size Boolean circuits. Furthermore, it can handle those disclosing functions which can be expressed as quadratic functions of the bits of P and $\mathbf{x}$.

Policies as Boolean Circuits. Let integers $n, q, k_1, k_2, k_3, K, N$ be system parameters. Set $K = k_1 + k_2 + k_3$, $\delta_\mathsf{P} = \lfloor \log(K + N - 2) \rfloor + 1$, and $\bar{k} = 2N\delta_\mathsf{P} + k_1$. Our construction is associated with message space $\mathcal{M} = \{0,1\}^{k_1}$, attribute space $\mathcal{X} = \{0,1\}^{k_2}$, witness space $\mathcal{W} = \{0,1\}^{k_3}$, disclosing space $\mathcal{DS} = \{0,1\}^{\bar{k}}$, and arbitrary polynomial-size policies $\mathcal{P} = \{P : \{0,1\}^K \to \{0,1\}\}$. In particular,

$P \in \mathcal{P}$ has K-bit inputs and N NAND gates, and whose topology is determined by two functions $g, h : [0, N-1] \to [0, K+N-2]$. Namely, if the inputs to P are $s_0, s_1, \ldots, s_{K-1}$ and the N gate outputs are ordered as $s_K, \ldots, s_{K+N-1}$, then we have

$$s_{K+i} = s_{g(i)} \ \mathsf{NAND} \ s_{h(i)}, \ \forall i = 0, \ldots, N-1.$$

Since the construction requires signing, committing, and proving relations about policies, we first need an effective method for policy representation. To this end, by running the decomposition function described in Appendix B.2 on $g(i), h(i)$ for all $i \in [0, N-1]$, we obtain $\mathsf{idec}_{\mathsf{P}}(g(i)) = (g_{i,0}, \ldots, g_{i,\delta_{\mathsf{P}}-1})^\top$ and $\mathsf{idec}_{\mathsf{P}}(h(i)) = (h_{i,0}, \ldots, h_{i,\delta_{\mathsf{P}}-1})^\top$ for $i \in [0, N-1]$. Then we consider the following representation of the circuit P:

$$\begin{aligned} \mathbf{z}_P = (&g_{0,0}, g_{0,1}, \ldots, g_{0,\delta_{\mathsf{P}}-1}, \ldots, g_{N-1,0}, g_{N-1,1}, \ldots, g_{N-1,\delta_{\mathsf{P}}-1}, \\ &h_{0,0}, h_{0,1}, \ldots, h_{0,\delta_{\mathsf{P}}-1}, \ldots, h_{N-1,0}, h_{N-1,1}, \ldots, h_{N-1,\delta_{\mathsf{P}}-1})^\top \in \{0,1\}^{2N\delta_{\mathsf{P}}}. \end{aligned} \quad (1)$$

Quadratic Disclosing functions. Our construction can work with a collection $\mathcal{F} = \{F : \mathcal{P} \times \mathcal{X} \to \{0,1\}^{\bar{k}}\}$ containing polynomially many disclosing functions F, each of which is determined by two matrices $\mathbf{G}_1 \in \{0,1\}^{\bar{k} \times \bar{k}^2}$, $\mathbf{G}_2 \in \{0,1\}^{\bar{k} \times \bar{k}}$, and defined as

$$F(P, \mathbf{x}) = \mathbf{G}_1 \cdot (\mathbf{b} \otimes_{\mathrm{Kron}} \mathbf{b}) + \mathbf{G}_2 \cdot \mathbf{b} \bmod 2, \quad \text{with} \quad \mathbf{b} = (\mathbf{z}_P^\top \mid \mathbf{x}^\top)^\top \in \{0,1\}^{\bar{k}}.$$

Here, $\mathbf{b} \otimes_{\mathrm{Kron}} \mathbf{b} \in \{0,1\}^{\bar{k}^2}$ denotes the Kronecker product, i.e., a flattening of the tensor product $\mathbf{b} \otimes \mathbf{b} \in \{0,1\}^{\bar{k} \times \bar{k}}$. According to this definition of F, each coordinate of the vector $t = F(P, \mathbf{x}) \in \{0,1\}^{\bar{k}}$ has the form

$$\sum_{i,j \in [1,\bar{k}]} \alpha_{i,j} \cdot (b_i \cdot b_j) + \sum_{\ell \in [1,\bar{k}]} \beta_\ell \cdot b_\ell \bmod 2,$$

where $\alpha_{i,j}$'s, β_ℓ's are entries of $\mathbf{G}_1, \mathbf{G}_2$ and b_i's are coordinates of $\mathbf{b}$ – which are essentially the bits determining policy P and attribute $\mathbf{x}$. This definition is thus quite general and captures arbitrary linear and quadratic relations with respect to the bits of $(P, \mathbf{x})$. By setting matrices $\mathbf{G}_1$ and $\mathbf{G}_2$ appropriately, one can enforce the disclosures of any bits of $(P, \mathbf{x})$, or any bit-products, or any linear and/or quadratic combinations of the bits.

4.1 Technical Overview

At a high level, our lattice-based BAPS scheme follows the sign-then-commit-then-prove design approach of the generic construction in Section 3. However, when it comes to middle-level techniques, we do introduce several adjustments.

– Policies, which often have long representations, are hashed before being signed by the policy-issuing authority. This can help to reduce the key size of the underlying signature system.

– We handle the problem of evaluating $P(m, \mathbf{x}, w)$ in zero-knowledge via several involved sub-protocols. These sub-protocols are connected with each other via the use of some extra commitments.

More concretely, we make use of the following ideal-lattice-based ingredients.

– A variant of the Ducas-Micciancio (DM) signature scheme [15] (recalled in Appendix B.4), which is stateful and is adaptively secure in the standard model. It is used to instantiate the signature systems for both authorities, i.e., $\mathcal{S}_{\mathsf{X}}$ and $\mathcal{S}_{\mathsf{P}}$. The scheme is carefully chosen among existing lattice-based signatures because it has the shortest keys among schemes known to admit a concrete zero-knowledge argument of a valid message-signature pair [42].
– A secure hash function family $\mathcal{H}_{m_{\mathsf{P}}}$ (described in Appendix B.1), which is adapted from [46] and which will be used to hash the binary representations of policies.
– Two commitment families $\mathsf{CMT}_{\rho',m}$ and $\mathsf{CMT}_{\ell,m}$ (described in Appendix B.3), that are adapted from [30] and that will be used to commit to various objects.
– We also need two simulation-sound extractable NIZK systems that can handle linear and quadratic relations with respect to two moduli q and 2 (recalled in Appendix B.5), and should be compatible with the DM signature scheme. To this end, we employ a framework proposed in [36] for interactive Stern-like argument systems [53], and then apply the Fiat-Shamir transform [21] to obtain the desired properties [20]. These two systems internally employ a string commitment scheme CMT [30] and a hash function $\mathcal{H}_{\mathsf{FS}}$. The latter will be modeled as a random oracle in the security proofs.

Proving in ZK that $P(m, \mathbf{x}, w) = 1$. The major technical challenge that we have to overcome is to prove in ZK that a hidden-and-certified policy P evaluates to 1 on a public message m, a hidden-and-certified attribute $\mathbf{x}$, and a hidden witness w. To demonstrate our techniques, we define some notations.

Let $s_0, s_1, \ldots, s_{K-1}$ be the input bits of a policy P represented by a Boolean circuit consisting of N NAND gates, whose topology is determined by functions g and h. The task is to prove that the NAND gates are computed faithfully, i.e.,

$$\begin{cases} s_K \oplus s_{g(0)} \cdot s_{h(0)} = 1 \mod 2; \\ s_{K+1} \oplus s_{g(1)} \cdot s_{h(1)} = 1 \mod 2; \\ \cdots\cdots \\ s_{K+N-2} \oplus s_{g(N-2)} \cdot s_{h(N-2)} = 1 \mod 2; \\ s_{g(N-1)} \cdot s_{h(N-1)} = 0 \mod 2, \end{cases} \tag{2}$$

where s_{K+i} is the output while $s_{g(i)}, s_{h(i)}$ are the two inputs of the i-th NAND gates for $i \in [0, N-1]$. In order to compute s_{K+i} as in (2), we have to retrieve the values $s_{g(i)}, s_{h(i)}$, which are either the inputs to P or some intermediate values. Most importantly, we must show that the retrieval process is honestly performed.

To this end, a first idea, inspired by Libert et al. [35], is to perform a dichotomic search on $\mathbf{s} = (s_0, s_1, \ldots, s_{K-1}, s_K, \ldots, s_{K+N-2}) \in \{0,1\}^{K+N-1}$. The

expected complexity for a single search is $\mathcal{O}(\log(K+N)\cdot\lambda\cdot\log\lambda)$ with λ being the security parameter. We however take a different approach and perform a "bucket" search. The expected complexity for a single search is $\mathcal{O}(\sqrt{K+N}+\lambda\cdot\log\lambda)$. The new complexity is smaller if $K+N$ is a small polynomial (say $o(\lambda^3)$) in λ.

The high-level idea of our "bucket" search is as follows. We divide $\mathbf{s}$ into ρ chunks $\mathbf{s}_0,\ldots,\mathbf{s}_{\rho-1}$, each of which is ρ-bit long[4]. Next, we commit to each chunk, obtaining ρ commitments: $\mathsf{com}_0,\ldots,\mathsf{com}_{\rho-1}$. Note that $\mathsf{idec}_{\mathsf{P}}(g(i)) = (g_{i,0},\ldots,g_{i,\delta_{\mathsf{P}}-1})^\top$ is the binary representation of $g(i)$. Let $a_{g,i}$ and $b_{g,i}$ be the integers whose binary representation are the first $\delta_{\mathsf{P}}/2$ bits and second $\delta_{\mathsf{P}}/2$ bits of $\mathsf{idec}_{\mathsf{P}}(g(i))$, respectively. Then the bit $s_{g(i)}$ is committed in the $a_{g,i}$-th "bucket", i.e., $\mathsf{com}_{a_{g,i}}$, and it is the $b_{g,i}$-th bit (the index starts from 0) within this bucket. To search $s_{g(i)}$, we then prove the knowledge of a bit y_i satisfying the statement "y_i is the $b_{g,i}$-th bit within the $a_{g,i}$-th bucket". Due to the correctness of the proof system, $y_i = s_{g(i)}$. Said otherwise, we can provably perform a "bucket" search for $y_i = s_{g(i)}$.

We stress that the above retrieval process would not have protected P completely if m were not committed. This is because if one sees a bit of m used in computing (2), then some partial information about P could be leaked. Thus, we commit to m even though it is public. To show the publicity of m, we output the commitment randomnesses in the final signature Σ.

Proving in ZK that $t = F(P,\mathbf{x})$. Another challenge we have to tackle is to show the correctness of disclosing value $t = F(P,\mathbf{x})$, where F is a multivariate quadratic function – as mentioned above. Although this task is not as sophisticated as the one for hidden circuits, it also requires several non-trivial steps, among which is a sub-protocol for demonstrating the well-formedness of a Knonecker product $\mathbf{b}\otimes_{\mathrm{Kron}}\mathbf{b}$, where the bits of $\mathbf{b}$ simultaneously satisfy various other relations, e.g., they were hashed, signed and committed.

4.2 Scheme Description

We now describe our lattice-based BAPS scheme in detail.

$\mathsf{Setup}(1^\lambda)$**:** Given the security parameter 1^λ, it generates parameters as follows.

- Let $n=\mathcal{O}(\lambda)$ be a power of 2, let $q=3^k$, and $f(X)=X^n+1$ be a irreducible polynomial. Define rings $R=\mathbb{Z}[X]/(f(X))$ and $R_q=R/(qR)$.
- Let $k_1=k_1(\lambda)$, $k_2'=k_2'(\lambda)$, $k_3=k_3(\lambda)$ be positive integers. The message space is $\mathcal{M}=\{0,1\}^{k_1}$, the attribute space is $\mathcal{X}=\{0,1\}^{k_2}$ with $k_2=n\cdot k_2'$, and the witness space is $\mathcal{W}=\{0,1\}^{k_3}$. Define $K=k_1+k_2+k_3$.
- Lets $\mathcal{P}=\{P:\{0,1\}^K\to\{0,1\}\}$, where each P is a Boolean circuit containing N NAND gates, and is uniquely determined by two functions $g,h:[0,N-1]\to[0,K+N-2]$. Let $\delta_{\mathsf{P}}=\lfloor\log(K+N-2)\rfloor+1$.

[4] For simplicity, assume that $K+N-1=2^{\delta_{\mathsf{P}}}$ for an even integer δ_{P}. Then $\rho=2^{\delta_{\mathsf{P}}/2}$.

- Let $m_{\mathsf{P}} = \lceil (2N\delta_{\mathsf{P}}/n) \rceil$. Generate a random matrix $\mathbf{A}_{\mathsf{hp}} \xleftarrow{\$} (R_q)^{1 \times m_{\mathsf{P}}}$, which determines a hash function in the family $\mathcal{H}_{m_{\mathsf{P}}}$ and which will be used to hash the description of policies.
- Let $\kappa = \omega(\log \lambda)$ and $\mathcal{H}_{\mathsf{FS}} : \{0,1\}^* \to \{0,1\}^{\kappa}$ be a collision-resistant hash function, which will be modeled as a random oracle in the Fiat-Shamir transformation [21].
- Let CMT be a computationally binding and statistically hiding string commitment scheme (adapted from [30], also recalled in Appendix B.3), that will be used in our zero-knowledge argument system.
- Initialize $S_{\mathsf{X}} := 0, S_{\mathsf{P}} := 0$. Set $c, \alpha_0, d, c_1, \ldots, c_d, m, \overline{m}, \ell$ and generate a verification-signing key-pair $(\mathsf{vk}_{\mathsf{P}}, \mathsf{msk}_{\mathsf{P}})$ for the DM signature scheme (recalled in Appendix B.4). Denote vk_{P} as

$$\mathbf{A}_{\mathsf{P}}, \mathbf{F}_{\mathsf{P},0} \in (R_q)^{1 \times \overline{m}}, \{\mathbf{A}_{\mathsf{P},[i]}\}_{i=0}^{d} \in (R_q^{1 \times k})^{d+1}, \mathbf{F}_{\mathsf{P}}, \mathbf{F}_{\mathsf{P},1} \in (R_q)^{1 \times \ell}, u_{\mathsf{P}} \in R_q,$$

  and signing key $\mathsf{msk}_{\mathsf{P}}$ as $\mathbf{R}_{\mathsf{P}} \in (R_q)^{m \times k}$. Looking ahead, this key pair is used to sign hashes of the description of policies.
- We also generate another verification key and signing key pair to sign the attributes. Let vk_{X} be

$$\mathbf{A}_{\mathsf{X}}, \mathbf{F}_{\mathsf{X},0} \in (R_q)^{1 \times \overline{m}}, \{\mathbf{A}_{\mathsf{X},[i]}\}_{i=0}^{d} \in (R_q^{1 \times k})^{d+1}, \mathbf{F}_{\mathsf{X}}, \mathbf{F}_{\mathsf{X},1} \in (R_q)^{1 \times \ell}, u_{\mathsf{X}} \in R_q,$$

  and its corresponding signing key $\mathsf{msk}_{\mathsf{X}}$ as $\mathbf{R}_{\mathsf{X}} \in (R_q)^{m \times k}$.
- For simplicity, let us assume that $K + N - 1$ is a perfect square and let $\rho = \sqrt{K + N - 1}$ be its square root. We also assume that ρ is a multiple of n such that $\{0,1\}^{\rho} \subset (R_q)^{\rho'}$ with $\rho' = \frac{\rho}{n}$. Sample $\mathbf{A}_0 \xleftarrow{\$} (R_q)^{1 \times \rho'}$ and $\mathbf{A}_1 \xleftarrow{\$} (R_q)^{1 \times m}$ from the commitment scheme $\mathsf{CMT}_{\rho',m}$. Here $\mathbf{A}_0, \mathbf{A}_1$ are used to commit ρ-bit strings.
- Sample $\mathbf{A}_c \xleftarrow{\$} (R_q)^{1 \times \ell}$. This matrix $\mathbf{A}_c$ together with $\mathbf{A}_1$ will be used to commit to $n\ell$-bit strings.

Set the master policy key as $\mathsf{msk}_{\mathsf{P}} = \mathbf{R}_{\mathsf{P}} \in (R_q)^{m \times k}$ and master attribute key as $\mathsf{msk}_{\mathsf{X}} = \mathbf{R}_{\mathsf{X}} \in (R_q)^{m \times k}$.

Attribute-lss$(\mathsf{msk}_{\mathsf{X}}, \mathbf{x})$: Given $\mathsf{msk}_{\mathsf{X}}$ and an attribute $\mathbf{x} \in \{0,1\}^{nk_2'} \subset (R_q)^{k_2'}$, this algorithm computes the tag $t_{\mathbf{x}} = (t_0, \ldots, t_{c_d-1})^{\top} \in \mathcal{T}_d$ such that $S_{\mathsf{X}} = \sum_{j=0}^{c_d-1} 2^{c_d-1-j} t_j$ and then updates S_{X} to $S_{\mathsf{X}}+1$. It then runs the DM signing algorithm, obtaining a signature $\mathsf{sk}_{\mathbf{x}} = (t_{\mathbf{x}}, \mathbf{r}_{\mathbf{x}}, \mathbf{v}_{\mathbf{x}}) \in \{0,1\}^{c_d} \times R^{\overline{m}} \times R^{\overline{m}+k}$ such that:

$$\begin{cases} [\ \mathbf{A}_{\mathsf{X}} \ | \ \mathbf{A}_{\mathsf{X},[0]} + \sum_{j=1}^{d} \mathbf{A}_{\mathsf{X},[i]} \cdot t_{\mathbf{x},[i]}\ ] \cdot \mathbf{v}_{\mathbf{x}} = y_{\mathbf{x}}; \\ y_{\mathbf{x}} = u_{\mathsf{X}} + \mathbf{F}_{\mathsf{X}} \cdot \mathsf{rdec}(\ \mathbf{F}_{\mathsf{X},0} \cdot \mathbf{r}_{\mathbf{x}} + \mathbf{F}_{\mathsf{X},1} \cdot \mathbf{x}\ ); \\ \|\mathbf{r}_{\mathbf{x}}\|_{\infty} \leq \beta; \quad \|\mathbf{v}_{\mathbf{x}}\|_{\infty} \leq \beta. \end{cases} \tag{3}$$

Here, rdec is a function that decomposes a ring vector to a vector of appropriate length over $\{-1, 0, 1\}$. (See Appendix B.2 for the formal description.)

$\mathsf{Policy\text{-}Iss}(\mathsf{msk}_\mathsf{P}, P)$**:** Given $\mathsf{msk}_\mathsf{P} = \mathbf{R}_\mathsf{P}$ and a policy P, this algorithm generates certificate Cert_P in the following manner.

- Let the binary representation of the policy P be $\mathbf{z}_P \in \{0,1\}^{2N\delta_\mathsf{P}} \subset (R_q)^{m_\mathsf{P}}$, as defined in (1).
- Compute a hash of policy P as $h_P = \mathbf{A}_\mathsf{hp} \cdot \mathbf{z}_P \in R_q$.
- Use the key $\mathbf{R}_\mathsf{P}$ to generate a signature $\mathsf{Cert}_P = (t_P, \mathbf{r}_P, \mathbf{v}_P) \in \{0,1\}^{c_d} \times R^{\overline{m}} \times R^{\overline{m}+k}$ on h_P such that:

$$\begin{cases} [\ \mathbf{A}_\mathsf{P}\ \mid\ \mathbf{A}_{\mathsf{P},[0]} + \sum_{j=1}^{d} \mathbf{A}_{\mathsf{P},[i]} \cdot t_{P,[i]}\ ] \cdot \mathbf{v}_P = y_P; \\ y_P = u_\mathsf{P} + \mathbf{F}_\mathsf{P} \cdot \mathsf{rdec}(\ \mathbf{F}_{\mathsf{P},0} \cdot \mathbf{r}_P + \mathbf{F}_{\mathsf{P},1} \cdot \mathbf{h}_P\ ); \\ \mathbf{h}_P = \mathsf{rdec}(\ \mathbf{A}_\mathsf{hp} \cdot \mathbf{z}_P\ ) \in \{-1,0,1\}^{n\ell}; \\ \|\mathbf{r}_P\|_\infty \leq \beta;\ \ \|\mathbf{v}_P\|_\infty \leq \beta. \end{cases} \tag{4}$$

$\mathsf{Sign}(\mathbf{x}, \mathsf{sk}_\mathbf{x}, m, w, P, \mathsf{Cert}_P)$**:** If $P(m, \mathbf{x}, w) = 0$, the signing algorithm returns $\perp$. Otherwise, proceed as follows.

Let $m \in \{0,1\}^{k_1}$, $\mathbf{x} \in \{0,1\}^{nk_2'} \subset (R_q)^{k_2'}$, $w \in \{0,1\}^{k_3}$. Parse $\mathsf{sk}_\mathbf{x} = (t_\mathbf{x}, \mathbf{r}_\mathbf{x}, \mathbf{v}_\mathbf{x})$ and $\mathsf{Cert}_P = (t_P, \mathbf{r}_P, \mathbf{v}_P)$.

- First, we rename some inputs to facilitate the presentation. Denote $m = (s_0, \ldots, s_{k_1-1})^\top \in \{0,1\}^{k_1}$, $\mathbf{x} = (s_{k_1}, \ldots, s_{k_1+k_2-1})^\top \in \{0,1\}^{k_2}$, and $w = (s_{k_1+k_2}, \ldots, s_{K-1})^\top \in \{0,1\}^{k_3}$. Let the intermediate wires in the circuit P be $s_K, \ldots, s_{K+N-2}$ and the output wire be s_{K+N-1}. Recall that we assume $K + N - 1 = \rho^2$.
- Next, we will divide the secret bits of $\mathbf{s} = (s_0, \ldots, s_{K+N-2}) \in \{0,1\}^{\rho^2}$ into ρ-bit chunks and commit to each of the ρ chunks. To do so, for $i \in [0, \rho-1]$, let $\mathbf{s}_i = (s_{i\cdot\rho}, s_{i\cdot\rho+1}, \ldots, s_{i\cdot\rho+\rho-1})^\top$, sample randomness $\mathbf{r}_{\mathsf{com},i} \xleftarrow{\$} \{0,1\}^{nm}$ and compute $\mathsf{com}_i = \mathbf{A}_0 \cdot \mathbf{s}_i + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},i} \in R_q$. Without loss of generality, we assume $k_1 = \mu_1\rho$, $k_2 = \mu_2\rho$ for some integers μ_1, μ_2. Thus, $\mathsf{com}_0, \ldots, \mathsf{com}_{\mu_1-1}$ are commitments of the message m and $\mathsf{com}_{\mu_1}, \ldots, \mathsf{com}_{\mu_1+\mu_2-1}$ are commitments of $\mathbf{x}$. It is worth noting that we still commit to m even though m is public. This is essential in proving $P(m, \mathbf{x}, w) = 1$ without revealing P. To demonstrate that m is public, we will reveal the underlying commitment randomness in the resulting signature.
- Third, we will commit to the hash of the policy description. Let $\mathbf{z}_P \subseteq (R_q)^{m_\mathsf{P}}$ be the representation of P. Compute $h_P = \mathbf{A}_\mathsf{hp} \cdot \mathbf{z}_P \in R_q$ and $\mathbf{h}_P = \mathsf{rdec}(h_P) \in \{-1,0,1\}^{n\ell}$. We then sample randomness $\mathbf{r}_{\mathsf{com},P} \xleftarrow{\$} \{0,1\}^{nm}$ and let $\mathsf{com}_P = \mathbf{A}_c \cdot \mathbf{h}_P + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},P}$ be a commitment of P.

For notational purposes, define

$$\begin{aligned} \mathbf{s}_{else} &= (s_{k_1+k_2}, s_{k_1+k_2+1}, \ldots, s_{K+N-2})^\top \in \{0,1\}^{K+N-1-k_1-k_2}, \\ \mathsf{com}_m &= (\mathsf{com}_0^\top \mid \cdots \mid \mathsf{com}_{\mu_1-1}^\top)^\top \in R_q^{\mu_1}, \\ \mathsf{com}_\mathbf{x} &= (\mathsf{com}_{\mu_1}^\top \mid \cdots \mid \mathsf{com}_{\mu_1+\mu_2-1}^\top)^\top \in R_q^{\mu_2}, \\ \mathsf{com}_{else} &= (\mathsf{com}_{\mu_1+\mu_2}^\top \mid \cdots \mid \mathsf{com}_{\rho-1}^\top)^\top \in R_q^{\rho-\mu_1-\mu_2}, \end{aligned}$$

$$\mathbf{r}_{\mathsf{com},m} = (\mathbf{r}_{\mathsf{com},0}^\top \mid \cdots \mid \mathbf{r}_{\mathsf{com},\mu_1-1}^\top)^\top \in \{0,1\}^{nm\mu_1},$$
$$\mathbf{r}_{\mathsf{com},\mathbf{x}} = (\mathbf{r}_{\mathsf{com},\mu_1}^\top \mid \cdots \mid \mathbf{r}_{\mathsf{com},\mu_1+\mu_2-1}^\top)^\top \in \{0,1\}^{nm\mu_2},$$
$$\mathbf{r}_{\mathsf{com},else} = (\mathbf{r}_{\mathsf{com},\mu_1+\mu_2}^\top \mid \cdots \mid \mathbf{r}_{\mathsf{com},\rho-1}^\top)^\top \in \{0,1\}^{nm(\rho-\mu_1-\mu_2)}.$$

Once the commitment process is done, this algorithm then generates a NIZK argument demonstrating the knowledge of

$$\eta = (\mathbf{x}, \mathsf{sk}_\mathbf{x}, \mathbf{z}_P, \mathbf{h}_P, \mathsf{Cert}_P, w, s_K, \ldots, s_{K+N-2}, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},else}, \mathbf{r}_{\mathsf{com},P})$$

such that the following conditions hold.

(a) The message is signable with respect to policy P, attribute $\mathbf{x}$, and witness w, i.e., $P(m, \mathbf{x}, w) = 1$, or equivalently, equations (2) hold.

(b) $\mathsf{com}_\mathbf{x}$ and com_{else} are valid commitments to $\mathbf{x}$ and $\mathbf{s}_{else}$ with randomnesses $\mathbf{r}_{\mathsf{com},\mathbf{x}}$ and $\mathbf{r}_{\mathsf{com},else}$, respectively. In addition, com_P is a valid commitment to $\mathbf{h}_P$ with randomness $\mathbf{r}_{\mathsf{com},P}$. In other words, the following equations are satisfied.

$$\begin{cases} \mathsf{com}_i = \mathbf{A}_0 \cdot \mathbf{s}_i + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},i}, & \forall i \in [\mu_1, \rho - 1] \\ \mathsf{com}_P = \mathbf{A}_c \cdot \mathbf{h}_P + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},P}. \end{cases} \tag{5}$$

(c) The signer owns a valid signing key $\mathsf{sk}_\mathbf{x}$ for attribute $\mathbf{x}$, i.e., $(\mathbf{x}, \mathbf{sk}_\mathbf{x})$ satisfies (3).

(d) The signer owns a valid certificate Cert_P for policy P, i.e., $(\mathbf{z}_P, \mathbf{h}_P, \mathsf{Cert}_P)$ satisfies (4).

This is done by running the argument system described in Section 4.4. The protocol is then repeated $\kappa = \omega(\log \lambda)$ times to achieve negligible soundness error and made non-interactive via the Fiat-Shamir heuristic [21]. Let the resultant proof be

$$\mathit{\Pi} \;=\; (\; \{\mathsf{COM}_i\}_{i=0}^{\kappa-1},\; \{\mathsf{CH}_i\}_{i=0}^{\kappa-1},\; \{\mathsf{RSP}_i\}_{i=0}^{\kappa-1}\;), \tag{6}$$

where $(\mathsf{CH}_0, \ldots, \mathsf{CH}_{\kappa-1})^\top = \mathcal{H}_{\mathsf{FS}}(\{\mathsf{COM}_i\}_{i=0}^{\kappa-1},\ \xi)$ and ξ is of the form

$$(\; m,\; \mathsf{vk}_\mathsf{X},\; \mathsf{vk}_\mathsf{P},\; \mathbf{A}_{\mathsf{hp}},\; \mathbf{A}_c,\; \mathbf{A}_0,\; \mathbf{A}_1,\; \mathsf{com}_m,\; \mathsf{com}_\mathbf{x},\; \mathsf{com}_{else},\; \mathsf{com}_P,\; \mathbf{r}_{\mathsf{com},m}\;). \tag{7}$$

Return the signature as $\mathit{\Sigma} = (\mathsf{com}_m,\ \mathsf{com}_\mathbf{x},\ \mathsf{com}_{else},\ \mathsf{com}_P,\ \mathbf{r}_{\mathsf{com},m},\ \mathit{\Pi})$ and the clue as $\mathsf{clue} = (\mathbf{x},\ P,\ \mathbf{r}_{\mathsf{com},\mathbf{x}},\ \mathbf{r}_{\mathsf{com},P})$. Note that all the commitments are needed for the verification of $\mathit{\Pi}$. Also, as m is public, the randomnesses $\mathbf{r}_{\mathsf{com},m}$ are included in $\mathit{\Sigma}$.

$\mathsf{Verify}(m, \mathit{\Sigma})$**:** This algorithm verifies the validity of the signature $\mathit{\Sigma}$ as follows.

- Let $m = (s_0, \ldots, s_{k_1-1})^\top \in \{0,1\}^{k_1}$ and $\mathbf{s}_i = (s_{i\cdot\rho}, s_{i\cdot\rho+1}, \ldots, s_{i\cdot\rho+\rho-1})^\top$ for $i \in [0, \mu_1 - 1]$.
- Parse $\mathit{\Sigma} = (\mathsf{com}_m, \mathsf{com}_\mathbf{x}, \mathsf{com}_{else}, \mathsf{com}_P, \mathbf{r}_{\mathsf{com},m}, \mathit{\Pi})$, where $\mathit{\Pi}$ is as in (6), $\mathsf{com}_m = (\mathsf{com}_0^\top \mid \cdots \mid \mathsf{com}_{\mu_1-1}^\top)^\top$, and $\mathbf{r}_{\mathsf{com},m} = (\mathbf{r}_{\mathsf{com},0}^\top \mid \cdots \mid \mathbf{r}_{\mathsf{com},\mu_1-1}^\top)^\top$.
- Return 1 if and only if the following conditions are satisfied.

(i) $(\mathsf{CH}_0, \ldots, \mathsf{CH}_{\kappa-1})^\top = \mathcal{H}_{\mathsf{FS}}(\{\mathsf{COM}_i\}_{i=0}^{\kappa-1},\ \xi)$ with ξ as in (7).

(ii) For all $i \in [0, \mu_1 - 1]$, $\mathsf{com}_i = \mathbf{A}_0 \cdot \mathbf{s}_i + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},i}$.

(iii) For all $j \in [0, \kappa - 1]$, response RSP_j is valid with respect to commitment COM_j and the challenge CH_j.

$\mathsf{Disclose}(m, \Sigma, \mathsf{clue}, F)$**:** This algorithm computes a testimony-attestation pair as follows.

– Let $\mathbf{x} = (s_{k_1}, \ldots, s_{k_1+k_2-1})^\top \in \{0,1\}^{k_2}$, and for $i \in [\mu_1, \mu_1 + \mu_2 - 1]$, let $\mathbf{s}_i = (s_{i\cdot\rho}, s_{i\cdot\rho+1}, \ldots, s_{i\cdot\rho+\rho-1})^\top$.

– Parse $\Sigma = (\mathsf{com}_m, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_{else}, \mathsf{com}_P, \mathbf{r}_{\mathsf{com},m}, \Pi)$, where Π is as in (6), $\mathsf{com}_{\mathbf{x}} = (\mathsf{com}_{\mu_1}^\top | \cdots | \mathsf{com}_{\mu_1+\mu_2-1}^\top)^\top$, $\mathbf{r}_{\mathsf{com},\mathbf{x}} = (\mathbf{r}_{\mathsf{com},\mu_1}^\top | \cdots | \mathbf{r}_{\mathsf{com},\mu_1+\mu_2-1}^\top)^\top$. Return 0 if $\mathsf{Verify}(m, \Sigma) = 0$.

– Parse $\mathsf{clue} = (\mathbf{x}, P, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},P})$.

– Let the binary representation of P be $\mathbf{z}_P \in \{0,1\}^{2N\delta_{\mathsf{P}}}$, and F be determined by two matrices $\mathbf{G}_1 \in \{0,1\}^{\bar{k}\times\bar{k}^2}$ and $\mathbf{G}_2 \in \{0,1\}^{\bar{k}\times\bar{k}}$. Next, denote $\mathbf{b} = (\mathbf{z}_P^\top \mid \mathbf{x}^\top)^\top$ and then compute the testimony as

$$t = \mathbf{G}_1 \cdot (\mathbf{b} \otimes_{\mathrm{Kron}} \mathbf{b}) + \mathbf{G}_2 \cdot \mathbf{b} \bmod 2.$$

– Generate a NIZK argument of knowledge of clue such that the following conditions hold.

(i) The testimony t is honestly computed, i.e., for $\mathbf{b} = (\mathbf{z}_P^\top \mid \mathbf{x}^\top)^\top$,

$$t = F(P, \mathbf{x}) = \mathbf{G}_1 \cdot (\mathbf{b} \otimes_{\mathrm{Kron}} \mathbf{b}) + \mathbf{G}_2 \cdot \mathbf{b} \bmod 2.$$

(ii) $\mathsf{com}_{\mathbf{x}}$ and com_P are valid commitments of $\mathbf{x}$ and P, i.e.,

$$\begin{aligned} \mathsf{com}_P &= \mathbf{A}_c \cdot \mathbf{h}_P + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},P}, \quad \text{with} \quad \mathbf{h}_P = \mathsf{rdec}(\mathbf{A}_{\mathsf{hp}} \cdot \mathbf{z}_P); \\ \mathsf{com}_i &= \mathbf{A}_0 \cdot \mathbf{s}_i + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},i}, \quad \forall\ i \in [\mu_1, \mu_1 + \mu_2 - 1]. \end{aligned}$$

This is done by running the argument system in Appendix F on public input $\xi_{\mathsf{D}} = (\mathbf{G}_1, \mathbf{G}_2, t, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_P, \mathbf{A}_{\mathsf{hp}}, \mathbf{A}_c, \mathbf{A}_0, \mathbf{A}_1)$ and secret input clue. The protocol is conducted $\kappa = \omega(\log\lambda)$ times in parallel to obtain negligible soundness error and then made non-interactive via the Fiat-Shamir heuristic [21]. The resultant proof a is a triple of a form

$$a = \left(\{\mathsf{COM}_{\mathsf{D},i}\}_{i=0}^{\kappa-1},\ \{\mathsf{CH}_{\mathsf{D},i}\}_{i=0}^{\kappa-1},\ \{\mathsf{RSP}_{\mathsf{D},i}\}_{i=0}^{\kappa-1}\right), \tag{8}$$

where $(\mathsf{CH}_{\mathsf{D},0}, \ldots, \mathsf{CH}_{\mathsf{D},\kappa-1})^\top = \mathcal{H}_{\mathsf{FS}}(\{\mathsf{COM}_{\mathsf{D},i}\}_{i=0}^{\kappa-1}, \xi_{\mathsf{D}})$.

– Return t and a.

$\mathsf{Judge}(m, \Sigma, F, t, a)$**:** If $\mathsf{Verify}(m, \Sigma) = 0$, return 0. Otherwise, it proceeds to verify the validity of a, which is quite similar to the verification of Π. Return 1 if a is valid and 0 otherwise.

4.3 Analyses

Correctness. The correctness of our construction relies on two facts: (i) The employed Ducas-Micciancio signature scheme is correct with overwhelming probability; (ii) The two underlying zero-knowledge argument systems are perfectly correct. Therefore, for any policy P, any attribute $\mathbf{x}$, any message m such that $P(m, \mathbf{x}, w) = 1$ for some witness w, if a signer owns honestly-generated signing key for $\mathbf{x}$ and certificate for P, and signs the message faithfully as in the Sign algorithm, then the Verify algorithm outputs 1 with overwhelming probability. In addition, clue contains P, $\mathbf{x}$, and randomnesses for committing them. Therefore, the testimony can be correctly computed as $t = F(P, \mathbf{x})$ and the attestation a, honestly generated with the knowledge of clue, will pass the Judge algorithm.

Asymptotic Efficiency. We now analyze the efficiency of our construction with respect to the security parameter λ, the number K of inputs in P, and the number N of NAND gates in P.

- The public parameters are dominated by $\mathsf{vk}_{\mathsf{X}}, \mathsf{vk}_{\mathsf{P}}, \mathbf{A}_{\mathsf{hp}}, \mathbf{A}_c, \mathbf{A}_0, \mathbf{A}_1$, which are of $\mathcal{O}\big(\ N \cdot \log(K+N) \cdot \log \lambda + \lambda \cdot (\log \lambda)^2\ \big)$ bits.
- The $\mathsf{msk}_{\mathsf{X}}$ and $\mathsf{msk}_{\mathsf{P}}$ have bit size $\mathcal{O}(\lambda \cdot (\log \lambda)^3)$.
- The signature size is dominated by the size of Π, which has bit size $\kappa \cdot \mathcal{O}\big(\ L_1 \cdot \log q + L_{\mathsf{P}}\big) = \kappa \cdot \mathcal{O}\big(\ \lambda \cdot (\log \lambda)^5 + N\sqrt{K+N} \log \lambda + N \cdot \lambda \cdot (\log \lambda)^2 + (K+N) \log \lambda + \lambda\sqrt{K+N}(\log \lambda)^2\ \big)$.
- The size of attestation is $\kappa \cdot \mathcal{O}\big(\ L_2 \cdot \log q + L_{\mathsf{D}}\ \big) = \kappa \cdot \mathcal{O}\big(\ k_2 \cdot \log \lambda + \mu_2 \lambda \cdot (\log \lambda)^2 + N^2 \cdot \log^2(K+N) + k_2^2\ \big)$. (See Appendix F for details.)

Security. In the random oracle model, our construction satisfies simulatability and extractability. In particular, the construction is simulatable based on the facts that (i) the two underlying zero-knowledge argument systems are statistical zero knowledge and (ii) the commitment schemes $\mathsf{CMT}_{\rho',m}, \mathsf{CMT}_{\ell,m}$ are statistically hiding. Extractability relies on (i) the computational soundness of the two underlying ZKAoK systems, (ii) the computational binding property of $\mathsf{CMT}_{\rho',m}$ and $\mathsf{CMT}_{\ell,m}$, (iii) $\mathsf{EUF\text{-}CMA}$ security of the Ducas-Micciancio signature scheme, and (iv) the collision resistance of the hash function family $\mathcal{H}_{m_{\mathsf{P}}}$. We summarize the security in the following theorems and provide detailed proofs in Appendix G.

Theorem 3. *Assume that the two underlying zero-knowledge argument systems are statistical zero-knowledge and the commitment schemes* $\mathsf{CMT}_{\rho',m}, \mathsf{CMT}_{\ell,m}$ *are statistically hiding. Then the proposed BAPS scheme is simulatable.*

Theorem 4. *The proposed BAPS scheme is extractable if the two underlying argument systems are computationally knowledge sound, the two commitment schemes* $\mathsf{CMT}_{\rho',m}$, $\mathsf{CMT}_{\ell,m}$ *are computationally binding, the Ducas-Micciancio signature scheme is* $\mathsf{EUF\text{-}CMA}$ *secure, and the hash function family* $\mathcal{H}_{m_{\mathsf{P}}}$ *is collision-resistant.*

4.4 The Main Zero-Knowledge Argument of Knowledge

Let us now summarize the relation $\mathcal{R}_{\mathsf{S}}$ appearing in the Sign algorithm.

Public inputs ξ: $\mathbf{A}_{\mathsf{hp}}$, vk_{P}, vk_{X}, $\mathbf{A}_c$, $\mathbf{A}_0$, $\mathbf{A}_1$, $\mathsf{com}_0, \ldots, \mathsf{com}_{\rho-1}$, com_P, $\mathbf{r}_{\mathsf{com},0}$, $\ldots$, $\mathbf{r}_{\mathsf{com},\mu_1-1}$, m.

Secret inputs η: $\mathbf{z}_P \in \{0,1\}^{2N\delta_{\mathsf{P}}}$, $\mathbf{x} \in \{0,1\}^{k_2}$, $w \in \{0,1\}^{k_3}$, $s_K, \ldots, s_{K+N-2} \in \{0,1\}$, $\mathsf{sk}_{\mathbf{x}} = (t_{\mathbf{x}}, \mathbf{r}_{\mathbf{x}}, \mathbf{v}_{\mathbf{x}}) \in \{0,1\}^{c_d} \times (R)^{\overline{m}} \times (R)^{\overline{m}+k}$, $\mathsf{Cert}_P = (t_P, \mathbf{r}_P, \mathbf{v}_P) \in \{0,1\}^{c_d} \times (R)^{\overline{m}} \times (R)^{\overline{m}+k}$, $\mathbf{r}_{\mathsf{com},\mu_1}, \ldots, \mathbf{r}_{\mathsf{com},\rho-1}, \mathbf{r}_{\mathsf{com},P} \in \{0,1\}^{nm}$.

Prover's goal: Proving knowledge of the secret inputs η such that equations (2), (3), (4), and (5) are satisfied.

Proving knowledge of η such that substatements involving equations (3), (4), and (5) can be handled by previous techniques, such as [41,42]. For completeness, we present them in Appendix E.1-E.3. Specifically, we transform (3) to an equivalent form $\mathbf{M}_{\mathsf{X}} \cdot \mathbf{w}_{\mathsf{X}} = \widetilde{\mathbf{v}}_{\mathsf{X}} \bmod q$ for $\mathbf{w}_{\mathsf{X}} \in \mathsf{VALID}_{\mathsf{X}} \subset \{-1,0,1\}^{L_{\mathsf{X}}}$ with associated $\mathcal{S}_{\mathsf{X}}$ and $\{\Gamma_\phi : \phi \in \mathcal{S}_{\mathsf{X}}\}$ such that conditions in (18) are conformed. Similarly, we transform (4) and (5) to $\mathbf{M}_{\mathsf{cert}} \cdot \mathbf{w}_{\mathsf{cert}} = \mathbf{v}_{\mathsf{cert}} \bmod q$ for $\mathbf{w}_{\mathsf{cert}} \in \mathsf{VALID}_{\mathsf{cert}} \subset \{-1,0,1\}^{L_{\mathsf{cert}}}$ with associated $\mathcal{S}_{\mathsf{cert}}, \{\Gamma_\phi : \phi \in \mathcal{S}_{\mathsf{cert}}\}$, and $\mathbf{M}_{\mathsf{com}} \cdot \mathbf{w}_{\mathsf{com}} = \mathbf{v}_{\mathsf{com}}$ for $\mathbf{w}_{\mathsf{com}} \in \mathsf{VALID}_{\mathsf{com}} \subset \{-1,0,1\}^{L_{\mathsf{com}}}$ with associated $\mathcal{S}_{\mathsf{com}}, \{\Gamma_\phi : \phi \in \mathcal{S}_{\mathsf{com}}\}$, respectively. Here we focus on the substatement involving the equations (2).

To this end, the challenge is how to retrieve the values $s_{g(i)}, s_{h(i)}$ for $i \in [0, N-1]$ without revealing $g(i), h(i)$ or $s_{g(i)}, s_{h(i)}$. In constructing their oblivious transfer with access control, Libert et al. [35] encountered a similar issue. They have to retrieve $s_{\mathsf{var}(\theta)}$ without revealing $s_{\mathsf{var}(\theta)}$ or $\mathsf{var}(\theta)$, among a secret vector $\mathbf{s} = (s_0, \ldots, s_{\upsilon-1})$ with $\mathsf{var}(\theta) \in [0, \upsilon - 1]$. To solve this issue, they perform a dichotomic search on $\mathbf{s}$. The expected complexity for each retrieval is $\mathcal{O}(\log \upsilon \cdot \lambda \cdot \log \lambda)$.

If we use their dichotomic search to retrieve $s_{g(i)}, s_{h(i)}$, the complexity for a single search would be $\mathcal{O}(\log(K+N-1) \cdot \lambda \cdot \log \lambda)$. We, however, take a different approach to retrieve $s_{g(i)}, s_{h(i)}$, and the expected complexity for a single search is $\mathcal{O}(\sqrt{K+N-1} + \lambda \cdot \log \lambda)$. When $K+N$ is a small polynomial in λ, our approach is more efficient[5]. Let us now describe our strategies.

Recall that we commit to $\mathbf{s}_0, \ldots, \mathbf{s}_{\rho-1}$, each of which is a ρ-bit string. Let $K+N-1 = \rho^2 = 2^{\delta_{\mathsf{P}}}$ and $\mathsf{idec}_{\mathsf{P}}(g(i)) = (g_{i,0}, g_{i,1}, \ldots, g_{i,\frac{\delta_{\mathsf{P}}}{2}-1}, g_{i,\frac{\delta_{\mathsf{P}}}{2}}, \ldots, g_{i,\delta_{\mathsf{P}}-1})^\top$ be the binary representation of $g(i) \in [0, K-N-2]$. Define

$$\begin{cases} a_{g,i} = \sum_{j=0}^{\frac{\delta_{\mathsf{P}}}{2}-1} 2^{\frac{\delta_{\mathsf{P}}}{2}-1-j} \cdot g_{i,j} \in [0, \rho-1], \\ b_{g,i} = \sum_{j=\frac{\delta_{\mathsf{P}}}{2}}^{\delta_{\mathsf{P}}-1} 2^{\delta_{\mathsf{P}}-1-j} \cdot g_{i,j} \in [0, \rho-1]. \end{cases} \tag{9}$$

[5] If $\lambda = 2^8$, and $K+N \leq \lambda^3 = 2^{24}$, then our method yields better efficiency. Note that policies with $K+N \leq 2^{24} \approx 10^7$ captures many policies of cryptographic interests [29].

We have seen that $s_{g(i)}$ is the $b_{g,i}$-th bit among the ρ bits $\mathbf{s}_{a_{g,i}}$ committed in $\mathsf{com}_{a_{g,i}}$. Proving the statement that "y_i is the $b_{g,i}$-th bit within the $a_{g,i}$-th bucket", however, is a non-trivial task. To this end, we introduce extra secrets $\widetilde{\mathbf{s}}_i = (\widetilde{s}_{i,0}, \ldots, \widetilde{s}_{i,b_{g,i}}, \ldots, \widetilde{s}_{i,\rho-1})^\top$, $\widetilde{\mathsf{com}}_{g,i}$, and instead proving knowledge of $y_i, \widetilde{\mathbf{s}}_i, \widetilde{\mathsf{com}}_{g,i}$ satisfying the statement "y_i is the $b_{g,i}$-th bit among the ρ bits $\widetilde{\mathbf{s}}_i$ committed in $\widetilde{\mathsf{com}}_{g,i}$. The latter, i.e., $\widetilde{\mathsf{com}}_{g,i}$, is further the $a_{g,i}$-th commitment among commitments $\mathsf{com}_0, \ldots, \mathsf{com}_{a_{g,i}}, \ldots, \mathsf{com}_{\rho-1}$". The binding property of the commitment scheme guarantees that $\widetilde{\mathbf{s}}_i = \mathbf{s}_{a_{g,i}}$, and hence $y_i = s_{g(i)}$.

The next challenge we have to overcome is to prove that y_i is the $b_{g,i}$-th bit among the ρ bits $\widetilde{s}_{i,0}, \ldots, \widetilde{s}_{i,\rho-1}$, where $y_i, b_{g,i}$ as well as $\widetilde{s}_{i,0}, \ldots, \widetilde{s}_{i,\rho-1}$ are all secret. Inspired by the technique of proving a well-formed regular word by Nguyen et al. [49], we introduce intermediate secret bits $b_{i,0}, \ldots, b_{i,\rho-1}$ and prove that

$$y_i = b_{i,0} \cdot \widetilde{s}_{i,0} + b_{i,1} \cdot \widetilde{s}_{i,1} + \cdots + b_{i,\rho-1} \cdot \widetilde{s}_{i,\rho-1}.$$

Then $y_i = \widetilde{s}_{i,b_{g,i}}$ if and only if $(b_{i,0}, \ldots, b_{i,\rho-1})^\top$ is a regular word defined as $\Delta_{\frac{\delta_\mathsf{P}}{2}}(g_{i,\frac{\delta_\mathsf{P}}{2}}, \ldots, g_{i,\delta_\mathsf{P}-1})$[6].

Yet there is a subtle issue here, i.e., a malicious prover may not use the same $(b_{i,0}, \ldots, b_{i,\rho-1})^\top$ for computing y_i and for proving that it is a well-formed regular word. This can be solved, fortunately, by adding extra intermediate bits $b'_{i,0}, \ldots, b'_{i,\rho-1}$ and forcing them equal with an additional constraint:

$$(b_{i,0}, \ldots, b_{i,\rho-1})^\top = (b'_{i,0}, \ldots, b'_{i,\rho-1})^\top.$$

The same techniques can be used to prove the second half statement that $\widetilde{\mathsf{com}}_{g,i}$ is the $a_{g,i}$-th commitment among the ρ commitments $\mathsf{com}_0, \ldots, \mathsf{com}_{\rho-1}$.

Combining all together, to show that we have "bucket" searched $s_{g(i)}$ correctly, we need to argue knowledge of $\widetilde{\mathsf{com}}_{g,i} \in R_q$, $y_i \in \{0,1\}$, $(\widetilde{s}_{i,0}, \ldots, \widetilde{s}_{i,\rho-1})^\top \in \{0,1\}^\rho$, $\mathbf{r}_{g,i} \in \{0,1\}^{nm}$, $(a_{i,0}, \ldots, a_{i,\rho-1})^\top \in \{0,1\}^\rho$, $(b_{i,0}, \ldots, b_{i,\rho-1})^\top \in \{0,1\}^\rho$, $(b'_{i,0}, \ldots, b'_{i,\rho-1})^\top \in \{0,1\}^\rho$, $\mathsf{idec}_\mathsf{P}(g(i)) \in \{0,1\}^{\delta_\mathsf{P}}$ such that

$$\begin{cases} \widetilde{\mathsf{com}}_{g,i} = \mathbf{A}_0 \cdot (\widetilde{s}_{i,0}, \ldots, \widetilde{s}_{i,\rho-1})^\top + \mathbf{A}_1 \cdot \mathbf{r}_{g,i}; \\ \widetilde{\mathsf{com}}_{g,i} = [\mathsf{com}_0, \ldots, \mathsf{com}_{\rho-1}] \cdot (a_{i,0}, \ldots, a_{i,\rho-1})^\top; \\ \qquad \text{we show } (a_{i,0}, a_{i,1}, \ldots, a_{i,\rho-1}) = \Delta_{\frac{\delta_\mathsf{P}}{2}}(g_{i,0}, g_{i,1}, \ldots, g_{i,\frac{\delta_\mathsf{P}}{2}-1}); \\ y_i = b_{i,0} \cdot \widetilde{s}_{i,0} + b_{i,1} \cdot \widetilde{s}_{i,1} + \cdots + b_{i,\rho-1} \cdot \widetilde{s}_{i,\rho-1}; \\ (b_{i,0}, b_{i,1}, \ldots, b_{i,\rho-1}) - (b'_{i,0}, \ldots, b'_{i,\rho-1})^\top = \mathbf{0}^\rho; \\ \qquad \text{we show } (b'_{i,0}, \ldots, b'_{i,\rho-1})^\top = \Delta_{\frac{\delta_\mathsf{P}}{2}}(g_{i,\frac{\delta_\mathsf{P}}{2}}, \ldots, g_{i,\delta_\mathsf{P}-1}). \end{cases} \tag{10}$$

[6] Given $\mathbf{x} = (x_0, \ldots, x_{c-1}) \in \{0,1\}^c$, a regular word $\Delta_c(\mathbf{x}) \in \{0,1\}^{2^c}$ has the sole 1 entry at the $(\sum_{i=0}^{c-1} 2^{c-1-i} x_i)$-th position.

We emphasize that $(a_{i,0}, a_{i,1}, \ldots, a_{i,\rho-1})^\top = \Delta_{\frac{\delta_\mathsf{P}}{2}}(g_{i,0}, g_{i,1}, \ldots, g_{i,\frac{\delta_\mathsf{P}}{2}-1})$ ensures that $\widetilde{\mathsf{com}}_{g,i}$ is actually $\mathsf{com}_{a_{g,i}}$ while $(b'_{i,0}, \ldots, b'_{i,\rho-1})^\top = \Delta_{\frac{\delta_\mathsf{P}}{2}}(g_{i,\frac{\delta_\mathsf{P}}{2}}, \ldots, g_{i,\delta_\mathsf{P}-1})$ makes sure that y_i is the $b_{g,i}$-th bit committed in $\widetilde{\mathsf{com}}_{g,i}$.

Now, to prove that equations in (2) are satisfied, we need to prove that (i) the searches of $s_{g(0)}, s_{h(0)}, \ldots, s_{g(N-1)}, s_{h(N-1)}$ yield $y_0, z_0, \ldots, y_{N-1}, z_{N-1}$ and that (ii) the policy P evaluates to 1 if $y_0, z_0, \ldots, y_{N-1}, z_{N-1}$ are used, i.e., we need to show

$$\begin{cases} s_K \oplus y_0 \cdot z_0 = 1 \mod 2; \\ s_{K+1} \oplus y_1 \cdot z_1 = 1 \mod 2; \\ \ldots\ldots \\ s_{K+N-2} \oplus y_{N-2} \cdot z_{N-2} = 1 \mod 2; \\ y_{N-1} \cdot z_{N-1} = 0 \mod 2. \end{cases} \tag{11}$$

Dedicated Stern-like techniques like [41,42,35,49] can be used to prove statements involving equations (10)(11). Specifically, we transform (10) to an equivalent form $\mathbf{M}_{g,i} \cdot \mathbf{w}_{g,i} = \mathbf{v}_{g,i} \bmod q$ for $\mathbf{w}_{g,i} \in \mathsf{VALID}_{g,i} \subset \{0,1\}^{L_{g,i}}$ with associated $\mathcal{S}_{g,i}$ and $\{\Gamma_\phi : \phi \in \mathcal{S}_{g,i}\}$ such that the conditions in (18) are all satisfied. Similar transformations are employed to search $s_{h(i)}$ for $i \in [0, N-1]$.

Also, we are able to transform (11) to an equivalent form $\mathbf{M}_\mathsf{P} \cdot \mathbf{w}_\mathsf{P} = \mathbf{v}_\mathsf{P} \bmod 2$ for $\mathbf{w}_\mathsf{P} \in \mathsf{VALID}_\mathsf{P} \subset \{0,1\}^{L_\mathsf{P}}$ with associated $\mathcal{S}_\mathsf{P}$ and $\{\Gamma_\phi : \phi \in \mathcal{S}_\mathsf{P}\}$ such that the conditions in (18) are all fulfilled. For completeness, they are presented in Appendix E.4 and E.5.

Putting Pieces Altogether. At the final stage of the process, we connect the previous steps by combining them into

$$\mathbf{M}_1 \cdot \mathbf{w}_1 = \mathbf{v}_1 \mod q; \qquad \mathbf{M}_\mathsf{P} \cdot \mathbf{w}_\mathsf{P} = \mathbf{v}_\mathsf{P} \mod 2,$$

where $\mathbf{w}_1 \in \{-1, 0, 1\}^{L_1}$ with $L_1 = L_\mathsf{X} + L_\mathsf{cert} + L_\mathsf{com} + 2N \cdot L_{g,i}$. As a result, we can specify a suitable sets $\mathsf{VALID}_\mathsf{baps}$, $\mathcal{S}_\mathsf{baps}$ and permutation $\{\Gamma_\phi : \phi \in \mathcal{S}_\mathsf{baps}\}$, for which conditions (18) are satisfied. Details are in Appendix E.6.

At this point, our desired argument protocol between the prover and verifier works as follows. Given the public inputs ξ, both parties construct $\mathbf{M}_1, \mathbf{M}_\mathsf{P}$ and $\mathbf{v}_1, \mathbf{v}_\mathsf{P}$ as above. The prover additionally constructs $\mathbf{w}_1, \mathbf{w}_\mathsf{P}$. Next, they run the protocol described in **Table** 2. If the commitment scheme COM is statistically hiding and computationally binding, the resulting protocol is a statistical ZKAoK protocol with perfect completeness, soundness error 2/3, and communication cost

$$\begin{aligned} \mathcal{O}(L_1 \cdot \log q + L_\mathsf{P}) = \mathcal{O}\big(& \lambda \cdot (\log \lambda)^5 + N\sqrt{K+N} \log \lambda \\ & + N \cdot \lambda \cdot (\log \lambda)^2 + (K+N) \log \lambda + \lambda\sqrt{K+N} (\log \lambda)^2 \big). \end{aligned}$$

Acknowledgements. We thank the anonymous reviewers of ASIACRYPT 2023 for their helpful comments and suggestions. The work of Yanhong Xu

was supported in part by the National Key Research and Development Program under Grant 2022YFA1004900. Willy Susilo was partially supported by the Australian Research Council (ARC) Discovery project (DP200100144) and the Australian Laureate Fellowship (FL230100033).

References

1. Attrapadung, N., Hanaoka, G., Yamada, S.: Conversions among several classes of predicate encryption and applications to ABE with various compactness tradeoffs. In: ASIACRYPT 2015. LNCS, vol. 9452, pp. 575–601. Springer (2015)
2. Bellare, M., Fuchsbauer, G.: Policy-based signatures. In: PKC 2014. LNCS, vol. 8383, pp. 520–537. Springer (2014)
3. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In: EUROCRYPT 2003. LNCS, vol. 2656, pp. 614–629. Springer (2003)
4. Bender, A., Katz, J., Morselli, R.: Ring signatures: Stronger definitions, and constructions without random oracles. J. Cryptol. **22**(1), 114–138 (2009)
5. Benhamouda, F., Camenisch, J., Krenn, S., Lyubashevsky, V., Neven, G.: Better zero-knowledge proofs for lattice encryption and their application to group signatures. In: ASIACRYPT 2014. LNCS, vol. 8873, pp. 551–572. Springer (2014)
6. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer (2011)
7. Bootle, J., Lyubashevsky, V., Nguyen, N.K., Sorniotti, A.: A framework for practical anonymous credentials from lattices. In: CRYPTO 2023. LNCS, vol. 14082, pp. 384–417. Springer (2023)
8. Bootle, J., Lyubashevsky, V., Seiler, G.: Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In: CRYPTO 2019. LNCS, vol. 11692, pp. 176–202. Springer (2019)
9. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: PKC 2014. LNCS, vol. 8383, pp. 501–519. Springer (2014)
10. Brickell, E.F., Pointcheval, D., Vaudenay, S., Yung, M.: Design validations for discrete logarithm based signature schemes. In: PKC 2000. LNCS, vol. 1751, pp. 276–292. Springer (2000)
11. Camenisch, J., Hohenberger, S., Lysyanskaya, A.: Balancing accountability and privacy using e-cash (extended abstract). In: SCN 2006. LNCS, vol. 4116, pp. 141–155. Springer (2006)
12. Chase, M., Lysyanskaya, A.: On signatures of knowledge. In: CRYPTO 2006. LNCS, vol. 4117, pp. 78–96. Springer (2006)
13. Chaum, D., van Heyst, E.: Group signatures. In: EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer (1991)
14. Cheng, S., Nguyen, K., Wang, H.: Policy-based signature scheme from lattices. Des. Codes Cryptogr. **81**(1), 43–74 (2016)
15. Ducas, L., Micciancio, D.: Improved short lattice signatures in the standard model. In: CRYPTO 2014. LNCS, vol. 8616, pp. 335–352. Springer (2014), `http://eprint.iacr.org/2014/495`
16. Esgin, M.F., Nguyen, N.K., Seiler, G.: Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. In: ASIACRYPT 2020. LNCS, vol. 12492, pp. 259–288. Springer (2020)

17. Esgin, M.F., Steinfeld, R., Liu, D., Ruj, S.: Efficient hybrid exact/relaxed lattice proofs and applications to rounding and vrfs. In: CRYPTO 2023. LNCS, vol. 14085, pp. 484–517. Springer (2023)
18. Esgin, M.F., Steinfeld, R., Liu, J.K., Liu, D.: Lattice-based zero-knowledge proofs: New techniques for shorter and faster constructions and applications. In: CRYPTO 2019. LNCS, vol. 11692, pp. 115–146. Springer (2019)
19. Ezerman, M.F., Lee, H.T., Ling, S., Nguyen, K., Wang, H.: Provably secure group signature schemes from code-based assumptions. IEEE Trans. Inf. Theory **66**(9), 5754–5773 (2020)
20. Faust, S., Kohlweiss, M., Marson, G.A., Venturi, D.: On the non-malleability of the fiat-shamir transform. In: INDOCRYPT 2012. LNCS, vol. 7668, pp. 60–79. Springer (2012)
21. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer (1986)
22. Fujisaki, E., Suzuki, K.: Traceable ring signature. In: PKC 2007. LNCS, vol. 4450, pp. 181–200. Springer (2007)
23. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM J. Comput. **18**(1), 186–208 (1989)
24. Groth, J.: Simulation-sound nizk proofs for a practical language and constant size group signatures. In: ASIACRYPT 2006. pp. 444–459. LNCS, Springer (2006)
25. Groth, J., Ostrovsky, R., Sahai, A.: Perfect non-interactive zero knowledge for NP. In: EUROCRYPT 2006. LNCS, vol. 4004, pp. 339–358. Springer (2006)
26. Jain, A., Krenn, S., Pietrzak, K., Tentes, A.: Commitments and efficient zero-knowledge proofs from learning parity with noise. In: ASIACRYPT 2012. LNCS, vol. 7658, pp. 663–680. Springer (2012)
27. Jeudy, C., Roux-Langlois, A., Sanders, O.: Lattice signature with efficient protocols, application to anonymous credentials. In: CRYPTO 2023. LNCS, vol. 14082, pp. 351–383. Springer (2023)
28. Katsumata, S., Yamada, S.: Group signatures without NIZK: from lattices in the standard model. In: EUROCRYPT 2019. LNCS, vol. 11478, pp. 312–344. Springer (2019)
29. Katz, J., Kolesnikov, V., Wang, X.: Improved non-interactive zero knowledge with applications to post-quantum signatures. In: CCS 2018. pp. 525–537. ACM (2018)
30. Kawachi, A., Tanaka, K., Xagawa, K.: Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In: ASIACRYPT 2008. LNCS, vol. 5350, pp. 372–389. Springer (2008)
31. Kiayias, A., Tsiounis, Y., Yung, M.: Traceable signatures. In: EUROCRYPT 2004. LNCS, vol. 3027, pp. 571–589. Springer (2004)
32. Kohlweiss, M., Miers, I.: Accountable metadata-hiding escrow: A group signature case study. Proc. Priv. Enhancing Technol. **2015**(2), 206–221 (2015)
33. Kroll, J., Huey, J., Barocas, S., Felten, E., Reidenberg, J., Robinson, D., Yu, H.: Accountable algorithms. U. of Pennsylvania Law Review **165**(3), 633–705 (2017)
34. Libert, B., Ling, S., Mouhartem, F., Nguyen, K., Wang, H.: Zero-knowledge arguments for matrix-vector relations and lattice-based group encryption. In: ASIACRYPT 2016. LNCS, vol. 10032, pp. 101–131 (2016)
35. Libert, B., Ling, S., Mouhartem, F., Nguyen, K., Wang, H.: Adaptive oblivious transfer with access control from lattice assumptions. In: ASIACRYPT 2017. LNCS, vol. 10624, pp. 533–563. Springer (2017)

36. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-knowledge arguments for lattice-based prfs and applications to e-cash. In: ASIACRYPT 2017. LNCS, vol. 10626, pp. 304–335. Springer (2017)
37. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. J. Cryptol. **36**(3), 23 (2023)
38. Libert, B., Nguyen, K., Passelègue, A., Titiu, R.: Simulation-sound arguments for LWE and applications to KDM-CCA2 security. In: ASIACRYPT 2020. pp. 128–158. LNCS, Springer (2020)
39. Libert, B., Nguyen, K., Peters, T., Yung, M.: Bifurcated signatures: Folding the accountability vs. anonymity dilemma into a single private signing scheme. In: EUROCRYPT 2021. LNCS, vol. 12698, pp. 521–552. Springer (2021)
40. Ling, S., Nguyen, K., Phan, D.H., Tang, H., Wang, H.: Zero-knowledge proofs for committed symmetric boolean functions. In: PQCrypto 2021. LNCS, vol. 12841, pp. 339–359. Springer (2021)
41. Ling, S., Nguyen, K., Stehlé, D., Wang, H.: Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In: PKC 2013. LNCS, vol. 7778, pp. 107–124. Springer (2013)
42. Ling, S., Nguyen, K., Wang, H., Xu, Y.: Constant-size group signatures from lattices. In: PKC 2018. LNCS, vol. 10770, pp. 58–88. Springer (2018)
43. Lyubashevsky, V., Micciancio, D.: Generalized compact knapsacks are collision resistant. In: ICALP 2006. LNCS, vol. 4052, pp. 144–155. Springer (2006)
44. Lyubashevsky, V., Nguyen, N.K., Plan**cc**on, M.: Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In: CRYPTO 2022. LNCS, vol. 13508, pp. 71–101. Springer (2022)
45. Maji, H.K., Prabhakaran, M., Rosulek, M.: Attribute-based signatures. In: CT-RSA 2011. LNCS, vol. 6558, pp. 376–392. Springer (2011)
46. Micciancio, D.: Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. Comput. Complex. **16**(4), 365–411 (2007)
47. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer (2012)
48. Nguyen, K., Guo, F., Susilo, W., Yang, G.: Multimodal private signatures. In: CRYPTO 2022. LNCS, vol. 13508, pp. 792–822. Springer (2022)
49. Nguyen, K., Tang, H., Wang, H., Zeng, N.: New code-based privacy-preserving cryptographic constructions. In: ASIACRYPT 2019. LNCS, vol. 11922, pp. 25–55. Springer (2019)
50. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: CRYPTO 2019. LNCS, vol. 11692, pp. 89–114. Springer (2019)
51. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer (2001)
52. Sakai, Y., Emura, K., Hanaoka, G., Kawai, Y., Matsuda, T., Omote, K.: Group signatures with message-dependent opening. In: Pairing-Based Cryptography - Pairing 2012. LNCS, vol. 7708, pp. 270–294. Springer (2012)
53. Stern, J.: A new paradigm for public key identification. IEEE Trans. Inf. Theory **42**(6), 1757–1768 (1996)
54. Xu, S., Yung, M.: Accountable ring signatures: A smart card approach. In: CARDIS 2004. IFIP, vol. 153, pp. 271–286. Kluwer/Springer (2004)
55. Yang, R., Au, M.H., Zhang, Z., Xu, Q., Yu, Z., Whyte, W.: Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications. In: CRYPTO 2019. LNCS, vol. 11692, pp. 147–175. Springer (2019)

A Cryptographic Primitives

A.1 Signature Schemes

Definition 3. *A signature scheme* $\mathcal{S} = (\mathsf{S.Kg}, \mathsf{S.Sign}, \mathsf{S.Ver})$ *is a triple of algorithms defined as follows*:

- $\mathsf{S.Kg}(1^\lambda) \to (\mathsf{sk}, \mathsf{vk})$*: On input the security parameter* λ*, the key generation algorithm outputs a signing key* sk *and a verification key* vk.
- $\mathsf{S.Sign}(m, \mathsf{sk}) \to \Sigma$*: On input a signing key* sk*, and a message* $m \in \{0,1\}^*$*, the signing algorithm outputs a signature* Σ.
- $\mathsf{S.Ver}(\mathsf{vk}, m, \Sigma) \to \{0,1\}$*: On input a verification key* vk*, a message* m*, and a signature* Σ*, the verification algorithm accepts or rejects.*

We require a signature scheme $\mathcal{S}$ to satisfy the following correctness and security properties.

Correctness. A signature scheme $\mathcal{S}$ is correct if for all $\lambda \in \mathbb{N}$, for all m, we have that

$$\mathsf{Pr}\left[\mathsf{S.Verify}(\mathsf{vk}, m, \Sigma) = 1) \;\middle|\; \begin{array}{l}(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{S.Kg}(1^\lambda), \\ \Sigma \leftarrow \mathsf{S.Sign}(\mathsf{vk}, m).\end{array}\right] = 1.$$

Security. A signature scheme is existentially unforgeable under adaptive chosen message attack ($\mathsf{EUF\text{-}CMA}$) if for any PPT adversary $\mathcal{A}$, the following experiment $\mathbf{Exp}^{\mathsf{uf}}_{\mathcal{A}}(\lambda)$ outputs 1 with negligible probability in λ.

- The challenger runs $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{S.Kg}(1^\lambda)$, and provides vk to $\mathcal{A}$.
- $\mathcal{A}$ issues a polynomial number of signature queries on messages of its choice. The challenger computes $\Sigma \leftarrow \mathsf{S.Sign}(m, \mathsf{sk})$ for each query m and sends Σ to $\mathcal{A}$.
- At the end of the experiment, $\mathcal{A}$ outputs a forgery (m^*, Σ^*). The experiment outputs 1 if $\mathsf{S.Ver}(\mathsf{vk}, m^*, \Sigma^*) = 1$ and (m^*, Σ^*) was not obtained from the queries.

A.2 Commitment Schemes

Definition 4. *A commitment scheme* $\mathcal{COM} = (\mathsf{Setup},\ \mathsf{Com},\ \mathsf{Open})$ *is a triple of algorithms defined as follows:*

$\mathsf{C.Setup}(1^\lambda)$**:** *On input the security parameter* λ*, it returns system parameters* $\mathsf{pp_C}$.

$\mathsf{C.Com}(\mathsf{pp_C}, m)$ *On input* $\mathsf{pp_C}$ *and a message* m*, this probabilistic algorithm samples a random string* $\mathbf{r}$ *and computes* com*. It then returns the commitment* com *and* $(m, \mathbf{r})$.

$\mathsf{C.Open}(\mathsf{pp_C}, \mathsf{com}, m, \mathbf{r})$**:** *On input* $\mathsf{pp_C}$*,* com*, a message* m*, and randomness* $\mathbf{r}$*, this deterministic algorithm outputs* 1 *if* $\mathsf{com} = \mathsf{C.Com}(\mathsf{pp_C}, m, \mathbf{r})$ *or* 0 *otherwise.*

We require a commitment scheme $\mathcal{COM}$ to satisfy the following correctness, hiding, and binding properties.

Correctness. We say that a commitment scheme $\mathcal{COM}$ is correct if for all $\lambda \in \mathbb{N}$, all m, we have that

$$\Pr\left[\mathsf{C.Open}(\mathsf{pp_C}, \mathsf{com}, m, \mathbf{r}) = 1) \;\middle|\; \begin{array}{l} \mathsf{pp_C} \leftarrow \mathsf{C.Setup}(1^\lambda), \\ (\mathsf{com}, (m, \mathbf{r})) \leftarrow \mathsf{C.Com}(\mathsf{pp_C}, m). \end{array}\right] = 1.$$

Statistical hiding. For any $\mathsf{pp_C} \leftarrow \mathsf{C.Setup}(1^\lambda)$ and for any m_0, m_1, the distributions of $\{\mathsf{C.Com}(\mathsf{pp_C}, m_0)\}$ and $\{\mathsf{C.Com}(\mathsf{pp_C}, m_1)\}$ are statistically indistinguishable.

Computational binding. For any PPT adversary $\mathcal{A}$,

$$\Pr\left[\begin{array}{l} \mathsf{C.Com}(m_0, \mathbf{r}_0) = \mathsf{C.Com}(m_1, \mathbf{r}_1) \\ \wedge\ m_0 \neq m_1 \end{array} \;\middle|\; \begin{array}{l} \mathsf{pp_C} \leftarrow \mathsf{C.Setup}(1^\lambda), \\ (m_0, \mathbf{r}_0;\ m_1, \mathbf{r}_1) \leftarrow \mathcal{A}(\mathsf{pp_C}). \end{array}\right] = 1.$$

A.3 Simulation-Sound Extractable NIZKs

Here, we recall the definitions and security properties of simulation-sound extractable NIZK argument systems following the presentation by Bellare and Fuchsbauer [2].

Definition 5. *A simulation sound extractable NIZK for an NP-relation $\mathcal{R}$ is a tuple of polynomial-time algorithms $\mathcal{NIZK} = (\mathsf{ZK.Setup}, \mathsf{ZK.SimSetup}, \mathsf{ZK.Prove}, \mathsf{ZK.Ver}, \mathsf{ZK.Sim}, \mathsf{ZK.Extr})$, defined as follows.*

$\mathsf{ZK.Setup}(1^\lambda)$**:** *On input of the security parameter λ, this setup algorithm outputs a common reference string* crs.

$\mathsf{ZK.SimSetup}(1^\lambda)$**:** *On input λ, this simulated setup algorithm outputs a common reference string* crs *and a trapdoor* tr.

$\mathsf{ZK.Prove}(\mathsf{crs}, x, w)$**:** *This proof algorithm takes as inputs* crs*, a statement x and a witness w, and outputs a proof π.*

$\mathsf{ZK.Sim}(\mathsf{crs}, \mathsf{tr}, x)$**:** *This simulated proof algorithm takes as inputs* crs*, a trapdoor* tr*, a statement x, and outputs a simulated proof π.*

$\mathsf{ZK.Ver}(\mathsf{crs}, x, \pi)$**:** *On input* crs*, a statement-proof pair (x, π), it outputs either 1 or 0, indicating the (in)validity of proof π.*

$\mathsf{ZK.Extr}(\mathsf{crs}, \mathsf{tr}, x, \pi)$**:** *On input* crs*, a trapdoor* tr*, a statement x and a proof π, this algorithm outputs a witness w.*

Perfect completeness. For any $\lambda \in \mathbb{N}$, any $(x, w) \in \mathcal{R}$, we have

$$\Pr\left[\mathsf{ZK.Ver}(\mathsf{crs}, x, \pi) = 1 \;\middle|\; \begin{array}{r} \mathsf{crs} \leftarrow \mathsf{ZK.Setup}(1^\lambda), \\ \pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, x, w) \end{array}\right] = 1.$$

Computational soundness. For any PPT adversary $\mathcal{A}$,

$\mathbf{Exp}^{\mathsf{ZK}}_{\mathcal{NIZK},\mathcal{A}}(\lambda)$	$\mathbf{Exp}^{\mathsf{SE}}_{\mathcal{NIZK},\mathcal{A}}(\lambda)$
Initialize	Initialize
$b \leftarrow \{0,1\}$	$(\mathsf{crs},\mathsf{tr}) \leftarrow \mathsf{ZK.SimSetup}(1^\lambda)$; $Q \leftarrow \emptyset$
$(\mathsf{crs}_0,\mathsf{tr}) \leftarrow \mathsf{ZK.SimSetup}(1^\lambda)$	Return crs
$(\mathsf{crs}_1) \leftarrow \mathsf{ZK.Setup}(1^\lambda)$	SimPROOF(x)
Return crs_b	$\pi \leftarrow \mathsf{ZK.Sim}(\mathsf{crs},\mathsf{tr},x)$; $Q \leftarrow Q \cup \{(x,\pi)\}$
PROOF(x,w)	Return π
If $\mathcal{R}(x,w)=1$, then $\pi_0 \leftarrow \mathsf{ZK.Sim}(\mathsf{crs}_0,\mathsf{tr},x)$	Finalize(x,π)
else $\pi_0 \leftarrow \perp$;	$w \leftarrow \mathsf{ZK.Extr}(\mathsf{crs},\mathsf{tr},x,\pi)$
$\pi_1 \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}_1,x,w)$	Return 1 if all of the following hold:
Return π_b	$(x,\pi) \notin Q$
Finalize(b')	$\mathsf{ZK.Ver}(\mathsf{crs},x,\pi)=1$
Return $(b=b')$	$\mathcal{R}(x,w)=0$

Table 1. Games defining zero-knowledge and simulation sound extractability for NIZK

$$\mathsf{Pr}\left[\begin{matrix}\mathsf{ZK.Ver}(\mathsf{crs},x,\pi)=1 \\ \wedge \;\; x \notin \mathcal{L}_{\mathcal{R}}\end{matrix}\;\middle|\;\begin{matrix}\mathsf{crs} \leftarrow \mathsf{ZK.Setup}(1^\lambda), \\ (x,\pi) \leftarrow \mathcal{A}(\mathsf{crs}).\end{matrix}\right] = 1 - \mathsf{negl}(\lambda).$$

Zero-knowledge and simulation-sound extractability are defined via the two experiments $\mathbf{Exp}^{\mathsf{ZK}}_{\mathcal{NIZK},\mathcal{A}}(\lambda)$ and $\mathbf{Exp}^{\mathsf{SE}}_{\mathcal{NIZK},\mathcal{A}}(\lambda)$ in Table 1.

Statistical zero-knowledge. For any $\mathcal{A}$ and for all $(x,w) \in \mathcal{R}$,

$$\mathbf{Adv}^{\mathsf{ZK}}_{\mathcal{NIZK},\mathcal{A}} = |\Pr\left[\mathbf{Exp}^{\mathsf{ZK}}_{\mathcal{NIZK},A}(\lambda)\right] - \frac{1}{2}| = \mathsf{negl}(\lambda).$$

Simulation-sound extractability. For every PPT adversary $\mathcal{A}$,

$$\mathbf{Adv}^{\mathsf{SE}}_{\mathcal{NIZK},\mathcal{A}} = \Pr\left[\mathbf{Exp}^{\mathsf{SE}}_{\mathcal{NIZK},A}(\lambda) = 1\right] = \mathsf{negl}(\lambda).$$

B Some Preliminary Lattice-Based Techniques and Tools

Throughout this work, all vectors are column vectors. Let q be a positive integer, identify $\mathbb{Z}_q = \{\frac{q-1}{2}, \ldots, -1, 0, 1, \ldots, \frac{q-1}{2}\}$.

B.1 Rings and Ring-Based Hash Functions

Let n be a power of 2, q be a large prime $q = q(n) = n^{\mathcal{O}(1)}$, and $f(X) = X^n + 1 \in \mathbb{Z}[X]$ be a monic and irreducible polynomial of degree n. Define quotient rings $R = \mathbb{Z}[X]/(f(X))$ and $R_q = R/(qR)$.

Let $\tau : (R_q)^{m'} \rightarrow \mathbb{Z}_q^{nm'}$ be the coefficient embedding that maps a ring vector $\mathbf{a} = (a_0, a_1, \ldots, a_{m'-1})^\top \in (R_q)^{m'}$ to $\tau(\mathbf{a}) \in \mathbb{Z}_q^{nm'}$, where each

$$a_i = a_{i,0} + a_{i,1} \cdot X + \cdots + a_{i,n-1} \cdot X^{n-1} \in R_q,$$

$$\tau(\mathbf{a}) = (a_{0,0}, \ldots, a_{0,n-1}, a_{1,0}, \ldots, a_{1,n-1}, \ldots, a_{m'-1,0}, \ldots, a_{m'-1,n-1})^\top \in \mathbb{Z}_q^{nm'}.$$

Since τ is bejective, we will use $(R_q)^{m'}$ and $\mathbb{Z}_q^{nm'}$ interchangeably in this work. Therefore, the (infinity) norm of $\mathbf{a}$ is defined as $\|\mathbf{a}\|_\infty = \|\tau(\mathbf{a})\|_\infty$.

Define $\mathsf{rot} : R_q \to \mathbb{Z}_q^{n\times n}$ as

$$\mathsf{rot}(v) = [\tau(v), \tau(v \cdot X), \ldots, \tau(v \cdot X^{n-1})] \in \mathbb{Z}_q^{n\times n}. \tag{12}$$

Let $\mathbf{A} = [a_0, a_1, \ldots, a_{m'-1}] \in R_q^{1\times m'}$, we abuse the notion of rot by defining $\mathsf{rot}(\mathbf{A}) = [\mathsf{rot}(a_0), \mathsf{rot}(a_1), \ldots, \mathsf{rot}(a_{m'-1})] \in \mathbb{Z}_q^{n\times nm'}$. It is verifiable that

$$y = \mathbf{A} \cdot \mathbf{x} \Leftrightarrow \tau(y) = \mathsf{rot}(\mathbf{A}) \cdot \tau(\mathbf{x}), \tag{13}$$

where $\mathbf{A} \in (R_q)^{1\times m'}$ and $\mathbf{x} \in (R_q)^{m'}$. We note that for $\mathbf{x} \in \{0,1\}^{m'}$ with a sole 1 in some position, then $y = \mathbf{A} \cdot \mathbf{x}$ where each bit of $\mathbf{x}$ is treated as either constant polynomial 0 or 1 can be simplified as $\tau(y) = [\tau(a_0), \tau(a_1), \ldots, \tau(a_{m'-1})] \cdot \mathbf{x}$.

We now describe a family of hash functions as in [43,30].

$$\mathcal{H}_{m'} = \{h_{\mathbf{A}} : \{0,1\}^{nm'} \to R_q | \mathbf{A} \in (R_q)^{1\times m'}\}, \tag{14}$$

where $h_{\mathbf{A}}(\mathbf{x}) = \mathbf{A} \cdot \mathbf{x}^7$ and $\mathbf{x} \in \{0,1\}^{nm'} \subset (R_q)^{m'}$.

Lemma 1 (Collision resistance [43,30]). *For any $m' = m'(n) > \log q$, there exists an integer $q > 6m'n^{3/2}\log n$ and $\gamma = 72m'n\log^2 n$, such that $\mathcal{H}_{m'}$ is collision resistant if $\mathsf{SVP}_\gamma^\infty$ is hard for any ideal in the ring R.*

B.2 Decompositions

Let us recall the decomposition technique from [41]. Let B be any positive integer. Define $\delta_B = \lfloor \log B \rfloor + 1$ and a sequence $B_0, B_1, \ldots, B_{\delta_B - 1}$ where $B_j = \left\lfloor \frac{B+2^j}{2^{j+1}} \right\rfloor$ for $j \in [0, \delta_B - 1]$. It is verifiable that $\sum_{j=0}^{\delta_B-1} B_j = B$. For any $v \in [0, B]$, let us perform the following algorithm.

1. $v' := v$.
2. For $j \in [0, \delta_B - 1]$,
   (a) If $v' > B_j$, set $v^{(j)} := 1$, $v' := v' - B_j$
   (b) Else: set $v^{(j)} := 0$.
3. Return $\mathsf{idec}_B(v) = (v^{(0)}, v^{(1)}, \ldots, v^{(\delta_B-1)})^\top \in \{0,1\}^{\delta_B}$.

It is not hard to see that $v = \sum_{j=0}^{\delta_B-1} v^{(j)} \cdot B_j$. For $v \in [-B, 0]$, $\mathsf{idec}_B(v)$ is defined as $(-1) \cdot \mathsf{idec}_B(|v|) \in \{-1, 0\}^{\delta_B}$.

For example, for $B = 15$, we have $B_0 = 8$, $B_1 = 4$, $B_2 = 2$, $B_1 = 1$, and $7 = 0 \cdot B_0 + 1 \cdot B_1 + 1 \cdot B_2 + 1 \cdot B_3$. Thus $\mathsf{idec}_{15}(7) = (0, 1, 1, 1)^\top$.

In this work, we also employ the above decomposition on ring vectors. Let B be $2 \le B \le \frac{q-1}{2}$ and $\mathbf{a} = (a_0, a_1, \ldots, a_{m'-1})^\top \in (R_q)^{m'}$, where $a_i = a_{i,0} + a_{i,1} \cdot$

[7] Note that we write $h_{\mathbf{A}}(\mathbf{x}) = \mathbf{A} \cdot \mathbf{x}$ instead of $h_{\mathbf{A}}(\mathbf{x}) = \mathbf{A} \cdot \tau^{-1}(\mathbf{x})$ if there is no risk of ambiguity.

$X + \cdots + a_{i,n-1} \cdot X^{n-1} \in R_q$ and $a_{i,0}, \ldots, a_{i,n-1} \in [-B, B]$. Define $\mathsf{rdec}_B(\mathbf{a}) \in \{-1, 0, 1\}^{nm'\delta_B} \subset R^{m'\delta_B}$ as

$$\mathsf{rdec}_B(\mathbf{a}) = \big(\ \mathsf{idec}_B(a_{0,0})^\top, \mathsf{idec}_B(a_{0,1})^\top, \ldots, \mathsf{idec}_B(a_{0,n-1})^\top, \ldots, \\ \mathsf{idec}_B(a_{m'-1,0})^\top, \mathsf{idec}_B(a_{m'-1,1})^\top, \ldots, \mathsf{idec}_B(a_{m'-1,n-1})^\top \ \big)^\top.$$

Define two matrices $\mathbf{H}_B \in \mathbb{Z}^{n \times n\delta_B}$ and $\mathbf{H}_{m',B} \in \mathbb{Z}^{nm' \times nm'\delta_B}$ as

$$\mathbf{H}_B = \begin{bmatrix} B_0 \ldots B_{\delta_B - 1} & & \\ & \ddots & \\ & & B_0 \ldots B_{\delta_B - 1} \end{bmatrix}, \quad \mathbf{H}_{m',B} = \begin{bmatrix} \mathbf{H}_B & & \\ & \ddots & \\ & & \mathbf{H}_B \end{bmatrix}. \tag{15}$$

Then we have

$$\mathbf{H}_{m',B} \cdot \mathsf{rdec}_B(\mathbf{a}) = \tau(\mathbf{a}) \in \mathbb{Z}^{nm'}. \tag{16}$$

For simplicity, we write $\mathbf{H}, \mathbf{H}_{m'}, \delta, \mathsf{idec}, \mathsf{rdec}$ instead of $\mathbf{H}_{\frac{q-1}{2}}$, $\mathbf{H}_{m',\frac{q-1}{2}}$, $\delta_{\frac{q-1}{2}}$, $\mathsf{idec}_{\frac{q-1}{2}}$, $\mathsf{rdec}_{\frac{q-1}{2}}$ when $B = \frac{q-1}{2}$.

B.3 A Lattice-Based Commitment Scheme

Let us now describe an ideal-lattice-based commitment scheme, adapted from [30]. The scheme is computationally binding and statistically hiding by setting suitable parameters and assuming the hardness of $\mathsf{SVP}_\gamma^\infty$ for any ideal in the ring R with some $\gamma = 72m'n\log^2 n$.

$\mathsf{Setup}(1^\lambda)$ This algorithm first determines the public parameters for the commitment scheme. Choose $n = \mathcal{O}(\lambda)$ which is also a power of 2, $m' > 2\log q$, and appropriate $q = q(n) > 6m'n^{3/2}\log n$. Let $f(X) = X^n + 1 \in \mathbb{Z}[X]$ and define R, R_q as above. Let the message space be $\{0,1\}^{nm'} \subset (R_q)^{m'}$. Next, sample $\mathbf{A}_0 \xleftarrow{\$} (R_q)^{1\times m'}$ and $\mathbf{A}_1 \xleftarrow{\$} (R_q)^{1\times m'}$. Output $\mathsf{pp} = \{n, q, f, m', \mathbf{A}_0, \mathbf{A}_1\}$.

$\mathsf{Com}(\mathsf{pp}, M)$ On input a message $M \in \{0,1\}^{nm'} \subset (R_q)^{m'}$, this algorithm first samples $\mathbf{r} \xleftarrow{\$} \{0,1\}^{nm'} \subset (R_q)^{m'}$ and then computes $c = \mathbf{A}_0 \cdot M + \mathbf{A}_1 \cdot \mathbf{r} \in R_q$. Output c together with the opening $\mathbf{r}$.

$\mathsf{Open}(\mathsf{pp}, c, M, \mathbf{r})$ On input pp, commitment $c \in R_q$, message $M \in \{0,1\}^{nm'}$ and opening $\mathbf{r} \in \{0,1\}^{nm'}$, this algorithm returns 1 if $c = \mathbf{A}_0 \cdot M + \mathbf{A}_1 \cdot \mathbf{r}$.

Denote the scheme as $\mathsf{CMT}_{m',m'}$ if the rings R, R_q are clear from the context. Here the first subscript is regarding the message length while the other one is regarding the randomness length. We also need a string commitment scheme in our zero-knowledge argument of knowledge protocol. This can be achieved by applying the Merkle-Damgard technique to the above commitment scheme. For simplicity, we will use CMT to denote such a scheme without explicitly specifying the parameters.

B.4 A Variant of the Ducas-Micciancio Signature Scheme

We recall a variant of the Ducas-Micciancio signature scheme [15], which is stateful and secure against an adaptive adversary who can make at most polynomial number of signature queries. Looking ahead, the signature scheme is used to issue attribute signing keys and policy certificates, thus this variant suffices. Let n, f, q, R, R_q as before. Fix a real constant $c > 1$, define $\alpha_0 \geq \frac{1}{c-1}$ and $d \geq \log_c(\omega(\log n))$. For $i \in [1, d]$, let $c_i = \lfloor \alpha_0 \cdot c^i \rfloor$ and $\mathcal{T}_i = \{0,1\}^{c_i}$. It is required that $c_d \leq n/2$. An element $t = (t_0, t_1, \ldots, t_{c_i-1})^\top \in \mathcal{T}_i$ is identified as a ring element $t(X) = \sum_{j=0}^{c_i-1} t_j \cdot X^j$. For a prefix $t_{\leq i} = (t_0, \ldots, t_{c_i-1})^\top \in \mathcal{T}_i$ of an element $t = (t_0, t_1, \ldots, t_{c_d-1}) \in \mathcal{T}_d$, let $t_{[i]}$ be the ring element $t_{\leq i}(X) - t_{\leq i-1}(X)$.

Details of the scheme are described below.

$\mathsf{Keygen}(1^\lambda)$ Given the security parameter, this algorithm proceeds as follows.

- Let $n = \mathcal{O}(\lambda)$ be a power of 2, modulus $q = 3^k$, and an irreducible polynomial $f(X) = X^n + 1$. Define rings $R = \mathbb{Z}[X]/(f(X))$ and $R_q = R/(qR)$.
- Choose $c, \alpha_0, d, c_1, \ldots, c_d$ as described above.
- Define $m = 2\lceil \log q \rceil + 2$, $\overline{m} = m + k$, $\ell = \delta_{\frac{q-1}{2}} = \lfloor \log \frac{q-1}{2} \rfloor + 1$, $\beta = \widetilde{\mathcal{O}}(n)$ to be an integer.
- Set the initial state $S := 0$.
- The verification key vk then consists of the following:

$$\mathbf{A}, \mathbf{F}_0 \in (R_q)^{1 \times \overline{m}}, \mathbf{A}_{[0]}, \ldots, \mathbf{A}_{[d]} \in (R_q)^{1 \times k}, \mathbf{F}, \mathbf{F}_1 \in (R_q)^{1 \times \ell}, u \in R_q.$$

- The signing key sk is a Micciancio-Peikert [47] trapdoor $\mathbf{R} \in (R_q)^{m \times k}$.

$\mathsf{Sign}(\mathsf{sk}, M)$ On input the signing key and a message $M \in \{0,1\}^{n\ell} \subset (R_q)^\ell$, this algorithm performs the following steps.

- First, compute $t = (t_0, \ldots, t_{c_d-1})^\top \in \mathcal{T}_d$ such that $S = \sum_{j=0}^{c_d-1} 2^{c_d-1-j} t_j$ and then update S to $S + 1$.
- Next, compute $\mathbf{A}_t = [\mathbf{A}, \mathbf{A}_{[0]} + \sum_{i=1}^{d} \mathbf{A}_{[i]} t_{[i]}] \in (R_q)^{1 \times (\overline{m}+k)}$.
- Sample $\mathbf{r} \in (R_q)^{\overline{m}}$ satisfying $\|\mathbf{r}\|_\infty \leq \beta$ and compute $u_{\mathsf{msg}} = \mathbf{F} \cdot \mathsf{rdec}(\mathbf{F}_0 \cdot \mathbf{r} + \mathbf{F}_1 \cdot M) + u \in R_q$.
- Utilizing the trapdoor $\mathbf{R}$, output a short vector $\mathbf{v} \in (R_q)^{\overline{m}+k}$ such that $\|\mathbf{v}\|_\infty \leq \beta$ and $\mathbf{A}_t \cdot \mathbf{v} = u_{\mathsf{msg}}$.
- Output the signature of M as $\sigma = (t, \mathbf{r}, \mathbf{v})$.

$\mathsf{Verify}(\mathsf{vk}, M, \sigma)$ Given the inputs, the verification algorithm computes $\mathbf{A}_t$ and u_{msg} as in the signing algorithm and then verifies the following three conditions.

$$\begin{cases} \mathbf{A}_t \cdot \mathbf{v} = u_{\mathsf{msg}}; \\ \|\mathbf{r}\|_\infty \leq \beta; \|\mathbf{v}\|_\infty \leq \beta; \end{cases} \tag{17}$$

Return 1 if all conditions are satisfied and 0 otherwise.

Lemma 2. *The above stateful Ducas-Micciancio signature scheme is correct with overwhelming probability and existentially unforgeable against adaptive chosen message attacks* ($\mathsf{EUF\text{-}CMA}$) *assuming the hardness of* SVP_γ *in any ideal of ring R with $\gamma = \widetilde{\mathcal{O}}(n^3)$ and at most polynomial number of signature queries.*

B.5 Zero-Knowledge Arguments and Stern-like Protocols

We will work with statistical zero-knowledge argument systems, namely, interactive protocols where the zero-knowledge property holds against *any* cheating verifier, while the soundness property only holds against *computationally bounded* cheating provers. The statistical zero-knowledge arguments of knowledge presented in this work are Stern-like [53] protocols. In particular, they are Σ-protocols in the generalized sense defined in [26,5] (where 3 valid transcripts are needed for extraction, instead of just 2). The basic protocol consists of 3 moves: commitment, challenge, and response. If a statistically hiding and computationally binding string commitment is employed in the first move, then one obtains a statistical zero-knowledge argument of knowledge (ZKAoK) with perfect completeness, constant soundness error 2/3.

An abstraction of Stern's protocol. Here, we recall a simplified abstract Stern-type protocol, adapted from [36]. The simplified variant handles modular equations with respect to 2 moduli q_1, q_2, where secret witnesses may simultaneously appear across multiple equations.

Let K_i, L_i be positive integers with $K_i \geq L_i$. Let $K = K_1 + K_2$ and $L = L_1 + L_2$. Suppose that VALID is a subset of $\{-1, 0, 1\}^L$ and $\mathcal{S}$ is a finite set such that every $\phi \in \mathcal{S}$ can be associated with a permutation $\mathit{\Gamma}_\phi$ of L elements satisfying the conditions

$$\begin{cases} \mathbf{w} \in \mathsf{VALID} \iff \mathit{\Gamma}_\phi(\mathbf{w}) \in \mathsf{VALID}; \\ \text{If } \mathbf{w} \in \mathsf{VALID} \text{ and } \phi \text{ is uniform in } \mathcal{S}, \text{ then } \mathit{\Gamma}_\phi(\mathbf{w}) \text{ is uniform in } \mathsf{VALID}. \end{cases} \tag{18}$$

In the abstract protocol, for public matrices $\{\mathbf{M}_i \in \mathbb{Z}_{q_i}^{K_i \times L_i}\}_{i \in \{1,2\}}$ and vectors $\{\mathbf{v}_i \in \mathbb{Z}_{q_i}^{K_i}\}_{i \in \{1,2\}}$, the prover argues in zero-knowledge the possession of integer vectors $\{\mathbf{w}_i \in \{-1, 0, 1\}^{L_i}\}_{i \in \{1,2\}}$ such that:

$$\mathbf{w} = \left(\mathbf{w}_1^\top \mid \mathbf{w}_2^\top\right)^\top \in \mathsf{VALID}, \tag{19}$$

$$\forall i \in \{1, 2\} : \mathbf{M}_i \cdot \mathbf{w}_i = \mathbf{v}_i \bmod q_i. \tag{20}$$

Formally, the goal is to construct a statistical ZKAoK for the following relation:

$$\mathrm{R}_{\text{abstract}} = \Big\{\{(\mathbf{M}_i, \mathbf{v}_i) \in \mathbb{Z}_{q_i}^{K_i \times L_i} \times \mathbb{Z}_{q_i}^{K_i}\}_{i \in \{1,2\}},\ \mathbf{w} = \left(\mathbf{w}_1^\top \mid \mathbf{w}_2^\top\right)^\top \in \mathsf{VALID} : \mathbf{M}_i \cdot \mathbf{w}_i = \mathbf{v}_i \bmod q_i, \forall i \in \{1, 2\}\Big\}.$$

The two relations underlying the signing and disclosing algorithms will be reduced to an instance of $\mathrm{R}_{\text{abstract}}$ with moduli q and 2.

The main ideas underlying the protocol are as follows. To prove (19), the prover samples $\phi \xleftarrow{\$} \mathcal{S}$ and provides evidence that $\mathit{\Gamma}_\phi(\mathbf{w}) \in \mathsf{VALID}$. The verifier should be convinced while learning nothing else, owing to the aforementioned properties of the sets VALID and $\mathcal{S}$. Meanwhile, to prove that equations (20) hold, the prover uses masking vectors $\{\mathbf{r}_i \xleftarrow{\$} \mathbb{Z}_{q_i}^{L_i}\}_{i \in \{1,2\}}$ and demonstrates instead that $\mathbf{M}_i \cdot (\mathbf{w}_i + \mathbf{r}_i) = \mathbf{v}_i + \mathbf{M}_i \cdot \mathbf{r}_i \bmod q_i$.

Inputs. Common input: $\{(\mathbf{M}_i, \mathbf{v}_i)\}_{i\in\{1,2\}}$. Prover's secret: $\mathbf{w} = \left(\mathbf{w}_1^\top \mid \mathbf{w}_2^\top\right)^\top$.

1. **Commitment:** $\mathcal{P}$ samples $\phi \xleftarrow{\$} \mathcal{S}$, $\{\mathbf{r}_i \xleftarrow{\$} \mathbb{Z}_{q_i}^{L_i}\}_{i\in\{1,2\}}$, and then computes vectors $\mathbf{r} = (\mathbf{r}_1^\top \mid \mathbf{r}_2^\top)^\top$, $\mathbf{z} = \mathbf{w} \boxplus \mathbf{r}$.
   Then $\mathcal{P}$ samples randomness ρ_1, ρ_2, ρ_3 for COM, and sends $\mathrm{CMT} = \left(C_1, C_2, C_3\right)$ to $\mathcal{V}$, where $C_1 = \mathsf{COM}(\phi, \{\mathbf{M}_i \cdot \mathbf{r}_i \bmod q_i\}_{i\in\{1,2\}};\, \rho_1)$, and
$$C_2 = \mathsf{COM}(\Gamma_\phi(\mathbf{r});\, \rho_2), \quad C_3 = \mathsf{COM}(\Gamma_\phi(\mathbf{z});\, \rho_3).$$
2. **Challenge:** $\mathcal{V}$ sends a challenge $Ch \xleftarrow{\$} \{1, 2, 3\}$ to $\mathcal{P}$.
3. **Response:** $\mathcal{P}$ sends RSP computed according to Ch, as follows:
   - $Ch = 1$: $\mathrm{RSP} = (\mathbf{t}, \mathbf{s}, \rho_2, \rho_3)$, where $\mathbf{t} = \Gamma_\phi(\mathbf{w})$ and $\mathbf{s} = \Gamma_\phi(\mathbf{r})$.
   - $Ch = 2$: $\mathrm{RSP} = (\pi, \mathbf{x}, \rho_1, \rho_3)$, where $\pi = \phi$ and $\mathbf{x} = \mathbf{z}$.
   - $Ch = 3$: $\mathrm{RSP} = (\psi, \mathbf{y}, \rho_1, \rho_2)$, where $\psi = \phi$ and $\mathbf{y} = \mathbf{r}$.

Verification: Receiving RSP, $\mathcal{V}$ proceeds as follows:

- $Ch = 1$: Check that $\mathbf{t} \in \mathsf{VALID}$, and $C_2 = \mathsf{COM}(\mathbf{s}; \rho_2)$, $C_3 = \mathsf{COM}(\mathbf{t} \boxplus \mathbf{s}; \rho_3)$.
- $Ch = 2$: Parse $\mathbf{x} = (\mathbf{x}_1^\top \mid \mathbf{x}_2^\top)^\top$, where $\mathbf{x}_i \in \mathbb{Z}_{q_i}^{L_i}$ for all $i \in \{1, 2\}$, and check that
$$C_1 = \mathsf{COM}(\pi, \{\mathbf{M}_i \cdot \mathbf{x}_i - \mathbf{v}_i \bmod q_i\}_{i\in\{1,2\}}; \rho_1), \quad C_3 = \mathsf{COM}(\Gamma_\pi(\mathbf{x}); \rho_3).$$
- $Ch = 3$: Parse $\mathbf{y} = (\mathbf{y}_1^\top \mid \mathbf{y}_2^\top)^\top$, where $\mathbf{y}_i \in \mathbb{Z}_{q_i}^{L_i}$ for all $i \in \{1, 2\}$, and check that
$$C_1 = \mathsf{COM}(\psi, \{\mathbf{M}_i \cdot \mathbf{y}_i \bmod q_i\}_{i\in\{1,2\}}; \rho_1), \quad C_2 = \mathsf{COM}(\Gamma_\psi(\mathbf{y}); \rho_2).$$

In each case, $\mathcal{V}$ outputs 1 if and only if all the conditions hold.

Table 2. A Stern-type ZKAoK for the relation $\mathrm{R}_{\mathrm{abstract}}$.

The interaction between prover $\mathcal{P}$ and verifier $\mathcal{V}$ is described in Table 2. The common input consists of $\{\mathbf{M}_i \in \mathbb{Z}_{q_i}^{K_i \times L_i}\}_{i\in\{1,2\}}$ and $\{\mathbf{v}_i \in \mathbb{Z}_{q_i}^{K_i}\}_{i\in[\nu]}$, while $\mathcal{P}$'s secret input is $\mathbf{w} = \left(\mathbf{w}_1^\top \mid \mathbf{w}_2^\top\right)^\top$. The protocol makes use of a statistically hiding and computationally binding string commitment scheme COM such as the SIS-based commitment of [30].

For simplicity of presentation, for vectors $\mathbf{w} = \left(\mathbf{w}_1^\top \mid \mathbf{w}_2^\top\right)^\top \in \mathbb{Z}^L$ and $\mathbf{r} = \left(\mathbf{r}_1^\top \mid \mathbf{r}_2^\top\right)^\top \in \mathbb{Z}^L$, we denote by $\mathbf{w} \boxplus \mathbf{r}$ the operation that computes $\mathbf{z}_i = \mathbf{w}_i + \mathbf{r}_i \bmod q_i$ for all $i \in \{1, 2\}$, and outputs L-dimensional integer vector $\mathbf{z} = \left(\mathbf{z}_1^\top \mid \mathbf{z}_2^\top\right)^\top$. We note that, for all $\phi \in \mathcal{S}$, if $\mathbf{t} = \Gamma_\phi(\mathbf{w})$ and $\mathbf{s} = \Gamma_\phi(\mathbf{r})$, then we have $\Gamma_\phi(\mathbf{w} \boxplus \mathbf{r}) = \mathbf{t} \boxplus \mathbf{s}$.

The properties of the protocol are summarized in Theorem 5.

Theorem 5 ([36]). *Suppose that* COM *is a statistically hiding and computationally binding string commitment. Then, the protocol of Table 2 is a statistical* ZKAoK *for* $\mathrm{R}_{\mathrm{abstract}}$*, with perfect completeness, soundness error* $2/3$*, and communication cost* $\mathcal{O}\left(\sum_{i=1}^{2} L_i \cdot \log q_i\right)$*. In particular:*

– *There exists an efficient simulator that, on input* $\{\mathbf{M}_i, \mathbf{v}_i\}_{i\in\{1,2\}}$*, outputs an accepted transcript statistically close to that produced by the real prover.*
– *There exists an efficient knowledge extractor that, on input a commitment* CMT *as well as valid responses* $(\mathrm{RSP}_1, \mathrm{RSP}_2, \mathrm{RSP}_3)$ *to all three possible values of the challenge Ch, outputs a witness* $\mathbf{w}' = (\mathbf{w}_1'^{\top} \mid \mathbf{w}_2'^{\top})^{\top} \in \mathsf{VALID}$ *such that* $\mathbf{M}_i \cdot \mathbf{w}_i' = \mathbf{v}_i \bmod q_i$*, for all* $i \in \{1, 2\}$.

The proof of the Theorem 5 employs standard simulation and extraction techniques for Stern-type protocols [30,41]. Details are omitted here and can be found in [36].

C Deferred Security Proofs for the Generic Construction

C.1 Auxiliary Algorithms for the Generic Construction

To show that our construction satisfies the requirements as defined in Section 2.2, we would need to construct the following auxiliary algorithms.

$\mathsf{SimSetup}(1^\lambda)$**:** Compared to the real Setup algorithm, the following changes are introduced.
  – At Step 2, the common reference strings and associated trapdoors are generated via simulated algorithms $(\mathsf{crs}_\mathsf{S}, \mathsf{tr}_\mathsf{S}) \leftarrow \mathsf{ZK}_\mathsf{S}.\mathsf{SimSetup}(1^\lambda)$ for $\mathcal{NIZK}_\mathsf{S}$ and $(\mathsf{crs}_\mathsf{D}, \mathsf{tr}_\mathsf{D}) \leftarrow \mathsf{ZK}_\mathsf{D}.\mathsf{SimSetup}(1^\lambda)$ for $\mathcal{NIZK}_\mathsf{D}$.
  – The algorithm outputs $\mathsf{tr} := (\mathsf{tr}_\mathsf{S}, \mathsf{tr}_\mathsf{D})$ in addition to $(\mathsf{PP}, \mathsf{msk}_\mathsf{X}, \mathsf{msk}_\mathsf{P})$.

$\mathsf{SimSign}(\mathsf{tr}, m, P, \mathbf{x}, \mathbf{w})$**:** If $P(m, \mathbf{x}, w) = 0$, i.e., then the algorithm returns 0. Otherwise, it uses trapdoor $\mathsf{tr} = (\mathsf{tr}_\mathsf{S}, \mathsf{tr}_\mathsf{D})$ to simulate a signature on message m via the following steps.

1. Compute $\mathsf{com}_\mathbf{x}$ and com_P as commitments to all-zero strings. Namely, $\mathsf{com}_\mathbf{x} = \mathsf{C.Com}(\mathsf{pp}_\mathsf{C}, \mathbf{0}, \mathbf{r}_{\mathsf{com},\mathbf{x}})$ and $\mathsf{com}_P = \mathsf{C.Com}(\mathsf{pp}_\mathsf{C}, \mathbf{0}, \mathbf{r}_{\mathsf{com},P})$.
2. Simulate the NIZK argument Π using tr_S via

$$\Pi \leftarrow \mathsf{ZK}_\mathsf{S}.\mathsf{Sim}(\mathsf{crs}_\mathsf{S}, \mathsf{tr}_\mathsf{S}, (m, \mathsf{vk}_\mathsf{X}, \mathsf{vk}_\mathsf{P}, \mathsf{pp}_\mathsf{C}, \mathsf{com}_\mathbf{x}, \mathsf{com}_P)).$$

3. Output the simulated signature $\Sigma = (\mathsf{com}_\mathbf{x}, \mathsf{com}_P, \Pi)$.

$\mathsf{SimDisclose}(m, \Sigma, \mathsf{tr}, P, \mathbf{x}, F)$**:** If $\mathsf{Verify}(m, \Sigma) = 0$, then the algorithm returns $\perp$. Otherwise, let $\mathsf{tr} = (\mathsf{tr}_\mathsf{S}, \mathsf{tr}_\mathsf{D})$ and parse $\Sigma = (\mathsf{com}_\mathbf{x}, \mathsf{com}_P, \Pi)$. The disclosure process is then simulated via the following steps.

1. Compute $t = F(P, \mathbf{x})$.
2. Simulate the NIZK argument a using tr_D via

$$a \leftarrow \mathsf{ZK}_\mathsf{D}.\mathsf{Sim}(\mathsf{crs}_\mathsf{D}, \mathsf{tr}_\mathsf{D}, (F, t, \mathsf{com}_\mathbf{x}, \mathsf{com}_P, \mathsf{pp}_\mathsf{C})).$$

3. Output the simulated testimony-attestation pair (t, a).

$\mathsf{Extract}(\mathsf{tr}, m, \Sigma)$**:** Let $\mathsf{tr} = (\mathsf{tr}_\mathsf{S}$ and parse $\Sigma = (\mathsf{com}_\mathbf{x}, \mathsf{com}_P, \Pi)$. The algorithm uses tr_S to extract

$$\eta' \leftarrow \mathsf{ZK}_\mathsf{S}.\mathsf{Extr}(\mathsf{crs}_\mathsf{S}, \mathsf{tr}_\mathsf{S}, (m, \mathsf{vk}_\mathsf{X}, \mathsf{vk}_\mathsf{P}, \mathsf{pp}_\mathsf{C}, \mathsf{com}_\mathbf{x}, \mathsf{com}_P), \Pi),$$

where $\eta' = (\mathbf{x}', \mathsf{sk}', P', \mathsf{Cert}', w', \mathbf{r}'_{\mathsf{com},\mathbf{x}}, \mathbf{r}'_{\mathsf{com},P})$. It then returns $(P', \mathbf{x}', w')$.

With the above auxiliary algorithms, we are ready to show that our generic construction satisfies simulatability and extractability.

C.2 Proof of Theorem 1

Proof. We prove the theorem via a sequence of games that are statistically indistinguishable, where the first is the experiment $\mathbf{Exp}^{\mathsf{sim}}_{\mathcal{A}}(\lambda)$ with b set to 1 while the last is $\mathbf{Exp}^{\mathsf{sim}}_{\mathcal{A}}(\lambda)$ with b set to 0. In the process, we rely on the statistical zero-knowledge property of the NIZK systems $\mathcal{NIZK}_\mathsf{S}$ and $\mathcal{NIZK}_\mathsf{D}$, and the statistical hiding property of $\mathcal{COM}$.

Game 0: We start with $\mathbf{Exp}^{\mathsf{sim}|b=1}_{\mathcal{A}}(\lambda)$. In the experiment, the challenger runs the setup algorithm $(\mathsf{PP}^1, \mathsf{msk}^1_\mathsf{X}, \mathsf{msk}^1_\mathsf{P}) \leftarrow \mathsf{Setup}(1^\lambda)$, and returns $(\mathsf{PP}^1, \mathsf{msk}^1_\mathsf{X}, \mathsf{msk}^1_\mathsf{P})$ to $\mathcal{A}$. Regarding all the queries made by $\mathcal{A}$, the challenger replies them honestly. In particular, signature queries and disclosing queries are replied by running $\mathsf{Sign}(\mathbf{x}, \mathsf{sk}_\mathbf{x}, P, \mathsf{Cert}_P, m, w)$ and $\mathsf{Disclose}(m, \Sigma, c^1, F)$, respectively. The adversary can make a polynomial number of queries and outputs a bit b' eventually.

Game 1: This game introduces the following changes to Game 0. At Step 2 of the Setup algorithm, the challenger runs $(\mathsf{crs}_\mathsf{S}, \mathsf{tr}_\mathsf{S}) \leftarrow \mathsf{ZK}_\mathsf{S}.\mathsf{SimSetup}(1^\lambda)$ for $\mathcal{NIZK}_\mathsf{S}$ and $(\mathsf{crs}_\mathsf{D}, \mathsf{tr}_\mathsf{D}) \leftarrow \mathsf{ZK}_\mathsf{D}.\mathsf{SimSetup}(1^\lambda)$ for $\mathcal{NIZK}_\mathsf{D}$. It then outputs $\mathsf{tr} = (\mathsf{tr}_\mathsf{S}, \mathsf{tr}_\mathsf{D})$ in addition to $\mathsf{st} = (\mathsf{PP}^1, \mathsf{msk}^1_\mathsf{X}, \mathsf{msk}^1_\mathsf{P})$. Similar to Game 0, only st is given to the adversary. All the queries are replied exactly the same as in Game 0. These changes are indistinguishable to the adversary due to the statistical zero-knowledge property of $\mathcal{NIZK}_\mathsf{S}$ and $\mathcal{NIZK}_\mathsf{D}$.

Game 2: This game is similar to Game 1 except that, in calls to the $\mathcal{O}^{\mathsf{sor}}_{\mathsf{sign}}$ oracle, the challenger commits to $\mathbf{x}$ and P honestly, but then simulates the proof Π instead of generating it faithfully. Simulating the proof is possible since the challenger has the trapdoor tr_S. Due to the statistical zero-knowledge property of $\mathcal{NIZK}_\mathsf{S}$, this game is statistically indistinguishable from Game 1.

Game 3: This game modifies Game 2 in the following manner. When $\mathcal{A}$ queries the $\mathcal{O}^{\mathsf{sor}}_{\mathsf{Disclose}}$ oracle, the challenger computes $t = F(P, \mathbf{x})$ faithfully. Then the challenger simulates the proof a without employing the witness $\mathsf{clue} = (\mathbf{x}, P, \mathbf{r}_{\mathsf{com},\mathbf{x}}, \mathbf{r}_{\mathsf{com},P})$, i.e., $a \leftarrow \mathsf{ZK}_\mathsf{D}.\mathsf{Sim}(\mathsf{crs}_\mathsf{D}, \mathsf{tr}_\mathsf{D}, (F, t, \mathsf{com}_\mathbf{x}, \mathsf{com}_P, \mathsf{pp}_\mathsf{C}))$. Due to the statistical zero-knowledge property of $\mathcal{NIZK}_\mathsf{D}$, this game is statistically indistinguishable from Game 2.

Game 4: In this game, we make a modification regarding the queries $\mathcal{O}^{\mathsf{sor}}_{\mathsf{sign}}$ again. Instead of committing to real $\mathbf{x}$ and P, we commit to the all-zero string. Specifically, $\mathsf{com}_\mathbf{x}$ and com_P are all commitment to the all-zero string and the proof Π is generated as $\mathsf{ZK}_\mathsf{S}.\mathsf{Sim}(\mathsf{crs}_\mathsf{S}, \mathsf{tr}_\mathsf{S}, (m, \mathsf{vk}_\mathsf{X}, \mathsf{vk}_\mathsf{P}, \mathsf{pp}_\mathsf{C}, \mathsf{com}_\mathbf{x}, \mathsf{com}_P))$. Due

to the statistical hiding property of $\mathcal{COM}$, the views of the adversary in Game 3 and Game 4 are statistically close.

Finally, observe that Game 4 is identical to the experiment $\mathbf{Exp}^{\mathsf{sim}}_{\mathcal{A}}(\lambda)$ with b set to 0. As a result, we can deduce that the two experiments $\mathbf{Exp}^{\mathsf{sim}|b=1}_{\mathcal{A}}(\lambda)$ and $\mathbf{Exp}^{\mathsf{sim}|b=0}_{\mathcal{A}}(\lambda)$ are statistically indistinguishable. This further implies that the advantage of $\mathcal{A}$ in guessing the bit b in experiment $\mathbf{Exp}^{\mathsf{sim}}_{\mathcal{A}}(\lambda)$ is at most negligible in λ. This ends the proof.

□

C.3 Proof of Theorem 2

Proof. To show our generic construction is extractable, we have to prove that the advantages of any PPT adversary $\mathcal{A}$ in experiments $\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda)$, $\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathcal{A}}(\lambda)$, and $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathcal{A}}(\lambda)$ are negligible in λ. Let us first consider the experiment $\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda)$.

Soundness. Suppose $\mathcal{A}$ wins the experiment $\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda)$, in which the challenger $\mathcal{C}$ possess a trapdoor $\mathsf{tr} = (\mathsf{tr}_{\mathsf{S}}, \mathsf{tr}_{\mathsf{D}})$. Then the experiment returns 1 either at Line 7 or Line 9 in Fig. 2. Let (m, Σ) be the output of $\mathcal{A}$. Hence, $\mathsf{Verify}(m, \Sigma) = 1$. At this point, the challenger runs $\mathsf{Extract}(\mathsf{tr}, m, \Sigma)$. In particular, $\mathcal{C}$ runs

$$\eta' \leftarrow \mathsf{ZK}_{\mathsf{S}}.\mathsf{Extr}(\mathsf{crs}_{\mathsf{S}}, \mathsf{tr}_{\mathsf{S}}, (m, \mathsf{vk}_{\mathsf{X}}, \mathsf{vk}_{\mathsf{P}}, \mathsf{pp}_{\mathsf{C}}, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_{P}), \mathit{\Pi}),$$

where $\Sigma = (\mathsf{com}_{\mathbf{x}}, \mathsf{com}_{P}, \mathit{\Pi})$ and $\eta' = (\mathbf{x}', \mathsf{sk}', w', P', \mathsf{Cert}', \mathbf{r}'_{\mathsf{com},\mathbf{x}}, \mathbf{r}'_{\mathsf{com},P})$. If the experiment returns 1 at Line 7, implying $P'(m, \mathbf{x}', w') = 0$. This, however, will break the simulation-soundness of $\mathcal{NIZK}_{\mathsf{S}}$. Thus, the probability of outputting 1 at Line 7 is at most negligible in λ. Hence, $\mathcal{A}$ can only win the experiment by exploiting the condition of Line 9 in Fig. 2.

Let (F, t, a) be the output of $\mathcal{A}$ in this second stage. Then $(m, \Sigma, F, t, a) \notin Q_D$ and $\mathsf{Judge}(\mathsf{PP}, m, \Sigma, F, t, a) = 1$. Now, using tr_{D}, the challenger runs

$$\tilde{\eta} \leftarrow \mathsf{ZK}_{\mathsf{D}}.\mathsf{Extr}(\mathsf{crs}_{\mathsf{D}}, \mathsf{tr}_{\mathsf{D}}, (F, t, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_{P}, \mathsf{pp}_{\mathsf{C}}), a),$$

where $\tilde{\eta} = (\tilde{\mathbf{x}}, \tilde{P}, \tilde{\mathbf{r}}_{\mathsf{com},\mathbf{x}}, \tilde{\mathbf{r}}_{\mathsf{com},P})$. On the one hand, the assumption that $\mathcal{A}$ wins the experiment in the second stage implies that $t \neq F(P', \mathbf{x}')$. On the other hand, $t = F(\tilde{P}, \tilde{\mathbf{x}})$ by the simulation soundness of $\mathcal{NIZK}_{\mathsf{D}}$. This immediately implies that $(P', \mathbf{x}') \neq (\tilde{P}, \tilde{\mathbf{x}})$, which in turn violates the binding property of $\mathcal{COM}$. The reason is that $\mathsf{com}_{\mathbf{x}}$ and com_{P} are valid commitments to $(\mathbf{x}', P')$ and $(\tilde{\mathbf{x}}, \tilde{P})$ due to the simulation soundness of $\mathcal{NIZK}_{\mathsf{S}}$ and $\mathcal{NIZK}_{\mathsf{D}}$, respectively. Therefore, the probability of outputting 1 at Line 9 is also negligible in λ. Thus the probability of $\mathcal{A}$ winning $\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda)$ is negligible in λ, assuming the simulation-sound extractability of two NIZK systems and the binding property of $\mathcal{COM}$.

Unforgeability-I. Let us now consider the experiment $\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathcal{A}}(\lambda)$ and let ϵ_1 be the advantage of $\mathcal{A}$. Our goal is to construct another PPT adversary $\mathcal{B}$ for breaking the $\mathsf{EUF\text{-}CMA}$ security of the signature scheme $\mathcal{S}_{\mathsf{X}}$. Given a verification key vk from the challenger in experiment $\mathbf{Exp}^{\mathsf{uf}}_{\mathcal{S}_{\mathsf{X}},\mathcal{B}}(\lambda)$, $\mathcal{B}$ first performs the following steps.

– Run $(\mathsf{msk}_\mathsf{P}, \mathsf{vk}_\mathsf{P}) \leftarrow \mathsf{S}_\mathsf{P}.\mathsf{Kg}(1^\lambda)$. Set $\mathsf{vk}_\mathsf{X} = \mathsf{vk}$.
– Run $(\mathsf{crs}_\mathsf{S}, \mathsf{tr}_\mathsf{S}) \leftarrow \mathsf{ZK}_\mathsf{S}.\mathsf{SimSetup}(1^\lambda)$ and $(\mathsf{crs}_\mathsf{D}, \mathsf{tr}_\mathsf{D}) \leftarrow \mathsf{ZK}_\mathsf{S}.\mathsf{SimSetup}(1^\lambda)$ for two NIZK systems.
– Run $\mathsf{pp}_\mathsf{C} \leftarrow \mathsf{C}.\mathsf{Setup}(1)^\lambda$.

Set $\mathsf{PP} := (\mathsf{crs}_\mathsf{S}, \mathsf{crs}_\mathsf{D}, \mathsf{pp}_\mathsf{C}, \mathsf{vk}_\mathsf{P}, \mathsf{vk}_\mathsf{X})$ and $\mathsf{tr} = (\mathsf{tr}_\mathsf{S}, \mathsf{tr}_\mathsf{D})$. Then $\mathcal{B}$ triggers $\mathcal{A}$ by sending PP and msk_P to $\mathcal{A}$ as described in $\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathsf{BAPS},\mathcal{A}}(\lambda)$ in Fig. 2. When $\mathcal{A}$ queries the oracle $\mathcal{O}_{\mathsf{AttriKey}}$ with an attribute $\mathbf{x}$, $\mathcal{B}$ queries its own challenger in experiment $\mathbf{Exp}^{\mathsf{uf}}_{\mathcal{S}_\mathsf{X},\mathcal{B}}(\lambda)$, obtaining $\mathsf{sk}_\mathbf{x}$, which is then passed to $\mathcal{A}$. Observe that $\mathcal{B}$ can also easily answer all the queries to the oracles $\mathcal{O}_{\mathsf{Sign}}$ and $\mathcal{O}_{\mathsf{Dis}}$ since $\mathcal{B}$ holds the two trapdoors tr_S and tr_D. This is how $\mathcal{B}$ simulates the views of $\mathcal{A}$ in experiment $\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathsf{BAPS},\mathcal{A}}(\lambda)$.

Eventually, $\mathcal{A}$ outputs a forgery (m, Σ). Suppose $\mathcal{A}$ wins the experiment $\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathsf{BAPS},\mathcal{A}}(\lambda)$. We will show how $\mathcal{B}$ employs such a forgery for BAPS to find a forgery for $\mathcal{S}_\mathsf{X}$. Since $\mathcal{A}$ wins, then (m, Σ) is a valid message-signature pair and is never obtained from the oracle $\mathcal{O}_{\mathsf{Sign}}$ query. Now $\mathcal{B}$ can run $\mathsf{Extract}(\mathsf{tr}, m, \Sigma)$, obtaining a tuple $\eta' = (\mathbf{x}', \mathsf{sk}', w', P', \mathsf{Cert}', \mathbf{r}'_{\mathsf{com},\mathbf{x}}, \mathbf{r}'_{\mathsf{com},P})$. With overwhelming probability, $P'(m, \mathbf{x}', w') = 1$ and $\mathsf{S}_\mathsf{X}.\mathsf{Ver}(\mathsf{vk}_\mathsf{X}, \mathbf{x}', \mathsf{sk}') = 1$, thanks to the simulation-soundness of the $\mathcal{NIZK}_\mathsf{S}$ system. The fact that $\mathcal{A}$ wins the experiment implies that $\mathbf{x}' \notin Q_X$. In other words, $\mathcal{B}$ never queries its challenger on $\mathbf{x}'$. Thus $(\mathbf{x}', \mathsf{sk}')$ is a valid forgery of $\mathcal{S}_\mathsf{X}$. Therefore, if ϵ_1 is non-negligible in λ, $\mathcal{B}$ is able to find a valid forgery of the $\mathcal{S}_\mathsf{X}$ scheme with non-negligible probability as well. Due to the security of the $\mathcal{S}_\mathsf{X}$ scheme, ϵ_1 is negligible.

Unforgeability-II. Let ϵ_2 be the advantage of $\mathcal{A}$ in the experiment $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathcal{A}}(\lambda)$. We construct a PPT adversary $\hat{\mathcal{B}}$ for breaking the $\mathsf{EUF\text{-}CMA}$ security of the signature scheme $\mathcal{S}_\mathsf{P}$. Given a verification key vk from the challenger in experiment $\mathbf{Exp}^{\mathsf{uf}}_{\mathcal{S}_\mathsf{P},\hat{\mathcal{B}}}(\lambda)$, $\hat{\mathcal{B}}$ first performs the following steps.

– Set $\mathsf{vk}_\mathsf{P} = \mathsf{vk}$, and then run $(\mathsf{msk}_\mathsf{X}, \mathsf{vk}_\mathsf{X}) \leftarrow \mathsf{S}_\mathsf{X}.\mathsf{Kg}(1^\lambda)$.
– Run $(\mathsf{crs}_\mathsf{S}, \mathsf{tr}_\mathsf{S}) \leftarrow \mathsf{ZK}_\mathsf{S}.\mathsf{SimSetup}(1^\lambda)$ and $(\mathsf{crs}_\mathsf{D}, \mathsf{tr}_\mathsf{D}) \leftarrow \mathsf{ZK}_\mathsf{S}.\mathsf{SimSetup}(1^\lambda)$ for two NIZK systems.
– Run $\mathsf{pp}_\mathsf{C} \leftarrow \mathsf{C}.\mathsf{Setup}(1)^\lambda$.

Set $\mathsf{PP} := (\mathsf{crs}_\mathsf{S}, \mathsf{crs}_\mathsf{D}, \mathsf{pp}_\mathsf{C}, \mathsf{vk}_\mathsf{P}, \mathsf{vk}_\mathsf{X})$ and $\mathsf{tr} = (\mathsf{tr}_\mathsf{S}, \mathsf{tr}_\mathsf{D})$. Then $\mathcal{B}$ invokes $\mathcal{A}$ by sending PP and msk_X to $\mathcal{A}$ as described in $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathsf{BAPS},\mathcal{A}}(\lambda)$ in Fig. 2. When $\mathcal{A}$ queries the oracle $\mathcal{O}_{\mathsf{PolicyKey}}$ with a policy P, $\hat{\mathcal{B}}$ turns to its own challenger in experiment $\mathbf{Exp}^{\mathsf{uf}}_{\mathcal{S}_\mathsf{P},\hat{\mathcal{B}}}(\lambda)$, obtaining a signature Cert_P on P. Next, $\hat{\mathcal{B}}$ relays Cert_P to $\mathcal{A}$. Regarding queries to $\mathcal{O}_{\mathsf{Sign}}$ and $\mathcal{O}_{\mathsf{Dis}}$, $\hat{\mathcal{B}}$ employs the corresponding trapdoors. This is how $\hat{\mathcal{B}}$ simulates the views of $\mathcal{A}$ in experiment $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathsf{BAPS},\mathcal{A}}(\lambda)$.

Eventually, $\mathcal{A}$ outputs a forgery (m, Σ). Suppose $\mathcal{A}$ wins the experiment $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathsf{BAPS},\mathcal{A}}(\lambda)$. Then (m, Σ) is a valid message-signature pair and is not from the oracle $\mathcal{O}_{\mathsf{Sign}}$ query. Now $\hat{\mathcal{B}}$ runs the $\mathsf{Extract}(\mathsf{tr}, m, \Sigma)$ algorithm, obtaining a tuple $\eta' = (\mathbf{x}', \mathsf{sk}', w', P', \mathsf{Cert}', \mathbf{r}'_{\mathsf{com},\mathbf{x}}, \mathbf{r}'_{\mathsf{com},P})$. Due to the simulation-soundness of the $\mathcal{NIZK}_\mathsf{S}$ system, $P'(m, \mathbf{x}', w') = 1$ and $\mathsf{S}_\mathsf{P}.\mathsf{Ver}(\mathsf{vk}, P', \mathsf{Cert}') = 1$ with overwhelming probability. By the winning condition of $\mathcal{A}$, $P' \notin Q_P$. Thus, (P', Cert')

is a valid forgery for $\mathcal{S}_{\mathsf{P}}$. Therefore, if ϵ_2 is non-negligible in λ, $\hat{\mathcal{B}}$ is able to find a valid forgery of the $\mathcal{S}_{\mathsf{P}}$ scheme with non-negligible probability as well. Due to the security of the $\mathcal{S}_{\mathsf{P}}$ scheme, ϵ_2 is negligible.

□

D Previous Extension and Permutation Techniques

In this section, we recall some previous extension and permutation techniques that will be used in reducing the considered statements to instances of $\mathcal{R}_{\mathsf{abstract}}$.

To prove knowledge of some secret vector such that a statement is true, our strategy is to reduce the considered statement to an instance of $\mathcal{R}_{\mathsf{abstract}}$. As we see in Section B.5, the abstracted Stern protocol can only handle secret $\mathbf{w}$ such that the conditions in (18) are conformed. To this end, decomposition, extension, and permutation techniques are then proposed in various previous works, such as [41,37,34,49,19]. We will recall some of them, which will be used in this work.

Techniques for proving $\mathbf{w} \in \{0,1\}^{\mathfrak{m}}$. To prove the knowledge of a binary vector $\mathbf{w} \in \{0,1\}^{\mathfrak{m}}$, define the extension process and permutation as follows.

- For a binary vector $\mathbf{w} = (\, w_0 \,, \ldots \,, w_{\mathfrak{m}-1} \,)^\top \in \{0,1\}^{\mathfrak{m}}$, where $\mathfrak{m} \in \mathbb{Z}^+$, denote by $\mathsf{ext}_2(\mathbf{w})$ the vector $(\, \overline{w}_0 \,, w_0 \,, \ldots \,, \overline{w}_{\mathfrak{m}-1} \,, w_{\mathfrak{m}-1} \,)^\top \in \{0,1\}^{2\mathfrak{m}}$.
- Let $\mathbf{J}^\star_{\mathfrak{m}} \in \mathbb{Z}_q^{\mathfrak{m} \times 2\mathfrak{m}}$ be an extension of the identity matrix $\mathbf{I}_{\mathfrak{m}}$, obtained by inserting a zero-column $\mathbf{0}^{\mathfrak{m}}$ right before each column of $\mathbf{I}_{\mathfrak{m}}$. We have for $\mathbf{w} \in \{0,1\}^{\mathfrak{m}}$,

$$\mathbf{w} = \mathbf{J}^\star_{\mathfrak{m}} \cdot \mathsf{ext}_2(\mathbf{w}). \tag{21}$$

- For $\mathbf{f} = (\, f_0 \,, \ldots \,, f_{\mathfrak{m}-1} \,)^\top \in \{0,1\}^{\mathfrak{m}}$, define the permutation $F_{\mathsf{bin}}(\mathbf{f}, \cdot)$ that transforms vector $\mathbf{z} = (\, z_{0,0} \,, z_{0,1} \,, \ldots \,, z_{\mathfrak{m}-1,0} \,, z_{\mathfrak{m}-1,1} \,)^\top \in \mathbb{Z}_q^{2\mathfrak{m}}$ into:

$$F_{\mathsf{bin}}(\mathbf{f}, \mathbf{z}) = (\, z_{0,f_0} \,, z_{0,\bar{f}_0} \,, \ldots \,, z_{\mathfrak{m}-1,f_{\mathfrak{m}-1}} \,, z_{\mathfrak{m}-1,\bar{f}_{\mathfrak{m}-1}} \,)^\top.$$

  Note that, for any $\mathbf{f}, \mathbf{w} \in \{0,1\}^{\mathfrak{m}}$, we have:

$$\mathbf{z} = \mathsf{ext}_2(\mathbf{w}) \iff F_{\mathsf{bin}}(\mathbf{f}, \mathbf{z}) = \mathsf{ext}_2(\mathbf{w} \oplus \mathbf{f}). \tag{22}$$

The above equivalence (22) suffices for proving knowledge of a binary vector $\mathbf{w}$ such that $\mathbf{M} \cdot \mathbf{w} = \mathbf{v}$ for some public $\mathbf{M}, \mathbf{v}$. We can simply define the valid set as $\mathsf{VALID}_{\mathsf{bin}} = \{\mathbf{z} : \exists\ \mathbf{w} \in \{0,1\}^{\mathfrak{m}}$ such that $\mathbf{z} = \mathsf{ext}_2(\mathbf{w})\}$, let $\mathcal{S}_{\mathsf{bin}} = \{0,1\}^{\mathfrak{m}}$ and its associated permutation be $\{\Gamma_\phi = F_{\mathsf{bin}}(\mathbf{f}, \cdot),\ \phi = \mathbf{f} \in \mathcal{S}_{\mathsf{bin}}\}$. It is easy to see that the desired properties in (18) are achieved and thus we can run the interactive protocol in Table 2. We also remark that vector $\mathbf{f}$ serves as a "one-time pad" that perfects hides $\mathbf{w}$. If we have to show that $\mathbf{w}$ appears in other equations, we can use the same $\mathbf{f}$ at those places to enforce consistency.

Techniques for proving $\mathbf{w} \in \{-1,0,1\}^{\mathfrak{m}}$. For any integer w, let $[a]_3$ be $a' \in \{-1,0,1\}$ such that $a = a' \bmod 3$. To prove knowledge of a vector over $\{-1,0,1\}$, define the following extension and permutation.

– Let $\mathbf{w} = (w_0, \ldots, w_{\mathsf{m}-1})^\top \in \{-1,0,1\}^{\mathsf{m}}$, let $\mathsf{ext}_3(\mathbf{w})$ be the extended vector $(\,[w_0+1]_3\,,\,[w_0]\,,[w_0-1]_3\,\ldots\,,\,[w_{\mathsf{m}-1}+1]_3\,,[w_{\mathsf{m}-1}]_3,\,[w_{\mathsf{m}-1}-1]_3\,)^\top \in \{-1,0,1\}^{3\mathsf{m}}$.
– Let $\mathbf{I}^\star_{\mathsf{m}} \in \mathbb{Z}_q^{\mathsf{m}\times 3\mathsf{m}}$ be an extension of the identity matrix $\mathbf{I}_{\mathsf{m}}$, obtained by inserting a zero-column $\mathbf{0}^{\mathsf{m}}$ before and after each column of $\mathbf{I}_{\mathsf{m}}$. We have for $\mathbf{w} \in \{-1,0,1\}^{\mathsf{m}}$,

$$\mathbf{w} = \mathbf{I}^\star_{\mathsf{m}} \cdot \mathsf{ext}_3(\mathbf{w}). \tag{23}$$

– For $\mathbf{f} = (\,f_0\,,\,\ldots\,,\,f_{\mathsf{m}-1}\,)^\top \in \{-1,0,1\}^{\mathsf{m}}$, define the permutation $F_{\mathsf{tri}}(\mathbf{f},\cdot)$ that transforms $\mathbf{z} = (\,z_{0,-1},\,z_{0,0}\,,\,z_{0,1}\,,\,\ldots\,,z_{\mathsf{m}-1,-1},\,z_{\mathsf{m}-1,0}\,,\,z_{\mathsf{m}-1,1}\,)^\top \in \mathbb{Z}_q^{3\mathsf{m}}$ into:

$$F_{\mathsf{tri}}(\mathbf{f},\mathbf{z}) = (\,z_{0,[-f_0-1]_3}\,,\,z_{0,[-f_0]_3}\,,\,z_{0,[-f_0+1]_3}\,,\,\ldots\,,$$
$$z_{\mathsf{m}-1,[-f_{\mathsf{m}-1}-1]_3}\,,\,z_{\mathsf{m}-1,[-f_{\mathsf{m}-1}]_3},z_{\mathsf{m}-1,[-f_{\mathsf{m}-1}+1]_3}\,)^\top.$$

Note that, for any $\mathbf{f},\mathbf{w} \in \{-1,0,1\}^{\mathsf{m}}$, we have:

$$\mathbf{z} = \mathsf{ext}_3(\mathbf{w}) \iff F_{\mathsf{tri}}(\mathbf{f},\mathbf{z}) = \mathsf{ext}_3([\mathbf{w}+\mathbf{f}]_3). \tag{24}$$

Techniques for proving $\mathbf{w} = \mathbf{x} \odot \mathbf{y} = (x_0 \cdot y_0, \ldots, x_{\mathsf{m}-1} \cdot y_{\mathsf{m}-1}) \in \{0,1\}^{\mathsf{m}}$. To show the well-formedness of a vector $\mathbf{w} \in \{0,1\}^{\mathsf{m}}$ such that it is the component-wise multiplication of two vectors $\mathbf{x} = (x_0,\ldots,x_{\mathsf{m}-1})^\top, \mathbf{y} = (y_0,\ldots,y_{\mathsf{m}-1})^\top \in \{0,1\}^{\mathsf{m}}$, the following techniques are used.

– Let the extended vector of $\mathbf{w}$ be $\mathsf{ext}_{\mathsf{mult}}(\mathbf{w})$ or $\mathsf{ext}_{\mathsf{mult}}(\mathbf{x},\mathbf{y}) \in \{0,1\}^{4\mathsf{m}}$ with the following form:

$$\big(\overline{x}_0 \cdot \overline{y}_0\,,\,\overline{x}_0 \cdot y_0\,,\,x_0 \cdot \overline{y}_0\,,\,x_0 \cdot y_0\,,\,\ldots\,,$$
$$\overline{x}_{\mathsf{m}-1} \cdot \overline{y}_{\mathsf{m}-1}\,,\,\overline{x}_{\mathsf{m}-1} \cdot y_{\mathsf{m}-1}\,,\,x_{\mathsf{m}-1} \cdot \overline{y}_{\mathsf{m}-1}\,,\,x_{\mathsf{m}-1} \cdot y_{\mathsf{m}-1}\big).$$

– Let $\mathbf{K}^\star_{\mathsf{m}} \in \mathbb{Z}_q^{\mathsf{m}\times 4\mathsf{m}}$ be an extension of the identity matrix $\mathbf{I}_{\mathsf{m}}$, obtained by inserting three zero-columns $\mathbf{0}^{\mathsf{m}}$ before each column of $\mathbf{I}_{\mathsf{m}}$. We have for $\mathbf{w} \in \{0,1\}^{\mathsf{m}}$,

$$\mathbf{w} = \mathbf{K}^\star_{\mathsf{m}} \cdot \mathsf{ext}_{\mathsf{mult}}(\mathbf{w}). \tag{25}$$

– For $\mathbf{f} = (\,f_0\,,\,\ldots\,,\,f_{\mathsf{m}-1}\,)^\top \in \{0,1\}^{\mathsf{m}}$ and $\mathbf{g} = (\,g_0\,,\,\ldots\,,\,g_{\mathsf{m}-1}\,)^\top \in \{0,1\}^{\mathsf{m}}$, define the permutation $F_{\mathsf{mult}}((\mathbf{f},\mathbf{g}),\cdot)$ that transforms

$$\mathbf{z} = (z_0^{(0,0)}, z_0^{(0,1)}, z_0^{(1,0)}, z_0^{(1,1)}, \ldots, z_{\mathsf{m}-1}^{(0,0)}, z_{\mathsf{m}-1}^{(0,1)}, z_{\mathsf{m}-1}^{(1,0)}, z_{\mathsf{m}-1}^{(1,1)})^\top \in \mathbb{Z}_q^{4\mathsf{m}}$$

into $F_{\mathsf{mult}}((\mathbf{f},\mathbf{g}),\mathbf{z})$ of the following form

$$F_{\mathsf{mult}}((\mathbf{f},\mathbf{g}),\mathbf{z}) = \big(\; z_0^{(f_0,g_0)}, z_0^{(f_0,\overline{g}_0)}, z_0^{(\overline{f}_0,g_0)}, z_0^{(\overline{f}_0,\overline{g}_0)}, \ldots,$$
$$z_{\mathsf{m}-1}^{(f_{\mathsf{m}-1},g_{\mathsf{m}-1})}, z_{\mathsf{m}-1}^{(f_{\mathsf{m}-1},\overline{g}_{\mathsf{m}-1})}, z_{\mathsf{m}-1}^{(\overline{f}_{\mathsf{m}-1},g_{\mathsf{m}-1})}, z_{\mathsf{m}-1}^{(\overline{f}_{\mathsf{m}-1},\overline{g}_{\mathsf{m}-1})}\big)^\top$$

Note that, for any $\mathbf{f},\mathbf{g},\mathbf{x},\mathbf{y} \in \{0,1\}^{\mathsf{m}}$, we have:

$$\mathbf{z} = \mathsf{ext}_{\mathsf{mult}}(\mathbf{x} \odot \mathbf{y}) \iff F_{\mathsf{mult}}((\mathbf{f},\mathbf{g}),\mathbf{z}) = \mathsf{ext}_{\mathsf{mult}}((\mathbf{x} \oplus \mathbf{f}) \odot (\mathbf{g} \oplus \mathbf{y})). \tag{26}$$

The above techniques are critical in proving knowledge of binary vectors $\mathbf{x}, \mathbf{y}, \mathbf{w} \in \{0,1\}^{\mathfrak{m}}$ such that $\mathbf{w} = \mathbf{x} \odot \mathbf{y}$ and $\mathbf{x}, \mathbf{y}, \mathbf{w}$ all appear in a unified equation $\mathbf{M}' \cdot \mathbf{w}' = \mathbf{v} \bmod q$, where $\mathbf{w}' = (\mathbf{x}^\top, \mathbf{y}^\top, \mathbf{w}^\top)^\top$.

To this end, the prover extends $\mathbf{w}'$ to $\mathsf{ext}_{\mathsf{merge}}(\mathbf{x}, \mathbf{y}) \stackrel{\triangle}{=} \mathsf{ext}_{\mathsf{merge}}(\mathbf{w}')$ of the following

$$(\mathsf{ext}_2(\mathbf{x})^\top \mid \mathsf{ext}_2(\mathbf{y})^\top \mid \mathsf{ext}_{\mathsf{mult}}(\mathbf{x}, \mathbf{y})^\top)^\top \in \{0,1\}^{8\mathfrak{m}}.$$

Next, define a permutation $F_{\mathsf{merge}}((\mathbf{f}, \mathbf{g}), \cdot)$ for two vectors $\mathbf{f}, \mathbf{g} \in \{0,1\}^{\mathfrak{m}}$ as follows. When applying to vectors $\mathbf{z} = (\mathbf{z}_0^\top, \mathbf{z}_1^\top, \mathbf{z}_2^\top)^\top \in \mathbb{Z}_q^{8\mathfrak{m}}$, with each block having appropriate sizes, it maps $\mathbf{z}$ to

$$F_{\mathsf{merge}}((\mathbf{f}, \mathbf{g}), \mathbf{z}) = \big(F_{\mathsf{bin}}(\mathbf{f}, \mathbf{z}_0)^\top \mid F_{\mathsf{bin}}(\mathbf{g}, \mathbf{z}_1)^\top \mid F_{\mathsf{mult}}((\mathbf{f}, \mathbf{g}), \mathbf{z}_2)^\top\big)^\top. \quad (27)$$

Due to the equivalences in (22), (26), one can see that the following equivalence holds as well.

$$\mathbf{z} = \mathsf{ext}_{\mathsf{merge}}(\mathbf{x}, \mathbf{y}) \Longleftrightarrow F_{\mathsf{merge}}((\mathbf{f}, \mathbf{g}), \mathbf{z}) = \mathsf{ext}_{\mathsf{merge}}(\mathbf{x} \oplus \mathbf{f}, \mathbf{y} \oplus \mathbf{g}). \quad (28)$$

Techniques for proving $w = x \cdot y$ for $x \in \{0,1\}$ and $y \in \{-1,0,1\}$. To show that w is well-formed, we recall the following techniques from [42].

- For any $x \in \{0,1\}$ and $y \in \{-1,0,1\}$, let vector $\mathsf{ext}_{\mathsf{mig}}(x, y) \in \{-1,0,1\}^6$ be of the following form:

$$\big(\, \overline{x} \cdot [y{+}1]_3,\ \ x \cdot [y{+}1]_3,\ \ \overline{x} \cdot [y]_3,\ \ x \cdot [y]_3,\ \ \overline{x} \cdot [y{-}1]_3,\ \ x \cdot [y{-}1]_3 \,\big)^\top.$$

- Let $f \in \{0,1\}$ and $g \in \{-1,0,1\}$, define the permutation $F_{\mathsf{mig}}((f, g), \cdot)$ associated with f, g as follows. It transforms vector

$$\mathbf{z} = \big(z^{(0,-1)}, z^{(1,-1)}, z^{(0,0)}, z^{(1,0)}, z^{(0,1)}, z^{(1,1)}\big)^\top \in \mathbb{Z}^6$$

into vector $F_{\mathsf{mig}}((f, g), \cdot)$ of form

$$\big(z^{(f,[-g-1]_3)},\ z^{(\overline{f},[-g-1]_3)},\ z^{(f,[-e]_3)},\ z^{(\overline{f},[-g]_3)},\ z^{(f,[-e+1]_3)},\ z^{(\overline{f},[-g+1]_3)}\big)^\top.$$

It can be easily checked that for any $x, f \in \{0,1\}$ and any $y, g \in \{-1,0,1\}$, the following equivalence is satisfied.

$$\mathbf{z} = \mathsf{ext}_{\mathsf{mig}}(x, y) \iff F_{\mathsf{mig}}((f, g), \mathbf{z}) = \mathsf{ext}_{\mathsf{mig}}(\, x \oplus f,\ [y + g]_3 \,). \quad (29)$$

Techniques for handling a "mix" vector. We now recall the techniques for proving knowledge of a vector $\mathbf{w}$ of the form:

$$\mathbf{w} = (\mathbf{y}^\top \mid\ t_0 \cdot \mathbf{y}^\top \mid \cdots \mid\ t_{c-1} \cdot \mathbf{y}^\top)^\top \in \{-1,0,1\}^{c\mathfrak{m}+\mathfrak{m}}, \quad (30)$$

where $\mathbf{t} = (t_0, \ldots, t_{c-1})^\top \in \{0,1\}^c$ and $\mathbf{y} = (y_0, \ldots, y_{\mathfrak{m}-1}) \in \{-1,0,1\}^{\mathfrak{m}}$. This is actually extended from previous techniques.

– Extend $\mathbf{w}$ to $\mathsf{ext}_{\mathsf{mix}}(\mathbf{w}) \stackrel{\triangle}{=} \mathsf{ext}_{\mathsf{mix}}(\mathbf{t},\mathbf{y}) \in \{-1,0,1\}^{6c\mathfrak{m}+3\mathfrak{m}}$ of the following form:

$$\big(\,\mathsf{ext}_3(\mathbf{y})^\top \mid \mathsf{ext}_{\mathsf{mig}}(t_0,y_0)^\top \mid \cdots \mid \mathsf{ext}_{\mathsf{mig}}(t_0,y_{\mathfrak{m}-1})^\top \mid \cdots \mid \mathsf{ext}_{\mathsf{mig}}(t_{c-1},y_0)^\top \mid \cdots \mid \mathsf{ext}_{\mathsf{mig}}(t_{c-1},y_{\mathfrak{m}-1})^\top \,\big)^\top.$$

– Next, for $\mathbf{f} \in \{0,1\}^c$ and $\mathbf{g} \in \{-1,0,1\}^{\mathfrak{m}}$, define the permutation $F_{\mathsf{mix}}((\mathbf{f},\mathbf{g}),\cdot)$ as follows. It permutes vector

$$\mathbf{z} = \big(\mathbf{z}_{-1}^\top \mid \mathbf{z}_{0,0}^\top \mid \cdots \mid \mathbf{z}_{0,\mathfrak{m}-1}^\top \mid \cdots \mid \mathbf{z}_{c-1,0}^\top \mid \cdots \mid \mathbf{z}_{c-1,\mathfrak{m}-1}^\top\big)^\top \in \mathbb{Z}^{6c\mathfrak{m}+3\mathfrak{m}}$$

into $F_{\mathsf{mix}}((\mathbf{f},\mathbf{g}),\mathbf{z})$ of the following form

$$\big(F_{\mathsf{tri}}(\mathbf{g},\mathbf{z}_{-1})^\top \mid F_{\mathsf{mig}}((f_0,g_0),\mathbf{z}_{0,0})^\top \mid \cdots \mid F_{\mathsf{mig}}((f_0,g_{\mathfrak{m}-1}),\mathbf{z}_{0,\mathfrak{m}-1})^\top \mid \cdots \mid F_{\mathsf{mig}}((f_{c-1},g_0),\mathbf{z}_{c-1,0})^\top \mid \cdots \mid F_{\mathsf{mig}}((f_{c-1},g_{\mathfrak{m}-1}),\mathbf{z}_{c-1,\mathfrak{m}-1})^\top\big)^\top.$$

Due to equivalence (24) and (29), the following equivalence holds

$$\mathbf{z} = \mathsf{ext}_{\mathsf{mix}}(\mathbf{t},\mathbf{y}) \iff F_{\mathsf{mix}}((\mathbf{f},\mathbf{g}),\mathbf{z}) = \mathsf{ext}_{\mathsf{mix}}(\,\mathbf{t}\oplus\mathbf{f},\,[\mathbf{y}+\mathbf{g}]_3\,) \tag{31}$$

for any $\mathbf{t},\mathbf{f} \in \{0,1\}^c$ and any $\mathbf{y},\mathbf{g} \in \{-1,0,1\}^{\mathfrak{m}}$.

Let $\mathbf{w} = (w_0,\ldots,w_{c-1})^\top \in \{0,1\}^c$. A regular word defined by $\mathbf{w}$, denoted as $\Delta_c(\mathbf{w}) \in \{0,1\}^{2^c}$, is a vector that has a sole 1 in t-th position with $t = \sum_{j=0}^{c-1}(2^{c-1-j}\cdot w_j)$.

Techniques for proving z such that $\mathbf{z} = \Delta_c(\mathbf{w})$ for $\mathbf{w} \in \{0,1\}^c$. Let $\mathbf{w} = (w_0,\ldots,w_{c-1})^\top \in \{0,1\}^c$. Compute $w = \sum_{j=0}^{c-1} 2^{c-1-i}\cdot w_j \in [0,2^c-1]$. Recall that $\mathbf{z} = \Delta_c(\mathbf{w}) \in \{0,1\}^{2^c}$ has a sole 1 in its w-th position. To prove knowledge of a such vector $\mathbf{z}$, we recall the permutation techniques first presented by Nguyen et al. [49].

For $\mathbf{f} = (\,f_0\,,\cdots,\,f_{c-1}\,)^\top \in \{0,1\}^c$, define $F_{\mathsf{reg}}(\mathbf{f},\cdot) : \{0,1\}^{2^c} \mapsto \{0,1\}^{2^c}$ as follows. It transforms vector

$$\mathbf{z} = (\,z_{0,0,\ldots,0}\,,\cdots,\,z_{i_0,\ldots,i_{c-1}}\,,\cdots,\,z_{1,1,\ldots,1}\,)^\top \in \{0,1\}^{2^c}$$

into vector $F_{\mathsf{reg}}(\mathbf{f},\mathbf{z}) = (\,z'_{0,0,\ldots,0}\,,\cdots,\,z'_{1,1,\ldots,1}\,)^\top$, where for each $(i_0,\ldots,i_{c-1})^\top \in \{0,1\}^c$, we have $z_{i_0,\ldots,i_{c-1}} = z'_{i_0\oplus f_0,\ldots,i_{c-1}\oplus f_{c-1}}$. It is verifiable that for any $\mathbf{w},\mathbf{f} \in \{0,1\}^c$, we have:

$$\mathbf{z} = \Delta_c(\mathbf{w}) \iff F_{\mathsf{reg}}(\mathbf{f},\mathbf{z}) = \Delta_c(\mathbf{w}\oplus\mathbf{f}). \tag{32}$$

The above equivalence also plays a crucial rule in proving knowledge of $\mathbf{z}$ such that $\mathbf{z}$ satisfies some equation(s). We remark that, however, the above permutation F_{reg} only makes sense if $\mathbf{w}$ satisfies some constraints as well. This is similar to previous techniques $\mathsf{ext}_{\mathsf{mult}}$ and F_{mult}. Also, this is the reason that we introduce a redundant vector $(b'_{i,0},\ldots,b'_{i,1})^\top$ in equations (10).

E Deferred Details of the Main Zero-Knowledge Protocol

In this section we present how to reduce the five substatements involving equations (3), (4), (5), (10) and (11) to instances of $\mathcal{R}_{\mathsf{abstract}}$, as required in Section 4.4.

E.1 The Substatement Involving (3)

To reduce this statement to an instance of $\mathcal{R}_{\mathsf{abstract}}$, it is actually proving knowledge of a message-signature pair for the Ducas-Micciancio signature scheme [15]. Ling et al. [42] first gave such a proof of knowledge $(\mathbf{x}, t_{\mathbf{x}}, \mathbf{r}_{\mathbf{x}}, \mathbf{v}_{\mathbf{x}})$ such that equations (33) hold.

$$\begin{cases} [\mathbf{A}_{\mathsf{X}}, \mathbf{A}_{\mathsf{X},[0]} + \sum_{j=1}^{d} \mathbf{A}_{\mathsf{X},[i]} t_{\mathbf{x},[i]}] \cdot \mathbf{v}_{\mathbf{x}} = u_{\mathsf{X}} + \mathbf{F}_{\mathsf{X}} \cdot \mathbf{e}_{\mathbf{x}}; \\ \mathbf{e}_{\mathbf{x}} = \mathsf{rdec}(\mathbf{F}_{\mathsf{X},0} \cdot \mathbf{r}_{\mathbf{x}} + \mathbf{F}_{\mathsf{X},1} \cdot \mathbf{x}) \in \{-1,0,1\}^{n\ell}; \\ t_{\mathbf{x}} \in \{0,1\}^{c_d}; \mathbf{x} \in \{0,1\}^{nk_2'}; \mathbf{r}_{\mathbf{x}} \in (R)^{\overline{m}}; \|\mathbf{r}_{\mathbf{x}}\|_{\infty} \leq \beta; \\ \mathbf{v}_{\mathbf{x}} \in (R)^{\overline{m}+k}; \|\mathbf{v}_{\mathbf{x}}\|_{\infty} \leq \beta. \end{cases} \tag{33}$$

The First Step-Decomposition. Let $\mathbf{v}_{\mathbf{x}} = (\mathbf{v}_{\mathbf{x},0}^{\top} \mid \mathbf{v}_{\mathbf{x},1}^{\top})^{\top}$ with $\mathbf{v}_{\mathbf{x},0}^{\top} \in (R)^{\overline{m}}$ and $\mathbf{v}_{\mathbf{x},1}^{\top} \in (R)^{k}$. Since $\mathbf{v}_{\mathbf{x},0}$, $\mathbf{v}_{\mathbf{x},1}$, and $\mathbf{r}_{\mathbf{x}}$ are β bounded, we first perform the following extensions.

- Let $\mathbf{v}'_{\mathbf{x},0} = \mathsf{rdec}_{\beta}(\mathbf{v}_{\mathbf{x},0}) \in \{-1,0,1\}^{n\overline{m}\delta_{\beta}}$, $\mathbf{v}'_{\mathbf{x},1} = \mathsf{rdec}_{\beta}(\mathbf{v}_{\mathsf{x},1}) \in \{-1,0,1\}^{nk\delta_{\beta}}$, and $\mathbf{r}'_{\mathbf{x}} = \mathsf{rdec}_{\beta}(\mathbf{r}_{\mathbf{x}}) \in \{-1,0,1\}^{n\overline{m}\delta_{\beta}}$.

Due to the equations observed in (13) and (16), we have the following equivalent formulas

$$\begin{cases} [\mathsf{rot}(\mathbf{A}_{\mathsf{X},[0]}) \cdot \mathbf{H}_{k,\beta}] \cdot \mathbf{v}'_{\mathbf{x},1} + \sum_{i=1}^{d} \sum_{j=c_{i-1}}^{c_i-1} [\mathsf{rot}(\mathbf{A}_{\mathsf{X},[i]} \cdot X^j) \cdot \mathbf{H}_{k,\beta}] \cdot t_{\mathbf{x},j} \cdot \mathbf{v}'_{\mathbf{x},1} \\ \qquad + [\mathsf{rot}(\mathbf{A}_{\mathsf{x}}) \cdot \mathbf{H}_{\overline{m},\beta}] \cdot \mathbf{v}'_{\mathbf{x},0} - [\mathsf{rot}(\mathbf{F}_{\mathsf{X}})] \cdot \mathbf{e}_{\mathbf{x}} = \tau(u_{\mathsf{X}}) \bmod q, \\ [\mathsf{rot}(\mathbf{F}_{\mathsf{X},0}) \cdot \mathbf{H}_{\overline{m},\beta}] \cdot \mathbf{r}'_{\mathbf{x}} + [\mathsf{rot}(\mathbf{F}_{\mathsf{X},1})] \cdot \mathbf{x} - [\mathbf{H}] \cdot \mathbf{e}_{\mathbf{x}} = \mathbf{0} \bmod q. \end{cases} \tag{34}$$

Let $L'_{\mathsf{X}} = 2n\overline{m}\delta_{\beta} + n\ell + nk'_2 + nk\delta_{\beta}(1+c_d)$. Form secret vector $\mathbf{w}'_{\mathsf{X}} \in \{-1,0,1\}^{L'_{\mathsf{X}}}$ that is column concatenation of the following vectors.

$$\begin{cases} \mathbf{w}'_{\mathsf{X},0} = (\ \mathbf{v}'^{\top}_{\mathbf{x},0} \mid \ \mathbf{r}'^{\top}_{\mathbf{x}} \mid \ \mathbf{e}^{\top}_{\mathbf{x}}\ ) \in \{-1,0,1\}^{2n\overline{m}\delta_{\beta}+n\ell}, \\ \mathbf{w}'_{\mathsf{X},1} = \mathbf{x} \in \{0,1\}^{nk'_2}, \\ \mathbf{w}'_{\mathsf{X},2} = (\ \mathbf{v}'^{\top}_{\mathbf{x},1} \mid\ t_{\mathbf{x},0} \cdot \mathbf{v}'^{\top}_{\mathbf{x},1} \mid \cdots \mid\ t_{\mathbf{x},c_d-1} \cdot \mathbf{v}'^{\top}_{\mathbf{x},1}\ )^{\top} \in \{-1,0,1\}^{nk\delta_{\beta}(1+c_d)}. \end{cases}$$

Through some basic algebra, one can obtain from (34) an unifying equation of the form

$$\mathbf{M}'_{\mathsf{X}} \cdot \mathbf{w}'_{\mathsf{X}} = \widetilde{\mathbf{v}}_{\mathsf{X}} \text{ for some vector } \widetilde{\mathbf{v}}_{\mathsf{X}}. \tag{35}$$

The Second Step-Extension and Permutation. We now employ the techniques described in Section D to reduce (35) to an instance of $\mathcal{R}_{\mathsf{abstract}}$.

We first perform the following extension.

$$\begin{cases} \mathbf{w}_{\mathsf{X},0} = \mathsf{ext}_3(\mathbf{w}'_{\mathsf{X},0}) \in \{-1,0,1\}^{3(2n\overline{m}\delta_\beta + n\ell)}; \\ \mathbf{w}_{\mathsf{X},1} = \mathsf{ext}_2(\mathbf{w}'_{\mathsf{X},1}) \in \{0,1\}^{2nk'_2}; \\ \mathbf{w}_{\mathsf{x},2} = \mathsf{ext}_{\mathsf{mix}}(\mathbf{w}'_{\mathsf{X},2}) \in \{-1,0,1\}^{nk\delta_\beta(6c_d+3)}. \end{cases} \tag{36}$$

Let $L_{\mathsf{X}} = 3(2n\overline{m}\delta_\beta + n\ell) + 2nk'_2 + nk\delta_\beta(6c_d+3)$ and $\mathbf{w}_{\mathsf{X}} = (\mathbf{w}_{\mathsf{X},0}^\top | \mathbf{w}_{\mathsf{X},1}^\top | \mathbf{w}_{\mathsf{X},2}^\top)^\top \in \{-1,0,1\}^{L_{\mathsf{X}}}$, one can further transform equation (35) to $\mathbf{M}_{\mathsf{X}} \cdot \mathbf{w}_{\mathsf{X}} = \widetilde{\mathbf{v}}_{\mathsf{x}}$ by inserting suitable zero-columns into $\mathbf{M}'_{\mathsf{X}}$.

We now specify the sets $\mathsf{VALID}_{\mathsf{X}} \subset \{-1,0,1\}^{L_{\mathsf{X}}}$, $\mathcal{S}_{\mathsf{X}}$ and associated permutation $\{\Gamma_\phi : \phi \in \mathcal{S}_{\mathsf{X}}\}$ such that $\mathbf{w}_{\mathsf{X}} \in \mathsf{VALID}_{\mathsf{X}}$ and the conditions in (18) are satisfied.

Let $\mathsf{VALID}_{\mathsf{X}}$ contain vector $\mathbf{w}$ of the form $\mathbf{w} = (\mathbf{w}_0^\top \mid \mathbf{w}_1^\top \mid \mathbf{w}_2^\top)^\top$ satisfying the following:

- There exists $\mathbf{w}'_0 \in \{-1,0,1\}^{2n\overline{m}\delta_\beta + n\ell}$ such that $\mathbf{w}_0 = \mathsf{ext}_3(\mathbf{w}'_0)$.
- There exists $\mathbf{w}'_1 \in \{0,1\}^{nk'_2}$ such that $\mathbf{w}_1 = \mathsf{ext}_2(\mathbf{w}'_1)$.
- There exist $\mathbf{t} \in \{0,1\}^{c_d}$ and $\mathbf{v} \in \{-1,0,1\}^{nk\delta_\beta}$ such that $\mathbf{w}_2 = \mathsf{ext}_{\mathsf{mix}}(\mathbf{t}, \mathbf{v})$.

It is straightforward to verify that $\mathbf{w}_{\mathsf{X}} \in \mathsf{VALID}_{\mathsf{X}}$.

Let $\mathcal{S}_{\mathsf{X}} = \{-1,0,1\}^{2n\overline{m}\delta_\beta + n\ell} \times \{0,1\}^{nk'_2} \times \{0,1\}^{c_d} \times \{-1,0,1\}^{nk\delta_\beta}$. For $\phi = (\mathbf{f}_0, \mathbf{f}_1, \mathbf{f}_t, \mathbf{f}_v) \in \mathcal{S}_{\mathsf{X}}$, the permutation Γ_ϕ works in the following manner. It maps a vector $\mathbf{w} = (\mathbf{w}_0^\top | \mathbf{w}_1^\top | \mathbf{w}_2^\top)^\top \in \mathbb{Z}_q^{L_{\mathsf{X}}}$, with each block having appropriate sizes, to a vector of the following format

$$\Gamma_\phi(\mathbf{w}) = (F_{\mathsf{tri}}(\mathbf{f}_0, \mathbf{w}_0)^\top \mid F_{\mathsf{bin}}(\mathbf{f}_1, \mathbf{w}_1)^\top \mid F_{\mathsf{mix}}((\mathbf{f}_t, \mathbf{f}_v), \mathbf{w}_2)^\top)^\top. \tag{37}$$

Due to the equivalences observed in (24), (22), (31), one can see that $\mathbf{w} \in \mathsf{VALID}_{\mathsf{X}}$ if and only if $\Gamma_\phi(\mathbf{w}) \in \mathsf{VALID}_{\mathsf{X}}$. In addition, the randomness in ϕ ensures that $\Gamma_\phi(\mathbf{w})$ is randomly distributed in $\mathsf{VALID}_{\mathsf{X}}$ if $\mathbf{w}$ belongs to $\mathsf{VALID}_{\mathsf{X}}$. We thus successfully reduce the substatement involving (3) into an instance of $\mathcal{R}_{\mathsf{abstract}}$.

E.2 The Substatement Involving (4)

Again, we aim to transform this statement to an instance of $\mathcal{R}_{\mathsf{abstract}}$. We note that this is quite similar to (3) except that the signed message is $\mathsf{rdec}(h_P)$ where $\mathbf{A}_{\mathsf{hp}} \cdot \mathbf{z}_P = h_P$. In other words, we need to prove knowledge of $(\mathbf{z}_P, t_P, \mathbf{r}_P, \mathbf{v}_P)$ such that the following conditions hold.

$$\begin{cases} [\mathbf{A}_{\mathsf{P}}, \mathbf{A}_{\mathsf{P},[0]} + \sum_{j=1}^{d} \mathbf{A}_{\mathsf{P},[i]} t_{P,[i]}] \cdot \mathbf{v}_P = u_{\mathsf{P}} + \mathbf{F}_{\mathsf{P}} \cdot \mathbf{e}_P; \\ \mathbf{e}_P = \mathsf{rdec}(\mathbf{F}_{\mathsf{P},0} \cdot \mathbf{r}_P + \mathbf{F}_{\mathsf{P},1} \cdot \mathbf{h}_P) \in \{-1,0,1\}^{n\ell}; \\ \mathbf{h}_P = \mathsf{rdec}(h_P) \in \{-1,0,1\}^{n\ell}; \\ \mathbf{A}_{\mathsf{hp}} \cdot \mathbf{z}_P = h_P; \\ \mathbf{z}_P \in \{0,1\}^{2N\delta_{\mathsf{P}}}; t_P \in \{0,1\}^{c_d}; \mathbf{r}_P \in (R)^{\overline{m}}; \|\mathbf{r}_P\|_\infty \leq \beta; \\ \mathbf{v}_P \in (R)^{\overline{m}+k}; \|\mathbf{v}_P\|_\infty \leq \beta. \end{cases} \tag{38}$$

We follow the footprint of decomposition, extension and permutation as before. *The First Step-Decomposition.* Let $\mathbf{v}_P = (\mathbf{v}_{P,0}^\top, \mathbf{v}_{P,1}^\top)^\top$ with $\mathbf{v}_{P,0}^\top \in (R)^{\overline{m}}$ and $\mathbf{v}_{P,1}^\top \in (R)^k$ perform the following extensions.

- Let $\mathbf{v}'_{P,0} = \mathsf{rdec}_\beta(\mathbf{v}_{P,0}) \in \{-1,0,1\}^{n\overline{m}\delta_\beta}$, $\mathbf{v}'_{P,1} = \mathsf{rdec}_\beta(\mathbf{v}_{P,1}) \in \{-1,0,1\}^{nk\delta_\beta}$, and $\mathbf{r}'_P = \mathsf{rdec}_\beta(\mathbf{r}_P) \in \{-1,0,1\}^{n\overline{m}\delta_\beta}$.

Due to the equations observed in (13) and (16), equations (38) are equivalent to the following.

$$\begin{cases} [\mathsf{rot}(\mathbf{A}_{\mathsf{P},[0]}) \cdot \mathbf{H}_{k,\beta}] \cdot \mathbf{v}'_{P,1} + \sum_{i=1}^{d} \sum_{j=c_{i-1}}^{c_i-1} [\mathsf{rot}(\mathbf{A}_{\mathsf{P},[i]} \cdot X^j) \cdot \mathbf{H}_{k,\beta}] \cdot t_{P,j} \cdot \mathbf{v}'_{P,1} \\ \qquad + [\mathsf{rot}(\mathbf{A}_\mathsf{P}) \cdot \mathbf{H}_{\overline{m},\beta}] \cdot \mathbf{v}'_{P,0} - [\mathsf{rot}(\mathbf{F}_\mathsf{P})] \cdot \mathbf{e}_P = \tau(u_\mathsf{P}) \bmod q, \\ [\mathsf{rot}(\mathbf{F}_{\mathsf{P},0}) \cdot \mathbf{H}_{\overline{m},\beta}] \cdot \mathbf{r}'_P + [\mathsf{rot}(\mathbf{F}_{\mathsf{P},1})] \cdot \mathbf{h}_P - [\mathbf{H}] \cdot \mathbf{e}_P = \mathbf{0} \bmod q, \\ [\mathsf{rot}(\mathbf{A}_\mathsf{hp})] \cdot \mathbf{z}_P - [\mathbf{H}] \cdot \mathbf{h}_P = \mathbf{0} \bmod q. \end{cases} \tag{39}$$

Let $L'_\mathsf{cert} = 2n\overline{m}\delta_\beta + 2n\ell + 2N\delta_\mathsf{P} + nk\delta_\beta(1+c_d)$. Form secret vector $\mathbf{w}'_\mathsf{cert} \in \{-1,0,1\}^{L'_\mathsf{cert}}$ that is column concatenation of the following vectors.

$$\begin{cases} \mathbf{w}'_{\mathsf{cert},0} = (\ \mathbf{v}'^\top_{P,0} \mid\ \mathbf{r}'^\top_P \mid\ \mathbf{e}_P^\top \mid\ \mathbf{h}_P^\top\ ) \in \{-1,0,1\}^{2n\overline{m}\delta_\beta + 2n\ell}, \\ \mathbf{w}'_{\mathsf{cert},1} = \mathbf{z}_P \in \{0,1\}^{2N\delta_\mathsf{P}}, \\ \mathbf{w}'_{\mathsf{cert},2} = (\ \mathbf{v}'^\top_{P,1} \mid\ t_{P,0} \cdot \mathbf{v}'^\top_{P,1} \mid \cdots \mid\ t_{P,c_d-1} \cdot \mathbf{v}'^\top_{P,1}\ )^\top \in \{-1,0,1\}^{nk\delta_\beta(1+c_d)}. \end{cases}$$

Through some basic algebra, one can obtain from (39) an unifying equation of the form

$$\mathbf{M}'_\mathsf{cert} \cdot \mathbf{w}'_\mathsf{cert} = \mathbf{v}_\mathsf{cert} \text{ for some vector } \mathbf{v}_\mathsf{cert}. \tag{40}$$

The Second Step-Extension and Permutation. We note that the secret vector $\mathbf{w}'_\mathsf{cert}$ in (40) has the same structure as $\mathbf{w}'_\mathsf{x}$ in equation (35). We thus follow the same extension and permutation techniques. Specifically, we extend $\mathbf{w}'_\mathsf{cert}$ to $\mathbf{w}_\mathsf{cert}$ such that (40) can be transform to an equivalent form $\mathbf{M}_\mathsf{cert} \cdot \mathbf{w}_\mathsf{cert} = \mathbf{v}_\mathsf{cert}$. In addition, we can easily form $\mathsf{VALID}_\mathsf{cert} \subset \{-1,0,1\}^{L_\mathsf{cert}}$ that contains $\mathbf{w}_\mathsf{cert}$, $\mathcal{S}_\mathsf{cert}$ and associated permutation $\{\Gamma_\phi : \phi \in \mathcal{S}_\mathsf{cert}\}$ such that the conditions in (18) are satisfied. Here $L_\mathsf{cert} = 3(2n\overline{m}\delta_\beta + 2n\ell) + 2 \cdot 2N\delta_\mathsf{P} + nk\delta_\beta(3+6c_d)$. Due to high similarity, we omit the details.

E.3 The Substatement Involving (5)

This statement is to prove knowledge of $(\mathbf{s}_{\mu_1}, \ldots, \mathbf{s}_{\rho-1}, \mathbf{h}_P, \mathbf{r}_{\mathsf{com},\mu_1}, \ldots, \mathbf{r}_{\mathsf{com},\rho-1}, \mathbf{r}_{\mathsf{com},P})$ such that

$$\begin{cases} \mathsf{com}_i = \mathbf{A}_0 \cdot \mathbf{s}_i + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},i}, \quad \mathbf{s}_i \in \{0,1\}^\rho, \mathbf{r}_{\mathsf{com},i} \in \{0,1\}^{nm}, \quad \forall i \in [\mu_1, \rho-1], \\ \mathsf{com}_P = \mathbf{A}_c \cdot \mathbf{h}_P + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},P}, \quad \mathbf{h}_P \in \{-1,0,1\}^{n\ell}, \quad \mathbf{r}_{\mathsf{com},P} \in \{0,1\}^{nm}. \end{cases} \tag{41}$$

First, note that $\mathsf{com}_P = \mathbf{A}_c \cdot \mathbf{h}_P + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},P}$ can be rewritten as

$$\tau(\mathsf{com}_P) = [\mathsf{rot}(\mathbf{A}_c)] \cdot \mathbf{h}_P + [\mathsf{rot}(\mathbf{A}_1)] \cdot \mathbf{r}_{\mathsf{com},P}.$$

Let $\mathbf{w}'_{\mathsf{com},0} \in \{0,1\}^{(\rho-\mu_1)\rho+(\rho-\mu_1)nm+nm}$ be a secret vector of the following form

$$\mathbf{w}'_{\mathsf{com},0} = \left(\mathbf{s}_{\mu_1}^\top \mid \cdots \mid \mathbf{s}_{\rho-1}^\top \mid \mathbf{r}_{\mathsf{com},\mu_1}^\top \mid \cdots \mid \mathbf{r}_{\mathsf{com},\rho-1}^\top \mid \mathbf{r}_{\mathsf{com},P}^\top \right)^\top$$

and let $\mathbf{w}'_{\mathsf{com}}$ be the column concatenation of $\mathbf{w}'_{\mathsf{com},0}$ and $\mathbf{h}_P \in \{-1,0,1\}^{n\ell}$. Through some basic algebra, (41) can be rewritten as $\mathbf{M}'_{\mathsf{com}} \cdot \mathbf{w}'_{\mathsf{com}} = \mathbf{v}_{\mathsf{com}}$ for some public inputs $\mathbf{M}'_{\mathsf{com}}, \mathbf{v}_{\mathsf{com}}$.

Extension and Permutation. Observe that the secret vectors are either all binary or ternary, and the techniques for proving a binary vector and a ternary vector in Section D can be directly applied here. To this end, let $\mathbf{w}_{\mathsf{com},0} = \mathsf{ext}_2(\mathbf{w}'_{\mathsf{com},0}) \in \{0,1\}^{L_{\mathsf{com},0}}$ with $L_{\mathsf{com},0} = 2(\rho-\mu_1)(\rho+nm)+2nm$, $\mathbf{w}_{\mathsf{com},1} = \mathsf{ext}_3(\mathbf{h}_P) \in \{-1,0,1\}^{3n\ell}$, and $\mathbf{w}_{\mathsf{com}} = (\mathbf{w}_{\mathsf{com},0}^\top \mid \mathbf{w}_{\mathsf{com},1}^\top)^\top \in \{-1,0,1\}^{L_{\mathsf{com}}}$ with $L_{\mathsf{com}} = L_{\mathsf{com},0} + 3n\ell$.

By inserting zero-columns in $\mathbf{M}'_{\mathsf{com}}$, we can obtain $\mathbf{M}_{\mathsf{com}}$ such that $\mathbf{M}'_{\mathsf{com}} \cdot \mathbf{w}'_{\mathsf{com}} = \mathbf{v}_{\mathsf{com}}$ is further equivalent to $\mathbf{M}_{\mathsf{com}} \cdot \mathbf{w}_{\mathsf{com}} = \mathbf{v}_{\mathsf{com}}$. Let $\mathsf{VALID}_{\mathsf{com}} \subset \{0,1\}^{L_{\mathsf{com}}}$ contain vectors of form $\mathbf{w} = (\mathbf{w}_0^\top \mid \mathbf{w}_1^\top)^\top \in \{0,1\}^{L_{\mathsf{com}}}$ such that

- There exits $\mathbf{w}'_0 \in \{0,1\}^{L_{\mathsf{com},0}/2}$ such that $\mathbf{w}_0 = \mathsf{ext}_2(\mathbf{w}'_0) \in \{0,1\}^{L_{\mathsf{com},0}}$.
- There $\mathbf{w}'_1 \in \{-1,0,1\}^{n\ell}$ such that $\mathbf{w}_1 = \mathsf{ext}_3(\mathbf{w}'_1) \in \{-1,0,1\}^{3n\ell}$.

Our secret vector $\mathbf{w}_{\mathsf{com}}$ belongs to $\mathsf{VALID}_{\mathsf{com}}$. Let $\mathcal{S}_{\mathsf{com}} = \{0,1\}^{L_{\mathsf{com},0}/2} \times \{-1,0,1\}^{n\ell}$. For $\phi = (\mathbf{f}_0, \mathbf{f}_1) \in \mathcal{S}_{\mathsf{com}}$, the permutation Γ_ϕ transforms a vector $\mathbf{w} = (\mathbf{w}_0^\top \mid \mathbf{w}_1^\top)^\top \in \mathbb{Z}_q^{L_{\mathsf{com}}}$ to $\Gamma_\phi(\mathbf{w}) = \left(F_{\mathsf{bin}}(\mathbf{f}_0, \mathbf{w}_0)^\top \mid F_{\mathsf{tri}}(\mathbf{f}_1, \mathbf{w}_1)^\top \right)^\top$.

Due to the equivalences observed in (22), and (24), one can see that we have reduce the substatement involving (5) into an instance of $\mathcal{R}_{\mathsf{abstract}}$.

E.4 The Substatement Involving (10)

This statement proves that we have performed the "bucket" search step properly. The goal is also to transform (10) into an instance of $\mathcal{R}_{\mathsf{abstract}}$ so that conditions in (18) are fulfilled. To this end, we first transform the secret vectors involved in (10) into vectors over $\{-1,0,1\}$. Next, we perform some extension techniques to the secret vectors, obtaining a valid set $\mathsf{VALID}_{g,i}$ so that conditions in (18) can be satisfied if proper permutations $\{\Gamma_\phi : \phi \in \mathcal{S}_{g,i}\}$ are employed for some fine-grained set $\mathcal{S}_{g,i}$. Note that the search of $s_{h(i)}$ is similar and omitted here to avoid redundancy.

The First Step-Decomposition. Due to the equivalence observed in (13), we transform $\widetilde{\mathsf{com}}_{g,i} = \mathbf{A}_0 \cdot (\widetilde{s}_{i,0}, \ldots, \widetilde{s}_{i,\rho-1})^\top + \mathbf{A}_1 \cdot \mathbf{r}_{g,i}$ into the following

$$\tau(\widetilde{\mathsf{com}}_{g,i}) = \mathsf{rot}(\mathbf{A}_0) \cdot (\widetilde{s}_{i,0}, \ldots, \widetilde{s}_{i,\rho-1})^\top + \mathsf{rot}(\mathbf{A}_1) \cdot \mathbf{r}_{g,i},$$

and write $\widetilde{\mathsf{com}}_{g,i} = [\mathsf{com}_0, \ldots, \mathsf{com}_{\rho-1}] \cdot (a_{i,0}, \ldots, a_{i,\rho-1})^\top$ as

$$\tau(\widetilde{\mathsf{com}}_{g,i}) = [\tau(\mathsf{com}_0), \ldots, \tau(\mathsf{com}_{\rho-1})] \cdot (a_{i,0}, \ldots, a_{i,\rho-1})^\top.$$

In addition, we have $\tau(\widetilde{\mathsf{com}}_{g,i}) = \mathbf{H} \cdot \mathsf{rdec}(\widetilde{\mathsf{com}}_{g,i})$ due to (16), where $\mathbf{H} \in \mathbb{Z}_q^{n \times n\delta}$ and $\mathsf{rdec}(\widetilde{\mathsf{com}}_{g,i}) = (h_{i,0}, \ldots, h_{i,n\delta-1})^\top \in \{-1,0,1\}^{n\delta}$. We now have all the secret vectors over $\{-1,0,1\}$.

Let $L' = 1 + n\delta + nm + 5\rho$. Form a secret vector $\mathbf{w}'_{g,i} \in \{-1,0,1\}^{L'}$ that is the column concatenation of the following five vectors

$$\begin{cases} \mathbf{w}'_{g,i,0} = \left(\ h_{i,0}, \ldots, h_{i,n\delta-1}\ \right)^\top \in \{-1,0,1\}^{n\delta}; \\ \mathbf{w}'_{g,i,1} = y_i \in \{0,1\}; \\ \mathbf{w}'_{g,i,2} = \left(\ \mathbf{r}_{g,i}^\top, \widetilde{s}_{i,0}, \ldots, \widetilde{s}_{i,\rho-1}, b_{i,0}, \ldots, b_{i,\rho-1}\ \right)^\top \in \{0,1\}^{2\rho+nm}; \\ \mathbf{w}'_{g,i,3} = \left(\ \widetilde{s}_{i,0} \cdot b_{i,0}, \ldots, \widetilde{s}_{i,\rho-1} \cdot b_{i,\rho-1}\ \right)^\top \in \{0,1\}^{\rho}; \\ \mathbf{w}'_{g,i,4} = \left(\ a_{i,0}, \ldots, a_{i,\rho-1}, b'_{i,0}, \ldots, b'_{i,\rho-1}\ \right)^\top \in \{0,1\}^{2\rho}; \end{cases}$$

with $(a_{i,0}, a_{i,1}, \ldots, a_{i,\rho-1})^\top = \Delta_{\frac{\delta_\mathsf{P}}{2}}(g_{i,0}, g_{i,1}, \ldots, g_{i,\frac{\delta_\mathsf{P}}{2}-1})$ and $(b'_{i,0}, \ldots, b'_{i,\rho-1})^\top = \Delta_{\frac{\delta_\mathsf{P}}{2}}(g_{i,\frac{\delta_\mathsf{P}}{2}}, \ldots, g_{i,\delta_\mathsf{P}-1})$. Through some basic algebra, we can also form public $\mathbf{M}'_{g,i} \in \mathbb{Z}_q^{(2n+\rho+1)\times L'}, \mathbf{v}_{g,i} = \mathbf{0}^{2n+\rho+1}$ such that equations (10) can be transformed into a unified equation $\mathbf{M}'_{g,i} \cdot \mathbf{w}'_{g,i} = \mathbf{v}_{g,i}$.

The Second Step-Extension. The purpose of extension is to make sure that some structure of the extended vector is invariant under a (random) permutation and that revealing the permuted vector does not reveal the original one. To this end, we perform extension techniques described in Section D on $\mathbf{w}'_{g,i,0}, \mathbf{w}'_{g,i,1}, \mathbf{w}'_{g,i,2}, \mathbf{w}'_{g,i,3}$.

1. Extend $\mathbf{w}'_{g,i,0}$ to $\mathbf{w}_{g,i,0} = \mathsf{ext}_3(\mathbf{w}'_{g,i,0}) \in \{-1,0,1\}^{3n\delta}$. Then according to (23), $\mathbf{w}'_{g,i,0} = \mathbf{I}^\star_{n\delta} \cdot \mathbf{w}_{g,i,0}$.
2. Extend $\mathbf{w}'_{g,i,1}$ to $\mathbf{w}_{g,i,1} = \mathsf{ext}_2(\mathbf{w}'_{g,i,1}) \in \{0,1\}^2$ and extend $\mathbf{w}'_{g,i,2}$ to $\mathbf{w}_{g,i,2} = \mathsf{ext}_2(\mathbf{w}'_{g,i,2}) \in \{0,1\}^{2(2\rho+nm)}$. Then according to (21), $\mathbf{w}'_{g,i,2} = \mathbf{J}^\star_{2\rho+nm} \cdot \mathbf{w}_{g,i,2}$.
3. Extend $\mathbf{w}'_{g,i,3}$ to $\mathbf{w}_{g,i,3} = \mathsf{ext}_{\mathsf{mult}}(\mathbf{w}'_{g,i,3}) \in \{0,1\}^{4\rho}$. Then according to (25), $\mathbf{w}'_{g,i,3} = \mathbf{K}^\star_\rho \cdot \mathbf{w}_{g,i,3}$.
4. For the consistency of notations, let $\mathbf{w}_{g,i,4} = \mathbf{w}'_{g,i,4}$.

Let $L_{g,i} = 3n\delta + 2(1 + 2\rho + nm) + 4\rho + 2\rho$ and $\mathbf{w}_{g,i} = (\mathbf{w}_{g,i,0}^\top \mid \cdots \mid \mathbf{w}_{g,i,4}^\top)^\top \in \{-1,0,1\}^{L_{g,i}}$. By inserting suitable zero-columns into the matrix $\mathbf{M}'_{g,i}$, one can obtain $\mathbf{M}_{g,i} \in \mathbb{Z}_q^{(2n+\rho+1)\times L_{g,i}}$ such that $\mathbf{M}'_{g,i} \cdot \mathbf{w}'_{g,i} = \mathbf{v}_{g,i}$ is now $\mathbf{M}_{g,i} \cdot \mathbf{w}_{g,i} = \mathbf{v}_{g,i}$.

The Third Step-Permutation. We are now at the stage of identifying the sets $\mathsf{VALID}_{g,i} \subset \{-1,0,1\}^{L_{g,i}}$, $\mathcal{S}_{g,i}$ and a corresponding permutation $\{\Gamma_\phi : \phi \in \mathcal{S}_{g,i}\}$ so that conditions in (18) are conformed.

Let $\mathsf{VALID}_{g,i}$ contain vectors $\mathbf{w}$ of the form $\mathbf{w} = (\mathbf{w}_0^\top \mid \mathbf{w}_1^\top \mid \mathbf{w}_2^\top \mid \mathbf{w}_3 \mid \mathbf{w}_{4,0}^\top \mid \mathbf{w}_{4,1}^\top)^\top$ such that the following conditions hold. Those conditions are imposed to make sure that the secret vector $\mathbf{w}_{g,i}$ belongs to $\mathsf{VALID}_{g,i}$.

- $\mathbf{w}_0 \in \{-,1,0,1\}^{3n\delta}$ and there exists a vector $\mathbf{w}'_0 \in \{-1,0,1\}^{n\delta}$ such that $\mathbf{w}_0 = \mathsf{ext}_3(\mathbf{w}'_0)$.
- $\mathbf{w}_1 \in \{0,1\}^2$ and there exists a bit w'_1 such that $\mathbf{w}_1 = \mathsf{ext}_2(w'_1)$.
- $\mathbf{w}_2 \in \{0,1\}^{2(2\rho+nm)}$ and there exists a vector $\mathbf{w}'_2 \in \{0,1\}^{2\rho+nm}$ such that $\mathbf{w}_2 = \mathsf{ext}_2(\mathbf{w}'_2)$. Let the last 2ρ bits of $\mathbf{w}'_2$ be $\widetilde{\mathbf{s}} = (\widetilde{s}_0, \ldots, \widetilde{s}_{\rho-1})^\top$, $\mathbf{b} = (b_0, \ldots, b_{\rho-1})^\top$.

– $\mathbf{w}_3 \in \{0,1\}^{4\rho}$ and $\mathbf{w}_3 = \mathsf{ext}_{\mathsf{mult}}(\widetilde{\mathbf{s}}, \mathbf{b})$. This ensures the interweaving connection between $\mathbf{w}_2$ and $\mathbf{w}_3$.
– $\mathbf{w}_{4,0}, \mathbf{w}_{4,1} \in \{0,1\}^{\rho}$ and there exist vectors $\mathbf{w}'_{4,0} = (g_0, \ldots, g_{\frac{\delta_{\mathsf{P}}}{2}-1})^\top$, $\mathbf{w}'_{4,1} = (g_{\frac{\delta_{\mathsf{P}}}{2}}, \ldots, g_{\delta_{\mathsf{P}}-1})^\top$ such that $\mathbf{w}_{4,0} = \Delta_{\delta_{\mathsf{P}}/2}(\mathbf{w}'_{4,0})$ and $\mathbf{w}_{4,1} = \Delta_{\delta_{\mathsf{P}}/2}(\mathbf{w}'_{4,1})$.

Let the set $\mathcal{S}_{g,i}$ be $\{-1,0,1\}^{n\delta} \times \{0,1\} \times \{0,1\}^{nm+2\rho} \times (\{0,1\}^{\delta_{\mathsf{P}}/2})^2$. For $\phi = (\mathbf{f}_0, f_1, \mathbf{f}_2, \mathbf{f}_{4,0}, \mathbf{f}_{4,1}) \in \mathcal{S}_{g,i}$, the permutation Γ_ϕ functions as follows. When applying to a vector $\mathbf{w} = (\mathbf{w}_0^\top \mid \mathbf{w}_1^\top \mid \mathbf{w}_2^\top \mid \mathbf{w}_3^\top \mid \mathbf{w}_{4,0}^\top \mid \mathbf{w}_{4,1}^\top)^\top \in \mathbb{Z}^{L_{g,i}}$, it permutes $\mathbf{w}$ to $\Gamma_\phi(\mathbf{w})$ of the following form:

$$\big(F_{\mathsf{tri}}(\mathbf{f}_0, \mathbf{w}_0)^\top \mid F_{\mathsf{bin}}(f_1, \mathbf{w}_1)^\top \mid F_{\mathsf{bin}}(\mathbf{f}_2, \mathbf{w}_2)^\top \mid F_{\mathsf{mult}}((\mathbf{f}_s, \mathbf{f}_b), \mathbf{w}_3)^\top \mid F_{\mathsf{reg}}(\mathbf{f}_{4,0}, \mathbf{w}_{4,0})^\top \mid F_{\mathsf{reg}}(\mathbf{f}_{4,1}, \mathbf{w}_{4,1})^\top\big)^\top,$$

where $\mathbf{f}_s = (f_{s,0}, \ldots, f_{s,\rho-1})^\top$, $\mathbf{f}_b = (f_{b,0}, \ldots, f_{b,\rho-1})^\top$ are the last 2ρ bits of $\mathbf{f}_2$.

Based on the equivalences observed in (24), (22), (26), (32), it is verifiable that $\mathbf{w} \in \mathsf{VALID}_{g,i}$ if and only if $\Gamma_\phi(\mathbf{w}) \in \mathsf{VALID}_{g,i}$. In addition, if ϕ is sampled uniformly at random from $\mathcal{S}_{g,i}$ and $\mathbf{w} \in \mathsf{VALID}_{g,i}$, then $\Gamma_\phi(\mathbf{w})$ is evenly distributed in $\mathsf{VALID}_{g,i}$. Therefore, we have finally reached the point where the considered statement in (10) has been reduced to an instance of $\mathcal{R}_{\mathsf{abstract}}$.

E.5 The Substatement Involving (11)

Now we need to transform (11) to an instance of $\mathcal{R}_{\mathsf{abstract}}$ so that the properties in (18) are complied. Let $\mathbf{w}'_{\mathsf{P},0} = (s_K, \ldots, s_{K+N-2})^\top \in \{0,1\}^{N-1}$ and $\mathbf{w}'_{\mathsf{P},1} = (y_0 \cdot z_0, \ldots, y_{N-1} \cdot z_{N-1})^\top \in \{0,1\}^N$. Since the secret vectors are binary, it suffices to perform extension and permutation techniques.

The First Step-Extension. Observe that $s_i \in \{0,1\}$ is a bit and $y_i \cdot z_i$ is multiplication of two bits, we perform the following extension.

1. Extend $\mathbf{w}'_{\mathsf{P},0}$ to $\mathbf{w}_{\mathsf{P},0} = \mathsf{ext}_2(\mathbf{w}'_{\mathsf{P},0}) \in \{0,1\}^{2(N-1)}$. Then according to (21), $\mathbf{w}'_{\mathsf{P},0} = \mathbf{J}^\star_{N-1} \cdot \mathbf{w}_{\mathsf{P},0}$.
2. Extend $\mathbf{w}'_{\mathsf{P},1}$ to $\mathbf{w}_{\mathsf{P},1} = \mathsf{ext}_{\mathsf{mult}}(\mathbf{w}'_{\mathsf{P},1}) \in \{0,1\}^{4N}$. Then according to (25), $\mathbf{w}'_{\mathsf{P},1} = \mathbf{K}^\star_N \cdot \mathbf{w}_{\mathsf{P},1}$.

Let $L_{\mathsf{P}} = 2(N-1) + 4N$ and $\mathbf{w}_{\mathsf{P}} = (\mathbf{w}_{\mathsf{P},0}^\top \mid \mathbf{w}_{\mathsf{P},1}^\top)^\top \in \{0,1\}^{L_{\mathsf{P}}}$. Through some basic algebra, one can form public matrix $\mathbf{M}_{\mathsf{P}} \in \mathbb{Z}_q^{N \times L_{\mathsf{P}}}$ and vector $\mathbf{v}_{\mathsf{P}} \in \mathbb{Z}_q^N$ such that equations in (11) are simply $\mathbf{M}_{\mathsf{P}} \cdot \mathbf{w}_{\mathsf{P}} = \mathbf{v}_{\mathsf{P}}$.

The Second Step-Permutation. Let us now specify the sets $\mathsf{VALID}_{\mathsf{P}} \subset \{0,1\}^{L_{\mathsf{P}}}$, $\mathcal{S}_{\mathsf{P}}$ and an associated permutation $\{\Gamma_\phi : \phi \in \mathcal{S}_{\mathsf{P}}\}$ such that the restrictions in (18) are obeyed.

Let $\mathsf{VALID}_{\mathsf{P}}$ contain all vectors $\mathbf{w} = (\mathbf{w}_0^\top \mid \mathbf{w}_1^\top)^\top$ such that the following two conditions are applied to $\mathbf{w}_0, \mathbf{w}_1$, respectively.

– $\mathbf{w}_0 \in \{0,1\}^{2(N-1)}$ and there exists a vector $\mathbf{w}'_0 \in \{0,1\}^{N-1}$ satisfying $\mathbf{w}_0 = \mathsf{ext}_2(\mathbf{w}'_0)$.

– $\mathbf{w}_1 \in \{0,1\}^{4N}$ and there exist $\mathbf{y} \in \{0,1\}^N$ and $\mathbf{z} \in \{0,1\}^N$ satisfying $\mathbf{w}_1 = \mathsf{ext}_{\mathsf{mult}}(\mathbf{y}, \mathbf{z})$.

Let the set $\mathcal{S}_{\mathsf{P}} = \{0,1\}^{N-1} \times (\{0,1\}^N)^2$. For $\phi = (\mathbf{f}_0, \mathbf{f}_y, \mathbf{f}_z) \in \mathcal{S}_{\mathsf{P}}$, the permutation Γ_ϕ maps a vector $\mathbf{w} = (\mathbf{w}_0^\top \mid \mathbf{w}_1^\top)^\top \in \mathbb{Z}^{L_{\mathsf{P}}}$ into

$$\Gamma_\phi(\mathbf{w}) = (F_{\mathsf{bin}}(\mathbf{f}_0, \mathbf{w}_0)^\top \mid F_{\mathsf{mult}}((\mathbf{f}_y, \mathbf{f}_z), \mathbf{w}_1)^\top)^\top.$$

Based on the equivalences observed in (22), (26), it is verifiable that $\mathbf{w} \in \mathsf{VALID}_{\mathsf{P}}$ if and only if $\Gamma_\phi(\mathbf{w}) \in \mathsf{VALID}_{\mathsf{P}}$. In addition, if ϕ is sampled uniformly at random from $\mathcal{S}_{\mathsf{p}}$ and $\mathbf{w} \in \mathsf{VALID}_{\mathsf{P}}$, then $\Gamma_\phi(\mathbf{w})$ is evenly distributed in $\mathsf{VALID}_{\mathsf{P}}$. Therefore, we have reduced the statement involving (11) to an instance of $\mathcal{R}_{\mathsf{abstract}}$.

E.6 Deferred Description of $\mathsf{VALID}_{\mathsf{baps}}$, $\mathcal{S}_{\mathsf{baps}}$, Γ_ϕ

We remark that $\mathsf{VALID}_{\mathsf{baps}}, \mathcal{S}_{\mathsf{baps}}$ are not as direct as they may look like. We have to make sure that some vector, say $\mathbf{x}$ that appears in $\mathbf{w}_{\mathsf{X}}$, $\mathbf{w}_{\mathsf{com}}$, has to be the same in all appearances. This requires fine-grained design of $\mathsf{VALID}_{\mathsf{baps}}, \mathcal{S}_{\mathsf{baps}}$. Let us recall components of $\mathbf{w}_{\mathsf{type}}$ for $\mathsf{type} \in \{\mathsf{X}, \mathsf{cert}, \mathsf{com}, (g,0), \ldots, (g, N-1), (h,0), \ldots, (h, N-1), \mathsf{P}\}$. Note that $\rho\mu_1 = k_1$ and $\rho^2 = K + N - 1$.

$$\mathbf{w}_{\mathsf{X}}: \begin{cases} \mathbf{w}_{\mathsf{X},0} = \mathsf{ext}_3(\mathbf{w}'_{\mathsf{X},0}) \in \{-1,0,1\}^{3(2n\overline{m}\delta_\beta + n\ell)}; \\ \mathbf{w}_{\mathsf{X},1} = \mathsf{ext}_2(\mathbf{x}) \in \{0,1\}^{2nk'_2}; \ \mathbf{x} = (s_{\rho\mu_1}, \ldots, s_{\rho\mu_1+k_2-1})^\top; \\ \mathbf{w}_{\mathsf{X},2} = \mathsf{ext}_{\mathsf{mix}}(\mathbf{w}'_{\mathsf{X},2}) \in \{-1,0,1\}^{nk\delta_\beta(6c_d+3)}; \end{cases}$$

$$\mathbf{w}_{\mathsf{cert}}: \begin{cases} \mathbf{w}_{\mathsf{cert},0} = \mathsf{ext}_3(\mathbf{w}'_{\mathsf{cert},0}) \in \{-1,0,1\}^{3(2n\overline{m}\delta_\beta + n\ell)}; \\ \mathbf{w}_{\mathsf{cert},1} = \mathsf{ext}_3(\mathbf{h}_P) \in \{-1,0,1\}^{3n\ell}; \\ \mathbf{w}_{\mathsf{cert},2} = \mathsf{ext}_2(\mathbf{z}_P) \in \{0,1\}^{2(2N\delta_{\mathsf{P}})}; \\ \mathbf{w}_{\mathsf{cert},3} = \mathsf{ext}_{\mathsf{mix}}(\mathbf{w}'_{\mathsf{cert},3}) \in \{-1,0,1\}^{nk\delta_\beta(3+6c_d)}; \end{cases}$$

$$\mathbf{w}_{\mathsf{com}}: \begin{cases} \mathbf{w}_{\mathsf{com},0} = \mathsf{ext}_2((s_{\rho\mu_1}, \ldots, s_{\rho^2-1})^\top) \in \{0,1\}^{2(\rho-\mu_1)\rho}; \\ \mathbf{w}_{\mathsf{com},1} = \mathsf{ext}_2(\mathbf{w}'_{\mathsf{com},1}) \in \{0,1\}^{2(\rho-\mu_1)\cdot nm + 2nm}; \\ \mathbf{w}_{\mathsf{com},2} = \mathsf{ext}_3(\mathbf{h}_P) \in \{-1,0,1\}^{3n\ell}; \end{cases} \tag{42}$$

$$\forall\ i \in [0, N-1]\ \mathbf{w}_{g,i}: \begin{cases} \mathbf{w}_{g,i,0} = \mathsf{ext}_3(\mathbf{w}'_{g,i,0}) \in \{-1,0,1\}^{3n\delta}; \\ \mathbf{w}_{g,i,1} = \mathsf{ext}_2(y_i) \in \{0,1\}^2; \\ \mathbf{w}_{g,i,2} = \mathsf{ext}_2(\mathbf{r}_{g,i}) \in \{0,1\}^{2nm}; \\ \mathbf{w}_{g,i,3} = \mathsf{ext}_{\mathsf{merge}}((\widetilde{s}_{i,0}, \ldots, \widetilde{s}_{i,\rho-1})^\top, (b_{i,0}, \ldots, b_{i,\rho-1})^\top) \in \{0,1\}^{8\rho}; \\ \mathbf{w}_{g,i,4} = \left(a_{i,0}, \ldots, a_{i,\rho-1}, b'_{i,0}, \ldots, b'_{i,\rho-1} \right)^\top \in \{0,1\}^{2\rho}; \\ (a_{i,0}, a_{i,1}, \ldots, a_{i,\rho-1})^\top = \Delta_{\frac{\delta_{\mathsf{P}}}{2}}(g_{i,0}, g_{i,1}, \ldots, g_{i,\frac{\delta_{\mathsf{P}}}{2}-1}); \\ (b'_{i,0}, \ldots, b'_{i,\rho-1})^\top = \Delta_{\frac{\delta_{\mathsf{P}}}{2}}(g_{i,\frac{\delta_{\mathsf{P}}}{2}}, \ldots, g_{i,\delta_{\mathsf{P}}-1}); \end{cases}$$

$$\forall\ i \in [0, N-1]\ \mathbf{w}_{h,i}: \begin{cases} \mathbf{w}_{h,i,0} = \mathsf{ext}_3(\mathbf{w}'_{h,i,0}) \in \{-1,0,1\}^{3n\delta}; \\ \mathbf{w}_{h,i,1} = \mathsf{ext}_2(z_i) \in \{0,1\}^2; \\ \mathbf{w}_{h,i,2} = \mathsf{ext}_2(\mathbf{r}_{h,i}) \in \{01\}^{2nm}; \\ \mathbf{w}_{h,i,3} = \mathsf{ext}_{\mathsf{merge}}((\widetilde{t}_{i,0}, \ldots, \widetilde{t}_{i,\rho-1})^\top, (d_{i,0}, \ldots, d_{i,\rho-1})^\top) \in \{0,1\}^{8\rho}; \\ \mathbf{w}_{h,i,4} = (\ c_{i,0}, \ldots, c_{i,\rho-1}, d'_{i,0}, \ldots, d'_{i,\rho-1}\ )^\top \in \{0,1\}^{2\rho}; \\ (c_{i,0}, c_{i,1}, \ldots, c_{i,\rho-1})^\top = \Delta_{\frac{\delta_{\mathsf{P}}}{2}}(h_{i,0}, h_{i,1}, \ldots, h_{i,\frac{\delta_{\mathsf{P}}}{2}-1}); \\ (d'_{i,0}, \ldots, d'_{i,\rho-1})^\top = \Delta_{\frac{\delta_{\mathsf{P}}}{2}}(h_{i,\frac{\delta_{\mathsf{P}}}{2}}, \ldots, h_{i,\delta_{\mathsf{P}}-1}); \end{cases}$$

$$\mathbf{w}_{\mathsf{P}}: \begin{cases} \mathbf{w}_{\mathsf{P},0} = \mathsf{ext}_2((s_K, \ldots, s_{K+N-2})^\top) \in \{0,1\}^{2(N-1)}; \\ \mathbf{w}_{\mathsf{P},1} = \mathsf{ext}_{\mathsf{mult}}(\mathbf{y}, \mathbf{z}) \in \{0,1\}^{4N}; \\ \mathbf{y} = (y_0, \ldots, y_{N-1})^\top \in \{0,1\}^N; \\ \mathbf{z} = (z_0, \ldots, z_{N-1})^\top \in \{0,1\}^N. \end{cases}$$

A vector (or bit) are marked in the same color if it appears in multiple locations. Specifically, $\mathbf{z}_P$ (partially) appears in $\mathbf{w}_{\mathsf{cert}}, \mathbf{w}_{g,0}, \ldots, \mathbf{w}_{g,N-1}, \mathbf{w}_{h,0}, \ldots, \mathbf{w}_{h,N-1}$. Vector $\mathbf{h}_P$ appears in both $\mathbf{w}_{\mathsf{cert}}$ and $\mathbf{w}_{\mathsf{com}}$. Vector $\mathbf{s} = (s_0, \ldots, s_{K+N-2})^\top$ (partially) appears in $\mathbf{w}_{\mathsf{X}}$, $\mathbf{w}_{\mathsf{com}}$, $\mathbf{w}_{\mathsf{P}}$. Vector $\mathbf{y}$ (partially) appears in $\mathbf{w}_{g,0}, \ldots, \mathbf{w}_{g,N-1}$, $\mathbf{w}_{\mathsf{P}}$ while $\mathbf{z}$ (partially) appears in $\mathbf{w}_{h,0}, \ldots, \mathbf{w}_{h,N-1}, \mathbf{w}_{\mathsf{P}}$.

We now briefly specify $\mathsf{VALID}_{\mathsf{baps}}, \mathcal{S}_{\mathsf{baps}}, \{\Gamma_\phi : \phi \in \mathcal{S}_{\mathsf{baps}}\}$. Let $\mathsf{VALID}_{\mathsf{baps}} \subset \{-1,0,1\}^{L_1+L_{\mathsf{P}}}$ be a set that contains concatenation of vectors

$$\mathbf{w}_{\mathsf{X}}, \mathbf{w}_{\mathsf{cert}}, \mathbf{w}_{\mathsf{com}}, \mathbf{w}_{g,0}, \ldots, \mathbf{w}_{g,N-1}, \mathbf{w}_{h,0}, \ldots, \mathbf{w}_{h,N-1}$$

such that equations (42) are satisfied. Regarding $\mathcal{S}_{\mathsf{baps}}$, we can think of it as

$$\mathcal{S}_{\mathsf{X}} \times \mathcal{S}_{\mathsf{cert}} \times \mathcal{S}_{\mathsf{com}} \times (\mathcal{S}_{g,0})^{2N} \times \mathcal{S}_{\mathsf{P}}$$

yet we force some vectors in $\mathcal{S}_{\mathsf{baps}}$ to be the same. This enforcement applies to all possible positions if the vectors in those positions are used to permute the same vectors/bits in $\mathbf{w}_{\mathsf{baps}}$. For instance, for $\phi = (\phi_{\mathsf{X}}, \phi_{\mathsf{cert}}, \phi_{\mathsf{com}}, \ldots, \phi_{\mathsf{P}}) \in \mathcal{S}_{\mathsf{baps}}$, let $\phi_{\mathsf{X}} = (\mathbf{f}_{\mathsf{x},0}, \mathbf{f}_{\mathsf{x},1}, \mathbf{f}_{\mathsf{x},t}, \mathbf{f}_{\mathsf{x},v})$ with $\mathbf{f}_{\mathsf{x},1} = (f_{\mathsf{x},\rho\mu_1}, \ldots, f_{\mathsf{x},\rho\mu_1+k_2-1})^\top \in \{0,1\}^{k_2}$ and $\phi_{\mathsf{com}} = (\mathbf{f}_{\mathsf{com},s}^\top, \mathbf{f}_{\mathsf{com},r}^\top, \mathbf{f}_{\mathsf{com},h}^\top)^\top \in \{0,1\}^{\rho^2-\rho\mu_1} \times \{0,1\}^{(\rho-\mu_1)nm+nm} \times \{-1,0,1\}^{n\ell}$ with $\mathbf{f}_{\mathsf{com},s} = (f_{\mathsf{com},\rho\mu_1}, \ldots, f_{\mathsf{com},\rho^2-1})^\top$, we should have, for $j \in [\rho\mu_1, \rho\mu_1+k_2-1]$, $f_{\mathsf{x},j} = f_{\mathsf{com},j}$ since they are used to permute the same bit s_j. Finally, Γ_ϕ for $\phi \in \mathcal{S}_{\mathsf{baps}}$ can be defined by applying $\Gamma_{\phi_{\mathsf{X}}}, \Gamma_{\phi_{\mathsf{cert}}}, \ldots, \Gamma_{\phi_{\mathsf{P}}}$ respectively. It is also verifiable that $\mathsf{VALID}_{\mathsf{baps}}, \mathcal{S}_{\mathsf{baps}}, \{\Gamma_\phi : \phi \in \mathcal{S}_{\mathsf{baps}}\}$ satisfy the conditions (18).

F The Zero-Knowledge Argument System Underlying the Disclose Algorithm

We first describe the relation $\mathcal{R}_{\mathsf{D}}$ incurred in the Disclose algorithm in Section 4.2. Recall that a policy P is represented by $\mathbf{z}_P \in \{0,1\}^{2N\delta_{\mathsf{P}}}$, a disclosing function F is specified by two matrices $\mathbf{G}_1 \in \{0,1\}^{\bar{k} \times \bar{k}^2}$, $\mathbf{G}_2 \in \{0,1\}^{\bar{k} \times \bar{k}}$, and the attribute $\mathbf{x} = (s_{k_1}, \ldots, s_{k_1+k_2-1})^\top$ is divided into chunks $\mathbf{s}_i = (s_{i\cdot\rho}, s_{i\cdot\rho+1}, \ldots, s_{i\cdot\rho+\rho-1})^\top$ for $i \in [\mu_1, \mu_1+\mu_2-1]$. Denote $\mathbf{b} = (\mathbf{z}_P^\top \mid \mathbf{x}^\top)^\top \in \{0,1\}^{\bar{k}}$.

Public inputs ξ_{D}: $\mathbf{A}_c,\ \mathbf{A}_0, \mathbf{A}_1,\ \mathbf{A}_{\mathsf{hp}}, m, \mathsf{com}_0,\ \ldots,\ \mathsf{com}_{\rho-1}, \mathsf{com}_P, \mathbf{r}_{\mathsf{com},0}, \ldots,$ $\mathbf{r}_{\mathsf{com},\mu_1-1}, \mathbf{G}_1, \mathbf{G}_2, t.$
Secret inputs clue: $\mathbf{z}_P \in \{0,1\}^{2N\delta_{\mathsf{P}}}$, $\mathbf{x} \in \{0,1\}^{k_2}$, $\mathbf{r}_{\mathsf{com},P} \in \{0,1\}^{nm}$, $\mathbf{r}_{\mathsf{com},\mu_1}$, $\ldots$, $\mathbf{r}_{\mathsf{com},\mu_1+\mu_2-1} \in \{0,1\}^{nm}$.
Prover's goal: Proving knowledge of the secret inputs clue such that

$$t = \mathbf{G}_1 \cdot (\mathbf{b} \otimes_{\mathrm{Kron}} \mathbf{b}) + \mathbf{G}_2 \cdot \mathbf{b} \mod 2, \tag{43}$$

and

$$\begin{cases} \mathsf{com}_i = \mathbf{A}_0 \cdot \mathbf{s}_i + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},i}, & \forall\ i \in [\mu_1, \mu_1+\mu_2-1]; \\ \mathsf{com}_P = \mathbf{A}_c \cdot \mathbf{h}_p + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},P}, & \text{with} \quad \mathbf{h}_P = \mathsf{rdec}(\mathbf{A}_{\mathsf{hp}} \cdot \mathbf{z}_P) \end{cases} \tag{44}$$

holds.

The substatement involving (43). As for equation (43), we perform the following extension step.

1. Extend $\mathbf{b} \otimes_{\mathrm{Kron}} \mathbf{b}$ to $\mathbf{w}_{\mathsf{D},0} = \mathsf{ext}_{\mathsf{mult}}(\mathbf{b} \otimes_{\mathrm{Kron}} \mathbf{b}) \in \{0,1\}^{4\bar{k}^2}$. Then based on (25), $\mathbf{b} \otimes \mathbf{b} = \mathbf{K}^{\star}_{\bar{k}^2} \cdot \mathbf{w}_{\mathsf{D},0}$.
2. Extend $\mathbf{b}$ to $\mathbf{w}_{\mathsf{D},1} = \mathsf{ext}_2(\mathbf{b}) \in \{0,1\}^{2\bar{k}}$. Based on (21), $\mathbf{b} = \mathbf{J}^{\star}_{\bar{k}} \cdot \mathbf{w}_{\mathsf{D},1}$.

Let $L_{\mathsf{D}} = 2\bar{k} + 4\bar{k}^2$ and $\mathbf{w}_{\mathsf{D}} = (\mathbf{w}_{\mathsf{D},0}^{\top} \mid \mathbf{w}_{\mathsf{D},1}^{\top})^{\top} \in \{0,1\}^{L_{\mathsf{D}}}$. One can easily form public matrix $\mathbf{M}_{\mathsf{D}}$ and vector $\mathbf{v}_{\mathsf{D}} = t$ such that (43) is equivalent to $\mathbf{M}_{\mathsf{D}} \cdot \mathbf{w}_{\mathsf{D}} = \mathbf{v}_{\mathsf{D}}$ for $\mathbf{M}_{\mathsf{D}}$.

Let us now specify the sets $\mathsf{VALID}_{\mathsf{D}} \subset \{0,1\}^{L_{\mathsf{D}}}$, $\mathcal{S}_{\mathsf{D}}$ and an associated permutation $\{\Gamma_\phi : \phi \in \mathcal{S}_{\mathsf{D}}\}$ such that the restrictions in (18) are obeyed.

Let $\mathsf{VALID}_{\mathsf{D}}$ contain all vectors $\mathbf{w} = (\mathbf{w}_0^{\top} \mid \mathbf{w}_1^{\top})^{\top}$ such that the following two conditions are applied to $\mathbf{w}_0, \mathbf{w}_1$, respectively.

- $\mathbf{w}_0 \in \{0,1\}^{2\bar{k}}$ and there exists a vector $\mathbf{w}'_0 \in \{0,1\}^{\bar{k}}$ satisfying $\mathbf{w}_0 = \mathsf{ext}_2(\mathbf{w}'_0)$.
- $\mathbf{w}_1 \in \{0,1\}^{4\bar{k}^2}$ and satisfying $\mathbf{w}_1 = \mathsf{ext}_{\mathsf{mult}}(\mathbf{w}'_0, \mathbf{w}'_0)$.

Let the set $\mathcal{S}_{\mathsf{D}} = \{0,1\}^{\bar{k}}$. For $\phi = \mathbf{f} \in \mathcal{S}_{\mathsf{D}}$, the permutation Γ_ϕ maps a vector $\mathbf{w} = (\mathbf{w}_0^{\top} \mid \mathbf{w}_1^{\top})^{\top} \in \mathbb{Z}^{L_{\mathsf{D}}}$ into

$$\Gamma_\phi(\mathbf{w}) = (F_{\mathsf{bin}}(\mathbf{f}, \mathbf{w}_0)^{\top} \mid F_{\mathsf{mult}}((\mathbf{f}, \mathbf{f}), \mathbf{w}_1)^{\top})^{\top}.$$

Based on the equivalences observed in (22), (26), it is verifiable that $\mathbf{w} \in \mathsf{VALID}_{\mathsf{D}}$ if and only if $\Gamma_\phi(\mathbf{w}) \in \mathsf{VALID}_{\mathsf{D}}$. In addition, if ϕ is sampled uniformly at random from $\mathcal{S}_{\mathsf{D}}$ and $\mathbf{w} \in \mathsf{VALID}_{\mathsf{D}}$, then $\Gamma_\phi(\mathbf{w})$ is evenly distributed in $\mathsf{VALID}_{\mathsf{D}}$. Therefore, we have reduced the statement involving (43) to an instance of $\mathcal{R}_{\mathsf{abstract}}$ with the modulus 2.

The substatement involving (44). Since the relations in (44) have appeared in the relations $\mathcal{R}_{\mathsf{S}}$, we can apply the same techniques to reduce (44) to an equivalent form $\mathbf{M}_2 \cdot \mathbf{w}_2 = \mathbf{v}_2 \bmod q$ for $\mathbf{w}_2 \in \mathsf{VALID}_2 \subset \{-1,0,1\}^{L_2}$ with associated $\mathcal{S}_2$ and $\{\Gamma_\phi : \phi \in \mathcal{S}_2\}$ such that the constraints in (18) hold.

Similar to Section 4.4, we can design $\mathsf{VALID}_{\mathsf{disclose}}, \mathcal{S}_{\mathsf{disclose}}$ and associated permutation $\{\Gamma_\phi : \phi \in \mathcal{S}_{\mathsf{disclose}}\}$ so that (18) are all satisfied and $\mathsf{VALID}_{\mathsf{disclose}}$ contains our secret vector $(\mathbf{w}_2^\top \mid \mathbf{w}_{\mathsf{D}}^\top)^\top \in \{0,1\}^{L_2+L_{\mathsf{D}}}$. We omit the details due to similarities.

Putting Pieces Altogether. At this point, our desired argument protocol works as follows. Given the public inputs ξ_{D}, both parties construct matrices $\mathbf{M}_2, \mathbf{M}_{\mathsf{D}}$ and $\mathbf{v}_2, \mathbf{v}_{\mathsf{D}}$ as above. The prover additionally constructs $\mathbf{w}_2, \mathbf{w}_{\mathsf{D}}$. Next, they run the protocol described in Table 2. If the commitment scheme COM is statistically hiding and computationally binding, the resulting protocol is a statistical ZKAoK protocol with perfect completeness, soundness error $2/3$, and communication cost

$$\mathcal{O}(L_2 \cdot \log q + L_{\mathsf{D}}) = \mathcal{O}\big(\ k_2 \cdot \log \lambda + \mu_2 \lambda \cdot (\log \lambda)^2 + N^2 \cdot \log^2(K+N) + k_2^2\ \big).$$

G Deferred Security Proofs for the Lattice-Based Construction

G.1 Auxiliary Algorithms for the Lattice-Based Construction

To show that our construction satisfies the requirements as defined in Section 2.2, we would need the following auxiliary algorithms.

$\mathsf{SimSetup}(1^\lambda)$**:** The simulated setup algorithm is almost the same as the real Setup algorithm. The only difference is that the hash function $\mathcal{H}_{\mathsf{FS}}$ is modelled as a random oracle. It then outputs $\mathsf{PP}, \mathsf{msk}_{\mathsf{X}}, \mathsf{msk}_{\mathsf{P}}$. We consider the capability of modelling the hash function and rewinding the adversary as the trapdoor tr.

$\mathsf{SimSign}(\mathsf{tr}, m, P, \mathbf{x}, w)$**:** If $P(m, \mathbf{x}, w) = 0$, return 0. Otherwise, this algorithm generates a simulated signature in the following manner.

1. First, it commits to m honestly as in the Sign algorithm. Sample randomnesses $\mathbf{r}_{\mathsf{com},0}, \ldots, \mathbf{r}_{\mathsf{com},\mu_1-1} \xleftarrow{\$} \{0,1\}^{nm}$ and compute $\mathsf{com}_i = \mathbf{A}_0 \cdot \mathbf{s}_i + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},i}$ for $i \in [0, \mu_1 - 1]$.
2. Next, it commits to all-zero strings instead of real $\mathbf{s}_{\mu_1}, \ldots, \mathbf{s}_{\rho-1}$ and $\mathbf{h}_P$. Specifically, sample $\mathbf{r}_{\mathsf{com},\mu_1}, \ldots, \mathbf{r}_{\mathsf{com},\rho-1}, \mathbf{r}_{\mathsf{com},P} \xleftarrow{\$} \{0,1\}^{nm}$ and let $\mathsf{com}_i = \mathbf{A}_0 \cdot \mathbf{0} + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},i}$ for $i \in [\mu_1, \rho-1]$ and $\mathsf{com}_P = \mathbf{A}_c \cdot \mathbf{0} + \mathbf{A}_1 \cdot \mathbf{r}_{\mathsf{com},P}$.
3. Next, it generates a simulated proof Π instead of a faithfully generated one. Specifically, it runs the simulator of the argument system in Section 4.4 κ times, obtaining κ number of commitments $\mathsf{COM}'_0, \ldots, \mathsf{COM}'_{\kappa-1}$. Next, it programs the random oracle $\mathcal{H}_{\mathsf{FS}}$ as

$$(\mathsf{CH}'_0, \ldots, \mathsf{CH}'_{\kappa-1})^\top = \mathcal{H}_{\mathsf{FS}}(\{\mathsf{COM}'_i\}_{i=0}^{\kappa-1},\ \xi)$$

with ξ as in (7). Then it produces responses $\mathsf{RSP}'_0, \ldots, \mathsf{RSP}'_{\kappa-1}$ accordingly. Let the resultant simulated proof be

$$\Pi = \big(\ \{\mathsf{COM}'_i\}_{i=0}^{\kappa-1},\ \{\mathsf{CH}'_i\}_{i=0}^{\kappa-1},\ \{\mathsf{RSP}'_i\}_{i=0}^{\kappa-1}\ \big).$$

4. Let $\Sigma = (\mathsf{com}_m, \mathsf{com}_{\mathbf{x}}, \mathsf{com}_{else}, \mathsf{com}_P, \mathbf{r}_{\mathsf{com},m}, \Pi)$. Return Σ.

$\mathsf{SimDisclose}(m, \Sigma, \mathsf{tr}, P, \mathbf{x}, F)$**:** Given all the inputs, this simulated disclosing algorithm proceeds as follows.

1. Compute $t = F \cdot (\mathbf{z}_P^\top, \mathbf{x}^\top)^\top$.
2. Generate a simulated proof a by running the simulator of the argument system in Appendix F κ times and programming the random oracle $\mathcal{H}_{\mathsf{FS}}$ as in SimSign algorithm. Let the resultant simulated proof be $a = \left(\{\mathsf{COM}_{\mathsf{D},i}\}_{i=0}^{\kappa-1}, \{\mathsf{CH}_{\mathsf{D},i}\}_{i=0}^{\kappa-1}, \{\mathsf{RSP}_{\mathsf{D},i}\}_{i=0}^{\kappa-1}\right)$.
3. Return the testimony-attestation pair (t, a).

$\mathsf{Extract}(\mathsf{tr}, m, \Sigma)$**:** If $\mathsf{Verify}(m, \Sigma) = 0$, return 0. Otherwise, this algorithm outputs $(P^*, \mathbf{x}^*, w^*)$. Parse $\Sigma = (\{\mathsf{com}_i\}_{i=0}^{\rho-1}, \mathsf{com}_P, \mathbf{r}_{\mathsf{com},0}, \ldots, \mathbf{r}_{\mathsf{com},\mu_1-1}, \Pi)$, let the proof $\Pi = (\{\mathsf{COM}_i\}_{i=0}^{\kappa-1}, \{\mathsf{CH}_i\}_{i=0}^{\kappa-1}, \{\mathsf{RSP}_i\}_{i=0}^{\kappa-1})$, and the public input ξ be of the form (7). First of all, we argue that $\mathcal{A}$ must have queried the random oracle $\mathcal{H}_{\mathsf{FS}}$ on input $(\{\mathsf{COM}_i\}_{i=0}^{\kappa-1}, \xi)$, since otherwise, the probability that guessing correctly $\mathcal{H}_{\mathsf{FS}}(\{\mathsf{COM}_i\}_{i=0}^{\kappa-1}, \xi)$ is at most $3^{-\kappa}$, which is negligible. Let $q_{\mathcal{H}_{\mathsf{FS}}}$ be the total number of oracle queries to $\mathcal{H}_{\mathsf{FS}}$ and $(\{\mathsf{COM}_i\}_{i=0}^{\kappa-1}, \xi)$ be the t-th oracle query with $1 \leq t \leq q_{\mathcal{H}_{\mathsf{FS}}}$. Now algorithm Extract picks t as the target forking point and replays $\mathcal{A}$ polynomial times with the same random tape and input as in the original run. In each rerun, for the first $t-1$ queries, $\mathcal{A}$ is given the same answers $rply_1, \cdots, rply_{t-1}$ as in the original run, while from the t-th query onwards, algorithm Extract replies with fresh random values $rply'_t, \cdots, rply'_{q_{\mathcal{H}_{\mathsf{FS}}}} \leftarrow \{1,2,3\}^\kappa$. According to the Improved Forking Lemma by Brickell et al. [10], with probability larger than $1/2$, algorithm Extract can obtain a 3-fork involving the tuple $(\{\mathsf{COM}_i\}_{i=0}^{\kappa-1}, \xi)$ after polynomial-time executions of algorithm $\mathcal{A}$. Let $rply_t^{(1)} = \left(\mathsf{CH}_{t,0}^{(1)}, \cdots, \mathsf{CH}_{t,\kappa-1}^{(1)}\right)^\top$; $rply_t^{(2)} = \left(\mathsf{CH}_{t,0}^{(2)}, \cdots, \mathsf{CH}_{t,\kappa-1}^{(2)}\right)^\top$; $rply_t^{(3)} = \left(\mathsf{CH}_{t,0}^{(3)}, \cdots, \mathsf{CH}_{t,\kappa-1}^{(3)}\right)^\top$ be the replies to the $\mathcal{H}_{\mathsf{FS}}$ queries with respect to the 3-fork branches. It is easy to see that

$$\mathsf{Pr}\left[\exists\ j \in [0, \kappa-1] : \{\mathsf{CH}_{t,j}^{(1)}, \mathsf{CH}_{t,j}^{(2)}, \mathsf{CH}_{t,j}^{(3)}\} = \{1,2,3\}\right] = 1 - (7/9)^\kappa.$$

Conditioned on the existence of such index j, one can parse the 3 responses corresponding to the fork branches to obtain $\left(\mathsf{RSP}_{t,j}^{(1)}, \mathsf{RSP}_{t,j}^{(2)}, \mathsf{RSP}_{t,j}^{(3)}\right)$. They are indeed 3 valid responses with respect to all 3 possible challenges for the same commitment COM_j. Now algorithm Extract can run the knowledge extractor as in Theorem 5 to extract $\mathbf{w}_1^*, \mathbf{w}_{\mathsf{P}}^*$ such that $(\mathbf{w}_1^{*\top} \mid \mathbf{w}_{\mathsf{P}}^{*\top})^\top \in \mathsf{VALID}_{\mathsf{baps}}$, $\mathbf{M}_1 \cdot \mathbf{w}_1^* = \mathbf{v}_1 \bmod q$, and $\mathbf{M}_{\mathsf{P}} \cdot \mathbf{w}_{\mathsf{P}}^* = \mathbf{v}_{\mathsf{P}} \bmod 2$. Parse $\mathbf{w}_1^*$ as

$$\mathbf{w}_1^* = (\mathbf{w}_{\mathsf{X}}^{*\top} \mid \mathbf{w}_{\mathsf{cert}}^{*\top} \mid \mathbf{w}_{\mathsf{com}}^{*\top} \mid \mathbf{w}_{g,0}^{*\top} \mid \cdots \mid \mathbf{w}_{g,N-1}^{*\top} \mid \mathbf{w}_{h,0}^{*\top} \mid \cdots \mid \mathbf{w}_{h,N-1}^{*\top})^\top. \quad (45)$$

Backtracking the transformations we have performed, we are able to obtain $(\mathbf{x}^*, \mathsf{sk}^*)$, $(\mathbf{z}^*, \mathbf{h}^*, \mathsf{Cert}^*)$, and $(\mathbf{s}_{\mu_1}^*, \ldots, \mathbf{s}_{\rho-1}^*, \{\mathbf{r}_{\mathsf{com},i}^*\}_{i=\mu_1}^{\rho-1}, \mathbf{r}_{\mathsf{com},P}^*, \mathbf{h}^*)$ from $\mathbf{w}_{\mathsf{X}}^*$, $\mathbf{w}_{\mathsf{cert}}^*$, $\mathbf{w}_{\mathsf{com}}^*$ such that they satisfy equations (3), (4), (5), respectively. Note that the first k_2 bits from $(\mathbf{s}_{\mu_1}^{*\top} \mid \cdots \mid \mathbf{s}_{\rho-1}^{*\top})^\top$ should be $\mathbf{x}^*$ by the soundness

of the argument system underlying the Sign algorithm and let the next k_3 bits be w^*. Also the string $\mathbf{z}^*$ specifies a policy P^*. The $\mathsf{Extract}$ algorithm then outputs $(P^*, \mathbf{x}^*, w^*)$.

In this following, we prove the simulatability and extractability of the proposed concrete BAPS scheme.

G.2 Proof of Theorem 3

Proof. We proceed by defining a sequence of indistinguishable hybrid games between a challenger and an adversary where the first is the experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{sim}}(\lambda)$ with b set to 1 while the last is $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{sim}}(\lambda)$ with b set to 0.

We start with $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{sim}|b=1}(\lambda)$. Our first change is to replace the Setup algorithm by the $\mathsf{SimSetup}$ algorithms. This change is unnoticeable by the adversary since $\mathsf{SimSetup}$ is the same as Setup in the random oracle model.

For the signing queries, we replace the proof Π by a simulated one. This can be achieved since the challenger has the capability to program the random oracle $\mathcal{H}_{\mathsf{FS}}$. By the statistical zero-knowledge property of the underlying argument system, this game is indistinguishable from the preceding one. Note that in this game, all $\mathsf{com}_0, \ldots, \mathsf{com}_{\rho-1}, \mathsf{com}_P$ are honestly generated as commitments to $\mathbf{s}_0, \ldots, \mathbf{s}_{\rho-1}, \mathbf{h}_P$, respectively. Therefore, the challenger has all the needed information to generate a real proof for the disclosing queries.

Then for the disclosing queries, we replace the proof a by a simulated one. By the same argument as above, this modification is achievable and indistinguishable to the adversary due to the statistical zero-knowledge of the second argument system. At this point, the challenger generates simulated proofs for signing queries and disclosing queries even though it has all the information for generating real proofs.

Next, we focus on modifying the signing queries again. Specifically, we now consider changing the commitments $\mathsf{com}_{\mu_1}, \ldots, \mathsf{com}_{\rho-1}, \mathsf{com}_P$. For $i \in [\mu_1, \rho - 1]$, com_i is now a commitment of an all-zero string instead of $\mathbf{s}_i$ and com_P is a commitment of an all-zero string instead of $\mathbf{h}_P$. Due to the statistically hiding property of the $\mathsf{CMT}_{\rho',m}$ and $\mathsf{CMT}_{\ell,m}$, the advantage of the adversary to distinguish this modification is negligible. This game is in fact $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{sim}|b=0}(\lambda)$, which concludes the proof.

□

G.3 Proof of Theorem 4

Proof. To show our lattice-based construction is extractable, we have to prove that the advantages of any PPT adversary $\mathcal{A}$ in $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{sound}}(\lambda)$, $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{Uf\text{-}I}}(\lambda)$, and $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{Uf\text{-}II}}(\lambda)$ are negligible in λ.

Soundness. Consider the experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{sound}}(\lambda)$ and let ϵ be the advantage of $\mathcal{A}$. Suppose $\mathcal{A}$ wins and denote (m, Σ) as the output of $\mathcal{A}$. The experiment returns 1 either at Line 7 or Line 9 in Fig. 2. Therefore, (m, Σ) is not from

querying the oracles and $\mathsf{Verify}(m, \Sigma) = 1$. We then run $\mathsf{Extract}(\mathsf{tr}, m, \Sigma)$ and let

$$\mathbf{w}_1^* = (\mathbf{w}_{\mathsf{X}}^{*\top} \mid \mathbf{w}_{\mathsf{cert}}^{*\top} \mid \mathbf{w}_{\mathsf{com}}^{*\top} \mid \mathbf{w}_{g,0}^{*\top} \mid \cdots \mid \mathbf{w}_{g,N-1}^{*\top} \mid \mathbf{w}_{h,0}^{*\top} \mid \cdots \mid \mathbf{w}_{h,N-1}^{*\top})^\top$$

be the intermediate outputs of $\mathsf{Extract}$. Similar to how we get $P^*, \mathbf{x}^*, w^*$, from $\{\mathbf{w}_{g,i}^*\}_{i=0}^{N-1}, \{\mathbf{w}_{h,i}^*\}_{i=0}^{N-1}, \mathbf{w}_{\mathsf{P}}^*$, we can obtain

$$\{(\widetilde{\mathsf{com}}_{g,i}^*, y_i^*, \widetilde{\mathbf{s}}_i^*, \mathbf{r}_{g,i}^*, a_{i,0}^*, \ldots, a_{i,\rho-1}^*, b_{i,0}^*, \ldots, b_{i,\rho-1}^*)\}_{i\in[0,N-1]}; \tag{46}$$

$$\{(\widetilde{\mathsf{com}}_{h,i}^*, z_i^*, \widetilde{\mathbf{t}}_i^*, \mathbf{r}_{h,i}^*, c_{i,0}^*, \ldots, c_{i,\rho-1}^*, d_{i,0}^*, \ldots, d_{i,\rho-1}^*)\}_{i\in[0,N-1]}; \tag{47}$$

$$(y_0^*, \ldots, y_{N-1}^*, z_0^*, \ldots, z_{N-1}^*, s_K^*, s_{K+1}^*, \ldots, s_{K+N-2}^*). \tag{48}$$

such that (46) and (48) satisfy equations (10) and (11), respectively. Note that (47) should satisfy equations similar to (10). By the soundness of the argument system underlying the signing algorithm, we have the following observations.

- The last $N-1$ bits from $(\mathbf{s}_{\mu_1}^{*\top}, \ldots, \mathbf{s}_{\rho-1}^{*\top})^\top$ should be $(s_K^*, \ldots, s_{K+N-2}^*)$.
- $\{y_i^*, z_i^*\}_{i\in[0,N-1]}$ appeared in (46)(47) should be the same as those appeared in (48).
- For $i \in [0, N-1]$, y_i^* is the $b_{g,i}$-th bit among $\widetilde{\mathbf{s}}_i^*$ and $\widetilde{\mathsf{com}}_{g,i}^*$ is $\mathsf{com}_{a_{g,i}}$. Here $a_{g,i}, b_{g,i}$ are computed as in (9) with respect to policy P^*.

The computational binding property of $\mathsf{CMT}_{\rho',m}$ assures that $\mathbf{s}_{a_{g,i}}^* = \widetilde{\mathbf{s}}_i^*$ and in particular $s_{g(i)}^* = y_i^*$ for $i \in [0, N-1]$. By the same arguments, we have $s_{h(i)}^* = z_i^*$ for $i \in [0, N-1]$.

As a result, the tuple in (48) satisfies (11) implies that

$$(\ s_{g(0)}^*,\ \ldots,\ s_{g(N-1)}^*,\ s_{h(0)}^*,\ \ldots,\ s_{h(N-1)}^*,\ s_K^*,\ s_{K+1}^*,\ \ldots,\ s_{K+N-2}^*\ )$$

satisfies (2). The latter in turn is equivalent to $P^*(m, \mathbf{x}^*, w^*) = 1$. Therefore, assuming the binding property of the commitment scheme $\mathsf{CMT}_{\rho',m}$ and the soundness of the first argument system, the probability of outputting 1 at Line 7 is negligible.

Let us now consider the second stage where the adversary outputs (F, t, a). Then $(m, \Sigma, F, t, a) \notin Q_D$ and $\mathsf{Judge}(\mathsf{PP}, m, \Sigma, F, t, a) = 1$. By the soundness of the argument system underlying the disclosing algorithm, similar to the above $\mathsf{Extract}$ algorithm, we can also extract $\tilde{\mathbf{z}},\ \tilde{\mathbf{h}},\ \tilde{\mathbf{x}},\ \tilde{\mathbf{r}}_{\mathsf{com},P},\ \{\tilde{\mathbf{r}}_{\mathsf{com},i}\}_{i\in[\mu_1,\mu_1+\mu_2-1]}$ such that the following equations hold.

$$\begin{aligned} t &= F \cdot (\tilde{\mathbf{z}}, \tilde{\mathbf{x}}); \\ \mathsf{com}_P = \mathbf{A}_c \cdot \tilde{\mathbf{h}} + \mathbf{A}_1 \cdot \tilde{\mathbf{r}}_{\mathsf{com},P}, &\quad \tilde{\mathbf{h}} = \mathsf{rdec}(\mathbf{A}_{\mathsf{hp}} \cdot \tilde{\mathbf{z}}); \\ \mathsf{com}_i = \mathbf{A}_0 \cdot \tilde{\mathbf{s}}_i + \mathbf{A}_1 \cdot \tilde{\mathbf{r}}_{\mathsf{com},i} &\ \forall\ i \in [\mu_1, \mu_1+\mu_2-1]. \end{aligned}$$

Let $\tilde{P}$ be the policy specified by $\tilde{\mathbf{z}}$. On the one hand, the winning condition of $\mathcal{A}$ implies that $t = F \cdot (\tilde{\mathbf{z}}, \tilde{\mathbf{x}}) \neq F(\mathbf{z}^*, \mathbf{x}^*)$. This immediately implies that

$$(\tilde{\mathbf{z}}, \tilde{\mathbf{x}}) \neq (\mathbf{z}^*, \mathbf{x}^*). \tag{49}$$

On the other hand, by the computational binding property of the commitment schemes $\mathsf{CMT}_{\ell,m}$ and $\mathsf{CMT}_{\rho',m}$, it must be that

$$(\tilde{\mathbf{h}}, \tilde{\mathbf{x}}) = (\mathbf{h}^*, \mathbf{x}^*). \tag{50}$$

Combining (49) and (50), we obtain that $\tilde{\mathbf{z}} \neq \mathbf{z}^*$ yet $\tilde{\mathbf{h}} = \mathbf{h}^*$. This, however, violates the collision resistance of the hash function family $\mathcal{H}_{m_\mathsf{P}}$. Therefore, assuming the binding property of the two commitment schemes, the soundness of the second argument system, and the collision resistance of the hash function $\mathcal{H}_{m_\mathsf{P}}$, the probability of outputting 1 at Line 9 is also negligible.

Thus the probability of $\mathcal{A}$ winning $\mathbf{Exp}^{\mathsf{sound}}_{\mathcal{A}}(\lambda)$ is negligible in λ.

Unforgeability-I. Let (m, Σ) be the output of the adversary $\mathcal{A}$ in the experiment $\mathbf{Exp}^{\mathsf{Uf\text{-}I}}_{\mathcal{A}}(\lambda)$. If $\mathcal{A}$ wins the experiment with non-negligible probability ϵ_1, then we construct a PPT algorithm $\mathcal{B}$ that breaks the EUF-CMA security of the DM scheme with non-negligible probability.

Given vk, $\mathcal{B}$ sets $\mathsf{vk}_\mathsf{X} = \mathsf{vk}$ and then generates all other parameters faithfully as in the Setup algorithm described in Section 4.2. Next, $\mathcal{B}$ internally runs $\mathcal{A}$ by feeding $\mathcal{A}$ with PP and msk_P. $\mathcal{B}$ can simulates $\mathcal{A}$'s view either by querying its own oracle or programming the hash function $\mathcal{H}_\mathsf{FS}$. Eventually, $\mathcal{A}$ outputs a forgery (m, Σ). Suppose $\mathcal{A}$ wins the experiment. $\mathcal{B}$ then proceeds in the following manner.

First, $\mathcal{B}$ runs $\mathsf{Extract}(\mathsf{tr}, m, \Sigma)$, obtaining $(\mathbf{x}^*, \mathsf{sk}^*, P^*, \mathbf{h}^*, \mathsf{Cert}^*, w^*)$. Due to the soundness of the first argument system, $P^*(m^*, \mathbf{x}^*, w^*) = 1$ and $(\mathbf{x}^*, \mathsf{sk}^*)$ is a valid message-signature pair for the DM scheme with overwhelming probability. Furthermore, $\mathbf{x}^* \notin Q_X$ implies that $\mathcal{B}$ did not query its own oracle for any signature on $\mathbf{x}^*$. Then $\mathcal{B}$ outputs $(\mathbf{x}^*, \mathsf{sk}^*)$, which is a valid forgery of the DM scheme. Therefore, if $\mathcal{A}$ wins the experiment with non-negligible probability, then we can construct an algorithm $\mathcal{B}$ which breaks the EUF-CMA security of the DM scheme with non-negligible probability as well. Due to the security of the DM scheme, ϵ_1 is negligible.

Unforgeability-II. Let ϵ_2 be $\mathcal{A}$'s advantage in the experiment $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathcal{A}}(\lambda)$. If ϵ_2 is non-negligible, we construct an algorithm $\hat{\mathcal{B}}$ that breaks the EUF-CMA security of the DM scheme.

On receiving vk from its challenger, $\hat{\mathcal{B}}$ sets $\mathsf{vk}_\mathsf{P} = \mathsf{vk}$ and then creates all other parameters as in the Setup algorithm. Next, $\hat{\mathcal{B}}$ runs the adversary $\mathcal{A}$ on PP and msk_X. When $\mathcal{A}$ makes a $\mathcal{O}_{\mathsf{PolicyKey}}$ query, $\hat{\mathcal{B}}$ computes Cert_P by querying its signing oracle on $\mathbf{h}_P = \mathsf{rdec}(\mathbf{A}_{\mathsf{hp}} \cdot \mathbf{z}_P)$. When $\mathcal{A}$ makes $\mathcal{O}_{\mathsf{Sign}}$ and $\mathcal{O}_{\mathsf{Dis}}$ queries, $\hat{\mathcal{B}}$ computes the required signature/testimony-attestation pair by programming the hash function $\mathcal{H}_\mathsf{FS}$. In the end, $\mathcal{A}$ outputs a forgery (m, Σ). Suppose $\mathcal{A}$ wins, $\hat{\mathcal{B}}$ works as follows.

Run the Extract algorithm, obtaining a tuple $(\mathbf{x}^*, \mathsf{sk}^*, P^*, \mathbf{h}^*, \mathsf{Cert}^*, w^*)$. With all but negligible probability, $P^*(m^*, \mathbf{x}^*, w^*) = 1$ and $(\mathbf{h}^*, \mathsf{Cert}^*)$ is a valid message-signature pair for the DM scheme thanks to the soundness property of the first argument system underlying the Sign algorithm. In addition, the fact that $\mathcal{A}$ wins the game implies that $P^* \notin Q_P$. Let us consider the following two cases.

Case 1: There exists $P \in Q_P$ such that $P^* \neq P$ and $\mathbf{h}^* = \mathbf{h}_P = \mathsf{rdec}(\mathbf{A}_{\mathsf{hp}} \cdot \mathbf{z}_P)$. This, however, breaks the collision resistance of the hash function family $\mathcal{H}_{m_{\mathsf{P}}}$.

Case 2: For all $P \in Q_P$, $\mathbf{h}^* \neq \mathbf{h}_P$. As a result, $\hat{\mathcal{B}}$ did not query its signing oracle for any signature on $\mathbf{h}^*$. $\hat{\mathcal{B}}$ then outputs $(\mathbf{h}^*, \mathsf{Cert}^*)$, which is a valid forgery for the DM scheme.

In the random oracle model, assuming the soundness of the first argument system, the collision resistance of the hash function family $\mathcal{H}_{m_{\mathsf{P}}}$, the security of the DM scheme, the advantage of the adversary $\mathcal{A}$ in experiment $\mathbf{Exp}^{\mathsf{Uf\text{-}II}}_{\mathcal{A}}(\lambda)$ has to be negligible.

□