

Nowhere Man: Device Binding for Post-Quantum Anonymous Credentials

Calvin Abou Haidar, Ryo Kikuchi*, Octavio Perez-Kempner, and Marcel Tiepelt

NTT Social Informatics Laboratories

Tokyo, Japan

Abstract—Anonymous credentials are seeing widespread deployment. Notably, the European Commission plans to roll out age-verification solutions based on anonymous credentials by the end of 2026. Similarly, Google open-sourced Longfellow, a zero-knowledge proof system that lifts the standardized mobile Driver License framework into a full-fledged anonymous credentials, and is currently being integrated with Google Wallet.

Any secure deployment of anonymous credentials must address two major challenges: 1) provide guarantees against credential sharing, and 2) support proof-of-personhood. The standard approach adopted by the industry to address both is *device binding*, which is mandatory for mDL and was only recently formalized for anonymous credentials by Friedrichs *et al.* (Eurocrypt '26). Simply put, the idea is to have the user's phone Secure Element sign a fresh challenge from the verifier, and to prove knowledge of this signature in zero-knowledge at each credential presentation.

To date, all existing approaches, including Vega (S&P '26), have only considered non-post-quantum solutions based on legacy signature schemes. Crucial for near-term deployments and adoption is to work with *standardized* post-quantum signature schemes, such as ML-DSA, which were not designed for privacy. Because of this, solutions based on general non-interactive zero-knowledge proof systems (e.g., Longfellow) appear to be needed. This genericity, however, requires proving complex statements, significantly impacting proving time.

In this work, we present a modular device binding protocol tailored to work with ML-DSA. We avoid the pitfalls of the generic approach by leveraging the structure of ML-DSA resulting in an efficient solution, but retaining the security underlying the post-quantum standard. Our most efficient variants additionally rely on a weak-MLWR assumption for unlinkability, which can be removed at the cost of lower efficiency. While tailored for ML-DSA and requiring no modification to the FIPS 204 signing interface, our solution is modular enough to be integrated with different anonymous credential schemes, allowing real-world solutions to pick the most efficient building blocks. To demonstrate the feasibility of our solution, we provide a proof-of-concept implementation.

Index Terms—ML-DSA, Device Binding, Anonymous Credentials

1. Introduction

Although anonymous credentials were initially introduced in the mid 80s [1], [2] and efficiently realized as early as in the late 90s through a series of works by Camenisch and Lysyanskaya [3], [4], [5], they have only recently gained traction in real-world applications. Recent examples include Privacy Pass [6] and related follow-up works on rate-limiting anonymous credentials that are being actively discussed in the IETF/IRTF ecosystem [7], [8], [9], and adopted by major companies like Apple, Cloudflare and Google. While more ambitious digital identity frameworks have emerged in the blockchain ecosystem, these remain far from mass adoption.

In this work, we focus on government-backed large scale deployments of digital identity frameworks. One such case is the EUDI Wallet, a mobile identity wallet created by Regulation (EU) 2024/1183 and scheduled for release by the end of 2026. Notably, it must provide selective disclosure and unlinkable authentication—core features of anonymous credentials. This strict cryptographic requirement has triggered numerous technical discussions and proposals concerning the design and deployment of its first version [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20]. The US government is also progressing in a similar direction with the Transportation Security Administration (TSA) Digital ID [21] already being available for use in several states. Although the TSA Digital ID is a (plain) verifiable credential based on the mDL standard (ISO/IEC 18013-5, [22]), improvements to upgrade such systems into full-fledged anonymous credentials are already underway. Concretely, Google recently developed and open sourced a zero-knowledge proof system called Longfellow [23] that is designed to work on top of mDL credentials. Google is actively pushing Longfellow for standardization and adoption both in the US through Google's Wallet ID Pass as well as within the EU as candidate to implement the EUDI Wallet [24], [25], [26], [27]. It is also worth mentioning that Microsoft has recently proposed an alternative to Longfellow called Vega [28], [29], highlighting the interest of big tech companies in the space.

Unlike deployments tailored to use cases where technology adoption can be flexible, such as the blockchain ecosystem, large scale deployments in regulated environments result in stricter requirements particular to the setting. These requirements include (1) the need for a proof-of-personhood to prevent credential sharing, (2) compatibility

*Ryo Kikuchi is now with the Secretariat, Personal Information Protection Commission, Japan. He contributed to this work while with NTT.

Table 1: Overview of State-of-the-art anonymous credential schemes with device binding.

Scheme	Device Binding	Infrastructure Compatibility	Industry Compatibility	Design Model	PQ
Longfellow [23]	ECDSA	Good (No changes required)	Poor (Only recently proposed for standardization at IETF [27])	Monolithic	Plausible
Vega [28]	ECDSA	Good (No changes required)	Very Poor (No known efforts to standardize its core building blocks)	Monolithic	No
BBS-based [16], [30], [31]	ECDSA	Poor (Issuer changes required)	Good (Undergoing standardization at IETF [32], ISO [33] and ETSI [34])	Modular	No

with standardized primitives to ease deployment, (3) interoperability and a modular design to support extensions in the long run, and, looking ahead, (4) post-quantum security.

The first requirement (1), providing guarantees against credential sharing while revealing nothing else beyond what is disclosed by the user during a presentation, appears to be in conflict with the unlinkability feature of anonymous credentials. At the same time, with the rise of AI agents, many services online will require such a proof-of-personhood so to verify that the request is being triggered by a human.

This requirement can be achieved by means of device binding, a mechanism consisting of securely storing a cryptographic signing key¹ within the user’s phone Secure Element (SE)², and linking the corresponding public key to the user’s attributes. To obtain a credential, the user creates a key pair with the SE, receives the corresponding public key (the secret key never leaves the SE), and registers it as an attribute with the credential Issuer. To generate a fresh presentation, the user is required to prove that it can obtain a signature on a nonce specified by the Verifier under the (hidden) public key in its credential. Security of the SE, alongside the phone PIN or biometric check, guarantees that the rightful user (also a human) is triggering the presentation while providing non-transferability. Device binding has initially been adopted by the industry when it was specified for mDL and, very recently, also formalized in the context of anonymous credentials [13].

With the above in mind, in the remainder of this work we consider device binding in a purely cryptographic sense, focusing on the non-transferability it provides. The biometric control checks (or related usability controls) that are also required to realize full proof-of-personhood in the above sense are outside the scope of this work, and we assume them to be implemented whenever the user accesses the SE.

1.1. Recent Approaches & Open Challenges

Taking the ongoing discussions surrounding the development of the EUDI Wallet as a departure point, we identify two main approaches to constructing anonymous credentials with device binding: those based on general-purpose zero-

knowledge proofs and those based on “proof-friendly” signatures. The former includes Longfellow and Vega, as well as other works such as Crescent [36] and zk-Creds [37]. The latter includes an extensive body of work on BBS credentials [12], [13], [14], [15], [16], [17], [18], [19], [30], [31].

In Table 1, we provide a summary of the state-of-the-art contenders being discussed/proposed to implement the EUDI Wallet [38]. We discuss them in more detail below.

Longfellow and Vega. Both solutions construct circuits to efficiently prove a user’s possession of a device bound public key signed by an Issuer. They achieve this by proving (1) possession of a (signature₁, message) pair that verifies under the Issuer’s public key and (2) of a (signature₂, dpk) that verifies for a nonce (chosen by the Verifier) where dpk is the Secure Element’s device public key encoded as a user attribute in message. Longfellow follows the recipe from [39] based on running an interactive proof (the sumcheck protocol), then committing to its transcript and proving its validity in zero-knowledge. To that end, Longfellow builds on recent techniques from [40], [41] and the Ligerio proof system [42], together with several optimizations to arithmetize ECDSA and SHA-256 and to tailor the circuit to the standard mDL/MDOC format. Vega, on the other hand, relies on very recent folding techniques [43], [44] and the Spartan proof system [45], pre-computing a significant part of the proof and re-using it via re-randomizable commitments across presentations. This, together with an efficient lookup-centric arithmetization, allows Vega to achieve smaller execution times than Longfellow. However, security-wise, Vega’s use of folding schemes relies on the framework of [46], which does not support a traditional straight-line extraction algorithm as Ligerio does for Longfellow. Furthermore, Vega uses non-post-quantum building blocks and relies on less-scrutinized schemes, positioning it as a much less conservative alternative than Longfellow.

BBS Credentials. This paradigm relies on a signature scheme, a commitment scheme and non-interactive proof of knowledge. Informally, the user commits to its attributes and obtains a signature on the commitment from the issuer. A credential showing consists of proving in zero-knowledge the possession of a valid signature on a commitment (or set of commitments) that open to the attributes being shown. Importantly, the signature scheme must be “proof-friendly” in the sense that it should allow efficient proofs of possession. Several recent works have proposed efficient device binding protocols for BBS [16], [30], [31]. Compared to

1. Unsurprisingly, yet another evidence that Cryptography is a tool for turning a whole swathe of problems into key management problems [35].

2. A Secure Element is a hardware module embedded in the device and designed to resist physical tampering and side-channel attacks, making key extraction extremely difficult.

the circuit-based approaches, BBS credentials offer a more modular framework, built upon simpler building blocks that have been long scrutinized by the community. However, deploying them requires changes on the Issuer side (it is no longer enough to simply issue a signed MDOC document) and they are inherently non post-quantum³. That said, given the on-going standardization efforts and recent progress in the field, they remain a solid candidate for consideration.

Regardless of the technology chosen for the first version of the EUDI Wallet and similar initiatives, there is an urgent need to consider post-quantum alternatives. The US National Institute of Standards and Technology (NIST) has decided to deprecate ECDSA and RSA after 2030, and to disallow them after 2035 [47]. An even more aggressive timeline is being pushed by Google, which has announced its own plan to migrate to post-quantum cryptography by 2029 [48]. Critically, this transition should be carried out with *crypto agility*, defined by NIST as the ability to replace and adapt cryptographic algorithms without interrupting running systems [49]. Arguably, the cryptographic community has not been particularly good at this: anonymous credentials are a clear example, as BBS credentials date back to 2004, yet their adoption and standardization efforts are still underway.

Looking at the current landscape of candidate constructions, only Longfellow could be adapted to a post-quantum signature scheme. In this regard, ML-DSA [50] emerges as the clearest candidate, given its rapid adoption by the industry. Prominent examples include the Linux kernel [51], Secure Elements and HSMs produced by major hardware manufacturers such as Thales [52], Samsung Exynos [53], and Apple’s core crypto library (recently open sourced [54]). Moreover, Samsung’s Secure Element S3SSE2A recently won the CES 2026 *Best Cybersecurity Innovation Award* as the first embedded security chip to feature hardware-based post-quantum cryptography (specifically, ML-DSA) [55].

Quite recently, Google started to explore the possibility of adapting Longfellow to ML-DSA as evidenced by their recent presentations at IETF [25], [26] and their open-sourced codebase tests [56]. Unfortunately, Longfellow’s monolithic design requires the verification of two ML-DSA signatures in *zero-knowledge* and just verifying one of them with respect to a (known) public key takes 850ms, making the resulting proof system impractical for real-world applications. The main source of inefficiency is that Longfellow takes the full ML-DSA verification algorithm as input and proves its correct execution in zero-knowledge, something that requires several calls to SHAKE [57].

Alternatively, one could consider a post-quantum analog of BBS credentials, but such an approach would require changes to the standard. To see why, we refer the reader to related work on randomizing ML-DSA signatures [58], [59], [60], [61], which is a necessary step to achieve the kind of proof-friendliness needed for device binding under the BBS approach. Another worrying aspect of adopting a randomizable version of ML-DSA is that it would require

an application-specific seed from which all public keys are derived, departing from established design practices.

The above-mentioned limitations stress the need to address the following question:

Can we construct a device binding protocol for ML-DSA that (i) is standard-compatible (i.e., requires no modification to the signing algorithm), (ii) can support a modular integration into different anonymous credential frameworks, and (iii) is concretely efficient?

1.2. Our Work: Nowhere Man

We propose the first efficient device binding protocol for ML-DSA that is compatible with different anonymous credential frameworks, does not require any modification to the standard, and achieves practical efficiency. Our most efficient variants rely on the weak-MLWR assumption for unlinkability (see Appendix C for further details), while unforgeability relies on SelfTargetMSIS with the same parameters as ML-DSA. We also discuss less efficient variants that avoid the weak-MLWR assumption and directly reduce unforgeability to ML-DSA.

Our initial observation is that compatibility with the standard is mandatory for signing but not for verification. Indeed, the SE should be treated in a black-box way as it seems unlikely that hardware vendors would be willing to extend their APIs to support additional features in the near future. Verification, however, happens within the anonymous credential’s application layer, where one is free to replace the standard verification algorithm with an alternative proof relation tailored to signatures produced by the FIPS 204 signing interface. This preserves compatibility with standard ML-DSA implementations while giving us the freedom to restructure the verification computations without sacrificing security. Consequently, we propose a verifier-side decomposition of ML-DSA’s verification into proof-friendly subrelations. First, we restructure the ML-DSA verification relation to avoid proving several computations that are unnecessary in our setting. This optimization is useful even when the whole relation is proved with a general-purpose system such as Longfellow. Second, we further decompose the relation into bit-oriented and “lattice-friendly” components, linked by a commitment. This decomposition introduces a modest overhead in exchange for greater interoperability between proof systems and anonymous credential frameworks.

As mentioned, anonymous credentials can be built from “proof-friendly” signatures: signature schemes that, combined with a commitment scheme and a compatible non-interactive zero-knowledge proof system, admit efficient proofs of knowledge of a valid signature on a commitment to the user’s attributes. At a minimum, they should also allow for selective disclosure of attributes. Many well-known schemes, including recent lattice-based constructions, follow this approach. To support device binding, the credential scheme should additionally allow for an efficient encoding of the device public key as a user attribute, enabling a proof of possession of said key. We frame our work in this setting

3. They provide everlasting privacy (zero-knowledge holds unconditionally) but soundness relies on the hardness of the discrete logarithm problem.

— though it readily adapts to generic zero-knowledge proof systems such as Longfellow (see Section 4) as mentioned.

To present the technical details of our protocol for an efficient proof of possession of a device public key, we need to provide a high-level overview of the ML-DSA standard, including the main algorithms for signing and verification.

The ML-DSA signature scheme can be seen as an adapted Schnorr signature on structured lattices. The overall signing process can be simplified as follows:

Given a (small) secret key s and a public key $\mathbf{A}, \mathbf{t} := \mathbf{A}s$, a signature for message m consists of a tuple $(\mathbf{z} := \mathbf{y} + c\mathbf{s}, c, \mathbf{w} = \mathbf{A}\mathbf{y})$, with the challenge c being computed as in the Fiat-Shamir paradigm (that is, computed using a hash function evaluated on some value depending on m and $\mathbf{w}$). Contrary to classical Schnorr signatures, the vector $\mathbf{z}$ is required to be small in order to make the underlying lattice computational problem hard. Verification consists simply in rederiving c from the message m and vector $\mathbf{w}$, checking the size constraint on $\mathbf{z}$ and verifying the algebraic equality $\mathbf{A}\mathbf{z} = \mathbf{w} + c\mathbf{t}$. The actual ML-DSA signing algorithm (see Appendix B, Fig. 5), applies some optimizations and encodings, which make it slightly differ from the above description. For instance, the matrix $\hat{\mathbf{A}}$ (the NTT representation of $\mathbf{A}$), is derived from a seed ρ , the message representative is $\mu = H(\text{BytesToBits}(tr) || M', 64)$, where $tr = H(pk, 64)$ and M' denotes the formatted message, the challenge is sampled through a sampler whose randomness is derived from μ and $\mathbf{w}$, and bits of $\mathbf{w}$ are dropped in the signing process, requiring an additional vector $\mathbf{h}$ (for *hint*) to recover the right $\mathbf{w}$ during verification. The whole verification algorithm is given in Figure 1.

The above differences allow to build a more efficient signature scheme, but introduce complex non-linear operations both in the signing and verification algorithm. In this work, we will focus mostly on the verification process since in our scenario, it is the user’s task to prove knowledge of a valid ML-DSA signature under a *hidden* device public key. Most of the expensive operations are hash computations. In Figure 1, we classify the operations by the number of non-linear operations needed, to underline their cost in the verification process, and complexity considering general-purpose proof systems. The functions `ExpandA` and `SampleInBall` require a large number of hash computations, making them the most costly operations. One surprising fact is that the computation $\mathbf{w} = \mathbf{A}\mathbf{z} - c\mathbf{t}$ (in the NTT domain) is marked as a medium-complexity operation, since at first glance, it is a linear operation. In a classical execution of the verification algorithm, this is indeed true, but in our context the user has to hide both the public key $(\mathbf{A}, \mathbf{t})$ and the vector $\mathbf{z}$, making the overall equation quadratic with respect to the witnesses.

Our approach allows us to avoid all the most expensive operations, while getting rid of most of the other ones. The overall idea is that while the user has to prove knowledge of a valid ML-DSA signature, it does not have to do this via direct application of the verification algorithm. Of course, the proof has to ensure that *if* the signature was run inside the verification algorithm, it would accept it, but it is important to notice that most of the computations performed

ML-DSA.Verify_internal(pk, M', σ)

```

1 :  $(\rho, \mathbf{t}_1) \leftarrow \text{pkDecode}(pk)$ 
2 :  $(\tilde{c}, \mathbf{z}, \mathbf{h}) \leftarrow \text{sigDecode}(\sigma)$ 
3 : if  $\mathbf{h} = \perp$  then return false
4 :  $\hat{\mathbf{A}} \leftarrow \text{ExpandA}(\rho)$ 
5 :  $tr \leftarrow H(pk, 64)$ 
6 :  $\mu \leftarrow H(\text{BytesToBits}(tr) || M', 64)$ 
7 :  $c \in R_q \leftarrow \text{SampleInBall}(\tilde{c})$ 
8 :  $\mathbf{w}'_{\text{Approx}} \leftarrow \text{NTT}^{-1}(\hat{\mathbf{A}} \circ \text{NTT}(\mathbf{z}) - \text{NTT}(c) \circ \text{NTT}(\mathbf{t}_1 \cdot 2^d))$ 
9 :  $\mathbf{w}'_1 \leftarrow \text{UseHint}(\mathbf{h}, \mathbf{w}'_{\text{Approx}})$ 
10 :  $\tilde{c}' \leftarrow H(\mu || \text{w1Encode}(\mathbf{w}'_1), \lambda/4)$ 
11 : return  $(\|\mathbf{z}\|_\infty < \gamma_1 - \beta) \wedge (\tilde{c} = \tilde{c}')$ 

```

Figure 1: ML-DSA verification algorithm (FIPS 204 [62]), we use color to highlight the complexity of the operations with respect to proving zero-knowledge execution of the circuit. The color blue is used to denote ‘cheap’ operations, yellow for medium-complexity operations and orange for expensive ones.

inside the verification algorithm are not essential for the signature, since they are a consequence of optimizations. As a simple illustration of this, consider the heavy `ExpandA`, that recomputes $\mathbf{A}$ from a seed ρ . The computation can be fully avoided if $\mathbf{A}$ is given as an input, and since $\mathbf{A}$ will be hidden from the final verifier anyway, this simple observation allows to reduce the proving cost.

Moreover, the fact that $\mathbf{A}$ is computed from a seed ρ will be ensured through the credential system, since the identity issuer will have to validate the public key and the user will be given access to the seed ρ as part of the ML-DSA signing process. We show that following this thread of ideas allows to keep only three medium-complexity operations:

- the algebraic computation $\mathbf{w} = \mathbf{A}\mathbf{z} - c\mathbf{t}$,
- the norm bound $\|\mathbf{z}\|_\infty < (\gamma_1 - \beta)$ for FIPS.
- and the computation $\mu \leftarrow H(\text{BytesToBits}(tr) || M', 64)$.

Recall that our goal is to prove knowledge of a signature that verifies for a hidden/committed device public key whose encoding is an attribute of the user’s credential, and where the challenge message is chosen by the verifier. This is the standard way to prove non-transferability via device binding. However, we propose a different approach.

Our key insight (quite evident in retrospect) is that, instead of letting the verifier control the full message m , it suffices to ensure that their contribution to m leads to a uniformly random signed message. Note that if the verifier fully controls m , for a fixed pk , the corresponding formatted message M' and hence μ are uniquely determined by pk and m . Since μ is used during verification, a malicious verifier could simply try candidate public keys to de-anonymize the user whenever given a presentation. To avoid the naive solution used in Longfellow (*i.e.*, take everything as a witness), we propose to sign a message with contributions from both the user and the verifier. Upon receiving a ctx including a

nonce in the length of the security parameter n , say 128 bits, the user samples a random value R of n bits and pads it with the verifier's ctx to form the message $M = R||\text{ctx}$. The user then computes $M' \leftarrow \text{BytesToBits}(0x00||0x00)||M$, $tr \leftarrow H(\text{dpk}, 64)$ and $\mu \leftarrow H(\text{BytesToBits}(tr||M', 64))$ exactly as done in the standard and sends R to the SE. Since μ is computed deterministically, the signature $(\tilde{c}, \mathbf{z}, \mathbf{h})$ output by the SE uses a consistent value, which the user can verify using the public key dpk and M' . Once the signature is verified, the user proceeds to prove in zero-knowledge that (1) it knows a valid signature for μ , and (2) μ is correctly computed w.r.t. the ctx.

To that end, it first decodes the public key, computes $\mathbf{w}_1$ as done in the standard using the hint vector. Subsequently, it proves knowledge of a signature such that the encoded device public key is contained in the non-disclosed attributes of the user's credential presentation (see Section 2 for a detailed presentation of the anonymous credential framework adopted in this work). By separating the computation of μ from the algebraic check used to verify the signature, we can split the statement to be proven. Furthermore, the resulting device-binding relation can either be proved monolithically or further decomposed into two sub-relations linked by a bridge commitment C_w on $\mathbf{w}$ to improve efficiency. Concretely, by making $\mathbf{w}_1$ public the verifier can check $\tilde{c}' \leftarrow H(\mu || \text{w1Encode}(\mathbf{w}_1), \lambda/4)$ by herself, and both proofs are bridged by C_w . We formalize this decomposition in Section 3 and discuss its security under the weak MLWR assumption in Appendix C. Crucially, the user binds the ctx and μ used to the selective disclosure proof to finalize the presentation. This way we can keep things modular and compatible with credential schemes that implement a basic scheme providing selective disclosure of attributes.

2. Preliminaries

We use n to denote the security parameter, H for a hash function and $[l]$ for the set $\{1, 2, \dots, l\}$. We use negl to denote a negligible probability in the security parameter.

2.1. Anonymous Credentials with Device Binding

We first adopt and present the *informal* framework from [30] as it nicely presents a unified and modular syntax to describe the core functionalities of an anonymous credential scheme in terms of a signature scheme, a commitment scheme and a non-interactive zero-knowledge proof system. We stress that the goal of this work is to present a *general* device binding protocol. The concrete anonymous credential scheme used on top of it might require extra modules to provide revocation, auditability or other advanced features and security guarantees.

At a minimum, a credential showing consists of proving knowledge of a signature on a commitment to the user's attributes, allowing for a selective disclosure of attributes and to prove possession of a device public key.

Definition 1 ((Multi-message) Signature Scheme [30]). A signature scheme Sig is a tuple of algorithms:

$\text{Sig.Setup}(\text{pp}) \rightarrow \text{pp}_{\text{Sig}}$: probabilistically generates parameters pp_{Sig} from pp .

$\text{Sig.KeyGen}(\text{pp}_{\text{Sig}}) \rightarrow (sk, pk)$: probabilistically generates a key pair (sk, pk) .

$\text{Sig.Sign}(sk, m) \rightarrow \sigma$: a probabilistic algorithm that takes as input a secret key sk and a message $m \in \mathcal{M}$ and outputs a signature σ .

$\text{Sig.Vf}(pk, m, \sigma) \rightarrow \{0, 1\}$: a deterministic algorithm that takes as input a public key pk , a message m and a signature σ and outputs a bit.

The term multi-message signature scheme (MSig) is reserved for schemes whose message space is a vector of ℓ elements, $\mathcal{M}^\ell$. Security-wise, signature schemes are required to provide correctness and unforgeability.

Definition 2 (Commitment Scheme [30]). A commitment scheme with canonical opening Com is defined by:

$\text{Com.Setup}(\text{pp}) \rightarrow \text{pp}_{\text{Com}}$: probabilistically generates parameters pp_{Com} from pp .

$\text{Com.Commit}(\text{pp}_{\text{Com}}, m; \rho) \rightarrow C$: probabilistically outputs a commitment C from parameters pp_{Com} and a message m using explicit randomness ρ .

Verification is canonical: to verify a commitment C to a message m given randomness ρ , the verifier asserts that $C \stackrel{?}{=} \text{Com.Commit}(\text{pp}_{\text{Com}}, m; \rho)$.

Security-wise we require commitments to be binding (they can only be opened to a unique message), and hiding (no information about the underlying message is revealed).

Next, we adopt the definitions from [63], [64], [65] for non-interactive proof systems in the RO model where parties have oracle access to a random oracle H , and require them to satisfy completeness, parameter indistinguishability, knowledge soundness (i.e., a convincing proof implies knowledge of a valid witness w) and zero-knowledge.

Definition 3 (Parameter Indistinguishability). A scheme Π has parameter indistinguishability if for any PPT adversary $\mathcal{A}$ and public parameters pp , the following condition holds:

$$\text{Adv}_{\Pi}^{\text{ParInd}}(\mathcal{A}) := \left| \Pr[\mathcal{A}(\text{pp}_{\Pi}) = 1 : \text{pp}_{\Pi} \leftarrow \Pi.\text{Setup}(\text{pp})] - \Pr[\mathcal{A}(\text{pp}_{\Pi}^*) = 1 : (\text{pp}_{\Pi}^*, \mathcal{T}) \leftarrow \Pi.\text{SimSetup}(\text{pp})] \right| \leq \text{negl}$$

Definition 4 (Non-Interactive Zero-Knowledge Proof). A Non-Interactive Zero-Knowledge (NIZK) proof system for an NP relation $\mathcal{R}$ is a tuple of efficient algorithms:

$\text{NIZK.Setup}(\text{pp}) \rightarrow \text{pp}_{\mathcal{R}}$: on input the global parameters pp , outputs parameters $\text{pp}_{\mathcal{R}}$.

$\text{NIZK.Prove}(\text{pp}_{\mathcal{R}}, x, w) \rightarrow \pi$: on input $\text{pp}_{\mathcal{R}}$ and a statement/witness pair $(x, w) \in \mathcal{R}$, outputs a proof π .

$\text{NIZK.Vrfy}(\text{pp}_{\mathcal{R}}, x, \pi) \rightarrow \{0, 1\}$: on input $\text{pp}_{\mathcal{R}}$, the statement x and a proof π , outputs a bit.

$\text{NIZK.SimSetup}(\text{pp}) \rightarrow (\text{pp}_{\mathcal{R}}^*, \mathcal{T})$: on input $\text{pp}_{\mathcal{R}}$, outputs simulated parameters $\text{pp}_{\mathcal{R}}^*$ and a trapdoor $\mathcal{T}$.

Completeness: for every $(x, w) \in \mathcal{R}$ and every pp , $\text{NIZK.Vrfy}(\text{pp}_{\mathcal{R}}, x, \pi)$ accepts with probability 1, over the choice of $\text{pp}_{\mathcal{R}} \leftarrow \text{NIZK.Setup}(\text{pp})$ and $\pi \leftarrow \text{NIZK.Prove}(\text{pp}_{\mathcal{R}}, x, w)$.

Parameter Indistinguishability: as in Definition 3.

Knowledge-Soundness: for every PPT algorithm Prove^* there exists a PPT algorithm Ext such that the following probability is negligible

$$\Pr \left[1 \leftarrow \text{NIZK.Vrfy}(\text{pp}_{\mathcal{R}}, x, \pi) \wedge (x, w) \notin \mathcal{R} \mid \begin{array}{l} \text{pp}_{\mathcal{R}} \leftarrow \text{NIZK.Setup}(\text{pp}); \rho \leftarrow \{0, 1\}^{\text{poly}(n)}; \\ (x, \pi) \leftarrow \text{Prove}^*(\text{pp}_{\mathcal{R}}; \rho); w \leftarrow \text{Ext}(\text{pp}_{\mathcal{R}}; \rho) \end{array} \right]$$

Zero-Knowledge: there exists a PPT algorithm SimProve such that for any PPT adversary $\mathcal{A}$, we have that $\text{Adv}_{\text{NIZK}}^{\text{zk}}(\mathcal{A}) := \left| \Pr[\text{Exp}_{\text{NIZK}, \mathcal{A}}^{\text{real}}(\text{pp}) = 1] - \Pr[\text{Exp}_{\text{NIZK}, \mathcal{A}}^{\text{sim}}(\text{pp}) = 1] \right| \leq \text{negl}$. Both experiments first sample $(\text{pp}_{\Pi}^*, \mathcal{T}) \leftarrow \text{SimSetup}(\text{pp})$ and run $b' \leftarrow \mathcal{A}^{\mathcal{O}}(\text{pp}_{\Pi}^*)$, giving $\mathcal{A}$ oracle access to a proving oracle $\mathcal{O}$ and outputting the bit b' . On a query $(x, w) \in \mathcal{R}$, the oracle $\mathcal{O}$ answers with $\text{Prove}(\text{pp}_{\Pi}^*, x, w)$ in $\text{Exp}_{\text{NIZK}, \mathcal{A}}^{\text{real}}$ and with $\text{SimProve}(\text{pp}_{\Pi}^*, x, \mathcal{T})$ in $\text{Exp}_{\text{NIZK}, \mathcal{A}}^{\text{sim}}$ (discarding w), and with $\perp$ if $(x, w) \notin \mathcal{R}$.

We further make use of NIZK proof systems providing a strong form of *knowledge soundness* known as (*multi-proof*) *straight-line extractability*. For ease of exposition, we only present the (classical) definition by Katsumata and Del Pino [64] but stress that our work is compatible with constructions proven secure in the QROM.

Definition 5 (Multi-proof straight-line extractability [64]). *An NIZK proof system is multi-proof straight-line extractable, if there exists a PPT extractor Ext with the following property:*

Straight-line extractability: *there exist constants c, e_1, e_2 and a polynomial $p(\text{pp})$ such that for any $Q_H = \text{poly}(\text{pp})$ and any PPT adversary $\mathcal{A}$ making at most Q_H random-oracle queries with*

$$\Pr \left[\begin{array}{l} (\widetilde{\text{pp}}_{\mathcal{R}}, \mathcal{T}) \leftarrow \text{NIZK.SimSetup}(\text{pp}), \\ \{(x_i, \pi_i)\}_{i \in [Q_s]} \leftarrow \mathcal{A}(\widetilde{\text{pp}}_{\mathcal{R}}) : \\ \forall i \in [Q_s], \text{NIZK.Vrfy}(\widetilde{\text{pp}}_{\mathcal{R}}, x_i, \pi_i) = 1 \end{array} \right] \geq \mu(\text{pp}),$$

we have

$$\Pr \left[\begin{array}{l} (\text{pp}_{\Pi}^*, \mathcal{T}) \leftarrow \text{NIZK.SimSetup}(\text{pp}), \\ \{(x_i, \pi_i)\}_{i \in [Q_s]} \leftarrow \mathcal{A}(\text{pp}_{\Pi}^*), \\ \{w_i \leftarrow \text{NIZK.Ext}(Q_H, Q_s, 1/\mu, \text{pp}_{\Pi}^*, \mathcal{T}, x_i, \pi_i)\}_{i \in [Q_s]} : \\ \forall i \in [Q_s], (x_i, w_i) \in \mathcal{R} \\ \wedge \text{NIZK.Vrfy}(\text{pp}_{\Pi}^*, x_i, \pi_i) = 1 \\ \geq \frac{1}{2} \cdot \mu(\text{pp}) - \text{negl}(\text{pp}). \end{array} \right]$$

where Ext runs in time at most $Q_H^{e_1} \cdot Q_s^{e_2} \cdot \mu^{-c} \cdot p(\text{pp})$.

The above definition considers exact polynomial-time extraction. We also note that a proof system can be straight-line extractable without being multi-proof extractable: the

single-proof variant (see [66], Definition 2.5) extracts from one proof without rewinding but offers no guarantee under many adaptively-chosen proofs. In the remainder, unless otherwise stated, we use straight-line extractability to refer to the multi-proof notion of Definition 5. This is the extraction notion we require from the NIZK proof systems composing a credential showing protocol (cf. Section 4).

Whenever proofs are bound to some context ctx we write $\text{Prove}(\text{pp}_{\mathcal{R}}, x, w)[\text{ctx}]$ and $\text{Verify}(\text{pp}_{\mathcal{R}}, x, \pi)[\text{ctx}]$. We also use the Camenisch–Stadler notation [67] to succinctly present the relation to be proven for a NIZK with the following pattern:

$$\underbrace{\mathcal{R}_{\text{name}}}_{\text{relation name}} \left\{ \underbrace{(w)}_{\text{witness}} : \underbrace{(x, w) \in \mathcal{R}_{\text{name}}}_{\text{statement}} \right\}$$

where any variable in the statement that is not part of the witness is assumed to be known to the verifier.

Anonymous credential systems involve three entities: *issuers*, *users*, and *verifiers*. Issuers authenticate users and grant them credentials attesting to their attributes. Whenever a user wants to authenticate towards a verifier, they derive a presentation from the credential. Presentations should allow for granularity in the amount of information shown, which is generally achieved via a selective disclosure feature. Furthermore, different presentations originating from the same user should not be traced back to them (unlinkability). This should hold even when issuers and verifiers collude. Finally, credentials and presentations should be unforgeable. The latter notion captures that users should not be able to collude with others to prove statements beyond what is true with respect to the attributes they hold, while the former captures that only issuers can create valid credentials.

To capture selective disclosure, [30] considers the following relation in terms of MSig and Com with $S_{\text{Rev}}, S_{\text{Com}} \subseteq [\ell]$ being two disjoint subsets corresponding to the revealed and committed attributes, respectively.

$$\mathcal{R}_{\text{CD}} = \left\{ \begin{array}{l} (\mathbf{a} \in \mathcal{M}^{\ell}, \text{cred}, \{\rho_i\}_{i \in S_{\text{Com}}}) : \\ \text{MSig.Vf}(\text{ipk}, \mathbf{a}, \text{cred}) \\ \wedge (\forall i \in S_{\text{Rev}} : a_{\text{Rev}, i} = \mathbf{a}[i]) \wedge (\forall i \in S_{\text{Com}} : \\ C_i = \text{Com.Commit}(\text{pp}_{\text{Com}}, \mathbf{a}[i]; \rho_i)) \end{array} \right\}$$

The above relation captures (1) possession of a credential issued under the issuer public key ipk , and (2) selective disclosure of a subset of the attested attributes, while binding the remaining hidden attributes to fresh commitments that can be reused by other modules.

For device binding, we only assume black-box access to the Secure Element via a minimal API that can be queried to generate a key pair providing the corresponding device public key and signatures for any message of the user's preference on said public key.

The first step to bind a credential presentation to a key pair stored in the Secure Element consists of encoding the device public key as an attribute (or set of attributes) that are included in the user's credential during issuance. Without loss of generality we also assume the Secure Element to provide an attestation mechanism, allowing the issuer to

verify that the proposed device public key is a legitimate public key originating from an accepted Secure Element at issuance time.⁴

Upon credential issuance, the second step is executed on every presentation that requires device binding and consists of proving knowledge of a signature on a ctx under the committed encoding of the device public key. [30] captures this via the following relation where $\text{Encode}(\text{dpk}) \rightarrow \mathbf{dpk} \in \mathcal{M}^t$ is an efficient algorithm that encodes dpk as a vector of t attribute-space elements.

$$\mathcal{R}_{\text{DB}} = \left\{ \begin{array}{l} (\{ \text{dpk}_i, \rho_i \}_{i \in [t]}, \sigma) : \\ \text{Sig.Vf}(\text{dpk}, \text{nonce}, \sigma) \wedge \text{Encode}(\text{dpk}) = \mathbf{dpk} \wedge \\ \forall i \in [t] : C_i = \text{Com.Commit}(\text{pp}_{\text{Com}}, \text{dpk}_i; \rho_i) \end{array} \right\}$$

The above building blocks and relations are already sufficient to introduce a minimal syntax for anonymous credentials with selective disclosure and device binding. However, to formally prove our device-binding protocol we now incorporate the *formal* model and syntax for Device-Bound Anonymous Credentials (DBAC) from [13], which captures the specificities of device binding in a more explicit way. Again, we stress that the AC scheme used might incorporate more relations and algorithms to capture other functionalities through different modules.

Compared to [13], we replace the non-interactive issuance algorithm with an interactive protocol between the issuer and the user. This allows the user to participate in credential generation (e.g., by contributing commitments or zero-knowledge proofs) and in particular enables hiding some attributes from the issuer during issuance. Second, we make the two-move nature of the showing protocol towards the verifier explicit: the verifier initiates the interaction by sending a context ctx computed conditioned on a nonce, and the user responds with a presentation τ . For simplicity, we consider showings with respect to a single predicate ϕ for selective disclosure defined by $\mathcal{R}_{\text{CD}}$ and $\mathcal{R}_{\text{DB}}$.

Definition 6 (Device-Bound Anonymous Credential Scheme (adapted from [13], [30])). *A (context-aware) Device-Bound Anonymous Credential scheme (DBAC) with selective disclosure, defined with respect to a predicate ϕ and message space $\mathcal{M}$, is a tuple of algorithms and protocols (Setup, IssKGen, DevKGen, Issuelss, IssueUser, ShowUssr, ShowSE, ShowVf), parameterized by relations $\mathcal{R}_{\text{CD}}$ and $\mathcal{R}_{\text{DB}}$, with the following syntax:*

$\text{Setup}(1^n, \ell) \rightarrow \text{par}$: On input the security parameter and the number of attributes ℓ , outputs the public parameters par, specializing the underlying primitives MSig, Com, NIZK_{CD}, and NIZK_{DB}.

$\text{IssKGen}(\text{par}) \rightarrow (\text{isk}, \text{ipk})$: Generates the issuer's key pair.

$\text{DevKGen}(\text{par}) \rightarrow (\text{dsk}, \text{dpk})$: Generates the SE's key pair.

$\langle \text{Issuelss}(\text{isk}, \text{ipk}, \text{dpk}, \text{par}, \mathbf{a}) \Rightarrow \text{IssueUser}(\text{dpk}, \text{ipk}, \text{par}, \mathbf{a}) \rangle \rightarrow (\text{st}, \text{cred})/\perp$: An interactive protocol between the issuer Iss holding $(\text{isk}, \text{ipk}, \text{dpk}, \text{par}, \mathbf{a})$ and the user User holding $(\text{dpk}, \text{ipk}, \text{par}, \mathbf{a})$. If the protocol succeeds,

the user obtains a credential cred on the attribute vector $\mathbf{a} \in \mathcal{M}^\ell$ bound to device key dpk, and the issuer updates its state st. The issuer only learns whether the execution was successful.

$\text{VfCred}(\text{ipk}, \text{cred}, \text{dpk}, \mathbf{a}) \rightarrow 01$: Run by the user to verify that a credential is valid w.r.t. dpk and set of attributes $\mathbf{a}$.
 $\langle \text{ShowVf}(\text{ipk}, \phi) \Rightarrow \text{ShowUser}(\text{ipk}, \text{dpk}, \text{cred}, \mathbf{a}, \phi) \rangle \rightarrow (\perp, 0/1)$: An interactive protocol between the user and the verifier to present a device-bound credential with selective disclosure. The predicate ϕ specifies the indices $S_{\text{Rev}}, S_{\text{Com}} \subseteq [\ell]$ of revealed and committed attributes, the indices $S_{\text{DB}} \subseteq S_{\text{Com}}$ involved in the device binding module, and (optionally) additional auxiliary information. Internally, the user interacts with the SE to compute the presentation. For schemes that are round-optimal, this can be decomposed into the following protocols:

$\text{ShowVf}_1(\text{ipk}, \phi) \rightarrow (\text{vst}, \text{ctx})$

$\text{ShowUssr}_1(\text{ipk}, \text{dpk}, \text{cred}, \mathbf{a}, \text{ctx}, \phi) \rightarrow (\text{ust}, \text{umsg})$

$\text{ShowSE}_1(\text{dsk}, \text{ipk}, \text{umsg}, \text{ctx}) \rightarrow \text{smsg}$

$\text{ShowUssr}_2(\text{ust}, \text{smsg}) \rightarrow \tau$, where the presentation has the form $\tau = (\{C_i\}_{i \in S_{\text{Com}}}, \pi_{\text{CD}}, \pi_{\text{DB}})$

$\text{ShowVf}_2(\text{vst}, \text{ipk}, \tau, \text{ctx}, \phi) \rightarrow \{0, 1\}$: verifies the two NIZK proofs π_{CD} and π_{DB} on the commitments $\{C_i\}_{i \in S_{\text{Com}}}$ in τ and outputs a bit.

Where ctx typically encodes the service context ctx_0 and nonce is a fresh session nonce. We sometimes overload the term ctx to mean $(\text{nonce}, \text{ctx}_0, \text{ctx})$. The user obtains no output; the verifier learns only whether $\phi(\mathbf{a}) = \text{true}$.
 $\text{Verify}(\text{ipk}, \tau, \text{ctx}, \phi) \rightarrow 01$: Syntactic sugar for the verifier-side execution of the showing protocol, where the validity of ctx is checked internally.

Security Model. Security is proven via two independent games that show the two properties:

Unforgeability Attributes $\mathbf{a}$ are unforgeable, presentation τ is fresh, device binding is non-transferable.

Unlinkability Presentation τ does not reveal anything to the verifier, except that $\phi(\mathbf{a}) = \text{true}$.

We only consider the strongest notion for unforgeability corresponding to a corrupted user and secure element [13, Sec. 4, “ $T_1 \cup T_2 \cup T_3$ ”], and unlinkability “Unlink-1”, which holds for an honest secure element [13, Sec. 5.1].

2.1.1. Unforgeability. The unforgeability game covers attribute unforgeability, presentation freshness, and device-binding non-transferability. The issuer is honest, verifiers are corrupt, and users and SEs may be either corrupt or honest. The adversary $\mathcal{A}$ has access to oracles for creating SEs, issuing credentials, requesting presentations from honest users/SEs, and corrupting SEs. Eventually, $\mathcal{A}$ outputs a forgery $(\tau^*, \text{ctx}^*, \phi^*)$. The unforgeability game requires dpk^* to be extractable from the representation τ^* output by the adversary. More precisely, it requires the existence of an efficient extractor $\text{Ext} = (\text{Ext.Setup}, \text{Ext.EKey})$, where parameters output by Ext.Setup are indistinguishable from those output by DBAC.Setup are embedded in the game.

Oracles and Records. $\mathcal{O}_{\text{NewSE}}$ lets the adversary create an honest SE (which can later be corrupted via $\mathcal{O}_{\text{CorruptSE}}$)

4. For example, a certificate chain rooted at the hardware manufacturer is used to verify a challenge for the device public key.

Table 2: Records kept in the UNF game [13, Table 2].

Record	Tuples	Explanation
SEKeys	$\{(dsk, dpk)\}$	All device key pairs generated by the game
CSEs	$\{dpk\}$	Corrupt SEs
HUsers	$\{dpk\}$	Device keys of honest users
CCreds	$\{(dpk, a)\}$	Attributes issued to corrupt users
HCreds	$\{(dpk, a, cred)\}$	Credentials issued to honest users
HShown	$\{(dpk, ctx, \phi)\}$	Presentations queried for honest users and SEs
Granted	$\{(dpk, ctx)\}$	Contexts queried to honest SEs by corrupt users

and set the associated user to honest or corrupt. Honest users' keys are stored in HUsers. $\mathcal{O}_{\text{Issue}}$ issues credentials to corrupt users; $\mathcal{O}_{\text{hIssue}}$ creates (but does not output) credentials for honest users. $\mathcal{O}_{\text{hShow}}$ lets the adversary request presentations from honest users with honest SEs. $\mathcal{O}_{\text{ShowSE}}$ lets a corrupt user invoke the honest SE for its device-bound credentials. Table 2 summarizes the records maintained.

Winning Conditions. $\mathcal{A}$ wins by computing a non-trivial presentation. Given the extracted dpk^* , context ctx^* , and predicate ϕ^* , trivial presentations are classified into three types depending on the corruption setting of SE and user:

T_1 (**Corrupt User, Corrupt SE**): If dpk^* is corrupt and associated to a corrupt user, a trivial presentation is of the form: $T_1 = \{(dpk, ctx, \phi) \mid \exists a : (dpk, a) \in CCreds \wedge dpk \in CSEs \wedge \phi(a) = 1 \wedge ctx \in \{0, 1\}^*\}$. Thus, in this setting, the adversary wins only by producing a presentation for a predicate ϕ^* not satisfied by any attributes issued to dpk^* (reflected in CCreds).

T_2 (**Corrupt User, Honest SE**): If the user is corrupt but uses an honest SE, a trivial presentation has the following form: $T_2 = \{(dpk, ctx, \phi) \mid \exists a : (dpk, a) \in CCreds \wedge dpk \notin CSEs \wedge (dpk, ctx) \in Granted \wedge \phi(a) = 1\}$. Consequently, the adversary additionally wins if ctx^* was never queried to the SE (reflected in Granted).

T_3 (**Honest User, Honest SE**): If both are honest, a trivial presentation is of the form: $T_3 = \{(dpk, ctx, \phi) : (dpk, ctx, \phi) \in HShown\}$. This means that the adversary wins if the presentation is for a context and predicate never queried to the honest user/SE (reflected in HShown).

Definition 7 (Unforgeability [13, Def. 3]). A DBAC scheme Π is unforgeable if there exists an extractor $\text{Ext} = (\text{Setup}, \text{EKey})$ such that for all PPT adversaries $\mathcal{A}$ and $\ell \in \mathbb{N}$ the following conditions hold:

- 1) The distributions of par from $\Pi.\text{Setup}$ and $\text{Ext}.\text{Setup}$ are computationally indistinguishable.
- 2) The following advantage is negligible:

$$\text{Adv}_{\Pi, \ell, \text{Ext}}^{\text{Unf}}(\mathcal{A}, n) := \Pr[\text{Unf}_{\Pi, \ell, \text{Ext}}^{\mathcal{A}}(n) = 1]$$

where $\text{Unf}_{\Pi, \ell, \text{Ext}}^{\mathcal{A}}(n)$ is the game defined in Figure 2.

2.1.2. Unlinkability (Unlink-1). This notion provides privacy guarantees in a setting where the adversary is assumed to have oracle access to an *honest* SE. In particular, it ensures that a presentation τ reveals no information about the

Game $\text{Unf}_{\text{DBAC}, \ell, \text{Ext}}^{\mathcal{A}}(n)$
$1 : (\mathcal{T}, \text{par}) \leftarrow \text{Ext}.\text{Setup}(1^n, \ell)$ $2 : \text{CSEs}, \text{HUsers}, \text{CCreds}, \text{Granted} \leftarrow \emptyset$ $3 : \text{SEKeys}, \text{HCreds} \leftarrow \emptyset$ $4 : (\text{isk}, \text{ipk}) \leftarrow \text{IssKGen}(\text{par})$ $5 : (\tau^*, \text{ctx}^*, \phi^*) \leftarrow \mathcal{A}^{\mathcal{O}_*}(\text{par}, \text{ipk})$ $6 : dpk^* \leftarrow \text{Ext}.\text{EKey}(\mathcal{T}, \text{ipk}, \tau^*, \text{ctx}^*, \phi^*)$ $7 : \text{if } \text{Verify}(\text{ipk}, \tau^*, \text{ctx}^*, \phi^*) \neq 1 \text{ return } 0$ $8 : T \leftarrow T_1 \cup T_2 \cup T_3$ $9 : \text{if } (dpk^*, \text{ctx}^*, \phi^*) \notin T \text{ return } 1 \text{ else } 0$ $\mathcal{O}_{\text{NewSE}}(\rho_{\text{Unf}}, \text{isHonestUser})$
$\text{if } \rho_{\text{Unf}} = \perp \text{ then } (dsk, dpk) \leftarrow \text{DevKGen}(\text{par})$ else $\quad (dsk, dpk) \leftarrow \text{DevKGen}(\text{par}; \rho_{\text{Unf}}); \text{CSEs} \leftarrow \text{CSEs} \cup \{dpk\}$ $\text{if } dpk \in \text{SEKeys} \text{ return } \perp; \text{SEKeys}[dpk] \leftarrow dsk$ $\text{if } \text{isHonestUser} = 1: \text{HUsers} \leftarrow \text{HUsers} \cup \{dpk\}$ $\text{return } dpk$ $\mathcal{O}_{\text{Issue}}(dpk, a)$
$\text{if } dpk \in \text{HUsers} \text{ return } \perp; \text{cred} \leftarrow \text{Issue}(\text{isk}, dpk, a)$ $\text{CCreds} \leftarrow \text{CCreds} \cup \{(dpk, a)\}; \text{return cred}$ $\mathcal{O}_{\text{ShowSE}}(dpk, ctx, \text{umsg})$
$\text{msg} \leftarrow \text{ShowSE}_1(\text{SEKeys}[dpk], \text{ipk}, \text{umsg}, \text{ctx})$ $\text{Granted} \leftarrow \text{Granted} \cup \{(dpk, ctx)\}; \text{return msg}$ $\mathcal{O}_{\text{hIssue}}(\text{cid}, dpk, a)$
$\text{if } dpk \notin \text{SEKeys} \vee \text{cid} \in \text{HCreds} \text{ return } \perp$ $\text{cred} \leftarrow \text{Issue}(\text{isk}, dpk, a)$ $\text{HCreds}[\text{cid}] \leftarrow (dpk, a, \text{cred}); \text{return } \perp$ $\mathcal{O}_{\text{hShow}}(\text{cid}, ctx, \phi)$
$\text{if } \text{cid} \notin \text{HCreds} \text{ return } \perp; (dpk, a, \text{cred}) \leftarrow \text{HCreds}[\text{cid}]$ $\text{if } dpk \notin \text{HUsers} \vee dpk \in \text{CSEs} \text{ return } \perp$ $dsk \leftarrow \text{SEKeys}[dpk]$ $(\tau, \perp) \leftarrow (\text{ShowUser}(\text{ipk}, dpk, \text{cred}, \text{ctx}, \phi)$ $\quad \quad \quad \Rightarrow \text{ShowSE}(dsk, \text{ipk}, \text{ctx}))$ $\text{HShown} \leftarrow \text{HShown} \cup \{(dpk, \text{ctx}, \phi)\}; \text{return } \tau$ $\mathcal{O}_{\text{CorruptSE}}(dpk)$
$\text{if } dpk \notin \text{SEKeys} \text{ return } \perp; \text{CSEs} \leftarrow \text{CSEs} \cup \{dpk\}$ $\text{return } \text{SEKeys}[dpk]$

Figure 2: Unforgeability game for context-aware DBAC [13, Fig. 1]. The extractor $\text{Ext}.\text{EKey}$ also gets access to the entire game state (e.g., isk , SEKeys , all signatures output by Issue) and can program random oracles. $\mathcal{O}_* = \{\mathcal{O}_{\text{NewSE}}, \mathcal{O}_{\text{Issue}}, \mathcal{O}_{\text{ShowSE}}, \mathcal{O}_{\text{hIssue}}, \mathcal{O}_{\text{hShow}}, \mathcal{O}_{\text{CorruptSE}}\}$.

user beyond the fact that the predicate holds. Furthermore, even if the same user generates multiple presentations, those presentations cannot be linked to one another.

The formal definition relies on a simulation-based approach: it requires the existence of a simulator $\text{Sim} = (\text{Sim}.\text{Setup}, \text{Sim}.\text{Show})$, where parameters generated by $\text{Sim}.\text{Setup}$ are indistinguishable from those generated by

5. We stress that `ctx` denotes our protocol context, while the ML-DSA context argument is fixed to the empty string in this work.

alongside the correct computation of UseHint in zero-knowledge w.r.t. a commitment to dpk and $\mathbf{w}$ ($\mathbf{w}'_{\text{Approx}}$ in 1). We do so by keeping $\mathbf{z}$, $\mathbf{h}$ and $\mathbf{w}$ as a witness, and letting the verifier re-derive $\tilde{c}'$ from public μ and $\mathbf{w}'_1$ and run $c' \leftarrow \text{SampleInBall}(\tilde{c}')$, which is then used as the public challenge in the algebraic proof. This is possible because without seeing $\mathbf{z}$, the vector $\mathbf{w}'_1$ is a weak MLWR instance (see Appendix C), ensuring that it doesn't leak any exploitable information on the key used to sign.

We now observe that any valid signature $(\tilde{c}, \mathbf{z}, \mathbf{h})$ from the SE is key-prefixed at dpk and exploit this fact to let the user simply prove that $\mu = H(R_\mu || \text{ctx}, 64)$. If $R_\mu \neq \text{BytesToBits}(H(\text{dpk}, 64)) || \text{BytesToBits}(0x00 || 0x00) || R$, then the user cannot produce a valid proof without forging the inner signing relation for dpk . Similarly, if R isn't chosen properly, then the user will not be able to produce unlinkable proofs. Alternatively, one could have the Issuer attest the encoding of dpk as $(tr = H(\text{dpk}, 64), \mathbf{A}, \mathbf{t}_1)$ so that tr also becomes part of the witness. In Section 4.4 we further elaborate on these trade-offs.

To glue both proofs together, we require a commitment scheme that is compatible with the underlying AC scheme and to bind the μ used to the showing proofs and the ctx . This avoids replay attacks where the user could obtain a single signature and try to re-use it across different presentations. The resulting $\mathcal{R}_{\text{DB}}$ relation is defined as:

$$\mathcal{R}_{\text{DB}} = \left\{ \begin{array}{l} (\mathbf{z}, \mathbf{w}, \mathbf{h}, \text{dpk}, o_{i \in S_{\text{DB}}}, R_\mu) : \\ \mu = H(R_\mu || \text{ctx}) \\ \wedge \text{Encode}(\text{dpk}) = \{\mathbf{A}_i, \mathbf{t}_{1,i}\}_{i \in S_{\text{DB}}} \wedge \\ \forall i \in S_{\text{DB}} : C_i = \text{Com.Commit}((\mathbf{A}_i, \mathbf{t}_{1,i}); o_i) \\ \wedge \mathbf{A}\mathbf{z} = \mathbf{w} + \mathbf{c}\mathbf{t}_1 \cdot 2^d \wedge \|\mathbf{z}\|_\infty < \gamma_1 - \beta \\ \wedge \mathbf{w}_1 = \text{UseHint}(\mathbf{h}, \mathbf{w}) \wedge \text{hw}(\mathbf{h}) \leq \omega \end{array} \right\}$$

where Encode is a deterministic encoding function that takes dpk as specified in the standard and outputs values $\{\mathbf{A}_i, \mathbf{t}_{1,i}\}_{i \in S_{\text{DB}}}$ compatible with the attribute space. Importantly, the Issuer ensures that the encoding is correct during issuance while the user is required to prove consistency of the equation with the commitment of the encoding.

In Figure 4, we present the showing protocol of our DBAC scheme, which integrates the device binding module with a basic anonymous credential scheme that supports selective disclosure. The showing protocol consists of two rounds, where the user first interacts with the SE to obtain a signature on a message that is derived from the ctx and a random value. Subsequently, the user computes the commitments for all attributes that are not disclosed according to S_{Com} . Note that $S_{\text{DB}} \subseteq S_{\text{Com}}$ by definition as dpk is never revealed. At this point, the user computes each proof separately but with respect to the same context, which includes the ctx (and thus the verifier's nonce), μ and all commitments. Looking ahead, by the straight-line extractability of NIZK_{CD} and NIZK_{DB} , an accepting transcript allows the extractor to recover the openings of the commitments $\{C_i\}_{i \in S_{\text{DB}}}$, which give the same dpk due to the binding property of Com. This guarantees that both proofs are bound to the same session determined by ctx , nonce and μ . Finally,

```

ShowVf1(ipk,  $\phi$ )
nonce  $\leftarrow$   $\$ D_1$ ;  $\text{ctx}_0 \leftarrow D_2$ ;  $\text{ctx} \leftarrow H_{\text{ctx}}(\text{nonce} || \text{ctx}_0 || \text{Encode}(\phi))$ 
vst  $\leftarrow$  (nonce,  $\text{ctx}_0$ ,  $\text{ctx}$ ); return (vst, (nonce,  $\text{ctx}_0$ ,  $\text{ctx}$ ))
ShowUstr1(ipk,  $\text{dpk}$ , cred,  $\mathbf{a}$ , (nonce,  $\text{ctx}_0$ ,  $\text{ctx}$ ),  $\phi$ )
if  $\text{ctx} \neq H_{\text{ctx}}(\text{nonce} || \text{ctx}_0 || \text{Encode}(\phi))$  return  $\perp$ 
 $R \leftarrow \$ D_1$ ;  $M \leftarrow R || \text{ctx}$ ;  $M' \leftarrow \text{BytesToBits}(0x00 || 0x00) || M$ 
6 :  $tr \leftarrow H(\text{dpk}, 64)$ ; 7 :  $\mu \leftarrow H(\text{BytesToBits}(tr) || M', 64)$ 
return (( $\text{dpk}$ , cred,  $\mathbf{a}$ ,  $\text{ctx}$ ,  $\phi$ ,  $R$ ,  $\mu$ ),  $R$ )
ShowSE1(dsk, ipk, umsg,  $\text{ctx}$ )
 $M \leftarrow \text{umsg} || \text{ctx}$ ; return ML-DSA.Sign(dsk,  $M$ ,  $\epsilon$ )
ShowUstr2(ust, smsg)
parse ust as ( $\text{dpk}$ , cred,  $\mathbf{a}$ ,  $\text{ctx}$ ,  $\phi$ ,  $R$ ,  $\mu$ )
parse  $\phi$  as ( $\mathcal{R}_{\text{CD}}$ ,  $\mathcal{R}_{\text{DB}}$ ,  $S_{\text{Rev}}$ ,  $S_{\text{Com}}$ ,  $S_{\text{DB}}$ )
1 : ( $\rho$ ,  $\mathbf{t}_1$ )  $\leftarrow$  pkDecode( $\text{dpk}$ )
2 : ( $\tilde{c}$ ,  $\mathbf{z}$ ,  $\mathbf{h}$ )  $\leftarrow$  sigDecode(smsg)
3 : if  $\mathbf{h} = \perp$  then return false
4 : end if // Assume a valid signature
5 :  $\mathbf{A} \leftarrow \text{ExpandA}(\rho)$ ; 8 :  $c \leftarrow \text{SampleInBall}(\tilde{c})$ 
9 :  $\mathbf{w} \leftarrow \text{NTT}^{-1}(\mathbf{A} \cdot \text{NTT}(\mathbf{z}) - \text{NTT}(c) \cdot \text{NTT}(\mathbf{t}_1 \cdot 2^d))$ 
10 :  $\mathbf{w}_1 \leftarrow \text{UseHint}(\mathbf{h}, \mathbf{w})$ 
 $\forall i \in S_{\text{Com}} : \text{Com.Commit}(a_i; o_i)$ 
 $x_{\text{CD}} \leftarrow (\text{ipk}, S_{\text{Rev}}, S_{\text{Com}}, \{a_i\}_{i \in S_{\text{Rev}}}, \{C_i\}_{i \in S_{\text{Com}}})$ 
 $w_{\text{CD}} \leftarrow (\mathbf{a}, \text{cred}, \{o_i\}_{i \in S_{\text{Com}}})$ 
 $x_{\text{DB}} \leftarrow (S_{\text{DB}}, \{C_i\}_{i \in S_{\text{DB}}}, c, \mathbf{w}_1, \mu, \text{ctx})$ 
 $w_{\text{DB}} \leftarrow (\{a_i, o_i\}_{i \in S_{\text{DB}}}, \mathbf{z}, \mathbf{w}, \mathbf{h}, tr, R)$ 
 $\text{ctx}' \leftarrow \text{ctx} || \mu || \{C_i\}_{i \in S_{\text{Com}}}$ 
 $\pi_{\text{CD}} \leftarrow \text{NIZK}_{\text{CD}}.\text{Prove}(x_{\text{CD}}, w_{\text{CD}})[\text{ctx}']$ 
 $\pi_{\text{DB}} \leftarrow \text{NIZK}_{\text{DB}}.\text{Prove}(x_{\text{DB}}, w_{\text{DB}})[\text{ctx}']$ 
 $\tau \leftarrow (\{C_i\}_{S_{\text{Com}}}, \mathbf{w}_1, \mu, \pi_{\text{CD}}, \pi_{\text{DB}})$ ; return  $\tau$ 
ShowVf2(vst, ipk,  $\phi$ ,  $\{C_i\}_{i \in S_{\text{Com}}}$ ,  $\{a_i\}_{i \in S_{\text{Rev}}}$ ,  $\tau$ ,  $\text{ctx}$ )
parse  $\phi$  as ( $\mathcal{R}_{\text{CD}}$ ,  $\mathcal{R}_{\text{DB}}$ ,  $S_{\text{Rev}}$ ,  $S_{\text{Com}}$ ,  $S_{\text{DB}}$ )
parse  $\tau$  as ( $\{C_i\}_{S_{\text{Com}}}$ ,  $\mathbf{w}_1$ ,  $\mu$ ,  $\pi_{\text{CD}}$ ,  $\pi_{\text{DB}}$ )
parse vst as (nonce,  $\text{ctx}_0$ ,  $\text{ctx}'$ )
if  $\text{ctx}' \neq \text{ctx} \vee \text{ctx} \neq H_{\text{ctx}}(\text{nonce} || \text{ctx}_0 || \text{Encode}(\phi))$  return 0
11 :  $\tilde{c}' \leftarrow H(\mu || \mathbf{w}_1 \text{Encode}(\mathbf{w}_1), \lambda/4)$ ;  $c' \leftarrow \text{SampleInBall}(\tilde{c}')$ 
 $x_{\text{CD}} \leftarrow (\text{ipk}, S_{\text{Rev}}, S_{\text{Com}}, \{a_i\}_{i \in S_{\text{Rev}}}, \{C_i\}_{i \in S_{\text{Com}}})$ 
 $x_{\text{DB}} \leftarrow (S_{\text{DB}}, \{C_i\}_{i \in S_{\text{DB}}}, c', \mathbf{w}_1, \mu, \text{ctx})$ 
 $\text{ctx}' \leftarrow \text{ctx} || \mu || \{C_i\}_{i \in S_{\text{Com}}}$ 
 $b_{\text{CD}} \leftarrow \text{NIZK}_{\text{CD}}.\text{Vf}(\pi_{\text{CD}}, x_{\text{CD}})[\text{ctx}']$ 
 $b_{\text{DB}} \leftarrow \text{NIZK}_{\text{DB}}.\text{Vf}(\pi_{\text{DB}}, x_{\text{DB}})[\text{ctx}']$ 
 $b_S \leftarrow S_{\text{DB}} \stackrel{?}{\subseteq} S_{\text{Com}} \wedge S_{\text{Com}}, S_{\text{Rev}} \stackrel{?}{\subseteq} [\ell] \wedge S_{\text{Com}} \cap S_{\text{Rev}} \stackrel{?}{=} \emptyset$ 
return  $b_{\text{CD}} \wedge b_{\text{DB}} \wedge b_S$ 

```

Figure 4: Our device binding protocol for ML-DSA. Operations coming from ML-DSA's verification algorithm (Figure 1) are highlighted in orange. For these lines, we indicate the matching line number in Figure 1.

the verifier re-computes the signature challenge from μ and $\mathbf{w}'$ and checks both proofs against the same context. Notice that the whole verification algorithm is executed in our protocol up to some reordering.

3.1. Security Analysis

In this section, we establish the unforgeability and unlinkability guarantees of our construction. We refer the reader to Appendix A for the full proofs.

Unforgeability relies on the computational hardness of SelfTargetMSIS as in Definition 9, the same assumption used to prove unforgeability of new messages for CRYSTALS-Dilithium [68, Sec. 4.1], and thus of ML-DSA.

Definition 9 (SelfTargetMSIS [68]). *Let H be a cryptographic hash function and $|\cdot\rangle$ denote quantum access to H .*

$$\begin{aligned} \text{Adv}_{H,m,k,\gamma}^{\text{SelfTargetMSIS}}(\mathcal{A}, n) := \\ \Pr[0 \leq \|y\|_\infty \leq \gamma \wedge H([I||A] \cdot y||M) = c \mid \\ A \leftarrow R_q^{m \times k}; (y := [r||c]^T, M) \leftarrow \mathcal{A}^{H(\cdot)}(A)] \end{aligned}$$

We underline that even though our reduction will not be directly to the security of ML-DSA, it will be equivalent since we reduce to SelfTargetMSIS with the same parameters as ML-DSA. The technical detail preventing us from directly reducing to ML-DSA is the core component allowing our performance gains. One can see ML-DSA as a two layer algorithm, the outside layer handles randomness and formats the real message to be signed using the user’s input message (and the public key), while the inside layer performs the real signing on data derived by the outside layer. Our technique essentially bypasses the outside layer, since a correct ML-DSA message for the inside layer is supposed to be of the form $H(\text{BytesToBits}(H(pk, 64))||M', 64)$, while we can only guarantee that $\mu = H(R_\mu||\text{ctx})$. This prevents us from directly using ML-DSA’s “API” for the reduction, but we still rely on the inside layer for security. We stress, however, that a direct reduction to ML-DSA could be obtained by additionally proving $tr = H(pk, 64)$ and using it to derive μ but we avoid doing so for efficiency.⁶

Theorem 1 (Unforgeability). *Assume that MSig is unforgeable, Com is binding, NIZK_{CD} and NIZK_{DB} are straight-line extractable and parameter-indistinguishable, and SelfT-MSIS is hard. Let Q_{SE} bound the number of ShowSE queries and q_H the number of H queries. Then*

$$\begin{aligned} \text{Adv}_{\text{DBAC}, \ell, \text{Ext}}^{\text{Unf}}(\mathcal{A}, n) \leq & \text{Adv}_{\text{NIZK}_{\text{CD}}}^{\text{ParInd}}(\mathcal{B}_1, n) + \text{Adv}_{\text{NIZK}_{\text{DB}}}^{\text{ParInd}}(\mathcal{B}_2, n) \\ & + \text{Adv}_{\text{MSig}, \ell}^{\text{Unf}}(\mathcal{B}_3, n) + \text{Adv}_{\text{Com}}^{\text{Bind}}(\mathcal{B}_4, n) \\ & + \mathcal{O}\left(\frac{q_H^2}{2^n}\right) + Q_{SE} \cdot \text{Adv}_H^{\text{SelfT-MSIS}}(\mathcal{B}_5, n) + \text{negl}(n) \end{aligned}$$

where we omit constant factor losses arising from straight-line extraction.

Turning to unlinkability, we show that the adversary cannot distinguish between a real presentation and a simulated presentation. Allowing w_1 to be public requires an

additional assumption: the weak-MLWR assumption. Intuitively, weak-MLWR is used to argue that the rounded value $w_1 = \text{HighBits}_q(w, 2\gamma_2)$ exposed in the presentation reveals no exploitable information about the user’s secrets. Appendix C formalizes this assumption and discusses its concrete hardness. In Section 4.4.2, we give an alternative that keeps w_1 hidden at additional proving cost.

Theorem 2 (Unlinkability).

$$\begin{aligned} \text{Adv}_{\Pi, \ell, \text{Sim}, Q_{SE}}^{\text{Unlink-1}}(\mathcal{A}, n) \leq & \text{Adv}_{\text{DB}}^{\text{ParInd}}(\mathcal{B}_1, n) + \text{Adv}_{\text{CD}}^{\text{ParInd}}(\mathcal{B}_2, n) \\ & + \text{Adv}_{\pi_{\text{DB}}}^{\text{ZK}}(\mathcal{B}_3, n) + \text{Adv}_{\pi_{\text{CD}}}^{\text{ZK}}(\mathcal{B}_4, n) + \text{Adv}_{\text{wMLWR}}(\mathcal{B}_5) \end{aligned}$$

4. Concrete Instantiation and Evaluation

In this section, we show how to instantiate our device binding module, discussing alternative approaches and how it can be plugged into existing anonymous credential schemes. We conclude by giving concrete performance numbers and discussing the efficiency of our approach.

As discussed in the introduction, $\mathcal{R}_{\text{DB}}$ can be split into two sub-relations, one to prove correctness of μ and $\mathbf{w}_1$, and one to prove the signature equation given knowledge of $\mathbf{z}$ and opening of the commitments C_i for $i \in S_{\text{DB}}$ alongside a bridge commitment for $\mathbf{w}$. The decomposition separates bit-level from algebraic work: $\mathcal{R}_{\text{DB1}}$ takes the SHAKE evaluation together with the bit decomposition underlying UseHint, and $\mathcal{R}_{\text{DB2}}$ covers the lattice-algebraic part of ML-DSA verification. While both sub-relations open C_w , this duplication is affordable because the opening is *linear* in the committed values, so it adds no quadratic constraints. We prove $\mathcal{R}_{\text{DB}}$ as a conjunction of the following two relations:

$$\begin{aligned} \mathcal{R}_{\text{DB1}} = & \left\{ \begin{array}{l} (\mathbf{w}, \mathbf{h}, R_\mu, o_w) : \mu = H(R_\mu||\text{ctx}) \\ \wedge \mathbf{w}_1 = \text{UseHint}(\mathbf{h}, \mathbf{w}) \wedge \text{hw}(\mathbf{h}) \leq w \\ \wedge C_w = \text{Com.Commit}(\mathbf{w}; o_w) \end{array} \right\} \\ \mathcal{R}_{\text{DB2}} = & \left\{ \begin{array}{l} (\mathbf{z}, \mathbf{w}, \rho, \text{dpk}, o_{i \in S_{\text{DB}}}, o_w) : \\ \text{Encode}(\text{dpk}) = \{\mathbf{A}_i, \mathbf{t}_{1,i}\}_{i \in S_{\text{DB}}} \wedge \\ \forall i \in S_{\text{DB}} : C_i = \text{Com.Commit}((\mathbf{A}_i, \mathbf{t}_{1,i}); o_i) \\ \wedge C_w = \text{Com.Commit}(\mathbf{w}; o_w) \\ \wedge \mathbf{A}\mathbf{z} = \mathbf{w} + \mathbf{c}\mathbf{t}_1 \cdot 2^d \wedge \|\mathbf{z}\|_\infty < \gamma_1 - \beta \end{array} \right\} \end{aligned}$$

The encoding of the dpk as a set of commitments has to be consistent with $\mathcal{R}_{\text{CD}}$. Since the issuer verifies the encoding during issuance, we do not need to prove correctness again and can instead rely on the soundness of $\mathcal{R}_{\text{CD}}$. Moreover, straight-line extraction of both proofs yields openings of the same C_w , so binding of Com ensures that the extracted witnesses contain the same $\mathbf{w}$, except with negligible probability. Hence, the conjunction of $\mathcal{R}_{\text{DB1}}$ and $\mathcal{R}_{\text{DB2}}$ implies $\mathcal{R}_{\text{DB}}$. We stress that C_w is an artifact of the decomposition, a monolithic proof of $\mathcal{R}_{\text{DB}}$ would not contain it (see Section 4.4 for the related discussion).

4.1. Proving $\mathcal{R}_{\text{DB1}}$

First, we recall that the function H used in ML-DSA is SHAKE256. Note that R_μ is chosen by the prover and well-formedness is not enforced here. Since the secure element

6. Hashing pk for ML-DSA-65 requires another 15 Keccak permutations.

signs a correctly formatted μ by construction, an R_μ of any other shape yields a μ for which no valid signature exists under the authenticated but hidden public key, resulting in $\mathcal{R}_{DB2}$ failing. Thus, formally, the input size is 98 bytes (64 bytes for tr , 2 bytes for the prefix defined in FIPS, 16 bytes for R , and 16 bytes for the ctx). Since SHAKE256 has a rate of 136 bytes, the entire input fits in a single absorb block; the circuit then applies the SHAKE padding and Keccak permutation, and squeezes the required 64-byte output. However, this is only one component of $\mathcal{R}_{DB1}$. The proof system must also handle bit-level decomposition and arithmetic in R_q . We considered the following post-quantum secure approaches to efficiently prove $\mathcal{R}_{DB1}$.

4.1.1. Garbled Circuits. This is the most conceptually simple approach and has the advantage of only requiring symmetric-key primitives in our case. The idea is to let the user act as the garbler and the verifier as the evaluator. Given public input ctx , the user garbles a circuit for $H(R_\mu || ctx, 64)$ and sends it alongside the garbled input labels for R and the output decoding table to the verifier. The verifier then evaluates the garbled circuit and obtains μ , which it uses to check the second proof. Note that no OT is required since the verifier does not have any private input (communication is round-optimal). The downside of this approach is communication. Even in the optimistic semi-honest setting, a garbled circuit for a single SHAKE256 evaluation is already large: the circuit contains one Keccak-f[1600] permutation, and standard garbling techniques lead to communication in the megabyte range. This is already substantially larger than the proof sizes we obtain with Longfellow for the whole of $\mathcal{R}_{DB1}$, and the comparison is generous to garbling: the previous outline only covers the μ component. Moreover, semi-honest security is not the right target for our final protocol, since the user acts as the prover and the verifier needs soundness against maliciously generated proofs. Achieving this with garbled circuits would require cut-and-choose or a maliciously secure garbling scheme, increasing communication further, potentially to tens of megabytes. We therefore consider garbled circuits a simple but communication-heavy baseline for proving $\mathcal{R}_{DB1}$.

4.1.2. VOLE-based Proofs. Another approach would be to use a state-of-the-art VOLE-based proof system. For example, a FAEST-style approach [69] could in principle be instantiated with a SHAKE256 circuit, yielding a publicly verifiable proof based on QuickSilver [70]. This direction is attractive because VOLE-based systems can provide malicious soundness with communication largely governed by the number of nonlinear gates. Unfortunately, existing FAEST implementations are heavily optimized for AES-style circuits and are not a drop-in backend for the SHAKE256 computation required by $\mathcal{R}_{DB1}$. Adapting them to our setting would require a SHAKE256-specific implementation, parameter selection, and benchmarking. The same caveat as above also applies: the remaining parts of the statement are arithmetic in R_q , which is not where VOLE-based systems over small fields are strongest. We therefore

view VOLE-based proofs as a promising future direction, but use Longfellow for our proof-of-concept.

4.1.3. Longfellow. As mentioned in the introduction, Longfellow could in principle be used to prove the full device-binding relation. However, using it in this monolithic way means the algebraic part is also compiled into a general-purpose circuit, so $\mathcal{R}_{DB2}$ can no longer exploit a native lattice proof system. Instead, we suggest using Longfellow only for $\mathcal{R}_{DB1}$, reusing the ML-DSA components of its reference implementation (the SHAKE circuit, UseHint, and the supporting gadgets) so that the components we add are the opening of C_w and the canonical UseHint constraints. Since Longfellow already arithmetizes over $\mathbb{F}_q$ with q the ML-DSA modulus, that opening is native field arithmetic.

4.2. Proving $\mathcal{R}_{DB2}$

The natural candidate for proving $\mathcal{R}_{DB2}$ would be to use the LNP proof system alongside the (A)BDLOP commitment scheme [71] as it allows one to efficiently prove the smallness of $\mathbf{z}$ and the quadratic relation $\mathbf{A}\mathbf{z} = \mathbf{w} + \mathbf{c}_1 \cdot 2^d$ (where witnesses are $(\mathbf{A}, \mathbf{z}, \mathbf{t}_1, \mathbf{w})$). Since the issuer ensures the encoding is correct during issuance, the commitments C_i for $i \in S_{DB}$ are guaranteed to be well-formed with respect to the device public key dpk as long as the selective disclosure proof of the credential is valid. However, since the DBAC model implicitly assumes straight-line extractability for Ext, we require it for NIZK_{DB2} . Unfortunately, LNP is not straight-line extractable. Two approaches are known to obtain straight-line extraction: the Katsumata approach consisting of using linear homomorphic commitments [64], [65], [66], and via the encrypt-and-prove paradigm [63]. Fortunately, quite recently, Nguyen and O’Rourke [72] showed how to build a more efficient ABDLOP variant (henceforth referred to as NO25) that yields a straight-line extractable proof system as required by our construction. Notably, extractability holds with respect to the parameter indistinguishability property of the commitment scheme, guaranteed under the hardness of MLWE. Hence, by using their commitment scheme one can keep the same hardness assumptions underlying ML-DSA, which is a plus for compatibility. That said, this approach leads to proofs that are four times bigger compared to plain LNP without extraction.

4.3. Integration with Anonymous Credentials

We now discuss how to integrate our device binding module with state-of-the-art anonymous credentials. We focus on lattice-based schemes, but the same approach can be applied to other types of schemes as well as long as the requirements for the device binding module are met.

As mentioned in the introduction, to the best of our knowledge, all recent approaches in the lattice-based setting [63], [65], [73], [74], [75], [76] follow the well-known paradigm of “signatures with efficient protocols” [4], which we adopt in this work as well. We can broadly categorize these schemes into two classes: those that rely on standard

lattice assumptions such as M-SIS and M-LWE [63], [73], [74], and those that rely on newer assumptions such as ISIS_f and SIS-with-Hints [65], [75], [76]. As one would expect, the former class of schemes is less efficient, while the latter are more efficient but require stronger assumptions.

Since the focus of this work is on enabling real-world deployments of anonymous credentials compatible with ML-DSA we turn our attention to [73], [74] and [63]. In this regard, both works rely on variants of the BD-LOP scheme and, as mentioned in [72], one can directly switch their commitment scheme with NO25, whose proof system inherently avoids rewinding and thereby provides the straight-line extractability needed for NIZK_{CD}. Indeed, these schemes are among those explicitly listed as out-of-the-box applications of NO25. The same also holds for the construction from [65], demonstrating the compatibility of our approach to upgrade plain lattice-based anonymous credential schemes with device binding. We stress, however, that NO25 has no public implementation, and LaZer [77], [78], the most performant LNP library available, is still under active development and tied to specific hardware instructions. We therefore see the portability and engineering of these components as an interesting area for future work.

Looking at Longfellow, a naive implementation based on ML-DSA would require the full verification of two signatures in-circuit, one from the issuer and one from the SE. Based on the available benchmarks from [56], this would already set a baseline of more than a second to verify the two signatures, highlighting the need of our decomposition to bring Longfellow on ML-DSA to a practical level.

4.4. Evaluation

Our preferred modular instantiation uses a general-purpose proof system for $\mathcal{R}_{DB1}$ and a native lattice proof system for $\mathcal{R}_{DB2}$. However, the straight-line extractable lattice proof systems discussed before do not currently provide a sufficiently mature public implementations. We therefore instantiate both sub-relations with Longfellow, preserving the same interface required by future instantiations.

The official Longfellow repository already contains a test circuit for ML-DSA-65 verification [56], which we take as both our baseline and starting point. Two aspects of that circuit are worth making explicit. First, it proves verification with respect to a *known* public key: $\hat{\mathbf{A}}$ and $\text{NTT}(\mathbf{t}_1 \cdot 2^d)$ are public inputs, so ExpandA is not proven and the statement is not bound to the seed ρ . Second, μ is an unconstrained private witness (the circuit proves knowledge of a signature on *some* message representative), with the binding of μ to the key and to the message assumed to be checked externally. While both are reasonable choices for a standard PoP, we adjusted the implementation to supply $\hat{\mathbf{A}}$ and $\mathbf{t}_1$ from a BDLOP commitment, and we prove the derivation of μ inside $\mathcal{R}_{DB1}$. We additionally enforce the canonical FIPS decomposition for UseHint, which the reference circuit leaves underdetermined because there $\mathbf{w}_1$ is an internal witness rather than published. The related artifacts and explanatory notes are available at [79]. An Apple M4 with

Table 3: Performance of our three variants for ML-DSA-65, running Longfellow with rate = 4, nreq = 128. We take the medians over five runs. $|x_{DB}|$ denotes the public part of the statement that is transmitted with the proof: $|C_{dpk}|$, $|C_w|$, $|\mathbf{w}_1|$, $|\mu|$ or $|\tilde{c}|$, and $|\text{ctx}|$. In the totals C_w is counted once.

Variant	$\mathbf{w}_1$	Proof	Prove	Verify	$ \pi $	$ x_{DB} $
Bridged	public	π_{DB1}	233 ms	118 ms	517 kB	6.0 kB
		π_{DB2}	329 ms	215 ms	553 kB	32.4 kB
		<i>total</i>	<i>562 ms</i>	<i>334 ms</i>	<i>1069 kB</i>	<i>33.2 kB</i>
Hidden	$\mathbf{w}_1$ private	π'_{DB1}	741 ms	358 ms	823 kB	5.2 kB
		π_{DB2}	329 ms	215 ms	553 kB	32.4 kB
		<i>total</i>	<i>1069 ms</i>	<i>573 ms</i>	<i>1375 kB</i>	<i>32.4 kB</i>
Monolithic	public	π_{DB}	485 ms	289 ms	852 kB	28.1 kB
Longfellow’s test PoP [56]			923 ms	499 ms	614 kB	—

24GB of RAM was used to run the benchmarks. We consider different variants, showcasing the versatility of our work.

4.4.1. Bridged decomposition. This is the preferred construction where $\mathcal{R}_{DB1}$ and $\mathcal{R}_{DB2}$ take common input C_w . As the Longfellow’s circuit operates over the native ML-DSA field, the algebraic relation and the commitment openings are direct field arithmetic; and because $\mathbf{w}_1$ is public, the costly SampleInBall and the derivation of $\tilde{c}$ are avoided in-circuit. To keep things modular, $\mathcal{R}_{DB2}$ takes dpk in a committed form using BDLOP commitments with suitable parameters obtained with the lattice-estimator [80]. This enables the interoperability needed with NIZK_{CD}.

4.4.2. Variant with hidden $\mathbf{w}_1$. The weak MLWR assumption is used to make $\mathbf{w}_1$ public for efficiency. However, we can keep $\mathbf{w}_1$ private and compute $\tilde{c} \leftarrow H(\mu \parallel \text{w1Encode}(\mathbf{w}_1), \lambda/4)$ in-circuit. The concrete cost is seven further Keccak permutations, as $\mu \parallel \text{w1Encode}(\mathbf{w}_1)$ is 832 bytes. Importantly, $\mathcal{R}_{DB2}$ is reused *unchanged*: it consumes only c , and the commitments to dpk are likewise unchanged, so the interface to NIZK_{CD} is preserved.

4.4.3. Monolithic variant. This variant proves $\mathcal{R}_{DB}$ as a monolithic circuit. We retain the commitment to dpk, the in-circuit derivation of μ , and the canonical UseHint constraints; only C_w is dropped ($\mathbf{w}$ becomes an internal wire). Note that this preserves interoperability with lattice-based credentials: C_{dpk} is the same BDLOP commitment as in the other variants, so NIZK_{CD} can open it unchanged. Avoiding the second opening of C_w and doing a single sumcheck is what makes this the most performant variant assuming sequential execution. We recall that the commitment to dpk assumes the Issuer certifies it at issuance time. This matters considering the integration to Longfellow’s credential model, which does not consider any change on the issuer side (*i.e.*, it evidences a critical aspect for future MDOC standardization efforts, as the alternative of proving ExpandA in-circuit would add roughly 150 Keccak permutations, making it prohibitively expensive).

Table 4: Performance of our proving times on an iPhone 17.

Variant	w_1	Proof	Prove	$ \pi $	$ x_{DB} $
Bridged	public	π_{DB_1}	210 ms	517 kB	6.0 kB
		π_{DB_2}	306 ms	553 kB	32.4 kB
		<i>total</i>	<i>516 ms</i>	<i>1069 kB</i>	<i>33.2 kB</i>
		<i>concurrent</i>	<i>357 ms</i>	<i>1069 kB</i>	<i>33.2 kB</i>
Hidden w_1	private	π'_{DB_1}	724 ms	823 kB	5.2 kB
		π_{DB_2}	306 ms	553 kB	32.4 kB
		<i>total</i>	<i>1030 ms</i>	<i>1375 kB</i>	<i>32.4 kB</i>
		<i>concurrent</i>	<i>823 ms</i>	<i>1375 kB</i>	<i>32.4 kB</i>
Monolithic	public	π_{DB}	455 ms	852 kB	28.1 kB
Longfellow's test PoP [56]			959 ms	614 kB	—

We note that one could modify the above variants to additionally prove $tr \leftarrow H(\text{dpk}, 64)$, obtaining a direct reduction to ML-DSA instead of SelfTargetMSIS at the cost of 15 additional Keccak permutations. Alternatively, the Issuer could certify tr alongside the encoding of dpk at issuance time, avoiding this additional in-circuit computation.

We present our experimental evaluation in Table 3. All in all, the running times and proof sizes confirm the little overhead of Nowhere Man, positioning our solution as a flexible device binding module to be integrated with the next generation of anonymous credential schemes.

4.4.4. Mobile deployment. Preliminary experiments targeting a mobile deployment were conducted after the artifact of this paper was submitted. In this extended version we therefore report the numbers obtained by running the artifact code on an iPhone 17. Changes with respect to the original artifact are minor: we used Apple's CommonCrypto rather than OpenSSL, and we added tests for the concurrent execution of the two sub-proofs. The latter is possible because the prover derives c locally from μ and w_1 , so $\mathcal{R}_{DB_2}$ does not wait on a proof of $\mathcal{R}_{DB_1}$ and the bridged prover costs $\max(\pi_{DB_1}, \pi_{DB_2})$ rather than their sum. Table 4 reports the results. The key takeaway is that the monolithic variant is no longer the fastest one, as the bridged variant can be run concurrently on two cores.

References

- [1] D. Chaum, "Security without identification: transaction systems to make big brother obsolete," *Commun. ACM*, vol. 28, no. 10, Oct. 1985. [Online]. Available: <https://doi.org/10.1145/4372.4373>
- [2] —, "Showing credentials without identification: Signatures transferred between unconditionally unlinkable pseudonyms," in *EUROCRYPT'85*, ser. LNCS, F. Pichler, Ed., vol. 219. Springer, Berlin, Heidelberg, Apr. 1986, pp. 241–244.
- [3] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials with optional anonymity revocation," in *EUROCRYPT 2001*, ser. LNCS, B. Pfitzmann, Ed., vol. 2045. Springer, Berlin, Heidelberg, May 2001, pp. 93–118.
- [4] —, "A signature scheme with efficient protocols," in *SCN 02*, ser. LNCS, S. Cimato, C. Galdi, and G. Persiano, Eds., vol. 2576. Springer, Berlin, Heidelberg, Sep. 2003, pp. 268–289.
- [5] —, "Signature schemes and anonymous credentials from bilinear maps," in *CRYPTO 2004*, ser. LNCS, M. Franklin, Ed., vol. 3152. Springer, Berlin, Heidelberg, Aug. 2004, pp. 56–72.
- [6] A. Davidson, J. Iyengar, and C. A. Wood, "The Privacy Pass Architecture," RFC 9576, Jun. 2024. [Online]. Available: <https://www.rfc-editor.org/info/rfc9576>
- [7] C. Yun, C. A. Wood, and A. Faz-Hernandez, "Anonymous Rate-Limited Credentials Cryptography," Internet Engineering Task Force, Internet-Draft draft-ietf-privacypass-arc-crypto-01, Mar. 2026, work in Progress. [Online]. Available: <https://datatracker.ietf.org/doc/draft-ietf-privacypass-arc-crypto/01/>
- [8] S. Schlesinger and J. Katz, "Anonymous Credit Tokens," Internet Engineering Task Force, Internet-Draft draft-schlesinger-cfrg-act-01, Feb. 2026, work in Progress. [Online]. Available: <https://datatracker.ietf.org/doc/draft-schlesinger-cfrg-act/01/>
- [9] C. Yun and C. A. Wood, "Anonymous Rate-Limited Credentials," Internet Engineering Task Force, Internet-Draft draft-yun-cfrg-arc-00, Feb. 2025, work in Progress. [Online]. Available: <https://www.ietf.org/archive/id/draft-yun-cfrg-arc-00.html>
- [10] C. Baum, O. Blazy, J. Camenisch, J.-H. Hoepman, E. Lee, A. Lehmann, A. Lysyanskaya, R. Mayrhofer, H. Montgomery, N. K. Nguyen, B. Preneel, a. shelat, D. Slamanig, S. Tessaro, S. E. Thomsen, and C. Troncoso, "Cryptographers' feedback on the EU digital identity's ARF," <https://github.com/user-attachments/files/15904122/cryptographers-feedback.pdf>, 2024.
- [11] C. Knabenhans, S. Veitch, M. Raynal, T. Stadler, S. Chatel, W. Lueks, and C. Troncoso, "On the (privacy) harms of the european digital identity framework," Cryptology ePrint Archive, Paper 2026/867, 2026. [Online]. Available: <https://eprint.iacr.org/2026/867>
- [12] S. Tessaro and C. Zhu, "Revisiting BBS signatures," in *EUROCRYPT 2023, Part V*, ser. LNCS, C. Hazay and M. Stam, Eds., vol. 14008. Springer, Cham, Apr. 2023, pp. 691–721.
- [13] K. Friedrichs, F. Harding, A. Lehmann, and A. Lysyanskaya, "Device-bound anonymous credentials with(out) trusted hardware," in *EUROCRYPT 2026*. Springer Nature Switzerland, 2026.
- [14] R. Chairattana-Apirom, D. Hofheinz, and S. Tessaro, "Tight security for BBS signatures," Cryptology ePrint Archive, Paper 2025/1973, 2025. [Online]. Available: <https://eprint.iacr.org/2025/1973>
- [15] R. Chairattana-Apirom and S. Tessaro, "On the concrete security of BBS/BBS+ signatures," in *ASIACRYPT 2025, Part VI*, ser. LNCS, G. Hanaoka and B.-Y. Yang, Eds., vol. 16250. Springer, Singapore, Dec. 2025, pp. 402–434.
- [16] N. Desmoulins, A. Dumanois, S. Kane, and J. Traoré, "Making BBS anonymous credentials eIDAS 2.0 compliant," Cryptology ePrint Archive, Paper 2025/619, 2025. [Online]. Available: <https://eprint.iacr.org/2025/619>
- [17] R. Chairattana-Apirom, F. Harding, A. Lysyanskaya, and S. Tessaro, "Server-aided anonymous credentials," in *CRYPTO 2025, Part VI*, ser. LNCS, Y. T. Kalai and S. F. Kamara, Eds., vol. 16005. Springer, Cham, Aug. 2025, pp. 291–324.
- [18] A. Flamini, E. Lee, and A. Lysyanskaya, "Multi-holder anonymous credentials from BBS signatures," in *CRYPTO 2025, Part VI*, ser. LNCS, Y. T. Kalai and S. F. Kamara, Eds., vol. 16005. Springer, Cham, Aug. 2025, pp. 325–357.
- [19] M. Orrù, "Revisiting keyed-verification anonymous credentials," in *ACM CCS 2025*, C.-Y. Huang, J.-C. Chen, S.-P. Shieh, D. Lie, and V. Cortier, Eds. ACM Press, Oct. 2025, pp. 1188–1199.
- [20] R. Mayrhofer, M. Roland, T. Höller, P. Hofer, and M. Lins, "An architecture for distributed digital identities in the physical world," 2025. [Online]. Available: <https://arxiv.org/abs/2508.10185>
- [21] Transportation Security Administration, "TSA digital ID," <https://www.tsa.gov/digital-id>, 2026.

- [22] International Organization for Standardization and International Electrotechnical Commission, “Personal identification — ISO-compliant driving licence — Part 5: Mobile driving licence (mDL) application,” International Organization for Standardization, Standard ISO/IEC 18013-5:2021, september 2021. [Online]. Available: <https://www.iso.org/standard/69084.html>
- [23] M. Frigo and abhi shelat, “Anonymous credentials from ECDSA,” *IACR Communications in Cryptology*, vol. 3, no. 1, 2026.
- [24] Google LLC, “Opening up zero-knowledge proof technology to promote privacy in age assurance,” <https://blog.google/innovation-and-ai/technology/safety-security/opening-up-zero-knowledge-proof-technology-to-promote-privacy-in-age-assurance/>, may 2024.
- [25] M. Frigo, a. shelat, T. Geoghegan, and D. Cook, “libZK: A zero knowledge proof library (The Longfellow ZK Scheme),” Presentation at the Crypto Forum Research Group (CFRG), IETF 125, Shenzhen, Mar. 2026, slides. [Online]. Available: <https://datatracker.ietf.org/meeting/125/materials/slides-125-cfrg-longfellow-zk-00>
- [26] M. Frigo and a. shelat, “libZK: A zero-knowledge proof library: The Longfellow ZK Scheme,” Presentation at the Crypto Forum Research Group (CFRG), IETF 123, 2025. [Online]. Available: <https://datatracker.ietf.org/meeting/123/materials/slides-123-cfrg-lib-zk-a-zero-knowledge-proof-library-the-longfellow-zk-scheme-00>
- [27] —, “libZK: a zero-knowledge proof library,” Internet Engineering Task Force, Internet-Draft draft-google-cfrg-libzk-00, March 2025, work in progress. [Online]. Available: <https://datatracker.ietf.org/doc/draft-google-cfrg-libzk/00/>
- [28] D. Kaviani and S. Setty, “Vega: Low-latency zero-knowledge proofs over existing credentials,” in *IEEE S&P*. IEEE, 2026. [Online]. Available: <https://doi.org/10.1109/SP63933.2026.00182>
- [29] S. Setty, “Vega: Zero-knowledge proofs for digital identity in the age of AI,” Microsoft Research Blog, May 2026. [Online]. Available: <https://www.microsoft.com/en-us/research/blog/vega-zero-knowledge-proofs-for-digital-identity-in-the-age-of-ai/>
- [30] A. Lehmann, A. Sidorenko, and A. Zacharakis, “Vision: A modular framework for anonymous credential systems,” in *SSR*, ser. Lecture Notes in Computer Science, vol. 16466. Springer, 2025. [Online]. Available: https://doi.org/10.1007/978-3-032-19567-8_3
- [31] S. Celi, A. Lehmann, S. Levin, and A. Zacharakis, “Device binding for anonymous credentials on legacy phones,” in *IEEE S&P*, 2027, to appear. [Online]. Available: <https://eprint.iacr.org/2026/965>
- [32] T. Looker, V. Kalos, A. Whitehead, and M. Lodder, “The BBS Signature Scheme,” Internet Engineering Task Force, Internet-Draft draft-irtf-cfrg-bbs-signatures-10, Jan. 2026, work in Progress. [Online]. Available: <https://datatracker.ietf.org/doc/draft-irtf-cfrg-bbs-signatures/10/>
- [33] ISO/IEC JTC 1/SC 27, “Information security — Attribute-based credentials,” International Organization for Standardization, Working Draft ISO/IEC WD 24843.2, 2026, under development; stage 20.60, close of comment period. [Online]. Available: <https://www.iso.org/standard/91042.html>
- [34] ETSI Technical Committee Electronic Signatures and Trust Infrastructures (ESI), “Electronic Signatures and Trust Infrastructures (ESI); Analysis of selective disclosure and zero-knowledge proofs applied to Electronic Attestation of Attributes,” European Telecommunications Standards Institute, Technical Report ETSI TR 119 476 V1.2.1, Jul. 2024, reference RTR/ESI-0019476v121. [Online]. Available: https://www.etsi.org/deliver/etsi_tr/119400_119499/119476/01.02.01_60/tr_119476v010201p.pdf
- [35] L. Kissner, “Crypto is a tool for turning a whole swathe of problems into key management problems,” Post on X, Nov. 2019. [Online]. Available: <https://x.com/LeaKissner/status/1198595109756887040>
- [36] C. Paquin, G.-V. Policharla, and G. Zaverucha, “Crescent: Stronger privacy for existing credentials,” Cryptology ePrint Archive, Paper 2024/2013, 2024. [Online]. Available: <https://eprint.iacr.org/2024/2013>
- [37] M. Rosenberg, J. D. White, C. Garman, and I. Miers, “zk-creds: Flexible anonymous credentials from zkSNARKs and existing identity infrastructure,” in *2023 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 2023, pp. 790–808.
- [38] European Digital Identity Cooperation Group, “European Digital Identity Wallet: Architecture and Reference Framework,” <https://eudi.dev/2.9.0/architecture-and-reference-framework-main/>, 2026, version 2.9.0.
- [39] M. Ben-Or, O. Goldreich, S. Goldwasser, J. Håstad, J. Kilian, S. Micali, and P. Rogaway, “Everything provable is provable in zero-knowledge,” in *CRYPTO ’88*, ser. LNCS, S. Goldwasser, Ed., vol. 403. Springer, New York, Aug. 1990, pp. 37–56.
- [40] R. S. Wahby, Y. Ji, A. J. Blumberg, a. shelat, J. Thaler, M. Walfish, and T. Wies, “Full accounting for verifiable outsourcing,” in *ACM CCS 2017*, B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, Eds. ACM Press, Oct. / Nov. 2017, pp. 2071–2086.
- [41] R. S. Wahby, I. Tzialis, a. shelat, J. Thaler, and M. Walfish, “Doubly-efficient zkSNARKs without trusted setup,” in *2018 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 2018, pp. 926–943.
- [42] S. Ames, C. Hazay, Y. Ishai, and M. Venkatasubramanian, “Ligero: lightweight sublinear arguments without a trusted setup,” *DCC*, vol. 91, no. 11, pp. 3379–3424, 2023.
- [43] A. Kothapalli, S. Setty, and I. Tzialis, “Nova: Recursive zero-knowledge arguments from folding schemes,” in *CRYPTO 2022, Part IV*, ser. LNCS, Y. Dodis and T. Shrimpton, Eds., vol. 13510. Springer, Cham, Aug. 2022, pp. 359–388.
- [44] A. Kothapalli and S. Setty, “NeutronNova: Folding everything that reduces to zero-check,” Cryptology ePrint Archive, Paper 2024/1606, 2024. [Online]. Available: <https://eprint.iacr.org/2024/1606>
- [45] S. Setty, “Spartan: Efficient and general-purpose zkSNARKs without trusted setup,” in *CRYPTO 2020, Part III*, ser. LNCS, D. Micciancio and T. Ristenpart, Eds., vol. 12172. Springer, Cham, Aug. 2020, pp. 704–737.
- [46] A. Kothapalli and B. Parno, “Algebraic reductions of knowledge,” in *CRYPTO 2023, Part IV*, ser. LNCS, H. Handschuh and A. Lysyanskaya, Eds., vol. 14084. Springer, Cham, Aug. 2023, pp. 669–701.
- [47] D. Moody, “NIST PQC: The Road Ahead,” Presentation slides at the Real World Crypto (RWC) PQC event, March 2025. [Online]. Available: <https://csrc.nist.gov/csrc/media/Presentations/2025/nist-pqc-the-road-ahead/images-media/rwcpqc-march2025-moody.pdf>
- [48] H. Adkins and S. Schmieg, “Quantum frontiers may be closer than they appear,” The Keyword (Google Blog), March 2026. [Online]. Available: <https://blog.google/innovation-and-ai/technology/safety-security/cryptography-migration-timeline/>
- [49] National Institute of Standards and Technology, “Crypto Agility,” <https://csrc.nist.gov/projects/crypto-agility>, 2025.
- [50] National Institute of Standards and Technology (US), “Module-lattice-based digital signature standard,” National Institute of Standards and Technology (U.S.), Tech. Rep. NIST FIPS 204, Aug. 2024. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.204.pdf>
- [51] M. Larabel, “Linux 7.0 Lands ML-DSA Quantum-Resistant Signature Support,” Phoronix, February 2026. [Online]. Available: <https://www.phoronix.com/news/Linux-7.0-Crypto-ML-DSA>
- [52] Thales Group, “The post-quantum cryptography future: What telcos need to know,” Online, May 2026. [Online]. Available: <https://www.thalesgroup.com/en/news-centre/insights/enterprise/mobile-communications/post-quantum-cryptography-future-what-telcos>
- [53] Samsung, “Where trust begins: Exynos anchors post-quantum security at the root of mobile SoCs,” Online, feb 2026. [Online]. Available: <https://semiconductor.samsung.com/news-events/tech-blog/where-trust-begins-exynos-anchors-post-quantum-security-at-the-root-of-mobile-socs/>

- [54] Apple Inc., “The corecrypto (cc) project,” <https://github.com/apple/corecrypto>, May 2026.
- [55] Consumer Technology Association (CTA), “CES 2026 innovation awards honoree: S3SSE2A,” <https://www.ces.tech/ces-innovation-awards/2026/s3sse2a/>, Nov. 2025.
- [56] Google LLC, “Longfellow: A zero-knowledge proof library for post-quantum signature verification,” <https://github.com/google/longfellow-zk>, 2025, mL-DSA signature verification test circuits available at commit 5f348de0bedcc49fe3800a57f6bf2eecd0389391.
- [57] National Institute of Standards and Technology (US), “SHA-3 standard : permutation-based hash and extendable-output functions,” National Institute of Standards and Technology (U.S.), Tech. Rep. NIST FIPS 202, 2015. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf>
- [58] E. Eaton, D. Stebila, and R. Stracovsky, “Post-quantum key-blinding for authentication in anonymity networks,” in *LATINCRYPT*. Springer-Verlag, 2021.
- [59] C. Deegan, J. Fitzwater, K. D. Gur, and D. Nugent, “Lattice HD wallets: Post-quantum BIP32 hierarchical deterministic wallets from lattice assumptions,” Cryptology ePrint Archive, Paper 2026/380, 2026. [Online]. Available: <https://eprint.iacr.org/2026/380>
- [60] M. Seck and A. Roux-Langlois, “Towards post-quantum bitcoin blockchain using dilithium signature,” *CiC*, vol. 2, no. 3, p. 3, 2025.
- [61] N. Alkeilani Alkadri, P. Das, A. Erwig, S. Faust, J. Krämer, S. Riahi, and P. Struck, “Deterministic wallets in a quantum world,” in *ACM CCS 2020*, J. Ligatti, X. Ou, J. Katz, and G. Vigna, Eds. ACM Press, Nov. 2020, pp. 1017–1031.
- [62] National Institute of Standards and Technology, “FIPS 204: Module-lattice-based digital signature standard,” Aug. 2024. [Online]. Available: <https://doi.org/10.6028/NIST.FIPS.204>
- [63] Q. Lai, C. Chen, F.-H. Liu, A. Lysyanskaya, and Z. Wang, “Lattice-based commit-transferrable signatures and applications to anonymous credentials,” Cryptology ePrint Archive, Report 2023/766, 2023. [Online]. Available: <https://eprint.iacr.org/2023/766>
- [64] R. del Pino and S. Katsumata, “A new framework for more efficient round-optimal lattice-based (partially) blind signature via trapdoor sampling,” in *CRYPTO 2022, Part II*, ser. LNCS, Y. Dodis and T. Shrimpton, Eds., vol. 13508. Springer, Cham, Aug. 2022, pp. 306–336.
- [65] J. Bootle, V. Lyubashevsky, N. K. Nguyen, and A. Sorniotti, “A framework for practical anonymous credentials from lattices,” in *CRYPTO*, ser. Lecture Notes in Computer Science. Springer, 2023. [Online]. Available: https://doi.org/10.1007/978-3-031-38545-2_13
- [66] S. Katsumata, “A new simple technique to bootstrap various lattice zero-knowledge proofs to QROM secure NIZKs,” in *CRYPTO 2021, Part II*, ser. LNCS, T. Malkin and C. Peikert, Eds., vol. 12826. Virtual Event: Springer, Cham, Aug. 2021, pp. 580–610.
- [67] J. Camenisch and M. Stadler, “Efficient group signature schemes for large groups,” in *CRYPTO 97*. Springer Berlin Heidelberg, 1997.
- [68] V. Lyubashevsky, L. Ducas, E. Kiltz, T. Lepoint, P. Schwabe, G. Seiler, and D. Stehlé, “CRYSTALS-DILITHIUM,” National Institute of Standards and Technology, Tech. Rep., 2017, available at <https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-1-submissions>.
- [69] C. Baum, L. Braun, C. Delpech de Saint Guilhem, M. Klooß, E. Orsini, L. Roy, and P. Scholl, “Publicly verifiable zero-knowledge and post-quantum signatures from VOLE-in-the-head,” in *CRYPTO 2023, Part V*, ser. LNCS, H. Handschuh and A. Lysyanskaya, Eds., vol. 14085. Springer, Cham, Aug. 2023, pp. 581–615.
- [70] K. Yang, P. Sarkar, C. Weng, and X. Wang, “QuickSilver: Efficient and affordable zero-knowledge proofs for circuits and polynomials over any field,” in *ACM CCS 2021*, G. Vigna and E. Shi, Eds. ACM Press, Nov. 2021, pp. 2986–3001.
- [71] V. Lyubashevsky, N. K. Nguyen, and M. Plançon, “Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general,” in *CRYPTO 2022, Part II*, ser. LNCS, Y. Dodis and T. Shrimpton, Eds., vol. 13508. Springer, Cham, Aug. 2022, pp. 71–101.
- [72] N. K. Nguyen and G. O’Rourke, “More efficient lattice-based zero-knowledge proofs with straight-line extractability,” in *ACM APKC*. Association for Computing Machinery, 2025. [Online]. Available: <https://doi.org/10.1145/3709015.3728673>
- [73] C. Jeudy, A. Roux-Langlois, and O. Sanders, “Lattice signature with efficient protocols, application to anonymous credentials,” in *CRYPTO 2023, Part II*, ser. LNCS, H. Handschuh and A. Lysyanskaya, Eds., vol. 14082. Springer, Cham, Aug. 2023, pp. 351–383.
- [74] S. Argo, T. Güneysu, C. Jeudy, G. Land, A. Roux-Langlois, and O. Sanders, “Practical post-quantum signatures for privacy,” in *ACM CCS 2024*, B. Luo, X. Liao, J. Xu, E. Kirda, and D. Lie, Eds. ACM Press, Oct. 2024, pp. 1523–1537.
- [75] A. Dubois, M. Klooß, R. W. F. Lai, and I. K. Y. Woo, “Lattice-based proof-friendly signatures from vanishing short integer solutions,” in *PKC 2025, Part I*, ser. LNCS, T. Jager and J. Pan, Eds., vol. 15674. Springer, Cham, May 2025, pp. 452–486.
- [76] N. K. Nguyen and J. N. Siemer, “Tight reductions for SIS-with-hints assumptions with applications to anonymous credentials,” To appear at PKC 2026, 2026. [Online]. Available: <https://eprint.iacr.org/2026/291>
- [77] V. Lyubashevsky, G. Seiler, and P. Steuer, “The LaZer library: Lattice-based zero knowledge and succinct proofs for quantum-safe privacy,” in *ACM CCS 2024*, B. Luo, X. Liao, J. Xu, E. Kirda, and D. Lie, Eds. ACM Press, Oct. 2024, pp. 3125–3137.
- [78] B. Biasioli, M. Bolboceanu, V. Lyubashevsky, A. Merino-Gallardo, M. Osadnik, G. Seiler, and P. Steuer, “A toolkit for succinct lattice-based zero knowledge proofs,” Cryptology ePrint Archive, Paper 2026/1289, 2026. [Online]. Available: <https://eprint.iacr.org/2026/1289>
- [79] NTT, Inc., “Nowhere man: Device binding for post-quantum anonymous credentials – research prototype,” 2026. [Online]. Available: <https://doi.org/10.5281/zenodo.22793031>
- [80] M. R. Albrecht, R. Player, and S. Scott, “On the concrete hardness of learning with errors,” *Journal of Mathematical Cryptology*, October 2015.
- [81] A. Banerjee, C. Peikert, and A. Rosen, “Pseudorandom functions and lattices,” in *EUROCRYPT 2012*, ser. LNCS, D. Pointcheval and T. Johansson, Eds., vol. 7237. Springer, Berlin, Heidelberg, Apr. 2012, pp. 719–737.
- [82] S. Belaïd, R. Benadjila, J. Devevey, M. Guereau, T. Legavre, A. Martinelli, T. Ricosset, M. Rivain, and M. Rossi, “ML-dsa masking sweetened with sucre: Shuffle-and-unmask countermeasure for rejection sampling,” *IACR TCHES*, vol. 2026, no. 1, Jan. 2026. [Online]. Available: <https://tches.iacr.org/index.php/TCHES/article/view/12695>
- [83] J. Alwen, S. Krenn, K. Pietrzak, and D. Wichs, “Learning with rounding, revisited - new reduction, properties and applications,” in *CRYPTO 2013, Part I*, ser. LNCS, R. Canetti and J. A. Garay, Eds., vol. 8042. Springer, Berlin, Heidelberg, Aug. 2013, pp. 57–74.
- [84] F.-H. Liu and Z. Wang, “Rounding in the rings,” in *CRYPTO 2020, Part II*, ser. LNCS, D. Micciancio and T. Ristenpart, Eds., vol. 12171. Springer, Cham, Aug. 2020, pp. 296–326.
- [85] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, “CRYSTALS-Dilithium: A lattice-based digital signature scheme,” *IACR TCHES*, vol. 2018, no. 1, pp. 238–268, 2018. [Online]. Available: <https://tches.iacr.org/index.php/TCHES/article/view/839>

Appendix A. Deferred Security Proofs

Theorem 1 (Unforgeability). *Assume that MSig is unforgeable, Com is binding, NIZK_{CD} and NIZK_{DB} are straight-line extractable and parameter-indistinguishable, and SelfT-MSIS is hard. Let Q_{SE} bound the number of ShowSE queries and q_H the number of H queries. Then*

$$\begin{aligned} \text{Adv}_{\text{DBAC}, \ell, \text{Ext}}^{\text{Unf}}(\mathcal{A}, n) &\leq \text{Adv}_{\text{NIZK}_{\text{CD}}}^{\text{ParInd}}(\mathcal{B}_1, n) + \text{Adv}_{\text{NIZK}_{\text{DB}}}^{\text{ParInd}}(\mathcal{B}_2, n) \\ &\quad + \text{Adv}_{\text{MSig}, \ell}^{\text{Unf}}(\mathcal{B}_3, n) + \text{Adv}_{\text{Com}}^{\text{Bind}}(\mathcal{B}_4, n) \\ &\quad + \mathcal{O}\left(\frac{q_H^2}{2^n}\right) + Q_{\text{SE}} \cdot \text{Adv}_{\text{H}}^{\text{SelfT-MSIS}}(\mathcal{B}_5, n) + \text{negl}(n). \end{aligned}$$

where we omit constant factor losses arising from straight-line extraction.

Proof. We define our extractor $\text{Ext} = (\text{Setup}, \text{EKey})$ as:

$\text{Ext.Setup}(1^n, \ell)$: It sets $\text{pp} \leftarrow (1^n, \ell)$ and runs $\text{pp}_{\text{Sig}} \leftarrow \text{Sig.Setup}(\text{pp})$, $\text{pp}_{\text{Com}} \leftarrow \text{Com.Setup}(\text{pp})$, $(\text{pp}_{\text{CD}}, \mathcal{T}_{\text{CD}}) \leftarrow \text{NIZK}_{\text{CD}}.\text{SimSetup}(\text{pp})$, and $(\text{pp}_{\text{DB}}, \mathcal{T}_{\text{DB}}) \leftarrow \text{NIZK}_{\text{DB}}.\text{SimSetup}(\text{pp})$. It outputs (td, par) with $\text{td} \leftarrow (\mathcal{T}_{\text{CD}}, \mathcal{T}_{\text{DB}})$ and $\text{par} \leftarrow (\text{pp}_{\text{Sig}}, \text{pp}_{\text{Com}}, \text{pp}_{\text{CD}}, \text{pp}_{\text{DB}})$.

$\text{Ext.EKey}(\text{td}, \text{ipk}, \tau^*, \text{ctx}^*, \phi^*)$: It parses $(\{C_i\}_{S_{\text{Com}}}, \mathbf{w}_1, \mu, \pi_{\text{CD}}, \pi_{\text{DB}}) \leftarrow \tau^*$, straight-line extracts a witness w_{DB} using $\mathcal{T}_{\text{DB}}$, and returns the dpk^* encoded in w_{DB} .

Since Ext.Setup differs from DBAC.Setup only in replacing $\text{NIZK}_{\text{CD}}.\text{Setup}$ and $\text{NIZK}_{\text{DB}}.\text{Setup}$ by their respective simulated setups, the parameter indistinguishability of NIZK_{CD} and NIZK_{DB} (Definition 3) bounds the advantage of any adversary in distinguishing the two setups by $\text{Adv}_{\text{NIZK}_{\text{CD}}}^{\text{ParInd}}(\mathcal{B}_1, n) + \text{Adv}_{\text{NIZK}_{\text{DB}}}^{\text{ParInd}}(\mathcal{B}_2, n)$.

We formalize the proof over a sequence of game hops, where we denote $\Pr[\text{Game } i] := \Pr[\text{Unf}_{\text{DBAC}, \ell, \text{Ext}}^{\text{A}}(n) = 1 \mid \text{Game } i]$ the probability of the experiment outputting 1 under the conditions imposed in Game i .

Game 0 This is the unforgeability experiment $\text{Unf}_{\text{DBAC}, \ell, \text{Ext}}^{\text{A}}(n)$ (cf. Figure 2) with the adversary $\mathcal{A}$, our construction from Figure 4, and the extractor Ext defined above, where Ext.EKey straight-line extracts the accepted NIZK_{DB} proof and recovers the device public key dpk^* from its witness. By parameter indistinguishability of the two NIZK setups, this results in an initial advantage of

$$\begin{aligned} \text{Adv}_{\text{DBAC}, \ell, \text{Ext}}^{\text{Unf}}(\mathcal{A}, n) &\leq \text{Adv}_{\text{NIZK}_{\text{CD}}}^{\text{ParInd}}(\mathcal{B}_1, n) \\ &\quad + \text{Adv}_{\text{NIZK}_{\text{DB}}}^{\text{ParInd}}(\mathcal{B}_2, n) + \Pr[\text{Game } 0], \end{aligned}$$

Game 1 The experiment simulates the hash function H used by $\text{ShowU}_{\text{sr}_1}$ in Figure 4 as a random oracle via lazy sampling and always returns a unique output. In the real execution, the adversary would expect a collision with probability $\mathcal{O}(q_H^2/2^n)$.

We limit the random oracle to unique outputs since a given, honest presentation τ_1 for context ctx_1 along with a collision $H(R_\mu \| \text{ctx}_1) = H(R'_\mu \| \text{ctx}_2)$ in μ allows to forge another valid presentation τ_2 . The new τ_2 is identical to τ_1 , except that the adversary computes a fresh proof

π_{DB} for the witness ctx_2 . Since ctx_2 was only queried to H , but never queried to the secure element while corresponding to an honest user and honest secure element, the forgery is not in any of the sets T_1, T_2, T_3 . The same conditioning excludes collisions in the derivation $\text{ctx} = H(\text{nonce} \| \text{ctx}_0 \| \text{Encode}(\phi))$, which will be used below for type T_3 . After Game 1, the advantage is thus bound by $\Pr[\text{Game } 0] \leq \Pr[\text{Game } 1] + \mathcal{O}(q_H^2/2^n)$.

For any accepting final presentation, straight-line extraction of π_{CD}^* and π_{DB}^* yields witnesses w_{CD}^* and w_{DB}^* . If a shared commitment C_i opens to different values in these two witnesses, $\mathcal{B}_4$ breaks the binding property of Com . Thus, except with probability $\text{Adv}_{\text{Com}}^{\text{Bind}}(\mathcal{B}_4, n) + \text{negl}(n)$, both witnesses encode the same dpk^* , and we condition on this event below.

Game 2 We now consider non-trivial forgeries for type T_1 and abort if such a forgery verifies correctly. We reduce the probability that this happens to the unforgeability of MSig . Again, by straight-line extractability of NIZK_{CD} , except with negligible probability, an accepting proof π_{CD}^* yields a witness w_{CD}^* satisfying $\mathcal{R}_{\text{CD}}$. By the binding event excluded above, the device public key encoded in this witness equals the extracted dpk^* .

Corrupted dpk^* (type T_1). The set of attributes associated to dpk^* is recorded via $(\text{dpk}^*, \mathbf{a}) \in \text{CCreds}$. Since the forgery is non-trivial, $\phi^*(\mathbf{a}) = 0$. Nevertheless, the proof π_{CD}^* verifies with respect to $x^* = (\text{dpk}^*, \text{ctx}^*, \phi^*)$. By straight-line extraction, w_{CD}^* contains a valid signature σ' on some pair $(\text{dpk}^*, \mathbf{a}')$ with $\phi^*(\mathbf{a}') = 1$. Since a non-trivial type T_1 forgery means that there is no tuple $(\text{dpk}^*, \mathbf{a}') \in \text{CCreds}$ for which $\phi^*(\mathbf{a}') = 1$, this pair was never signed during the experiment. Thus we can build a reduction $\mathcal{B}_3$, which obtains signatures from its MSig signing oracle and outputs $((\text{dpk}^*, \mathbf{a}'), \sigma')$ as its forgery.

Honest dpk^* (type T_3). Type T_3 is handled in Game 3 together with type T_2 .

Putting all together, we have, $\Pr[\text{Game } 1] \leq \Pr[\text{Game } 2] + \text{Adv}_{\text{Com}}^{\text{Bind}}(\mathcal{B}_4, n) + \text{Adv}_{\text{MSig}, \ell}^{\text{Unf}}(\mathcal{B}_3, n) + \text{negl}(n)$.

Game 3 We consider non-trivial forgeries of type T_2 and T_3 and abort whenever such a forgery passes verification. We reduce the event that this happens to an adversary $\mathcal{B}_5$ against the SelfTargetMSIS problem.

Using the extraction trapdoor generated by Ext.Setup , we straight-line extract the accepted NIZK_{DB} proof. All oracle queries are answered honestly. Given $(\tau^*, \text{ctx}^*, \phi^*)$ output by $\mathcal{A}$, the experiment obtains $\text{dpk}^* \leftarrow \text{Ext.EKey}(\text{td}, \text{ipk}, \tau^*, \text{ctx}^*, \phi^*)$. By the binding event excluded above, the device public key in the extracted NIZK_{DB} witness is the same dpk^* .

Recall that a non-trivial forgery for type T_2 involves a corrupted user and honest secure element such that $(\text{dpk}^*, \mathbf{a}^*) \in \text{CCreds}$ and $(\text{dpk}^* \notin \text{CSEs})$ (which can be determined from the bookkeeping $(\text{CCreds}, \text{CSEs}, \text{HUsers}, \text{SEKeys}, \text{Granted}, \text{HShown})$).

For any non-trivial type T_2 forgery, the experiment can check if $(\text{dpk}^*, \text{ctx}^*)$ was never queried to ShowSE (since $(\text{dpk}^*, \text{ctx}^*) \notin \text{Granted}$). Hence, the adversary never

obtained a signature corresponding to the target context ctx^* from the honest SE.

For a non-trivial type T_3 forgery, $(\text{dpk}^*, \text{ctx}^*, \phi^*) \notin \text{HShown}$. Since dpk^* belongs to an honest user in type T_3 , the adversary cannot obtain signatures under dpk^* through $\mathcal{O}_{\text{ShowSE}}$; such signatures only come from honest presentation queries recorded in HShown . By construction, the verifier uses the context in its state generated as $\text{ctx} = \text{H}(\text{nonce} \parallel \text{ctx}_0 \parallel \text{Encode}(\phi))$. Since both the user and the SE are honest in type T_3 , signatures under dpk^* are obtained through honest presentations. Under the collision-free random oracle of Game 1, if the target context ctx^* had appeared in an earlier honest presentation, that presentation must have used the same predicate ϕ^* , which would imply $(\text{dpk}^*, \text{ctx}^*, \phi^*) \in \text{HShown}$, a contradiction. Hence the adversary never obtained a signature corresponding to the target context ctx^* from the honest SE in type T_3 either.

By straight-line extractability of NIZK_{DB} , except with negligible probability, an accepting proof π_{DB}^* yields a valid witness w^* such that $(x_{\text{DB}}^*, w^*) \in \mathcal{R}_{\text{DB}}$. By the definition of $\mathcal{R}_{\text{DB}}$, this witness contains values $\mathbf{z}^*, \mathbf{w}^*, \mathbf{h}$, from which we can build a reduction $\mathcal{B}_5$ to the SelfTargetMSIS problem:

From dpk^* we have $(\mathbf{t}_1, \mathbf{A})$ and since $(x_{\text{DB}}^*, w^*) \in \mathcal{R}_{\text{DB}}$ we know that $\mathbf{A}\mathbf{z}^* = \mathbf{w}^* + \text{ct}_1 \cdot 2^d$ holds for $\mathbf{z}^*$ such that:

- $\tilde{c} \leftarrow \text{H}(\mu^*, \text{w1Encode}(\mathbf{w}_1'), \lambda/4)$, where $\text{w1Encode}(\mathbf{w}_1')$ is $\text{UseHint}(\mathbf{h}, \mathbf{w}^*) \equiv \mathbf{A}\mathbf{z} - \text{ct} + u'$ for $u' \leq 4\gamma_2$ following [68, Lem. 1]
- $c \leftarrow \text{SampleInBall}(\tilde{c})$

With the above in mind, by the freshness arguments above, no signature corresponding to the target context ctx^* was generated by the honest SE for dpk^* . Since we excluded any collisions in the random oracle in Game 1, the value of μ^* was computed correctly from ctx^* . Furthermore, since $\text{dpk}^* \notin \text{CSEs}$, $\mathbf{A}, \mathbf{t}$ are samples according to the desired distribution.

Thus, we can follow the same argument as [68, Sec. 4.2.1] used for the security of ML-DSA and guess the device public key extracted during the experiment and embed a SelfTargetMSIS challenge. An adversary who has knowledge of a witness for π_{DB} can find $\mathbf{z}, c, u'$ such that $c = \text{H}([\mathbf{A} \parallel \mathbf{t} \parallel \mathbf{I}_k][\mathbf{z}] - c \parallel u')^T, M := \mu)$, which allows to break SelfTargetMSIS (cf. Definition 9) for $M := \mu$ and $\mathbf{A}, \mathbf{t}$ as extracted from dpk^* . Note that we get an additional security loss of Q_{SE} to guess the oracle call of $\mathcal{O}_{\text{NewSE}}$ that returns the challenge public key.

Consequently, $\Pr[\text{Game 2}] \leq \Pr[\text{Game 3}] + \text{Q}_{\text{SE}} \cdot \text{Adv}_{\text{H}}^{\text{SelfT-MSIS}}(\mathcal{B}_5, n) + \text{negl}(n)$. At the end of Game 3 the experiment aborts for all non-trivial forgeries of the adversary, and thus the experiment always outputs 0: $\Pr[\text{Game 3}] = 0$.

Combining the inequalities from Games 0 to 3 and using $\Pr[\text{Game 3}] = 0$ yields the claimed bound. $\square$

Theorem 2 (Unlinkability).

$$\text{Adv}_{\Pi, \ell, \text{Sim}, \text{Q}_{\text{SE}}}^{\text{Unlink-1}}(\mathcal{A}, n) \leq \text{Adv}_{\text{DB}}^{\text{ParInd}}(\mathcal{B}_1, n) + \text{Adv}_{\text{CD}}^{\text{ParInd}}(\mathcal{B}_2, n) + \text{Adv}_{\pi_{\text{DB}}}^{\text{ZK}}(\mathcal{B}_3, n) + \text{Adv}_{\pi_{\text{CD}}}^{\text{ZK}}(\mathcal{B}_4, n) + \text{Adv}_{\text{wMLWR}}(\mathcal{B}_5)$$

Proof. We consider the experiment $\text{Unlink-1}_{\text{DBAC}, \ell, \text{Sim}}^{\mathcal{A}, b}(1^n)$ (cf. Figure 3) with the adversary $\mathcal{A}$ and our construction from Figure 4. We define our simulator Sim as $(\text{SimProve}, \text{SimSetup})$. Given $(1^n, \ell)$ as input, it sets $\text{pp} \leftarrow \text{SimSetup}(1^n, \ell)$ and runs $(\text{pp}_{\text{CD}}^*, \mathcal{T}_{\text{CD}}) \leftarrow \text{NIZK}_{\text{CD}}.\text{SimSetup}(\text{pp})$ and $(\text{pp}_{\text{DB}}^*, \mathcal{T}_{\text{DB}}) \leftarrow \text{NIZK}_{\text{DB}}.\text{SimSetup}(\text{pp})$. It outputs $\text{par} \leftarrow (\text{pp}_{\text{CD}}^*, \text{pp}_{\text{DB}}^*)$. Since Sim.Setup differs from DBAC.Setup only in replacing the real setup procedures with Sim.Setup for NIZK_{CD} and NIZK_{DB} , the parameter indistinguishability (Definition 3) bounds the advantage of any adversary in distinguishing the two setups by $\text{Adv}_{\text{DB}}^{\text{ParInd}}(\mathcal{B}_1, n) + \text{Adv}_{\text{CD}}^{\text{ParInd}}(\mathcal{B}_2, n)$.

In the Unlinkability game (cf. Figure 3), the adversary's goal is to distinguish whether they are given a simulated presentation τ_0 , or a real presentation τ_1 . We construct a simulator $\mathcal{S}$ that, given $\text{ipk}, \text{ctx}, \phi$, outputs a presentation of the form $\tau_0 = (\{C_i\}_{\text{SCom}}^*, r_{\mathbf{w}'}, r_{\mu}, \pi_{\text{CD}}^*, \pi_{\text{DB}}^*)$, where $r_{\mathbf{w}'} \leftarrow \text{HighBits}_q(\mathcal{U}(R_q^k), 2\gamma_2)$, and r_{μ} is chosen from the appropriate range of H .

We denote the advantage of the adversary to distinguish τ_0, τ_1 as $\text{Adv}_{\Pi, \ell, \text{Sim}, \text{Q}_{\text{SE}}}^{\text{Unlink-1}}(\mathcal{A}, n)$. To show indistinguishability, we modify the real presentation τ_1 over a sequence of hybrid to have the same distribution as τ_0 , such that the adversary is left with a guessing probability of $1/2$.

Game 0 This is the original unlinkability game with advantage $\text{Adv}_{\Pi, \ell, \text{Sim}, \text{Q}_{\text{SE}}}^{\text{Unlink-1}}(\mathcal{A}, n) \leq \text{Adv}_{\text{DB}}^{\text{ParInd}}(\mathcal{B}_1, n) + \text{Adv}_{\text{CD}}^{\text{ParInd}}(\mathcal{B}_2, n) + \Pr[\text{Game 0}]$.

Game 1 We modify the oracle $\mathcal{O}_{\text{DoPres}}$ by simulating the zero-knowledge proofs of τ_1 : Sim makes a call to $\text{SimProve}^*(\text{pp}_{\text{DB}}^*, x_{\text{DB}}, \mathcal{T}_{\text{DB}})$ to get π_{DB}^* , and to $\text{SimProve}_{\text{CD}}(\text{pp}_{\text{CD}}^*, x_{\text{CD}}, \mathcal{T}_{\text{CD}})$ to obtain π_{CD}^* , where $x_{\text{DB}}, x_{\text{CD}}$ are the statements corresponding to the respective relation. Now the values $\{C_i\}_{\text{SCom}}, \mathbf{w}_1, \mu$ in the presentation are independent of the zero-knowledge proofs. Due to the zero-knowledge property of $\mathcal{R}_{\text{DB}}$ and $\mathcal{R}_{\text{CD}}$ the advantage of the adversary to distinguish the simulated presentation from a *real* presentation can be bounded after Game 1 by $\Pr[\text{Game 0}] \leq \text{Adv}_{\pi_{\text{DB}}}^{\text{ZK}}(\mathcal{B}_3, n) + \text{Adv}_{\pi_{\text{CD}}}^{\text{ZK}}(\mathcal{B}_4, n) + \Pr[\text{Game 1}]$.

Game 2 We modify the oracle ShowSE_1 to run the algorithm **Real**($\mathcal{A}, s_1, c$) from Figure 7 to compute $\mathbf{w}_1$ instead of its genuine code. Since this change just modifies the behavior of the oracle, but does not change the output distribution (i.e., the algorithm takes the real public and the oracle's real secret key) this is indistinguishable for the adversary, thus $\Pr[\text{Game 1}] = \Pr[\text{Game 2}]$.

Game 3 The oracle ShowSE is further modified to use the algorithm **Sim** instead of **Real** from Figure 7. Since this change only modifies the value w_1 , which is independent of the other values of the presentation due

to Game 1, the adversary's probability to notice the change is precisely bounded by Lemma 2. Thus we have $\Pr[\text{Game 2}] \leq +\text{Adv}_{\text{wMLWR}}(\mathcal{B}_5) + \Pr[\text{Game 3}]$

Game 4 We modify the oracle $\mathcal{O}_{\text{DoPres}}$ for τ_1 to exchange w_1 for $r_{w'}$ $\leftarrow \text{HighBits}_q(\mathcal{U}(R_q^k), 2\gamma_2)$ and to output r_μ sampled uniformly from the range of H instead of μ . Additionally, the simulator Sim samples a fresh key pair $\text{dsk}^*, \text{dpk}^*$ and computes the commitments $\{C_i^*\}_{S_{\text{com}}}$ to be used in the presentation τ_1 . This change is indistinguishable with $\Pr[\text{Game 3}] = \Pr[\text{Game 4}]$, since the $r_{w'}$ has the same distribution as w_1 from Game 3, and μ (and correspondingly r_μ) is independent of the signature. At the end of Game 4 τ_1 is generate form the same distribution as τ_0 , and thus $\Pr[\text{Game 4}] = 1/2$. This completes the proof. $\square$

Appendix B. ML-DSA Standard

The external `ML-DSA.Sign` interface first formats the message as $M' = \text{BytesToBits}(0x00\|0x00)\|M$ when the ML-DSA context is empty, and then invokes `ML-DSA.Sign_internal` below.

```

ML-DSA.Sign_internal( $sk, M', rnd$ )
( $\rho, K, tr, s_1, s_2, t_0$ )  $\leftarrow \text{skDecode}(sk)$ 
 $\hat{s}_1 \leftarrow \text{NTT}(s_1); \hat{s}_2 \leftarrow \text{NTT}(s_2); \hat{t}_0 \leftarrow \text{NTT}(t_0)$ 
 $\hat{A} \leftarrow \text{ExpandA}(\rho); \mu \leftarrow H(\text{BytesToBits}(tr) \| M', 64)$ 
 $\rho'' \leftarrow H(K \| rnd \| \mu, 64); \kappa \leftarrow 0; (z, h) \leftarrow \perp$ 
while  $(z, h) = \perp$  do
   $y \in R_q^\ell \leftarrow \text{ExpandMask}(\rho'', \kappa)$ 
   $w \leftarrow \text{NTT}^{-1}(\hat{A} \circ \text{NTT}(y)); w_1 \leftarrow \text{HighBits}(w)$ 
   $\tilde{c} \leftarrow H(\mu \| w1\text{Encode}(w_1), \lambda/4)$ 
   $c \in R_q \leftarrow \text{SampleInBall}(\tilde{c}); \hat{c} \leftarrow \text{NTT}(c)$ 
   $\langle\langle cs_1 \rangle\rangle \leftarrow \text{NTT}^{-1}(\hat{c} \circ \hat{s}_1); \langle\langle cs_2 \rangle\rangle \leftarrow \text{NTT}^{-1}(\hat{c} \circ \hat{s}_2)$ 
   $z \leftarrow y + \langle\langle cs_1 \rangle\rangle; r_0 \leftarrow \text{LowBits}(w - \langle\langle cs_2 \rangle\rangle)$ 
  if  $\|z\|_\infty \geq \gamma_1 - \beta \vee \|r_0\|_\infty \geq \gamma_2 - \beta$  then  $(z, h) \leftarrow \perp$ 
  else
     $\langle\langle ct_0 \rangle\rangle \leftarrow \text{NTT}^{-1}(\hat{c} \circ \hat{t}_0)$ 
     $h \leftarrow \text{MakeHint}(-\langle\langle ct_0 \rangle\rangle, w - \langle\langle cs_2 \rangle\rangle + \langle\langle ct_0 \rangle\rangle)$ 
    if  $\|\langle\langle ct_0 \rangle\rangle\|_\infty \geq \gamma_2 \vee \#\{i : h_i = 1\} > \omega$ 
      then  $(z, h) \leftarrow \perp$ 
   $\kappa \leftarrow \kappa + \ell$ 
 $\sigma \leftarrow \text{sigEncode}(\tilde{c}, z \bmod^\pm q, h); \text{return } \sigma$ 

```

Figure 5: ML-DSA signing algorithm (FIPS 204 [62])

Appendix C. Weak MLWR Assumption

```

ML-DSA.KeyGen_internal( $\xi$ ) // Takes seed  $\xi$  as input
( $\rho, \rho', K$ )  $\leftarrow H(\xi \| \text{IntegerToBytes}(k, 1) \| \text{IntegerToBytes}(\ell, 1), 128)$ 
 $\hat{A} \leftarrow \text{ExpandA}(\rho); (s_1, s_2) \leftarrow \text{ExpandS}(\rho')$ 
 $t \leftarrow \text{NTT}^{-1}(\hat{A} \circ \text{NTT}(s_1)) + s_2$ 
 $(t_1, t_0) \leftarrow \text{Power2Round}(t)$ 
 $pk \leftarrow \text{pkEncode}(\rho, t_1); tr \leftarrow H(pk, 64)$ 
 $sk \leftarrow \text{skEncode}(\rho, K, tr, s_1, s_2, t_0); \text{return } (pk, sk)$ 

```

Figure 6: ML-DSA KeyGen algorithm (FIPS 204 [62])

We recall the weak variant of module learning with rounding (MLWR) from [81]. As in the original MLWR, the adversary is asked to distinguish between $\lfloor \mathbf{A}\mathbf{y} \rfloor_p$ and $\lfloor \mathbf{u} \rfloor_p$, for some uniform $\mathbf{A}, \mathbf{u}$ and a secret $\mathbf{y}$. In this variant however, the secret is uniform in some ℓ_∞ -ball and some information about it is provided to the adversary.

Our instance considers an ML-DSA public key $\text{pk} = (\mathbf{A}, \mathbf{t})$, secret key $\text{sk} = \mathbf{s}$ and a challenge c . The MLWR secret $\mathbf{y}$ is conditioned on the fact that setting $\mathbf{w} = \mathbf{A}\mathbf{y}$ will lead to an accepting signature for $(\text{pk}, \text{sk}, c)$. This corresponds to the information that an adversary would get about $\mathbf{y}$ by only seeing $\mathbf{w}_1 = \text{HighBits}_q(\mathbf{w}, 2\gamma_2)$, since it must have been part of an accepting signature. By Lemma 1, $\mathbf{w}_1 = \lfloor \mathbf{w} \rfloor_{(q-1)/2\gamma_2}$ and thus we argue that the hardness of this weak MLWR instance ensures that $\mathbf{w}_1$ carries no exploitable information about the key that was used in the signature.

From an information point of view, the information about $\mathbf{y}$ corresponds to a slight decrease in the secret entropy. Let R the probability that a signature is rejected in ML-DSA, and S the entropy of $\mathbf{y}$. The space of valid secrets is reduced by a factor $1 - R$, hence giving $S + \log(1 - R)$ bits of entropy (or a loss of $\log(1/(1 - R))$). The maximal loss occurs for ML-DSA-65, the rejection rate R is around 80%, giving a loss of roughly 2.3 bits of entropy on the secret key.

For an integer p , we define the rounding operation $\lfloor x \rfloor_p = \lfloor p/q \cdot x \rfloor$, with q the ML-DSA modulus and $\lfloor \cdot \rfloor$ the rounding to nearest integer operator. When applied to a vector or polynomial, we consider the coefficient-wise application of the operator.

Definition 10. Let λ be the security parameter, $k = k(\lambda), l = l(\lambda), d = d(\lambda), q = q(\lambda), p = p(\lambda)$ such that $\mathcal{R} = \mathbb{Z}_q[X]/(X^d + 1)$. For $\gamma \in [0, q/2]$ and $h > 0$, we define the $\text{wMLWR}_{k,l,d,q,p}^{(\gamma,h)}$ problem with auxiliary information aux such that the distribution

$$(\mathbf{A}, \lfloor \mathbf{A}\mathbf{y} \rfloor_p, \text{aux})$$

is computationally indistinguishable from

$$(\mathbf{A}, \lfloor \mathbf{u} \rfloor_p, \text{aux})$$

where $\mathbf{A} \leftarrow \mathcal{U}(\mathcal{R}^{k \times l}), \mathbf{y} \leftarrow \mathcal{U}(\{\mathbf{x} \in \mathcal{R}^l \mid \|\mathbf{x}\|_\infty < \gamma\}), \mathbf{u} \leftarrow \mathcal{U}(\mathcal{R}^k)$ and $H_\infty(\mathbf{y}|\text{aux}) > h$.

We denote the advantage of a distinguisher $\mathcal{A}$ for this problem as $\text{Adv}_{\text{wMLWR}}(\mathcal{A})$, where we omit the parameters when implicit.

The following Lemma from [82] ensures that HighBits_q maps to a valid rounding for MLWR.

Lemma 1 (Lemma 5 from [82]). *Let q, α be integers satisfying $q > 2\alpha$, $q-1 \equiv 0 \pmod{\alpha}$ and $\alpha \equiv 0 \pmod{2}$. We have $\forall r \in [0, q-1]$:*

$$\begin{aligned} \text{HighBits}_q(r, \alpha) &= \left\lfloor \frac{r-1}{\alpha} \right\rfloor \bmod \frac{q-1}{\alpha} \\ &= \lfloor r \rfloor_{\frac{q-1}{\alpha}} \end{aligned}$$

Lemma 2. *Let $n = 256$ and $(k, l, q, \gamma_1, \gamma_2)$ be the corresponding parameters from ML-DSA- $\{44, 65, 87\}$. Let $\mathcal{A}$ an adversary for the distinguishing game between outputs of **Real** and **Sim** from Figure 7. Then there exists an adversary $\mathcal{B}$ for the $\text{wMLWR}_{k, l, n, q, p}^{(\gamma_1, h)}$ game such that:*

$$\begin{aligned} &|\Pr[\mathcal{A}(\text{pk}, s_1, c, \text{Real}(\text{pk}, c)) = 1] \\ &\quad - \Pr[\mathcal{A}(\text{pk}, s_1, c, \text{Sim}()) = 1]| \leq \text{Adv}_{\text{wMLWR}}(\mathcal{B}) \end{aligned}$$

with $p = (q-1)/2\gamma_2$, $\text{pk} = (\mathbf{A}, \mathbf{t})$ an ML-DSA public key, $\text{sk} = s_1$ the corresponding secret key and $h = \ln \log(2\gamma_1) - \log(1-R)$ with R the probability to reject a signature in ML-DSA.

Proof. The proof relies on the fact that the vector $\mathbf{y}$ can be seen as sampled conditioned on the fact that it will lead to a valid ML-DSA signature. This restricts the set of possible $\mathbf{y}$ for the MLWR instance, losing $\log(1/(1-R))$ bits of entropy, corresponding to the rejection rate.

We prove this by introducing the hybrid in Figure 7.

First observe that **Hybrid** is just a rewrite of **Real**, the rejection conditions are exactly the same and are computed with respect to the same values. Any adversary has thus an advantage 0 in distinguishing them.

In **Hybrid**, the $\mathbf{y}$ is explicitly sampled such that it produces a valid instance for the key pk . Let the auxiliary information aux consist of $(\text{pk}, s_1, c, P(\mathbf{y}) = 1)$ for P the logical proposition indicating that $\mathbf{y}$ leads to a valid ML-DSA signature for key pk and challenge c . We now show that indistinguishability of **Hybrid** and **Sim** relies on the weak (leaky) MLWR with auxiliary information aux . Let $\mathcal{B}$ be an adversary for the weak (leaky) MLWR instance $\text{wMLWR}_{k, l, n, q, p}^{(\gamma_1, h)}$. Adversary $\mathcal{A}$ receives $\text{pk}, s_1, c, \mathbf{w}_1$ where $\mathbf{w}_1$ is either the output of **Hybrid**(pk, s_1, c) or **Sim**(). Adversary $\mathcal{A}$ forwards $(\mathbf{A}, \mathbf{w}_1, \text{aux})$ to $\mathcal{B}$ and returns its answer.

By Lemma 1 applied with $\alpha = 2\gamma_2$, $\mathbf{w}_1 = \text{HighBits}_q(\mathbf{A}\mathbf{y}) = \lfloor \mathbf{A}\mathbf{y} \rfloor_p$ (such that $\mathbf{y}$ leads to a valid ML-DSA signature) if $\mathcal{A}$ received a sample from **Hybrid**, and $\mathbf{w}_1 = \text{HighBits}_q(\mathbf{u}) = \lfloor \mathbf{u} \rfloor_p$ for a uniform $\mathbf{u}$ if $\mathcal{A}$ received a sample from **Sim**. The inequality follows. $\square$

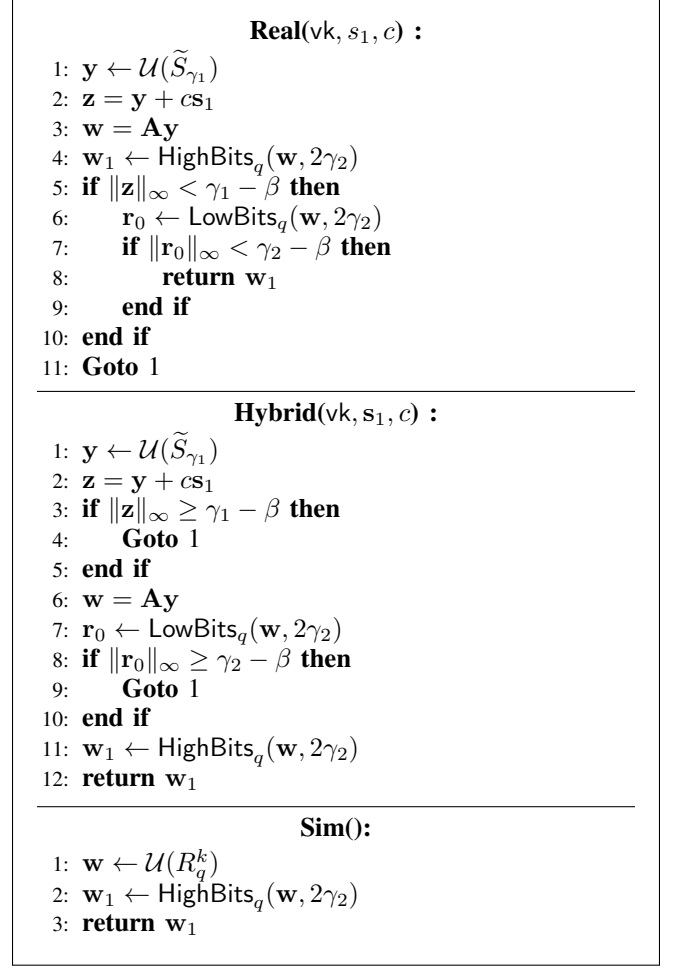


Figure 7: Hybrid for Lemma 2.

C.1. On the hardness of weak MLWR

The learning with rounding assumption was introduced by Banerjee et al. in [81]. It was first proven to be as hard as LWE in the super-polynomial modulus-to-noise ratio. In [83], Alwen et al. show that the reduction also holds in the polynomial modulus-to-noise ratio, confirming the link between LWR and LWE. Later on, an analysis of the ring and module variants was carried by Liu et al. in [84] in which they show that the weak Module-LWR problem can be shown to be hard under some high-entropy condition.

As for all concrete and efficient constructions, the parameters for our instantiation of the problem do not fit with the theoretical reduction, but a recent work of Belaid et al. [82] introduces a variant of MLWR with hints that they apply to the same $\mathbf{w}_1$ from ML-DSA and carry a short cryptanalytic analysis to evaluate the concrete hardness of the problem. In their variant, together with $\mathbf{w}_1 = \text{HighBits}_q(\mathbf{A}\mathbf{y})$, they reveal the values of $\|\mathbf{y} + cs_1\|_\infty$ and $\|\text{LowBits}_q(\mathbf{w}, 2\gamma_2)\|_\infty$, hiding only their positions in their respective vectors. This is arguably much more information than what is given in our case, since we only reveal to the

adversary that no values of those vectors cross the rejection threshold.

They show that, without any hints or leaks on $\mathbf{y}$ for $\mathbf{w}_1 = \text{HighBits}_q(\mathbf{A}\mathbf{y}, 2\gamma_2)$, the corresponding MLWR instance can be viewed as an MSIS instance for which the parameters of ML-DSA-65 already offer at least 256 bits of security, due to the aggressive rounding of 17 bits out of 27. They then conclude that the hints/leaks marginally reduce the dimension of the lattice in which to search for an MSIS solution to break the MLWR instance.

We rely on their analysis to assert the hardness of our weak MLWR instance, underlining that we both give out less information and lose a marginal amount (at most 2.3 bits) of entropy on the secret.

Appendix D. On FIPS verification equivalence

Our verification relation form [Section 3](#) does not syntactically execute the exact FIPS 204 verification algorithm. In particular, it proves $\mu = H(R_\mu \| \text{ctx}, 64)$ without enforcing in-circuit that $R_\mu = \text{BytesToBits}(H(\text{dpk}, 64)) \| \text{BytesToBits}(0x00 \| 0x00) \| R$. We show that this optimization does not give a corrupt user any additional ability to produce accepting presentations when the SE is honest and FIPS-compliant.

We consider the same corruption setting as type T_2 in the DBAC model: the user is corrupt, the SE is honest.⁷ We also recall that the credential is bound to a device public key generated according to the setup assumptions of [Section 3](#). Next, we show that an accepting presentation for which the extracted device signature is not consistent with FIPS 204 implies a solution to SelfTargetMSIS with overwhelming probability.

Definition 11 (FIPS 204 Equivalence). *We call a device binding protocol FIPS 204 equivalent if for all PPT adversaries $\mathcal{A}$ the following advantage is negligible in n*

$$\text{Adv}_{\Pi}^{\text{FIPSEq}}(\mathcal{A}, n) := \Pr[\text{FIPSEq}_{\Pi}^{\mathcal{A}}(n) = 1].$$

where $\text{FIPSEq}_{\Pi}^{\mathcal{A}}(n)$ is the game defined in [Figure 8](#).

The goal in the FIPS equivalence game for the adversary is to come up with a valid presentation for a non-corrupt SE such that the triple $(\mathbf{z}, \mathbf{h}, \tilde{c})$ from the extracted witnesses do not verify under FIPS 204. Line 9 & 10 ensure that we are considering only type T_2 setting. Line 11-15 compute the ML-DSA signature extracted from the presentation τ^* .

In [Theorem 3](#) we show that our device binding protocol for ML-DSA from [Figure 4](#) is *FIPS 204 equivalent*, as long as the SelfT-MSIS remains computationally hard.

7. If the SE is corrupt, the adversary knows dsk and can bypass the SE. Thus, it can craft a signature for a presentation that verifies correctly but which is not necessarily FIPS-conform. However, this is of no use to the adversary since any such forgery would be considered trivial for type T_1 .

Game $\text{FIPSEq}_{\Pi, \ell, \text{Ext}}^{\mathcal{A}}(n)$
1 : $(\mathcal{T}, \text{par}) \leftarrow \text{Ext.Setup}(1^n, \ell)$
2 : $\text{CSEs}, \text{HUsers}, \text{CCreds}, \text{Granted}, \text{SEKeys}, \text{HCreds} \leftarrow \emptyset$
3 : $(\text{isk}, \text{ipk}) \leftarrow \text{IssKGen}(\text{par})$
4 : $(\text{vst}^*, \tau^*, \text{ctx}^*, \phi^*) \leftarrow \mathcal{A}^{\mathcal{O}^*}(\text{par}, \text{ipk})$
5 : parse ϕ^* as $(\mathcal{R}_{\text{CD}}, \mathcal{R}_{\text{DB}}, S_{\text{Rev}}, S_{\text{Com}}, S_{\text{DB}})$
6 : parse τ^* as $(\{C_i^*\}_{S_{\text{Com}}}, \mathbf{w}_1^*, \mu^*, \pi_{\text{CD}}^*, \pi_{\text{DB}}^*)$
7 : $\text{dpk}^* \leftarrow \text{Ext.EKey}(\mathcal{T}, \text{ipk}, \text{vst}^*, \tau^*, \text{ctx}^*, \phi^*)$
8 : if $\text{ShowVf}_2(\text{vst}^*, \text{ipk}, \phi^*, \tau^*, \text{ctx}^*) \neq 1$ return 0
9 : if $\text{dpk}^* \notin \text{SEKeys} \vee \text{dpk}^* \in \text{CSEs} \vee \text{dpk}^* \in \text{HUsers}$ return 0
10 : if $\nexists \mathbf{a} : (\text{dpk}^*, \mathbf{a}) \in \text{CCreds} \wedge \phi^*(\mathbf{a}) = 1$ return 0
11 : $\tilde{c}' \leftarrow H(\mu^* \ \text{w1Encode}(\mathbf{w}_1^*), \lambda/4)$; $c' \leftarrow \text{SampleInBall}(\tilde{c}')$
12 : $x_{\text{DB}}^* \leftarrow (S_{\text{DB}}, \{C_i^*\}_{i \in S_{\text{DB}}}, c', \mathbf{w}_1^*, \mu^*, \text{ctx}^*)$
13 : $(\mathbf{z}^*, \mathbf{w}^*, h^*, \text{dpk}^*, o_{i \in S_{\text{DB}}}^*, R_\mu^*) \leftarrow \text{NIZK}_{\text{DB}}.\text{Ext}(\mathcal{T}_{\text{DB}}, x_{\text{DB}}^*, \pi_{\text{DB}}^*)$
14 : $M^* \leftarrow R_\mu^* \ \text{ctx}^*$; $M'^* \leftarrow \text{BytesToBits}(0x00 \ 0x00) \ M^*$
15 : $\sigma^* \leftarrow (\tilde{c}', \mathbf{z}^*, h^*)$
16 : if $\text{ML-DSA.Verify_internal}(\text{dpk}^*, M'^*, \sigma^*) = 0$ return 1
17 : return 0

Figure 8: FIPS 204 verification-equivalence game. The oracle set $\mathcal{O}^*$ and bookkeeping are as in the unforgeability game of [Figure 2](#). We highlight in *gray* the bookkeeping/unforgeability related parts for readability.

Theorem 3.

$$\text{Adv}_{\Pi}^{\text{FIPSEq}}(\mathcal{A}, n) \leq q_{\text{H}} q_{\text{newSE}} \left(\frac{q_{\text{H}}}{2^n} + \text{Adv}_{\text{H}}^{\text{SelfT-MSIS}}(\mathcal{B}_2, n) \right),$$

where $q_{\text{H}}, q_{\text{newSE}}$ are a bound on the number of queries to the random oracle H and the oracle NewSE respectively.

Proof. We bound the advantage $\text{Adv}_{\Pi}^{\text{FIPSEq}}(\mathcal{A}, n)$ by a reduction to the SelfT-MSIS problem by a sequence of game hops. Consider a SelfT-MSIS challenger $\mathcal{C}$ that provides a reduction $\mathcal{B}$ with a matrix $\mathbf{A}$ and access to a random oracle H_2 . $\mathcal{C}$ simulates the FIPSEq game for an adversary.

Game 0 This is the unmodified game.

$$\Pr[\text{Game 0}] = \Pr[\text{FIPSEq}_{\Pi}^{\mathcal{A}} = 1]$$

Game 1 The reduction $\mathcal{B}$ guesses the query to H used by the adversary used to derive μ^* for τ^* . Additionally, it makes a guess for the query to ShowSE the dpk of which is extracted by the experiment later. The reduction checks if μ^* and dpk^* match the guesses and abort otherwise, which happens with probability $(q_{\text{H}} q_{\text{newSE}})^{-1}$. We thus have

$$\Pr[\text{Game 0}] = q_{\text{H}} q_{\text{newSE}} \Pr[\text{Game 1}].$$

Game 2 In this game we modify the ShowSE oracle to abort, whenever it receives a query that results in μ^* as the signed message for dpk^* . This has no impact on the probability that the game outputs 1, because any such query would result in an extracted tuple $(\mathbf{z}, \mathbf{h}, c', M^*)$ such that $\text{ML-DSA.Verify_internal}(\text{dpk}^*, M'^*, \sigma^*) =$

$(\tilde{c}, \mathbf{z}^*, h^*) = 1$. Indeed, the proof π_{DB}^* ensures that all the equations for ML-DSA.Verify_internal are verified by the signature, except for the hash evaluation of μ^* . Thus, if the adversary makes an SE query on message M^* and key dpk^* that results in μ^* as the signed message, the hash is honestly computed by the SE and establishes that $\mu^* = \text{H}(\text{BytesToBits}(\text{H}(\text{dpk}^*, 64)) \parallel \text{BytesToBits}(0x00 \parallel 0x00) \parallel M^*)$ yielding a valid ML-DSA signature. In such a case the experiment outputs 0, and the adversary loses. Consequently, after this game, the signatures output by ShowSE never corresponds to μ^* as an output of H. This is indistinguishable, because an abort corresponds to a losing event anyway:

$$\Pr[\text{Game 1}] = \Pr[\text{Game 2}]$$

Game 3 The game is further modified to forward the query to H that was guessed in Game 1 to the oracle H_2 of challenger $\mathcal{C}$. Further, all subsequent queries including μ^* are forwarded to the challenger.

For all other queries the reduction maintains their own random oracle through lazy sampling. This is only noticeable if H_2 returns the the same value assigned by lazy sampling earlier, which can be bound by:

$$\Pr[\text{Game 2}] = \frac{q_H}{2^n} + \Pr[\text{Game 3}]$$

Game 4 The game is modified to provide a dpk derived from the challenge matrix $\mathbf{A}$ by embedding the public key into the oracle call of NewSE guessed in Game 1. Whenever the adversary $\mathcal{A}$ queries the oracle ShowSE on values $(\text{dsk}, \text{ipk}, \text{umsg}_i, \text{ctx}_i)$, the reduction simulates the signing oracle ShowSE by using the simulator from ML-DSA as in [85, App. B]. The simulation is indistinguishable since the reduction can choose the challenge in advance, program the random oracle H, and thus choose a challenge value as if it was performing lazy sampling.

The only queries to ShowSE that cannot be simulated are exactly those, that result in μ^* as a signing message, because for these the simulator cannot choose and program the random oracle. However, these are exactly the ShowSE queries for which the experiment aborts from Game 2 on. Therefore,

$$\Pr[\text{Game 3}] = \Pr[\text{Game 4}].$$

After Game 4, upon receiving the presentation τ^* , the reduction can now extract $(\mathbf{z}^*, h^*, M^*) \leftarrow \text{NIZK}_{\text{DB}}.\text{Ext}(\pi_{\text{DB}})$ and can find $c, \mathbf{u}'$ such that $c = \text{H}([\mathbf{A} \parallel \mathbf{t} \parallel \mathbf{I}_k][\mathbf{z}^* \parallel -c^* \parallel \mathbf{u}']^T, M := \mu^*)$, for $u' \leq 4\gamma_2$ following [68, Lem. 1], which $\mathcal{B}$ can output to break SelfTargetMSIS (cf. Definition 9).

$$\Pr[\text{Game 4}] \leq \text{Adv}_{\text{H}}^{\text{SelfT-MSIS}}(\mathcal{B}_2, n)$$

This completes the proof. $\square$

Appendix E.

Declaration of Competing Interests

Calvin Abou Haidar, Octavio Perez-Kempner and Marcel Tiepelt are employees of NTT, Inc. and are named inventors on a patent application filed by NTT, Inc., entitled “Anonymous Proof of Possession of an ML-DSA Signing Key” (Ref. PCT/JP2026/036794, filed October 13, 2026, with the Japan Patent Office (JPO) as receiving Office under the PCT; status: pending). The application covers the construction in Section 3 and its use for device binding in anonymous credentials (Section 4).

Appendix F.

Ethics Considerations

This work presents a cryptographic protocol to support privacy-preserving use cases for digital identity. It is foundational work in applied cryptography: we conducted no human-subjects experiments, collected no personal or identifiable data, and did not interact with any live or third-party systems; our proof-of-concept evaluation only uses synthetic inputs, standard cryptographic parameters, and open-source libraries used in accordance with their licenses. Based on this, we understand that our work do not raises human-subjects, PII, or live-systems concerns. As with any privacy-enhancing technology, the anonymity (by means of our unlinkability notion) our protocol provides could in principle reduce accountability or aid malicious actors, but this depends on its use alongside other protocols built on top of ours (i.e., the anonymous credential scheme used). Such situation is inherent to anonymous credentials rather than specific to our construction. We note, however, that such risks can be mitigated by established mechanisms such as revocation, auditability or rate-limiting features. All of which are compatible with our building blocks. Following the Menlo Report’s principle of Beneficence, we judge that the benefits (stronger privacy for honest users and more efficient, auditable primitives) outweigh these risks, and we release our assumptions, proofs, and implementation openly to enable review and responsible use.

Appendix G.

Generative AI Usage Considerations

Generative AI was used for editorial purposes, and all outputs were inspected by the authors. Specifically, the authors used Generative AI (OpenAI’s ChatGPT, Google’s Gemini and Anthropic’s Claude) to revise the text, improve flow, and correct typographical errors, grammatical errors, and awkward phrasing. Generative AI was not used for technical parts of the work. Additionally, the authors used Claude as an AI-assisted development tool during the implementation and documentation of this project, to help produce code and accompanying explanatory text. All AI-generated code and related text were reviewed, tested, and validated by the authors, who retain full responsibility for the entire

contents of this submission. The use of generative AI did not introduce reproducibility limitations, as the affected artifacts (manuscript text and source code) are fully inspectable and under the authors' control.