

# Context-Blind Device-Bound Anonymous Credentials from Signatures and Proofs

Mark Manulis<sup>1</sup> , Alan Pulval-Dady<sup>1</sup> , Daniel Slamanig<sup>1</sup> , and Dave Wittig<sup>2</sup>

<sup>1</sup> Research Institute CODE, Universität der Bundeswehr München, Germany  
{alan.pulval-dady, mark.manulis, daniel.slamanig}@unibw.de

<sup>2</sup> Universität der Bundeswehr München, Germany  
davewittig@t-online.de

**Abstract.** Anonymous credentials enable privacy-preserving authentication, but their transferability motivates binding each credential to a key protected by a secure element (SE). Such binding should preserve privacy even when the SE is remote or corrupted: every presentation must require fresh SE participation without revealing its context to the SE (Friedrichs et al., EC’26). Existing so called context-blind device-bound constructions rely on classical assumptions, whereas existing plausibly post-quantum anonymous credentials do not provide this functionality. We give a modular construction of context-blind device-bound anonymous credentials from a position-binding vector commitment, a signature scheme, a round-optimal blind signature, and a non-interactive zero-knowledge argument. We prove correctness, one-more unforgeability, unlinkability against a colluding issuer, verifier, and fully corrupted SE, and SE-obliviousness. We instantiate the construction with the CAPSS framework (Feneuil and Rivain, TCHES’26), Griffin/Jive Merkle commitments, and proofs of the complete presentation relation. For 16 attributes and four disclosures, using the Liger-based proof backend from Longfellow (Frigo and Shelat, CiC’26) yields a presentation taking 2.60 s on the user side and 0.29 s to verify, with a 506.11 KB proof. We also evaluate an Aurora backend for the same relation. We complement these measurements with published results to explore the proof-size/proving-time trade-offs offered by alternative transparent, plausibly post-quantum proof systems for relations of comparable size. To the best of our knowledge, this is the first concrete construction and evaluation of context-blind device-bound anonymous credentials assembled entirely from plausibly post-quantum components.

**Keywords:** Anonymous credentials · Device binding · Post-Quantum Cryptography

## 1 Introduction

Anonymous credentials, introduced by Chaum [Cha90] and subsequently developed into practical signature-based systems [LRSW99, CL01], allow a user to obtain a credential from an issuer and later demonstrate possession of certified attributes without revealing their identity or unnecessary information. Since their introduction, they have become a central primitive for privacy-preserving authentication and digital identity systems. Most practical signature-based systems, however, inherit the quantum vulnerability of their underlying RSA, discrete-logarithm, or pairing assumptions.

Building anonymous credentials from plausibly post-quantum components is therefore an important part of the transition toward long-lived privacy-preserving identities, e.g., when adopting anonymous credential technology within the European Digital Identity (EUID) framework [BBC<sup>+</sup>24].

Recent work has made substantial progress in this direction. Lattice-based constructions provide plausibly post-quantum anonymous credentials, with concrete implementations and evaluations among the more recent results [JRS23, BLNS23b, LCL<sup>+</sup>23, AGJ<sup>+</sup>24]. A complementary signature-and-proof approach proves verification of a signature inside a general-purpose zero-knowledge proof. For traditional credentials (i.e., ECDSA signatures) this approach is followed by what is called the Longfellow system<sup>3</sup> by Frigo and Shelat [Fs26]. This approach is also explored for unlinkable tokens by Policharla et al. [PFW23] and for anonymous credentials in CAPSS [FR26a], where latter works establish complementary routes to plausibly post-quantum anonymous credentials. But they do neither (formally) consider device-binding nor do they address the additional trust and privacy issues that come up when a credential must be bound to a device.

<sup>3</sup> <https://blog.google/innovation-and-ai/technology/safety-security/opening-up-zero-knowledge-proof-technology-to-promote-privacy-in-age-assurance/>

An inherent issue of anonymous credentials is that due to their anonymity feature, they can easily be cloned and thus credentials can easily be shared with other users. This is undesirable for credentials intended to remain under the control of their legitimate holder. Device-bound anonymous credentials address this problem by binding a credential to a secret key held by a secure element (SE). A valid presentation must then contain evidence of fresh participation by the device holding that key. This approach is relevant not only to local hardware such as trusted platform modules and embedded secure elements, but also to remote or cloud-based SEs that provide key protection to constrained clients.

Device binding introduces a privacy problem of its own. If the SE receives the presentation context in the clear, it may learn where, when, or for which service a credential is being used. This information can allow a malicious SE to trace an otherwise unlinkable credential. Friedrichs et al. [FHLL26] formalize this tension through a framework for device-bound anonymous credentials with and without trusted hardware. Their framework distinguishes context-aware and context-blind constructions and gives security notions capturing different corruption patterns. Its strongest unlinkability notion, Unlink-3, considers a fully corrupted SE over which the adversary has a live view and control during the presentation protocol. SE-obliviousness additionally protects the presentation context from the SE. We refer to these context-blind schemes as blind device-bound anonymous credentials (BDBAC). Our focus is the context-blind setting combining these guarantees: presentation privacy at the Unlink-3 level, together with context privacy against the SE.

Friedrichs et al. [FHLL26] realize their framework through constructions tailored to specific algebraic building blocks. Their security analysis also highlights why a generic signature-and-proof realization is not immediate. In the unforgeability game, the device public key remains hidden inside the final credential presentation but must nevertheless be recovered by the reduction. It is therefore not enough for one isolated component to admit extraction: the extraction guarantee must remain available after that component has been combined with the complete credential proof. Their concrete constructions resolve this issue within their particular algebraic structure, but do not yield a generic theorem for modular signature-and-proof components in this setting. Thus, these two lines of work do not yet meet: to our knowledge, no existing post-quantum anonymous credential scheme supports context-blind device binding against a potentially malicious SE.

Recent SNARK-friendly signatures make *signature-and-proof* approach to construct post-quantum anonymous credentials increasingly plausible. In particular, CAPSS [FR26a] is built from arithmetization-oriented symmetric primitives and is designed so that signature verification admits an efficient arithmetic representation. Together with transparent post-quantum proof systems, such signatures open a complementary route to specialized lattice-based anonymous credentials and diversify the assumptions available for post-quantum credentials. The open question is whether this modular route can realize context-blind device binding while preserving this strongest unlinkability guarantee.

*Technical challenge.* A natural way to combine post-quantum anonymous credentials with context-blind device binding is to place the issuer credential, the device authorization, and their consistency relations inside the witness of a general-purpose NIZK. At the protocol level, this composition appears simple: one proof can assert possession of an issuer credential and a fresh device authorization under the same hidden device key. Establishing its security, however, is not immediate. In particular, a valid BDBAC forgery does not necessarily constitute a forgery of the issuer signature, since an adversary may reuse a legitimately issued credential inside a new presentation. Moreover, the credential and the device public key remain hidden inside the presentation, although the security reduction must recover and classify them.

This raises three challenges: (1) *compose modular and plausibly post-quantum components while binding the credential and the fresh device authorization to the same hidden key and preserving context blindness*; (2) *bridge the gap between BDBAC unforgeability and the unforgeability notions of the underlying signatures, despite credential reuse and hidden presentation witnesses*; and (3) *determine whether proving the complete presentation relation remains practical and quantify the concrete overhead introduced by context-blind device binding*. These challenges lead to the following central question:

*Can a generic signature-and-proof approach provide a practical plausibly post-quantum realization of context-blind device-bound anonymous credentials with unlinkability even against a fully corrupted SE?*

## 1.1 Our Contributions

We answer this question positively through a generic construction, a security analysis, and a feasibility prototype implementation.

*Generic construction.* Following the context-blind model of Friedrichs et al. [FHLL26], we specify the issuance, disclosure, and presentation interfaces, and give a generic construction from a position-binding vector commitment, an issuer signature, an auxiliary signature, a round-optimal blind signature, and a NIZK. The issuer certifies the pair consisting of the attribute commitment and the device public key. The blind-signing protocol authorizes a fresh presentation context without revealing that context to the SE. The presentation proof connects these components through one hidden device key. Our contribution is a generic construction and security proof for this setting at the Unlink-3 level. Generality concerns the choice of cryptographic components satisfying the stated assumptions; the security notions and their hierarchy are inherited from Friedrichs et al.

*Security analysis.* We prove correctness, one-more unforgeability, unlinkability against a colluding issuer, verifier, and fully corrupted SE (Unlink-3), and SE-obliviousness. The reductions expose the precise role of each component. Position binding protects the disclosed attributes, issuer-signature unforgeability protects the credential, one-more unforgeability of the blind signature enforces fresh device participation, and blindness hides the presentation context from the SE. The NIZK links all these relations while hiding the credential, the attribute commitment, and the device public key. Its presentation relation is extended to an OR-relation containing an auxiliary signature branch: Honest presentations use the credential branch; in the unforgeability reduction, the auxiliary branch provides a second valid witness. Witness indistinguishability first hides the switch to this auxiliary witness; the reduction then omits the corresponding hidden credentials and their signing queries. Every proof remains honestly generated under the honest setup.

*Proof-system requirements.* The unforgeability theorem requires witness indistinguishability for the change of witness and straight-line knowledge soundness for the final extraction. Zero knowledge is used separately for unlinkability. We evaluate the presentation relation with Aurora [BCR<sup>+</sup>19] and the Ligero-based proof backend from Longfellow [Fs26].

*Post-quantum instantiation and evaluation.* We instantiate the issuer, device, and auxiliary signatures with CAPSS, use a Griffin [GHR<sup>+</sup>23]/Jive [BBC<sup>+</sup>23] Merkle tree as the attribute vector commitment, and prove the complete presentation relation with Aurora and the Longfellow proof backend. We measure issuance, presentation generation, verification, and communication for both backends on the same circuit, for credentials containing 16 attributes with four attributes disclosed. This reflects a mid-sized identity credential and a realistic selective-disclosure setting, consistent with wallet use cases for identity credentials. To the best of our knowledge, this is the first implementation and concrete evaluation of a context-blind device-bound anonymous credential assembled entirely from plausibly post-quantum components. With the Longfellow proof backend, measured presentation generation takes 2.60 seconds on the user side, with a 506.11 KB proof. Beyond these local measurements, we explore the design space of transparent, plausibly post-quantum proof systems using published results for R1CS relations of comparable size. This comparison illustrates how alternative systems may favor either lower proving time or smaller proofs.

## 1.2 Related Work

*Anonymous credentials.* The modern signature-based approach to anonymous credentials originates in pseudonym systems and efficient non-transferable credentials [LRSW99, CL01]. A credential is typically a signature on an attribute vector, and a presentation proves knowledge of a valid signature while hiding the credential and any attributes not needed by the verifier.

*Device-bound anonymous credentials.* Friedrichs et al. [FHLL26] introduce the BDBAC framework, formalize context blindness and several unlinkability levels, and give constructions meeting these definitions. We adopt their framework and investigate a modular signature-and-proof realization of its context-blind setting with Unlink-3 and SE-obliviousness, using plausibly post-quantum components.

*Post-quantum credentials from lattices.* Jeudy et al. [JRS23] obtain the first lattice-based anonymous credentials from a signature with efficient protocols; Argo et al. [AGJ<sup>+</sup>24] subsequently improve, implement, and evaluate this approach. Bootle et al. [BLNS23b] give a different lattice framework with compact presentation proofs and concrete estimates. Lai et al. [LCL<sup>+</sup>23] combine commit-transferrable signatures with a straight-line extractable proof that links hidden signed messages to the presentation commitment. LaZer [LSS24] provides an implementation framework for lattice-based proofs and privacy protocols, including anonymous credentials. These works offer different constructions and levels of implementation and evaluation, but do not address a context-blind interaction with a potentially malicious SE.

*Signatures and general-purpose proofs.* Policharla et al. [PFW23] prove verification of their STARK-friendly zkDilithium signature to obtain post-quantum Privacy Pass tokens within a generic anonymous-credential approach. Loquat [ZSE<sup>+</sup>24] and CAPSS [FR26a] design post-quantum signatures for efficient verification inside a general proof system. CAPSS demonstrates an anonymous credential by proving signature verification and an arithmetic predicate with Aurora, obtaining a 140 KB proof in its reported setting. This is the closest baseline for our implementation: our presentation also proves device authorization under the device public key included in the issuer-signed credential. SmallWood [FR26b] underlies the CAPSS signatures used here. Flock [BRW26] provides a complementary indication of where the signature-and-proof approach could develop: it efficiently proves batches of Keccak and BLAKE3 computations and identifies hash-based signature verification as a possible extension. It does not, by itself, establish a hash-based signature-and-proof instantiation satisfying our framework’s proof-system requirements.

*Credentials from existing identity infrastructure.* Frigo and shelat [Fs26] construct anonymous credentials from existing ECDSA credentials using a proof system based on sumcheck and Ligerio [AHIV17]. Their Longfellow implementation targets the ISO MDOC presentation flow and proves knowledge of both a valid issuer signature and a device signature, without requiring changes to existing issuer processes or mobile devices. It therefore provides a signature-and-proof approach to device binding with a plausibly post-quantum presentation proof, although the underlying credential and device authentication remain based on ECDSA. Unlike our context-blind setting, the device signs the presentation transcript and therefore learns the context it authorizes.

*Direct anonymous attestation.* DAA combines a host with a protected device for privacy-preserving platform authentication [BCC04]. Post-quantum DAA constructions use hash-based techniques [CDE<sup>+</sup>23] or lattice-based collaborative proofs [CHE24]. This is related to device-bound credentials, but DAA attests a platform rather than authorizing presentations of issuer-certified user attributes without revealing their context to the SE.

### 1.3 Technical Overview

The starting point is to connect two assurances within a single presentation: the issuer has certified the user’s attributes for a particular device, and that device has authorized the current presentation context. The device key provides the link between them, while the proof keeps this key and the supporting signatures hidden. We first describe the relation linking the credential and device authorization, then explain how an auxiliary branch enables the unforgeability reduction.

*Credential issuance.* To bind certified attributes to a device, the issuer commits to the attribute vector  $\text{Attrs} = (\text{Attrs}_1, \dots, \text{Attrs}_\ell)$  in a vector commitment  $\text{cmt}$ . The device secret key remains in the SE. The issuer then signs the commitment together with the device public key  $\text{dpk}$ :

$$\text{cred} \leftarrow \text{Sig}_I.\text{Sign}((\text{cmt}, \text{dpk}), \text{isk}).$$

It returns the credential and the commitment state needed to derive selective openings. This credential can subsequently be used to demonstrate possession of certified attributes while hiding both the commitment and the device key. Only position binding is required of this commitment; its hiding is not needed because the root is kept inside the presentation witness.

*Context-blind presentation.* At presentation time, the device must contribute without learning the verifier’s context  $\text{ctx}$ . The user therefore obtains a blind signature  $\text{bsig}$  on  $\text{ctx}$  from the SE. For disclosed positions  $I$  and values  $\mathbf{d} = (d_i)_{i \in I}$ , the public statement is  $x = (\text{ipk}, \text{ctx}, I, \mathbf{d})$ . The credential relation  $\mathcal{R}_{\text{pres}}$  checks three

conditions: the issuer signature authenticates  $(\text{cmt}, \text{dpk})$ ; the blind signature authenticates  $\text{ctx}$  under that same  $\text{dpk}$ ; and each disclosed value  $d_i$  has a valid opening at position  $i$  of  $\text{cmt}$ . The root, device key, signatures, and openings remain in the witness. Zero knowledge hides this credential material from the verifier, while blindness protects the context during the interaction with the SE.

*An auxiliary witness for the reduction.* The difficulty arises when turning an accepting presentation into a forgery of an underlying signature. Extracting a credential witness is insufficient on its own: a new presentation may contain a credential that the reduction has already obtained from its issuer-signing oracle. To avoid those signing queries for selected honest users, the reduction needs another way to produce their presentations.

We use the established idea of extending a proof language with a trapdoor branch, as in the simulation-sound NIZK literature [Sah01, DDO<sup>+</sup>01, Gro06]. Here setup generates a key pair  $(\text{sk}_A, \text{vk}_A)$  for an auxiliary signature scheme  $\text{Sig}_A$ , independently of the issuer key, publishes  $\text{vk}_A$ , and erases  $\text{sk}_A$  in the real protocol. The presentation proves knowledge of a witness for

$$\underbrace{\text{issuer credential and device authorization}}_{\mathcal{R}_{\text{pres}}} \vee \underbrace{\text{Sig}_A \text{ signature on } (x, \text{dpk})}_{\text{auxiliary branch}}.$$

Both branches are part of the relation proved by every presentation; honest users use the credential branch. Section 4.1 gives the complete relation and its encoding.

In the reduction, retaining  $\text{sk}_A$  allows a second valid witness to be constructed for each targeted honest presentation. Witness indistinguishability hides the switch to this witness. Once these presentations use the auxiliary branch, the reduction can omit the corresponding hidden issuer credentials and their signing queries. Throughout this change, proofs are honestly generated under the honest proof-system parameters.

The specific structure of the unforgeability game enables a particularly simple reduction. This proof strategy overhead boils down to an auxiliary OR branch in the NIZK relation and, in a concrete realization, a trusted setup for the corresponding auxiliary verification key. All proofs remain honestly generated, so simulation and extraction never need to be combined within the same hybrid. Without this structure, a generic proof could instead require strong simulation extractability (sSE), which would restrict the choice of admissible NIZKs, especially in the existing landscape of relevant plausibly post-quantum proof systems. In particular, to the best of our knowledge, known results either *i*) cover the non-sufficient weak simulation extractability notion [CF24, dHLM<sup>+</sup>26], *ii*) need to rely on non-succinct Unruh-transformed MPCitH protocols (e.g., [CDG<sup>+</sup>17]) or *iii*) need to rely on a generic compiler involving OR composition and strongly EUF-CMA secure one-time signatures (e.g., [KZM<sup>+</sup>15, ARS20]). The latter is generic, but adds additional overhead to our tailored approach and any non-generic approach to sSE may require additional scheme-specific analysis to establish this stronger property for existing proof systems.

In a deployment, the same abstraction requires establishing  $\text{vk}_A$  without leaving  $\text{sk}_A$  available to an operational party. We do not prescribe a concrete setup ceremony. A setup authority may generate and erase  $\text{sk}_A$ , or the key may be generated distributively. For hash-based signatures such as CAPSS, the verification key is obtained by evaluating a one-way function on the secret key. Participants can therefore secret-share the auxiliary secret key, jointly evaluate this circuit using MPC, and reveal only the resulting verification key. Multi-party parameter-generation ceremonies provide a well-established precedent for this trust model [BGM17]. If at least one participant is honest and erases its share, the signing key remains unrecoverable even if all remaining participants collude.

The auxiliary signature binds the complete presentation statement to the device key used in that presentation. Consequently, after extraction, an auxiliary witness either corresponds to an honest presentation already authorized by the challenger or yields a forgery of the auxiliary signature. In the remaining credential-branch case, position binding excludes inconsistent disclosures, while the omitted honest-issuance queries ensure that the extracted issuer signature is fresh. This bridges the gap between a BDBAC forgery and a forgery of one of the underlying signature schemes.

*Security overview.* We analyze each security property separately:

- **Correctness.** An honestly generated presentation satisfies each component of the presentation relation: the issuer credential verifies, the blind signature authorizes the verifier context, and every disclosed attribute has a valid opening. Correctness therefore follows from the correctness of the underlying signature, blind-signature, commitment, and proof systems.

**Table 1.** High-level comparison with representative anonymous-credential constructions. Presentation sizes are in KB; our two rows report proof sizes, excluding the remaining protocol communication. For Lai et al., the range is calculated by adding the reported signature and pseudonym sizes for the selectively and adaptively secure variants, respectively; the pseudonym includes the commitment and its proof. “General predicates” means efficiently computable predicates expressible in the underlying proof system; their practical cost depends on the complexity of the corresponding relation. IP and AGM denote the ideal-permutation and algebraic-group models. N/R indicates that the cited work does not report a presentation size.

| Reference                             | Main assumptions/model | Presentation capability | Device-bound | SE-oblivious | Presentation (KB)           |
|---------------------------------------|------------------------|-------------------------|--------------|--------------|-----------------------------|
| [FHLL26]                              | DL, RO, AGM            | Attribute disclosure    | Yes          | Yes          | N/R                         |
| [BLNS23b]                             | ISIS- $f$ , LWE        | Attribute disclosure    | No           | No           | 122–133                     |
| [LCL <sup>+</sup> 23]                 | SIS, LWE               | Attribute disclosure    | No           | No           | $\approx 1\,368$ – $2\,247$ |
| [AGJ <sup>+</sup> 24]                 | SIS, LWE               | Attribute disclosure    | No           | No           | 80                          |
| [PWF23]                               | SIS, LWE, RO           | Token validity          | No           | No           | 85–175                      |
| [FR26a]                               | IP, RO                 | General predicates      | No           | No           | 140                         |
| <b>This work (Longfellow backend)</b> | IP, RO                 | Attribute disclosure    | Yes          | Yes          | 506.11                      |
| <b>This work (Aurora)</b>             | IP, RO                 | Attribute disclosure    | Yes          | Yes          | 214.65                      |

- **One-more unforgeability.** Straight-line knowledge soundness supplies the witnesses needed to classify accepting outputs. The auxiliary-branch argument above handles the nontrivial-presentation case. The separate one-more case yields blind signatures on more distinct contexts than the honest device has authorized, contradicting one-more unforgeability of BS.
- **Unlinkability.** Zero knowledge replaces the setup and the proofs returned in successful presentation sessions by their simulated counterparts. The simulator uses only the public issuer key, context, and disclosure predicate, without the credential or device secret key. Failed sessions and adversarial device messages remain unchanged. This gives the real-versus-simulated comparison required by Unlink-3 across the adversary’s adaptively chosen sessions.
- **SE-obliviousness.** This property is reduced to blindness of the device-side blind signature. The reduction places the real presentation context and an independently sampled dummy context in the two executions of the blindness game. Blind-signature correctness bounds the case in which the honestly answered dummy execution aborts while the adversarially answered execution succeeds.

*Prototype implementation.* Our implementation follows this blueprint with CAPSS for all three signature roles, a Merkle-tree commitment for selective disclosure, and Aurora and the Longfellow proof backend for the final R1CS relation. For 16 attributes and four disclosed attributes, the latter proves the exported relation in 0.45 seconds and verifies it in 0.29 seconds; its proof is 506.11 KB. Including witness preparation, R1CS export, and circuit compilation, the user-side presentation takes 2.60 seconds, while total presentation communication is 522.48 KB. In the measured Aurora workflow, credential issuance takes 0.54 seconds in the representative compact run. On the presentation side, the complete user computation takes 13.18 seconds and verification takes 1.37 seconds. The serialized proof has size 214.65 KB, while the complete online presentation communicates 231.02 KB once the verifier request, device interaction, and disclosed attributes are included.

Table 1 compares presentation sizes with representative post-quantum anonymous credentials and the closest device-bound construction. The comparison is not strictly like-for-like: the schemes provide different predicate languages and security properties, and only our reported post-quantum presentation includes a fresh context-blind SE authorization linked to the credential, which adds additional overhead. The detailed experimental methodology and a discussion of these differences appear in Section 6.2. It needs to be stressed that our implementation is a feasibility prototype, not a fully optimized implementation and there remains significant potential for optimization.

The CAPSS anonymous-credential application is the closest hash-based baseline for our implementation. Its 140 KB figure covers one hidden CAPSS credential signature and an arithmetic predicate. Our presentation additionally hides and verifies a CAPSS signature authorizing the fresh context, proves that its public key is the same device key certified by the issuer, and includes the auxiliary CAPSS branch required by the generic reduction. It also uses two proof backends for the complete relation. End-to-end measurements for

both appear in Section 6.2. The definitions and security properties of the cryptographic building blocks used throughout the paper are recalled in Section 2.

## 2 Background

### 2.1 Digital Signatures

A digital signature scheme is a tuple of PPT algorithms

$$\text{Sig} = (\text{Sig.Setup}, \text{Sig.KeyGen}, \text{Sig.Sign}, \text{Sig.Verify})$$

with the following syntax:

- $\text{Sig.Setup}(1^\lambda) \rightarrow \text{par}_{\text{Sig}}$  outputs public parameters specifying a message space  $\mathcal{M}$ .
- $\text{Sig.KeyGen}(\text{par}_{\text{Sig}}) \rightarrow (\text{sk}, \text{pk})$  outputs a signing key and a verification key.
- $\text{Sig.Sign}(m, \text{sk}) \rightarrow \sigma$ , for  $m \in \mathcal{M}$ , outputs a signature.
- $\text{Sig.Verify}(m, \sigma, \text{pk}) \rightarrow \{0, 1\}$  verifies a signature.

Public parameters are omitted from the notation whenever clear from context.

*Correctness.* For every public parameter string  $\text{par}_{\text{Sig}}$  in the support of  $\text{Sig.Setup}(1^\lambda)$ , every key pair  $(\text{sk}, \text{pk})$  in the support of  $\text{Sig.KeyGen}(\text{par}_{\text{Sig}})$ , and every  $m \in \mathcal{M}$ , honestly generated signatures verify except with negligible probability:

$$\Pr \left[ \sigma \leftarrow \text{Sig.Sign}(m, \text{sk}) : \text{Sig.Verify}(m, \sigma, \text{pk}) = 0 \right] \leq \text{negl}(\lambda).$$

**EUFCMA.** Let  $\mathcal{O}_{\text{Sign}}$  record every queried message in a set  $\mathcal{Q}_{\text{Sign}}$  and return  $\text{Sig.Sign}(m, \text{sk})$ . For every PPT adversary  $\mathcal{A}$ , define

$$\text{Adv}_{\text{Sig}, \mathcal{A}}^{\text{EUFCMA}}(\lambda) := \Pr \left[ \begin{array}{l} \text{par}_{\text{Sig}} \leftarrow \text{Sig.Setup}(1^\lambda), \\ (\text{sk}, \text{pk}) \leftarrow \text{Sig.KeyGen}(\text{par}_{\text{Sig}}), \\ \mathcal{Q}_{\text{Sign}} \leftarrow \emptyset, \\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\text{Sign}}}(\text{par}_{\text{Sig}}, \text{pk}) : \\ m^* \notin \mathcal{Q}_{\text{Sign}} \wedge \text{Sig.Verify}(m^*, \sigma^*, \text{pk}) = 1 \end{array} \right].$$

We say that  $\text{Sig}$  is EUFCMA secure if this advantage is negligible for every PPT adversary  $\mathcal{A}$ .

The generic construction uses  $\text{Sig}_i$  under two independently generated key pairs. The issuer key authenticates credentials, while the auxiliary key authenticates presentation statements only inside the auxiliary branch used by the security reduction. Thus both roles rely on the same EUFCMA assumption.

### 2.2 Blind Signatures

We use the round-optimal interaction syntax of Beullens et al. [BLNS23a], while making the generation of common public parameters explicit. A signing session consists of one user message, one signer response, and a local user finalization step. A round-optimal blind signature scheme is a tuple of PPT algorithms

$$\text{BS} = (\text{BS.Setup}, \text{BS.KeyGen}, \text{BS.Sign}_1, \text{BS.Sign}_2, \text{BS.Sign}_3, \text{BS.Verify})$$

with the following syntax:

- $\text{BS.Setup}(1^\lambda) \rightarrow \text{par}_{\text{BS}}$  outputs common public parameters specifying a message space  $\mathcal{M}_{\text{BS}}$ .
- $\text{BS.KeyGen}(\text{par}_{\text{BS}}) \rightarrow (\text{sk}, \text{pk})$  outputs a signing key and a verification key.
- $\text{BS.Sign}_1(\text{pk}, \mu) \rightarrow (\rho_1, \text{state}_U)$ , for  $\mu \in \mathcal{M}_{\text{BS}}$ , is run by the user and outputs the first protocol message together with a private user state.
- $\text{BS.Sign}_2(\text{sk}, \rho_1) \rightarrow \rho_2$  is run by the signer and outputs its response.
- $\text{BS.Sign}_3(\rho_2, \text{state}_U) \rightarrow \sigma$  is run by the user and outputs either a signature or  $\perp$ .
- $\text{BS.Verify}(\text{pk}, \mu, \sigma) \rightarrow \{0, 1\}$  verifies a signature.

The parameters  $\text{par}_{\text{BS}}$  are public inputs to all algorithms and are omitted from the notation whenever clear from context.

*Correctness.* For every public parameter string  $\text{par}_{\text{BS}}$  in the support of  $\text{BS.Setup}(1^\lambda)$ , every key pair  $(\text{sk}, \text{pk})$  in the support of  $\text{BS.KeyGen}(\text{par}_{\text{BS}})$ , and every  $\mu \in \mathcal{M}_{\text{BS}}$ ,

$$\Pr \left[ \begin{array}{l} (\rho_1, \text{state}_{\text{U}}) \leftarrow \text{BS.Sign}_1(\text{pk}, \mu), \\ \rho_2 \leftarrow \text{BS.Sign}_2(\text{sk}, \rho_1), \\ \sigma \leftarrow \text{BS.Sign}_3(\rho_2, \text{state}_{\text{U}}) : \\ \text{BS.Verify}(\text{pk}, \mu, \sigma) = 0 \end{array} \right] \leq \text{negl}(\lambda).$$

*Blindness.* For every stateful PPT adversary  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$  whose first algorithm outputs two equal-length messages in  $\mathcal{M}_{\text{BS}}$ , define

$$\text{Adv}_{\text{BS}, \mathcal{A}}^{\text{BLD}}(\lambda) := \Pr \left[ \begin{array}{l} \text{par}_{\text{BS}} \leftarrow \text{BS.Setup}(1^\lambda), \\ (\text{pk}, \mu_0, \mu_1, \text{state}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\text{par}_{\text{BS}}), \\ b \leftarrow \{0, 1\}, \\ (\rho_1^0, \text{state}_{\text{U}}^0) \leftarrow \text{BS.Sign}_1(\text{pk}, \mu_0), \\ (\rho_1^1, \text{state}_{\text{U}}^1) \leftarrow \text{BS.Sign}_1(\text{pk}, \mu_1), \\ (\rho_{1,L}, \rho_{1,R}) \leftarrow (\rho_1^b, \rho_1^{1-b}), \\ (\rho_{2,L}, \rho_{2,R}, \text{state}'_{\mathcal{A}}) \leftarrow \mathcal{A}_2(\text{state}_{\mathcal{A}}, \rho_{1,L}, \rho_{1,R}), \\ (\rho_2^b, \rho_2^{1-b}) \leftarrow (\rho_{2,L}, \rho_{2,R}), \\ \sigma_0 \leftarrow \text{BS.Sign}_3(\rho_2^0, \text{state}_{\text{U}}^0), \\ \sigma_1 \leftarrow \text{BS.Sign}_3(\rho_2^1, \text{state}_{\text{U}}^1), \\ \text{if } \text{BS.Verify}(\text{pk}, \mu_0, \sigma_0) = 0 \vee \text{BS.Verify}(\text{pk}, \mu_1, \sigma_1) = 0 \\ \quad \text{then } (\sigma_0, \sigma_1) \leftarrow (\perp, \perp) : \\ \mathcal{A}_3(\text{state}'_{\mathcal{A}}, \sigma_0, \sigma_1) = b \end{array} \right] - \frac{1}{2}.$$

We say that BS is blind if this advantage is negligible for every such adversary.

*One-more unforgeability.* We use the adaptive-output formulation of Friedrichs et al. [FHLL26]. Let  $\mathcal{O}_{\text{Sign}_2}$ , on input  $\rho_1$ , compute  $\rho_2 \leftarrow \text{BS.Sign}_2(\text{sk}, \rho_1)$ , increment a counter  $q$ , and return  $\rho_2$ . Thus,  $q$  denotes the total number of signer responses returned to the adversary.

For every PPT adversary  $\mathcal{A}$ , define

$$\text{Adv}_{\text{BS}, \mathcal{A}}^{\text{OMUF}}(\lambda) := \Pr \left[ \begin{array}{l} \text{par}_{\text{BS}} \leftarrow \text{BS.Setup}(1^\lambda), \\ (\text{sk}, \text{pk}) \leftarrow \text{BS.KeyGen}(\text{par}_{\text{BS}}), \\ q \leftarrow 0, \\ (k, (\mu_i, \sigma_i)_{i \in [k]}) \leftarrow \mathcal{A}^{\mathcal{O}_{\text{Sign}_2}}(\text{par}_{\text{BS}}, \text{pk}) : \\ q < k \wedge |\{\mu_i : i \in [k]\}| = k \wedge \\ \bigwedge_{i \in [k]} \text{BS.Verify}(\text{pk}, \mu_i, \sigma_i) = 1 \end{array} \right].$$

We say that BS is one-more unforgeable if this advantage is negligible for every PPT adversary  $\mathcal{A}$ .

## 2.3 Vector Commitments

Vector commitments allow a committer to bind to an ordered vector and later open the commitment at selected positions. We use the standard position-binding notion introduced by Catalano and Fiore [CF13], with an interface close to that of Peikert, Pepin, and Sharp [PPS21].

A vector commitment scheme for vectors of length  $\ell$  over a message space  $\mathcal{M}$  is a tuple of PPT algorithms

$$\text{Com} = (\text{Com.Setup}, \text{Com.Commit}, \text{Com.Open}, \text{Com.Verify})$$

with the following syntax:

- $\text{Com.Setup}(1^\lambda, \ell) \rightarrow \text{par}_{\text{Com}}$  outputs public parameters.
- $\text{Com.Commit}(\text{par}_{\text{Com}}, \text{Attrs}) \rightarrow (\text{cmt}, \text{cst})$ , for  $\text{Attrs} \in \mathcal{M}^\ell$ , outputs a commitment and a private committer state.
- $\text{Com.Open}(\text{par}_{\text{Com}}, \text{cst}, i) \rightarrow \text{opening}_i$ , for  $i \in [\ell]$ , outputs an opening for the  $i$ -th position.
- $\text{Com.Verify}(\text{par}_{\text{Com}}, \text{cmt}, i, a, \text{opening}_i) \rightarrow \{0, 1\}$  verifies that  $a$  is the value committed at position  $i$ .

Public parameters are omitted from the notation whenever clear from context.

*Correctness.* The scheme is correct if, for every  $\lambda \in \mathbb{N}$ , every polynomially bounded  $\ell$ , every  $\text{Attrs} \in \mathcal{M}^\ell$ , and every  $i \in [\ell]$ ,

$$\Pr \left[ \begin{array}{l} \text{par}_{\text{Com}} \leftarrow \text{Com.Setup}(1^\lambda, \ell), \\ (\text{cmt}, \text{cst}) \leftarrow \text{Com.Commit}(\text{par}_{\text{Com}}, \text{Attrs}), \\ \text{opening}_i \leftarrow \text{Com.Open}(\text{par}_{\text{Com}}, \text{cst}, i) : \\ \text{Com.Verify}(\text{par}_{\text{Com}}, \text{cmt}, i, \text{Attrs}_i, \text{opening}_i) = 1 \end{array} \right] \geq 1 - \text{negl}(\lambda).$$

*Position binding.* For every PPT adversary  $\mathcal{A}$  and every polynomially bounded  $\ell$ , define

$$\text{Adv}_{\text{Com}, \mathcal{A}}^{\text{bind}}(\lambda) := \Pr \left[ \begin{array}{l} \text{par}_{\text{Com}} \leftarrow \text{Com.Setup}(1^\lambda, \ell), \\ (\text{cmt}, i, a, a', \text{opening}, \text{opening}') \leftarrow \mathcal{A}(\text{par}_{\text{Com}}) : \\ i \in [\ell] \wedge a \neq a' \wedge \\ \text{Com.Verify}(\text{par}_{\text{Com}}, \text{cmt}, i, a, \text{opening}) = 1 \wedge \\ \text{Com.Verify}(\text{par}_{\text{Com}}, \text{cmt}, i, a', \text{opening}') = 1 \end{array} \right].$$

We say that  $\text{Com}$  is position binding if this advantage is negligible in  $\lambda$ .

For convenience, for an index set  $I \subseteq [\ell]$  and values  $\mathbf{d} = (d_i)_{i \in I}$ , we use the component-wise notation

$$\mathbf{opening}_I := (\text{opening}_i)_{i \in I}, \quad \text{Com.Open}(\text{cst}, I) := (\text{Com.Open}(\text{cst}, i))_{i \in I},$$

and

$$\begin{aligned} \text{Com.Verify}(\text{cmt}, I, \mathbf{d}; \mathbf{opening}_I) &= 1 \\ \iff \bigwedge_{i \in I} \text{Com.Verify}(\text{cmt}, i, d_i; \text{opening}_i) &= 1, \end{aligned}$$

where public parameters are omitted whenever clear from context. This notation is only shorthand for a collection of positional openings and does not require an aggregatable opening algorithm. We require neither hiding nor update algorithms, and impose no asymptotic size requirement at the generic level.

## 2.4 Non-Interactive Zero-Knowledge Arguments of Knowledge

We recall the standard syntax of a non-interactive zero-knowledge argument of knowledge and separate the three properties used later: zero knowledge, witness indistinguishability, and straight-line knowledge soundness.

Let  $\mathcal{R} \subseteq \{0, 1\}^* \times \{0, 1\}^*$  be a polynomial-time decidable relation, and let

$$\mathcal{L}_{\mathcal{R}} := \{x : \exists w, (x, w) \in \mathcal{R}\}$$

be its associated language.

**Definition 1 (NIZK interface).** A non-interactive argument for  $\mathcal{R}$  is a tuple of PPT algorithms

$$\text{NIZK}_{\mathcal{R}} = (\text{NIZK}_{\mathcal{R}}.\text{Setup}, \text{NIZK}_{\mathcal{R}}.\text{Prove}, \text{NIZK}_{\mathcal{R}}.\text{Verify})$$

with the following syntax:

- $\text{NIZK}_{\mathcal{R}}.\text{Setup}(1^\lambda) \rightarrow \text{par}_{\text{NIZK}}$  outputs public proof-system parameters.
- $\text{NIZK}_{\mathcal{R}}.\text{Prove}(\text{par}_{\text{NIZK}}, x, w) \rightarrow \pi$  outputs a proof for  $(x, w) \in \mathcal{R}$ .
- $\text{NIZK}_{\mathcal{R}}.\text{Verify}(\text{par}_{\text{NIZK}}, x, \pi) \rightarrow \{0, 1\}$  verifies the proof.

Public parameters are omitted from the notation whenever clear from context.

*Correctness.* For every  $(x, w) \in \mathcal{R}$ , honestly generated proofs verify except with negligible probability:

$$\Pr \left[ \begin{array}{l} \text{par}_{\text{NIZK}} \leftarrow \text{NIZK}_{\mathcal{R}}.\text{Setup}(1^\lambda), \\ \pi \leftarrow \text{NIZK}_{\mathcal{R}}.\text{Prove}(\text{par}_{\text{NIZK}}, x, w) : \\ \text{NIZK}_{\mathcal{R}}.\text{Verify}(\text{par}_{\text{NIZK}}, x, \pi) = 0 \end{array} \right] \leq \text{negl}(\lambda).$$

For the security properties below, let  $\text{Sim} = (\text{Sim.Setup}, \text{Sim.Prove})$  be a stateful PPT simulator. Its setup algorithm outputs

$$(\widetilde{\text{par}}_{\text{NIZK}}, \text{td}_{\text{sim}}, \text{state}_{\text{sim}}) \leftarrow \text{Sim.Setup}(1^\lambda).$$

Each invocation of  $\text{Sim.Prove}$  may update the persistent state  $\text{state}_{\text{sim}}$ . Let  $\mathcal{Q}_{\text{sim}}$  record the statements successfully submitted to the simulated proof oracle.

For fixed public parameters, define the real proof oracle by

$$\begin{aligned} \mathcal{O}_{\text{P}}^{\text{real}}(x, w) : \\ \text{if } (x, w) \notin \mathcal{R} \text{ then return } \perp, \\ \pi \leftarrow \text{NIZK}_{\mathcal{R}}.\text{Prove}(\text{par}_{\text{NIZK}}, x, w), \\ \text{return } \pi, \end{aligned}$$

and the simulated proof oracle by

$$\begin{aligned} \mathcal{O}_{\text{P}}^{\text{sim}}(x, w) : \\ \text{if } (x, w) \notin \mathcal{R} \text{ then return } \perp, \\ (\pi, \text{state}_{\text{sim}}) \leftarrow \text{Sim.Prove}(\text{td}_{\text{sim}}, \text{state}_{\text{sim}}, x), \\ \mathcal{Q}_{\text{sim}} \leftarrow \mathcal{Q}_{\text{sim}} \cup \{x\}, \\ \text{return } \pi. \end{aligned}$$

*Zero knowledge.* Zero knowledge holds for polynomially many proofs generated under the same public parameters, where later queries may depend on earlier proofs and public information. For every stateful PPT distinguisher  $\mathcal{A}$ , define

$$\text{Real}_{\text{NIZK}_{\mathcal{R}}, \mathcal{A}}^{\text{ZK}}(\lambda) := \Pr \left[ \begin{array}{l} \text{par}_{\text{NIZK}} \leftarrow \text{NIZK}_{\mathcal{R}}.\text{Setup}(1^\lambda), \\ b' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{P}}^{\text{real}}}(\text{par}_{\text{NIZK}}) : \\ b' = 1 \end{array} \right]$$

and

$$\text{Ideal}_{\text{NIZK}_{\mathcal{R}}, \text{Sim}, \mathcal{A}}^{\text{ZK}}(\lambda) := \Pr \left[ \begin{array}{l} (\widetilde{\text{par}}_{\text{NIZK}}, \text{td}_{\text{sim}}, \text{state}_{\text{sim}}) \leftarrow \text{Sim.Setup}(1^\lambda), \\ \mathcal{Q}_{\text{sim}} \leftarrow \emptyset, \\ b' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{P}}^{\text{sim}}}(\widetilde{\text{par}}_{\text{NIZK}}) : \\ b' = 1 \end{array} \right].$$

Define

$$\text{Adv}_{\text{NIZK}_{\mathcal{R}}, \text{Sim}, \mathcal{A}}^{\text{ZK}}(\lambda) := \left| \text{Real}_{\text{NIZK}_{\mathcal{R}}, \mathcal{A}}^{\text{ZK}}(\lambda) - \text{Ideal}_{\text{NIZK}_{\mathcal{R}}, \text{Sim}, \mathcal{A}}^{\text{ZK}}(\lambda) \right|.$$

The argument is zero knowledge if there exists such a stateful simulator  $\text{Sim}$  for which this advantage is negligible for every stateful PPT distinguisher  $\mathcal{A}$ . All simulated proofs in the ideal experiment are generated under one setup and one persistent simulator state.

*Witness indistinguishability.* For any stateful PPT adversary  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$  that always outputs  $(x, w_0), (x, w_1) \in \mathcal{R}$ , define

$$\text{Adv}_{\text{NIZK}_{\mathcal{R}}, \mathcal{A}}^{\text{WI}}(\lambda) := \left| \Pr \left[ \begin{array}{l} \text{par}_{\text{NIZK}} \leftarrow \text{NIZK}_{\mathcal{R}}.\text{Setup}(1^\lambda), \\ (x, w_0, w_1, \text{state}) \leftarrow \mathcal{A}_1(\text{par}_{\text{NIZK}}), \\ b \leftarrow \$_\{0, 1\}, \\ \pi \leftarrow \text{NIZK}_{\mathcal{R}}.\text{Prove}(\text{par}_{\text{NIZK}}, x, w_b), \\ b' \leftarrow \mathcal{A}_2(\text{state}, \pi) : \\ b' = b \end{array} \right] - \frac{1}{2} \right|.$$

The argument is witness indistinguishable if this advantage is negligible for every such adversary. Zero knowledge implies witness indistinguishability, but we state the latter separately because the unforgeability proof uses only this weaker property.

*Straight-line knowledge soundness.* There must exist a PPT extractor  $\text{Ext}$  that, on an accepting argument produced by an adversary, returns a valid witness without rewinding or re-running that adversary.

For every PPT adversary  $\mathcal{A}$ , define

$$\text{Adv}_{\text{NIZK}_{\mathcal{R}}, \text{Ext}, \mathcal{A}}^{\text{SL-KS}}(\lambda) := \Pr \left[ \begin{array}{l} \text{par}_{\text{NIZK}} \leftarrow \text{NIZK}_{\mathcal{R}}.\text{Setup}(1^\lambda), \\ (x^*, \pi^*) \leftarrow \mathcal{A}(\text{par}_{\text{NIZK}}), \\ w^* \leftarrow \text{Ext}(\text{par}_{\text{NIZK}}, x^*, \pi^*) : \\ \text{NIZK}_{\mathcal{R}}.\text{Verify}(\text{par}_{\text{NIZK}}, x^*, \pi^*) = 1 \wedge \\ (w^* = \perp \vee (x^*, w^*) \notin \mathcal{R}) \end{array} \right].$$

We require this advantage to be negligible for every PPT adversary  $\mathcal{A}$ . In models where extraction observes a public oracle, this observation is part of the extractor's execution rather than an additional simulation-extractability requirement.

### 3 Adaptation of the BDBAC Framework

We adopt the context-blind device-bound anonymous credential framework of Friedrichs et al. [FHLL26]. We preserve its parties, threat model, and security goals, while making the issuance and presentation interfaces explicit. As in the original framework, credentials are defined with respect to a predicate family  $\Phi$  over attribute vectors, and each presentation is associated with a public context  $\text{ctx}$ .

#### 3.1 Syntax and Correctness

**Definition 2 (Context-blind device-bound anonymous credentials).** A context-blind device-bound anonymous credential scheme BDBAC, defined with respect to a predicate family  $\Phi = \{\Phi_{\text{par}}\}_{\text{par}}$  and a message space  $\mathcal{M} = \{\mathcal{M}_{\text{par}}\}_{\text{par}}$ , uses a public fixed-length context space  $\mathcal{C}_{\text{par}} := \{0, 1\}^{\kappa(\lambda, \ell)}$ , where  $\lambda \leq \kappa(\lambda, \ell) = \text{poly}(\lambda, \ell)$ , and consists of the algorithms ( $\text{Setup}$ ,  $\text{IssKGen}$ ,  $\text{DKGen}$ ,  $\text{Issue}$ ,  $\text{ShowUser}$ ,  $\text{ShowSE}$ ,  $\text{Verify}$ ,  $\text{VfCred}$ ) with the following syntax:

- $\text{Setup}(1^\lambda, \ell) \rightarrow \text{par}$  generates public parameters.
- $\text{IssKGen}(\text{par}) \rightarrow (\text{isk}, \text{ipk})$  generates an issuer signing key and verification key.
- $\text{DKGen}(\text{par}) \rightarrow (\text{dsk}, \text{dpk})$  generates a device secret key and public key.
- $\text{Issue}(\text{isk}, \text{dpk}, \text{Attrs}) \rightarrow \sigma$  generates a credential  $\sigma$  bound to the device public key  $\text{dpk}$  and attribute vector  $\text{Attrs}$ .
- Showing is a two-message protocol with the secure element followed by local user finalization; it returns a presentation  $\tau$  for  $(\text{ctx}, \phi)$ :
  - $\text{ShowUser}_1(\text{ipk}, \text{dpk}, \sigma, \text{Attrs}, \text{ctx}, \phi) \rightarrow (\text{ust}, \text{umsg})$ ;
  - $\text{ShowSE}_1(\text{dsk}, \text{ipk}, \text{umsg}) \rightarrow \text{smsg}$ ;
  - $\text{ShowUser}_2(\text{ust}, \text{smsg}) \rightarrow \tau$ .
- $\text{Verify}(\text{ipk}, \tau, \text{ctx}, \phi) \rightarrow \{0, 1\}$  verifies a presentation with respect to context  $\text{ctx}$  and predicate  $\phi$ .
- $\text{VfCred}(\text{ipk}, \sigma, \text{dpk}, \text{Attrs}) \rightarrow \{0, 1\}$  verifies that  $\sigma$  is a valid credential for device public key  $\text{dpk}$  and attribute vector  $\text{Attrs}$ .

*Correctness.* A BDBAC scheme  $\Pi$  is  $\eta$ -correct if, for every  $\lambda, \ell \in \mathbb{N}$  and every PPT input generator  $\mathcal{G}$  that, on input  $\text{par}$ , outputs  $(\text{Attrs}, \phi, \text{ctx}) \in \mathcal{M}_{\text{par}} \times \Phi_{\text{par}} \times \mathcal{C}_{\text{par}}$  such that  $\phi(\text{Attrs}) = 1$ , the following experiment outputs 1 with probability at least  $1 - \eta(\lambda)$ :

$$\begin{aligned} & \text{par} \leftarrow \text{Setup}(1^\lambda, \ell), \\ & (\text{Attrs}, \phi, \text{ctx}) \leftarrow \mathcal{G}(\text{par}), \\ & (\text{isk}, \text{ipk}) \leftarrow \text{IssKGen}(\text{par}), \\ & (\text{dsk}, \text{dpk}) \leftarrow \text{DKGen}(\text{par}), \\ & \sigma \leftarrow \text{Issue}(\text{isk}, \text{dpk}, \text{Attrs}), \\ & (\tau, \perp) \leftarrow \langle \text{ShowUser}(\text{ipk}, \text{dpk}, \sigma, \text{Attrs}, \text{ctx}, \phi) \rightleftharpoons \text{ShowSE}(\text{dsk}, \text{ipk}) \rangle, \\ & \text{return } \text{VfCred}(\text{ipk}, \sigma, \text{dpk}, \text{Attrs}) \wedge \text{Verify}(\text{ipk}, \tau, \text{ctx}, \phi). \end{aligned}$$

Thus, correctness requires both that an honestly issued credential verifies and that an honest presentation of every satisfied predicate is accepted by the verifier.

### 3.2 Unforgeability

Unforgeability rules out accepting presentations that cannot be explained by the adversary's interactions with the issuer, users, and secure elements. Since the device public key is hidden from an ordinary verifier, the security experiment must nevertheless recover the device key underlying an accepting presentation in order to relate it to the interactions recorded by the game.

Following Friedrichs et al. [FHLL26], we therefore parameterize the definition by an extraction interface  $\text{Ext} := (\text{Ext.Setup}, \text{EKey})$ . The setup algorithm generates public parameters together with extraction state, while  $\text{EKey}$  recovers the device public key bound to an accepting presentation.

The adversary outputs  $k$  accepting presentations for pairwise distinct contexts and a common extracted device key. The sets  $T_1, T_2, T_3$  in Figure 1 capture presentations already justified by the experiment, corresponding respectively to a corrupt user with a corrupt secure element, a corrupt user with an honest secure element, and an honest user with an honest secure element. For honest secure elements, an additional one-more condition prevents the adversary from producing more presentations than the number of device interactions granted by the game.

**Definition 3 (Unforgeability).** *A BDBAC scheme  $\Pi$  is unforgeable if there exists a PPT extraction interface  $\text{Ext}$  such that, for every polynomially bounded attribute length  $\ell$ , the public-parameter marginal of  $\text{Ext.Setup}(1^\lambda, \ell)$  is computationally indistinguishable from  $\text{Setup}(1^\lambda, \ell)$  and, for every PPT adversary  $\mathcal{A}$ ,*

$$\text{Adv}_{\Pi, \ell, \text{Ext}, \mathcal{A}}^{\text{UNF}}(\lambda) := \Pr \left[ \text{UNF}_{\Pi, \ell, \text{Ext}}^{\mathcal{A}}(\lambda) = 1 \right]$$

*is negligible in  $\lambda$ .*

*Simulated setup.* The privacy notions below use a simulator  $\text{Sim}$  whose remaining interface depends on the experiment under consideration. In every case,  $\text{Sim}$  includes a setup algorithm  $\text{Sim.Setup}(1^\lambda, \ell)$  that generates the public parameters used in the simulated experiment. These parameters must be computationally indistinguishable from honestly generated ones.

For every PPT distinguisher  $\mathcal{D}$ , define

$$\begin{aligned} \text{Adv}_{\Pi, \text{Sim}, \mathcal{D}}^{\text{Setup-Ind}}(\lambda) \\ := \left| \Pr[\mathcal{D}(\text{par}) = 1 \mid \text{par} \leftarrow \text{Setup}(1^\lambda, \ell)] - \Pr[\mathcal{D}(\text{par}) = 1 \mid \text{par} \leftarrow \text{Sim.Setup}(1^\lambda, \ell)] \right|. \end{aligned}$$

### 3.3 Unlinkability

We adopt the Unlink-3 notion of Friedrichs et al. [FHLL26], which captures presentation privacy even when the secure element is fully corrupted. The adversary provides a valid credential and its attribute vector and then interacts with the honest user-side showing algorithms. The interfaces  $\mathcal{O}_{\text{User},1}$  and  $\mathcal{O}_{\text{User},2}$  expose the two user rounds of the showing protocol. Session identifiers allow polynomially many concurrent sessions with arbitrary adversarial interleaving. For each completed session, the adversary receives either the real presentation or a simulated presentation generated from the public issuer key, context, predicate, and permitted leakage  $\mathcal{L}(\text{isk})$ . Any selectively disclosed information is represented by the public predicate itself. The simulator  $\text{Sim} := (\text{Sim.Setup}, \text{Sim.Show})$  maintains persistent state across presentation queries. The complete experiment is given in Figure 2 below.

**Definition 4 (Unlink-3).** *A BDBAC scheme  $\Pi$  satisfies Unlink-3 with respect to a leakage function  $\mathcal{L}$  if there exists a PPT simulator  $\text{Sim}$  such that, for every PPT distinguisher  $\mathcal{D}$  and every polynomially bounded attribute length  $\ell$ ,  $\text{Adv}_{\Pi, \text{Sim}, \mathcal{D}}^{\text{Setup-Ind}}(\lambda) \leq \text{negl}(\lambda)$  and, for every PPT adversary  $\mathcal{A}$ ,*

$$\begin{aligned} \text{Adv}_{\Pi, \ell, \text{Sim}, \mathcal{L}, \mathcal{A}}^{\text{Unlink-3}}(\lambda) \\ := \left| \Pr \left[ \text{Unlink-3}_{\Pi, \ell, \text{Sim}, \mathcal{L}}^{\mathcal{A},1}(\lambda) = 1 \right] - \Pr \left[ \text{Unlink-3}_{\Pi, \ell, \text{Sim}, \mathcal{L}}^{\mathcal{A},0}(\lambda) = 1 \right] \right| \end{aligned}$$

*is negligible in  $\lambda$ .*

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><math>\text{Unf}_{\text{BDBAC}, \ell, \text{Ext}}^A(\lambda)</math></p> <hr/> <p> <math>(\text{state}_{\text{ext}}, \text{par}) \leftarrow \text{Ext.Setup}(1^\lambda, \ell)</math><br/> <math>\text{CSEs}, \text{HUsers}, \text{CUCreds}, \text{HPres} \leftarrow \emptyset</math><br/> <math>\text{SEKeys}, \text{HUCreds}, \text{GrantedCount} \leftarrow ()</math><br/> <math>(\text{isk}, \text{ipk}) \leftarrow \text{IssKGen}(\text{par})</math><br/> <math>(\tau_i^*, \text{ctx}_i^*, \phi_i^*)_{i \in [k]} \leftarrow \mathcal{A}^{\text{O}^*}(\text{par}, \text{ipk})</math><br/> <math>\text{dpk}_i^* \leftarrow \text{Ext.EKey}(\text{state}_{\text{ext}}, \text{ipk}, \tau_i^*, \text{ctx}_i^*, \phi_i^*), \forall i \in [k]</math><br/> <b>if</b> <math>\exists i, j \in [k] : \text{dpk}_i^* \neq \text{dpk}_j^*</math> <b>then</b><br/>             <b>return</b> 0<br/> <b>if</b> <math>\exists i \in [k] : \text{Verify}(\text{ipk}, \tau_i^*, \text{ctx}_i^*, \phi_i^*) \neq 1</math> <b>then</b><br/>             <b>return</b> 0<br/> <b>if</b> <math>\exists i \neq j : \text{ctx}_i^* = \text{ctx}_j^*</math> <b>then</b><br/>             <b>return</b> 0<br/> <math>\text{dpk}^* \leftarrow \text{dpk}_1^*</math><br/> <b>► Type 1: Corrupt user, corrupt SE</b><br/> <math>T_1 \leftarrow \{(\text{dpk}, \text{ctx}, \phi) \mid</math><br/>             <math>\exists \text{Attrs} : (\text{dpk}, \text{Attrs}) \in \text{CUCreds}</math><br/>             <math>\wedge \text{dpk} \in \text{CSEs}</math><br/>             <math>\wedge \phi(\text{Attrs}) = 1</math><br/>             <math>\wedge \text{ctx} \in \{0, 1\}^*\}</math><br/> <b>► Type 2: Corrupt user, honest SE</b><br/> <math>T_2 \leftarrow \{(\text{dpk}, \text{ctx}, \phi) \mid</math><br/>             <math>\exists \text{Attrs} : (\text{dpk}, \text{Attrs}) \in \text{CUCreds}</math><br/>             <math>\wedge \text{dpk} \notin \text{CSEs}</math><br/>             <math>\wedge \phi(\text{Attrs}) = 1</math><br/>             <math>\wedge \text{ctx} \in \{0, 1\}^*\}</math><br/> <b>► Type 3: Honest user, honest SE</b><br/> <math>T_3 \leftarrow \{(\text{dpk}, \text{ctx}, \phi) \mid (\text{dpk}, \text{ctx}, \phi) \in \text{HPres}\}</math><br/> <math>T \leftarrow T_1 \cup T_2 \cup T_3</math><br/> <b>if</b> <math>(\text{dpk}^*, \text{ctx}_1^*, \phi_1^*) \notin T</math> <b>then</b><br/>             <b>return</b> 1<br/> <b>► One-more-forgery condition</b><br/> <b>if</b> <math>\text{SEKeys}[\text{dpk}^*] \neq \perp \wedge \text{dpk}^* \notin \text{CSEs}</math><br/>             <math>\wedge \text{dpk}^* \notin \text{HUsers} \wedge \text{GrantedCount}[\text{dpk}^*] &lt; k</math> <b>then</b><br/>                 <b>return</b> 1<br/> <b>return</b> 0<br/> <math>\mathcal{O}_{\text{NewSE}}(\rho, \text{isHonestUser})</math> <hr/> <p> <b>if</b> <math>\rho = \perp</math> <b>then</b><br/>             <math>(\text{dsk}, \text{dpk}) \leftarrow \text{DKGen}(\text{par})</math><br/> <b>else</b><br/>             <math>(\text{dsk}, \text{dpk}) \leftarrow \text{DKGen}(\text{par}; \rho)</math><br/> <b>if</b> <math>\text{dpk} \in \text{SEKeys}</math> <b>then</b> <b>return</b> <math>\perp</math><br/> <b>if</b> <math>\rho \neq \perp</math> <b>then</b><br/>             <math>\text{CSEs} \leftarrow \text{CSEs} \cup \{\text{dpk}\}</math><br/> <math>\text{SEKeys}[\text{dpk}] \leftarrow \text{dsk}</math><br/> <math>\text{GrantedCount}[\text{dpk}] \leftarrow 0</math><br/> <b>if</b> <math>\text{isHonestUser} = 1</math> <b>then</b><br/>             <math>\text{HUsers} \leftarrow \text{HUsers} \cup \{\text{dpk}\}</math><br/> <b>return</b> <math>\text{dpk}</math> </p> </p> | <p><math>\mathcal{O}_{\text{Issue}}(\text{dpk}, \text{Attrs})</math></p> <hr/> <p> <b>if</b> <math>\text{dpk} \in \text{HUsers} \vee \text{dpk} \notin \text{SEKeys}</math> <b>then</b><br/>             <b>return</b> <math>\perp</math><br/> <math>\sigma \leftarrow \text{Issue}(\text{isk}, \text{dpk}, \text{Attrs})</math><br/> <b>if</b> <math>\sigma = \perp</math> <b>then</b><br/>             <b>return</b> <math>\perp</math><br/> <math>\text{CUCreds} \leftarrow \text{CUCreds} \cup \{(\text{dpk}, \text{Attrs})\}</math><br/> <b>return</b> <math>\sigma</math> </p> <hr/> <p><math>\mathcal{O}_{\text{ShowSE}}(\text{dpk}, \text{umsg})</math></p> <hr/> <p> <b>if</b> <math>\text{dpk} \notin \text{SEKeys}</math> <b>then</b><br/>             <b>return</b> <math>\perp</math><br/> <math>\text{smsg} \leftarrow \text{ShowSE}_1(\text{SEKeys}[\text{dpk}], \text{ipk}, \text{umsg})</math><br/> <math>\text{GrantedCount}[\text{dpk}] \leftarrow \text{GrantedCount}[\text{dpk}] + 1</math><br/> <b>return</b> <math>\text{smsg}</math> </p> <hr/> <p><math>\mathcal{O}_{\text{hIssue}}(\text{cid}, \text{dpk}, \text{Attrs})</math></p> <hr/> <p> <b>if</b> <math>\text{dpk} \notin \text{SEKeys} \vee \text{cid} \in \text{HUCreds}</math> <b>then</b><br/>             <b>return</b> <math>\perp</math><br/> <math>\sigma \leftarrow \text{Issue}(\text{isk}, \text{dpk}, \text{Attrs})</math><br/> <b>if</b> <math>\sigma = \perp</math> <b>then</b><br/>             <b>return</b> <math>\perp</math><br/> <math>\text{HUCreds}[\text{cid}] \leftarrow (\text{dpk}, \text{Attrs}, \sigma)</math><br/> <b>return</b> <math>\perp</math> </p> <hr/> <p><math>\mathcal{O}_{\text{hShow}}(\text{cid}, \text{ctx}, \phi)</math></p> <hr/> <p> <b>if</b> <math>\text{cid} \notin \text{HUCreds}</math> <b>then</b><br/>             <b>return</b> <math>\perp</math><br/> <math>(\text{dpk}, \text{Attrs}, \sigma) \leftarrow \text{HUCreds}[\text{cid}]</math><br/> <b>if</b> <math>\text{dpk} \notin \text{HUsers} \vee \text{dpk} \in \text{CSEs}</math> <b>then</b><br/>             <b>return</b> <math>\perp</math><br/> <b>if</b> <math>\text{dpk} \notin \text{SEKeys}</math> <b>then</b><br/>             <b>return</b> <math>\perp</math><br/> <math>\text{dsk} \leftarrow \text{SEKeys}[\text{dpk}]</math><br/> <math>(\tau, \perp) \leftarrow \begin{cases} \text{ShowUser}(\text{ipk}, \text{dpk}, \sigma, \text{Attrs}, \text{ctx}, \phi) \\ \text{ShowSE}(\text{dsk}, \text{ipk}) \end{cases}</math><br/> <b>if</b> <math>\tau = \perp</math> <b>then</b><br/>             <b>return</b> <math>\perp</math><br/> <math>\text{HPres} \leftarrow \text{HPres} \cup \{(\text{dpk}, \text{ctx}, \phi)\}</math><br/> <b>return</b> <math>\tau</math> </p> <hr/> <p><math>\mathcal{O}_{\text{CorruptSE}}(\text{dpk})</math></p> <hr/> <p> <b>if</b> <math>\text{dpk} \notin \text{SEKeys}</math> <b>then</b><br/>             <b>return</b> <math>\perp</math><br/> <math>\text{CSEs} \leftarrow \text{CSEs} \cup \{\text{dpk}\}</math><br/> <b>return</b> <math>\text{SEKeys}[\text{dpk}]</math> </p> |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Fig. 1. Unforgeability game for context-blind device-bound anonymous credential schemes.

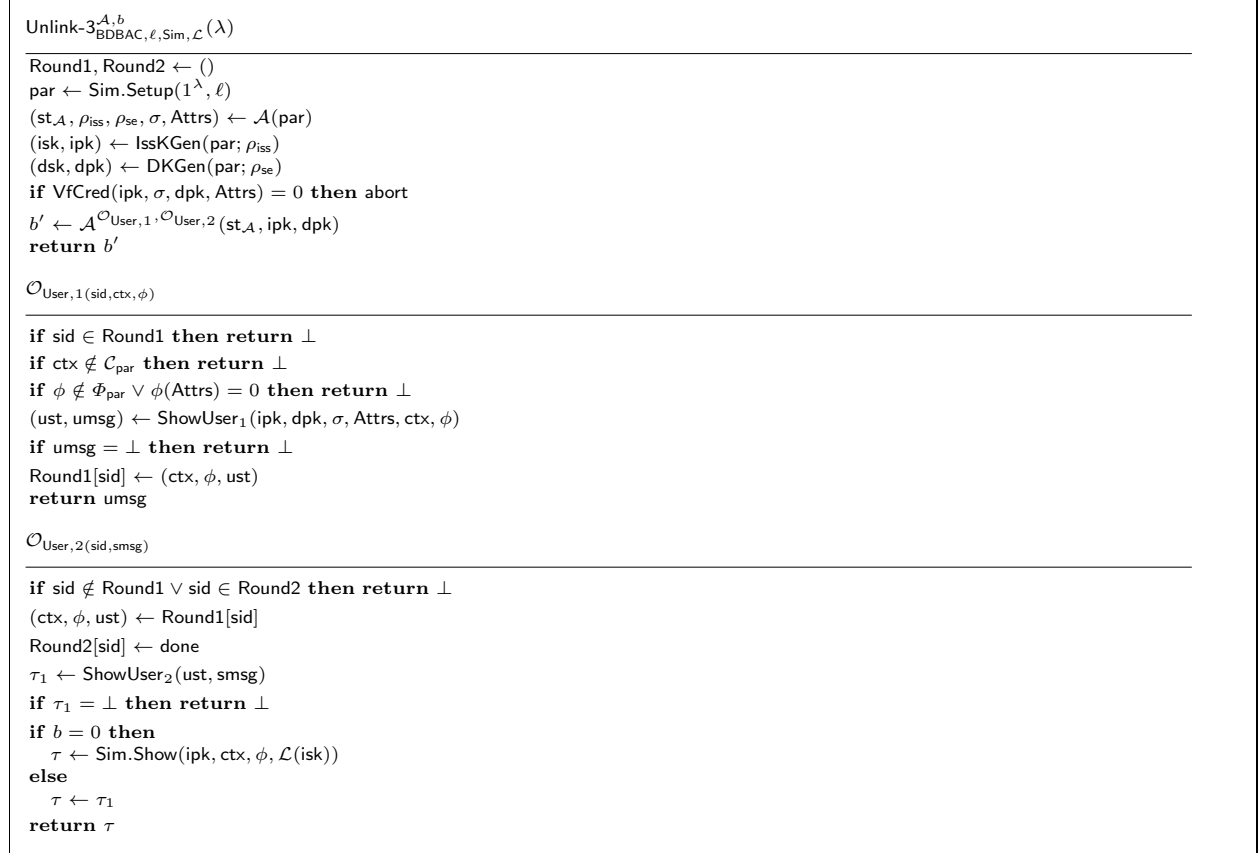


Fig. 2. Unlink-3 game with a fully corrupted secure element.

### 3.4 SE-Obliviousness

SE-obliviousness captures privacy against the secure element during the showing protocol. It requires that the user messages observed by a potentially malicious secure element can be simulated without the credential, attributes, presentation context, or predicate, except for the explicitly permitted leakage  $\mathcal{L}(\text{isk})$ .

The interfaces expose the two user rounds and use session identifiers to support concurrent executions. In the simulated experiment,  $\text{Sim.Check}$  determines whether the adversarial response is consistent with the simulated first-round state; malformed responses yield  $\perp$ , as in a real execution. The simulator  $\text{Sim} := (\text{Sim.Setup}, \text{Sim.ShowUser}_1, \text{Sim.Check})$  maintains persistent state across concurrent sessions. The complete experiment is given in Figure 3 below.

**Definition 5 (SE-obliviousness).** A BDBAC scheme  $\Pi$  is SE-oblivious with respect to a leakage function  $\mathcal{L}$  if there exists a PPT simulator  $\text{Sim}$  such that, for every PPT distinguisher  $\mathcal{D}$  and every polynomially bounded attribute length  $\ell$ ,  $\text{Adv}_{\Pi, \text{Sim}, \mathcal{D}}^{\text{Setup-Ind}}(\lambda) \leq \text{negl}(\lambda)$  and, for every PPT adversary  $\mathcal{A}$ ,

$$\begin{aligned} & \text{Adv}_{\Pi, \ell, \text{Sim}, \mathcal{L}, \mathcal{A}}^{\text{SE-Obliv}}(\lambda) \\ &:= \left| \Pr \left[ \text{SE-Obliv}_{\Pi, \ell, \text{Sim}, \mathcal{L}}^{\mathcal{A}, 1}(\lambda) = 1 \right] - \Pr \left[ \text{SE-Obliv}_{\Pi, \ell, \text{Sim}, \mathcal{L}}^{\mathcal{A}, 0}(\lambda) = 1 \right] \right| \end{aligned}$$

is negligible in  $\lambda$ .

## 4 Generic Construction

We now present a generic construction of a context-blind device-bound anonymous credential scheme following the model of [FHLL26], with the explicit interfaces described in Section 3. The construction is modular in its signature, blind-signature, vector-commitment, and NIZK components at the level of its syntax.

---

SE-Obliv<sup>A, b</sup><sub>BDBAC, ℓ, Sim, L</sub>(λ)

---

Round1, Round2 ← ()  
par ← Sim.Setup(1<sup>λ</sup>, ℓ)  
(st<sub>A</sub>, ρ<sub>iss</sub>, ρ<sub>se</sub>, σ, Attrs) ← A(par)  
(isk, ipk) ← IssKGen(par; ρ<sub>iss</sub>)  
(dsk, dpk) ← DKGen(par; ρ<sub>se</sub>)  
**if** VFCred(ipk, σ, dpk, Attrs) = 0 **then** abort  
b' ← A<sup>⓪<sub>User, 1</sub></sup>(st<sub>A</sub>, ipk, dpk)  
**return** b'

---

⓪<sub>User, 1</sub>(sid, ctx, φ)

---

**if** sid ∈ Round1 **then** **return** ⊥  
**if** ctx ∉ C<sub>par</sub> **then** **return** ⊥  
**if** φ ∉ Φ<sub>par</sub> ∨ φ(Attrs) = 0 **then** **return** ⊥  
**if** b = 0 **then**  
(ust, umsg) ← Sim.ShowUser<sub>1</sub>(ipk, dpk, L(isk))  
**else**  
(ust, umsg) ← ShowUser<sub>1</sub>(ipk, dpk, σ, Attrs, ctx, φ)  
**if** umsg = ⊥ **then** **return** ⊥  
Round1[sid] ← (ctx, φ, ust)  
**return** umsg

---

⓪<sub>User, 2</sub>(sid, smsg)

---

**if** sid ∉ Round1 ∨ sid ∈ Round2 **then** **return** ⊥  
(ctx, φ, ust) ← Round1[sid]  
**if** b = 0 **then**  
**if** Sim.Check(ust, smsg, L(isk)) = 0 **then**  
τ ← ⊥  
**else**  
(τ, ⊥) ← ⟨ShowUser(ipk, dpk, σ, Attrs, ctx, φ) ⇌ ShowSE(dsk, ipk)⟩  
**else**  
τ ← ShowUser<sub>2</sub>(ust, smsg)  
Round2[sid] ← done  
**return** τ

---

**Fig. 3.** SE-obliviousness game for context-blind device-bound anonymous credential schemes.

#### 4.1 Protocol Overview

Our construction combines an issuer signature scheme  $\text{Sig}_I$ , an auxiliary signature scheme  $\text{Sig}_A$  with an independently generated key pair, a vector commitment scheme  $\text{Com}$  supporting selective openings, a blind signature scheme  $\text{BS}$  and a non-interactive zero-knowledge proof system  $\text{NIZK}_{\widehat{\mathcal{R}}_{\text{pres}}}$ . We require  $\mathcal{C}_{\text{par}} \subseteq \mathcal{M}_{\text{BS}}$ , so every presentation context is a valid blind-signature message. We also assume that every tuple displayed below is encoded unambiguously in the message space of the primitive to which it is submitted. We define the messages authenticated during issuance and presentation as  $\text{msg}_I := (\text{cmt}, \text{dpk})$ , and  $\text{msg}_P := \text{ctx}$ .

At a high level, the issuer commits to the complete attribute vector and issues a credential that binds the resulting commitment to the device public key. It returns the credential together with the commitment state from which the user can derive selective openings. During presentation, the user obtains a blind signature from the device on a fresh presentation context and proves in zero knowledge that it holds a valid issuer credential and a fresh device contribution bound to the same device public key. The proof establishes that the selectively disclosed attributes are consistent with the attribute vector committed to during issuance. The attribute commitment need not be hiding: the issuer already receives the attributes, whereas the verifier sees only a zero-knowledge proof, not the commitment. Let  $\mathcal{M}$  denote the component message space of  $\text{Com}$ . We consider the selective-disclosure predicate family

$$\Phi_{\text{par}} := \left\{ \phi_{I, \mathbf{d}} \mid I \subseteq [\ell], \mathbf{d} = (d_i)_{i \in I} \in \mathcal{M}^{|I|} \right\},$$

where

$$\phi_{I, \mathbf{d}}(\text{Attrs}) = 1 \iff \forall i \in I : \text{Attrs}_i = d_i.$$

We represent each predicate canonically by the tuple  $(I, \mathbf{d})$ , with the indices in  $I$  written in increasing order and the entries of  $\mathbf{d}$  ordered accordingly. The deterministic procedure **Parse** therefore satisfies  $\text{Parse}(\phi_{I,\mathbf{d}}) = (I, \mathbf{d})$ , and two representations in  $\Phi_{\text{par}}$  are equal exactly when their parsed tuples are equal. For an index set  $I \subseteq [\ell]$ , we write  $\mathbf{opening}_I := (\text{opening}_i)_{i \in I}$  for the corresponding set of openings. The freshness of a device contribution is supplied by the external context  $\text{ctx}$ ; the blind signature authenticates this context but does not itself generate or enforce freshness. For fixed public parameters  $(\text{par}_{\text{Sig}_I}, \text{par}_{\text{Sig}_A}, \text{par}_{\text{BS}}, \text{par}_{\text{Com}}, \text{vk}_A)$ , we first define the credential relation  $\mathcal{R}_{\text{pres}}$ . Its associated language is  $\mathcal{L}_{\text{pres}} := \{x : \exists w, \mathcal{R}_{\text{pres}}(x, w) = 1\}$ . For a selective-disclosure predicate  $\phi_{I,\mathbf{d}}$ , the public statement is  $x := (\text{ipk}, \text{ctx}, I, \mathbf{d})$ , while the witness is  $w := (\text{cmt}, \text{bsig}, \text{dpk}, \text{cred}, \mathbf{opening}_I)$ . The relation accepts if and only if

$$\mathcal{R}_{\text{pres}}(x, w) = 1 \iff \begin{cases} \text{Sig}_I.\text{Verify}((\text{cmt}, \text{dpk}), \text{cred}, \text{ipk}) = 1, \\ \text{BS}.\text{Verify}(\text{dpk}, \text{ctx}, \text{bsig}) = 1, \\ \forall i \in I : \text{Com}.\text{Verify}(\text{cmt}, i, d_i; \text{opening}_i) = 1 \end{cases}$$

The first condition establishes possession of a valid issuer credential bound to  $(\text{cmt}, \text{dpk})$ . The second establishes a fresh device contribution under the same device public key. The remaining condition ensures that the disclosed values admit valid openings at the selected positions of the commitment authenticated by the issuer. The proof system uses the public statement  $x$  and augmented witness  $\hat{w} = (b, w_{\text{cred}}, \text{dpk}_A, \sigma_A)$ . The augmented relation is

$$\begin{aligned} \widehat{\mathcal{R}_{\text{pres}}}(x; \hat{w}) &= 1 \\ \iff (b = 0 \wedge \mathcal{R}_{\text{pres}}(x, w_{\text{cred}}) = 1) \\ &\quad \vee (b = 1 \wedge \text{Sig}_A.\text{Verify}(\text{Encode}(x, \text{dpk}_A), \sigma_A, \text{vk}_A) = 1). \end{aligned}$$

Here  $\text{Encode}(x, \text{dpk}_A)$  canonically encodes  $(\text{ipk}, \text{ctx}, I, \mathbf{d}, \text{dpk}_A)$ ; in particular, the auxiliary signature is bound to the presentation context. Its associated language is

$$\widehat{\mathcal{L}_{\text{pres}}} := \{x : \exists b, w_{\text{cred}}, \text{dpk}_A, \sigma_A, \widehat{\mathcal{R}_{\text{pres}}}(x; (b, w_{\text{cred}}, \text{dpk}_A, \sigma_A)) = 1\}.$$

Honest users always set  $b = 0$ . The auxiliary branch is present in the same relation and circuit in every execution and is used only by the security reduction. Its signature binds the complete public presentation statement to the hidden device key used for the final classification. Here  $\text{Encode}(x, \text{dpk}) = \text{Encode}(\text{ipk}, \text{ctx}, I, \mathbf{d}, \text{dpk})$  is an injective encoding, including the context and canonical disclosure predicate. This authenticates a semantic authorization, not a unique proof transcript. The schemes  $\text{Sig}_I$  and  $\text{Sig}_A$  may be instantiated by the same signature construction, but their keys are independent.

The input  $\text{ipk}$  to  $\text{ShowSE}_1$  is retained to match the BDBAC interface, although it is not used by this construction.

**Theorem 1 (Correctness).** *Let  $\Pi$  be the BDBAC scheme in Figure 4. Suppose that  $\text{Sig}_I$ ,  $\text{BS}$ ,  $\text{Com}$ , and  $\text{NIZK}_{\widehat{\mathcal{R}_{\text{pres}}}}$  are correct. Then  $\Pi$  is correct according to the notion in Section 3.1.*

*Proof (Proof sketch).* Correctness follows directly. Commitment correctness makes the openings valid at issuance and presentation; issuer- and blind-signature correctness make the remaining checks in the credential branch of  $\widehat{\mathcal{R}_{\text{pres}}}$  valid. NIZK correctness then gives an accepting presentation, with the failure probability bounded by a union bound over the correctness errors. The complete proof is given in Section 5.1.

## 4.2 Security of the Protocol

The security reductions require the hidden device key to be extracted from an accepting presentation without rewinding the adversary. Here we state how the straight-line extractor is used by the BDBAC experiment.

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $\text{Setup}(1^\lambda, \ell)$<br>$\text{par}_{\text{Sig}_I} \leftarrow \text{Sig}_I.\text{Setup}(1^\lambda)$<br>$\text{par}_{\text{Sig}_A} \leftarrow \text{Sig}_A.\text{Setup}(1^\lambda)$<br>$\text{par}_{\text{BS}} \leftarrow \text{BS}.\text{Setup}(1^\lambda)$<br>$\text{par}_{\text{Com}} \leftarrow \text{Com}.\text{Setup}(1^\lambda, \ell)$<br>$(\text{sk}_A, \text{vk}_A) \leftarrow \text{Sig}_A.\text{KeyGen}(\text{par}_{\text{Sig}_A})$<br>$\text{erase } \text{sk}_A$<br>$\text{par}_{\text{NIZK}} \leftarrow \text{NIZK}_{\widehat{\mathcal{R}}_{\text{pres}}}.\text{Setup}(1^\lambda)$<br>$\text{par} \leftarrow (\text{par}_{\text{Sig}_I}, \text{par}_{\text{Sig}_A}, \text{par}_{\text{BS}}, \text{par}_{\text{Com}}, \text{vk}_A, \text{par}_{\text{NIZK}})$<br><b>return</b> $\text{par}$<br>$\text{IssKGen}(\text{par})$<br><hr/> $(\text{isk}, \text{ipk}) \leftarrow \text{Sig}_I.\text{KeyGen}(\text{par}_{\text{Sig}_I})$<br><b>return</b> $(\text{isk}, \text{ipk})$<br>$\text{DKGen}(\text{par})$<br><hr/> $(\text{dsk}, \text{dpk}) \leftarrow \text{BS}.\text{KeyGen}(\text{par}_{\text{BS}})$<br><b>return</b> $(\text{dsk}, \text{dpk})$<br>$\text{Issue}(\text{isk}, \text{dpk}, \text{Attrs})$<br><hr/> $(\text{cmt}, \text{cst}) \leftarrow \text{Com}.\text{Commit}(\text{Attrs})$<br>$\text{cred} \leftarrow \text{Sig}_I.\text{Sign}((\text{cmt}, \text{dpk}), \text{isk})$<br>$\sigma \leftarrow (\text{cred}, \text{cmt}, \text{cst})$<br><b>return</b> $\sigma$<br>$\text{VfCred}(\text{ipk}, \sigma, \text{dpk}, \text{Attrs})$<br><hr/> $(\text{cred}, \text{cmt}, \text{cst}) \leftarrow \sigma$<br>$v_{\text{cred}} \leftarrow \text{Sig}_I.\text{Verify}((\text{cmt}, \text{dpk}), \text{cred}, \text{ipk})$<br><b>for all</b> $i \in [\ell]$ :<br>$\text{opening}_i \leftarrow \text{Com}.\text{Open}(\text{cst}, i)$<br>$v_i \leftarrow \text{Com}.\text{Verify}(\text{cmt}, i, \text{Attrs}_i; \text{opening}_i)$<br><b>return</b> $v_{\text{cred}} \wedge (\bigwedge_{i \in [\ell]} v_i)$ | $\text{ShowUser}_1(\text{ipk}, \text{dpk}, \sigma, \text{Attrs}, \text{ctx}, \phi)$<br><b>if</b> $\phi \notin \Phi_{\text{par}}$ <b>then</b><br><b>return</b> $\perp$<br><b>if</b> $\text{ctx} \notin C_{\text{par}}$ <b>then</b><br><b>return</b> $\perp$<br>$(I, \mathbf{d}) \leftarrow \text{Parse}(\phi)$<br>$(\text{cred}, \text{cmt}, \text{cst}) \leftarrow \sigma$<br><b>if</b> $\phi(\text{Attrs}) = 0$ <b>then</b><br><b>return</b> $\perp$<br><b>for all</b> $i \in I$ :<br>$\text{opening}_i \leftarrow \text{Com}.\text{Open}(\text{cst}, i)$<br>$\text{opening}_I \leftarrow (\text{opening}_i)_{i \in I}$<br>$\text{msgp} \leftarrow \text{ctx}$<br>$(\text{umsg}, \text{bst}) \leftarrow \text{BS}.\text{Sign}_1(\text{dpk}, \text{msgp})$<br>$\text{ust} \leftarrow (\text{bst}, \text{ipk}, \text{dpk}, \text{cred}, \text{cmt}, \text{ctx}, I, \mathbf{d}, \text{opening}_I)$<br><b>return</b> $(\text{ust}, \text{umsg})$<br>$\text{ShowSE}_1(\text{dsk}, \text{ipk}, \text{umsg})$<br><hr/> $\text{smsg} \leftarrow \text{BS}.\text{Sign}_2(\text{dsk}, \text{umsg})$<br><b>return</b> $\text{smsg}$<br>$\text{ShowUser}_2(\text{ust}, \text{smsg})$<br><hr/> $(\text{bst}, \text{ipk}, \text{dpk}, \text{cred}, \text{cmt}, \text{ctx}, I, \mathbf{d}, \text{opening}_I) \leftarrow \text{ust}$<br>$\text{bsig} \leftarrow \text{BS}.\text{Sign}_3(\text{smsg}, \text{bst})$<br><b>if</b> $\text{BS}.\text{Verify}(\text{dpk}, \text{ctx}, \text{bsig}) = 0$ <b>then</b><br><b>return</b> $\perp$<br>$x \leftarrow (\text{ipk}, \text{ctx}, I, \mathbf{d})$<br>$w \leftarrow (\text{cmt}, \text{bsig}, \text{dpk}, \text{cred}, \text{opening}_I)$<br>$\pi \leftarrow \text{NIZK}_{\widehat{\mathcal{R}}_{\text{pres}}}.\text{Prove}(x; (0, w, \perp, \perp))$<br><b>return</b> $\tau := \pi$<br>$\text{Verify}(\text{ipk}, \tau, \text{ctx}, \phi)$<br><hr/> <b>if</b> $\phi \notin \Phi_{\text{par}}$ <b>then</b><br><b>return</b> 0<br><b>if</b> $\text{ctx} \notin C_{\text{par}}$ <b>then</b><br><b>return</b> 0<br>$(I, \mathbf{d}) \leftarrow \text{Parse}(\phi)$<br>$x \leftarrow (\text{ipk}, \text{ctx}, I, \mathbf{d})$<br><b>return</b> $\text{NIZK}_{\widehat{\mathcal{R}}_{\text{pres}}}.\text{Verify}(x, \tau)$ |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**Fig. 4.** Generic construction of a context-blind device-bound anonymous credential scheme with selective disclosure.

*Device-key extraction.* For unforgeability, we instantiate the extractor  $\text{Ext} = (\text{Setup}, \text{EKey})$  required by the BDBAC security definition using the straight-line extractor of  $\text{NIZK}_{\widehat{\mathcal{R}}_{\text{pres}}}$ . The algorithm  $\text{Ext}.\text{Setup}(1^\lambda, \ell)$  runs the honest scheme setup and retains the public NIZK parameters in  $\text{state}_{\text{ext}}$ . Its public output is identically distributed to the output of  $\text{Setup}(1^\lambda, \ell)$ , as required by Definition 3. Given  $(\text{state}_{\text{ext}}, \text{ipk}, \tau, \text{ctx}, \phi)$ ,  $\text{Ext}.\text{EKey}$  returns  $\perp$  if  $\phi \notin \Phi_{\text{par}}$ ; otherwise it parses  $(I, \mathbf{d}) \leftarrow \text{Parse}(\phi)$  and returns  $\perp$  if  $\tau$  fails to parse as a NIZK proof. It forms  $x = (\text{ipk}, \text{ctx}, I, \mathbf{d})$  and invokes the straight-line extractor to obtain

$$\hat{w} \leftarrow \text{Ext}_{\text{NIZK}}(\text{par}_{\text{NIZK}}, x, \tau).$$

If extraction fails,  $\text{Ext}.\text{EKey}$  returns  $\perp$ . Otherwise it parses  $\hat{w} = (b, w_{\text{cred}}, \text{dpk}_A, \sigma_A)$ . If  $b = 0$ , it further parses  $w_{\text{cred}} = (\text{cmt}, \text{bsig}, \text{dpk}, \text{cred}, \text{opening}_I)$  and returns  $\text{dpk}$ . If  $b = 1$ , it returns  $\text{dpk}_A$ . Any malformed witness causes it to return  $\perp$ . Thus the device public key remains hidden from an ordinary verifier but can be recovered by the extractor in the unforgeability experiment.

*Security requirements.* Unforgeability relies on straight-line knowledge soundness and witness indistinguishability, EUF-CMA security of  $\text{Sig}_I$  and  $\text{Sig}_A$ , position binding of  $\text{Com}$ , and one-more unforgeability of  $\text{BS}$ . Zero knowledge of  $\text{NIZK}_{\widehat{\mathcal{R}}_{\text{pres}}}$  directly gives Unlink-3 because the complete credential, device public key, and blind signature occur only in the NIZK witness. SE-obliviousness follows from the blindness of  $\text{BS}$ : the only value sent to the secure element is the first blind-signature message. The reduction uses a second, honestly answered blind signing execution; correctness bounds the probability that this companion execution aborts while the adversarially answered target execution succeeds.

**Theorem 2 (Unforgeability).** *Let  $\Pi$  be the BDBAC scheme in Figure 4. Suppose that  $\text{Sig}_I$  and  $\text{Sig}_A$  are correct and EUF-CMA secure,  $\text{Com}$  is correct and position binding, and  $\text{BS}$  is one-more unforgeable.*

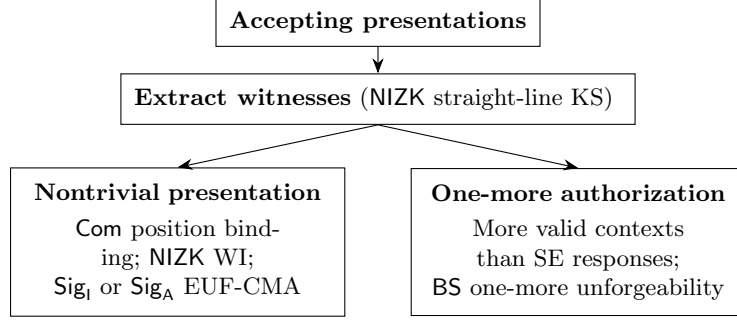


**Fig. 5.** The two cases of the generic unforgeability reduction. The game transitions and their bounds are detailed in the proof.

Suppose further that the underlying argument for  $\widehat{\mathcal{R}}_{\text{pres}}$  is witness indistinguishable and satisfies straight-line knowledge soundness. Then  $\Pi$  satisfies the unforgeability notion of Definition 3.

*Proof (Proof sketch).* We proceed through a sequence of games. In  $G_0$  we run the real experiment, and in  $G_1$  we extract witnesses from the adversary’s accepting presentations using straight-line knowledge soundness. Game  $G_2$  rules out auxiliary-branch witnesses by EUF-CMA security and separates nontrivial from one-more wins. For the nontrivial case,  $G_3$  records adversarial issuances, while  $G_4$  uses position binding to exclude inconsistent disclosures of previously signed commitments. Thus the issuer message in a nontrivial output was not signed by that issuance oracle.

This does not yet give an issuer-signature forgery: a new presentation may reuse a credential from an honest issuance. We guess its device key in  $G_5$  and ensure valid honest-showing witnesses in  $G_6$ . Game  $G_7$  uses witness indistinguishability to switch these proofs to auxiliary witnesses, allowing  $G_8$  to omit the hidden target issuer signatures. Extraction in  $G_8$  then yields either a recorded showing, an auxiliary-signature forgery, or an issuer-signature forgery. In the separate one-more case,  $M_1$  converts any additional valid device authorization beyond the number of signer responses into a blind-signature forgery. The complete proof is given in Section 5.2.

**Theorem 3 (Unlinkability).** *Let  $\Pi$  be the BDBAC scheme in Figure 4. Suppose that  $\text{NIZK}_{\widehat{\mathcal{R}}_{\text{pres}}}$  is zero knowledge. Then  $\Pi$  satisfies Unlink-3 according to Definition 4 with respect to the empty leakage function  $\mathcal{L}_{\perp}(\text{isk}) := \perp$ .*

*Proof (Proof sketch).* The proof compares three games. In  $G_0$ , the real user produces presentations under the simulator’s public parameters, as required by the Unlink-3 game. In  $G_1$ , the parameters are generated honestly; setup indistinguishability justifies this change. In  $G_2$ , zero knowledge changes the setup and all successful proofs jointly to their simulated counterparts. This is the ideal game: the simulator uses only the public statement, while failed sessions and device messages are unchanged. The zero-query case of zero knowledge also gives the setup-indistinguishability step. The complete proof is given in Section 5.3.

**Theorem 4 (SE-obliviousness).** *Let  $\Pi$  be the BDBAC scheme in Figure 4. Suppose that BS is correct and blind. Then  $\Pi$  is SE-oblivious with respect to the empty leakage function  $\mathcal{L}_{\perp}(\text{isk}) := \perp$ .*

*Proof (Proof sketch).* Let  $H_0$  be the real SE-obliviousness game. In  $H_j$ , the first  $j$  initiated sessions use a blind-signing request for an independent dummy context; upon successful finalization, the challenger generates a fresh presentation for the original context. For each transition  $H_{j-1} \rightarrow H_j$ , a blindness reduction places the real and dummy contexts in the two challenge sessions, forwards one first message to the adversarial SE, and answers the other honestly. It need not know which message is which: the blindness challenger returns the signatures indexed by context. The honest companion may fail, which contributes only the blind-signature correctness error. The last hybrid is the ideal game. The complete proof is given in Section 5.4.

## 5 Security of the Generic Construction

### 5.1 Correctness

*Proof (Proof of Theorem 1).* Fix public parameters, attributes  $\text{Attrs}$ , a context  $\text{ctx}$ , and  $\phi_{I,d}$  such that  $\phi_{I,d}(\text{Attrs}) = 1$ . Correctness of  $\text{Com}$  implies that the openings derived from the commitment state during presentation are valid except with negligible probability. Conditioned on this event, correctness of  $\text{Sig}_I$  gives a credential on  $(\text{cmt}, \text{dpk})$  and correctness of  $\text{BS}$  gives a device signature on  $\text{ctx}$ . These values satisfy the credential branch of  $\widehat{\mathcal{R}}_{\text{pres}}$ . NIZK correctness therefore gives an accepting presentation. A union bound over the polynomially many checks is negligible.

### 5.2 Unforgeability

*Proof (Proof of Theorem 2).* Let  $\mathcal{A}$  be a PPT adversary in the experiment of Figure 1. Let  $q_{\text{SE}}$ ,  $q_{\text{hShow}}$ , and  $q_{\text{out}}$  be polynomial upper bounds on its numbers of  $\text{NewSE}$  queries, honest-showing queries, and output presentations. Let  $\epsilon_{\text{Sig}_I}^{\text{corr}}$ ,  $\epsilon_{\text{Sig}_A}^{\text{corr}}$ , and  $\epsilon_{\text{Com}}^{\text{corr}}$  denote the correctness errors of  $\text{Sig}_I$ ,  $\text{Sig}_A$ , and one positional opening of  $\text{Com}$ , respectively. Let  $k \leq q_{\text{out}}$  be the actual output length.

The proof separates the two ways in which the adversary can win. After extracting the common device key and the witnesses underlying its accepting outputs, a nontrivial win must contain a credential not already explained by one of the authorization sets  $T_1, T_2, T_3$ . We turn this case into an issuer-signature forgery. A one-more win instead contains more valid device authorization than the honest secure element granted, and yields a forgery against the blind signature. The main difficulty in the first case is that honest issuances normally force the reduction to query its issuer-signing oracle. Games  $G_3$ – $G_8$  use the auxiliary branch to remove exactly those queries without changing the public statement or the need to simulate a proof.

More precisely, the three sets describe presentations already explained by the adversary's interactions with the game. The set  $T_1$  concerns a credential issued through  $\text{Issue}$  to an adversarial user and bound to a corrupted secure element. Since the adversary knows the device secret key, every context is covered whenever the disclosed predicate is true for the issued attributes. The set  $T_2$  concerns such an adversarially held credential bound to an uncorrupted secure element. It likewise records the credential and predicate as already authorized by the experiment, while distinguishing this case from  $T_1$  through the corruption status of the device key. Finally,  $T_3$  concerns an honest user with an honest secure element. Here the credential and attributes are never released to the adversary, so only the exact triples  $(\text{dpk}, \text{ctx}, \phi)$  returned by successful  $\text{hShow}$  calls are covered. Thus an accepting output outside  $T_1 \cup T_2 \cup T_3$  cannot be explained by a credential exposed through adversarial issuance or by an honest presentation previously authorized by the challenger. This is the nontrivial case addressed by the issuer-signature reduction.

$G_0$  (real unforgeability game) This is the experiment of Figure 1, instantiated with the extraction interface described above. Let  $S_i$  denote the event that game  $G_i$  returns one. We instrument  $\text{Setup}$  to retain the auxiliary secret key  $\text{sk}_A$ . This secret state is not given to  $\mathcal{A}$  and is not used before  $G_7$ , so the real distribution is unchanged. We also instrument each invocation of  $\text{Ext.EKey}$  to retain the complete output of the underlying NIZK extractor from which the returned device key is parsed. This is internal bookkeeping of the same extraction call, not a second invocation of the extractor.

$G_1$  (extract accepting outputs) For every output  $(\tau_i^*, \text{ctx}_i^*, \phi_i^*)$ , parse  $\phi_i^* = \phi_{I_i, d_i}$ , set  $x_i^* = (\text{ipk}, \text{ctx}_i^*, I_i, d_i)$ , and use the complete extractor output retained by the corresponding  $\text{Ext.EKey}$  call. The game aborts if verification accepts but that extraction does not return  $\widehat{w}_i^*$  satisfying  $\widehat{\mathcal{R}}_{\text{pres}}(x_i^*; \widehat{w}_i^*) = 1$ . A union bound gives a PPT adversary  $\mathcal{B}_{\text{SL-KS},0}$  such that

$$|\Pr[S_0] - \Pr[S_1]| \leq q_{\text{out}} \cdot \text{Adv}_{\text{NIZK}_{\widehat{\mathcal{R}}_{\text{pres}}}, \text{ExtNIZK}, \mathcal{B}_{\text{SL-KS},0}}^{\text{SL-KS}}(\lambda).$$

Thus, from this game onward, every accepting output can be analyzed through an actual witness of the augmented relation. In particular, the proof never infers the credential merely from acceptance of the NIZK.

G<sub>2</sub> (exclude the auxiliary branch) Parse every extracted witness as

$$\hat{w}_i^* = (b_i^*, w_{\text{cred},i}^*, \text{dpk}_{A,i}^*, \sigma_A^{*,i}).$$

The game aborts if  $b_i^* = 1$ . No auxiliary message has yet been signed, so such a witness contains an EUF-CMA forgery under  $\text{vk}_A$  on  $\text{Encode}(x_i^*, \text{dpk}_{A,i}^*)$ . Hence

$$|\Pr[\text{S}_1] - \Pr[\text{S}_2]| \leq \text{Adv}_{\text{Sig}_A, \mathcal{B}_{A,0}}^{\text{EUF-CMA}}(\lambda).$$

This step concerns the original experiment, before the reduction starts using  $\text{sk}_A$  for honest showings. It ensures that, except with the displayed loss, every initial winning output is explained by the credential branch. The right branch will be introduced deliberately in  $\text{G}_7$  and analyzed against the resulting signing table in  $\text{G}_8$ . Conditioned on no abort, write

$$w_i^* = (\text{cmt}_i^*, \text{bsig}_i^*, \text{dpk}_i^*, \text{cred}_i^*, \text{opening}_{I_i}^*).$$

Every accepting output then satisfies

$$\text{Sig}.Verify((\text{cmt}_i^*, \text{dpk}_i^*), \text{cred}_i^*, \text{ipk}) = 1, \quad (1)$$

$$\text{BS}.Verify(\text{dpk}_i^*, \text{ctx}_i^*, \text{bsig}_i^*) = 1, \quad (2)$$

$$\text{Com}.Verify(\text{cmt}_i^*, j, d_{i,j}; \text{opening}_{i,j}^*) = 1 \quad (j \in I_i). \quad (3)$$

All extracted device keys coincide; denote their common value by  $\text{dpk}^*$ . Let  $\text{NT}$  be the event

$$(\text{dpk}^*, \text{ctx}_1^*, \phi_1^*) \notin T_1 \cup T_2 \cup T_3,$$

and let  $\text{OM}$  be the complementary winning case governed by the one-more condition. These events are disjoint and exhaust  $\text{S}_2$ . The rest of the  $\text{G}$ -sequence bounds  $\text{NT}$ ; the independent  $\text{M}$ -sequence at the end bounds  $\text{OM}$ .

G<sub>3</sub> (record issuer signatures) The challenger records every successful **Issue** call in  $\mathcal{Q}_1$  as

$$(\text{dpk}, \text{Attrs}, \text{cmt}, (\text{opening}_j)_{j \in [\ell]}, \text{cred}).$$

This is internal bookkeeping and changes no distribution. Its purpose is to compare the issuer message extracted from the final presentation with every message that the reduction was forced to obtain for an adversarially controlled user.

G<sub>4</sub> (enforce consistent openings) The game aborts if  $m_1^* = (\text{cmt}_1^*, \text{dpk}^*)$  matches a signed message in  $\mathcal{Q}_1$  but the disclosed predicate is false for the recorded attribute vector. In that event, Equation (3) and the openings recorded at issuance give two valid openings of the same commitment position to distinct values. Thus

$$|\Pr[\text{NT in G}_3] - \Pr[\text{NT in G}_4]| \leq \text{Adv}_{\text{Com}, \mathcal{B}_{\text{Bind}}}^{\text{Bind}}(\lambda).$$

Conditioned on no abort, a matching issuer message would place the target in  $T_1$  if the device is corrupt, or in  $T_2$  if it remains honest. Indeed, position binding forces the disclosed values to agree with the attributes recorded at issuance, and hence the extracted predicate is true for that record. Consequently, event  $\text{NT}$  in  $\text{G}_4$  implies that  $m_1^*$  was not signed through **Issue**. This disposes of all issuer-signing queries made on behalf of corrupt users; the remaining queries to remove are those hidden inside honest issuance.

G<sub>5</sub> (guess the target device key) The challenger guesses uniformly one of the at most  $q_{\text{SE}}$  keys returned by **NewSE**, or the case  $\text{dpk}^* \notin \text{SEKeys}$ , and aborts unless the guess describes  $\text{dpk}^*$ . Therefore

$$\Pr[\text{NT in G}_4] \leq (q_{\text{SE}} + 1) \Pr[\text{NT in G}_5].$$

Fixing this target is necessary because the following witness switches must be made only for honest credentials bound to the device key whose hidden issuer-signing queries will be removed. All other credentials and device keys continue to be generated in the ordinary way.

**G<sub>6</sub>** (validate target witnesses) The game aborts whenever a successful target **hShow** call is about to invoke **Prove** with an invalid credential-branch witness. The blind-signature verification is already checked by **ShowUser<sub>2</sub>**; correctness and a union bound therefore give

$$|\Pr[\text{NT in } G_5] - \Pr[\text{NT in } G_6]| \leq q_{\text{hShow}}(\epsilon_{\text{Sig}_1}^{\text{corr}} + \ell\epsilon_{\text{Com}}^{\text{corr}}).$$

This check is what permits witness indistinguishability to be invoked in the next game: both the credential witness currently used and the auxiliary witness replacing it must satisfy the same augmented relation for the same public statement.

**G<sub>7</sub>** (switch target witnesses) For every successful target **hShow** call with public statement  $x_i = (\text{ipk}, \text{ctx}_i, I_i, \mathbf{d}_i)$  and device key  $\text{dpk}_i$ , the challenger computes

$$\sigma_A^{(i)} \leftarrow \text{Sig}_A.\text{Sign}(\text{Encode}(x_i, \text{dpk}_i), \text{sk}_A)$$

and aborts if this signature does not verify. By correctness, the probability of this event is at most  $q_{\text{hShow}}\epsilon_{\text{Sig}_A}^{\text{corr}}$ . Conditioned on no abort, it returns an honestly generated proof using the right-branch witness  $(1, \perp, \text{dpk}_i, \sigma_A^{(i)})$  instead of the credential-branch witness. It records  $(x_i, \text{dpk}_i)$  in an auxiliary-signing table  $\mathcal{Q}_A$ . Non-target operations are unchanged.

Order the successful target calls and define a sequence of  $q_{\text{hShow}}$  transitions: the first  $j$  successful calls use their right-branch witnesses, while the remaining calls use their credential-branch witnesses. If fewer than  $j$  target calls succeed, the corresponding transition makes no change. Otherwise adjacent hybrids differ in one honestly generated proof for the same statement and two valid witnesses. The reduction for the selected call generates all earlier and later proofs itself and preserves the adversary's state and adaptive interleaving. Witness indistinguishability therefore gives PPT adversaries  $\mathcal{B}_{\text{WI},j}$  such that

$$|\Pr[\text{NT in } G_6] - \Pr[\text{NT in } G_7]| \leq q_{\text{hShow}}\epsilon_{\text{Sig}_A}^{\text{corr}} + 2 \sum_{j=1}^{q_{\text{hShow}}} \text{Adv}_{\text{NIZK}_{\mathcal{R}_{\text{pres}}}, \mathcal{B}_{\text{WI},j}}^{\text{WI}}(\lambda).$$

Every hybrid uses the honest setup and honest proofs; no simulated proof is introduced in this game sequence. The public statement  $x_i$ , including its context and disclosure predicate, is unchanged. Only the witness supplied to **Prove** changes, and the auxiliary signature is bound to the full pair  $\text{Encode}(x_i, \text{dpk}_i)$ . Consequently, the adversary cannot reuse one auxiliary authorization for a different context, predicate, disclosed values, issuer key, or device key.

**G<sub>8</sub>** (omit target credentials and extract) During target **hIssue** calls, the challenger no longer requests the issuer signature and stores a placeholder for the credential. The credential is never returned to  $\mathcal{A}$ , and every later target presentation uses the right-branch witness from **G<sub>7</sub>**. This is a perfect change in the adversary's view. This is the point of the witness switch: before **G<sub>7</sub>**, the credential was needed to construct the presentation proof; after it, neither the credential nor the corresponding issuer-signing query is used anywhere visible to the adversary.

All presentations returned by honest **hShow** calls in **G<sub>8</sub>** are now honest proofs under the honest NIZK setup. The game reruns the extraction and aborts if any accepting adversarial output lacks a valid augmented witness. Straight-line knowledge soundness therefore gives

$$\Pr[\text{Bad}_{\text{ext}}] \leq q_{\text{out}} \cdot \text{Adv}_{\text{NIZK}_{\mathcal{R}_{\text{pres}}}, \text{Ext}_{\text{NIZK}}, \mathcal{B}_{\text{SL-KS},1}}^{\text{SL-KS}}(\lambda).$$

This application includes exact replays and new proofs for previously used statements: ordinary knowledge soundness has no freshness restriction.

We now classify the extracted witness rather than the proof transcript. This distinction is essential because an adversary may produce a fresh NIZK proof for a statement that appeared in an honest showing. Such a proof need not be a byte-for-byte replay, but any valid right-branch witness must still contain an auxiliary signature on the same encoded statement and device key.

If an extracted witness uses the right branch, it contains a valid auxiliary signature on  $\text{Encode}(x^*, \text{dpk}^*)$ . If  $(x^*, \text{dpk}^*) \in \mathcal{Q}_A$ , the corresponding honest showing inserted  $(\text{dpk}^*, \text{ctx}^*, \phi^*)$  into  $T_3$ , contradicting NT. This conclusion uses equality of the signed statement and device key, not equality of proof transcripts: even

a different proof for that pair has the same context and disclosure predicate and is covered by the same  $T_3$  entry. It cannot yield a one-more win either, since these  $\mathbf{hShow}$  keys belong to  $\mathbf{HUsers}$ , which that case excludes. Otherwise the signature is an EUF-CMA forgery under  $\mathbf{vk}_A$ . Thus, except with  $\mathbf{Adv}_{\mathbf{Sig}_A, \mathcal{B}_{A,1}}^{\text{EUF-CMA}}(\lambda)$ , event  $\mathbf{NT}$  yields a left-branch credential witness.

Hence the auxiliary branch closes exactly the gap between a new presentation and a new issuer signature. Re-proving an already authorized auxiliary statement is classified by  $T_3$ ; authorizing a new auxiliary statement forges  $\mathbf{Sig}_A$ ; and only the credential branch remains for the reduction against  $\mathbf{Sig}_I$ .

The reduction  $\mathcal{B}_I$  embeds its EUF-CMA challenge key as  $\mathbf{ipk}$ . It answers non-target issuer-signing operations with its signing oracle. Equation (1) supplies a valid signature on  $m_1^*$ . Game  $\mathbf{G}_4$  excludes messages signed through  $\mathbf{Issue}$ , and  $\mathbf{G}_8$  removed all target  $\mathbf{hIssue}$  signing queries. Issuances for another device key use a different message. Thus the extracted credential is valid under the challenge issuer key, while its complete signed message  $(\mathbf{cmt}_1^*, \mathbf{dpk}^*)$  was never submitted to the EUF-CMA signing oracle. Therefore

$$\begin{aligned} \Pr[\mathbf{NT} \text{ in } \mathbf{G}_8] &\leq q_{\text{out}} \mathbf{Adv}_{\mathbf{NIZK}_{\mathcal{R}_{\text{pres}}}, \text{Ext}_{\mathbf{NIZK}}, \mathcal{B}_{\text{SL-KS},1}}^{\text{SL-KS}}(\lambda) \\ &\quad + \mathbf{Adv}_{\mathbf{Sig}_A, \mathcal{B}_{A,1}}^{\text{EUF-CMA}}(\lambda) + \mathbf{Adv}_{\mathbf{Sig}_I, \mathcal{B}_I}^{\text{EUF-CMA}}(\lambda). \end{aligned}$$

Combining  $\mathbf{G}_3$ – $\mathbf{G}_8$  gives

$$\begin{aligned} \Pr[\mathbf{NT} \text{ in } \mathbf{G}_2] &\leq \mathbf{Adv}_{\text{Com}, \mathcal{B}_{\text{Bind}}}^{\text{Bind}}(\lambda) \\ &\quad + (q_{\text{SE}} + 1) \left( q_{\mathbf{hShow}} (\epsilon_{\mathbf{Sig}_I}^{\text{corr}} + \ell \epsilon_{\text{Com}}^{\text{corr}}) \right. \\ &\quad + q_{\mathbf{hShow}} \epsilon_{\mathbf{Sig}_A}^{\text{corr}} + 2 \sum_{j=1}^{q_{\mathbf{hShow}}} \mathbf{Adv}_{\mathbf{NIZK}_{\mathcal{R}_{\text{pres}}}, \mathcal{B}_{\text{WI},j}}^{\text{WI}}(\lambda) \\ &\quad + q_{\text{out}} \mathbf{Adv}_{\mathbf{NIZK}_{\mathcal{R}_{\text{pres}}}, \text{Ext}_{\mathbf{NIZK}}, \mathcal{B}_{\text{SL-KS},1}}^{\text{SL-KS}}(\lambda) \\ &\quad \left. + \mathbf{Adv}_{\mathbf{Sig}_A, \mathcal{B}_{A,1}}^{\text{EUF-CMA}}(\lambda) + \mathbf{Adv}_{\mathbf{Sig}_I, \mathcal{B}_I}^{\text{EUF-CMA}}(\lambda) \right). \end{aligned} \quad (4)$$

*One-more branch.* We analyze  $\mathbf{OM}$  in  $\mathbf{G}_2$ . On this event, the common key  $\mathbf{dpk}^*$  belongs to an honest, uncorrupted secure element and the extracted witnesses contain  $k$  valid blind signatures  $(\mathbf{ctx}_i^*, \mathbf{bsig}_i^*)_{i \in [k]}$  on pairwise distinct contexts, while  $q := \mathbf{GrantedCount}[\mathbf{dpk}^*] < k$ .

Unlike the nontrivial branch, this case does not require the extracted issuer credential to be fresh. Its violation is solely that the adversary obtained  $k$  distinct context signatures after fewer than  $k$  completed interactions with the corresponding honest device.

$\mathbf{M}_1$  (guess the honest SE) The challenger guesses which of the at most  $q_{\text{SE}}$   $\mathbf{NewSE}$  calls created  $\mathbf{dpk}^*$ . A reduction embeds its one-more challenge key there, forwards device-side signing sessions for that key, and generates all other keys honestly. It makes  $q < k$  completed signing queries and outputs the  $k$  extracted signatures on distinct messages. Notice that  $\mathbf{dpk}^* \notin \mathbf{HUsers}$  in this branch, so no  $\mathbf{hShow}$  call uses the embedded key; all signer responses under that key are precisely those counted by  $\mathbf{GrantedCount}[\mathbf{dpk}^*]$ . Consequently,

$$\Pr[\mathbf{OM} \text{ in } \mathbf{G}_2] \leq q_{\text{SE}} \cdot \mathbf{Adv}_{\mathcal{B}_S, \mathcal{B}_{\text{BS}}}^{\text{OMUF}}(\lambda). \quad (5)$$

Finally, combining the initial extraction games, the two branches, and Equations (4)–(5) gives a polynomial sum of the stated straight-line knowledge-soundness, witness-indistinguishability, EUF-CMA, position-binding, and one-more-unforgeability advantages, plus correctness errors. Every term is negligible, so the generic construction satisfies Definition 3.

### 5.3 Unlinkability

*Proof (Proof of Theorem 3).* Let  $\mathcal{L}_\perp(\text{isk}) := \perp$ . We first describe the simulator used in the ideal world. The algorithm  $\mathbf{Sim.Setup}$  runs the generic setup, replacing  $\mathbf{NIZK}_{\mathcal{R}_{\text{pres}}}.Setup$  by  $\mathbf{Sim.Setup}$ . It keeps the simulation trapdoor and persistent simulator state private and returns the resulting public parameters. On input  $(\mathbf{ipk}, \mathbf{ctx}, \phi, \perp)$ , the algorithm  $\mathbf{Sim.Show}$  parses  $(I, \mathbf{d}) \leftarrow \mathbf{Parse}(\phi)$ , forms  $x := (\mathbf{ipk}, \mathbf{ctx}, I, \mathbf{d})$  and computes

$$(\pi_{\text{sim}}, \text{state}_{\text{sim}}) \leftarrow \mathbf{Sim.Prove}(\text{td}_{\text{sim}}, \text{state}_{\text{sim}}, x).$$

It returns  $\tau_{\text{sim}} := \pi_{\text{sim}}$ . We now compare the real and ideal experiments.

G<sub>0</sub> Real presentations are generated under the simulated setup returned by  $\text{Sim.Setup}$ . This is the experiment  $\text{Unlink-3}_{\Pi, \ell, \text{Sim}, \mathcal{L}_\perp}^{\mathcal{A}, 1}(\lambda)$ .

G<sub>1</sub> The challenger replaces the simulated public parameters with honestly generated parameters and otherwise continues to return real presentations. A setup distinguisher can run the complete experiment after receiving the public parameters. Therefore,

$$|\Pr[G_0 = 1] - \Pr[G_1 = 1]| \leq \text{Adv}_{\Pi, \text{Sim}, \mathcal{B}_{\text{Setup}}}^{\text{Setup-Ind}}(\lambda).$$

G<sub>2</sub> The challenger simultaneously replaces the honest NIZK setup with  $\text{Sim.Setup}$  and every successful real presentation proof with a simulated proof for the same public statement. A distinguisher  $\mathcal{B}_{\text{ZK}}$  implements all user and secure-element interfaces and submits each corresponding extended instance–witness pair to its proof oracle. Failed sessions return  $\perp$  without making a proof query. The statements may depend on all preceding sessions and adversarial secure-element messages, as allowed by the zero-knowledge experiment. The real world of the ZK experiment gives  $G_1$ , whereas its ideal world gives  $G_2$ . Hence

$$|\Pr[G_1 = 1] - \Pr[G_2 = 1]| \leq \text{Adv}_{\text{NIZK}_{\widehat{\mathcal{R}}_{\text{pres}}}, \text{Sim}, \mathcal{B}_{\text{ZK}}}^{\text{ZK}}(\lambda).$$

In  $G_2$ , every successful final presentation is exactly the output of  $\text{Sim.Show}$  on  $(\text{ipk}, \text{ctx}, \phi, \perp)$ , while all failed sessions and adversarial messages are unchanged. Thus  $G_2$  is the ideal experiment  $\text{Unlink-3}_{\Pi, \ell, \text{Sim}, \mathcal{L}_\perp}^{\mathcal{A}, 0}(\lambda)$ , and

$$\text{Adv}_{\Pi, \ell, \text{Sim}, \mathcal{L}_\perp, \mathcal{A}}^{\text{Unlink-3}}(\lambda) \leq \text{Adv}_{\Pi, \text{Sim}, \mathcal{B}_{\text{Setup}}}^{\text{Setup-Ind}}(\lambda) + \text{Adv}_{\text{NIZK}_{\widehat{\mathcal{R}}_{\text{pres}}}, \text{Sim}, \mathcal{B}_{\text{ZK}}}^{\text{ZK}}(\lambda).$$

The setup-indistinguishability term is bounded by zero knowledge using a distinguisher that makes no proof query. Both terms are therefore negligible.

#### 5.4 SE-Obliviousness

*Proof (Proof of Theorem 4).* Let  $\mathcal{L}_\perp(\text{isk}) := \perp$ . The simulator uses the honest setup, so  $\text{Sim.Setup} := \text{Setup}$  and the setup-indistinguishability advantage is zero. Its first-round algorithm samples an independent dummy context  $\widehat{\text{ctx}} \leftarrow \$ \mathcal{C}_{\text{par}}$  and computes

$$(\text{umsg}_0, \text{bst}_0) \leftarrow \text{BS.Sign}_1(\text{dpk}, \widehat{\text{ctx}}).$$

It stores  $\text{ust}_0 := (\text{dpk}, \widehat{\text{ctx}}, \text{bst}_0)$  and returns  $(\text{ust}_0, \text{umsg}_0)$ . Given a response  $\text{smsg}$ , its checking algorithm computes

$$\widehat{\text{bsig}} \leftarrow \text{BS.Sign}_3(\text{smsg}, \text{bst}_0)$$

and returns

$$\text{Sim.Check}(\text{ust}_0, \text{smsg}, \perp) := \text{BS.Verify}(\text{dpk}, \widehat{\text{ctx}}, \widehat{\text{bsig}}).$$

Neither algorithm uses the credential, attributes, predicate, presentation context, or issuer-key leakage.

Let  $q_{\text{show}}$  be an upper bound on the number of first-round user queries. We order sessions by their first call to  $\mathcal{O}_{\text{User}, 1}$  and define the following hybrids.

H<sub>0</sub> All sessions are real. This is the experiment  $\text{SE-Obliv}_{\Pi, \ell, \text{Sim}, \mathcal{L}_\perp}^{\mathcal{A}, 1}(\lambda)$ .

H<sub>j</sub> For  $j \in [q_{\text{show}}]$ , the first  $j$  initiated sessions are simulated: they use  $\text{Sim.ShowUser}_1$  and  $\text{Sim.Check}$ . Whenever the check succeeds, their final presentation is generated by a fresh honest showing for the original  $(\text{ctx}, \phi)$ , exactly as in the ideal branch of Figure 3. All later sessions use the real user algorithms. The assignment of a branch to a session is fixed when its first round is initiated, so sessions may be interleaved arbitrarily.

Suppose that  $\mathcal{A}$  distinguishes  $H_{j-1}$  from  $H_j$ . We construct a blindness adversary  $\mathcal{B}_{\text{bld}, j}$ . It runs the complete SE-obliviousness experiment up to the  $j$ -th first-round query and submits

$$(\text{dpk}, \text{ctx}, \widehat{\text{ctx}})$$

to its blindness challenger, where  $\widehat{\text{ctx}} \leftarrow \$ \mathcal{C}_{\text{par}}$ . The two messages have the same public length. Write  $(\rho_{1,L}, \rho_{1,R})$  for the ordered pair returned by the challenger. The labels  $L$  and  $R$  identify positions only:

the reduction does not know which one encodes  $\text{ctx}$ . It forwards  $\rho_{1,L}$  to  $\mathcal{A}$  and receives a response  $\rho_{2,L}$ . It answers the companion execution honestly by computing

$$\rho_{2,R} \leftarrow \text{BS.Sign}_2(\text{dsk}, \rho_{1,R}).$$

The reduction can do so without knowing the message hidden in the right execution, since  $\text{Sign}_2$  takes only  $(\text{dsk}, \rho_{1,R})$ . It knows  $\text{dsk}$  because it runs  $\text{DKGen}$  using the coins selected in the SE-obliviousness experiment. It returns the ordered responses  $(\rho_{2,L}, \rho_{2,R})$  to the blindness challenger.

Let  $F_L$  and  $F_R$  denote failure of the left and right user finalizations, respectively. The blindness experiment returns the two signatures indexed by their messages when neither event occurs, and  $(\perp, \perp)$  otherwise. When both finalizations succeed,  $\mathcal{B}_{\text{bld},j}$  forms the presentation for the original context using the returned signature on  $\text{ctx}$ . If the left execution encoded  $\text{ctx}$ , that signature was obtained from the adversarial SE response and the target session is distributed as in  $H_{j-1}$ . If the left execution encoded  $\widehat{\text{ctx}}$ , the signature on  $\text{ctx}$  comes from the honest right execution and the target presentation is a fresh honest showing, as in  $H_j$ . Thus the reduction never needs to know the hidden order.

If  $F_L$  occurs, the target execution returns  $\perp$ , exactly as the reduction does after receiving  $(\perp, \perp)$ . If both executions succeed, the embedding is exact as argued above. Finally, suppose that  $\neg F_L \wedge F_R$  occurs. If the left execution encodes  $\text{ctx}$ , then the right execution is the honest showing on  $\text{ctx}$  used in  $H_j$ , so both the hybrid and the reduction return  $\perp$ . A discrepancy can therefore occur only if the left execution encodes  $\text{ctx}$ : the real target execution succeeds, whereas the blindness experiment collapses its output to  $(\perp, \perp)$  because the honest right companion failed. Thus the bad event is contained in  $\neg F_L \wedge F_R$ . Since the right execution is honest, correctness of BS gives

$$\Pr[\neg F_L \wedge F_R] \leq \Pr[F_R] \leq \eta_{\text{BS}},$$

where  $\eta_{\text{BS}}$  denotes the correctness error of BS. Consequently, for a suitable PPT adversary,

$$|\Pr[H_{j-1} = 1] - \Pr[H_j = 1]| \leq 2\text{Adv}_{\text{BS}, \mathcal{B}_{\text{bld},j}}^{\text{BLD}}(\lambda) + \eta_{\text{BS}},$$

where  $\eta_{\text{BS}}$  is negligible by correctness.

After all sessions have been replaced,  $H_{q_{\text{show}}}$  is exactly  $\text{SE-Obliv}_{\Pi, \ell, \text{Sim}, \mathcal{L}_{\perp}}^{\mathcal{A}, 0}(\lambda)$ . A hybrid argument therefore gives

$$\text{Adv}_{\Pi, \ell, \text{Sim}, \mathcal{L}_{\perp}, \mathcal{A}}^{\text{SE-Obliv}}(\lambda) \leq \sum_{j=1}^{q_{\text{show}}} 2\text{Adv}_{\text{BS}, \mathcal{B}_{\text{bld},j}}^{\text{BLD}}(\lambda) + q_{\text{show}}\eta_{\text{BS}},$$

which is negligible by blindness and correctness of BS.

## 6 Concrete Instantiation and Evaluation

### 6.1 Concrete Instantiation

*Building blocks.* We instantiate  $\text{Sig}_I$ ,  $\text{Sig}_A$ , and the device-side signature with CAPSS [FR26a] under independent keys, reusing and adapting its reference implementation, and commit to attributes using a Merkle tree built from the Griffin permutation [GHR<sup>+</sup>23] and Jive compression mode [BBC<sup>+</sup>23], following the CAPSS Merkle-tree construction [FR26a], and prove the resulting RICS relation with Aurora [BCR<sup>+</sup>19]. We also evaluate the same exported relation with the Liger-based proof backend from Longfellow by Frigo and shelat [Fs26]. A separate Griffin/Jive hash domain is used for the presentation context. The attributes  $\text{Attrs} = (\text{Attrs}_1, \dots, \text{Attrs}_{\ell}) \in \mathbb{F}^{\ell}$  form the leaves of a fixed-shape Griffin/Jive Merkle tree with root  $\text{cmt} := \text{MerkleRoot}_G(\text{Attrs}_1, \dots, \text{Attrs}_{\ell})$ . This attribute commitment is deterministic. Since the issuer receives all attributes, it computes this root directly during issuance. The issuer signs the canonical field encoding of  $(\text{cmt}, \text{dpk})$ , thereby binding the attribute root to the device public key. For the device authorization, let  $H_P$  denote a dedicated Griffin/Jive hash domain. The user samples  $r_P \leftarrow \mathbb{F}$  and commits to the presentation context as  $c_P := H_P(\text{Encode}_{\mathbb{F}}(\text{ctx}), r_P)$ . The blind-signature component follows the standard commit–sign–prove paradigm [Fis06]: the user commits to the context and the device signs this commitment. We embed the resulting proof statement directly into the final presentation proof. Thus, the context commitment, its opening, and the device signature are witnesses of the presentation relation, and no standalone blind-signature proof is used.

*Presentation relation.* For verifier context  $\text{ctx}$  and disclosed values  $\mathbf{d} = (d_i)_{i \in I}$ , the public statement is  $x = (\text{ipk}, \text{ctx}, I, \mathbf{d})$  and the credential-branch witness is  $w = (\text{cmt}, \text{dpk}, \text{cred}, c_P, r_P, \sigma_D, \text{opening}_I)$ . The credential relation  $R_{\text{cred}}$  accepts if and only if

$$\begin{cases} \text{CAPSS}_I.\text{Verify}((\text{cmt}, \text{dpk}), \text{cred}, \text{ipk}) = 1, \\ \text{CAPSS}_D.\text{Verify}(c_P, \sigma_D, \text{dpk}) = 1, \\ c_P = H_P(\text{Encode}_{\mathbb{F}}(\text{ctx}), r_P), \\ \forall i \in I : \text{Com}.\text{Verify}(\text{cmt}, i, d_i; \text{opening}_i) = 1. \end{cases}$$

The circuit includes the auxiliary branch of Section 4.1, with  $\text{Sig}_A = \text{CAPSS}$  and  $\text{Encode}(x, \text{dpk})$  incorporating a fixed domain prefix and canonical field encoding. Honest presentations use  $b = 0$ .

*Concrete parameters.* We use the CAPSS instance `G256-5.fast.extended`,  $\ell = 16$  field-element attributes with four disclosed, and a 4-ary Merkle tree. Aurora runs in zero-knowledge mode with a 256-bit Blake2b output; the instantiation targets 128-bit security. The Longfellow backend uses inverse rate 7 and 160 queries, likewise targeting 128-bit security.

## 6.2 Evaluation

*Experimental setup.* We benchmark a feasibility prototype, not a fully optimized system, under WSL2 on an Intel Core Ultra 7 165U laptop with 24 GiB of memory. Both proof backends use one thread, while the Rust signature backend uses 14 threads. Although the Aurora prover is implemented in C++, the overall workflow relies on Python and Sage wrappers; we therefore expect further implementation-level optimization to substantially improve the reported running times. We report a representative successful Aurora execution and two measured presentations using the Longfellow backend after an untimed warm-up; sizes use decimal KB. The verifier receives only the constraint system, public inputs, and proof. The Aurora implementation, executable issuance and presentation demos, and benchmark artifact are available at <https://github.com/Heltorfx/context-blind-bdac-benchmark>.

*Ligero-based proof backend.* We feed the complete exported presentation R1CS, including the auxiliary branch, to a local adapter for the proof implementation from Longfellow. This uses Ligero as a component, rather than benchmarking standalone Ligero. The proof is serialized, deserialized, and accepted by the backend’s verifier. Both backends use the same 63473-constraint relation. This backend reduces the measured prover cost at the expense of a larger proof.

*End-to-end protocol.* Table 2 includes witness preparation, R1CS export, and proving in the user-side time. The Longfellow-backend figure also includes compilation of the exported circuit (1.20 seconds on average). The first static R1CS construction is excluded from both rows. Credential issuance uses the same CAPSS implementation in both cases; it took 0.54 seconds in the Aurora run and 0.34 seconds in the Longfellow-backend run. Component costs and the stage-level decomposition appear in Section 7.

**Table 2.** Presentation costs for the complete protocol ( $\ell = 16$  and four disclosed attributes). Aurora reports a representative execution; The Longfellow backend reports the mean of two measured presentations after warm-up. User time sums witness preparation, export, and prover stages.

| Backend            | Prove (s) | User (s) | Verify (s) | Proof (KB) | Total (KB) |
|--------------------|-----------|----------|------------|------------|------------|
| Longfellow backend | 0.45      | 2.60     | 0.29       | 506.11     | 522.48     |
| Aurora             | 11.28     | 13.18    | 1.37       | 214.65     | 231.02     |

Complete communication includes the verifier request, device exchange, disclosed attributes, and proof. The relation contains 63473 constraints, 64726 variables, and 13 public inputs; Section 7 details the auxiliary-branch cost.

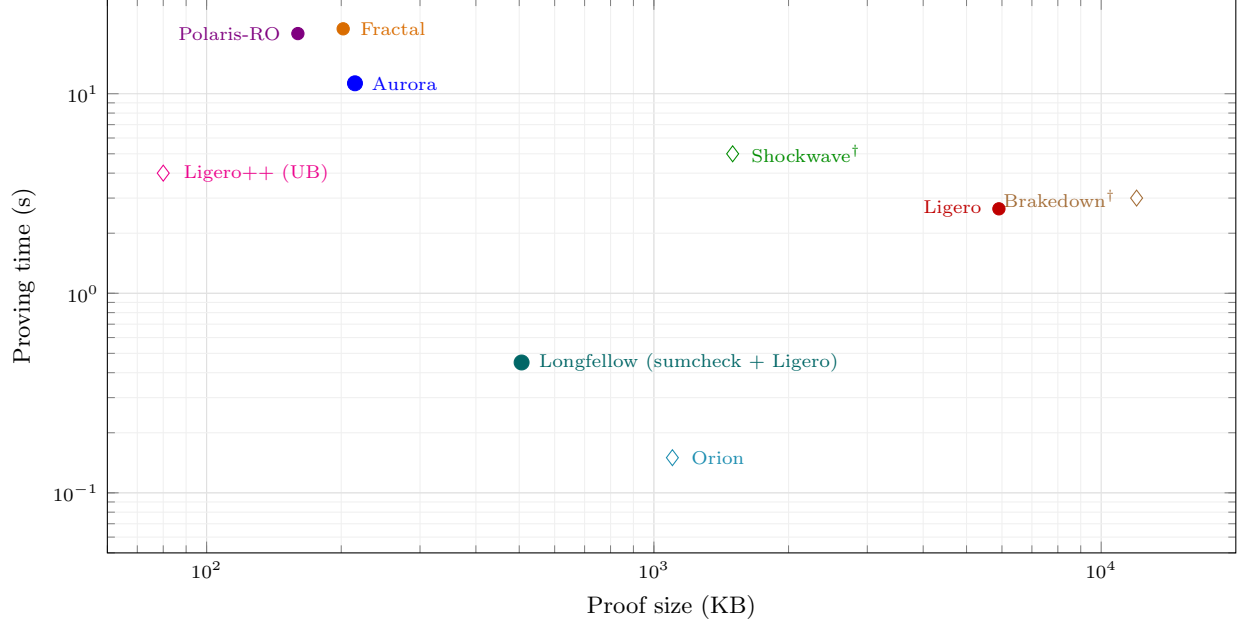


**Fig. 6.** Proof-size/prover-time trade-offs for transparent, plausibly post-quantum proof systems applicable to an R1CS relation near  $2^{16}$  constraints. The Longfellow and Aurora points are local measurements on our complete relation. Other filled circles denote published tabulated values; hollow diamonds denote values read approximately from published figures. † marks implementations that were benchmarked without zero knowledge. Liger++ is a conservative upper bound from its smallest reported instance, which contains approximately 216000 constraints. Platforms, implementations, and parameterizations differ, so the figure illustrates the design space rather than a like-for-like benchmark.

*Proof-system trade-offs.* Figure 6 places our measured Longfellow-backend and Aurora results in the design space of transparent, plausibly post-quantum R1CS proof systems. The external points are taken from the published evaluations of Liger and Liger++ [AHIV17, BFH<sup>+</sup>20], Fractal [COS20], Polaris [FG22], Brakedown and Shockwave [GLS<sup>+</sup>23], and Orion [XZS22].

*Methodology.* Our circuit contains 63473 constraints, so we select results at  $2^{16}$  constraints whenever the source reports powers-of-two R1CS sizes. Liger, Fractal, and Polaris are taken directly from the tables of Fu and Gong [FG22]; Orion [XZS22], Brakedown, and Shockwave [GLS<sup>+</sup>23] are approximate values read from the corresponding  $2^{16}$  points in their published figures. The Orion point reproduces the performance reported for the original construction; subsequent work identified a soundness issue and proposed a repair that preserves linear prover complexity while improving the expected proof size and verifier cost; implementation benchmarks for the repaired construction are not available [dHS25]. The published Brakedown and Shockwave implementations marked by † do not provide zero knowledge; their plotted costs therefore exclude the overhead of a zero-knowledge transformation. We treat these points as optimistic reference costs, not formal lower bounds for a zero-knowledge realization. Liger++ [BFH<sup>+</sup>20] reports Merkle-path benchmarks indexed by the number of SHA-256 evaluations, each accounting for approximately 27000 R1CS constraints. Its smallest reported instance contains eight evaluations, or approximately 216000 constraints, and is therefore shown as a conservative upper bound. Proof sizes are expressed in decimal KB. Because the experiments use different machines, implementations, and parameter choices, the plot illustrates possible trade-offs rather than a like-for-like benchmark or the predicted cost of a drop-in replacement.

The figure illustrates that transparent, plausibly post-quantum proof systems offer markedly different performance trade-offs for R1CS relations of this size. Systems such as Brakedown and Shockwave favor low proving time at the cost of substantially larger proofs, whereas Liger++, Fractal, and Polaris-RO indicate that smaller proofs are possible with a slower prover. Our Longfellow-backend measurement adds a fast-prover, larger-proof point for the actual BDAC relation; Aurora gives a smaller proof with a slower prover. These results suggest that the proof system can be selected according to the application’s primary

constraint, although achieving both very short proofs and very low proving time remains challenging. Because the reported results use different platforms, parameters, and implementation maturity levels, the figure should be interpreted as illustrating available trade-offs rather than providing a direct performance ranking.

*Comparison with post-quantum anonymous credentials.* Table 1 compares sizes, including the 140 KB CAPSS baseline without context-blind device binding. Different relations, models, and platforms preclude a like-for-like comparison of reported times.

## 7 Additional Experimental Results

This section reports component-level measurements that complement the end-to-end evaluation in Section 6.2. All measurements use the setup described there.

### 7.1 End-to-End Breakdown

Witness preparation includes the context commitment, the device interaction, and construction of the proof instance. Both backends prove the same 63473-constraint, 64726-variable relation with 13 public inputs.

*Ligero-based backend from Longfellow.* Table 3 gives the mean costs of two presentations after an untimed warm-up. The current adapter imports and compiles the circuit for each presentation. This cost is included in the user-side time, together with witness preparation, R1CS export, and proving.

**Table 3.** Presentation costs with the Ligero-based proof backend from Longfellow (mean of two measured presentations after warm-up).

| Phase                          | Time   |
|--------------------------------|--------|
| Witness preparation            | 0.24 s |
| R1CS export                    | 0.71 s |
| Circuit import and compilation | 1.20 s |
| Proof generation               | 0.45 s |
| <b>Presentation, user side</b> | 2.60 s |
| Verification                   | 0.29 s |

*Aurora backend.* Table 4 decomposes the representative Aurora execution. Its prover and verifier process timings also include loading the exported R1CS and writing or reading the proof file.

**Table 4.** Aurora presentation costs for the representative compact execution reported in Table 2.

| Phase                            | Time    |
|----------------------------------|---------|
| Presentation witness preparation | 0.26 s  |
| R1CS witness/instance export     | 1.64 s  |
| Aurora prover process            | 11.28 s |
| <b>Presentation, user side</b>   | 13.18 s |
| Aurora verifier process          | 1.37 s  |

For the same disclosure pattern, the credential-and-device circuit without the auxiliary branch contained 62888 constraints and 64139 variables. The augmented relation contains 63473 constraints, 64726 variables, and 13 public inputs. Adding the multiplexed auxiliary branch and conditionally enforcing its validity assertions therefore adds 585 constraints and 587 variables in the current R1CS compiler, because deterministic intermediate computations are shared between the two branches.

## 7.2 CAPSS Component Costs

Table 5 reports the cost of the CAPSS instance used for both issuer and device signatures. A public key contains two field elements, whereas a signature contains 505 field elements. The device signature accounts for most of the user–device communication during presentation.

**Table 5.** CAPSS microbenchmarks for `G256-5.fast.extended` in the representative compact run.

| Operation or object | Time      | Size     |
|---------------------|-----------|----------|
| Key generation      | 2.06 ms   | –        |
| Signing             | 245.67 ms | –        |
| Verification        | 90.02 ms  | –        |
| Public key          | –         | 0.064 KB |
| Signature           | –         | 16.16 KB |

To separate implementation effects from protocol costs, we also ran the implementations distributed with CAPSS [FR26a] on the same machine. Table 6 makes the provenance of each implementation explicit. The first two rows use the same BN254 Griffin-5 parameters and signature representation. The Anemoui rows use different CAPSS instances and document the design space exposed by the CAPSS artifact; they are not drop-in replacements for our presentation circuit.

**Table 6.** Local CAPSS implementation comparison (mean of 30 executions).

| Source                 | Implementation   | Field/permutation             | Threads | Signature (KB) | Sign      | Verify   |
|------------------------|------------------|-------------------------------|---------|----------------|-----------|----------|
| This work              | Rust backend     | BN254/Griffin-5 fast extended | 14      | 16.160         | 245.67 ms | 90.02 ms |
| CAPSS artifact [FR26a] | Reference Python | BN254/Griffin-5 fast extended | 1       | 16.160         | 7.692 s   | 536 ms   |
| CAPSS artifact [FR26a] | Optimized C      | BN254/Anemoui-5 fast          | 1       | 12.539         | 863 ms    | 52.0 ms  |
| CAPSS artifact [FR26a] | Optimized C      | Goldilocks/Anemoui-7 fast     | 1       | 10.720         | 42.8 ms   | 2.36 ms  |

Our fixed-width Rust backend preserves the Griffin parameters and serialized signatures of the CAPSS reference Python implementation while moving its dominant leaf-hashing and Merkle-compression work out of Sage. Changing to an Anemoui instance or another field would require a matching R1CS gadget and a compatible presentation-proof field.

## 7.3 Aurora Presentation-Proof Cost

The representative compact proof has size 214.65 KB. Proof sizes are not intrinsically constant because transcript-dependent query positions change the amount of Merkle authentication data shared between openings. A larger compact-configuration campaign is needed to characterize this variability. The main remaining costs are Aurora proof generation, R1CS serialization between Sage and the C++ backend, and proof size; the serialization cost is an integration overhead rather than an inherent cryptographic cost.

**Acknowledgments.** This work was supported in part by the EU Digital Europe Programme under Grant Agreement No. 101190366 (PiQASO) and the project LIONS within dtcc.bw, funded by the European Commission under the NextGenerationEU program.

## References

- AGJ<sup>+</sup>24. Sven Argo, Tim Güneysu, Corentin Jeudy, Georg Land, Adeline Roux-Langlois, and Olivier Sanders. Practical post-quantum signatures for privacy. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and

- David Lie, editors, *ACM CCS 2024: 31st Conference on Computer and Communications Security*, pages 1523–1537, Salt Lake City, UT, USA, October 14–18, 2024. ACM Press. 1, 4, 6
- AHIV17. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligerio: Lightweight sublinear arguments without a trusted setup. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 2087–2104, Dallas, TX, USA, October 31 – November 2, 2017. ACM Press. 4, 26
- ARS20. Behzad Abdolmaleki, Sebastian Ramacher, and Daniel Slamanig. Lift-and-shift: Obtaining simulation extractable subversion and updatable SNARKs generically. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020: 27th Conference on Computer and Communications Security*, pages 1987–2005, Virtual Event, USA, November 9–13, 2020. ACM Press. 5
- BBC<sup>+</sup>23. Clémence Bouvier, Pierre Briaud, Pyrros Chaidos, Léo Perrin, Robin Salen, Vesselin Velichkov, and Danny Willems. New design techniques for efficient arithmetization-oriented hash functions: Anemioi permutations and Jive compression mode. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 507–539, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland. 3, 24
- BBC<sup>+</sup>24. Carsten Baum, Olivier Blazy, Jan Camenisch, Jaap-Henk Hoepman, Eysa Lee, Anja Lehmann, Anna Lysyanskaya, René Mayrhofer, Hart Montgomery, Ngoc Khanh Nguyen, Bart Preneel, abhi shelat, Daniel Slamanig, Stefano Tessaro, Søren Eller Thomsen, and Carmela Troncoso. Cryptographers’ feedback on the EU digital identity’s ARF. [https://hpi.de/oldsite/fileadmin/user\\_upload/fachgebiete/lehmann/Publications/cryptographers-feedback.pdf](https://hpi.de/oldsite/fileadmin/user_upload/fachgebiete/lehmann/Publications/cryptographers-feedback.pdf), June 2024. Position Paper / Consensus Opinion. 1
- BCC04. Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *ACM CCS 2004: 11th Conference on Computer and Communications Security*, pages 132–145, Washington, DC, USA, October 25–29, 2004. ACM Press. 4
- BCR<sup>+</sup>19. Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 103–128, Darmstadt, Germany, May 19–23, 2019. Springer, Cham, Switzerland. 3, 24
- BFH<sup>+</sup>20. Rishabh Bhaduria, Zhiyong Fang, Carmit Hazay, Muthuramakrishnan Venkitasubramaniam, Tiancheng Xie, and Yupeng Zhang. Ligerio++: A new optimized sublinear IOP. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020: 27th Conference on Computer and Communications Security*, pages 2025–2038, Virtual Event, USA, November 9–13, 2020. ACM Press. 26
- BGM17. Sean Bowe, Ariel Gabizon, and Ian Miers. Scalable multi-party computation for zk-SNARK parameters in the random beacon model. Cryptology ePrint Archive, Report 2017/1050, 2017. 5
- BLNS23a. Ward Beullens, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Lattice-based blind signatures: Short, efficient, and round-optimal. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *ACM CCS 2023: 30th Conference on Computer and Communications Security*, pages 16–29, Copenhagen, Denmark, November 26–30, 2023. ACM Press. 7
- BLNS23b. Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Alessandro Sorniotti. A framework for practical anonymous credentials from lattices. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part II*, volume 14082 of *Lecture Notes in Computer Science*, pages 384–417, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland. 1, 4, 6
- BRW26. Benedikt Bünz, Ron D. Rothblum, and William Wang. Flock: Fast proving for batch boolean computations. Cryptology ePrint Archive, Report 2026/1329, 2026. 4
- CDE<sup>+</sup>23. Liqun Chen, Changyu Dong, Nada El Kassem, Christopher J. P. Newton, and Yalan Wang. Hash-based direct anonymous attestation. In Thomas Johansson and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 14th International Workshop, PQCrypto 2023*, pages 565–600, College Park, USA, August 16–18, 2023. Springer, Cham, Switzerland. 4
- CDG<sup>+</sup>17. Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Reicherberger, Daniel Slamanig, and Greg Zaverucha. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 1825–1842, Dallas, TX, USA, October 31 – November 2, 2017. ACM Press. 5
- CF13. Dario Catalano and Dario Fiore. Vector commitments and their applications. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013: 16th International Conference on Theory and Practice of Public Key Cryptography*, volume 7778 of *Lecture Notes in Computer Science*, pages 55–72, Nara, Japan, February 26 – March 1, 2013. Springer Berlin Heidelberg, Germany. 8
- CF24. Alessandro Chiesa and Giacomo Fenzi. zkSNARKs in the ROM with unconditional UC-security. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024: 22nd Theory of Cryptography Conference*,

- Part I*, volume 15364 of *Lecture Notes in Computer Science*, pages 67–89, Milan, Italy, December 2–6, 2024. Springer, Cham, Switzerland. 5
- Cha90. David Chaum. Showing credentials without identification transferring signatures between unconditionally unlinkable pseudonyms. In Jennifer Seberry and Josef Pieprzyk, editors, *Advances in Cryptology – AUSCRYPT’90*, volume 453 of *Lecture Notes in Computer Science*, pages 246–264, Sydney, Australia, January 8–11, 1990. Springer Berlin Heidelberg, Germany. 1
- CHE24. Liqun Chen, Patrick Hough, and Nada El Kassem. Collaborative, segregated NIZK (CoSNIZK) and more efficient lattice-based direct anonymous attestation. *Cryptology ePrint Archive*, Report 2024/864, 2024. 4
- CL01. Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfizmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118, Innsbruck, Austria, May 6–10, 2001. Springer Berlin Heidelberg, Germany. 1, 3
- COS20. Alessandro Chiesa, Dev Ojha, and Nicholas Spooner. Fractal: Post-quantum and transparent recursive proofs from holography. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 769–793, Zagreb, Croatia, May 10–14, 2020. Springer, Cham, Switzerland. 26
- DDO<sup>+</sup>01. Alfredo De Santis, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano, and Amit Sahai. Robust non-interactive zero knowledge. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 566–598, Santa Barbara, CA, USA, August 19–23, 2001. Springer Berlin Heidelberg, Germany. 5
- dHLM<sup>+</sup>26. Thomas den Hollander, Shai Levin, Marzio Mula, Robi Pedersen, Daniel Slamanig, and Sebastian A. Spindler. SPRINT: New isogeny proofs of knowledge and isogeny-based signatures. *Cryptology ePrint Archive*, Paper 2026/364, to appear at ASIACRYPT 2026, 2026. 5
- dHS25. Thomas den Hollander and Daniel Slamanig. A crack in the firmament: Restoring soundness of the orion proof system and more. In Goichiro Hanaoka and Bo-Yin Yang, editors, *Advances in Cryptology – ASIACRYPT 2025, Part V*, volume 16249 of *Lecture Notes in Computer Science*, pages 396–426, Melbourne, VIC, Australia, December 8–12, 2025. Springer, Singapore, Singapore. 26
- FG22. Shihui Fu and Guang Gong. Polaris: Transparent succinct zero-knowledge arguments for RICS with efficient verifier. *Proceedings on Privacy Enhancing Technologies*, 2022(1):544–564, January 2022. 26
- FHLL26. Karla Friedrichs, Franklin Harding, Anja Lehmann, and Anna Lysyanskaya. Device-bound anonymous credentials with(out) trusted hardware. In Joan Daemen and Emmanuel Thomé, editors, *Advances in Cryptology – EUROCRYPT 2026, Part II*, volume 16542 of *Lecture Notes in Computer Science*, pages 345–375, Rome, Italy, May 10–14, 2026. Springer, Cham, Switzerland. 2, 3, 6, 8, 11, 12, 14
- Fis06. Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In Cynthia Dwork, editor, *Advances in Cryptology – CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 60–77, Santa Barbara, CA, USA, August 20–24, 2006. Springer Berlin Heidelberg, Germany. 24
- FR26a. Thibault Feneuil and Matthieu Rivain. CAPSS: A framework for SNARK-friendly post-quantum signatures. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2026(4):731–758, September 2026. 1, 2, 4, 6, 24, 28
- FR26b. Thibault Feneuil and Matthieu Rivain. SmallWood: Hash-based polynomial commitments and zero-knowledge arguments for relatively small instances. *IACR Communications in Cryptology*, 3(2), 2026. 4
- Fs26. Matteo Frigo and abhi shelat. Anonymous credentials from ECDSA. *IACR Communications in Cryptology*, 3(1), 2026. 1, 3, 4, 24
- GHR<sup>+</sup>23. Lorenzo Grassi, Yonglin Hao, Christian Rechberger, Markus Schafneggger, Roman Walch, and Qingju Wang. Horst meets fluid-SPN: Griffin for zero-knowledge applications. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 573–606, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland. 3, 24
- GLS<sup>+</sup>23. Alexander Golovnev, Jonathan Lee, Srinath T. V. Setty, Justin Thaler, and Riad S. Wahby. Brakedown: Linear-time and field-agnostic SNARKs for RICS. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part II*, volume 14082 of *Lecture Notes in Computer Science*, pages 193–226, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland. 26
- Gro06. Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology – ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 444–459, Shanghai, China, December 3–7, 2006. Springer Berlin Heidelberg, Germany. 5

- JRS23. Corentin Jeudy, Adeline Roux-Langlois, and Olivier Sanders. Lattice signature with efficient protocols, application to anonymous credentials. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part II*, volume 14082 of *Lecture Notes in Computer Science*, pages 351–383, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland. 1, 4
- KZM<sup>+</sup>15. Ahmed Kosba, Zhichao Zhao, Andrew Miller, Yi Qian, Hubert Chan, Charalampos Papamanthou, Rafael Pass, abhi shelat, and Elaine Shi.  $C\emptyset c\emptyset$ : A framework for building composable zero-knowledge proofs. Cryptology ePrint Archive, Report 2015/1093, 2015. 5
- LCL<sup>+</sup>23. Qiqi Lai, Chongshen Chen, Feng-Hao Liu, Anna Lysyanskaya, and Zhedong Wang. Lattice-based commitment-transferrable signatures and applications to anonymous credentials. Cryptology ePrint Archive, Report 2023/766, 2023. 1, 4, 6
- LRSW99. Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In Howard M. Heys and Carlisle M. Adams, editors, *SAC 1999: 6th Annual International Workshop on Selected Areas in Cryptography*, volume 1758 of *Lecture Notes in Computer Science*, pages 184–199, Kingston, Ontario, Canada, August 9–10, 1999. Springer Berlin Heidelberg, Germany. 1, 3
- LSS24. Vadim Lyubashevsky, Gregor Seiler, and Patrick Steuer. The LaZer library: Lattice-based zero knowledge and succinct proofs for quantum-safe privacy. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *ACM CCS 2024: 31st Conference on Computer and Communications Security*, pages 3125–3137, Salt Lake City, UT, USA, October 14–18, 2024. ACM Press. 4
- PPS21. Chris Peikert, Zachary Pepin, and Chad Sharp. Vector and functional commitments from lattices. In Kobbi Nissim and Brent Waters, editors, *TCC 2021: 19th Theory of Cryptography Conference, Part III*, volume 13044 of *Lecture Notes in Computer Science*, pages 480–511, Raleigh, NC, USA, November 8–11, 2021. Springer, Cham, Switzerland. 8
- PWFW23. Guru-Vamsi Policharla, Bas Westerbaan, Armando Faz-Hernández, and Christopher A Wood. Post-quantum privacy pass via post-quantum anonymous credentials. Cryptology ePrint Archive, Report 2023/414, 2023. 1, 4, 6
- Sah01. Amit Sahai. Simulation-sound non-interactive zero knowledge. Technical Report RZ 3076, IBM Research, 2001. 5
- XZS22. Tiancheng Xie, Yupeng Zhang, and Dawn Song. Orion: Zero knowledge proof with linear prover time. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part IV*, volume 13510 of *Lecture Notes in Computer Science*, pages 299–328, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland. 26
- ZSE<sup>+</sup>24. Xinyu Zhang, Ron Steinfeld, Muhammed F. Esgin, Joseph K. Liu, Dongxi Liu, and Sushmita Ruj. Loquat: A SNARK-friendly post-quantum signature based on the Legendre PRF with applications in ring and aggregate signatures. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part I*, volume 14920 of *Lecture Notes in Computer Science*, pages 3–38, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland. 4