Privacy-Preserving Identity Systems

No Author Given

No Institute Given

1 Introduction

Digital Identity systems form the foundation of online trust and authentication, processing billions of verifications daily [?, ?]. Traditional centralized systems, while effective for regulatory compliance [?], suffer from significant privacy and security vulnerabilities. Ongoing data breaches [?] affect billions of users, demonstrating the risks of storing identity data and centralizing systems. Decentralized Identity (DID) is a loosely defined framework (W3C specification) having evolved over the past 2 decades, offering users greater control over their credentials [?], many concrete implementations struggle to balance privacy with accountability [?] for the required feature set.

Anonymous Credential Systems (ACS) [?, ?, ?, ?, ?] address privacy concerns but face challenges balancing privacy with accountability, orthogonally, unconditionally anonymous payment systems have demonstrated how unconditional anonymity can enable system abuse. Current systems focus on protecting against sybil attacks [?, ?], enabling revocation [?, ?, ?, ?], rich attribute-based credential authentication [?, ?] but very few implement all. The tension between privacy and accountability has become increasingly critical as governments worldwide, particularly the EU's Digital Identity Framework [?], move toward privacy-preserving digital identity wallets. These challenges motivate the need for a comprehensive identity system that achieves privacy, accountability, and practical deployment requirements simultaneously.

1.1 Organization

Short Summary: We build a new anonymous credential building block for multi-issuer, multi-credential systems. We use it to build a new private digital identity system with sybil resistance and revocation.

1. section ?? is the new anonymous credential system building block for multi-issuer, multi-credential
2. section ?? outlines all sigma zkp's used
3. section ?? builds a private identity system from the building block in section ?? and proofs in ??
4. section ?? shows privacy and sybil resistance can be low-overhead, also shows our benchmarks for proving complex statements about credentials is more efficient than SOTA

1.2 Related Work

Decentralized Identity (DID) enables entities to create and manage digital identities without relying on a central authority. W3C specifications for DID and Verifiable Credentials define standards for globally unique, publicly verifiable credentials, allowing a user to prove claims (information) about their identity attributes. DID typically uses Distributed Ledgers and public key cryptography to establish a "web of trust" and maintain revocation registries.

The DID model consists of

1. **Holders** Identities with Decentralized Identifiers (DID's) who manage their own keys, and credentials, and request access to resources
2. **Issuers** create and sign credentials about identity holders
3. **Verifiers** validate credentials by checking the presented cryptographic information against the registry
4. **Verifiable Data Registry**, often a DLT, is the root of trust, maintaining DID records, keys, and credential schemas, but doesn't store credential data
5. **Identity Wallet**: the user interface for storing, managing, and presenting verifiable credentials

Citations - U-Prove, U-Port, Connect.me, Sovrin, PingID, w3c

While the W3C DID specifications outline core functionalities such as cryptographic verification, privacy preservation, selective disclosure, and revocation, it requires a formal security definition and proofs to achieve these properties. Anonymous Credential Systems provide well-established formal security definitions for many properties that DID aims to achieve. Specifically, Correctness, Unforgeability, Anonymity, Sybil Resistance, and Revocation. By building DID systems on top of Anonymous Credential primitives, DID systems can inherit these formal security guarantees.

Multi Attribute Anonymous Credentials Anonymous Credentials are a long line of orthogonal work with the goal of providing privacy/anonymity to online interactions, These are now deployed in a number of real-world systems (U-Prove, Idemix, PrivacyPass). The line of work has improved with features, efficiency, and expressiveness throughout time. Abstractly, they provide private protocols to prove information on committed attributes. Pairing-based blind signatures enabled efficient multi-show credentials (CL, PS, BBS+). Other constructions are based on (chase's MAC's, etc) Further work was done to the Anonymous Credential primitive to support delegation, update, and pseudonyms and may also require incorporating other cryptographic primitives to enable revocation, auditing, tracing, and sybil resistance.

Anonymous Credential Systems (ACS) Anonymous Credential Systems implement primitives together in ways that preserve privacy and offer additional functionality required by systems. The combination of multiple primtiives to be used together in a privacy preserving way is complex it itself.

Accountable Privacy In a parallel industry, fully private financial systems like zcash, monero, and tornado cash where blockchain users can exploit and misuse the system is a big problem and will prevent governments and organisations to deploy private preserving techniques. Innovations in Identity often follow those in financial systems, (Chaum's e-cash and blind signatures, DAC and ZCash for example) as they share similar motivations and infrastructure, spending a coin anonymously is analogous to using an identity anonymously. Therefore, advancements in the field of Accountable Privacy (UTT, other examples), a series of techniques and protocols that balance the tradeoffs between privacy and accountability in financial systems, can inspire and motivate private identity systems.

Privacy Preserving Decentralized Identity Systems CanDID proposed a privacy-preserving decentralized identity system achieving Sybil Resistance, Accountability. They defined a system architecture that has been extended in many different directions, such as [?,?] optimising for blockchain performance and additional private-accountability features. [?, ?] optimize for non-interactivity and minimal stored state with the CanDID threshold committee

CanDID leaves integrating Decentralized Identity with Anonymous Credentials as open work.

Decentralized Identity with Anonymous Credentials The strawman approach to combining Decentralized Identity with Anonymous Credentials replaces the existing credential, usually in the form of a signature over attributes, signed by an identity provider and verifiable with their public key, with a blind signature over attributes, which introduces the accountable privacy problems such as how to issue and verify a credential to an anonymous user, and how to retain accountability of an anonymous user, such are the real-world identity system requirements. Compatibility for a decentralized architecture requires decentralized cryptography which has propelled the use of threshold cryptography in anonymous credential schemes and other settings e.g. (Coconut, Threshold BBS+, ...).

1.3 Contributions

We present a privacy-preserving decentralized identity system for multi-issuer environments. Our system combines anonymous credential primitives with decentralized identity architecture, achieving security and privacy properties that are challenging to realize when naively combining these building blocks. Our main contributions are:

1. A privacy-preserving identity system that enables secure credential chaining and complex anonymous identity verification across multiple issuers.

2. A novel zero-knowledge building block that enables private proofs of VRF derivations from committed messages.

3. A complete system implementation using PS Signatures over commitments that achieves sybil resistance and revocation while enabling multi-issuer credential chaining - validated through concrete implementations, benchmarks, and formal security analysis.

1.4 Gap Analysis

Digital identity is undergoing a fundamental transformation, evolving across three frontiers: decentralization, mandatory institutional adoption, and the emergence of attestation services. Identity systems are evolving from trusted, single-issuer models where a user authenticates with a single authority toward a decentralized paradigm where users publicly verify any multitude of credentials, manage multiple credentials for diverse issuers with their digital wallets.

As traditional organizations increasingly adopt decentralized identity capabilities and while it's also being mandated in the EU, they seek solutions that minimize changes to their existing infrastructure while enabling new DID capabilities and maintaining regulatory compliance. Beyond traditional organisations, a new frontier of credential issuance is emerging through automated attestation services like TLS Notary, Chainlink's DECO, Brave Browser's distefano, and Sui Labs zkLogin which enable verifiable data to become a credential. This transformation, while powerful, introduces challenges to identity systems run by governments and trusted organisations who require sybil resistance protecation and revocation while maintaining privacy in a system where traditional infrastructure assumptions such as ease of revocation no longer hold.

Core Challenges Evolving from single-issuer to multi-issuer, multi-credential environments introduces several challenges. While existing solutions support private identity systems with anonymity, sybil resistance and revocation for single issuers. The introduction of multiple credentials and their sources transforms solved problems into new challenges. A decentralized system for the frontier of credentials must maintain anonymity across credential presentations, implement cross credential sybil resistance and efficient revocation checks without centralized trust.

Additionally, composing privacy-preserving primitives together to achieve the properties we require introduces complexity. While individual primitives for anonymous credentials, Sybil resistance, and revocation are well understood in isolation, the integration highlights the trilemma of accountable privacy systems - the tension between privacy, accountability, and functionality. The core challenge lies in designing efficient zero knowledge proof systems that combine these primitives in protocols that maintain the security and privacy properties of our system with practical efficiency.

Thirdly arises when users verify attributes from multiple credentials. Secure credential composition is required, while allowing flexible zero knowledge proof attribute attestations and selective disclosure. Lastly, users with multiple credentials require to privately prove their credentials are not revoked, introducing a scaling challenge - enabling efficient zero-knowledge batch proofs of non-membership while maintaining privacy and practical verification times.

1.5 Technical Challenges

Building a privacy-preserving decentralized identity system requires balancing competing requirements: adhereing to strong security and privacy properties while retaining accountability measures and providing efficient verification of complex identity statements. While individual cryptographic primitives exist for many of these properties in isolation, combining them while maintaining security and efficiency introduces technical challenges, we identity three fundamental challenges below:

Sam: Rewrite this - start with rerand sigs over commitments, then extending that for multi-issuer, multi-cred, then using that for identity system

1. **Efficient Rerandomizable Signatures over Commitments** A key technical challenge was designing a signature scheme that efficiently supports both rerandomization and zero-knowledge proofs over-committed attributes. While existing schemes like BBS+, CL, and standard PS provide these properties, we use a customized PS signature with the lowest overhead in the randomization step. Unlike BBS+ and CL04, we maintain compatibility with standard Pedersen Commitments, enabling efficient proofs from standard techniques in the literature.

2. **Sybil-Resistant Context Credential Construction** Designing an efficient mechanism to link context credentials to a master credential while preserving privacy, our solution uses a novel building block that combines a VRF with committed attributes - the user's Master Credential contains a commitment with their VRF key and generates a context credential nullifier with a VRF parameterized by the key and input the context string. The complexity lies in efficiently proving in zero knowledge this nullifier was correctly derived from the committed key present in a valid, unrevoked master credential. This construction enables strong sybil resistance while maintaining unlinkability between presentations.

3. **Efficient Multi-Credential Proofs and Revocation** enabling efficient proofs over multiple credentials while ensuring practical revocation. Our construction leverages Sigma protocols and Pedersen commitments, which, although they scale linearly with the credential attributes, they are extremely efficient in practice and support the most expressive statements. We integrate existing efficient revocation mechanisms that support batch non-membership proofs, allowing multiple credentials to be efficiently verified simultaneously while maintaining anonymity through zero-knowledge proof protocols.

Table 1: Comparison of our construction over previous work.

Features	Multi Issuer	Sybil Resistance	Revocation	Efficient Cred. Chaining[1]	M-ABC[2]	Anonymity[3]
CanDID [?]	✓	✓	✓	✗	✗	✗
SyRA [?]	✗	✓	✗	✗	✗	✓
S3ID [?]	✓	✓	✗	✓	✗[4]	✓
Our Work	✓	✓	✓	✓	✓	✓

[1] Credential Chaining is a user presenting multiple credentials to be verified together for a complex identity statement.
[2] M-ABC is a Multi-Show Attribute Based Credential, allowing a user to satisfy rich, attribute-based identity statements
[3] Anonymity is defined in the Anonymous Credential model, no verifier and issuer (collaborating together) may learn more about the user or their credentials other than what the user discloses and what their credentials verify. Multiple credential verifications are unlinkable.
[4] While possible in S3ID, they mention
[5] Multi-issuer means supporting credentials from different authorities that can be cryptographically linked while preserving privacy

Comparison

Sam: S3ID is inefficient for attribute-based verification, this table doesn't show that

2 Preliminaries

2.1 Notation

Let $\mathbb{N}$ is the set of positive integers $\{1, 2, 3, \ldots\}$. If x is a string, then $|x|$ is its length, if S is a set, $|S|$ denotes its size. If $\lambda \in \mathbb{N}$, then 1^λ denotes the string of λ ones. If n is an integer, then $[n] = \{1, \ldots, n\}$. If S is a set, then $s \leftarrow\$\ S$ denotes the operation of picking an element s of S uniformly at random. \$ denotes randomized algorithm output. We use $z \leftarrow A()$ and $z = A()$ interchangably for deterministic assignment where z is the output of a deterministic algorithm A and $z \leftarrow A(x, y, \ldots)$ is the output of a deterministic algorithm with inputs $x, y, \ldots$.

We write $A(x, y, \ldots : \mathcal{O}_1, \mathcal{O}_2, \ldots)$ as an algorithm taking $(x, y, \ldots)$ as input with access to oracles $(\mathcal{O}_1, \mathcal{O}_2, \ldots)$ and $z \leftarrow\$\ A(x, y, \ldots : \mathcal{O}_1, \mathcal{O}_2, \ldots)$ as the assignment of the output of A to z.

3 Multi Issuer Multi Credential Anonymous Credentials (MIMC-ABC)

We base our Multi Issuer, Multi Credential Multi-show Attribute based Anonymous Credentials off the model in [?]

3.1 Syntax

- $\mathsf{Setup}(1^\lambda) \rightarrow (\mathsf{pp})$ Takes security parameter λ in unary, outputs public parameters pp.

- $\mathsf{OrgKeygen}(\mathsf{pp}, n) \xrightarrow{\$} (\mathsf{osk}, \mathsf{opk})$: is a probabilistic algorithm that takes public parameters pp and n the upper bound of credential attributes. Outputs organisations keypair $(\mathsf{osk}, \mathsf{opk})$

- $\mathsf{UserKeygen}(1^\lambda) \xrightarrow{\$} (\mathsf{usk})$ is a probabilistic algorithm. Outputs user secret key usk consisting of PRF key k and identity string s

- $(\mathsf{ObtainMaster}(usk_m, \mathbf{m}, aux), \mathsf{IssueMaster}(osk_m, cm, aux)) \rightarrow Cred_m$: An interactive protocol. $ObtainMaster$ is a probabilistic algorithm run by a user, inputs secret key, credential attribute vector $\mathbf{m}$, and auxiliary info. $IssueMaster$ is a probabilistic algorithm run by an issuing organization that takes a commitment cm, issuer secret key osk_m, and auxiliary info. Outputs a master credential $Cred_m$ binding cm to the issuer signature.

- $(\mathsf{ObtainContext}(usk_c, \mathbf{m}, Cred_m, aux), \mathsf{IssueContext}(opk_m, Cred'_m, osk_c, cm, aux)) \rightarrow Cred_c$: An interactive protocol. $ObtainContext$ is a probabilistic algorithm run by a user, inputs master credential $Cred_m$ and corresponding organisation public key opk_m, the users new secret key usk_c, credential attribute vector $\mathbf{m}$, and auxiliary info. $IssueContext$ is a probabilistic algorithm run by an issuing organization that takes a commitment cm, issuer secret key osk_c, and auxiliary info. Outputs a credential $Cred_c$ binding cm to the issuer signature.

- $(\mathsf{Show}(\{usk_i, cred_i\}_{i=1}^{n}, \phi), \mathsf{Verify}(\{cred'_i, opk_i\}_{i=1}^{n}, \pi)) \rightarrow \{0, 1\}$ An interactive protocol. $Show$ is a probabilistic algorithm run by a user. Takes n secret keys, n credentials, and predicate statement ϕ. $Verify$ is a deterministic algorithm run by a verifier, takes n randomized credentials $cred'$ and associated public keys opk and proof π. Outputs 1 if verification succeeds, otherwise 0.

3.2 Security Model

We extend the security model from [?] which considers malicious organization keys. We extend it by supporting multiple organisations (issuers) for unforgeability and anonymity. We add the Credential Binding property.

Security Properties Our Multi Issuer, Multi Credential Anonymous Credential system satisfies the following security and privacy properties:

1. **Correctness:** For any honestly generated credentials and valid witness values, verification accepts if and only if (1) the credentials were legitimately issued by the credential issuer. (2) the credentials satisfy the verification predicate ϕ. (3) all commitments are well-formed with respect to their corresponding signed credentials.

2. **Unforgeability**: No probabilistic polynomial time adversary can (1) forge valid credentials for honest users. (2) use credentials belonging to other users. (3) combine multiple credentials to create new ones. (4) replay a credential show from a different user. (5) present a valid multi-credential proof without controlling all constituent credentials. (6) Mix credentials from different users in a single proof.

3. **Anonymity:** Given polynomial-time adversary views of credential verification, no adversary can (1) learn information beyond the public predicate ϕ. (2) link multiple showings of the same credential. (3) For any set of credentials $\{Cred_1, \ldots, Cred_n\}$ satisfying a predicate ϕ, the adversary cannot distinguish whether they belong to the same or different users.

4. **Credential Binding**: For any polynomial-time adversary $\mathcal{A}$, given a set of credentials $\{cred_1, \ldots, cred_n\}$, producing a valid proof π for a statement ϕ that links the credentials is negligible unless (1) all credentials were legitimately issued to the same user with master secret key s, (2) the user knows the opening of all credential commitments, (3) The proof π must demonstrate consistency of s across all credential combinations used

4 Model and Definitions

We begin by presenting a model for privacy-preserving credential systems that supports multiple issuers and credential composition, extending the anonymous credential framework pioneered by [?], extended in group signatures [?] and modelled in an Attribute Based Anonymous Credential System [?].

4.1 Global Variables

At the beginning of each experiment, the following state is maintained:

System Parameters: For each issuer i, either the experiment computes an issuer keypair $(\mathsf{osk}_i, \mathsf{opk}_i)$ or the adversary outputs opk_i. In the anonymity game, there is a challenge bit b, which the adversary must guess.

User Management: The system maintains:

– Sets HU and CU of honest and corrupted users

– Lists UPK, USK tracking user keypairs

– Lists CRED, COM, OWNR recording credentials, their commitments, and ownership

Challenge State: For the anonymity game, sets $\mathcal{J}_{\mathsf{LoR}}$ and $\mathcal{I}_{\mathsf{LoR}}$ track credential tuples and corresponding users during challenge phases.

Our model extends the ABC system from [?] in two ways:

1. *Multi-Issuer Support:* Users can obtain and compose credentials from different issuers while maintaining privacy across contexts

2. *Credential Composition:* Users can prove statements about multiple credentials while preserving unlinkability both within and across showings

The Oracles. The security properties are defined via experiments where an adversary $\mathcal{A}$ interacts with the following oracles:

$\mathcal{O}_{\mathsf{HU}}(i)$ - By calling this honest user oracle with identity $i \in \mathbb{N}$, $\mathcal{A}$ can add i to the set HU of honest users. The oracle generates $(\mathsf{usk}[i], \mathsf{upk}[i]) \leftarrow\$ \mathsf{UserKeyGen}(\mathsf{opk})$, adds i to HU, and returns $\mathsf{upk}[i]$.

$\mathcal{O}_{\mathsf{CU}}(i, \mathsf{upk})$ - By calling this corrupt user oracle with identity i and optional public key upk, $\mathcal{A}$ can either register a new corrupt user (if $i \notin \mathsf{HU}$) or corrupt an honest user (if $i \in \mathsf{HU}$). In the latter case, $\mathcal{A}$ receives $\mathsf{usk}[i]$ and all credentials.

$\mathcal{O}_{\mathsf{ObtIss}}(i, \mathbf{attrs}, aux)$ - By calling this obtain-issue oracle with identity i, attribute vector $\mathbf{attrs}$, and auxiliary information aux, $\mathcal{A}$ initiates an honest credential issuance. If $i \notin \mathsf{HU}$, returns $\perp$. Otherwise, computes $(\mathsf{cm}, r) \leftarrow\$ \mathsf{Com}(\mathbf{attrs})$ and runs:

$$(\mathsf{cred}, \top) \leftarrow\$ (\mathsf{Obtain}(\mathsf{usk}[i], \mathsf{opk}, \mathsf{cm}, r), \mathsf{Issue}(\mathsf{upk}[i], \mathsf{osk}, \mathsf{cm}))$$

If $\mathsf{cred} = \perp$, returns $\perp$. Else appends $(i, \mathsf{cred}, \mathsf{cm})$ to $(\mathsf{OWNR}, \mathsf{CRED}, \mathsf{COM})$.

$\mathcal{O}_{\mathsf{Obtain}}(i, \mathsf{cm})$ - Models a malicious issuer interacting with an honest user $i \in \mathsf{HU}$. Takes a commitment cm to attributes. If $i \notin \mathsf{HU}$, returns $\perp$. Runs:

$$(\mathsf{cred}, \cdot) \leftarrow\$ (\mathsf{Obtain}(\mathsf{USK}[i], \mathsf{opk}, \mathsf{cm}), \cdot)$$

where Issue is executed by $\mathcal{A}$. On success, appends $(i, \mathsf{cred}, \mathsf{cm})$ to $(\mathsf{OWNR}, \mathsf{CRED}, \mathsf{COM})$.

$\mathcal{O}_{\mathsf{Issue}}(i, \mathsf{cm})$ - Models a malicious user interacting with an honest issuer. Takes identity i and commitment cm. If $i \notin \mathsf{CU}$, returns $\perp$. Runs:

$$(\cdot, I) \leftarrow\$ (\cdot, \mathsf{Issue}(\mathsf{UPK}[i], \mathsf{osk}, \mathsf{cm}))$$

where Obtain is executed by $\mathcal{A}$. If $I = \perp$, it returns $\perp$. Else, it appends $(i, \perp, \mathsf{cm}$ to $(\mathsf{OWNR}, \mathsf{CRED}, \mathsf{COM})$ and returns $\perp$.

$\mathcal{O}_{\mathsf{Show}}(\mathsf{cred}, \phi)$ - Models credential verification with malicious verifier. Takes credential cred and predicate ϕ. Let $i \leftarrow \mathsf{OWNR}[\mathsf{cred}]$. If $i \notin \mathsf{HU}$, returns $\perp$. Runs:

$$(S, \cdot) \leftarrow\$ (\mathsf{Show}(\mathsf{opk}, \phi, \mathsf{cred}), \cdot)$$

where Verify is executed by $\mathcal{A}$.

$\mathcal{O}_{\mathsf{LoR}}(\mathsf{Creds}_0, \mathsf{Creds}_1, \phi)$ - By calling this challenge oracle, $\mathcal{A}$ attempts to distinguish between two credential sets in an anonymous showing. Each $\mathsf{Creds}_b = (\mathsf{cred}_{b,1}, \ldots, \mathsf{cred}_{b,n})$ represents a tuple of credentials, where each credential contains a commitment to its attributes. The oracle enforces that credentials within each tuple belong to the same user (binding) while maintaining unlinkability across showings. The predicate $\phi : \mathsf{COM}^n \to \{0,1\}$ validates statements over committed attributes. If $\mathcal{J}_{\mathsf{LoR}} \neq \emptyset$ and $\mathcal{J}_{\mathsf{LoR}} \neq \{\mathsf{Creds}_0, \mathsf{Creds}_1\}$, returns $\perp$. Otherwise, verifies single ownership and honest user conditions, then executes: $(S, \cdot) \leftarrow\$ (\mathsf{Show}(\mathsf{opk}, \phi, \mathsf{Creds}_b), \cdot)$

Definition 1 (MIMC-ABC Correctness). *A Multi-Issuer Multi-Credential Anonymous Credential system (MIMC-ABC) is correct if for all security parameters $\lambda > 0$, the following properties hold for all honestly generated credentials where attributes are committed using commitment scheme* Com

1. *Individual Validity: For each credential $cred_i$ with commitment $cm_i = Com(attrs_i; r_i)$, the credential verifies independently under its issuer's public key opk_i*

2. *Multi-Issuer Composition: For any set of credentials $cred_1, ..., cred_n$ with corresponding commitments $cm_1, ..., cm_n$ belonging to the same user, any valid predicate ϕ over their committed attributes verifies successfully and ϕ enforces the user owning credentials*

3. *Commitment Binding: The showing protocol preserves the binding between credentials and their committed attributes while maintaining zero-knowledge*

More formally:

$$\Pr\left[\begin{array}{l} (osk_i, opk_i) \leftarrow\$ \, OrgKeyGen(1^\lambda \text{ for } i \in [n]), \\ (usk, upk) \leftarrow\$ \, UserKeyGen(opk), \\ (cm_i, r_i) \leftarrow\$ \, Com(\mathbf{attrs}_i) \text{ for } i \in [n], \\ (\mathsf{cred}_i, \top) \leftarrow\$ \, (Obtain(usk, opk_i, cm_i), \\ \quad Issue(upk, osk_i, cm_i)) \text{ for } i \in [n], \end{array} : \begin{array}{l} (\top, 1) \leftarrow\$ \, (Show(opk, \phi, \{\mathsf{cred}_i\}_{i \in [n]}), \\ \quad Verify(opk, \phi)) \end{array}\right] = 1$$

Definition 2 (MIMC-ABC Unforgeability). *A Multi-Issuer Multi-Credential Anonymous Credential system provides unforgeability if no probabilistic polynomial-time adversary $\mathcal{A}$ can:*

1. *forge valid credentials for honest users*
2. *use credentials belonging to other users*
3. *combine multiple credentials to create new ones*
4. *replay a credential showing from a different user*

More formally, for any PPT adversary $\mathcal{A}$, the following probability is negligible in λ:

$$\Pr[\mathsf{Exp}^{\mathsf{unf}}_{\mathsf{MIMC\text{-}ABC},\mathcal{A}}(1^\lambda) = 1] \leq \mathsf{negl}(\lambda)$$

MIMC-ABC Unforgeability The experiment $\mathsf{Exp}^{\mathsf{unf}}_{\mathsf{MIMC\text{-}ABC},\mathcal{A}}(1^\lambda)$ proceeds as follows:

1. Setup Phase:
   - For each issuer $i \in [n]$: $(osk_i, opk_i) \leftarrow\!\!\$\ \mathsf{OrgKeyGen}(1^\lambda)$
   - Initialize $\mathsf{HU}, \mathsf{CU} \leftarrow \emptyset$
   - Initialize $\mathsf{CRED}, \mathsf{COM}, \mathsf{OWNR} \leftarrow \emptyset$
2. Query Phase: $\mathcal{A}^{\mathcal{O}}(1^\lambda, \{opk_i\}_{i\in[n]})$ where $\mathcal{O}$ includes:
   - $\mathcal{O}_{\mathsf{HU}}, \mathcal{O}_{\mathsf{CU}}$ (user management)
   - $\mathcal{O}_{\mathsf{ObtIss}}, \mathcal{O}_{\mathsf{Obtain}}, \mathcal{O}_{\mathsf{Issue}}$ (credential operations)
   - $\mathcal{O}_{\mathsf{Show}}$ (showing protocol)
3. Forgery Phase: $\mathcal{A}$ outputs $(\{\mathsf{cred}_i\}_{i\in[k]}, \phi, \pi)$
4. The experiment returns 1 if:
   - $\mathsf{Verify}(opk, \phi, \{\mathsf{cred}_i\}_{i\in[k]}, \pi) = 1$ and
   - At least one of the following holds:
     (a) $\exists i : \mathsf{cred}_i \notin \mathsf{CRED}$ (credential forgery)
     (b) $\exists i, j : \mathsf{OWNR}(\mathsf{cred}_i) \neq \mathsf{OWNR}(\mathsf{cred}_j)$ (mixed ownership)
     (c) $\mathsf{OWNR}(\mathsf{cred}_1) \in \mathsf{HU}$ (honest user credential misuse)

Definition 3 (MIMC-ABC Anonymity). *A Multi-Issuer Multi-Credential Anonymous Credential system provides anonymity if no probabilistic polynomial-time adversary $\mathcal{A}$ can:*

1. *learn any information beyond the public information*
2. *link multiple showings of the same credential*
3. *correlate different credentials belonging to the same user*
4. *identify which credentials were issued by the same issuer*

More formally, for any PPT adversary $\mathcal{A}$, the following advantage is negligible in λ:

$$\mathsf{Adv}^{\mathsf{anon}}_{\mathsf{MIMC\text{-}ABC},\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{anon\text{-}1}}_{\mathsf{MIMC\text{-}ABC},\mathcal{A}}(1^\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{anon\text{-}0}}_{\mathsf{MIMC\text{-}ABC},\mathcal{A}}(1^\lambda) = 1]| \leq \mathsf{negl}(\lambda)$$

MIMC-ABC Anonymity Experiment. The experiment $\mathsf{Exp}^{\mathsf{anon\text{-}b}}_{\mathsf{MIMC\text{-}ABC},\mathcal{A}}(1^\lambda)$ proceeds as follows:

1. Setup Phase:
   - For each issuer $i \in [n]$: $(osk_i, opk_i) \leftarrow\$ \mathsf{OrgKeyGen}(1^\lambda)$
   - Initialize $\mathsf{HU}, \mathsf{CU} \leftarrow \emptyset$
   - Initialize $\mathsf{CRED}, \mathsf{COM}, \mathsf{OWNR} \leftarrow \emptyset$
   - Initialize $\mathcal{J}_{\mathsf{LoR}}, \mathcal{I}_{\mathsf{LoR}} \leftarrow \emptyset$
   - Sample challenge bit $b \leftarrow\$ \{0, 1\}$
2. Query Phase: $\mathcal{A}^{\mathcal{O}}(1^\lambda, \{opk_i\}_{i\in[n]})$ where $\mathcal{O}$ includes:
   - $\mathcal{O}_{\mathsf{HU}}, \mathcal{O}_{\mathsf{CU}}$ (user management)
   - $\mathcal{O}_{\mathsf{ObtIss}}, \mathcal{O}_{\mathsf{Obtain}}, \mathcal{O}_{\mathsf{Issue}}$ (credential operations)
   - $\mathcal{O}_{\mathsf{Show}}$ (showing protocol)
   - $\mathcal{O}_{\mathsf{LoR}}$ (challenge oracle)
3. Challenge Phase: $\mathcal{A}$ can query $\mathcal{O}_{\mathsf{LoR}}(\mathsf{Creds}_0, \mathsf{Creds}_1, \phi)$ where:
   - $\mathsf{Creds}_b = (\mathsf{cred}_{b,1}, ..., \mathsf{cred}_{b,n})$ for $b \in \{0, 1\}$
   - Credentials in each tuple must belong to the same user
   - Both credential sets must satisfy predicate ϕ
4. $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$
5. The experiment returns 1 if $b = b'$

Definition 4 (Credential Binding). *For any PPT adversary $\mathcal{A}$, given a set of credentials $\{\mathsf{cred}_1, ..., \mathsf{cred}_n\}$, the probability of producing a valid proof π for statement ϕ is negligible unless:*

1. *all credentials were legitimately issued to the same user with master secret key s*
2. *the user knows the opening of all credential commitments*
3. *the linking proof demonstrates the same s value across all credentials*

More formally, for any PPT *adversary $\mathcal{A}$,* $\mathsf{Adv}^{\mathsf{bind}}_{\mathsf{MIMC\text{-}ABC},\mathcal{A}}(\lambda)$ *is defined as:*

$$\Pr[\mathsf{Exp}^{\mathsf{bind}}_{\mathsf{MIMC\text{-}ABC},\mathcal{A}}(1^\lambda) = 1] \leq \mathsf{negl}(\lambda)$$

MIMC-ABC Credential Binding Experiment The experiment $\mathsf{Exp}^{\mathsf{bind}}_{\mathsf{MIMC\text{-}ABC},\mathcal{A}}(1^\lambda)$ proceeds as follows:

1. Setup Phase:
   - The challenger generates $(osk_i, opk_i) \leftarrow\$ \mathsf{OrgKeyGen}(1^\lambda)$ for each issuer $i \in [n]$
   - Initializes $\mathsf{HU}, \mathsf{CU} \leftarrow \emptyset$,
   - Initializes $\mathsf{CRED}, \mathsf{COM}, \mathsf{OWNR} \leftarrow \emptyset$.
2. Query Phase: $\mathcal{A}^{\mathcal{O}}(1^\lambda, \{opk_i\}_{i\in[n]})$ where $\mathcal{O}$ includes
   - $\mathcal{O}_{\mathsf{HU}}$, $\mathcal{O}_{\mathsf{CU}}$
   - $\mathcal{O}_{\mathsf{ObtIss}}$, $\mathcal{O}_{\mathsf{Obtain}}$
   - $\mathcal{O}_{\mathsf{Issue}}$, and $\mathcal{O}_{\mathsf{Show}}$
3. $\mathcal{A}$ outputs $(\{\mathsf{cred}_i\}_{i\in[n]}, \phi, \pi)$.
4. The experiment returns 1 if:
   - $\mathsf{Verify}(opk, \phi, \{\mathsf{cred}_i\}_{i\in[n]}, \pi) = 1\ \wedge$
   - $\exists\{i, j\} : \mathsf{OWNR}(\mathsf{cred}_i) \neq \mathsf{OWNR}(\mathsf{cred}_j)$ or $\exists i : \mathsf{cred}_i \notin \mathsf{CRED}$

5 MIMC-ABC Construction

5.1 Intuition of our Construction

5.2 Commitment

Syntax

- $CM.Setup(BG, 1^\lambda, 1^n) \to ck$: Sample $g \leftarrow\$ \mathbb{G}_1, \tilde{g} \leftarrow\$ \mathbb{G}_2$. Sample $\mathbf{y} \leftarrow\$ \mathbb{Z}_p^n$ and compute $(\mathbf{g}, \tilde{\mathbf{g}}) \leftarrow (g^{\mathbf{y}}, \tilde{g}^{\mathbf{y}})$. Output $ck \leftarrow (g, \mathbf{g}, \tilde{g}, \tilde{\mathbf{g}})$
- $CM.Com_{ck}(\mathbf{m}, r) \to (\mathsf{cm}, \tilde{cm})$: Parse ck as $(g, \mathbf{g}, \tilde{g}, \tilde{\mathbf{g}})$ and $\mathbf{m}$ as $(m_1, \ldots, m_\ell)$, return $(\mathsf{cm}, \tilde{cm})$ as $(\mathbf{g}^{\mathbf{m}} g^r, \tilde{\mathbf{g}}^{\mathbf{m}} \tilde{g}^r)$
- $CM.Rerand_{ck}((\mathsf{cm}, \tilde{cm}), \Delta_r) \to (\mathsf{cm}', \tilde{cm}')$: Parse ck as $(g, \mathbf{g}, \tilde{g}, \tilde{\mathbf{g}})$. Compute $(g^{\Delta_r}, \tilde{g}^{\Delta_r})$ and return $(\mathsf{cm}', \tilde{cm}')$ as $(\mathsf{cm} \cdot g^{\Delta_r}, \tilde{cm}' \cdot \tilde{g}^{\Delta_r})$

Construction

- $CM.Setup(1^\lambda, n) \to ck$: The setup Algorithm generates the commitment key ck. Samples $(g, \tilde{g}) \leftarrow\$ \mathbb{G}_1^* \times \mathbb{G}_2^*$ and n scalars $y_i \leftarrow \mathbb{Z}_p$, $Y_i \leftarrow g^{y_i}$, and $\tilde{Y}_i \leftarrow \tilde{g}^{y_i}$ $\forall\ 1 \leq i \leq n$; such that $Y_i = g_1^{y_1} \ldots g_n^{y_n}$ and $\tilde{Y}_i = \tilde{g}_1^{y_1} \ldots \tilde{g}_n^{y_n}$. Outputs ck a tuple $(g, Y_i, \tilde{g}, \tilde{Y}_i)$ with commitment keys in both $\mathbb{G}_1$ and $\mathbb{G}_2$ where $e(Y_i, \tilde{g}) = e(g, \tilde{Y}_i)$
- $CM.Com_{ck}(\{m_i\}_{i=1}^n) \to (cm, \tilde{cm})$: to commit to n messages $m_1, \ldots, m_n$, the committer samples $r \leftarrow\$ \mathbb{Z}_p$, and parses ck as $(g, Y_i, \tilde{g}, \tilde{Y}_i)$ and commits to $\{m_i\}_{i=1}^n$ over dual commitments in $\mathbb{G}_1$ and $\mathbb{G}_2$. Computes $g^r, \tilde{g}^r$, Commits to $\{m_i\}_{i=1}^n$ with respect to ck such that $\mathsf{cm} \leftarrow g_1^{m_1} \ldots g_n^{m_n} g^r$ and $\tilde{cm} \leftarrow \tilde{g}_1^{m_1} \ldots \tilde{g}_n^{m_n} \tilde{g}^r$ Then outputs $(\mathsf{cm}, \tilde{cm})$ where the equality of commitments can be verified by pairing $e(\mathsf{cm}, \tilde{g}) = e(g, \tilde{cm})$
- $CM.Rerand_{ck}((\mathsf{cm}, \tilde{cm}), \Delta_r) \to (\mathsf{cm}', \tilde{cm}')$: To rerandomize the commitment, the last element $g, \tilde{g}$ is randomized with Δ_r. Compute $g^{\Delta_r}, \tilde{g}^{\Delta_r}$, Compute $\mathsf{cm}' = \mathsf{cm} \cdot g^{\Delta_r}$ as $g_1^{m_1} \ldots g_n^{m_n} g^r \cdot g^{\Delta_r}$ which will equal $g_1^{m_1} \ldots g_n^{m_n} g^{r+\Delta_r}$ and equivalently for $\tilde{cm}'$. Return the rerandomized commitments $(\mathsf{cm}', \tilde{cm}')$

5.3 PS Signature

Syntax

- $PS.KeyGen(1^\lambda, \mathsf{ck}) \to (\mathsf{sk}, \mathsf{vk})$: Parse ck as $ck \leftarrow (g, \mathbf{g}, \tilde{g}, \tilde{\mathbf{g}})$. Sample $x \leftarrow\$ \mathbb{Z}_p$, set $(\mathsf{sk}, \mathsf{vk}) \leftarrow (g^x, \tilde{g}^x)$
- $PS.Sign_{ck}(\mathsf{sk}, \mathsf{cm}) \to \sigma$: Parse ck as $\leftarrow (g, \cdot, \tilde{g}, \cdot)$ Sample $u \leftarrow\$ \mathbb{Z}_p$, compute $\sigma_1 \leftarrow g^u$, $\sigma_2 \leftarrow (\mathsf{sk} \cdot \mathsf{cm})^u$ and return $\sigma \leftarrow (\sigma_1, \sigma_2)$
- $PS.Rerand(\sigma, \Delta_r, \Delta_u) \to \sigma'$: Parse σ as (σ_1, σ_2). Compute $\sigma_1' \leftarrow \sigma_1^{\Delta_u}$ and $\sigma_2' \leftarrow (\sigma_2 \cdot \sigma_1^{\Delta_r})^{\Delta_u}$. Return $\sigma' \leftarrow (\sigma_1', \sigma_2')$
- $PS.Verify_{ck,vk}(\sigma, (\mathsf{cm}, \tilde{cm})) \to \{0,1\}$: Verify $ZKPoK.Open_{ck}(\mathsf{cm})$. Parse ck as $(g, \cdot, \tilde{g}, \cdot)$ and σ as (σ_1, σ_2). Assert $e(\mathsf{cm}, \tilde{g}) = e(g, \tilde{cm})$ and $e(\sigma_2, \tilde{g}) = e(\sigma_1, \mathsf{vk} \cdot \tilde{cm})$

Construction

- $PS.KeyGen_{\mathsf{ck}}(1^\lambda) \to (\mathsf{sk}, \mathsf{vk})$: The PS Signature KeyGen algorithm is parameterized by the corresponding commitment key ck. The Signer retreives $(g, \cdot, \tilde{g}, \cdot)$ from ck, samples secret $x \leftarrow\$ \mathbb{Z}_p$, sets $X \leftarrow g^x$ and $\tilde{X} \leftarrow \tilde{g}^x$, sets sk as (g, X) and the public verification key vk as $(\mathsf{ck}, \tilde{X})$ and returns $(\mathsf{sk}, \mathsf{vk})$

- $PS.Sign_{\mathsf{sk}}(\mathsf{cm}) \to \sigma$: The signing algorithm signs the commitment. Retreives $(g, \cdot, \tilde{g}, \cdot)$ from ck, Samples $u \leftarrow\!\!\$\ \mathbb{Z}_p$, Computes σ_1 as g^u and $\sigma_2 \leftarrow (X \cdot \mathsf{cm})^u$ both are notably in $\mathbb{G}_1$ and thus $(X \cdot \mathsf{cm})^u = (g_1^{m_1 u} \ldots g_n^{m_n u} g^{xu+ru})$. Returns $\sigma \leftarrow (\sigma_1, \sigma_2)$
- $PS.Rerand(\sigma, \Delta_r, \Delta_u) \to \sigma'$: Parse σ as (σ_1, σ_2). Compute $\sigma_1' \leftarrow \sigma_1^{\Delta_u}$ and $\sigma_2' \leftarrow (\sigma_2 \cdot \sigma_1^{\Delta_r})^{\Delta_u}$. Return $\sigma' \leftarrow (\sigma_1', \sigma_2')$
- $PS.Verify(\sigma, (\mathsf{cm}, \tilde{cm})) \to \{0,1\}$: Verify $ZKPoK.Open_{ck}(\mathsf{cm})$. Parse ck as $(g, \cdot, \tilde{g}, \cdot)$ and σ as (σ_1, σ_2). Assert $e(\mathsf{cm}, \tilde{g}) = e(g, \tilde{cm})$ and $e(\sigma_2, \tilde{g}) = e(\sigma_1, \mathsf{vk} \cdot \tilde{cm})$

5.4 PS Signature over Commitment Construction

- $PS.Rerand_{ck,vk}(\sigma, \Delta_r, \Delta_u) \to \sigma'$: Rerandomization of the signature must preserve the algebraic properties of the commitment itself to allow the commitment to be used for zero knowledge proof protocols. To do so, the commitment is re-randomized with the random factor Δ_r such that a commitment and signature pair

Example

Outline

Freshness

Malicious Organization Keys

ZKPoKs and Concurrent Security

5.5 Master Credential Issuance

The master credential issuance protocol enables a user to obtain their root credential from the Master Credential Oracle $\mathcal{M}$ while preserving the privacy of their VRF key k and ensuring accountability. The protocol combines verifiable encryption, Verifiable Random Function, Zero Knowledge Proofs, and Rerandomizable PS Signatures over commitments to achieve Sybil Resistance, Revocation, and Anonymity.

$\underline{Setup(1^\lambda, 1^n)}$:

System: $(\mathbb{G}_1, \mathbb{G}_2, e, g, \tilde{g}) \leftarrow BG.Gen(1^\lambda, p),\ ck_m \leftarrow CM.Setup(BG, 1^\lambda, n)$

Credential Oracle: $(SK_m, PK_m) \leftarrow PS.KeyGen(ck_m)$

Auditor: $(SK_A, PK_A) \leftarrow TPKE.KeyGen(ck_m)$

Auditor setup Revocation List

$\underline{\text{User}(s)}$ $\underline{\text{MCO}(SK_M)}$

$k_1 \leftarrow \mathbb{Z}_p, \quad C_1 \leftarrow Com([0, k_1, 0, 0], r)$

$\Pi_1 \leftarrow ZKPoK.Prove_{Zeros}(C_1)(k_1, r)$

$$\xrightarrow{C_1, \Pi_1}$$

If $ZKPoK.Verify_{Zeros}(\Pi_1, C_1) = 0$, return $\perp$

$k_2 \leftarrow \mathbb{Z}_p,\ C_2 \leftarrow Com([s, k_2, \text{"}master\text{"}, attrs], 0)$

$C_m \leftarrow C_1 \cdot C_2 = Com([s, k_1 + k_2, \text{"}master\text{"}, attrs], r)$

$$\xleftarrow{C_2, C_m, s, k_2}$$

$k \leftarrow k_1 + k_2$

$\Pi_2 \leftarrow ZKPoK.Prove_{addition}(C_1, C_2, C_m)(k_1, k_2, k, r)$

$\tau \leftarrow Enc(PK_A, k)$

$\Pi_3 \leftarrow ZKPOK.Prove_{enc}(C_m)(\tau, k, r)$

$$\xrightarrow{C_1, C_2, C_m, \Pi_2, \Pi_3, \tau}$$

If $ZKPOK.Verify_{addition}(\Pi_2, C_1, C_2, C_m) = 0$, return $\perp$

If $ZKPOK.Verify_{enc}(C_m)(\Pi_3, PK_A, \tau) = 0$, return $\perp$

$\sigma_m \leftarrow PS.Sign(SK_M, C_1)$

Store Data Record MCO $(C_m, \sigma_m, \tau, \Pi_2, \Pi_3, C_1, C_2, k_2)$

$$\xleftarrow{\sigma_m}$$

If $PS.Verify(PK_A, \sigma_m, C_m) = 1$, Store $Cred_m(\sigma_m, C_m)$

Informal Security Analysis The two-party process between the user and master credential oracle ensures sybil resistance of unique identifier s; the oracle has access to the user information and checks duplicate issuance within their own identity system. During VRF key issuance, the anonymity of the master credential is preserved via the secrecy of the VRF key k. During the two-party computation, the user's share k_1 remains hidden to $\mathcal{M}$ via the hiding property of the commitment C_1, and malicious commitment usage is prevented by the soundness property of Π_1.

The user combines $k_1 + k_2$ to form their VRF key, Π_2 proves k_1 is correctly derived from C_1, k_2 is derived from C_2 and k combines $k_1 + k_2$, $\mathcal{M}$'s input to k prevents forgery attempts on the key k. The hiding property of the commitments and zero-knowledge property of the proofs ensures correct protocol adherence while maintaining private computation. Revocation is enabled by encryption of the VRF key k with the public key of the Auditor PK_A. Π_3 proves that τ is an encryption of the committed key k. τ is stored with the credential oracle maintaining privacy during normal operation. Finally, the protocol prevents replay attacks by using interactive zero-knowledge proofs requiring a challenge from the verifier, fresh commitment randomness, and $\mathcal{M}$'s input of their share of the VRF key k_2 preventing existing transcript reuse.

5.6 Create Context Credential

Context Credential Issuance enables a user to obtain a context-specific credential while proving ownership of a valid master credential. The user first constructs a commitment to their context credential attributes, including their identity s and a deterministic nullifier derived from their VRF key k and the credential context ctx. Through zero-knowledge proofs, the user demonstrates their master credential is valid and unrevoked, and proves the context commitment is well-formed with the same identity s. The nullifier $\tau = VRF(k, ctx)$ prevents multiple credentials for the same context while maintaining privacy. Upon successful verification, the Context Credential Oracle signs the commitment and records the nullifier.

$\underline{Setup(1^\lambda, 1^n)}$:

System: $(\mathbb{G}_1, \mathbb{G}_2, e, g, \tilde{g}) \leftarrow BG.Gen(1^\lambda, p),\ ck_c \leftarrow CM.Setup(BG, 1^\lambda, n)$

Credential Oracle: $(SK_c, PK_c) \leftarrow PS.KeyGen(ck_c)$

$\underline{\text{User}(Cred_m, s, k)}$ $\underline{\text{CCO}(SK_c)}$

$\delta \leftarrow VRF(k, ctx)$

$r* \leftarrow\$ Z_p,\ C_c \leftarrow Com([s, \delta, ctx, attrs], r*)$

$\Pi_4 \leftarrow ZKPoK.Prove_{selective-disclosure}(C_c, \delta, ctx, attrs)(s, r*)$

$r' \leftarrow\$ \mathbb{Z}_p, Cred'_m \leftarrow Cred.Rerand(Cred_m, r')$

$\Pi_5 \leftarrow ZKSoK.Prove(Cred'_m)(s, k, master, attrs, r')$

Parse $Cred'_m$ as C'_m, σ'_m

$\Pi_6 \leftarrow ZKPoK.Prove_{reciprocal}(C'_m, C_c, ctx)(s, k, r', r*)$

$\Pi_7 \leftarrow ZKPoK.Prove_{equality}(C'_m, C_c)(s, r)$

$$\xrightarrow{Cred'_m, C_c, \Pi_4, \Pi_5, \Pi_6, \Pi_7}$$

If

$ZKPok.Verify_{selective-disclosure}(\Pi_4, C_c)(\delta, ctx, attrs) = 1\ \wedge$

$ZKSoK.Verify(\Pi_5, Cred'_m) = 1\ \wedge$

$ZKPoK.Verify_{reciprocal}(\Pi_6, C'_m, C_c, ctx) = 1\ \wedge$

$ZKPoK.Verify_{equality}(\Pi_7, C'_m, C_c) = 1\ \wedge$

$\sigma_c \leftarrow PS.Sign(SK_c, C_c)$

Store Data Record CCO $(C_c, \sigma_c, \delta, \Pi_4, \Pi_5, \Pi_6, \Pi_7)$

$$\xleftarrow{\sigma_c}$$

If $PS.Verify(PK_c, \sigma_c, C_c) = 1$, Store $Cred_c(\sigma_c, C_c)$

Informal Security Analysis Sybil Resistance: The deterministic nullifier $\delta \leftarrow VRF(k, ctx)$ binds each context credential to a unique (user, context) pair, preventing multiple credentials for the same context. The reciprocal proof Π_6 ensures correct nullifier derivation from the master key k. Credential Binding: Context Credentials are bound to master credentials through shared identity s and Π_7. The selective disclosure proof Π_4 ensures correct commitment structure without revealing private values. Privacy: The protocol only reveals ctx and $attrs$ to CCO to allow identity verification while hiding s. The Master Credential $Cred_m$ remains unlinkable by being rerandomized and proven in zero knowledge it verifies with the Master Credential Oracles public key.

Verification A user $user$ wants to prove to any relying party $rely$ they have a valid credential that satisfies a verification statement ϕ. The protocol takes as input $(rcd, ccd, \phi, rpk, acc, n)$ and outputs success or failure. $rely$ starts by sending (ϕ, n, acc) to $user$ where ϕ is a statement that specifies the requirements for a successful verification and acc is the current accumulator value of revoked nullifiers. $user$ starts by randomizing their credentials $rcd' = psrerand(rcd)$ and $ccd' = psrerand(ccd)$ and verifies $psverify_{ck_{rcd}}(rcd')$ and $psverify_{ck_{ccd}}(ccd')$. $user$ generate their nullifiers $nullif_{pid} \leftarrow PRF_s(pid)$ and $nullif_{ctx} \leftarrow PRF_s(ctxid)$ and obtains non-membership witnesses $wpid$, $wctx$ for nullifiers against acc. $user$ generates a zero-knowledge proof π showing 1) their credentials are valid, 2) they're not expired $(expiry > current_time)$, 3) their nullifiers are correctly formed from s, 4) their nullifiers are not in acc using witnesses $wpid, wctx$, 5) the credential attributes satisfy ϕ, 6) proof freshness using n. $user$ sends $(\pi, attrs_\phi)$ to $rely$, $rely$ verifies π against acc and validates $attrs_\phi$ satisfies ϕ. Returns accept if all checks pass, reject otherwise.

5.7 Security Experiments

6 Construction of Σ-protocols

We use the notation of Camenisch and Stadler to represent proofs of knowledge of discrete logarithms relations and the validity of statements about discrete logarithms.

$$\mathcal{R} = \left\{(\alpha, \beta, \gamma) |\ y = g^{\alpha} h^{\beta}\ \wedge\ \tilde{y} = \tilde{g}^{\alpha} h^{\gamma}\right\}$$

The relation $\mathcal{R}$ denotes a zero knowledge proof ofknowledge of private integers α, β, γ such that $\tilde{y} = \tilde{g}^{\alpha} h^{\gamma}$ holds. y, g, h are elements of some group $\mathbb{G}$ and $\tilde{y}, \tilde{g}, \tilde{h}$ are elements of some group $\tilde{\mathbb{G}}$.

1. Completeness: an honest prover who knows the opening will always convince the verifier.
2. Special Soundness: Show that given two accepting transcripts $(T, c, \{s\}_{i=1}^{n}, u)$ and $(T, c', \{s'\}_{i=1}^{n}, u')$ for $c \neq c'$ we can extract a valid witness.
3. Honest Verifier Zero Knowledge: construct a simulator that given a challenge c, produces transcripts indistinguishable from real protocol runs.

Protocol 1: ZKPoK for Pedersen Commitment Opening

Public parameters: $g_1, \ldots, g_n, h \in \mathbb{G}$

Inputs: cm such that $\mathsf{cm} = \prod_{i=1}^n g_i^{m_i} h^r$, $\mathcal{P}$ knows $\{m\}_{i=1}^n, r \in \mathbb{Z}_q^{n+1}$.

1. $\mathcal{P}$ samples $\alpha_i \leftarrow\!\!\$\ \mathbb{Z}_p$ for $i \in [1,n]$, $\rho \leftarrow\!\!\$\ \mathbb{Z}_p$, computes $T \leftarrow (\prod_{i=1}^n g_i^{\alpha_i}) \cdot h^\rho$, sends T to $\mathcal{V}$
2. $\mathcal{V}$ sends challenge $c \leftarrow\!\!\$\ \mathbb{Z}_p$
3. $\mathcal{P}$ computes responses $s_i \leftarrow \alpha_i + cm_i$ for $i \in [1,n]$, and $u \leftarrow \rho + cr$. Sends $\{s_i\}_{i=1}^n$ and u to $\mathcal{V}$
4. $\mathcal{V}$ verifies that $(\prod_{i=1}^n g_i^{s_i}) \cdot h^u = \mathsf{cm}^c T$

Theorem 5 (Perfect Completeness). *Construction ?? is a $\Sigma-$protocol for the relation $\mathcal{R}$*

$$\mathcal{R} = \left\{ (\mathsf{cm}, g_1, \ldots, g_n, h), (m_1, \ldots, m_n, r) \mid \mathsf{cm} = (\prod_{i=1}^n g_i^{m_i}) h^r \right\}$$

with perfect completeness:

Proof. We prove completeness by showing that for any valid inputs and witnesses, when both $\mathcal{P}$ and $\mathcal{V}$ follow the protocol, $\mathcal{V}$ accepts with Pr $= 1$. Consider an execution of the protocol where:

1. $\mathcal{P}$ samples $\alpha_i \leftarrow\!\!\$\ \mathbb{Z}_p$ for $i \in [1,n]$, $\rho \leftarrow\!\!\$\ \mathbb{Z}_p$ and sends $T \leftarrow (\prod_{i=1}^n g_i^{\alpha_i}) h^\rho$
2. $\mathcal{V}$ sends challenge $c \leftarrow\!\!\$\ \mathbb{Z}_p$
3. $\mathcal{P}$ responds with $s_i \leftarrow \alpha_i + cm_i$ for $i \in [1,n]$, $u \leftarrow \rho + cr$

Verification holds by

$$\begin{aligned}
(\prod_{i=1}^n g_i^{s_i}) h^u &\overset{?}{=} \mathsf{cm}^c T \\
(\prod_{i=1}^n g_i^{\alpha_i + cm_i}) h^{\rho + cr} &\overset{?}{=} (\prod_{i=1}^n g_i^{m_i} h^r)^c (\prod_{i=1}^n g_i^{\alpha_i}) h^\rho \\
(\prod_{i=1}^n g_i^{\alpha_i + cm_i}) h^{\rho + cr} &= (\prod_{i=1}^n g_i^{cm_i + \alpha_i}) h^{cr + \rho}
\end{aligned}$$

Thus, an honest verifier always accepts an honest prover's proof.

Theorem 6 (Special Soundness). *Construction 1 satisfies the special soundness of relation $\mathcal{R}$.*

Proof. Let $Tr_1 = (T, c, \{s_i\}_{i=1}^n, u)$ and $Tr_2 = (T, c', \{s_i'\}_{i=1}^n, u')$ be two accepting transcripts for the same initial commitment T where $c \neq c'$. We construct a knowledge extractor $\mathcal{E}$ that extracts the witness $w = (\{m_i\}_{i=1}^n, r)$ as follows:

1. Since both transcripts are accepting, they satisfy: $(\prod_{i=1}^n g_i^{s_i}) h^u = \mathsf{cm}^c T$ and $(\prod_{i=1}^n g_i^{s_i'}) h^{u'} = \mathsf{cm}^{c'} T$

2. Evaluate as a system of linear equations:

$$\frac{(\prod_{i=1}^{n} g_i^{s_i})h^u}{(\prod_{i=1}^{n} g_i^{s'_i})h^{u'}} = \frac{\mathsf{cm}^c}{\mathsf{cm}^{c'}}$$
$$(\prod_{i=1}^{n} g_i^{s_i - s'_i})h^{u-u'} = (\prod_{i=1}^{n} g_i^{m_i(c-c')})h^{r(c-c')}$$

3. By the homomorphic property of the exponents and the uniqueness of discrete logarithm representations, we obtain the system of equations:

$$s_i - s'_i = m_i(c - c') \text{ for } i \in [1, n] \qquad u - u' = r(c - c')$$

therefore $\mathcal{E}$ extracts:

$$m_i = \frac{s_i - s'_i}{c - c'} \text{ for } i \in [1, n] \qquad r = \frac{u - u'}{c - c'}$$

Which satisfies the commitment construction $\mathsf{cm} = (\prod_{i=1}^{n} g_i^{m_i})h^r$

This shows that if $\mathcal{P}$ can respond correctly to 2 different challenges for the commitment T, they must "know" all messages $\{m_i\}_{i=1}^n$ and the randomness r.

Remark 7. The special soundness property of our protocol relies on the same discrete logarithm assumption as the binding property of Pedersen commitments. Specifically, if an adversary could find two different openings $(\{m_i\}_{i=1}^n, r) \neq (\{m'_i\}_{i=1}^n, r')$ for the same commitment cm, they could use these to compute discrete logarithms between the generators $(g_1, \ldots, g_n, h)$. Our extractor demonstrates that producing two valid transcripts for different challenges is equivalent to breaking this binding property.

Theorem 8 (Honest-Verifier Zero-Knowledge). *Construction 1 is honest-verifier zero-knowledge.*

Proof. We construct a simulator $\mathcal{S}$ that, given only public input cm and a challenge c, produces transcripts that are identically distributed to those of real protocol executions. $\mathcal{S}$ works as follows:

1. On input (cm, c), $\mathcal{S}$ samples:
   - $s_i \leftarrow\$ \mathbb{Z}_p$ for $i \in [1, n]$
   - $u \leftarrow\$ \mathbb{Z}_p$
2. Computes $T \leftarrow (\prod_{i=1}^{n} g_i^{s_i})h^u \mathsf{cm}^{-c}$
3. Outputs transcript $(T, c, \{s_i\}_{i=1}^n, u)$

To show perfect HVZK, we prove the distribution of simulated transcripts is identical to real transcripts:

- In real protocol:
  - $s_i = \alpha_i + cm_i$ where $\alpha_i \leftarrow\$ \mathbb{Z}_p$
  - $u = \rho + cr$ where $\rho \leftarrow\$ \mathbb{Z}_p$
- Since α_i, ρ are uniform in $\mathbb{Z}_p$, the real s_i, u are uniformly distributed in $\mathbb{Z}_p$
- In simulation, s_i, u are sampled uniformly from $\mathbb{Z}_p$
- Therefore, (s_i, u) have identical distributions in both cases

– Given fixed (s_i, u), T is uniquely determined by the verification equation in both real and simulated transcripts

Thus, the simulated transcripts are perfectly indistinguishable from real protocol transcripts.

Corollary 9 (Proof of Zero Values). *Let $I \subseteq [1, n]$ be a set of indices. Given a commitment cm and commitment key ck, we construct modified public parameters ck' by removing the generators g_i for all $i \in I$ from ck. Using ck' with Construction 1 proves that $m_i = 0$ for all $i \in I$ while maintaining the properties of a Σ-protocol.*

Proof. Consider commitment $\mathsf{cm} = (\prod_{i=1}^{n} g_i^{m_i})h^r$. When proving with modified generators, the prover must demonstrate knowledge of opening $(\{m_i\}_{i \notin I}, r)$ such that:

$$\mathsf{cm} = (\prod_{i \notin I} g_i^{m_i})h^r$$

If this equation holds and cm was constructed using all generators, then necessarily $m_i = 0$ for all $i \in I$, as otherwise the verification would fail.

We now show the protocol maintains all properties of a Σ-protocol:

Completeness: For any valid commitment cm where $m_i = 0$ for all $i \in I$, the protocol succeeds with probability 1. The prover samples $\alpha_i \leftarrow\!\!\$\ \mathbb{Z}_p$ for $i \notin I$ and $\rho \leftarrow\!\!\$\ \mathbb{Z}_p$, computes $T = (\prod_{i \notin I} g_i^{\alpha_i})h^\rho$, and responds to challenge c with $s_i = \alpha_i + cm_i$ for $i \notin I$ and $u = \rho + cr$. Verification holds because:

$$(\prod_{i \notin I} g_i^{s_i})h^u = (\prod_{i \notin I} g_i^{\alpha_i + cm_i})h^{\rho + cr} = \mathsf{cm}^c \cdot T$$

Special Soundness: Given two accepting transcripts $(T, c, \{s_i\}_{i \notin I}, u)$ and $(T, c', \{s_i'\}_{i \notin I}, u')$ with $c \neq c'$, the extractor from Theorem 1 works on the reduced generator set to extract valid openings $m_i = \frac{s_i - s_i'}{c - c'}$ for $i \notin I$ and $r = \frac{u - u'}{c - c'}$. Since these values must satisfy the verification equation with the full generator set, all omitted indices must correspond to zero values.

HVZK: The simulator from Theorem 1 operates identically on the reduced generator set, producing perfectly indistinguishable transcripts.

Corollary 10 (Equality of Committed Values). *Construction 1 can be used to prove equality of messages in two Pedersen commitments. Specifically, for commitments $\mathsf{cm}_1 = g_1^m h_1^{r_1}$ and $\mathsf{cm}_2 = g_2^m h_2^{r_2}$, we can prove knowledge of (m, r_1, r_2) satisfying the relation:*

$$\mathcal{R} = \{((\mathsf{cm}_1, \mathsf{cm}_2, g_1, g_2, h_1, h_2), (m, r_1, r_2)) \mid \mathsf{cm}_1 = g_1^m h_1^{r_1} \wedge \mathsf{cm}_2 = g_2^m h_2^{r_2}\}$$

Proof. We instantiate Construction 1 with generators (g_1, g_2, h_1, h_2) and prove knowledge of opening:

$$\mathsf{cm} = (g_1^m h_1^{r_1}, g_2^m h_2^{r_2})$$

The protocol runs as follows:

1. $\mathcal{P}$ samples $\alpha, \rho_1, \rho_2 \leftarrow\!\!\$\ \mathbb{Z}_p$ and sends $T_1 \leftarrow g_1^\alpha h_1^{\rho_1}, T_2 \leftarrow g_2^\alpha h_2^{\rho_2}$
2. $\mathcal{V}$ sends challenge $c \leftarrow\!\!\$\ \mathbb{Z}_p$
3. $\mathcal{P}$ sends $s \leftarrow \alpha + cm, u_1 \leftarrow \rho_1 + cr_1, u_2 \leftarrow \rho_2 + cr_2$
4. $\mathcal{V}$ checks $g_1^s h_1^{u_1} = \mathsf{cm}_1^c T_1 \wedge g_2^s h_2^{u_2} = \mathsf{cm}_2^c T_2$

Security follows from Theorem 1:

- Completeness: Inherits directly as this is a specific instantiation
- Special Soundness: The extractor obtains a single m from s and corresponding r_1, r_2 from u_1, u_2
- HVZK: The simulator from Theorem 1 extends naturally by sampling a single s for both equations

Crucially, using the same α for both T_1 and T_2 ensures the extracted message m must be identical in both commitments.

Corollary 11 (Proof of Multiplicative Inverse). *Construction 1 can be used to prove that two committed values are multiplicative inverses. Specifically, for commitments* $\mathsf{cm}_1 = g^{m_1}h^{r_1}$ *and* $\mathsf{cm}_2 = g^{m_2}h^{r_2}$*, we can prove knowledge of* (m_1, m_2, r_1, r_2) *satisfying the relation:*

$$\mathcal{R} = \{((\mathsf{cm}_1, \mathsf{cm}_2, g, h), (m_1, m_2, r_1, r_2)) \mid \mathsf{cm}_1 = g^{m_1}h^{r_1} \ \wedge\ \mathsf{cm}_2 = g^{m_2}h^{r_2} \ \wedge\ m_1 \cdot m_2 = 1\}$$

Proof. The protocol runs as follows:

1. $\mathcal{P}$ samples $\alpha_1, \alpha_2, \rho_1, \rho_2, \rho_3, \rho_4 \leftarrow\!\!\$\ \mathbb{Z}_q$ and computes:
   - $T_1 \leftarrow g^{\alpha_1}h^{\rho_1}$, $T_2 \leftarrow g^{\alpha_2}h^{\rho_2}$
   - $\mathsf{cm}_3 \leftarrow \mathsf{cm}_1^{m_2}h^{\rho_3}$, $\mathsf{cm}_4 \leftarrow h^{\rho_4}$
   - $T_3 \leftarrow \mathsf{cm}_1^{\alpha_2}h^{\rho_3}$, $T_4 \leftarrow h^{\rho_4}$
2. $\mathcal{V}$ sends challenge $c \leftarrow\!\!\$\ \mathbb{Z}_q$
3. $\mathcal{P}$ responds with:
   - $s_1 \leftarrow \alpha_1 + cm_1$, $s_2 \leftarrow \alpha_2 + cm_2$
   - $u_1 \leftarrow \rho_1 + cr_1$, $u_2 \leftarrow \rho_2 + cr_2$
   - $u_3 \leftarrow \rho_3 + cr_3$, $u_4 \leftarrow \rho_4 + cr_4$
4. $\mathcal{V}$ checks:
   - $g^{s_1}h^{u_1} = \mathsf{cm}_1^c T_1 \quad \wedge \quad g^{s_2}h^{u_2} = \mathsf{cm}_2^c T_2 \quad \wedge \quad \mathsf{cm}_1^{s_2}h^{u_3} = \mathsf{cm}_3^c T_3 \quad \wedge \quad h^{u_4} = \mathsf{cm}_4^c T_4 \quad \wedge$ $\mathsf{cm}_3/\mathsf{cm}_4 = g$

Security follows from Theorem 1 as we are effectively running multiple instances of the base protocol with additional algebraic constraints. The final check $\mathsf{cm}_3/\mathsf{cm}_4 = g$ ensures $m_1 \cdot m_2 = 1$ because:

- $\mathsf{cm}_3 = \mathsf{cm}_1^{m_2}h^{\rho_3} = g^{m_1 m_2}h^{r_1 m_2 + \rho_3}$
- $\mathsf{cm}_4 = h^{\rho_4}$
- Therefore $\mathsf{cm}_3/\mathsf{cm}_4 = g^{m_1 m_2}h^{r_1 m_2 + \rho_3 - \rho_4}$
- If this equals g, then $m_1 \cdot m_2 = 1$

Special Soundness: Given two accepting transcripts $(T_1, T_2, T_3, T_4, c, s_1, s_2, u_1, u_2, u_3, u_4)$ and $(T_1, T_2, T_3, T_4, c', s_1', s_2', u_1', u$ for $c \neq c'$, we can extract a valid witness $(m_1, m_2, r_1, r_2, r_3, r_4)$ as follows:

For $i \in \{1, 2\}$:

$$m_i = \frac{s_i - s_i'}{c - c'}$$
$$r_i = \frac{u_i - u_i'}{c - c'}$$

For $i \in \{3, 4\}$:

$$r_i = \frac{u_i - u_i'}{c - c'}$$

The extracted values satisfy the relation $\mathcal{R}$ because:

– From verification equations 1 and 2, we obtain valid openings of cm_1 and cm_2

– From equation 3, we get that $\mathsf{cm}_3 = \mathsf{cm}_1^{m_2} h^{r_3}$

– From equation 4, we get that $\mathsf{cm}_4 = h^{r_4}$

– The final check $\mathsf{cm}_3/\mathsf{cm}_4 = g$ ensures $m_1 \cdot m_2 = 1$

Honest-Verifier Zero-Knowledge: We construct a simulator $\mathcal{S}$ that produces transcripts indistinguishable from real protocol runs:

1. On input $(\mathsf{cm}_1, \mathsf{cm}_2, c)$, $\mathcal{S}$ samples:
   – $s_1, s_2 \leftarrow\$ \mathbb{Z}_q$
   – $u_1, u_2, u_3, u_4 \leftarrow\$ \mathbb{Z}_q$
2. Computes:
   – $T_1 \leftarrow g^{s_1} h^{u_1} \mathsf{cm}_1^{-c}$
   – $T_2 \leftarrow g^{s_2} h^{u_2} \mathsf{cm}_2^{-c}$
   – $T_3 \leftarrow \mathsf{cm}_1^{s_2} h^{u_3} \mathsf{cm}_3^{-c}$
   – $T_4 \leftarrow h^{u_4} \mathsf{cm}_4^{-c}$

   where $\mathsf{cm}_3, \mathsf{cm}_4$ are computed such that $\mathsf{cm}_3/\mathsf{cm}_4 = g$
3. Outputs $(T_1, T_2, T_3, T_4, c, s_1, s_2, u_1, u_2, u_3, u_4)$

The simulated transcript is perfectly indistinguishable from real transcripts because:

– $s_1, s_2, u_1, u_2, u_3, u_4$ are uniformly random in both real and simulated transcripts

– Given these values, T_1, T_2, T_3, T_4 are uniquely determined by the verification equations

– The relation $\mathsf{cm}_3/\mathsf{cm}_4 = g$ is maintained in both real and simulated transcripts

Corollary 12 (Selective Disclosure). *Given a commitment* $\mathsf{cm} = (\prod_{i=1}^{n} g_i^{m_i})h^r$ *and* $I \subseteq [1, k]$ *be a set of indices of disclosed messages where* $k < n$*, we construct* $\mathsf{cm}' = (\prod_{i \notin I}^{n-k} g_i^{m_i})h^r$ *and* $D = [m_j]_{j \in I}$*. Construction 1 can be used to prove that* cm *commits to all messages,* D *contains disclosed messages and* cm' *commits to* $m_i \notin D$ *satisfying the relation:*

$$\mathcal{R} = \left\{ (\mathsf{cm}, \mathsf{cm}', \mathsf{ck}, D)([m_i]_{i \notin D}, r) \mid \mathsf{cm}' = (\prod_{i=1, i \notin I}^{n} g_i^{m_i})h^r \;\wedge\; \mathsf{cm}_d = \prod_{i=1, i \in I}^{n} g_i^{m_i} \wedge \mathsf{cm}' \cdot \mathsf{cm}_d = \mathsf{cm} \right\}$$

Corollary 13 (Selective Disclosure). *Let* $\mathsf{cm} = (\prod_{i=1}^{n} g_i^{m_i})h^r$ *be a Pedersen commitment to messages* $(m_1, \ldots, m_n)$ *and let* $I \subseteq [1, n]$ *be the set of indices to be disclosed. The protocol proves knowledge of openings for undisclosed messages while revealing values for disclosed messages satisfying the relation:*

$$\mathcal{R} = \left\{((\mathsf{cm}, g_{i}{}_{i=1}^{n}, h, I, m_i i \in I), (m_{i i \notin I}, r)) \middle| \mathsf{cm} = (\textstyle\prod_{i=1}^{n} g_i^{m_i})h^r \wedge \text{ } \textit{disclosed values match } m_{i i \in I} \right\}$$

Proof. The protocol runs as follows:

1. $\mathcal{P}$ computes
   - cm_d, a commitment to disclosed attributes $\mathsf{cm}_d = \prod_{i \in I} g_i^{m_i}$
   - $\mathsf{cm}' = \mathsf{cm} \cdot \mathsf{cm}_d^{-1}$ the remaining commitment with private attributes
   - $\mathsf{cm}' = \mathsf{cm}/\mathsf{cm}_d$ the remaining commitment with private attributes
2. $\mathcal{P}$ executes Construction 1 with cm' and sends $\mathcal{V}$ cm and disclosed attributes
3. $\mathcal{V}$ computes $\mathsf{cm}'_d \leftarrow \prod_{i \in I} g_i^{m_i}$ from disclosed attributes and checks
   - $\mathcal{P}$'s proofs from cm'
   - Checks if $\mathsf{cm}'_d \cdot \mathsf{cm}' = \mathsf{cm}$

Security: Special Soundness and Honest Verifier Zero Knowledge follows from Theorem 1 as we are running the same protocol but with subset of attributes. Completeness holds

- cm_d is computed from the disclosed attributes
- cm' is computed from the private attributes
- The homomorphic property of the commitments ensures commitment completeness.

7 Private Identity System from MIMC-ABC

Our identity system establishes a secure framework for issuing and managing privacy-preserving credentials across multiple authorities while maintaining accountability. The system involves four key entities: users, credential oracles (which verify and attest to user attributes), auditor (who manage revocation), and credential verifiers.

At the core of our system is a master credential issued by a government credential oracle, which serves as a root of trust. This credential contains two crucial committed elements: a secret identifier s that enables secure credential linking, and a committed VRF key k that generates context-specific nullifiers. These nullifiers serve dual purposes: preventing Sybil attacks at credential oracles and enabling efficient revocation. During master credential issuance, the VRF key is verifiably encrypted, the ciphertext is stored in the government system which associates a plaintext user profile to their ciphertext for revocation.

Users can obtain context credentials from various credential oracles by proving possession of a valid, unrevoked master credential and deriving a unique nullifier using their VRF key and the credential context. This design allows credential oracles to restrict issuance to users with trusted government-issued credentials, ensuring their credentials are only issued to verified identities. The system supports expressive verification statements that can combine attributes across multiple credentials. Since master credentials are government-issued and require stringent security checks, verifiers can leverage this trust by incorporating master credential validity, expiration, and revocation checks into their verification statements, inheriting the strong security properties of government-issued credentials. This enables credential oracles to maintain trust by ensuring their credentials become unusable if the underlying government credential is revoked.

The system supports flexible revocation through two mechanisms: targeted revocation of specific credentials via their nullifiers, and complete revocation of all user credentials by recovering their VRF key through the auditors. Government systems can initiate revocation by using plaintext identifiers, with auditors managing the conversion to the appropriate nullifiers. This approach maintains privacy while enabling practical accountability and administration.

We use the Multi Issuer Multi Credential Anonymous Credential system to implement a privacy-preserving digital identity system

7.1 Entities

Our identity system involves users, credential issuers, auditors, and credential verifiers.

User ($\mathcal{U}$) holds a master credential $Cred_{master}$ and any number of context credentials $Cred_{ctx}$ in their identity wallet. The master credential contains a unique identifier s, a VRF key k, and additional attributes, and is issued by a government entity. Context credentials are issued by participating organizations like universities or licensing authorities.

Credential Oracle ($\mathcal{M}, \mathcal{C}$) verifies user identity and issues digital credentials. The Master Credential Oracle $\mathcal{M}$ operates with keypair (SK_m, PK_m) for issuing "root" credentials, while Context Credential Oracles $\mathcal{C}$ use (SK_c, PK_c) for issuing domain-specific credentials.

Auditor ($\mathcal{A}$) consists of a threshold of nodes holding encryption and accumulator keypairs; for simplicity, we refer to both as (sk_A, pk_A). Users encrypt their VRF keys under the auditors' public key, as in key escrow schemes. Auditors can decrypt this key during revocation. The Auditor updates the revocation list.

Verifier ($\mathcal{V}$) represents any party wishing to verify a user's credentials.

7.2 Data Objects

We now describe the data objects that form our privacy-preserving decentralized identity system. At its core, a Master Credential serves as a root of trust, from which Context Credentials can be derived.

During Context Credential issuance, users generate a deterministic nullifier unique to each context using their Master Credential's secrets and the context string, enabling privacy-preserving credential linking.

Master Credential $\boldsymbol{Cred_m}$: A master credential is a high-security root credential issued by a government entity containing:

- Identity string s: a unique identifier
- VRF key k: used to generate context-specific nullifiers
- Context type ctx: always set to "master" for master credentials
- Additional attributes $attrs$: including expiry date, date of birth, etc.
- Credential Structure:
  - Master Commitment $C_m = Com([s, k, ctx, attrs], r)$: A Pedersen commitment to the credential attributes using randomness r
  - Oracle signature σ_m: A rerandomizable signature over C_m, verifiable under PK_m

Master Credential Oracle Data Record: Following successful master credential issuance, the oracle maintains a record containing:

- Commitment-Signature Pair $(C_m, Cred_m)$:
  - Master commitment $C_m = Com([s, k, ctx, attrs], r)$: the Pedersen commitment over credential attributes
  - Oracle Signature $Cred_m$ The signature over commitment C_m
- Key Encryption and Proof:
  - Encrypted VRF Key CT_k: the encryption of the user's VRF key, encrypted with the Auditor's public key $Enc_{PK_a}(k)$
  - Consistency proof Π_{CT}: The zero-knowledge proof that CT_k encrypts the committed key k

Context Credential $\boldsymbol{Cred_c}$: A user interacts with the Context Credential oracle to obtain a context-specific credential, which contains:

- Identity string s: The user's unique identifier from their master credential
- Nullifier τ: A deterministic value generated from (s, ctx)
- Context string ctx: A hashed identifier of the credential type (e.g., dmv, $universityofsydney$)
- Attribute list $attrs$: Additional credential-specific information such as expiry date
- σ_c the rerandomizable signature over C_c from the context credential oracle that proves the user has been issued $Cred_c$ over C_c
- Credential Structure:
  - Context commitment C_c: A Pedersen commitment $Com([s, \tau, ctx, attrs], r')$ to the credential attributes using randomness r'
  - Oracle signature σ_c: A rerandomizable signature over C_c, verifiable under PK_c

Context Credential Oracle Data Record: During credential issuance, the oracle maintains a record of the interaction containing:

- Master Credential Verification:
  - Randomized credential $Cred'_m$: a rerandomized version of the master credential
  - Randomized commitment C'_m: the corresponding rerandomized commitment
  - Opening proof $\Pi_{ComOpen}$: Zero-knowledge proof of correct commitment opening
  - Revocation proof $\Pi_{NonRevoked}$: Zero-knowledge proof that the credential has not been revoked
- Nullifier Components:
  - Context nullifier τ: The value $VRF(k, ctx)$ derived from the user's committed VRF key and credential context
  - Derivation proof Π_τ: Zero-knowledge proof establishing that
    - The VRF computation is correct
    - The key k matches the one committed in $Cred_m$
    - The context string ctx is correctly incorporated

Revocation List:

- Accumulator Structure:
  - Accumulator value A: The current state of the accumulator representing non-revoked credentials
  - Secret key sk_A: The accumulator manager's key for updates
  - Auxiliary information aux: Additional data needed for witness updates
- Revoked Elements:
  - Master revocations k: VRF keys of revoked master credentials
  - Context revocations τ: Nullifiers of revoked context credentials
  - Timestamp t: Time of revocation
  - Reason code rc: Justification for revocation
- Witness Management:
  - Non-membership witness w: Proof that a credential is not in the revocation set
  - Update information upd: Data for users to update their witnesses after accumulator changes

Notes on Threat/Trust model: threat model has issuer/verifier, trust model has credd oracle, auditor, etc. Keep consistent. Also, state what's out of scope e.g. network, physical, side-channel. domain in arke = context

Trust Model

– Credential Oracles: trusted to verify real-world identity before issuing credentials, they aren't trusted for privacy and may be compromised but can't issue credentials without the user participating in their protocol

– Auditors: are trusted to only decrypt user keys for legitimate revocation requests

– Network: communication assumed to be over encrypted channels, any storage is not trusted for credential contents

Threat Model We assume the auditor maintaining the revocation cannot be corrupted.

– Malicious Credential Oracle: A malicious credential oracle could "falsely issue attestations and impersonate any user it desires. Fortunately, recent work on authenticating web data has shown privacy-preserving, untrusted and correct credential oracles can be realized in practice [DECO, distefano, etc]. Additionally, we mitigate the threat level by confining each credential oracle to a unique domain." - from Arke.

– Malicious User: attempts to obtain multiple credentials for the same context, tries to forge credentials or share them with others, attempts to link credentials with other credentials not issued to the same master secret key

– Malicious Issuer: attempts to link multiple showing, collude with issuers to deanonymize users, stores presentation proofs to track users

– Malicious Verifier: issue credentials without proper verification, attempts to track credential usage, colludes with issuers or other verifiers

7.3 Syntax

Sam: Sam to update this to include master and context cred and revocation

Syntax of Anonymous Identity System with Sybil Resistance and Revocation

– $\mathsf{Setup}(1^\lambda) \to (\mathsf{pp}, \mathcal{UL}, \mathcal{RL})$ Takes security parameter λ in unary, outputs public parameters pp, empty user list $\mathcal{UL}$ and revocation list $\mathcal{RL}$.

– $\mathsf{OrgKeygen}(\mathsf{pp}, n) \xrightarrow{\$} (\mathsf{osk}, \mathsf{opk})$: is a probabilistic algorithm that takes public parameters pp and n the upper bound of credential attributes. Outputs organisations keypair $(\mathsf{osk}, \mathsf{opk})$

– $\mathsf{UserKeygen}(1^\lambda) \xrightarrow{\$} (\mathsf{usk})$ is a probabilistic algorithm. Outputs user secret key usk consisting of PRF key k and identity string s

– $(\mathsf{ObtainMaster}(usk, \mathbf{m}, aux), \mathsf{IssueMaster}(osk, cm, aux)) \to cred$: An interactive protocol. *Obtain* is a probabilistic alrogithm run by a user, inputs secret key, credential attribute vector $\mathbf{m}$ and auxiliary info. $Issue$ is a probabilistic algorithm run by an issuing organization that takes a commitment cm, issuers secret key osk, and auxiliary info. Outputs a credential $cred$ binding cm to the issuer signature.

– $(\mathsf{ObtainContext}(usk, \mathbf{m}, aux), \mathsf{IssueContext}(osk, cm, aux)) \to cred$: An interactive protocol. *Obtain* is a probabilistic alrogithm run by a user, inputs secret key, credential attribute vector $\mathbf{m}$ and auxiliary info. $Issue$ is a probabilistic algorithm run by an issuing organization that takes a commitment cm, issuers secret key osk, and auxiliary info. Outputs a credential $cred$ binding cm to the issuer signature.

– $(\mathsf{Show}(usk, cred, \phi), \mathsf{Verify}(cred', cm, \pi)) \to \{0, 1\}$ An interactive protocol. *Show* is a probabilistic algorithm run by a Prover. Takes secret key, credential, and predicate statement ϕ. $Verify$ is a deterministic algorithm run by a verifier, takes a randomized credential and commitment $cred', cm'$ and proof π. Otuputs 1 if verification succeeds, otherwise 0.

– $\mathsf{Revoke}(\mathcal{RL}, k') \rightarrow \mathcal{RL}'$ revoke is a deterministic algorithm, updates revocation list with revoked key k'

7.4 Security Model

– **Sybil resistance**: For any given context, no probabilistic polynomial time adversary can obtain more than 1 valid credential with non-negligible probability

– **Revocability**: For any given context, no probabilistic polynomial time adversary use a revoked credential

Sybil Resistance

Revocation When ra needs to revoke a user's credential/s (due to user request or credential provider request), ra finds $escrow$ based on the user's pid, recall ra has a user list $ul = (pid, escrow)$ and requests the auditor $audit$ to decrypt $s \leftarrow tpkdec_{ask}(escrow)$. $audit$ computes the nullifiers to add to the revocation accumulator. $nullif_{rcd} \leftarrow PRF_s(pid)$ and for each context credential to revoke, $nullif_{ctxid} \leftarrow PRF_s(ctxid)$. $audit$ updates the accumulator $acc' \leftarrow Acc.Add(acc)$ If the registration credential requires revocation, $audit$ can compute each $nullif \leftarrow PRF_s(ctxid); \forall; ctxid; \in; ctxl$ and add $(nullif, timestamp, reason)$ to rl. For record-keeping, ra stores Revocation Information $ri = (nullif, timestamp, reason)$ allowing ra to track which credentials are revoked and why, $nullif$ in rl ensures revoked credentials can't be verified. During credential verification, verifiers check if a credential's nullifier appears in rl, if present, the verification fails.

8 Performance Evaluation

What is the takeaway message from the evaluation?

– For non-private system, we enable privacy with little overhead. Our building block sigma protocol for private sybil resistance adds negligible overhead.

– For private system, but better efficiency, we have a SOTA paper TACT/S3ID (in the comparison table). Their paper does multi-attribute/multi-issuer credentials (they issue 1 credential per attribute), but their benchmarks don't show the complexity in verifying credentials together, or proving statements about their credential which they say would have non-negligible impact and theirs is lower bound. For us, by using simpler and well-known construction, we are more efficient (need to test this but I think so due to their construction) and better functionality.

9 Appendix, Old Writings

9.1 Zero Knowledge Proof

- $ZKP.ComOpen_{ck}(cm, m_i, r) \to \Pi^{cm}$
- $ZKP.ComOpenVfy_{ck}(cm, \Pi^{cm}) \to \{0, 1\}$

9.2 Anonymous Credential

$\underline{OrgKeyGen(1^\lambda, 1^n)}$: Given $\lambda, n > 0$, compute $BG = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \tilde{g})$.
Run $ck \leftarrow CM.Setup(BG, 1^\lambda, 1^n)$ which defines $(g, g^{\mathbf{y}}, \tilde{g}, \tilde{g}^{\mathbf{y}})$ with $\mathbf{y}_{\in[n]}$
Run $(sk, vk) \leftarrow PS.KeyGen(ck, 1^\lambda, 1^n)$ which defines $sk = X \in \mathbb{G}_1$ and $vk = \tilde{X} \in \mathbb{G}_2$
$osk = (g, X), opk = (ck, \tilde{X}, BG)$, return (osk, opk)

Obtain(opk, $\mathbf{m}$) Issue(osk, cm)

Parse $opk = (ck, \tilde{X}, BG)$, sample $r \leftarrow\$ \mathbb{Z}_p$
Compute $(cm, \widehat{cm}) \leftarrow CM.Com(ck, \mathbf{m}, r)$
Run $\Pi^{cm} \leftarrow ZKP.ComOpen(ck, cm, \mathbf{m}, r)$

$$\xrightarrow{\Pi^{cm}}$$

If $ZKP.ComOpenVfy(\Pi^{cm}, cm) = 0$, return $\perp$

$$\xleftarrow{\sigma=(\sigma_1,\sigma_2)}$$

$\sigma \leftarrow \mathsf{PS.Sign}(osk, cm)$

If $\mathsf{PS.Verify}(opk, cm, \sigma) = 0$, return $\perp$

$Cred \leftarrow (\sigma = (\sigma_1, \sigma_2), opk, cm, \widehat{cm})$

(Show, Verify): Using ..... Show, and Verify interact as follows

Show$(opk, cm, Cred)$ Verify$(osk, Cred')$

9.3 Old Intro

Privately Linked Context Credentials The Internet Identity Workshop discussed a problem space summarised by the following problems: 1. issuing credentials that are both government and privately issued 2. retaining accountability in derived credentials, ensuring derived credentials are fit for purpose and have revocation (Proveable Provenance, Linked Data) 3. combining traditional digital identity with decentralized identity

As digital wallets are gradually introduced, one notable problem involves combining As digital identities are introduced, there must be methods to combine the old world and the new world with respect to identities. One problem use-case is organisations such as a university issuing certificates as credentials. Universities want to start issuing credentials to users

A university, a credential provider, and wants to issue a credential to a user sam.

The University is not yet using decentralized identity but would like to issue a credential to Sam's digital identity wallet. Sam's logged in to the university portal with his classical login. Sam presses "Issue Credential" and starts the credential-issuing process.

The university wants to check a few things before issuing this credential. 1. make sure their national credential is valid, that is, it verifies 2. as it's an anonymous credential, the university wants to make sure the user that's logged into their portal is the same user with the registration credential. The user may selectively disclose their attributes, or prove equality of attributes in zero knowledge, or may have another proof.

The university generates a credentialId and stores it in their system and carries out the following protocol with Sam

Sam wants to keep as many details as secret as possible, and thus, he carries out the following protocol 1. Sam creates a new commitment Com([pid, 0|r), proves opening of the commitment and equality of pid between rcm and this commitment. 2. The university generates Com([0,credId|0) and homomorphically creates ccm([pid, credId|r) = Com([pid, 0|r) x Com([0,credId|0). The university then signs ccm([pid, credId|r) with their own signature scheme. 3. Sam can now take rcm and ccm, sigma(ccm) to the blockchain of nodes. 4. Sam runs a protocol with the blockchain to be issued a Decentralied version of this credential with full private accountability.

9.4 ZKP Sigma Protocol for proving PRF output with pairing

$$ZKP\,\{(m_1, m_2, r_1, r_2) : C_1 = g_1^{m_1} h_1^{r_1} \in \mathbb{G}_1 \wedge C_2 = g_2^{m_2} h_2^{r_2} \in \mathbb{G}_2 \wedge m_1 \cdot m_2 = 1\}$$

$$\begin{array}{lcl}
\mathsf{P}[m_1, m_2, r_1, r_2] & & \mathsf{V}[C_1, C_2, g, h] \\
\hline
/\!/\ C_1 = g_1^{m_1} h_1^{r_1}, C_2 = g_2^{m_2} h_2^{r_2} & & \\
\{\rho_i\}_{i=1}^2, \{\beta_i\}_{i=1}^2, \{\gamma_i\}_{i=1}^4 \leftarrow\!\!\$\ \mathbb{Z}_q^8 & & \\
T_1 \in \mathbb{G}_1 \leftarrow g_1^{\beta_1} h_1^{\rho_1} & & \\
T_2 \in \mathbb{G}_2 \leftarrow g_2^{\beta_2} h_2^{\rho_2} & & \\
/\!/\ \text{generate interim elements} & & \\
\alpha_1 \leftarrow m_1 \cdot m_2 & & \\
\alpha_2 \leftarrow m_1 \cdot t_2 & & \\
\alpha_3 \leftarrow t_1 \cdot m_2 & & \\
\alpha_4 \leftarrow t_1 \cdot t_2 & & \\
A_1 \leftarrow e(g_1, g_2) & & \\
A_2 \leftarrow e(g_1, h_2) & & \\
A_3 \leftarrow e(h_1, g_2) & & \\
A_4 \leftarrow e(h_1, h_2) & & \\
/\!/\ C_3 = e(C_1, C_2) & & \\
C_3 \in \mathbb{G}_T \leftarrow A_1^{\alpha_1} A_2^{\alpha_2} A_3^{\alpha_3} A_4^{\alpha_4} & & \\
T_3 \in \mathbb{G}_T \leftarrow A_1^{\gamma_1} A_2^{\gamma_2} A_3^{\gamma_3} A_4^{\gamma_4} & & \\
 & \xrightarrow{\{C_i, T_i\}_{i=1}^3, \{A_i\}_{i=1}^4} & \\
 & \xleftarrow{\quad e \quad} & e \leftarrow\!\!\$\ \mathbb{Z}_q \\
\{z_{mi} = \beta_i + e \cdot m_i\}_{i=1}^2 & & \\
\{z_{ri} = \rho_i + e \cdot r_i\}_{i=1}^2 & & \\
\{z_{ai} = \gamma_i + e \cdot \alpha_i\}_{i=1}^4 & & \\
 & \xrightarrow{\{z_{mi}, z_{ri}\}_{i=1}^2, \{z_{ai}\}_{i=1}^4} & \\
 & & C_1^e \cdot T_1 \stackrel{?}{=} g_1^{z_{m1}} h_1^{z_{r1}} \\
 & & C_2^e \cdot T_2 \stackrel{?}{=} g_2^{z_{m2}} h_2^{z_{r2}} \\
 & & C_3^e \cdot T_3 \stackrel{?}{=} A_1^{z_{a1}} A_2^{z_{a2}} A_3^{z_{a3}} A_4^{z_{a4}}
\end{array}$$

Completeness Proof of knowledge of exponents $m1, r1$ by opening C_1

$$\begin{aligned} C_1^e \cdot T_1 &\overset{?}{=} g^{z_{m1}} h^{z_{r1}} \\ (g^{m1} h^{r_1})^e \cdot g^{\beta_1} h^{\rho_1} &= g^{\beta_1 + e\cdot m_1} h^{\rho_1 + e\cdot r_1} \\ g^{e\cdot m1 + \beta_1} h^{e\cdot r_1 + \rho_1} &= g^{\beta_1 + e\cdot m_1} h^{\rho_1 + e\cdot r_1} \end{aligned}$$

Proof of knowledge of exponents $m2, r2$ by opening C_2

$$\begin{aligned} C_2^e \cdot T_2 &\overset{?}{=} \tilde{g}^{z_{m2}} \tilde{h}^{z_{r2}} \\ (\tilde{g}^{m2} \tilde{h}^{r_2})^e \cdot \tilde{g}^{\beta_2} \tilde{h}^{\rho_2} &= \tilde{g}^{\beta_2 + e\cdot m_2} \tilde{h}^{\rho_2 + e\cdot r_2} \\ \tilde{g}^{e\cdot m2 + \beta_2} \tilde{h}^{e\cdot r_2 + \rho_2} &= \tilde{g}^{\beta_2 + e\cdot m_2} \tilde{h}^{\rho_2 + e\cdot r_2} \end{aligned} \tag{1}$$

$$\begin{aligned} C_3 = e(C_1, \tilde{C_2}) &= e(g^{m1} h^{r1}, \tilde{g}^{m2} \tilde{h}^{r2}) \\ = e(g^{m1}, \tilde{g}^{m2}) \cdot e(g^{m1}, \tilde{h}^{r2}) &\cdot e(h^{r1}, g^{m2}) \cdot e(h_1^{r1}, \tilde{h}^{r2}) \\ = e(g, \tilde{g})^{m1\cdot m2} \cdot e(g, \tilde{h})^{m1\cdot r2} &\cdot e(h, \tilde{g})^{r1\cdot m2} \cdot e(h, \tilde{h})^{r1\cdot r2} \end{aligned} \tag{2}$$

- Prove knowledge of exponents $(m1 \cdot r2), (r1 \cdot m2), (r1 \cdot r2)$ for $C_3/e(g, \tilde{g})$ with respect to base points $e(g, \tilde{h}) e(h, \tilde{g}) e(h, \tilde{h})$
- $C_3/e(g, \tilde{g})$
-

Prove that $e(g, \tilde{g})^{(m1\cdot m2)} = e(g, \tilde{g})$ by computing $C_4 = e(g, \tilde{h})^{m1\cdot r2} \cdot e(h, \tilde{g})^{r1\cdot m2} \cdot e(h, \tilde{h})^{r1\cdot r2}$, proving the opening and equality of $(m1 \cdot r2), (r1 \cdot m2), (r1 \cdot r2)$ with C_3, then proving $C_3/C_4 = e(g, \tilde{g}) \cdot e(g, \tilde{h}) \cdot e(h, \tilde{g}) \cdot e(h, \tilde{h})$

Analysis Pairing Method

- **Prover:** 2 $\mathbb{G}_1$ exp, 2 $\mathbb{G}_2$ exp, 8 $\mathbb{G}_T$ exp, 1 $\mathbb{G}_1$ add, 1 $\mathbb{G}_2$ add, 6 $\mathbb{G}_T$ mul, 12 $\mathbb{F}_p$ mul, 8 $\mathbb{F}_p$ add, 4 pairing
- **Verifier:** 3 $\mathbb{G}_1$ exp, 3 $\mathbb{G}_2$ exp, 5 $\mathbb{G}_T$ exp, 2 $\mathbb{G}_1$ add, 2 $\mathbb{G}_2$ add, 4 $\mathbb{G}_T$ mul

Soundness

Sam: Sam todo

Zero Knowledge

Sam: Sam todo

Operation	**Prover**	**Verifier**
Commitment Equality Method		
G1 exponentiations	4	5
G1 additions	2	4
Fp multiplications	3	0
Fp additions	3	0
G1 VRF Method		
G1 exponentiations	11	11
G1 additions	4	7
Fp multiplications	6	0
Fp additions	6	0
Pairing + VRF Method		
G1 exponentiations	2	3
G2 exponentiations	2	3
G1 additions	1	2
G2 additions	1	2
Fp multiplications	12	0
Fp additions	8	0
GT exponentiations	8	5
GT multiplications	6	4
Pairings	4	4

Table 2: Comparison of computational operations between G1 and Pairing methods

Operation	**Time**
Full Pairing	1.6218 ms
Miller Loop	0.6931 ms
Final Exponentiation	0.9287 ms
G1 Mixed Addition (Affine + Jacobian)	672 ns
G1 Point Doubling (2P)	414 ns
G2 Mixed Addition (Affine + Jacobian)	2143 ns
G2 Point Doubling (2P)	1302 ns
Estimated G1 Scalar Mult (255-bit) *(255 doublings + 128 additions)*	191.59 μs
Estimated G2 Scalar Mult (255-bit) *(255 doublings + 128 additions)*	606.01 μs

Table 3: Performance metrics for arkworks BLS12-381 implementation. Scalar multiplication estimates assume naive double-and-add implementation without optimizations.

[0] The G1 and G2 scalar multiplication estimates are derived using a naive double-and-add implementation analysis for 255-bit scalars. For a random scalar k, we assume approximately 255 doubling operations

9.5 Research Questions

Main Research Question:

How can credential systems be designed to maintain accountability and sybil resistance with privacy and unlinkability.

Sub Research Questions

How can we design a privacy-preserving credential system that enables hidden linking of government and private credentials while maintaining provable provenance and revocability?

How can we efficiently combine credentials for verification while retaining their security properties

9.6 Methods

Notation Let $a : \mathbb{N} \mapsto \mathbb{N} \bigcup \{*\}$ and $b : \mathbb{N} \mapsto \mathbb{N}$ be any functions for which $a(\lambda)$ and $b(\lambda)$ are computable in $poly(\lambda)$ time (expect when a takes value $*$).

Definition 14. *A function family $F_{(\cdot)}(\cdot) : \{0,1\}^{a(\lambda)} \mapsto \{0,1\}^{b(\lambda)}$ is a family of VRFs if there exists a PPT algorithm Gen and deterministic algorithms Prove and Ver such that $Gen(1^\lambda)$ outputs a pair of keys (sk, pk), $Prove_{sk}(x)$ computes (y, π) where π is the proof of correctness, and $Ver_{pk}(x, y, \pi)$ verifies that $y = e(g, g)^{1/x+sk}$ using the proof $\pi = g^{1/x+sk}$*

Notation - another version A probabilistic polynomial time algorithm $\mathsf{Algorithm}(\mathsf{in}) \rightarrow \mathsf{out}$ receives an input in and returns an output out. $r \overset{\$}{\leftarrow} \mathbb{Z}_p$ r is sampled uniformly from the set of field elements modulo p, $h \leftarrow y$ is a deterministic assignment. $[n]$ denotes a sample space of $\{1, \ldots, n\}$. We assume type 3 bilinear pairings, $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_t$ over groups of prime order p, $g, \tilde{g}$ are uniformly chosen generators for $\mathbb{G}_1, \mathbb{G}_2$ such that $e(g, \tilde{g}) = g_t$. We use bold variables to denote vectors as $\mathbf{m} = [m_1, \ldots, m_\ell]$, $\mathbf{g} \in \mathbb{G}^\ell$, $\mathbf{x} \in \mathbb{Z}_p^\ell$, $\mathbf{g}^\mathbf{x} = \sum_{i=1}^{\ell} g_i^{x_i}$. We use multiplicative notation for $\mathbb{G}$ points i.e. $g^k = g \cdot g$ (k times)

Pseudo Random Function We use the properties of a pseudo-random function to derive the relationship between a master and context credential. The pseudorandomness property ensures the output y is computationally indistinguishable from random and thus hides the relationship between the user's secret key sk and an input x which could be the context of the new credential we want to privately link to.

$$PRF_{sk}(x) \rightarrow y$$

Definition 15 (Pseudo Random Function). *A PRF is a couple of algorithms $(PRF.Gen, f)$ with key space $\mathcal{K}$, input space $\mathcal{X}$ and output space $\mathcal{Y}$ such that*

- *$Gen(1^\lambda) \rightarrow (sk) : s \leftarrow\$ \; S$, sets $sk = s$*
- *$f : \mathcal{K} \times \mathcal{X} \rightarrow \mathcal{Y}$ is a keyed function family*

The main security property of a *PRF* is pseudorandomness, which informally states the output should appear random and not be able to be guessed. Formally, defined as

(one per bit) and 128 addition operations (corresponding to an expected Hamming weight of $\frac{255}{2}$ for a random scalar). The G1 estimate of 191.59μs is computed as (255 × 414ns) + (128 × 672ns) using the measured doubling and mixed addition timings. Similarly, the G2 estimate of 606.01μs is computed as (255 × 1302ns) + (128 × 2143ns). These estimates represent upper bounds as they do not account for common optimizations such as windowing methods, NAF (Non-Adjacent Form) representations, or parallel computation strategies.

Definition 16 (Pseudorandomness). *A couple of PPT algorithms $(PRF.Gen, f)$ is a Pseudo Random Function if, for any* PPT *adversary $\mathcal{A}$ there exists a negligible function ϵ such that*

$$\mathsf{Adv}(\mathcal{A}) := \left|\Pr\left[\mathsf{Exp}^{\mathsf{prf}}(\mathcal{A}) = 1\right] - \frac{1}{2}\right| \leq \epsilon(\lambda)$$

```
$\mathrm{Exp}^{\mathsf{prf}}(\mathcal{A})$
$b \leftarrow\!\!\$ \{0,1\}$
Sample $k \leftarrow\!\!\$ PRF.Gen(1^\lambda)$
Sample $f^* \leftarrow\!\!\$ \{g : g : \mathcal{X} \to \mathcal{Y}\}$
Run $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{prf}}}(1^\lambda)$
return $b == b'$

$\mathcal{O}_{\mathsf{prf}(x)}$
if $b = 1$
    return $f_k(x)$
else
    return $f^*(x)$
```

Fig. 1: The pseudorandomness game with adversary $\mathcal{A}$ and PRF $(PRF.Gen, f)$

Definition 17 (Computational Indistinguishability).
Let $X = \{X(a,n)\}_{a\in\{0,1\}^*;n\in\mathbb{N}}$ and $Y = \{Y(a,n)\}_{a\in\{0,1\}^*;n\in\mathbb{N}}$ be two probability ensembles, where:

- *Each ensemble is an infinite sequence of random variables*
- *$a \in \{0,1\}^*$ represents parties' inputs*
- *$n \in \mathbb{N}$ represents the security parameter*

X and Y are said to be computationally indistinguishable, denoted by $X \overset{c}{\equiv} Y$, if for every non-uniform polynomial-time algorithm D (called a distinguisher), there exists a negligible function $\mu(\cdot)$ such that for every $a \in \{0,1\}^*$ and every $n \in \mathbb{N}$:

$$|\Pr[D(X(a,n)) = 1] - \Pr[D(Y(a,n)) = 1]| \leq \mu(n)$$

Remark 18. This definition captures the idea that no efficient algorithm can tell the difference between samples from X and samples from Y with non-negligible probability. The term "non-uniform" allows the distinguisher D to have hard-coded advice that may depend on the input length, potentially making it more powerful.

Definition 19 (Bilinear map). *Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups of prime order p, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are multiplicative and $\mathbb{G}_T$ is multiplicative. Let g and h be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. We call $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ a bilinear map or pairing if it is efficiently computable and the following holds:*

Bilinearity: $e(g^a, \tilde{g^b}) = e(g, \tilde{g})^{ab} \quad \forall a, b \in \mathbb{Z}_p$.

Non-degeneracy: $e(g, \tilde{g}) \neq 1_{\mathbb{G}_T}$, *i.e.*, $e(g, \tilde{g})$ *generates* $\mathbb{G}_T$.

If $\mathbb{G}_1 = \mathbb{G}_2$, then e is symmetric (Type-1) and asymmetric (Type-2 or 3) otherwise. For Type-2 pairings, there is an efficiently computable isomorphism $\Psi : \mathbb{G}_2 \to \mathbb{G}_1$ but none from $\mathbb{G}_1 \to \mathbb{G}_2$; for Type-3 pairings, no efficiently computable isomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$ are known. Type-3 pairings are currently the optimal choice in terms of efficiency for a given security level.

Definition 20 (Commitment scheme). *A commitment scheme is a tuple* $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open})$ *of PPT algorithms where:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{ck}$ *takes security parameter* λ *(in unary) and generates the commitment key* ck*;*
- $\mathsf{Commit}_{\mathsf{ck}}(m) \to (\mathsf{cm}, r)$ *obtains commitment* cm *from secret message* m *and an opening key* r *which may be the randomness used in the computation.*
- $\mathsf{Open}_{\mathsf{ck}}(\mathsf{cm}; m, r) \to b \in \{0,1\}$ *verifies the opening of the commitment* cm *to the message* m *provided with the opening hint* r*, outputting a decision as to whether* cm *commits to* m*.*

Definition 21 (Secret Sharing). *A* (t,n) *secret sharing scheme* SS *is a tuple of* PPT *algorithms* $(\mathsf{Share}, \mathsf{Combine})$ *over message space* $x \in X$*:*

- $\mathsf{Share}^{t,n}(x, r) \xrightarrow{\$} ([x]_1, \ldots, [x]_n)$ *takes input* $x \in X$*, randomness* r *and outputs* n *shares* $([x]_1, \ldots, [x]_n)$
- $\mathsf{Combine}^{t,n}([x]_i, \ldots, [x]_t) \to x'$ *takes a threshold of secret shares* $[x]_i$ *for* $i > t$ *as input and combines to form* x' *the representation of the original message* $x' \in X$

Definition 22 (Threshold Public-Key Encryption). *A Threshold Public-Key Encryption Scheme* TPK *is a set of* PPT *algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Verify}, \mathsf{Combine})$ *over* $\mathcal{M}$*:*

- $\mathsf{TPK.Setup}(1^\lambda, n, t) \xrightarrow{\$} \{\mathsf{pk}, \mathsf{vk}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)\}$: *input the* t *of* n *threshold, output* pk *the public key,* vk *the verification key, and* sk_i *the shared secret key for each party.*
- $\mathsf{TPK.Enc}(\mathsf{pk}, m, \rho) \xrightarrow{\$} \beta$: *input message* m *and randomness* ρ*, output encryption* β
- $\mathsf{TPK.Dec}(\beta, \mathsf{sk}_i)) \to m_i$: *each party decrypts* β *with their shared secret key* sk_i
- $\mathsf{TPK.Verify}(\mathsf{pk}, \mathsf{vk}, m_i) \to \{0,1\}$: *input* pk, vk *and share of* m_i*, verify* m_i *was computed correctly from* pk, vk
- $\mathsf{TPK.Combine}(\mathsf{pk}, \mathsf{vk}, m_{i i \in \mathcal{S} \subseteq [n] s.t. |\mathcal{S}| \geq t+1}) \to m$: *recovers message* m *given* $t+1$ *partial decryptions which verify successfully*

Definition 23 (Homomorphism). *Let* G *and* H *be groups. A function* $\phi : G \to H$ *is called a homomorphism if it preserves the group operation. Specifically, for any elements* $a, b \in G$*, the following equation holds:*

$$\phi(a * b) = \phi(a) \circ \phi(b)$$

where $*$ *denotes the group operation in* G *and* $\circ$ *denotes the group operation in* H*.*

Remark 24. Note that ϕ is not required to be injective (one-to-one) or surjective (onto).

Definition 25. *For a homomorphism* $\phi : G \to H$*, the image of* ϕ *is defined as:*

$$\mathrm{Im}(\phi) = \phi(G) = \{\phi(g) : g \in G\} \subseteq H$$

Definition 26 (Image of homomorphism). *For a homomorphism* $\phi : G \to H$*, the image of* ϕ *is defined as:*

$$\mathrm{Im}(\phi) = \{h \in H \mid \exists g \in G \text{ such that } \phi(g) = h\}$$

Remark 27. The image of a homomorphism $\phi : G \to H$ can be thought of as the "landing spot" in H for elements coming from G. It's the subset of H that includes all possible outputs when ϕ is applied to any element in G. In essence, $\mathrm{Im}(\phi)$ tells us which elements of H are "reachable" through ϕ from some element in G.

Verifiable Random Function

Syntax

- $Gen(1^\lambda) \rightarrow (sk, pk)$: samples $s \leftarrow\!\$\ \mathbb{Z}_p^*$, sets $sk = s$ and $pk \leftarrow g^s$, returns (sk, pk)
- $Prove_{sk}(x) \rightarrow (y, \pi) : y \leftarrow e(g, g)^{1/(x+sk)}$, $\pi \leftarrow g^{1/(x+sk)}$ where y is the PRF output and π is the proof of correctness
- $Verify_{pk}(x, y, \pi) \rightarrow \{0, 1\}$ to verify y was computed correctly, verify $e(g^x \cdot pk, \pi) = e(g, g)$ and $y = e(g, \pi)$, output 1 for success, 0 for fail

Definition 28 (Verifiable Random Function).

A PRF is a triplet of PPT *algorithms* $(VRF.Gen, VRF.Eval, VRF.Vfy)$ *satisfying:*

1. *Correctness: For any* (pk, sk) *in the image of* $VRF.Gen(1^\lambda)$ *and any* $x \in \mathcal{X}$

$$\begin{aligned}(y, \pi) &\leftarrow VRF.Eval_{sk}(x)\\ 1 &\leftarrow VRF.Vfy_{pk}(x, y, \pi)\end{aligned}$$

2. *Unique Provability: For any pk (not necessarily in the range of* $VRF.Gen$*, any input* $x \in \mathcal{X}$*, pair of outputs* $y_0, y_1 \in \mathcal{Y}$ *and pair of proofs* π_0, π_1 *the following holds*

$$\begin{aligned}1 &\leftarrow VRF.Vfy_{pk}(x, y_0, \pi_0)\\ 1 &\leftarrow VRF.Vfy_{pk}(x, y_1, \pi_1)\\ y_0 &= y_1\end{aligned}$$

3. *Pseudorandomness: For any* PPT *adversary* $\mathcal{A}$ *running the VRF Experiment, there exists a negligible function* ϵ *such that*

$$\mathsf{Adv}(\mathcal{A}) := \left|\Pr\left[\mathsf{Exp}^{\mathsf{rnd}}(\mathcal{A}) \rightarrow 1\right] - \frac{1}{2}\right| \leq \epsilon(\lambda)$$

```
$\mathrm{Exp}^{\mathsf{eval}}(\mathcal{A})$
Sample $b \leftarrow\!\$\ \{0,1\}$
$pk, sk \leftarrow\!\$\ VRF.Gen(1^\lambda)$
set $x^* = \perp$
$x^* \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{eval}}}(pk)$
if $x^*$was previously queried:
  return 0
$y_0 \leftarrow\!\$\ \mathcal{Y}$
$(y_1, \pi) \leftarrow VRF.Eval(pk, x^*)$
$\mathcal{A}^{\mathcal{O}_{\mathsf{eval}}}(sk, y_b) \rightarrow b'$
return $b == b'$
```

```
$\mathcal{O}_{\mathsf{eval}(x)}$
if $x \neq x^*$
  $(y, \pi) \leftarrow VRF.Eval_{sk}(x)$
  return $(y, \pi)$
else : return $\perp$
```

Fig. 2: The pseudorandomness game with adversary $\mathcal{A}$ and PRF $(PRF.Gen, f)$

9.7 Revocation

Credential revocation is a fundamental challenge in identity management systems. While credentials grant users access to services, there must be mechanisms to invalidate them when necessary. Since the introduction of public key infrastructure, numerous solutions have been proposed to handle certificate revocation such as time-based expiration, usage limits (k-times use), and revocation lists. In the latter approach, a trusted authority manages a whitelist of valid credentials or blacklist of revoked ones, requiring users to prove their credential status with respect to the list.

The challenge becomes more complex in privacy-preserving systems as users must be able to prove revocation status without revealing the credential or its attributes. Furthermore, the revocation list should not leak information about which credentials are valid or revoked.

Revocation Scheme A revocation scheme enables efficient proofs of credential validity while maintaining privacy of the revocation status. The scheme consists of a revocation authority that manages the revocation state, protocols for revoking credentials, and methods for users to prove their credentials remain valid. A privacy-preserving revocation scheme must satisfy several properties:

– Privacy: Users can prove their credential's status without revealing the credential

– Unlinkability: Multiple proofs by the same user cannot be linked

– Efficiency: Proofs should be succinct and verification efficient

– Dynamic Updates: The system supports real-time credential revocation

Syntax A revocation scheme consists of the following algorithms:

– $\mathsf{REV.Setup}(1^\lambda) \xrightarrow{\$} (\mathsf{pp}, \mathsf{sk}, \mathsf{pk}, \mathsf{vt})$: Given security parameter 1^λ, generates system parameters pp, authority's secret key sk, public key pk, and initial revocation state vt

– $\mathsf{REV.Revoke}(\mathsf{sk}, \mathsf{vt}, \mathsf{cred}) \to (\mathsf{vt}', \mathrm{RI})$: Revokes credential cred, updates revocation state from vt to vt', and outputs revocation information RI

– $\mathsf{REV.TokenGen}(\mathsf{cred}, \mathsf{vt}, \mathrm{RI}) \to \mathsf{rt}$: Generates a revocation token rt for credential cred using the current revocation state vt and revocation information RI

– $\mathsf{REV.TokenVer}(\mathsf{vt}, \mathsf{cred}, \mathsf{rt}) \to \{0, 1\}$: Verifies revocation token rt for credential cred against revocation state vt

Accumulator An accumulator allows for compact representation of a set while enabling efficient proofs of membership. Our construction uses a universal accumulator that supports both membership and non-membership proofs. The accumulator maintains a constant-size value regardless of the number of elements in the set, while allowing elements to be dynamically added and removed. For each element, the system can generate succinct witnesses that prove either membership or non-membership in the accumulated set.

Syntax An accumulator ACU is a set of PPT algorithms $\mathsf{ACU} = \mathsf{Setup}, \mathsf{Add}, \mathsf{Del}, \mathsf{VerMem}, \mathsf{VerNonMem}$.

– $\mathsf{ACU.Setup}(1^\lambda) \xrightarrow{\$} \mathsf{pp}, \mathsf{sk}, \mathsf{pk}, \mathsf{vt}$: generates system parameters, takes security parameter 1^λ as input, outputs system parameters pp, secret key sk, public key pk, and initial accumulator value vt

– $\mathsf{ACU.Add}(\mathsf{sk}, \mathsf{vt}, \mathbf{x}) \to (\mathsf{vt}', \mathsf{wx})$: adds element $\mathbf{x}$, takes secret key sk, current accumulator value vt, element $\mathbf{x}$ as input. Outputs updated accumulator value vt', and membership witness wx

– $\mathsf{ACU.Del}(\mathsf{sk}, \mathsf{vt}, \mathbf{x}) \to (\mathsf{vt}', \hat{\mathsf{w}}\mathsf{x})$: Deletes element $\mathbf{x}$, takes secret key sk, current accumulator value vt, element $\mathbf{x}$ as input. Outputs updated accumulator value vt', non-membership witness $\hat{\mathsf{w}}\mathsf{x}$

– $\mathsf{ACU.VerMem}(\mathsf{vt}, \mathbf{x}, \mathsf{wx}) \to \{0, 1\}$: verifies membership, takes current accumulator value vt, element $\mathbf{x}$, witness wx as input. Outputs accept/reject

– $\mathsf{ACU.VerNonMem}(\mathsf{vt}, \mathbf{x}, \hat{\mathsf{w}}\mathsf{x}) \to \{0, 1\}$: Verifies non-membership, takes current accumulator value vt, element $\mathbf{x}$, non-member witness $\hat{\mathsf{w}}\mathsf{x}$ as input. Outputs accept/reject

with additional witness operations $\mathsf{MemWitUpOnAdd/Del}$, $\mathsf{NonMemWitUpOnAdd/Del}$

9.8 Discussion

The Internet Identity Workshop discussed a problem space summarised by the following problems:

1. issuing credentials that are both government and privately issued
2. retaining accountability in derived credentials, ensuring derived credentials are fit for purpose and have revocation (Proveable Provenance, Linked Data)
3. combining traditional digital identity with decentralized identity

A user has an Identity linked to multiple credentials, such as a driver's license and university card. Users want to authenticate with various Relying Parties (Verifiers) without being linked between multiple uses of the same service (e.g. a user verifying multiple credentials with 1 service such as a bank requiring proof of multiple credentials linked to an identity), and between uses of different services (e.g. a user presenting their drivers license for age verification on multiple services).

Current decentralized identity systems either don't provide this functionality or provide it at the expense of either accountability or privacy. CanDID stores a map between users multi-layered credentials providing a solution to the problem at the expense of the user's privacy. Other credential systems and pseudonym systems prove equality of hidden attributes in a credential such as name or id, which can more-easily be forged and does not support the hierarchical structure leveraging a highly secure and accountable government identity with not-so secure private credentials.

Credential Chaining

Pseudonym Systems There are 2 main models for Pseudonyms. One where the user has a Master Credential and derives pseudonym, or context credentials from the Master Credential. The applications differ; for example, in **Model 1**, a user may have their Passport as their Master Credential and wish to use it in a different scenario, such as voting for an election. The user will derive, by themselves, a new credential with the context "voting-2024," which will be verified in the same way as the master credential. **Model 2** differs in application scenarios. A context credential represents a credential from a different issuer, for example, a driver's license. During Context Credential issuance, a user will present their Master Credential which will be used to verify the identity of a user and to link the 2 credentials together. During context credential verification, the user may be requested to present just their Context Credential, or perhaps in a high-security verification setting, where a user may need to prove attributes in multiple credentials both Master and Context will need to be presented together. We optimize for this setting while *preserving privacy.*

Pseudonym Model 1: Master Credential, One Issuer, derived Pseudonyms

SyRA and TACT optimize for Non-Interactivity They also define context differently to us. Which isn't what CanDID defines context as and doesn't work for the same usescases and CanDID was defined for.

Previous Methods

SyRA: Sybil-Resilient Anonymous Signatures with Applications to Decentralized Identity [?] 1. enables users to "derive" unlinkable, sybil resistant pseudonyms (signatures) for a context e.g. PRF(sk, ctx) without interacting with the issuer 2. does not store a mapping on the issuer 3. Security properties: Sybil resistance in the context. Anonymity: no information is leaked through their nyms or signatures. Proves UC security for unforgeability, Sybil resilience, privacy, and unlinkability 4. SyRA leaves support for Identity Attributes for future work.

Summary: First, a protocol with issuer creates a user with secret key s. s generates a context specific pseudonym, or tag T "that attaches itself to a signature"?

– Level 1: $Issue_{isk}(s)$: $VRF_{isk}(s) \rightarrow T, \pi_{prf}$ such that π_{prf} generates the users keys and is symmetric across $\mathbb{G}_1, \mathbb{G}_2$: This algorithm is computed by the issuer with their secret key isk on identity string s Outputs $T = e(g, \tilde{g})^{1/(s+isk)}$ and π_{prf} is $usk = g^{1/(s+isk)} \in \mathbb{G}_1$ and $\widehat{usk} = \tilde{g}^{1/(s+isk)} \in \mathbb{G}_2$. Their verify algorithm is the same as the Dodis Yampolskiy and also verifies the asymmetric keys with bilinear pairing.

– Level 2: $Sign_{usk,\widehat{usk}}(ctx, m, ivk)$: $VUF_{usk}(ctx) \rightarrow T = e(H(ctx), \widehat{usk})$: This is the "sybil resistant signature". Proof of correctness comes from sigma protocol. During the verify protocol, the

Conclusion: SyRA creates a signature scheme where a user can "sign" on ctx, m from their secret key based on a VRF of their identity and the issuer's key. This does not account for Attribute-based credentials.

Attribute-Based Threshold Issuance Anonymous Counting Tokens [?] What don't they do that we do? We avoid Groth Sahai proofs - page 26 they refer to proving the signature scheme with GS proofs. They use Naor Pinkas Reingold PRF and verify signature with GS proofs. They focus only on deriving credentials / pseudonyms from their

Pseudonym Model 2: Master Credential, Multiple Issuers, Different Pseudonyms The Pseudonym Model [?] presents as an interaction between a User, a Certificate Authority (CA), and a Pseudonym Organisation (O). The user's identity is registered to the CA with their keypair skU, pkU, receiving a Master Credential to act as a trust anchor for all pseudonyms. With their Master Credential, Users request *unlinkable* Pseudonyms for other organizations by first proving the knowledge of a Master Credential that verifies with the CA, and the pseudonym requested has the same keypair as the Master Credential. Organizations *blindly* issue Pseudonym credentials on the same keypair as the Master Credential.

– $MasterCredIssue(skU, pkU, identity, skCA) \rightarrow CredM$ is an interactive algorithm run by a user and a credential authority with keypair $skCA, pkCA$. The user is known to the CA and shares their identity and a keypair skU, pkU. The CA checks the skU, pkU relation and issues $CredM$, a signature $\sigma_{CA} \leftarrow Sign_{skCA}(pkU)$

– $NymGeneration(CredM, pkCA, Nym, skO) \rightarrow CredNym$ is an interactive algorithm run by a user and an organization the user wishes to create a pseudonym with. $Nym1$ is a commitment $Com(skU, pkU, r)$ with randomness r, r should be unique per pseudonym. U generates a zero-knowledge proof of knowledge of a new pseudonym $Nym1$ with skU, pkU corresponding to $CredM$, $CredM$ verifies correctly, and pkU, skU are related.

$$\begin{aligned} ZKP\{(skU, pkU, r) : Nym = Com(skU, pkU, r) \wedge \\ Verify_{pkCA}(CredM) = 1 \wedge \\ pkU = g^{skU}\} \end{aligned}$$

On successful ZKP verification, algorithm outputs $CredNym \leftarrow Sign_{skO}(Nym)$

– $NymVerify(CredNym, pkO) \rightarrow \{0, 1\}$ is an interactive algorithm run by a user and a verifier. Recall $CredNym$ is a signature over a commitment $Sign_{skO}(Nym)$. The user randomizes $CredNym' \leftarrow CredNym$ and $Nym' \leftarrow Com(skU, pkU, r)$, and in zero knowledge, proves $CredNym$ verifies correctly with respect to the original signature, and the organisation public key

$$\begin{aligned} ZKP\{(skU, pkU, r, r') : Nym' &= Com(skU, pkU, r') \wedge \\ \exists\ Nym \text{ such that } Verify_{pkO}(Nym) &= 1 \wedge \\ Nym &= Com(skU, pkU, r) \wedge \\ pkU &= g^{skU}\} \end{aligned}$$

9.9 NIZK for Sybil Resistant Issuance

We have $g^{\frac{1}{m}}$ and g^m. We want to prove their relationship is reciprocal. We will use this later when proving the output of the Dodis Yampolskiy VRF $g^{\frac{1}{sk+x}}$ Let $m_1 = m$, and $m_2 = \frac{1}{m}$. Therefore, we can prove that $m_1 \cdot m_2 = 1$

$$C_1 = g^{m_1} h^{r_1}$$
$$C_2 = g^{m_2} h^{r_2}$$
$$C_3 = C_1^{m_2} h^{r_3}$$
$$C_4 = h^{r_1 m_2 + r_3}$$

We use C_1, C_2 in a zero-knowledge proof protocol to prove the relation with public elements C_1, C_2, g, h

$$ZKP\left\{(m_1, m_2, r_1, r_2) : C_1 = g^{m_1} h^{r_1} \wedge C_2 = g^{m_2} h^{r_2} \wedge m_1 \cdot m_2 = 1\right\}$$

$\mathsf{P}[m_1, m_2, r_1, r_2]$ $\mathsf{V}[C_1, C_2, g, h]$

$/\!/\ C_1 = g^{m_1} h^{r_1}, C_2 = g^{m_2} h^{r_2}$

$\{\rho_i\}_{i=1}^4, \{\beta_i\}_{i=1}^2 \leftarrow\$ \mathbb{Z}_q^6$

$T_1 \leftarrow g^{\beta_1} h^{\rho_1}$

$T_2 \leftarrow g^{\beta_2} h^{\rho_2}$

$/\!/$ generate interim elements

$C_3 \leftarrow C_1^{m_2} h^{r_3}$

$T_3 \leftarrow C_1^{\beta_2} h^{\rho_3}$

$r_4 \leftarrow r_1 m_2 + r_3$

$C_4 \leftarrow h^{r4}$

$T_4 \leftarrow h^{\rho_4}$

$$\xrightarrow{\{C_i, T_i\}_{i=1}^4}$$

$$\xleftarrow{e} \qquad e \leftarrow\$ \mathbb{Z}_q$$

$z_{m1} = \beta_1 + e \cdot m_1$

$z_{r1} = \rho_1 + e \cdot r_1$

$z_{m2} = \beta_2 + e \cdot m_2$

$z_{r2} = \rho_2 + e \cdot r_2$

$z_{r3} = \rho_3 + e \cdot r_3$

$z_{r4} = \rho_4 + e \cdot r_4$

$$\xrightarrow{\{z_{mi}\}_{i=1}^2, \{z_{ri}\}_{i=1}^4}$$

$C_1^e \cdot T_1 \overset{?}{=} g^{z_{m1}} h^{z_{r1}}$

$C_2^e \cdot T_2 \overset{?}{=} g^{z_{m2}} h^{z_{r2}}$

$C_3^e \cdot T_3 \overset{?}{=} C_1^{z_{m2}} h^{z_{r3}}$

$C_4^e \cdot T_4 \overset{?}{=} h^{z_{r4}}$

$C_3 / C_4 = g$

Bellare Properties - Anonymity | uncorrupt opener | fully corrupt issuer - Traceability | partially corrupt opener | uncorrupt issuer - Non-frameability | fully corrupt opener | fully corrupt issuer

[?]

Correctness

1. the signature should be valid
2. the opening algo should correctly identify the Signer given the message and signature
3. the proof returned by opening algo should be accepted by the judge

Formalize correctness with an experiment involving an adversary.

involved: Adversary, signature scheme, adversary A, secparam k.

$$Adv^{corr}_{GS,\mathcal{A}}(k) = \Pr[\mathsf{Exp}^{corr}_{GS,\mathcal{A}}(k) = 1]$$

We say the dynamic group signature scheme GS is correct if $Adv^{corr}_{GS,\mathcal{A}}(k) = 0$ for any adversary $\mathcal{A}$ and any $k \in \mathbb{N}$, the adversary is not computationally restricted.

Experiment $\mathsf{Exp}^{corr}_{GS,\mathcal{A}}(k)$

This says 1. run GKg with secparam, get gpk, ik, ok 2. Corrupt users set is 0

Construction 1: Proof of Zero(C)

Public Parameters: $g_1, g_2, h_1 \in \mathbb{G}$

Inputs: C such that $C = g_1^m g_2 h^r$, $\mathcal{P}$ knows $m, r \in \mathbb{Z}_q$.

1. $\mathcal{P}$ samples $\alpha, \rho \leftarrow\$ [q-1]$ and sends $T \leftarrow g_1^\alpha g_2 h_1^\rho$
2. $\mathcal{V}$ sends challenge $c \leftarrow\$ [q-1]$
3. $\mathcal{P}$ sends $s \leftarrow \alpha + cm, u \leftarrow \rho + cr$
4. $\mathcal{V}$ verifies that $g_1^s g_2^c h_1^u = C^c T$

Theorem 29. *Construction* **??** *is a Σ-protocol for the relation:*

$$\mathcal{R} = \{(C, g_1, g_2, h, q), (m, r) \mid C = g_1^m g_2 h_1^r\}$$

Proof. Folklore

Theorem 30 (Perfect Completeness). *Construction* **??** *is a $\Sigma-$protocol for the relation $\mathcal{R}$ with perfect completeness:*

Proof. We prove completeness by showing that for any $(C, g, h, q), (m, r) \in \mathcal{R}$, when both $\mathcal{P}$ and $\mathcal{V}$ follow the protocol, $\mathcal{V}$ accepts with $\Pr = 1$.

Let $x = (C, g_1, g_2, h, q)$ be common input and $w = (m, r)$ be $\mathcal{P}$'s private input. Consider an execution of the protocol where:

1. $\mathcal{P}$ samples $\alpha, \rho \leftarrow\$ [q-1]$ and sends $T \leftarrow g_1^\alpha g_2 h^\rho$
2. $\mathcal{V}$ sends challenge $c \leftarrow\$ [q-1]$
3. $\mathcal{P}$ responds with $s \leftarrow \alpha + cm, u \leftarrow \rho + cr$

Verification holds by

$$\begin{aligned} g_1^s g_2^c h_1^u &\overset{?}{=} C^c T \\ g_1^{\alpha+cm} g_2^c h^{\rho+cr} &\overset{?}{=} (g_1^m g_2 h^r)^c g_1^\alpha g_2 h^\rho \\ g_1^{\alpha+cm} g_2^c h^{\rho+cr} &= g_1^{\alpha+cm} g_2^c h^{\rho+cr} \end{aligned} \tag{3}$$

Thus, an honest verifier always accepts an honest prover's proof.

Theorem 31 (Soundness). *Construction* **??** *is a $\Sigma-$protocol for the relation $\mathcal{R}$ with soundness:*

Proof. We prove completeness by showing that for any $(C, g, h, q), (m, r) \in \mathcal{R}$, when both $\mathcal{P}$ and $\mathcal{V}$ follow the protocol, $\mathcal{V}$ accepts with $\Pr = 1$.

Let $x = (C, g_1, g_2, h, q)$ be common input and $w = (m, r)$ be $\mathcal{P}$'s private input. Consider an execution of the protocol where:

1. $\mathcal{P}$ samples $\alpha, \rho \leftarrow\$ [q-1]$ and sends $T \leftarrow g_1^\alpha g_2 h^\rho$

2. $\mathcal{V}$ sends challenge $c \leftarrow\$ [q-1]$
3. $\mathcal{P}$ responds with $s \leftarrow \alpha + cm, u \leftarrow \rho + cr$

Verification holds by

$$\begin{aligned} g_1^s g_2^c h_1^u &\overset{?}{=} C^c T \\ g_1^{\alpha+cm} g_2^c h^{\rho+cr} &\overset{?}{=} (g_1^m g_2 h^r)^c g_1^\alpha g_2 h^\rho \\ g_1^{\alpha+cm} g_2^c h^{\rho+cr} &= g_1^{\alpha+cm} g_2^c h^{\rho+cr} \end{aligned} \tag{4}$$

Thus, an honest verifier always accepts an honest prover's proof.

Commitment Scheme

References

ASM06. M. H. Au, W. Susilo, and Y. Mu. Constant-Size Dynamic k-TAA. *Security and Cryptography for Networks*, 4116:111–125, 2006. Series Title: Lecture Notes in Computer Science.

BCD$^+$17. F. Baldimtsi, J. Camenisch, M. Dubovitskaya, A. Lysyanskaya, L. Reyzin, K. Samelin, and S. Yakoubov. Accumulators with Applications to Anonymity-Preserving Revocation, 2017. Publication info: Published elsewhere. Minor revision. IEEE European Symposium on Security and Privacy 2017.

BCK$^+$22. M. Bellare, E. Crites, C. Komlo, M. Maller, S. Tessaro, and C. Zhu. Better than Advertised Security for Non-interactive Threshold Signatures. In *Advances in Cryptology – CRYPTO 2022*, pages 517–550, Cham, 2022. Springer Nature Switzerland.

BS23. M. Babel and J. Sedlmeir. Bringing data minimization to digital wallets at scale with general-purpose zero-knowledge proofs, January 2023. arXiv:2301.00823 [cs].

BSZ05. M. Bellare, H. Shi, and C. Zhang. Foundations of Group Signatures: The Case of Dynamic Groups. In *Topics in Cryptology – CT-RSA 2005*, pages 136–153, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. Series Title: Lecture Notes in Computer Science.

CDH16. J. Camenisch, M. Drijvers, and J. Hajny. Scalable Revocation Scheme for Anonymous Credentials Based on n-times Unlinkable Proofs. In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*, pages 123–133, Vienna Austria, October 2016. ACM.

CDR16. J. Camenisch, M. Dubovitskaya, and A. Rial. UC Commitments for Modular Protocol Design and Applications to Revocation and Attribute Tokens. In *Advances in Cryptology – CRYPTO 2016*, pages 208–239. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016. Series Title: Lecture Notes in Computer Science.

Cha85. D. Chaum. Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

CKS24. E. Crites, A. Kiayias, and A. Sarencheh. SyRA: Sybil-Resilient Anonymous Signatures with Applications to Decentralized Identity, 2024. Publication info: Preprint.

CL02. J. Camenisch and A. Lysyanskaya. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In *Advances in Cryptology — CRYPTO 2002*, pages 61–76. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002. Series Title: Lecture Notes in Computer Science.

CL04. J. Camenisch and A. Lysyanskaya. Signature Schemes and Anonymous Credentials from Bilinear Maps. *Advances in Cryptology – CRYPTO 2004*, 3152:56–72, 2004. Series Title: Lecture Notes in Computer Science.

Elt24. O. E. O. Eltayeb. The Crucial Significance of Governance, Risk and Compliance in Identity and Access Management. *Journal of Ecohumanism*, 3(4):2395–2405, August 2024.

FHS19. G. Fuchsbauer, C. Hanser, and D. Slamanig. Structure-Preserving Signatures on Equivalence Classes and Constant-Size Anonymous Credentials. *Journal of Cryptology*, 32(2):498–546, April 2019.

LRSW00. A. Lysyanskaya, R. L. Rivest, A. Sahai, and S. Wolf. Pseudonym Systems. In *Selected Areas in Cryptography*, pages 184–199. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000. Series Title: Lecture Notes in Computer Science.

MMZ$^+$21. D. Maram, H. Malvai, F. Zhang, N. Jean-Louis, A. Frolov, T. Kell, T. Lobban, C. Moy, A. Juels, and A. Miller. Candid: Can-do decentralized identity with legacy compatibility, sybil-resistance, and accountability. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1348–1366. IEEE, 2021.

noa21. Happy 10th Birthday – AWS Identity and Access Management | AWS News Blog, May 2021. Section: Launch.

noa24. Regulation (EU) 2024/1183 of the European Parliament and of the Council of 11 April 2024 amending Regulation (EU) No 910/2014 as regards establishing the European Digital Identity Framework, April 2024. Doc ID: 32024R1183 Doc Sector: 3 Doc Title: Regulation (EU) 2024/1183 of the European Parliament and of the Council of 11 April 2024 amending Regulation (EU) No 910/2014 as regards establishing the European Digital Identity Framework Doc Type: R Usr_lan: en.

PCB$^+$. R. Pang, R. Caceres, M. Burrows, Z. Chen, P. Dave, N. Germer, A. Golynski, K. Graney, N. Kang, L. Kissner, J. L. Korn, A. Parmar, C. D. Richards, and M. Wang. Zanzibar: Google's Consistent, Global Authorization System.

PS16. D. Pointcheval and O. Sanders. Short Randomizable Signatures. *Topics in Cryptology - CT-RSA 2016*, 9610:111–126, 2016. Series Title: Lecture Notes in Computer Science.

RARM. R. Rabaninejad, B. Abdolmaleki, S. Ramacher, and A. Michalas. Attribute-Based Threshold Issuance Anonymous Counting Tokens and Its Application to Sybil-Resistant Self-Sovereign Identity.

RPX$^+$22. D. Rathee, G. V. Policharla, T. Xie, R. Cottone, and D. Song. ZEBRA: SNARK-based Anonymous Credentials for Practical, Private and Accountable On-chain Access Control, 2022. Publication info: Preprint.

RWGM22. M. Rosenberg, J. White, C. Garman, and I. Miers. zk-creds: Flexible Anonymous Credentials from zkSNARKs and Existing Identity Infrastructure, 2022. Publication info: Published elsewhere. Major revision. 2023 IEEE Symposium on Security and Privacy (SP).

SNA21. R. Soltani, U. T. Nguyen, and A. An. A Survey of Self-Sovereign Identity Ecosystem. *Security and Communication Networks*, 2021:1–26, July 2021. arXiv:2111.02003 [cs].

WGW$^+$23. K. Wang, J. Gao, Q. Wang, J. Zhang, Y. Li, Z. Guan, and Z. Chen. Hades: Practical decentralized identity with full accountability and fine-grained sybil-resistance. In *Proceedings of the 39th Annual Computer Security Applications Conference*, pages 216–228, 2023.

ZYD$^+$22. X. Zhang, M. M. Yadollahi, S. Dadkhah, H. Isah, D. P. Le, and A. A. Ghorbani. Data breach: analysis, countermeasures and challenges. *International Journal of Information and Computer Security*, 19(3/4):402, 2022.