A Modular Zero-Knowledge Credential Framework for Multi-System Attribute Verification with Scoped Unlinkability and Efficient Accumulator-Based Revocation

Sayan Bairagi[1], Sayan Singha Roy[1], Abir Rakshit[1], Anik Bhowmick[1]

[1]Department of Computer Science and Engineering
Supreme Knowledge Foundation,
Affiliated with Maulana Abul Kalam Azad University of Technology, India
Email: bairagisayan464@gmail.com

***Abstract*—This work presents a zero-knowledge credential framework designed to enable secure and privacy-preserving attribute verification across multiple independent systems. The framework allows a user to prove statements of the form $a \geq t$, where $a \in \mathbb{Z}_q$ represents a secret attribute and t denotes a public threshold, without revealing the attribute value itself. At the same time, the framework prevents the exposure of any globally stable identifier, thereby eliminating the risk of cross-domain tracking.**

The construction is based on Pedersen commitments, where each attribute is encoded as $C = g^a h^r \in G$, with $G \subseteq \mathbb{Z}_p^*$ denoting a cyclic group of prime order q. The generators g and h are selected such that the discrete logarithm relation between them is unknown. This ensures that the commitment is computationally binding under the discrete logarithm assumption and perfectly hiding due to the use of randomness r. As a result, the committed attribute remains concealed while still allowing verification of statements about it.

Predicate verification is achieved using a sigma protocol, which enables the prover to demonstrate knowledge of valid witnesses without revealing them. In particular, the protocol proves the relation $C \cdot g^{-t} = g^{\delta} h^r$, where $\delta = a - t$. This transformation allows the system to verify threshold conditions such as $a \geq t$ without disclosing the value of a. The zero-knowledge property of the protocol ensures that the verifier learns only the validity of the statement and no additional information about the underlying attribute or randomness.

To prevent correlation of user activity across different verification domains, the framework introduces scoped pseudonyms defined as $ID_S = pk^{H(S)}$, where $pk = g^x$ is a public key derived from a secret key x, and H is a cryptographic hash function modeled as a random oracle. The scope S represents a domain-specific identifier. This construction produces a unique identifier for each domain while ensuring that identifiers generated for different scopes cannot be linked without solving the discrete logarithm problem in G.

Revocation is supported through an RSA accumulator constructed under the Strong RSA assumption. For a revoked set $R = \{r_i\}$, the accumulator value is defined as $A = g^{\prod r_i} \mod N$, where N is an RSA modulus. The system enables efficient non-membership verification using witnesses derived from Bézout coefficients[1]. This mechanism allows a verifier to confirm that a credential has not been revoked, while maintaining constant verification cost that does not depend on the size of the revoked set. Importantly, this process does not introduce any additional identifiers that could compromise user privacy.

The framework follows a complete lifecycle consisting of system setup, credential issuance, proof generation, scoped identifier derivation, verification, and revocation checking. During issuance, attributes are committed and signed by an issuer, producing a credential that can later be used by the holder. During authentication, the holder generates a non-interactive zero-knowledge proof bound to a verifier-specific challenge, ensuring freshness and resistance to replay attacks. The verifier evaluates the proof, validates the credential, and performs revocation checks without accessing any underlying attribute values.

The security of the system is grounded in well-established cryptographic assumptions, including the discrete logarithm assumption in prime-order groups, the Strong RSA assumption for accumulator security, and the random oracle model for hash functions. Under these assumptions, the framework provides attribute privacy, soundness of predicate proofs, scoped unlinkability across verification domains, and resistance to replay and collusion-based inference attacks.

The complete protocol stack has been implemented using 2048-bit security parameters within a modular architecture that includes issuer, holder, and multiple verifier components. The system is designed to be compatible with decentralized identity frameworks through the integration of decentralized identifiers (DID) and verifiable credentials (VC). Experimental evaluation demonstrates that the framework achieves an average verification latency below 3 milliseconds, a compact presentation size of approximately 3 kilobytes, and stable revocation verification performance for revoked sets containing up to 200 elements.

The proposed framework demonstrates that selective disclosure, strong unlinkability, and efficient revocation can be achieved

[1]Bézout coefficients enable membership/non-membership proofs in RSA accumulators.

simultaneously without relying on pairing-based cryptography or trusted setup assumptions. The modular structure allows independent evolution of system components while maintaining consistent security guarantees. This makes the system suitable for practical deployment in multi-system identity verification scenarios where both security and privacy are essential.

***Index Terms*—Zero-knowledge proofs, Attribute-based credentials, Selective disclosure, Unlinkability, RSA accumulator, Pedersen commitments, Sigma protocols, Decentralized identity, Verifiable credentials**

I. Introduction

Digital identity infrastructures are increasingly deployed across heterogeneous and mutually independent systems. Each system operates under distinct policy requirements and demands verification of user attributes in a secure and reliable manner. In many practical scenarios, a user is required to demonstrate properties of an attribute, such as satisfying an age threshold, meeting residency conditions, or possessing a valid membership status. However, these requirements do not necessitate disclosure of the complete attribute value. Despite this, conventional authentication mechanisms typically expose full attribute information or rely on globally stable identifiers. Such approaches introduce significant privacy risks, including cross-domain correlation, long-term tracking, and uncontrolled data exposure.

The fundamental limitation of existing systems lies in their inability to separate *verification* from *disclosure*. Once identity data is shared, it can be stored, reused, or combined across systems without user control. This leads to the accumulation of sensitive information and enables profiling across independent domains. These challenges highlight the need for a cryptographic approach that ensures minimal disclosure while preserving verifiability.

Zero-knowledge proof (ZKP) [18] systems provide a principled solution to this problem. A zero-knowledge proof allows a prover to convince a verifier that a statement is true without revealing any additional information beyond the validity of the statement itself. This property enables verification of conditions over secret data while maintaining strict privacy guarantees. Modern constructions, including Bulletproofs [8] and SNARK-based systems, have demonstrated that such proofs can be made efficient and practical. In the context of identity systems, schemes such as Idemix [4], U-Prove [5], and BBS+ signatures [7] support selective disclosure of attributes.

Although these systems provide important functionality, several challenges remain when extending them to multi-system environments. First, identifiers used during verification must remain unlinkable across independent domains. Second, revocation must be enforceable without introducing mechanisms that compromise anonymity. Third, verification must remain computationally efficient under standard security parameters to support real-world deployment. Many existing approaches address only a subset of these requirements, or rely on pairing-based cryptography and trusted setup assumptions that increase complexity and limit deployability. In particular, pairing-based credential systems introduce higher computational overhead in multi-verifier settings, while some token-based constructions rely on deterministic identifiers or constrained randomness, which can enable correlation across sessions. Similarly, conventional revocation mechanisms often depend on certificate lists or persistent identifiers, which either increase verification cost or weaken privacy guarantees.

The problem addressed in this work is therefore defined as follows:

> *A user must be able to prove a predicate over a committed attribute to multiple independent verifiers such that the attribute remains hidden, verification sessions remain unlinkable across domains, and revocation remains enforceable without introducing globally stable identifiers.*

To address this problem, a modular credential framework is constructed using well-established cryptographic primitives. Attributes are encoded using Pedersen commitments of the form $C = g^a h^r$, which provide computational binding under the discrete logarithm assumption and perfect hiding due to randomness. Predicate verification of the form $a \geq t$ is achieved through sigma protocols that prove knowledge of valid witnesses without revealing the attribute value. The transformation $\delta = a - t$ enables the verification of threshold conditions while preserving zero-knowledge properties.

Unlinkability across verification domains is achieved through scoped pseudonyms defined as $ID_S = pk^{H(S)}$, where S represents a domain-specific scope and H is a cryptographic hash function modeled as a random oracle. This construction ensures that identifiers derived for different scopes are computationally independent, thereby preventing cross-domain correlation. Revocation is enforced using RSA accumulators constructed under the Strong RSA assumption. Non-membership witnesses allow efficient verification that a credential has not been revoked, with verification cost independent of the size of the revoked set.

The framework is designed with a modular architecture that separates credential issuance, proof generation, verification, and revocation. This separation allows each component to operate independently while maintaining overall security guarantees. The system has been implemented using 2048-bit security parameters, incorporating issuer, holder, and multiple verifier entities, along with a lightweight integration of decentralized identifiers (DID) [10] and verifiable credentials (VC) [9]. This design ensures compatibility with modern identity ecosystems.

Experimental evaluation demonstrates that the system achieves efficient performance, including low verification latency and compact proof size, while maintaining strong privacy guarantees. These results indicate that it is possible to simultaneously achieve selective disclosure, unlinkability, and efficient revocation without relying on pairing-based cryptog-

raphy or trusted setup assumptions.

Compared to existing systems such as Idemix [4] and U-Prove [5], the proposed framework avoids pairing-based constructions and reduces infrastructure complexity, while strengthening unlinkability across independent verification domains. At the same time, it maintains efficient revocation through accumulator-based techniques without introducing persistent identifiers.

Contributions

The primary contributions of this work are summarized as follows:

- A modular zero-knowledge credential framework is developed for secure attribute verification across multiple independent systems, eliminating the need for globally stable identifiers and reducing the risk of cross-domain correlation.
- A scoped pseudonym construction is proposed to achieve strong cross-domain unlinkability, where domain-specific identifiers are deterministically derived from holder-controlled secrets without revealing underlying identity information.
- An RSA accumulator-based revocation mechanism is incorporated, supporting efficient non-membership proofs and ensuring that verification complexity remains independent of the size of the revoked set.
- A complete end-to-end system design and implementation is presented, demonstrating that selective disclosure, unlinkability, and revocation can be achieved simultaneously under standard cryptographic assumptions without requiring trusted setup.
- An empirical evaluation is conducted to assess system performance, showing millisecond-level verification latency and compact proof sizes, thereby confirming the practical feasibility and scalability of the proposed framework.

II. Related Work

The development of privacy-preserving identity systems is closely connected to advances in Zero-Knowledge Proof (ZKP) systems, which allow verification of statements over secret data without revealing the data itself. Over time, a wide range of cryptographic constructions has been proposed, differing in their efficiency, level of expressiveness, trust assumptions, and practical deployability.

A. Anonymous Credential Systems

Anonymous credential systems are designed to allow users to prove possession of certified attributes while preserving anonymity. A key requirement in such systems is *selective disclosure*, where only necessary information is revealed during verification.

The Idemix system [4] is one of the earliest and most influential approaches in this area. It is based on Camenisch–Lysyanskaya (CL) signatures defined over bilinear groups. These groups support pairing operations, which are mathematical mappings used to construct advanced cryptographic protocols. Idemix provides strong privacy guarantees, including anonymity and unlinkability, but introduces significant computational overhead due to the use of pairing operations and complex proof constructions. In addition, its deployment requires careful parameter selection and a relatively complex cryptographic setup.

U-Prove [5] follows a different design based on discrete logarithm assumptions. It provides efficient credential issuance and verification, making it suitable for lightweight environments. However, unlinkability in U-Prove depends on careful management of randomness and pseudonyms. In practice, repeated usage or improper randomness can lead to correlation across sessions, which weakens privacy guarantees in multi-verifier settings.

More recent constructions based on BBS+ signatures [7] support efficient multi-attribute selective disclosure with short proof sizes. These systems are widely used in modern verifiable credential frameworks. However, they rely on bilinear pairings and structured reference strings, which introduce additional computational cost and increase deployment complexity, particularly in resource-constrained or distributed environments.

B. General-Purpose Zero-Knowledge Proof Systems

General-purpose ZKP systems aim to provide proofs for arbitrary computations rather than specific predicates.

zk-SNARKs [6] provide succinct proofs with fast verification and constant proof size. However, most zk-SNARK constructions require a trusted setup phase. This setup generates system parameters that must remain secret; if compromised, the security of the entire system can be broken. Furthermore, expressing even simple predicates requires conversion into arithmetic circuits, which increases implementation complexity.

zk-STARKs address the trusted setup limitation by relying on transparent assumptions and providing post-quantum security. However, this comes at the cost of significantly larger proof sizes and higher verification overhead. As a result, they are less suitable for systems where bandwidth and latency are critical.

Bulletproofs [8] provide efficient range proofs without requiring trusted setup. They are particularly useful for proving statements such as $a \geq t$. However, the verification cost grows linearly with proof size, which makes them less suitable for scenarios involving frequent verification across multiple independent systems.

C. Revocation Mechanisms in Credential Systems

Revocation is a fundamental requirement in credential systems, as it allows invalid or compromised credentials to be excluded from future use. However, achieving revocation without compromising user privacy remains challenging.

Traditional revocation approaches include certificate revocation lists (CRLs), which maintain lists of revoked credentials, and online verification mechanisms that require communication with the issuer during each authentication. These methods introduce scalability issues and may expose user activity patterns.

Accumulator-based revocation schemes [3] provide an alternative approach by representing a set of revoked elements in a compact form. Using cryptographic accumulators, it is possible to generate proofs of membership or non-membership that are independent of the size of the set. This property enables efficient and scalable verification. However, existing integrations often introduce linkability through persistent identifiers or require coordination between verifiers, which can weaken privacy guarantees.

D. Limitations of Existing Approaches

Despite extensive research, existing systems exhibit several limitations when applied to multi-system identity verification:

- **Dependence on Pairings:** Many credential systems rely on bilinear pairings, which increase computational overhead and limit deployment on devices with constrained resources.
- **Trusted Setup Requirements:** Some proof systems depend on trusted setup phases, introducing additional trust assumptions that are not suitable for decentralized environments.
- **Weak or Partial Unlinkability:** In several systems, pseudonym reuse or deterministic constructions can enable correlation across sessions, especially under verifier collusion.
- **Inefficient Revocation:** Traditional revocation techniques either increase verification complexity or introduce linkable identifiers that compromise privacy.
- **Limited Modularity:** Many systems tightly couple credential issuance, proof generation, and revocation, reducing flexibility and making system evolution difficult.

E. Positioning of This Work

The framework presented in this work is designed to address the above limitations through a modular construction based on standard cryptographic assumptions.

In contrast to existing approaches:

- The framework avoids pairing-based cryptography and relies on prime-order discrete logarithm groups, reducing computational complexity.
- The construction does not require a trusted setup, as it is based on sigma protocols transformed into non-interactive proofs using the Fiat–Shamir technique.
- Scoped pseudonyms are introduced to ensure strong unlinkability across independent verification domains, preventing cross-domain correlation.
- An RSA accumulator-based revocation mechanism is integrated, enabling efficient non-membership verification without introducing persistent identifiers.
- The system follows a modular architecture, allowing independent design and evolution of issuance, proof, and revocation components.

Existing systems typically optimize for specific properties such as efficiency, expressiveness, or privacy. However, achieving strong unlinkability across independent verification domains while simultaneously supporting efficient revocation and avoiding trusted setup remains a challenging combination. Unlike prior approaches that address these properties in isolation, the proposed framework combines unlinkability, efficient revocation, and deployability within a single construction, while avoiding both pairing-based cryptography and trusted setup assumptions.

This positioning highlights the framework as a practical and theoretically grounded approach for multi-system identity verification, where both privacy and efficiency are essential.

III. Motivation and Background

A. Direct Document-Based Verification

In many existing systems, identity verification is performed through direct sharing of documents. A user possessing identity documents creates digital copies, uploads them to a verifier, and the verifier receives and stores the complete data. This approach appears simple but introduces several fundamental risks.

The process exposes the entire document, even when only a small portion of the information is required. For example, proving that a user is above a certain age does not require disclosure of full identity details. Despite this, complete records are often transmitted and stored.

This model suffers from several limitations:

- **Full data exposure:** Every field and document is disclosed, regardless of necessity.
- **Uncontrolled storage:** Verifiers may retain data indefinitely without user awareness or consent.
- **Forgery risk:** Authenticity cannot be verified in a cryptographic sense, making forged documents difficult to detect reliably.
- **No selective disclosure:** Users cannot restrict which attributes are revealed.
- **Cross-verifier correlation:** Identical documents enable tracking across multiple services.

- **Loss of user control:** Once shared, the user no longer controls how the data is used or stored.

These limitations arise because the system is based on full data transfer rather than controlled verification.

B. Centralized Identity Management

Centralized identity systems rely on a trusted authority responsible for issuing, storing, and validating user credentials. While such systems simplify management and authentication, they introduce structural risks that directly impact privacy and security.

Since all identity information is managed by a single authority, a compromise at that point can expose the entire dataset. In addition, the authority can observe user activity across multiple services, creating a global view of user behavior.

The key limitations are summarized below:

- **Single point of failure:** A breach at the central authority can expose all user data.
- **Global visibility:** The identity provider can monitor interactions across different services.
- **Session linkability:** Repeated authentications can be correlated over time.
- **Trust concentration:** Control over identity is concentrated in a single entity.
- **Coercion risk:** The authority may be compelled to disclose data under external pressure.
- **Vendor lock-in:** Dependence on a single provider restricts system flexibility and portability.

These issues arise from centralized control and the reliance on persistent identifiers.

C. Limitations of Existing Approaches

Direct document sharing and centralized identity systems represent two dominant approaches to identity verification. However, both approaches fail to provide essential privacy and security guarantees.

Direct document sharing exposes complete identity data without any cryptographic assurance of authenticity or integrity. Centralized identity systems reduce exposure during transmission but introduce global observability and trust dependency.

As a result, these approaches fail to provide:

- Selective disclosure of attributes
- Cryptographic privacy guarantees
- Resistance to cross-domain correlation
- Protection against collusive action
- A secure and verifiable foundation for trusted interactions

These limitations highlight the need for systems that verify properties of data without revealing the data itself.

D. Real-World Failures

1) Large-Scale Data Breaches

Recent years have witnessed multiple large-scale data breaches exposing sensitive personal information of millions of users [21]–[23]. These incidents highlight fundamental structural weaknesses in traditional identity systems that rely on centralized storage and full data disclosure.

2) Representative Incidents

Aadhaar Data Exposure (2023): A reported breach exposed personal records of approximately 815 million Indian citizens, including names, phone numbers, and Aadhaar-linked identity data. The incident was allegedly linked to vulnerabilities in government-associated databases, demonstrating the risks of centralized identity repositories [21].

Cambridge Analytica Scandal (2018): Personal data from approximately 87 million Facebook users was harvested through third-party applications without informed consent and used for targeted political advertising. This case highlighted how data shared in one context can be repurposed across domains without user control [22].

Equifax Data Breach (2017): A major credit reporting agency suffered a breach affecting approximately 147 million individuals. Sensitive information, including social security numbers, birth dates, and addresses, was exposed, illustrating the systemic risks associated with centralized financial identity systems [23].

3) Root Cause Analysis

These incidents reveal recurring structural weaknesses:

- **Centralized Storage:** A single point of failure enables large-scale data exposure.
- **Over-Disclosure:** Systems collect and store more data than necessary for verification.
- **Persistent Identifiers:** Static identifiers enable cross-domain tracking and profiling.
- **Lack of Cryptographic Protection:** Data security relies on organizational safeguards rather than provable guarantees.

4) Privacy Impact

Once exposed, identity data cannot be effectively revoked or replaced. This enables long-term misuse, identity theft, and unauthorized profiling. Such risks persist indefinitely because sensitive attributes remain permanently linked to individuals.

Moreover, identity data is often reused across multiple domains, such as financial services, healthcare, and digital platforms. As a result, a single breach can cascade across multiple systems, amplifying the overall impact. This makes recovery from such incidents extremely difficult, if not impossible.

5) Motivation for Zero-Knowledge Systems

The above failures motivate a fundamental shift from data disclosure to proof-based verification. Zero-knowledge proof

systems eliminate the need to share raw identity data by enabling verification of predicates over hidden attributes.

$$\text{Verify}(f(a)) \quad \text{without revealing } a$$

This paradigm ensures that even if communication transcripts are exposed, no sensitive user data is leaked.

6) Residual Risks

While zero-knowledge systems eliminate exposure during verification, they do not:

- Prevent data breaches at the issuer level
- Address metadata leakage such as network identifiers or timing information

These aspects require complementary system-level protections.

7) Summary

Real-world data breaches demonstrate that privacy cannot be guaranteed through policy or access control alone. Cryptographic enforcement of data minimization, as achieved through zero-knowledge proofs, is essential for building secure and privacy-preserving identity systems.

E. Multi-Verifier Privacy Risks

In real-world deployments, users interact with multiple independent services. Even minimal disclosures across these interactions can accumulate into comprehensive user profiles.

This leads to:

- **Profile construction:** Aggregated interactions enable the construction of detailed user profiles.
- **Data correlation:** Information can be combined with external datasets to derive deeper insights.
- **Data sharing and monetization:** Collected data may be shared or sold without user control.
- **Cross-system linkability:** Activities across different services can be linked and tracked.
- **Inference attacks:** Sensitive or hidden attributes can be inferred from partial disclosures.

Observation: Repeated interactions across independent verifiers transform small disclosures into large-scale privacy leakage.

F. Attack Scenarios

*1) **Correlation Attack**:* If a user presents the same credential across multiple verifiers, deterministic identifiers (e.g., credential fingerprints) enable linkage:

$$FP_1 = FP_2 \Rightarrow \text{Same User}$$

This enables cross-organization tracking and profiling.

Mitigation: Use scoped pseudonyms:

$$ID_S = pk^{H(S)}$$

*2) **Replay Attack**:* A replay attack occurs when a valid credential presentation is reused by an adversary.

If proofs are not bound to session-specific randomness, the same proof can be reused:

$$\sigma = (\pi, ID_S)$$

Mitigation: Bind proofs to nonce and verifier identity:

$$c = H(T, C, V, \text{nonce})$$

*3) **Verifier Collusion**:* Multiple verifiers may share data to reconstruct full user profiles. Even partial disclosures across domains can be combined into sensitive composite information.

Root Cause: Lack of unlinkability and data isolation.

*4) **Linkability Attack**:* Repeated use of deterministic identifiers enables tracking across sessions, creating a unified view of user activity.

This allows long-term monitoring of user behavior across different services, effectively removing anonymity.

Mitigation: Use scoped pseudonyms:

$$ID_S = pk^{H(S)}$$

*5) **Metadata Leakage**:* Even when credentials remain hidden, external information such as timing, network identifiers, or device characteristics can reveal user identity.

For example, repeated access from the same device or network location can allow correlation even if the credential itself remains anonymous.

Mitigation: Combine cryptographic protection with network-level privacy mechanisms.

*6) **Issuer Tracking**:* If the issuer can observe credential usage, user activity can be tracked globally.

This may occur through embedded identifiers, back-channel communication, or logging mechanisms.

Mitigation: Use zero-knowledge proofs of credential possession to prevent issuer visibility.

*7) **Adaptive Inference Attack**:* An adversarial verifier may adapt queries across multiple sessions to gradually infer hidden attributes.

Even when each interaction reveals minimal information, combining multiple responses can reduce uncertainty about the underlying attribute.

Root Cause: Repeated interaction without sufficient disclosure control.

Mitigation: Limit repeated queries and ensure proofs reveal only necessary predicates without additional leakage.

G. Design Requirements

The limitations discussed in the previous sections indicate that a credential system must enforce privacy and security

at the level of system design, rather than relying on policy or access control. To address these challenges, a privacy-preserving credential framework must satisfy the following fundamental requirements:

- **Selective Disclosure:** The system must allow a user to reveal only the minimum necessary information required for verification. Instead of exposing complete identity data, it should support proving specific properties of an attribute, such as satisfying a threshold condition, without disclosing the attribute itself.
- **Unlinkability:** Verification interactions across different services must remain independent. It should not be possible to determine whether multiple sessions correspond to the same user, thereby preventing cross-domain tracking and long-term profiling.
- **Replay Resistance:** Each authentication instance must be bound to a specific session. A previously valid proof should not be reusable by an adversary, ensuring that intercepted data cannot be used for impersonation.
- **Collusion Resistance:** The system must prevent multiple verifiers from combining their observations to infer hidden information about a user. Even if partial information is shared, it should not lead to reconstruction of sensitive attributes or user identity.

These requirements collectively reflect the need for verification mechanisms that operate without direct exposure of identity data. Zero-knowledge proofs provide a natural foundation for achieving these properties, as they enable verification of statements while preserving the secrecy of the underlying information.

IV. Threat Model and Security Objectives

A distributed multi-verifier environment is considered, where credential presentations take place across independent verification domains. The system involves three primary entities: an *Issuer* $\mathcal{I}$, a *Holder* $\mathcal{H}$, and multiple *Verifiers* $\mathcal{V}_i$, each associated with a distinct scope (S_i).

A. Adversarial Capabilities

Adversaries are modeled with the following capabilities:

- **Malicious Verifier:** A verifier may attempt to extract additional information from proofs or reuse previously observed protocol transcripts.
- **Honest-but-Curious Verifiers:** Verifiers correctly follow the protocol but attempt to infer additional information from observed data, including attribute values or correlations across sessions.
- **Cross-Domain Collusion:** A subset of verifiers may share transcripts and derived identifiers to link user activity across different scopes.
- **Network Adversary:** An adversary may observe, intercept, and replay communication between the holder and verifiers, but cannot break underlying cryptographic assumptions.
- **Revocation Observation:** Adversaries may observe accumulator updates and revocation values, but cannot generate valid witnesses without violating the Strong RSA assumption.
- **Malicious Holder:** A holder may attempt to prove predicates for attributes that were not issued or attempt to bypass revocation constraints.

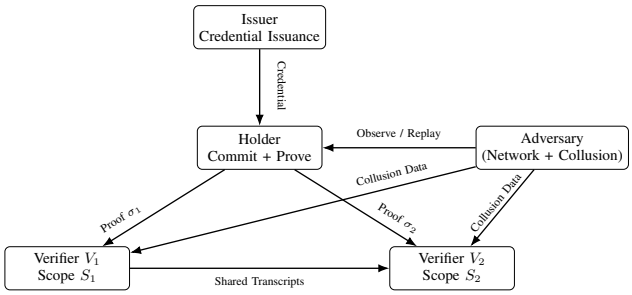

Fig. 1: Adversarial model in a multi-verifier setting. The holder interacts with independent verifiers under distinct scopes, while adversaries may observe, replay, or correlate transcripts across domains.

*a) **Adversarial Model Discussion**:* Figure 1 illustrates the interaction between system entities and adversarial behavior. The adversary may act as a network observer or as a colluding participant across verification domains.

The primary risk arises from cross-domain interactions. Even when each verifier observes only limited information, combining multiple transcripts may reveal patterns unless unlinkability is enforced.

The adversarial capabilities can be grouped into three categories:

- **Replay Attacks:** Reuse of previously valid protocol transcripts,
- **Verifier Collusion:** Sharing of observations across domains,
- **Network Observation:** Passive monitoring and traffic analysis.

These threats are addressed through nonce-bound proofs, scoped pseudonyms, and zero-knowledge constructions that prevent reuse and correlation.

The issuer is assumed to be honest, and private keys of honest holders are assumed to remain secure.

B. Attack Vectors

*1) **Replay Attack**:* A replay attack occurs when a valid credential presentation is captured and reused.

$$\sigma = (\pi, ID_S, \text{metadata})$$

Without session binding, the same proof may be accepted multiple times.

Mitigation:

$$\pi = \text{ZKP}(a; c, V)$$

where c is a verifier-generated challenge and V denotes verifier identity.

*2) **Correlation Attack**:* Deterministic identifiers enable linkage across sessions:

$$FP_1 = FP_2 \Rightarrow \text{Same User}$$

Mitigation:

$$ID_S = pk^{H(S)}$$

*3) **Verifier Collusion**:* Multiple verifiers may combine observations:

$$\text{Profile}_1 \cup \text{Profile}_2 \Rightarrow \text{Expanded Profile}$$

Mitigation: Unlinkable presentations and absence of shared identifiers prevent meaningful aggregation.

*4) **Metadata Leakage**:* External information such as timing or network identifiers may enable re-identification.

Mitigation: Network-level privacy mechanisms complement cryptographic protection.

*5) **Issuer Tracking**:* An issuer may attempt to track credential usage across verifiers.

Mitigation: Verification is performed without issuer involvement, preventing observation of credential usage.

C. Security Objectives

Under the defined adversarial model, the system satisfies the following properties:

1) **Attribute Privacy:** Protocol transcripts reveal no information about attribute values beyond the validity of the predicate, due to the hiding property of commitments and zero-knowledge proofs.
2) **Scoped Unlinkability:** Given transcripts from distinct scopes $S_1 \neq S_2$, no adversary can link them without solving the discrete logarithm problem in G.
3) **Soundness:** No adversary can produce a valid proof for a false statement without breaking underlying cryptographic assumptions.
4) **Revocation Soundness:** Revoked credentials cannot produce valid non-membership witnesses without violating the Strong RSA assumption.
5) **Replay Resistance:** Session-bound challenges prevent reuse of protocol transcripts.
6) **Collusion Resistance:** Even if multiple verifiers share transcripts, no adversary can reconstruct hidden attributes or reliably link sessions across domains beyond negligible probability.

D. Assumptions

Security relies on the following standard assumptions:

- Discrete logarithm assumption in group G
- Random oracle model for hash functions
- Strong RSA assumption for accumulator construction

E. Discussion

The model excludes fully malicious issuers and compromised holder keys, as these require separate trust mechanisms. Within the stated assumptions, the framework provides strong privacy guarantees while remaining practical for deployment.

V. System Model

A distributed credential verification system is considered, consisting of four primary entities:

- **Issuer** $\mathcal{I}$
- **Holder (Prover)** $\mathcal{H}$
- **Verifier** $\mathcal{V}$
- **Revocation Authority** $\mathcal{R}$

A. Entities

Issuer ($\mathcal{I}$).

The issuer is a trusted authority responsible for generating and issuing credentials. It generates a key pair (pk, x) and creates commitments to user attributes of the form:

$$C = g^a h^r$$

The issuer produces a credential signature:

$$\sigma_{\text{cred}} = \text{Sign}(x, C)$$

The issuer does not participate in the verification phase.

Security guarantee:

- Credentials cannot be forged without knowledge of the secret key x
- The issuer has no visibility into how credentials are used

Holder ($\mathcal{H}$).

The holder possesses:

$$(a, r, C, \sigma_{\text{cred}})$$

and generates proofs about the attribute a without revealing it.

Capabilities:

- Proves predicates of the form $f(a)$
- Generates zero-knowledge proof:

$$\pi = \text{ZKP}(a; c, V)$$

- Computes scoped pseudonym:

$$ID_S = pk^{H(S)}$$

Security guarantee:

- The attribute a remains hidden

- Interactions across verifiers remain unlinkable

Verifier ($\mathcal{V}$).

The verifier checks the validity of the proof without learning the attribute.

Verification:

$$b = \text{Verify}(\pi, c, V) \in \{0, 1\}$$

Responsibilities:

- Verify correctness of the proof
- Check credential signature σ_{cred}
- Ensure freshness using challenge c

Security guarantee:

- Only a binary decision is obtained
- No linkage across sessions is possible

Revocation Authority ($\mathcal{R}$).

The revocation authority maintains the validity status of credentials.

- Maintains revocation set R
- Maintains accumulator value A
- Supports non-membership verification

Goal: Efficient revocation without compromising privacy.

B. Credential Presentation

A credential presentation is defined as:

$$\sigma = (\pi, ID_S, \text{metadata})$$

where π is the proof and $ID_S = pk^{H(S)}$ is the scoped pseudonym. Only minimal auxiliary information is included, and no attribute values are disclosed.

C. System Workflow

This section presents the operational lifecycle of the zero-knowledge proof system. The protocol enables validation of a statement over a hidden attribute without disclosing the attribute itself.

System Phases

The system follows five sequential phases:

Setup.

Global public parameters are generated as:

$$pp \leftarrow \text{Setup}(1^\lambda)$$

The parameters include a cyclic group, generators (g, h), and a cryptographic hash function.

Credential issuance.

A commitment to the attribute is constructed as:

$$C = g^a h^r$$

where a denotes the secret attribute and r denotes randomness.

The commitment is signed by the issuer:

$$\sigma_{\text{cred}} = \text{Sign}(x, C)$$

The resulting credential is retained by the holder for subsequent use.

Proof generation.

A zero-knowledge proof is generated to establish a statement over a:

$$\pi = \text{ZKP}(a; c, V)$$

where c represents a challenge and V denotes the verifier identity.

The proof reveals no information about a or r.

Transmission.

Only the proof and minimal auxiliary data are transmitted:

$$\sigma = (\pi, ID_S, \text{metadata})$$

No underlying attribute values are disclosed.

Verification.

The verifier evaluates the proof as:

$$b = \text{Verify}(\pi, c, V) \in \{0, 1\}$$

where $b = 1$ indicates acceptance and $b = 0$ indicates rejection.

A revocation check may additionally be performed, such as verifying non-membership in an accumulator.

D. Statement, Witness, and Relation

A zero-knowledge proof is defined using the following components:

Statement ($stmt$).

The public claim under verification.

Example:

$$stmt = (a \geq 18)$$

The verifier learns only the validity of the statement.

Witness (w).

The secret information known exclusively to the prover.

Definition:

$$w = (a, r) \quad \text{or} \quad w = (\delta, r), \quad \text{where} \quad \delta = a - t$$

The witness remains hidden throughout the protocol.

Relation (Rel).

A polynomial-time computable relation defined as:

$$Rel(stmt, w) = 1 \iff (C = g^a h^r \ \wedge \ stmt(a) = 1)$$

E. Proof Objective

The objective of the prover is to demonstrate:

$$\exists w \text{ such that } Rel(stmt, w) = 1$$

without revealing the witness w.

F. Security Interpretation

The protocol satisfies standard zero-knowledge security properties:

- **Completeness:** Valid proofs are accepted
- **Soundness:** Invalid statements are rejected
- **Zero-Knowledge:** No information about w is disclosed

G. System Insight

The verifier obtains only a single-bit output:

$$b \in \{0, 1\}$$

while the attribute a remains hidden.

This approach eliminates unnecessary data exposure and addresses the **data amplification problem**, where conventional systems reveal excessive information for simple verification tasks.

H. Protocol Flow Diagram

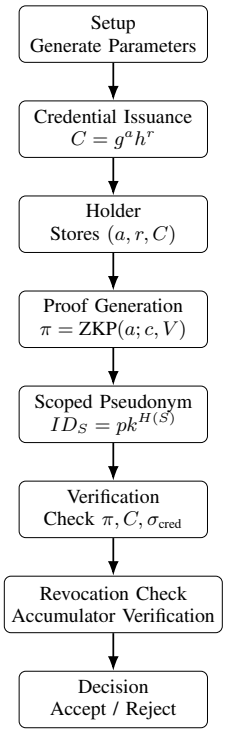

Fig. 2: End-to-end protocol flow of the proposed zero-knowledge credential system. The process includes setup, credential issuance, proof generation, scoped pseudonym derivation, verification, and accumulator-based revocation checking.

a) ***Protocol Flow Interpretation****:* Figure 2 illustrates the complete lifecycle of the credential system. The process begins with global setup and credential issuance, where user attributes are committed and signed by the issuer.

During the setup phase, public parameters are established, including the group structure, generators, and cryptographic hash function. These parameters define the environment in which all subsequent operations are performed. The issuer then generates a key pair and constructs commitments to user attributes in the form $C = g^a h^r$, where a represents the attribute and r provides randomness. The commitment is signed to produce a credential, which is later used by the holder for authentication.

During authentication, the holder generates a zero-knowledge proof bound to a verifier-specific nonce and identity, ensuring both privacy and replay resistance. A scoped pseudonym is derived to prevent cross-domain linkage.

More precisely, the holder computes a proof $\pi = \text{ZKP}(a; c, V)$, where the challenge c is derived from a verifier-specific context. This ensures that each proof instance is unique and cannot be reused across sessions. The scoped pseudonym $ID_S = pk^{H(S)}$ is computed using the scope identifier S, which represents the verification domain. This construction ensures that the same user produces different identifiers for different domains, thereby preventing linkage across verifiers.

The verifier then validates the proof and credential authenticity, followed by a revocation check using an accumulator-based non-membership proof. The final decision is based solely on verification outcomes without revealing any underlying attribute values.

The verification process involves checking the validity of the proof, confirming that the credential signature is correct, and ensuring that the proof corresponds to the current session. In addition, the verifier may perform a revocation check by verifying that the credential is not included in the revocation set. This is achieved using an accumulator-based mechanism, where non-membership proofs allow efficient validation without revealing the identity of revoked elements.

Throughout the entire process, the verifier learns only whether the required condition is satisfied. No information about the actual attribute value a, the randomness r, or the credential contents is disclosed. This ensures that verification is performed with minimal information exposure.

This flow separates credential issuance, proof generation, and verification into distinct stages. Such separation ensures that no single entity gains complete visibility into the system. The issuer does not observe verification events, the verifier does not access raw attributes, and the holder retains control over disclosure.

The protocol therefore achieves selective disclosure, unlinkability across domains, and resistance to replay and correlation

attacks, while maintaining efficient verification and compatibility with practical deployment requirements.

I. Interaction Model

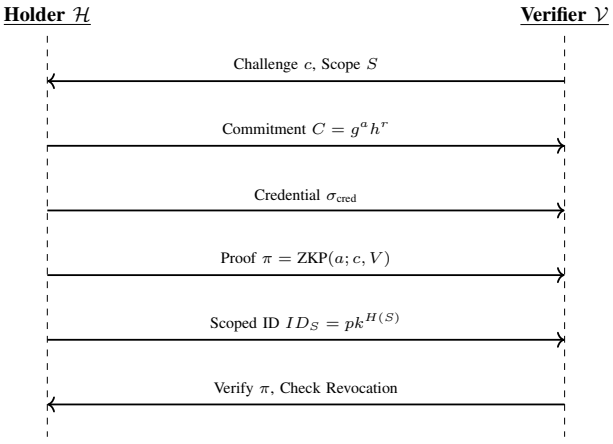

Fig. 3: Interaction protocol between holder $\mathcal{H}$ and verifier $\mathcal{V}$ illustrating commitment generation, zero-knowledge proof, scoped identifier derivation, and verification workflow.

*a) **Interaction Model**:* Figure 3 illustrates the message-level interaction between the holder and verifier. The verifier initiates the protocol by issuing a nonce and scope identifier, ensuring freshness and domain separation.

The nonce is a randomly generated value that is unique for each session. Its purpose is to ensure that every interaction remains fresh and cannot be reused in a different context. The scope identifier represents the verification domain and distinguishes one verifier from another. This separation ensures that interactions across different domains remain independent.

The holder responds by transmitting a commitment, a zero-knowledge proof of the predicate, and a scoped pseudonym derived from its secret key. The verifier then performs proof verification, credential validation, and revocation checking before producing a final decision.

The commitment represents the encoded attribute in the form $C = g^a h^r$, where a is the attribute and r is the associated randomness. This structure ensures that the attribute remains hidden while still allowing verification of statements about it. The zero-knowledge proof allows the holder to demonstrate that a required condition holds, such as a threshold predicate, without revealing the underlying value. The scoped pseudonym is derived as $ID_S = pk^{H(S)}$, where the hash of the scope ensures that the identifier is unique for each verification domain.

During verification, the verifier evaluates the received proof to confirm that the stated predicate is satisfied. The credential signature is also checked to ensure that the commitment was issued by a legitimate authority. In addition, a revocation check is performed to confirm that the credential has not been invalidated. This check is typically implemented using an accumulator-based mechanism, which allows efficient validation without exposing any information about other credentials.

This interaction ensures that all transmitted data is non-linkable across sessions and reveals no information beyond the validity of the proven predicate.

Each execution of the protocol is independent. Even if multiple interactions are observed, the absence of persistent identifiers prevents correlation across sessions. The use of fresh nonces ensures that previously valid proofs cannot be reused, while scoped pseudonyms prevent linkage across different verification domains. As a result, the interaction model supports secure and privacy-preserving authentication in a multi-verifier environment.

J. Security Objectives

The system satisfies:

- **Correctness:** Valid proofs generated by an honest holder are always accepted by the verifier. This ensures that legitimate users are able to successfully complete the verification process.
- **Soundness:** Invalid proofs are rejected. An adversary cannot construct a valid proof for a false statement without violating the underlying cryptographic assumptions.
- **Zero-Knowledge:** No information about a is revealed during the proof. The verifier learns only whether the predicate is satisfied, and nothing about the actual value of the attribute or the associated randomness.
- **Unlinkability:** Sessions cannot be linked. Interactions across different verifiers or across multiple sessions cannot be correlated to identify the same user.
- **Replay Resistance:** Each proof is bound to a challenge c, which is derived from session-specific information. This prevents reuse of previously valid proofs in new sessions.

These properties collectively ensure that the system provides secure and privacy-preserving verification. The design guarantees that correctness and soundness are maintained while protecting sensitive information from disclosure. At the same time, unlinkability and replay resistance ensure that repeated interactions do not introduce additional privacy risks.

VI. Preliminaries and Cryptographic Foundations

A. Cryptographic Notation

This section introduces the notation used throughout the paper. Each symbol is defined here so that its meaning remains consistent in all subsequent sections.

- a : Secret attribute associated with the holder
- r : Randomness used in commitment construction
- $C = g^a h^r$: Commitment to the attribute
- π : Zero-knowledge proof generated by the prover
- $\mathcal{P}$: Proof generation algorithm
- σ : Credential presentation transmitted to the verifier
- c : Challenge value derived during the proof protocol

- V : Verifier identity
- λ : Security parameter controlling cryptographic strength
- H : Cryptographic hash function
- g, h : Generators of the group G
- g_A : Generator used in RSA accumulator
- x : Secret key of the issuer
- pk : Public key corresponding to x
- S : Scope or domain identifier
- ID_S : Scoped pseudonym derived for scope S
- t : Public threshold
- δ : Transformed attribute value, typically defined as $a - t$
- A : Accumulator value used for revocation
- R : Revocation set
- r_i : Elements of the revocation set
- N : RSA modulus used in accumulator construction
- FP : Credential fingerprint
- T : Ephemeral commitment used during proof generation
- s : Response value in the Schnorr protocol
- $\mathcal{I}$: Issuer
- $\mathcal{H}$: Holder
- $\mathcal{V}$: Verifier
- $\mathcal{R}$: Revocation authority
- σ_{cred} : Credential signature issued by $\mathcal{I}$
- pp : Public parameters of the system
- $\mathcal{L}$: Language of valid statements
- w : Witness corresponding to a statement
- $stmt$: Public statement to be verified
- Rel : Relation defining validity between $stmt$ and w
- $\mathcal{A}$: Adversary
- HE : Homomorphic encryption
- Enc$(\cdot)$: Encryption algorithm
- Dec$(\cdot)$: Decryption algorithm
- Eval$(\cdot)$: Evaluation algorithm
- m : Message
- e_1 : Encryption randomness
- e_2 : Ciphertext component
- CT : Ciphertext
- C' : Adjusted commitment used in transformed relations
- C_1, C_2 : Commitments to different attributes
- u, v : Bézout coefficients used in accumulator proofs
- $\mathcal{C}$: Challenger in security games
- $\mathcal{B}$: Reduction algorithm
- r_1, r_2 : Random values used in proof commitment phase
- n : Number of attributes

B. Preliminaries

Let G be a cyclic subgroup of $\mathbb{Z}_p^*$ of prime order q generated by g. Let $h \in G$ be a randomly chosen independent generator such that $\log_g h$ is unknown.

The requirement that $\log_g h$ remains unknown ensures that no efficient relation between the generators g and h can be computed. This property is essential for the security of commitment schemes used later, as it prevents an adversary from manipulating commitments by exploiting hidden structure between generators.

All group operations are written multiplicatively, and all exponentiations are performed over $\mathbb{Z}_q$ unless stated otherwise.

This convention simplifies notation and ensures that all algebraic operations remain within the same mathematical structure. It also guarantees that exponent values remain bounded by the group order.

For any integer $x \in \mathbb{Z}_q$, the notation g^x denotes exponentiation in the group G.

This notation is used consistently in commitment construction, proof generation, and pseudonym derivation.

Let $H : \{0,1\}^* \to \mathbb{Z}_q$ be a cryptographic hash function modeled as a random oracle.

The random oracle model assumes that the hash function behaves like a truly random function. This abstraction is commonly used to analyze the security of non-interactive proof systems and ensures that hash outputs cannot be predicted or controlled by an adversary.

Random values are assumed to be sampled uniformly from $\mathbb{Z}_q$.

Uniform sampling ensures that all possible values are equally likely, which is necessary for achieving unpredictability in commitments, challenges, and proof randomness.

Discrete Logarithm (DL) Assumption. Let $y \in G$ be such that $y = g^x$ for a uniformly random $x \in \mathbb{Z}_q$. We assume that for any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the following probability is negligible in the security parameter λ:

$$\Pr\left[x \xleftarrow{\$} \mathbb{Z}_q;\ y = g^x :\ \mathcal{A}(g, y) = x\right] \leq \text{negl}(\lambda),$$

where the probability is taken over the random choice of x and the internal randomness of $\mathcal{A}$, and $\text{negl}(\cdot)$ denotes a negligible function.

This assumption states that it is computationally infeasible to recover the exponent x from g and $y = g^x$. The hardness of this problem forms the basis of security for many cryptographic constructions.

This assumption forms the computational foundation of the construction, ensuring the binding property of commitments, the soundness of the proof system, and the security of pseudonym and signature mechanisms used throughout this work.

In particular, the binding property of commitments relies on the difficulty of finding two distinct openings that produce the same group element. The soundness of the proof system follows from the inability of an adversary to generate valid proofs without knowledge of the corresponding witness. The same hardness assumption also protects pseudonym generation and credential-related operations, preventing inversion or cross-domain linkage of identifiers.

C. Mathematical Background

Let G be a cyclic group of prime order q with generator g, and let $h \in G$ be an independent generator such that $\log_g h$ is unknown.

This setting is commonly used in discrete logarithm based constructions. The assumption that $\log_g h$ is unknown ensures that no efficient algebraic relation between g and h can be exploited.

1) ***Representation in Groups****:* For any element $u \in G$, a pair $(a, r) \in \mathbb{Z}_q^2$ is said to be a representation of u if:

$$u = g^a h^r$$

Such a representation expresses a group element as a combination of two generators with exponents taken from $\mathbb{Z}_q$.

Fact 1 (Multiplicity of Representation): Each element $u \in G$ admits exactly q distinct representations.

This follows from the fact that for any fixed value of a, there exists a unique value of r that satisfies the equation, and vice versa. Since r can take any value in $\mathbb{Z}_q$, the total number of valid representations is q.

Fact 2 (Relation Between Representations): If two representations (a, r) and (a', r') satisfy:

$$g^a h^r = g^{a'} h^{r'}$$

then:

$$g^{a-a'} = h^{r'-r}$$

which implies:

$$\log_g h = \frac{a - a'}{r' - r}$$

Security Consequence: The existence of two distinct valid representations of the same element enables efficient computation of $\log_g h$, thereby violating the discrete logarithm assumption.

This shows that finding two different openings of the same group element is computationally hard under the discrete logarithm assumption.

2) ***Commitment Interpretation****:* A commitment of the form:

$$C = g^a h^r$$

encodes the value a while concealing it using randomness r.

This form of commitment is known as a Pedersen commitment. It combines the attribute and randomness in a way that preserves secrecy while allowing algebraic manipulation required for proof construction.

This construction satisfies:

- **Hiding:** The value a is statistically hidden when r is uniformly sampled from $\mathbb{Z}_q$
- **Binding:** It is computationally infeasible to open C to a different value under the discrete logarithm assumption

3) ***Security Intuition****:* The hiding property follows from the randomness of r, which ensures that for a fixed commitment C, multiple values of a remain indistinguishable. An observer cannot determine which value was used because the randomness masks the contribution of the attribute.

The binding property follows from the uniqueness constraint implied by the discrete logarithm assumption. Any successful attempt to produce two valid openings of C directly leads to the computation of $\log_g h$, which contradicts the assumed hardness of the problem.

This algebraic structure forms the foundation of the zero-knowledge proof system. It enables statements about committed values to be verified without revealing the values themselves, which is essential for achieving selective disclosure and privacy preservation in the proposed framework.

D. Commitment Scheme

A commitment scheme allows a prover to commit to a value while keeping it hidden, with the ability to reveal it later. It provides a way to fix a value at one point in time while preventing both early disclosure and later modification.

1) ***Formal Definition****:* A commitment scheme is defined as a tuple:

$$\Gamma = (\text{Setup}, \text{Commit}, \text{Open})$$

- $\text{Setup}(1^\lambda) \to pp$
- $\text{Commit}(pp, a) \to (C, r)$
- $\text{Open}(pp, C, a, r) \to b \in \{0, 1\}$

2) ***Correctness****:* For all valid (a, r):

$$\text{Open}(pp, C, a, r) = 1$$

This property ensures that a correctly generated commitment can always be opened successfully using the original value and randomness.

3) ***Security Properties****:* The security of a commitment scheme is defined through two fundamental properties.

Hiding.

A commitment is said to be hiding if it does not reveal any information about the committed value. Formally,

$$\Pr[\mathcal{A}(C_0) = 1] \approx \Pr[\mathcal{A}(C_1) = 1]$$

for any $a_0, a_1 \in \mathbb{Z}_q$, where C_0 and C_1 are commitments to a_0 and a_1, respectively. This ensures that an adversary cannot distinguish which value was committed.

Binding.

A commitment is said to be binding if it is computationally infeasible to find two different openings of the same commitment. That is, it is hard to find:

$$(a, r) \neq (a', r')$$

such that:

$$g^a h^r = g^{a'} h^{r'}$$

This property guarantees that once a value is committed, it cannot be changed without detection.

In the construction used in this work, commitments are instantiated using the form:

$$C = g^a h^r$$

which is a Pedersen commitment. This scheme provides perfect hiding due to the randomness r, and computational binding under the discrete logarithm assumption.

These properties make the commitment scheme suitable for use in zero-knowledge proofs, where values must remain hidden while still allowing verification of statements about them.

E. Pedersen Commitment Scheme

The Pedersen commitment is defined as:

$$C = g^a h^r$$

where:

- $a \in \mathbb{Z}_q$ is the secret attribute
- $r \in \mathbb{Z}_q$ is randomness

This construction combines the attribute and randomness in a way that preserves secrecy while supporting algebraic operations required for proof generation.

1. Algorithms

Setup:

$$(pp) \leftarrow \text{Setup}(1^\lambda)$$

The public parameters pp include the group G, its order q, and the generators g and h.

Commit:

$$C = g^a h^r$$

Open:

$$\text{Check: } C \stackrel{?}{=} g^a h^r$$

2. Security Properties

Perfect Hiding.

The commitment does not reveal any information about a. For any two values $a_0, a_1 \in \mathbb{Z}_q$, the distributions of commitments to a_0 and a_1 are identical when r is chosen uniformly at random. This ensures that an adversary cannot distinguish which value was committed.

Computational Binding.

Binding holds under the discrete logarithm assumption. It is computationally infeasible to find two distinct pairs (a, r) and (a', r') such that:

$$g^a h^r = g^{a'} h^{r'}$$

This guarantees that a commitment cannot be opened to two different values.

3. Homomorphic Property

Pedersen commitments are additively homomorphic:

$$C_1 = g^{a_1} h^{r_1}, \quad C_2 = g^{a_2} h^{r_2}$$

Then:

$$C_1 \cdot C_2 = g^{a_1 + a_2} h^{r_1 + r_2}$$

Thus:

$$\text{Commit}(a_1) \cdot \text{Commit}(a_2) = \text{Commit}(a_1 + a_2)$$

4. Practical Significance

This homomorphic property enables efficient construction of zero-knowledge proofs. It allows arithmetic operations to be performed directly on committed values without revealing them. As a result, complex statements such as threshold conditions and aggregated proofs can be verified while preserving privacy.

F. Homomorphic Encryption

While the proposed framework primarily relies on commitment schemes and zero-knowledge proofs, homomorphic encryption (HE) is a related cryptographic primitive that enables computation over encrypted data. It is included here to provide context for alternative approaches to privacy-preserving computation.

A homomorphic encryption scheme is defined as:

$$\text{HE} = (\text{KeyGen}, \text{Enc}, \text{Dec}, \text{Eval})$$

such that for a function f:

$$\text{Dec}(\text{Eval}(f, \text{Enc}(m_1), \ldots, \text{Enc}(m_n))) = f(m_1, \ldots, m_n)$$

This property allows operations to be performed directly on encrypted data without revealing the underlying plaintext values.

Homomorphic encryption schemes are commonly classified into three types:

- **Partially Homomorphic Encryption (PHE):** Supports a single operation, such as addition or multiplication
- **Somewhat Homomorphic Encryption (SHE):** Supports a limited number of operations
- **Fully Homomorphic Encryption (FHE):** Supports arbitrary computations on encrypted data

A standard example is the ElGamal encryption scheme. Let G be a cyclic group of order q.

Key Generation:

$$x \leftarrow \mathbb{Z}_q, \quad pk = g^x$$

Encryption:

$$r \leftarrow \mathbb{Z}_q$$

$$e_1 = g^r, \quad e_2 = m \cdot pk^r$$

$$CT = (e_1, e_2)$$

Decryption:

$$m = \frac{e_2}{e_1^x}$$

ElGamal encryption is multiplicatively homomorphic:

$$\text{Enc}(m_1) \cdot \text{Enc}(m_2) = \text{Enc}(m_1 \cdot m_2)$$

This property shows that encrypted values can be combined without requiring decryption.

In the context of this framework, homomorphic encryption is not required for core protocol execution. The necessary algebraic structure is already provided by Pedersen commitments, and secure computation is achieved through zero-knowledge proofs.

However, homomorphic encryption can be integrated in extended settings where computation is outsourced to untrusted environments. In such cases, it allows evaluation of functions over encrypted inputs while preserving confidentiality.

Despite its flexibility, homomorphic encryption introduces practical limitations. These include high computational overhead, large ciphertext sizes, and complex parameter selection, especially in fully homomorphic settings. As a result, the proposed framework avoids reliance on HE in order to maintain efficiency while preserving strong privacy guarantees.

G. Schnorr Signature Scheme

The credential issuance process relies on the Schnorr signature scheme [2], instantiated over a cyclic group G of prime order q with generator g. The scheme provides efficient signing and verification while ensuring security under the discrete logarithm assumption in the random oracle model.

The Schnorr construction is well suited to this framework due to its simplicity and its compatibility with discrete logarithm based commitments and proof systems.

Key Generation: The issuer samples a secret key $x_I \in \mathbb{Z}_q$ and computes the corresponding public key:

$$pk_I = g^{x_I}.$$

The secret key remains known only to the issuer, while the public key is used by verifiers to validate issued credentials.

Signing: To sign a message $m = \text{Encode}(C)$, the issuer proceeds as follows:

$$r \xleftarrow{\$} \mathbb{Z}_q, \quad T = g^r,$$

$$c = H(T, m), \quad s = r + cx_I \mod q.$$

The resulting signature is:

$$\sigma = (T, s).$$

The value T serves as an ephemeral commitment, while s binds the randomness and the secret key through the challenge c. The hash function ensures that the signature is non-interactive and binds the message to the signature.

Verification: Given (T, s), the verifier recomputes:

$$c = H(T, m),$$

and accepts the signature if:

$$g^s \stackrel{?}{=} T \cdot pk_I^c.$$

This verification equation ensures that the signature is consistent with the public key and the signed message, without revealing the issuer's secret key.

Security Properties: The Schnorr signature scheme is existentially unforgeable under chosen message attacks (EUF-CMA) in the random oracle model. Its security reduces to the hardness of the discrete logarithm problem in G.

This guarantee ensures that no efficient adversary can produce a valid signature on a new message without access to the issuer's secret key. In the context of this framework, this property is essential to prevent forgery of credentials.

In addition to security, the scheme offers low computational overhead and compact signature size. These characteristics make it well suited for credential issuance in systems that require both efficiency and strong privacy guarantees.

H. Cryptographic Accumulators

Cryptographic accumulators provide a compact representation of a dynamic set while enabling efficient membership and non-membership verification. In this work, an RSA-based accumulator is used to support privacy-preserving revocation.

Let $R = \{r_i\}$ denote the set of revoked credentials, where each element is mapped to a unique prime representative r_i using a hash-to-prime function. This mapping ensures that each credential corresponds to a distinct value that can be securely incorporated into the accumulator.

Accumulator Construction: The accumulator value is defined as:

$$A = g_A^{\prod_{r_i \in R} r_i} \mod N,$$

where $g_A \in \mathbb{Z}_N^*$ is a generator and N is an RSA modulus.

This construction compresses the entire revoked set into a single value A, allowing verification without explicitly storing or transmitting all elements of R.

Non-Membership Witness: For a credential represented by a prime $e \notin R$, the holder computes Bézout coefficients (u, v) such that:

$$ue + v \prod_{r_i \in R} r_i = 1.$$

These coefficients exist because e is co-prime with the product $\prod_{r_i \in R} r_i$. This condition holds as long as $e \notin R$.

Using these coefficients, the non-membership witness is constructed as:

$$w = g_A^u \mod N.$$

The value w serves as a proof that the element e is not included in the accumulated set.

Verification: Given (w, e), the verifier checks:

$$w^e \cdot A^v \equiv g_A \mod N.$$

This equation follows directly from Bézout's identity and confirms that e is not part of the accumulated set without revealing any additional information about other elements.

Security and Efficiency: The correctness of the construction follows from Bézout's identity, while security relies on the Strong RSA assumption. Under this assumption, it is computationally infeasible to extract roots modulo N without knowledge of the factorization of N.

The verification procedure is independent of the size of the revoked set $|R|$, ensuring constant-time verification. This property makes the approach scalable even when the revocation list grows large.

In the context of this framework, the accumulator enables revocation checking without revealing credential identities or introducing linkable information. As a result, revocation can be enforced while preserving unlinkability and privacy across verification sessions.

I. Scoped Pseudonym Construction

To ensure privacy-preserving authentication across multiple verification domains, the system employs scoped pseudonyms that are uniquely bound to a given context while remaining unlinkable across different scopes.

Let the holder possess a secret key $x \in \mathbb{Z}_q$ with corresponding public key:

$$pk_H = g^x.$$

For a given scope S, the holder computes a scoped pseudonym as:

$$ID_S = pk_H^{H(S)}$$

The value ID_S serves as a deterministic identifier that is specific to the scope S. This means that for a fixed scope, the same holder always produces the same pseudonym, enabling consistent identification within that domain. At the same time, different scopes produce different pseudonyms, ensuring separation across domains.

Since the exponent $H(S)$ is derived from a cryptographic hash function, each scope is mapped to a value that behaves like a random element in $\mathbb{Z}_q$. As a result, pseudonyms generated for different scopes are computationally independent.

Security Analysis: Under the random oracle model, the hash outputs $H(S_1)$ and $H(S_2)$ for distinct scopes $S_1 \neq S_2$ are computationally independent. Consequently, the corresponding pseudonyms:

$$ID_{S_1} = g^{xH(S_1)}, \quad ID_{S_2} = g^{xH(S_2)}$$

cannot be linked without recovering the secret exponent x, which reduces to solving the discrete logarithm problem in G.

This property ensures that observations of pseudonyms across different scopes do not reveal whether they belong to the same holder. At the same time, within a single scope, the pseudonym remains stable, allowing repeated interactions to be recognized without exposing the underlying identity.

Therefore, the construction achieves strong unlinkability across domains while maintaining deterministic consistency within a given scope. This balance is essential for systems that require both privacy and controlled re-identification within a defined context.

J. Zero-Knowledge Proof System

A Zero-Knowledge Proof (ZKP) is a protocol between a prover $\mathcal{H}$ and a verifier $\mathcal{V}$ for a language $\mathcal{L}$. The objective of the protocol is to allow the prover to convince the verifier that a given statement is valid, without revealing any additional information beyond its validity.

Goal: Prove that:

$$stmt \in \mathcal{L}$$

without revealing the witness $w = (a, r)$ or (δ, r). Here, $stmt = (C, t)$.

In this framework, the statement represents a commitment and a public threshold, while the witness consists of the hidden attribute and randomness used in the commitment, or the transformed value used for predicate verification.

1) Formal Definition: Let $Rel(stmt, w)$ be a polynomial-time computable relation such that:

$$\mathcal{L} = \{stmt = (C, t) \mid \exists w \text{ such that } Rel(stmt, w) = 1\}.$$

This relation defines when a statement is considered valid with respect to a corresponding witness.

A proof system (Prove, Verify) satisfies the following properties.

a) Completeness: For all $(stmt, w) \in Rel$:

$$\Pr[\text{Verify}(stmt, \pi) = 1 \mid \pi \leftarrow \text{Prove}(stmt, w)] = 1.$$

This property ensures that an honest prover, possessing a valid witness, can always convince the verifier.

b) Soundness: For any probabilistic polynomial-time adversary $\mathcal{A}$:

$$\Pr[\text{Verify}(stmt, \pi) = 1 \wedge stmt \notin \mathcal{L}] \leq \epsilon(\lambda),$$

where $\epsilon(\lambda)$ is a negligible function in the security parameter.

This guarantees that no efficient adversary can produce a convincing proof for an invalid statement.

c) Zero-Knowledge: There exists a simulator Sim such that:

$$\text{View}_{\mathcal{V}}(stmt, \pi) \approx \text{Sim}(stmt).$$

This means that the verifier's view of the protocol can be simulated without access to the witness. As a result, the verifier learns nothing beyond the validity of the statement.

These properties together ensure that the proof system provides correctness, security against forgery, and strong privacy guarantees. In particular, the zero-knowledge property ensures that sensitive values such as a, r, or δ remain hidden during the verification process.

2) Zero-Knowledge Proof Properties: Zero-knowledge proof (ZKP) systems provide strong cryptographic guarantees that enable secure verification without disclosure of sensitive data. The proposed framework relies on the following fundamental properties.

a) Zero-Knowledge: A proof reveals no information about the underlying secret beyond the validity of the statement being proven (Goldwasser, Micali, Rackoff, 1985) [39]. Formally, for any verifier, there exists a simulator that can generate transcripts indistinguishable from real protocol executions without access to the witness. This ensures that sensitive attributes remain hidden during verification.

b) Completeness: If the prover is honest and possesses a valid witness, the verifier accepts the proof with probability 1:

$$\Pr[\text{Verify}(\pi) = 1] = 1.$$

This guarantees that valid statements are always accepted under correct execution of the protocol.

c) Soundness: A malicious prover cannot convince the verifier of a false statement except with negligible probability:

$$\Pr[\text{Verify}(\pi) = 1 \mid stmt \notin \mathcal{L}] \leq \epsilon(\lambda),$$

where $\epsilon(\lambda)$ is negligible in the security parameter.

This property ensures that invalid statements cannot be accepted by any efficient adversary.

d) Unlinkability: Each proof instance is computationally unlinkable across different verification sessions. This is achieved through the use of fresh randomness in proof generation and domain-separated pseudonyms. As a result, multiple proofs generated by the same holder cannot be correlated by an adversary.

e) Non-Interactivity: The system employs the Fiat–Shamir transformation to convert interactive sigma protocols into non-interactive proofs:

$$c = H(T, C, t, V, \text{nonce}).$$

This construction replaces verifier interaction with a hash-based challenge, enabling single-message proofs suitable for asynchronous and distributed environments.

f) Selective Disclosure: The framework supports predicate-based verification, allowing users to prove statements about attributes, such as $a \geq t$, without revealing the attribute itself. This ensures that only the required information is disclosed during verification.

g) Security Against Honest-but-Curious Adversaries: The protocol ensures that even verifiers who follow the protocol correctly but attempt to infer additional information cannot learn anything beyond the validity of the statement. This follows directly from the zero-knowledge property of the proof system.

h) Scope of Cryptographic Guarantees: Zero-knowledge proofs protect data during the verification process by preventing leakage of sensitive information. However, they do not control how a verifier uses the outcome of verification. Such concerns lie outside the cryptographic model and must be addressed through policy, regulation, or system-level mechanisms.

3) Comparison of Major ZKP Protocols: Several zero-knowledge proof systems have been developed over time, each offering different trade-offs in terms of proof size, verification efficiency, setup requirements, and security assumptions. These differences make each protocol suitable for specific application scenarios.

Table I presents a comparison of widely used protocols. The comparison highlights key aspects that influence practical deployment, including whether a trusted setup is required, how large the proofs are, how efficiently they can be verified, and whether they provide resistance against quantum attacks.

Protocols such as Groth16 and PLONK belong to the class of succinct non-interactive arguments of knowledge. They offer very small proof sizes and fast verification, which makes them attractive for blockchain systems and applications where bandwidth and latency are critical. However, these systems rely on structured setup procedures, and their security is based on assumptions that are not resistant to quantum adversaries.

In contrast, zk-STARKs eliminate the need for a trusted setup and provide post-quantum security. This is achieved by relying on hash-based constructions instead of number-theoretic assumptions. The trade-off is that proof sizes are significantly larger and verification requires more computational resources.

Bulletproofs provide a different approach by avoiding trusted setup while supporting efficient range proofs. They are particularly useful in applications such as confidential transactions. However, their verification cost is higher compared to succinct proof systems.

Signature-based constructions such as BBS+ are designed for credential systems. They enable selective disclosure and

efficient verification, making them well suited for identity-related applications. Their design aligns closely with privacy-preserving authentication frameworks.

This comparison provides context for the design choices made in this work. The proposed construction prioritizes efficiency, minimal disclosure, and practical deployment without introducing heavy setup requirements or excessive computational overhead.

4) Protocol Selection Rationale: Based on the comparison presented earlier, the selection of suitable proof mechanisms in this work is guided by a set of practical and security-driven requirements.

The focus is on protocols that support:

- Efficient proof generation and verification
- No or minimal trusted setup assumptions
- Support for selective disclosure

These requirements arise from the need to design a system that remains practical for real-world deployment while preserving strong privacy guarantees. In particular, efficient proof generation ensures that the holder can produce proofs without significant computational burden, while efficient verification allows the verifier to process requests with low latency. Minimizing trusted setup assumptions reduces reliance on external trust and avoids risks associated with setup compromise.

Selective disclosure is a central requirement in credential systems. It allows a holder to prove statements about attributes without revealing the attributes themselves. This capability is essential for privacy-preserving authentication, where only the necessary information should be disclosed during verification.

In this context, BBS+ signature schemes are well suited for credential systems due to their native support for multi-attribute selective disclosure. They allow a holder to derive proofs that reveal only chosen attributes while keeping the remaining attributes hidden. This makes them particularly effective for identity and credential-based applications.

On the other hand, zk-SNARK-based constructions provide a more general framework for expressing complex predicates. They enable the verification of arbitrary statements over committed data, which is useful when the application requires flexible and expressive proof logic beyond simple attribute disclosure.

The choice between these approaches depends on the specific requirements of the application. Systems that prioritize lightweight credential presentation and minimal disclosure benefit from signature-based constructions, while applications requiring complex logical conditions may rely on zk-SNARK-based techniques.

Overall, the design adopted in this work emphasizes a balance between efficiency, limited trust assumptions, and strong privacy guarantees. This balance ensures that the system remains both secure and practical for deployment in environments where scalability and user privacy are critical.

5) ZKP vs Traditional Identity Systems: A comparison between traditional identity verification systems and zero-knowledge proof (ZKP) based systems is presented in Table II. The comparison highlights differences in data disclosure, privacy guarantees, trust assumptions, and underlying security models.

Traditional identity verification systems rely on direct disclosure of credentials. In such systems, users are typically required to share complete documents or full attribute sets, such as identity cards or account details. This approach leads to over-exposure of sensitive information and increases the risk of misuse, unauthorized storage, or data leakage.

In contrast, ZKP-based systems enable verification without revealing the underlying data. Instead of transmitting raw credentials, the holder generates a proof π that demonstrates the validity of a statement. The verifier can check this proof without learning any additional information about the hidden attributes.

As shown in Table II, traditional systems exhibit several structural limitations. These include over-disclosure of data, reliance on persistent identifiers, and the ability to link user activity across different services. Such properties enable tracking, profiling, and large-scale aggregation of personal data, which can lead to significant privacy concerns.

ZKP-based systems address these issues at the design level. Each proof instance is constructed in a way that limits information exposure and prevents reuse across sessions. In particular, proofs in the proposed framework are:

- **Non-interactive and self-contained**, requiring no repeated communication once generated
- **Bound to a session-specific nonce**, ensuring freshness and preventing replay attacks
- **Unlinkable across verifiers**, preventing correlation of user activity across domains

As a result, the verifier obtains only a binary decision regarding the validity of the statement, without access to sensitive data or reusable identifiers. This reflects a fundamental shift in the verification model. Traditional systems establish trust by exposing information, whereas ZKP-based systems establish trust through cryptographic proof of correctness.

K. Sigma Protocols and Proof of Knowledge

Sigma protocols are efficient three-move public-coin protocols that allow a prover to demonstrate knowledge of a secret without revealing the secret itself. These protocols form the foundation of many practical zero-knowledge constructions due to their simplicity and strong security guarantees.

1) Protocol Structure: A Sigma protocol follows a three-step interaction:

$$\text{Commit} \rightarrow \text{Challenge} \rightarrow \text{Response}$$

TABLE I: Comparison of Major Zero-Knowledge Proof Protocols

Protocol	Type	Trusted Setup	Proof Size	Verify Time	PQ Secure	Typical Use Cases
Groth16 (2016) [40]	zk-SNARK	Yes	∼192 bytes	∼1 ms	No	Ethereum L2, Zcash
PLONK (2019) [41]	zk-SNARK	Universal	∼500 bytes	∼5 ms	No	General-purpose circuits, zkSync
zk-STARKs (2018) [42]	zk-STARK	None	10–200 KB	∼50 ms	Yes	StarkNet, scalable systems
BBS+ (2023) [43]	Signature-based	CRS only	∼112 bytes	<1 ms	No	Verifiable credentials, selective disclosure
Bulletproofs (2017) [8]	NIZK (Range)	None	∼1 KB	∼100 ms	No	Confidential transactions, range proofs

TABLE II: Comparison Between Traditional Systems and ZKP-Based Systems

Requirement	Traditional Systems	ZKP-Based Systems
Data Disclosure	Full credentials or documents are shared	Only proof π is shared
Privacy Preservation	Sensitive data is exposed	Zero-knowledge property ensures no raw data is revealed
Selective Disclosure	Not supported; all attributes must be revealed	Fine-grained predicate-based proofs supported
Linkability Across Sessions	Static identifiers enable tracking	Fresh proofs ensure unlinkability
Verifier Trust Requirement	Verifier must securely store user data	No trust required beyond correct verification
Replay Resistance	Credentials can be reused or replayed	Nonce-bound proofs prevent replay attacks
Collusion Resistance	Data can be merged across verifiers	No shared data available for correlation
Issuer Dependency	Often required during verification	One-time issuance; no online dependency
Data Minimization	Violates data minimization principles	Only necessary information is disclosed
Security Model	Based on organizational trust	Based on cryptographic hardness assumptions

In this process, the prover first sends a commitment T. The verifier then issues a random challenge c. Finally, the prover responds with a value s that depends on both the commitment randomness and the secret witness.

2) Proof of Knowledge of Discrete Logarithm: Consider the relation:

$$C = g^a,$$

where C is public and a is the secret witness.

The goal is to prove knowledge of a such that the above relation holds, without revealing a.

3) Protocol Steps:

a) Commitment:

$$r \leftarrow \mathbb{Z}_q, \quad T = g^r$$

b) Challenge:

$$c \leftarrow \mathbb{Z}_q$$

c) Response:

$$s = r + c \cdot a \mod q$$

4) Verification: The verifier checks:

$$g^s \stackrel{?}{=} T \cdot C^c$$

5) Correctness: If the prover follows the protocol honestly, the verification equation holds:

$$g^s = g^{r+ca} = g^r \cdot g^{ca} = T \cdot C^c.$$

Therefore, valid proofs are always accepted.

6) Non-Interactive Version: The interaction can be removed using the Fiat–Shamir transformation. In this setting, the challenge is computed as:

$$c = H(T, C, t, V, \text{nonce}).$$

The resulting non-interactive proof is:

$$\pi = (T, s).$$

7) Proof Size and Complexity: The proof consists of a constant number of group elements and scalars for a single relation. For multiple attributes or statements, the size and verification cost scale linearly with the number of proven relations.

- Proof size: $O(n)$
- Verification complexity: $O(n)$

8) Security Properties:

a) Completeness: An honest prover possessing the correct witness always produces a proof that is accepted by the verifier.

b) Special Soundness: Given two accepting transcripts:

$$(T, c, s), \quad (T, c', s')$$

with $c \neq c'$, the witness can be extracted as:

$$a = \frac{s - s'}{c - c'} \mod q.$$

This property ensures that a prover who can answer multiple challenges must know the underlying secret.

c) Honest-Verifier Zero-Knowledge: There exists a simulator that can generate valid-looking transcripts without knowledge of the witness. A simulated commitment can be computed as:

$$T = g^s \cdot C^{-c}.$$

This shows that the verifier learns nothing beyond the validity of the statement.

9) Extension to Pedersen Commitments: For commitments of the form:

$$C = g^a h^r,$$

the prover demonstrates knowledge of both a and r.

Commitment:

$$T = g^{r_1} h^{r_2}$$

Response:

$$s_1 = r_1 + ca, \quad s_2 = r_2 + cr$$

Verification:

$$g^{s_1} h^{s_2} \stackrel{?}{=} T \cdot C^c$$

This extension allows proofs over committed values without revealing the underlying attributes.

10) Relevance to This Work: This construction is used to:

- Prove knowledge of committed attributes
- Support predicate-based proofs over hidden values
- Maintain zero-knowledge guarantees throughout the verification process

The simplicity and efficiency of Sigma protocols make them well suited for the design of privacy-preserving credential systems.

L. Advanced Sigma Protocol Constructions

Sigma protocols support several composition techniques that allow the construction of proofs for complex statements while preserving zero-knowledge, soundness, and efficiency. These techniques are essential for building expressive predicate proofs in credential systems, where multiple conditions must often be verified simultaneously without revealing underlying data.

1) And proofs (conjunctive composition): Conjunctive composition allows a prover to demonstrate knowledge of multiple witnesses at the same time.

Statement:

$$C_1 = g^{a_1}, \quad C_2 = g^{a_2}$$

Goal:

Prove knowledge of (a_1, a_2)

Protocol:

- Commitment:

$$T_1 = g^{r_1}, \quad T_2 = g^{r_2}$$

- Challenge (Fiat–Shamir):

$$c = H(T_1, T_2, C_1, C_2)$$

- Response:

$$s_1 = r_1 + ca_1, \quad s_2 = r_2 + ca_2$$

Verification:

$$g^{s_1} \stackrel{?}{=} T_1 \cdot C_1^c, \quad g^{s_2} \stackrel{?}{=} T_2 \cdot C_2^c$$

This construction ensures that both relations hold simultaneously. The use of a single challenge binds the two proofs together and prevents a prover from responding to each statement independently under different challenges.

2) Or proofs (disjunctive composition): Disjunctive composition enables a prover to demonstrate knowledge of one witness among multiple statements without revealing which one is known.

Statement:

$$C_1 = g^{a_1} \quad \text{or} \quad C_2 = g^{a_2}$$

Goal:

Prove knowledge of either a_1 or a_2

Construction overview:

- A real proof is generated for the known witness
- A simulated transcript is generated for the unknown witness
- The overall challenge is split across both branches

Challenge splitting:

$$c = c_1 + c_2 \mod q$$

Consistency condition:

$$c = H(T_1, T_2, C_1, C_2)$$

The prover selects one challenge value at random for the simulated branch and computes the other as:

$$c_1 = c - c_2 \mod q$$

ensuring that:

$$c = c_1 + c_2 \mod q.$$

Verification:

$$g^{s_1} \stackrel{?}{=} T_1 \cdot C_1^{c_1}, \quad g^{s_2} \stackrel{?}{=} T_2 \cdot C_2^{c_2}$$

This construction ensures that the verifier cannot determine which branch corresponds to the real witness. The simulated branch is computationally indistinguishable from a real execution.

Security intuition:

- Zero-knowledge is preserved because simulated transcripts are indistinguishable from real ones
- Soundness follows from the difficulty of producing valid responses for both branches without knowing at least one witness

3) Vector (multi-attribute) proofs: Vector proofs extend Sigma protocols to handle multiple attributes simultaneously. This construction is directly applicable to Pedersen commitments.

Commitment:

$$C = g^a h^r$$

Goal:

$$\text{Prove knowledge of } (a, r)$$

Protocol:

- Commitment:

$$T = g^{r_1} h^{r_2}$$

- Challenge:

$$c = H(T, C)$$

- Response:

$$s_1 = r_1 + ca, \quad s_2 = r_2 + cr$$

Verification:

$$g^{s_1} h^{s_2} \stackrel{?}{=} T \cdot C^c$$

This construction allows proofs over committed values while preserving secrecy of both the attribute and the randomness. The use of a single challenge ensures that both components remain cryptographically linked.

4) Proof compression and aggregation: Multiple Sigma proofs can be combined to reduce communication and verification overhead.

Techniques:

- Reusing challenges across related statements
- Batching verification using multi-exponentiation
- Aggregating multiple transcripts into a single proof object

Benefits:

- Reduced proof size
- Lower communication cost
- Improved verification efficiency

These optimizations are particularly useful in systems where multiple attributes or predicates must be verified in a single interaction.

5) Relevance to the proposed framework: These constructions form the basis for expressive and efficient proof generation in the proposed system. In particular, they enable:

- Multi-attribute selective disclosure without revealing full credentials
- Flexible policy enforcement using conjunctive and disjunctive conditions
- Efficient proof generation suitable for real-world deployment
- Scalable verification even in systems with complex attribute structures

The combination of these techniques ensures that the system can support practical privacy-preserving authentication while maintaining strong cryptographic guarantees.

M. Random Oracle Model

The Random Oracle Model (ROM) is an idealized framework in which a hash function H is modeled as a truly random function that maps arbitrary-length inputs to uniformly random outputs.

Formally, H is treated as a public oracle that, on input x, returns a value $H(x)$ sampled uniformly at random from its output space. This behavior is subject to a consistency condition: for any input x, repeated queries always return the same output $H(x)$. For distinct inputs, the outputs are independent and unpredictable.

This abstraction captures the ideal behavior of cryptographic hash functions, such as unpredictability and collision resistance, in a simplified and mathematically tractable form. Although real hash functions are deterministic algorithms, modeling them as random oracles allows rigorous analysis of protocol security.

In this work, all hash functions are modeled as random oracles. This assumption is standard in the analysis of Fiat–Shamir transformations, where the oracle replaces the verifier's challenge in interactive Sigma protocols. As a result, the interaction between prover and verifier is removed, and the proof becomes non-interactive.

In particular, challenges used in zero-knowledge proofs are derived as:

$$c = H(T, C, V, \text{nonce}),$$

where T is the commitment, C is the statement component, V denotes the verifier context, and nonce ensures freshness of the proof. This construction binds the proof to a specific session and prevents replay attacks.

The security of the resulting non-interactive proof system relies on the assumption that the hash function behaves like a random oracle. Under this model, the proof inherits the soundness and zero-knowledge properties of the underlying Sigma protocol.

It is important to note that the Random Oracle Model is an idealization. In practice, cryptographic hash functions such as SHA-256 are used as concrete instantiations. While no real function can achieve true randomness, widely used hash functions are designed to approximate this behavior closely enough for practical security.

Despite its idealized nature, the Random Oracle Model is widely adopted in modern cryptographic protocol design due to its ability to provide strong and analyzable security guarantees for non-interactive constructions.

N. Fiat–Shamir Transformation and Non-Interactive ZKP

Non-interactive zero-knowledge proofs are obtained by removing the interaction between the prover and the verifier in Sigma protocols. This is achieved using the Fiat–Shamir transformation [13], which replaces the verifier's challenge with a deterministic value derived from a cryptographic hash function.

In an interactive Sigma protocol, the proof follows a three-step structure:

$$\text{Commit} \rightarrow \text{Challenge} \rightarrow \text{Response}.$$

The prover first sends a commitment T, the verifier responds with a random challenge c, and the prover returns a response s. The verifier then checks the validity of the relation using these values.

Such interactive protocols require both parties to be online and involve multiple rounds of communication. This requirement limits their applicability in distributed and asynchronous environments.

To eliminate interaction, the Fiat–Shamir transformation replaces the verifier-generated challenge with a hash-based value:

$$c = H(T, C, V, \text{nonce}),$$

where T is the ephemeral commitment, C is the main commitment, V represents the verifier context, and nonce ensures freshness of the proof.

The resulting proof becomes non-interactive and is represented as:

$$\pi = (T, s).$$

During verification, the verifier recomputes the challenge:

$$c = H(T, C, V, \text{nonce}),$$

and checks whether the response s satisfies the verification equation defined by the underlying Sigma protocol.

This transformation preserves the correctness of the original protocol. An honest prover always produces a valid proof, and any valid proof corresponds to a correct execution of the protocol.

The security of the transformed protocol relies on the Random Oracle Model, as defined in Section VI-M. Under this model, the hash function behaves as a random oracle, ensuring that the generated challenge is unpredictable and cannot be influenced by the prover.

The transformation preserves the key properties of Sigma protocols, including completeness, special soundness, and zero-knowledge. In particular, the zero-knowledge property holds in the Random Oracle Model, where simulated transcripts are indistinguishable from real executions.

The inclusion of the verifier context V and a session-specific nonce ensures that each proof instance is bound to a unique session. This prevents reuse of proofs across different contexts. Even if the same statement is proven multiple times, the resulting proofs differ due to fresh randomness and distinct hash inputs.

As a result, replay attacks are prevented, and proofs remain unlinkable across sessions. The verifier cannot reuse a previously observed proof to gain advantage in a different interaction.

In this framework, all proofs are generated in non-interactive form. This enables asynchronous verification, where proofs can be generated and verified independently without requiring real-time communication between the prover and the verifier.

This design offers several practical advantages. The proof consists of a single message, communication overhead is reduced, and the system becomes suitable for deployment in distributed environments, including blockchain-based systems.

O. Security Guarantees

The resulting construction provides a set of strong security guarantees that follow from the properties of the underlying Sigma protocol and the Fiat–Shamir transformation.

Correctness ensures that valid statements are always accepted when the prover follows the protocol honestly. Soundness guarantees that no efficient adversary can produce a valid proof for a false statement.

The zero-knowledge property ensures that the proof reveals no information about the underlying secret values, such as committed attributes or randomness. As a result, sensitive information remains protected during verification.

Unlinkability is achieved through the use of fresh randomness and session-specific inputs to the hash function. Proofs generated for different sessions cannot be correlated, even if they correspond to the same underlying attribute.

Replay resistance is ensured by incorporating the verifier context and nonce into the challenge computation. This prevents reuse of proofs across sessions and guarantees that each proof is valid only for its intended context.

Together, these properties ensure that the system provides secure, privacy-preserving, and practical verification suitable for real-world deployment.

VII. Cryptographic Construction

This section describes the complete credential lifecycle, including commitment generation, credential issuance, scoped pseudonym derivation, predicate proof construction, and accumulator-based revocation verification. Each component is designed to operate independently while maintaining overall consistency, allowing modular analysis of security and correctness. The construction relies only on standard hardness assumptions such as the discrete logarithm assumption and the Strong RSA assumption.

A. System Parameters

Let p be a large prime, and let $G \subseteq \mathbb{Z}_p^*$ be a cyclic subgroup of prime order q generated by g, i.e., $G = \langle g \rangle$. All group operations are performed within G.

The generator g defines the group structure, and all exponentiations are computed modulo p. The prime order q ensures that the discrete logarithm problem in G is well-defined and computationally hard.

Let $h \in G$ be an independent generator derived using a domain-separated hash function. Independence of g and h means that the discrete logarithm relationship between them is unknown, which is necessary for the security of Pedersen commitments.

For revocation, let $N = p_1 p_2$ be an RSA modulus, where p_1 and p_2 are large primes and their factorization is unknown. Let $g_A \in \mathbb{Z}_N^*$ be a generator used as the base for the cryptographic accumulator.

These parameters are generated once during system initialization and remain fixed for all participants.

B. Attribute Commitment

For an attribute $a \in \mathbb{Z}_q$, the holder samples randomness $r \xleftarrow{\$} \mathbb{Z}_q$ and computes a Pedersen commitment [1]:

$$C = g^a h^r \mod p.$$

This commitment binds the value a while hiding it from any observer.

The Pedersen commitment scheme satisfies two important security properties. First, it is perfectly hiding, meaning that for any fixed commitment C, all possible values of a are equally likely from the perspective of an observer. This ensures that no information about the attribute is leaked.

Second, it is computationally binding under the discrete logarithm assumption, provided that the discrete logarithm relation between g and h is unknown. This means that it is infeasible to find two different pairs (a, r) and (a', r') such that:

$$g^a h^r = g^{a'} h^{r'}.$$

This property prevents the holder from changing the committed value after the commitment is created.

a) Notation.: The attribute is denoted as $a \in \mathbb{Z}_q$. Randomness used in commitments is denoted by $r \in \mathbb{Z}_q$. All derived values, such as δ, are defined as explicit functions of a, ensuring consistent notation throughout the construction.

The commitment C serves as the foundational object in the system. It is used during credential issuance, predicate proof construction, and verification, while preserving the privacy of the underlying attribute.

C. Credential Issuance

During the issuance phase, the issuer generates a verifiable credential by signing the holder's committed attribute without learning its underlying value.

Let $C = g^a h^r$ denote the Pedersen commitment to the attribute a, where $r \in \mathbb{Z}_q$ is the associated randomness. The commitment is transmitted to the issuer as the message to be signed.

To apply the signature scheme, the commitment is first converted into a suitable message representation. The issuer computes:

$$m = \text{Encode}(C),$$

where $\text{Encode}(\cdot)$ is a deterministic function that maps the group element C into a bit string or numerical representation compatible with the hash function used in the signature scheme.

The issuer then generates a signature using the Schnorr signature scheme defined in Section VI-G, with secret key x_I and corresponding public key $pk_I = g^{x_I}$.

The resulting credential is:

$$\sigma_{\text{cred}} = (T, s).$$

Correctness of the credential is verified by checking:

$$g^s \stackrel{?}{=} T \cdot pk_I^c,$$

where $c = H(T, m)$.

This verification equation ensures that the signature is consistent with both the issuer's public key and the encoded commitment.

This construction binds the committed attribute C to the issuer's signature, ensuring authenticity and integrity of the credential. Any modification to the commitment would invalidate the signature.

At the same time, due to the hiding property of the Pedersen commitment, the issuer learns no information about the underlying attribute a during the issuance process. The commitment reveals no partial information, and the signature is generated solely over its encoded form.

The issuance phase is executed once per credential and does not require the issuer to participate in subsequent verification or presentation phases. This property enables offline verification and reduces dependency on the issuer after credential creation.

As a result, the issued credential can be presented and verified independently, while preserving both authenticity and privacy.

D. Scoped Pseudonym Derivation

To prevent cross-domain linkage, the holder derives a scoped pseudonym for each verification context.

Let $pk_H = g^x$ denote the holder's public key. For a given scope S, the pseudonym is computed using the construction defined in Section VI-I as:

$$ID_S = pk_H^{H(S)}.$$

The resulting identifier ID_S is bound to the scope S, ensuring that the same holder produces different pseudonyms across distinct domains while remaining consistent within a fixed scope.

Since $pk_H \in G$ and $H(S) \in \mathbb{Z}_q$, the resulting pseudonym also lies in the group G, preserving compatibility with the underlying algebraic structure.

Under the random oracle model, pseudonyms generated for different scopes are computationally unlinkable. Linking two identifiers corresponding to distinct scopes would require recovering the secret exponent x, which reduces to solving the discrete logarithm problem in G.

E. Predicate Proof: Age Threshold Example

Range proofs [15] enable proving inequalities over committed values without revealing the underlying attribute values. In this work, a simplified construction is adopted by reducing the inequality check $a \geq t$ to proving knowledge of a derived witness $\delta = a - t$. The construction is based on sigma protocols [20], which are transformed into non-interactive proofs using the Fiat–Shamir heuristic [13].

To prove $a \geq t$, define:

$$\delta = a - t.$$

a) Remark.: It is assumed that attributes are encoded such that invalid cases (corresponding to $a < t$) are rejected at the application layer. Extending this construction to full range proofs remains future work.

Compute the adjusted commitment:

$$C' = C \cdot g^{-t}.$$

This transformation ensures that:

$$C' = g^a h^r \cdot g^{-t} = g^{a-t} h^r = g^\delta h^r,$$

and therefore $C' \in G$.

The holder then proves knowledge of (δ, r) such that:

$$C' = g^\delta h^r.$$

Using a sigma protocol, the prover samples random values $u_1, u_2 \in \mathbb{Z}_q$ and computes:

$$T = g^{u_1} h^{u_2}.$$

The challenge is derived using the Fiat–Shamir transformation:

$$c = H(T, C', V, \text{nonce}).$$

The responses are computed as:

$$s_1 = u_1 + c\delta, \quad s_2 = u_2 + cr.$$

The verifier checks:

$$g^{s_1} h^{s_2} = T \cdot (C')^c.$$

This verification equation ensures correctness of the relation while preserving secrecy of the underlying values.

The proof reveals only the validity of the predicate $a \geq t$ and no additional information about the attribute a, the derived value δ, or the randomness r.

F. Accumulator-Based Revocation

Revocation is handled using the cryptographic accumulator defined in Section VI-H, enabling efficient verification without revealing credential identities.

Each credential is associated with a unique prime representative. Revoked credentials are inserted into the accumulator set R, and the corresponding accumulator value A is maintained by the revocation authority.

For a non-revoked credential represented by a prime e, the holder computes a non-membership witness w using the construction described in Section VI-H. This witness proves that the credential does not belong to the revoked set R without disclosing any additional information.

During verification, the verifier checks the validity of the witness (w, e) against the current accumulator value A according to the accumulator verification equation defined in Section VI-H, thereby ensuring that $e \notin R$.

Acceptance implies that the credential is not revoked.

This approach ensures that revocation status can be verified efficiently while preserving privacy. The verification process remains independent of the size of the revoked set $|R|$, making the system scalable for large deployments.

G. Presentation Composition

A complete presentation consists of the following components:

- Commitment C
- Issuer signature σ
- Scoped pseudonym ID_S
- Predicate proof transcript
- Non-membership witness

All zero-knowledge components are bound to a verifier-generated nonce and context V through the Fiat–Shamir transformation. This ensures that all elements of the presentation correspond to the same session and are jointly bound to a common challenge derivation context.

p The verifier checks each component independently and accepts the presentation only if all verification conditions are

satisfied. This includes signature validity, correctness of zero-knowledge proofs, consistency of the pseudonym, and validity of the revocation witness.

H. Security Composition

The overall security of the construction follows from the security of the underlying cryptographic primitives and their correct composition.

- Hiding and binding of Pedersen commitments
- Unforgeability of Schnorr signatures
- Zero-knowledge and special soundness of sigma protocols
- Strong RSA security of the accumulator
- Domain separation under the random oracle model

Each primitive is used in isolation without circular dependency, ensuring modular security reasoning. The correctness and security of the full system follow from the composition of these primitives, where each component preserves its guarantees under standard assumptions.

In particular, zero-knowledge proofs protect attribute privacy, signatures ensure authenticity, and the accumulator enforces revocation without revealing credential identities. The use of domain separation and nonce-based challenges ensures that all proofs are bound to a specific session, preventing replay and cross-context linkage.

This modular design enables clear reasoning about security while maintaining efficiency and scalability in practical deployments.

VIII. Security Analysis and Proof Sketch

This section provides a high-level proof sketch establishing that the security objectives defined in Section IV-C reduce to standard cryptographic hardness assumptions. The analysis follows a modular approach, where each primitive is analyzed independently and then composed to argue overall system security.

A. Security Experiments and Definitions

To formally capture the security guarantees of the proposed framework, a set of game-based security experiments is defined between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$, modeled as a probabilistic polynomial-time algorithm.

Definition 1 (Attribute Privacy).

Experiment $\mathsf{Exp}^{\mathcal{A}}_{\textbf{privacy}}(\lambda)$**:**

1) The adversary outputs two attributes $(a_0, a_1) \in \mathbb{Z}_q$.
2) The challenger selects a random bit $b \leftarrow \{0, 1\}$.
3) The challenger computes:

$$C^* = g^{a_b} h^r$$

for random $r \leftarrow \mathbb{Z}_q$.

4) The adversary outputs a guess b'.

Advantage:

$$\mathrm{Adv}^{\mathcal{A}}_{\text{privacy}} = \left| \Pr[b' = b] - \frac{1}{2} \right|$$

Definition 2 (Scoped Unlinkability).

Experiment $\mathsf{Exp}^{\mathcal{A}}_{\textbf{unlink}}(\lambda)$**:**

1) The challenger samples $x \leftarrow \mathbb{Z}_q$.
2) The adversary outputs scopes (S_1, S_2).
3) The challenger selects $b \leftarrow \{0, 1\}$.
4) If $b = 0$, compute:

$$ID_1 = g^{xH(S_1)}, \quad ID_2 = g^{xH(S_2)}$$

Else, sample independent x_1, x_2:

$$ID_1 = g^{x_1 H(S_1)}, \quad ID_2 = g^{x_2 H(S_2)}$$

5) The adversary outputs b'.

Advantage:

$$\mathrm{Adv}^{\mathcal{A}}_{\text{unlink}} = \left| \Pr[b' = b] - \frac{1}{2} \right|$$

Definition 3 (Predicate Soundness).

Experiment $\mathsf{Exp}^{\mathcal{A}}_{\textbf{sound}}(\lambda)$**:**

1) The adversary outputs a statement $stmt \notin \mathcal{L}$.
2) The adversary outputs a proof π.
3) The challenger accepts if:

$$\text{Verify}(stmt, \pi) = 1.$$

Advantage:

$$\mathrm{Adv}^{\mathcal{A}}_{\text{sound}} = \Pr[\text{Verify}(stmt, \pi) = 1 \wedge stmt \notin \mathcal{L}]$$

Definition 4 (Revocation Soundness).

Experiment $\mathsf{Exp}^{\mathcal{A}}_{\textbf{rev}}(\lambda)$**:**

1) The challenger generates an accumulator value A over revoked set R.
2) The adversary outputs $e \in R$ and a witness w.
3) The challenger accepts if the accumulator verification relation holds for (w, e) with respect to A, as defined in Section VI-H.

Advantage:

$$\mathrm{Adv}^{\mathcal{A}}_{\text{rev}} = \Pr[\text{valid non-membership proof for } e \in R]$$

Definition 5 (Collusion Resistance).

Experiment $\mathsf{Exp}^{\mathcal{A}}_{\textbf{coll}}(\lambda)$**:**

1) The challenger issues credentials to multiple users.
2) The adversary obtains all corresponding witnesses.
3) The adversary outputs a proof π for a statement not satisfied by any individual credential.
4) The challenger accepts if:

$$\text{Verify}(stmt, \pi) = 1.$$

Advantage:

$$\mathrm{Adv}^{\mathcal{A}}_{\mathrm{coll}} = \Pr[\text{successful collusion attack}]$$

Security Requirement.

The system is secure if all above advantages are negligible in the security parameter λ.

B. Proof Sketch

The security of the construction follows from reductions to standard cryptographic hardness assumptions.

Attribute privacy follows from the perfect hiding property of the Pedersen commitment scheme. Any adversary capable of distinguishing between commitments can be used to break the hiding property, which contradicts its information-theoretic security.

Scoped unlinkability reduces to the hardness of the discrete logarithm problem in G. Distinguishing whether two pseudonyms are derived from the same secret requires recovering the exponent x, which is computationally infeasible under this assumption.

Predicate soundness follows from the special soundness property of the underlying sigma protocols. An adversary producing a valid proof for a false statement can be used to extract a witness, thereby solving the discrete logarithm problem.

Revocation soundness reduces to the Strong RSA assumption. Constructing a valid non-membership witness for an element in the revoked set implies solving an instance of the Strong RSA problem, which is assumed to be computationally infeasible.

Collusion resistance follows from the binding property of commitments and the unforgeability of Schnorr signatures. Combining multiple credentials does not allow construction of a valid proof for a statement that none of the individual credentials satisfy.

Since each primitive is secure under its respective assumption and the construction avoids circular dependencies, the overall system preserves these guarantees under composition.

C. Attribute Privacy

The commitment scheme is instantiated using Pedersen commitments:

$$C = g^a h^r.$$

Since h is chosen independently of g, and the randomness r is sampled uniformly from $\mathbb{Z}_q$, the commitment is perfectly hiding. This means that for any two attributes $a_0, a_1 \in \mathbb{Z}_q$, the corresponding commitment distributions are statistically identical. As a result, no information about the attribute a is revealed from the commitment C.

In addition to the commitment, the system uses predicate proofs constructed as sigma protocols [20]. These protocols satisfy special soundness and honest-verifier zero-knowledge. By applying the Fiat–Shamir transformation [13] in the random oracle model [19], the interactive protocol is converted into a non-interactive proof while preserving its zero-knowledge property.

Consequently, the proof transcripts reveal only the validity of the statement being proven and do not leak any additional information about the underlying attribute. Any adversary capable of distinguishing between attributes from the commitment and proof transcripts would contradict either the information-theoretic hiding property of the commitment scheme or the zero-knowledge property of the proof system.

D. Reduction to Discrete Logarithm (Attribute Privacy)

Theorem. The Pedersen commitment $C = g^a h^r$ is perfectly hiding and computationally binding. Under the discrete logarithm assumption in G, the binding property holds, and the attribute privacy experiment is secure.

Proof Sketch. The hiding property of the Pedersen commitment is information-theoretic and does not rely on computational assumptions. For any $a_0, a_1 \in \mathbb{Z}_q$, the distributions:

$$g^{a_0} h^r \quad \text{and} \quad g^{a_1} h^{r'}$$

are identical due to the uniform randomness of r and r'. Therefore, no adversary can distinguish between commitments to different attributes.

The binding property relies on the discrete logarithm assumption. Suppose an adversary finds two distinct openings:

$$g^a h^r = g^{a'} h^{r'}$$

with $(a, r) \neq (a', r')$. Then:

$$g^{a-a'} = h^{r'-r}.$$

This implies:

$$\log_g h = \frac{a - a'}{r' - r} \mod q,$$

which yields the discrete logarithm of h with respect to g, contradicting the hardness of the discrete logarithm problem.

Therefore, the commitment scheme is perfectly hiding and computationally binding, and the attribute privacy experiment is secure.

E. Formal Unlinkability Definition

We formalize unlinkability via a game-based definition capturing the inability of an adversary to determine whether two credential presentations originate from the same holder across distinct verification scopes.

a) Intuition.: Even if an adversary observes multiple protocol transcripts generated under different verifier identities, it should be computationally infeasible to determine whether they correspond to the same underlying secret key.

b) Experiment $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{unlink}}(\lambda)$*:* Let λ be the security parameter. The unlinkability experiment between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ proceeds as follows:

1) **Setup:** The challenger runs:

$$(pp) \leftarrow \text{Setup}(1^{\lambda})$$

and generates two independent holder key pairs:

$$(pk_0, x_0), \quad (pk_1, x_1)$$

2) **Adversarial Query Phase:** The adversary may adaptively request polynomially many auxiliary transcripts from $\mathcal{C}$ under arbitrary scopes of its choice. These transcripts are generated honestly using either key pair.
3) **Challenge:** The adversary submits two distinct scopes:

$$S_1, S_2$$

The challenger samples a random bit:

$$b \xleftarrow{\$} \{0, 1\}$$

- If $b = 0$: Both transcripts are generated using the same secret key x_0:

$$\sigma_1 \leftarrow \text{Present}(x_0, S_1), \quad \sigma_2 \leftarrow \text{Present}(x_0, S_2)$$

- If $b = 1$: Transcripts are generated using independent keys:

$$\sigma_1 \leftarrow \text{Present}(x_0, S_1), \quad \sigma_2 \leftarrow \text{Present}(x_1, S_2)$$

The adversary receives (σ_1, σ_2).
4) **Guess:** The adversary outputs a bit b' indicating whether it believes the transcripts originate from the same holder.

c) Advantage.: The advantage of the adversary is defined as:

$$\text{Adv}_{\mathcal{A}}^{\mathsf{unlink}}(\lambda) = \left| \Pr[b' = b] - \frac{1}{2} \right|$$

d) Definition (Unlinkability).: The credential system achieves unlinkability if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the advantage $\text{Adv}_{\mathcal{A}}^{\mathsf{unlink}}(\lambda)$ is negligible in λ.

e) Theorem.: Under the discrete logarithm assumption in the group G, the proposed scoped pseudonym construction achieves unlinkability in the random oracle model.

f) Proof Sketch.: Assume that there exists an adversary $\mathcal{A}$ that achieves non-negligible advantage in the unlinkability experiment. A reduction $\mathcal{B}$ is constructed that uses $\mathcal{A}$ to solve the discrete logarithm problem.

Given a discrete logarithm challenge (g, g^x), the reduction sets:

$$pk_H = g^x$$

and simulates the environment for $\mathcal{A}$.

For any scope S, pseudonyms are generated as:

$$ID_S = pk_H^{H(S)} = g^{x \cdot H(S)}.$$

Since the hash function is modeled as a random oracle, the values $H(S_1)$ and $H(S_2)$ are independent and uniformly distributed in $\mathbb{Z}_q$.

In the real case, both pseudonyms are of the form:

$$g^{xH(S_1)}, \quad g^{xH(S_2)}.$$

In the independent case, they are distributed as:

$$g^{x_1 H(S_1)}, \quad g^{x_2 H(S_2)},$$

for independent exponents x_1, x_2.

If the adversary can distinguish these two distributions with non-negligible advantage, then it distinguishes whether two group elements share a common hidden exponent. This implies the ability to extract information about x or test algebraic relations involving x, which can be used to construct an algorithm that solves the discrete logarithm problem.

This contradicts the discrete logarithm assumption in G. Therefore, the adversary's advantage must be negligible.

Hence, the scoped pseudonym construction satisfies unlinkability.

F. Scoped Unlinkability

Scoped pseudonyms are defined as:

$$ID_S = pk^{H(S)} = g^{xH(S)}.$$

a) Definition (Scoped Unlinkability).: Let $\mathcal{A}$ be a probabilistic polynomial-time adversary that is given two transcripts corresponding to scopes S_1 and S_2.

The adversary outputs a bit indicating whether the transcripts belong to the same holder.

The advantage of the adversary is defined as:

$$Adv_{\mathcal{A}} = \left| \Pr[\mathcal{A} \text{ outputs correct guess}] - \frac{1}{2} \right|.$$

The system is unlinkable if $Adv_{\mathcal{A}}$ is negligible.

We formalize this guarantee as follows.

Theorem. Under the discrete logarithm assumption in G, the scoped pseudonym construction achieves unlinkability in the random oracle model.

Proof Sketch. Assume that there exists an adversary $\mathcal{A}$ that can distinguish whether two identifiers ID_{S_1}, ID_{S_2} are generated by the same holder with non-negligible advantage.

Each identifier is computed as:

$$ID_{S_i} = g^{xH(S_i)}, \quad i \in \{1, 2\}.$$

Since the hash function H is modeled as a random oracle, the values $H(S_1)$ and $H(S_2)$ are independent and uniformly distributed in $\mathbb{Z}_q$.

In the real case, both identifiers are generated using the same secret exponent x, resulting in:

$$g^{xH(S_1)}, \quad g^{xH(S_2)}.$$

In the independent case, the identifiers are distributed as:

$$g^{x_1 H(S_1)}, \quad g^{x_2 H(S_2)},$$

for independently chosen exponents $x_1, x_2 \in \mathbb{Z}_q$.

If the adversary can distinguish between these two distributions with non-negligible advantage, then it can distinguish whether two group elements share a common hidden exponent or are independently generated.

This ability implies a non-negligible advantage in solving a decisional problem related to the discrete logarithm problem. Such an advantage can be used to construct an algorithm that violates the hardness of the discrete logarithm assumption in G.

This leads to a contradiction. Therefore, the adversary's advantage must be negligible.

Hence, the scoped pseudonym construction satisfies unlinkability.

G. Reduction to Discrete Logarithm (Unlinkability)

Theorem. The scoped pseudonym construction $ID_S = g^{xH(S)}$ is unlinkable under the discrete logarithm assumption in the random oracle model.

Proof Sketch. Assume that an adversary $\mathcal{A}$ wins the unlinkability experiment with non-negligible advantage. A reduction $\mathcal{B}$ is constructed that uses $\mathcal{A}$ to distinguish structured exponentiation from independent randomness.

Given a discrete logarithm instance (g, g^x), the reduction sets $pk_H = g^x$ and simulates the environment for $\mathcal{A}$, including programming the random oracle H.

For two scopes S_1, S_2, the pseudonyms are computed as:

$$ID_1 = g^{xH(S_1)}, \quad ID_2 = g^{xH(S_2)}.$$

Since H is modeled as a random oracle, the values $H(S_1)$ and $H(S_2)$ are independent and uniformly distributed in $\mathbb{Z}_q$.

In the real case, both identifiers are generated using the same exponent x, while in the independent case they are distributed as:

$$g^{x_1 H(S_1)}, \quad g^{x_2 H(S_2)},$$

for independently chosen $x_1, x_2 \in \mathbb{Z}_q$.

If $\mathcal{A}$ can distinguish between these two distributions with non-negligible advantage, then it can distinguish whether two group elements share a common hidden exponent or are independently generated.

This implies a non-negligible advantage in solving a decisional problem related to the discrete logarithm problem, which contradicts the assumed hardness of the discrete logarithm problem in G.

Therefore, unlinkability holds.

H. Predicate Soundness

The sigma protocol satisfies special soundness: from two accepting transcripts with distinct challenges, one can extract the witness (δ, r).

Therefore, if a malicious holder convinces the verifier of a false predicate, an extractor can recover a valid witness (δ, r) such that the underlying relation holds. This implies that the statement belongs to the language $\mathcal{L}$, which contradicts the assumption that the statement is false.

Hence, no adversary can produce a valid proof for a false statement with non-negligible probability, and predicate soundness holds.

I. Reduction via Special Soundness (Predicate Proof)

Theorem. The sigma protocol used for predicate proofs is sound, and its security relies on the hardness of the underlying relation defined over the group G.

Proof Sketch. Sigma protocols satisfy special soundness: given two accepting transcripts:

$$(T, c, s), \quad (T, c', s')$$

with $c \neq c'$, the witness can be extracted as:

$$a = \frac{s - s'}{c - c'} \mod q.$$

This extraction shows that any prover capable of producing valid responses to multiple challenges must possess a valid witness for the statement.

If a malicious prover produces a valid proof for a false statement, then the extractor derives a witness that satisfies the underlying relation. This implies that the statement belongs to the language $\mathcal{L}$, which contradicts the assumption that the statement is false.

Therefore, no adversary can produce a valid proof for a false statement with non-negligible probability, and soundness holds.

The hardness of computing valid witnesses without knowledge of the secret relies on standard assumptions such as the discrete logarithm assumption in G.

J. Signature Unforgeability

Credential issuance relies on Schnorr signatures. Under the discrete logarithm assumption in the random oracle model, Schnorr signatures are existentially unforgeable under chosen message attacks (EUF-CMA). Hence, no adversary can forge credentials for unissued commitments without violating the unforgeability of the signature scheme.

K. Reduction to Discrete Logarithm (Signature Unforgeability)

Theorem. The Schnorr signature scheme used for credential issuance is existentially unforgeable under chosen message attacks (EUF-CMA) in the random oracle model.

Proof Sketch. Assume that an adversary $\mathcal{A}$ produces a valid forgery (T, s) for a message m that was not previously queried.

The verification equation is:

$$g^s = T \cdot pk_I^c, \quad \text{where } c = H(T, m).$$

Using the Forking Lemma, one can rewind $\mathcal{A}$ to obtain two valid signatures:

$$(T, s), \quad (T, s')$$

corresponding to the same commitment T but different challenges $c \neq c'$.

From these transcripts:

$$s - s' = (c - c')x_I \Rightarrow x_I = \frac{s - s'}{c - c'} \mod q.$$

Thus, the adversary can be used to extract the secret signing key x_I. This extraction can be used to construct an algorithm that violates the hardness of the discrete logarithm problem in G.

Therefore, Schnorr signatures are unforgeable under the discrete logarithm assumption.

L. Revocation Soundness

The accumulator is instantiated under the Strong RSA assumption.

Suppose a revoked holder with prime representative $e \in R$ produces a valid non-membership witness (w, v). Then:

$$w^e \cdot A^v \equiv g_A \mod N.$$

This implies that the adversary constructs values that satisfy the accumulator relation for an element that belongs to the revoked set, which should be infeasible.

Rewriting the equation:

$$w^e \equiv g_A \cdot A^{-v} \mod N,$$

which shows that the adversary computes an e-th root of a value derived from the accumulator.

Such an ability contradicts the Strong RSA assumption, which states that given a random element in $\mathbb{Z}_N^*$, it is computationally infeasible to compute its non-trivial root without knowledge of the factorization of N.

Thus, revoked credentials cannot generate valid non-membership proofs.

M. Reduction to Strong RSA (Revocation Soundness)

Theorem. The accumulator-based revocation scheme is sound under the Strong RSA assumption.

Proof Sketch. Assume that an adversary $\mathcal{A}$ produces a valid non-membership witness (w, v) for an element $e \in R$.

Then:

$$w^e \cdot A^v \equiv g_A \mod N.$$

Rewriting:

$$w^e \equiv g_A \cdot A^{-v} \mod N.$$

This shows that the adversary computes an element w whose e-th power equals a value derived from the accumulator.

Such an ability implies computing a non-trivial e-th root modulo N. This contradicts the Strong RSA assumption, which states that for a given element in $\mathbb{Z}_N^*$, it is computationally infeasible to find any pair $(w, e > 1)$ such that:

$$w^e = u \mod N.$$

Therefore, forging non-membership witnesses for revoked elements is infeasible.

N. Replay Resistance

Theorem. The system is secure against replay attacks under the random oracle model.

Proof Sketch. All proofs are bound to a verifier-generated nonce and context:

$$c = H(T, C, V, \text{nonce}).$$

The nonce is required to be fresh and unique for each verification session. This ensures that each proof instance is tied to a specific interaction.

If an adversary reuses a previously valid transcript in a different session with a new nonce, the verifier recomputes the challenge using the new nonce, resulting in a different value of c. Consequently, the verification equation is no longer satisfied, and the proof is rejected.

If the adversary attempts to reuse the entire transcript including the nonce, then replay is only possible if the verifier accepts reused nonces. Enforcing freshness of the nonce prevents such attacks.

An adversary that successfully forges a valid proof for a new nonce must either:

- Find a collision in the hash function, or
- Predict or influence the output of the random oracle

Both events occur with negligible probability under standard cryptographic assumptions. Therefore, replay resistance holds.

O. Compositional Security

The construction is composed of standard cryptographic primitives, each providing a specific security guarantee:

- Pedersen commitments ensure information-theoretic hiding and computational binding.
- Schnorr signatures ensure credential authenticity and unforgeability.
- Sigma protocols ensure correctness and soundness of predicate proofs.
- Scoped exponentiation ensures unlinkability across verification contexts.

- Cryptographic accumulators ensure sound revocation.

Each component is instantiated independently and relies on well-established hardness assumptions. The construction avoids circular dependencies between primitives, and interactions between components are limited to clearly defined interfaces.

All proof transcripts are bound to session-specific nonces and verifier context through the Fiat–Shamir transformation. This ensures that different protocol executions remain independent and prevents cross-session interference.

Under these conditions, the overall system security follows from standard composition arguments. Since each primitive preserves its security properties in the presence of others, the combined construction maintains correctness, privacy, unlinkability, and soundness under the assumed hardness assumptions, without requiring additional trusted setup or specialized algebraic structures.

IX. Performance Evaluation

We evaluate the computational efficiency, proof size, and scalability characteristics of the proposed framework under realistic deployment parameters.

The evaluation focuses on the cost of core operations involved in credential issuance, proof generation, verification, and revocation checking. All results are presented in terms of standard group operations such as exponentiations, multiplications, and hash evaluations, which are commonly used to estimate performance in cryptographic systems.

A. Computational Cost

The computational cost of the system is determined by the number of algebraic operations required in each phase.

Commitment Generation. The computation of the Pedersen commitment:

$$C = g^a h^r$$

requires two exponentiations in the group G. This operation is efficient and can be performed on resource-constrained devices.

Credential Issuance. The Schnorr signature generation requires:

- One exponentiation to compute $T = g^k$
- One hash evaluation to derive $c = H(T, m)$
- One modular multiplication to compute the response s

Thus, issuance is lightweight and suitable for practical deployment.

Proof Generation. For predicate proofs, the prover computes:

- Two exponentiations for commitment $T = g^{u_1} h^{u_2}$
- One hash evaluation for challenge computation
- Two scalar operations for response generation

This cost grows linearly with the number of predicates.

Verification. Verification requires:

- Two exponentiations for checking the sigma protocol equation
- One hash evaluation

The cost remains constant per predicate and is efficient for real-time verification.

Revocation Checking. Accumulator verification involves modular exponentiation in $\mathbb{Z}_N^*$. The number of operations is constant and independent of the size of the revoked set $|R|$, ensuring scalability.

B. Proof Size

The size of the proof is determined by the number of group elements and scalars transmitted during presentation.

A complete proof includes:

- One commitment C
- One signature $\sigma = (T, s)$
- One scoped pseudonym ID_S
- Sigma protocol transcript (T, s_1, s_2)
- Non-membership witness components

Each element in G requires approximately $\log_2 p$ bits, while scalars require $\log_2 q$ bits. Therefore, the total proof size grows linearly with the number of predicates but remains compact for typical parameter sizes.

C. Communication Overhead

The protocol requires a single message from the prover to the verifier due to the use of non-interactive zero-knowledge proofs. This significantly reduces communication overhead compared to interactive protocols.

The absence of multiple rounds makes the system suitable for asynchronous and distributed environments.

D. Scalability Analysis

The system exhibits favorable scalability properties:

- Predicate proofs scale linearly with the number of attributes.
- Revocation verification is constant time with respect to $|R|$.
- Verification operations are independent across components and can be parallelized.

These properties ensure that the system remains efficient even as the number of users and credentials increases.

E. Asymptotic Complexity

The computational complexity of the system can be summarized as follows:

- Commitment generation: $O(1)$ group exponentiations

- Credential issuance: $O(1)$ exponentiations and hash evaluations
- Predicate proof generation: $O(n)$ exponentiations, where n is the number of predicates
- Verification: $O(n)$ exponentiations
- Revocation checking: $O(1)$ modular exponentiations

Here, n denotes the number of predicates proved in a single presentation. The constant-time revocation verification ensures scalability even for large revocation sets.

F. Practical Considerations

In practice, the efficiency of the system depends on the choice of parameters such as group size and modulus length. Standard choices, such as 256-bit elliptic curve groups or 2048-bit RSA moduli, provide a balance between security and performance.

The use of the Fiat–Shamir transformation eliminates interaction, enabling deployment in environments where real-time communication is not guaranteed. Furthermore, the modular design allows independent optimization of each component without affecting overall correctness.

Overall, the construction achieves a practical balance between security and efficiency, making it suitable for real-world applications that require privacy-preserving credential verification.

G. Experimental Setup

All experiments were conducted on a standard desktop environment running Windows 10 with Python 3.10.8. The implementation uses:

- 2048-bit prime-order discrete logarithm group parameters,
- 2048-bit RSA modulus for accumulator construction,
- SHA-256 for all hash operations,
- Pure modular arithmetic without external cryptographic acceleration.

The chosen parameter sizes correspond to widely accepted security levels. In particular, a 2048-bit RSA modulus provides security under the Strong RSA assumption, while 2048-bit discrete logarithm group parameters ensure resistance against known attacks on the discrete logarithm problem. SHA-256 is used as a standard cryptographic hash function, modeled as a random oracle in the security analysis.

All cryptographic operations are implemented using native big-integer arithmetic provided by the Python runtime, without relying on optimized external libraries or hardware acceleration. As a result, the reported performance reflects a conservative estimate of practical efficiency.

All reported measurements are averaged over multiple independent runs to reduce variance caused by operating system scheduling, background processes, and interpreter overhead. Each measurement is obtained under identical conditions to ensure consistency across experiments.

This setup provides a realistic baseline for evaluating the computational performance of the proposed construction in typical deployment environments, while maintaining reproducibility and alignment with standard cryptographic assumptions.

H. Prover-Side Costs

This subsection evaluates the computational cost incurred by the prover during the generation of a complete credential presentation. The analysis includes all major components involved in the presentation phase.

The prover performs the following operations:

- Pedersen commitment generation,
- Predicate proof construction,
- Schnorr proof of secret knowledge,
- Non-membership witness generation,
- Full presentation assembly.

The reported timings represent average execution times measured in seconds for each component. The measurements were obtained under a standard implementation setting with fixed cryptographic parameters, and averaged over multiple independent runs to reduce variance.

Observed average timings:

- Commitment generation: ≈ 0.0015
- Predicate proof generation: ≈ 0.0028
- Schnorr proof generation: ≈ 0.0014
- Non-membership proof generation: ≈ 0.0070
- Total presentation generation: ≈ 0.045

The results indicate that commitment generation and Schnorr proof construction incur relatively low computational overhead, as they rely on a small number of group exponentiations in G. Predicate proof generation requires additional operations but remains efficient due to its linear structure.

The non-membership proof generation dominates the prover-side cost. This is expected, as it involves modular exponentiation in $\mathbb{Z}_N^*$, which is computationally more expensive than operations in prime-order groups. This behavior is consistent with known performance characteristics of accumulator-based constructions under the Strong RSA assumption.

The total presentation generation time reflects the combined cost of all components, including auxiliary operations such as encoding, hashing, and data formatting. The observed value remains within practical limits for real-world applications, including scenarios with constrained computational resources.

Overall, the results demonstrate that the prover-side computation is efficient and scalable. The dominant cost arises from revocation-related operations, while all other components contribute only marginal overhead.

TABLE III: Performance Summary Under 2048-bit Security Parameters

Component	Mean Time (s)
Commitment Generation	0.0015
Predicate Proof Generation	0.0028
Schnorr Proof Generation	0.0014
Non-Membership Proof Generation	0.0070
Full Session Verification (Mean)	0.0028
Verification Std. Dev.	0.00077
Total Presentation Size	$\approx$ 3.2 KB

I. Verifier-Side Costs

Verification time is decomposed into:

- Issuer signature verification,
- Predicate proof verification,
- Schnorr proof verification,
- Non-membership witness verification,
- End-to-end session verification.

Average verification times (in seconds):

- Signature verification: ≈ 0.0024
- Predicate verification: ≈ 0.0034
- Schnorr verification: ≈ 0.0028
- Non-membership verification: ≈ 0.0138

Full-session verification latency averaged over 20 runs yields:

$$\text{Mean} \approx 0.0028 \text{ s}, \quad \text{Std Dev} \approx 7.7 \times 10^{-4}.$$

The low standard deviation indicates stable verification cost independent of transcript variability.

J. Proof Size Analysis

The total presentation size is approximately 3.2 KB, composed of:

- Schnorr signature: ~ 440 bytes,
- Predicate proof: ~ 740 bytes,
- Schnorr knowledge proof: ~ 440 bytes,
- Non-membership witness: ~ 612 bytes.

The resulting presentation remains compact relative to pairing-based credential systems or SNARK-based constructions with trusted setup.

K. Revocation Scalability

To evaluate revocation scalability, we vary the revoked set size from 1 to 200 elements and measure average verification time.

Revoked Set Size	Mean Verification Time (s)
1	≈ 0.0024
10	≈ 0.0020
50	≈ 0.0020
100	≈ 0.0020
200	≈ 0.0020

Observed mean verification times remain statistically stable.

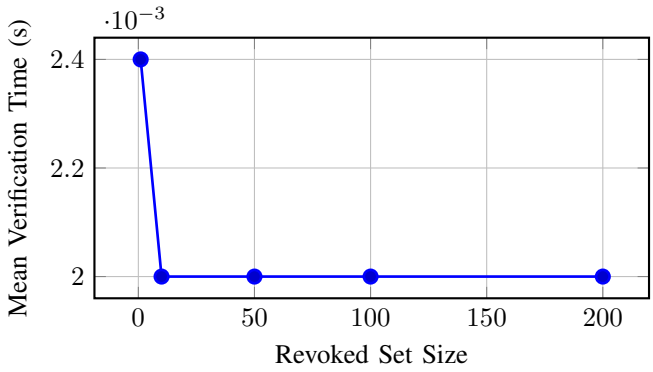

Fig. 4: Verification latency remains statistically stable as revocation set size increases, demonstrating constant-time accumulator verification.

Verification cost is empirically independent of $|R|$, confirming the constant-time verification property of RSA accumulator non-membership proofs.

L. Discussion

The results demonstrate that strong privacy guarantees, scoped unlinkability, and accumulator-based revocation can be achieved without sacrificing practical performance. End-to-end verification latency remains in the millisecond range under 2048-bit security parameters and without hardware acceleration.

The primary computational bottleneck arises from RSA exponentiation during non-membership verification, suggesting that future optimization may target multi-exponentiation techniques or batching strategies.

Overall, the framework achieves a practical balance between formal cryptographic security and deployable efficiency in multi-system identity environments.

X. Implementation Details

We implement the proposed zero-knowledge credential framework as a modular, end-to-end prototype in Python, emphasizing cryptographic correctness, reproducibility, and alignment with the formal construction described in Section IX. The implementation closely follows the mathematical specification of commitments, proofs, scoped pseudonyms, and accumulator-based revocation.

A. System Architecture

The implementation adopts a layered architecture that separates cryptographic primitives from protocol orchestration. This separation enables modular reasoning about security and simplifies future extensions.

The system consists of four primary components:

- **Issuer Module ($\mathcal{I}$):** Generates credentials by signing Pedersen commitments $C = g^a h^r$ using Schnorr signatures. This module is responsible for key generation and credential issuance.
- **Holder Module ($\mathcal{H}$):** Maintains secret attributes (a, r) and generates:
  - Pedersen commitments
  - Scoped pseudonyms $ID_S = pk_H^{H(S)}$
  - Zero-knowledge proofs $\pi = \text{ZKP}(a; c, V)$
  - Non-membership witnesses for revocation
- **Verifier Module ($\mathcal{V}$):** Validates credential presentations by:
  - Verifying Schnorr signatures
  - Checking zero-knowledge proofs
  - Ensuring nonce freshness
  - Performing accumulator-based revocation verification
- **Revocation Authority ($\mathcal{R}$):** Maintains the RSA accumulator over revoked credentials and provides parameters required for non-membership verification.

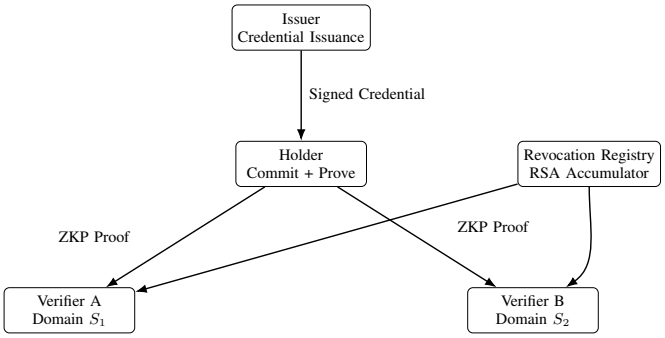

Fig. 5: Multi-system credential architecture with scoped unlinkability.

This modular design ensures clean separation of trust assumptions and allows independent optimization of each component.

B. Cryptographic Parameters

All experiments are conducted using standard 2048-bit security parameters:

- A cyclic group G of prime order q with generator g
- Independent generator $h \in G$ such that $\log_g h$ is unknown
- RSA modulus $N = p_1 p_2$ for accumulator construction, with unknown factorization
- Accumulator base $g_A \in \mathbb{Z}_N^*$
- Security parameter $\lambda = 2048$

The discrete logarithm assumption in G and the Strong RSA assumption over $\mathbb{Z}_N^*$ are assumed to hold.

C. Implementation of Core Primitives

a) Pedersen Commitment.: Attributes are encoded using:

$$C = g^a h^r$$

where $a \in \mathbb{Z}_q$ is the attribute and $r \in \mathbb{Z}_q$ is uniformly random.

b) Zero-Knowledge Proofs.: Sigma protocols are implemented in non-interactive form using the Fiat–Shamir transformation:

$$c = H(T, C, V, \text{nonce}), \quad \pi = (T, s)$$

All proofs are bound to verifier identity and nonce to ensure replay resistance.

c) Scoped Pseudonyms.: Each presentation derives:

$$ID_S = pk_H^{H(S)} = g^{x \cdot H(S)}$$

ensuring unlinkability across domains under the random oracle model.

d) RSA Accumulator.: Revocation is implemented using:

$$A = g_A^{\prod r_i} \mod N$$

with non-membership witnesses computed using Bézout coefficients.

D. Implementation Choices

- **Programming Language:** Python 3.10
- **Arithmetic:** Native arbitrary-precision integers (built-in big integer support)
- **Hash Function:** SHA-256, modeled as a random oracle
- **Randomness Source:** Cryptographically secure OS entropy (e.g., `os.urandom`)
- **Hash-to-Prime:** Iterative hashing followed by probabilistic primality testing (Miller–Rabin)

No external cryptographic libraries are used, ensuring transparency and direct control over all operations.

E. Performance Considerations

The implementation prioritizes correctness over low-level optimization. All group operations are implemented via modular exponentiation, and no precomputation or batching techniques are used.

The primary computational bottleneck arises from RSA exponentiation during non-membership witness generation and verification, consistent with theoretical expectations.

F. Reproducibility

The system supports reproducible experiments by:

- Fixing randomness seeds for deterministic execution
- Logging all protocol transcripts
- Isolating cryptographic operations into testable modules

This enables consistent benchmarking and validation of protocol correctness across multiple runs.

a) Code Availability.: To support reproducibility and independent verification, the implementation of the proposed system is available at:

https://github.com/sayan-hi/zkp-credential-framework.

The repository is currently under active maintenance, including code refactoring, documentation, and artifact packaging. The finalized version, along with complete documentation and reproducibility artifacts, will be fully available upon completion.

G. Security Alignment

The implementation strictly follows the formal model defined in Section V. All security properties—attribute privacy, unlinkability, soundness, and revocation correctness—are enforced at the protocol level without introducing additional assumptions beyond those stated.

H. Discussion

While the prototype is not optimized for production deployment, it demonstrates that the proposed framework is practically realizable using standard cryptographic primitives without reliance on pairing-based constructions or trusted setup.

The modular architecture also provides a clear pathway for future enhancements, including optimized exponentiation, batching techniques, and integration with real-world identity infrastructures.

XI. Applications

The proposed framework enables privacy-preserving attribute verification across multiple independent systems by combining Pedersen commitments, zero-knowledge proofs, scoped pseudonyms, and accumulator-based revocation. Unlike traditional identity systems that rely on explicit data disclosure, the framework supports verification through cryptographic proofs of predicates over hidden attributes.

A. Age Verification

A primary application is age verification without revealing the exact date of birth. Let a denote the user's age encoded as an integer. The holder computes a commitment:

$$C = g^a h^r$$

and proves the predicate $a \geq 18$ by constructing a zero-knowledge proof over the derived witness $\delta = a - 18$.

The verifier learns only the validity of the predicate and does not obtain any information about a.

Use Cases:

- Online content access control (e.g., age-restricted platforms)
- Regulatory compliance with minimal data disclosure

Security Advantage: Eliminates over-disclosure of sensitive personal data while maintaining verifiability.

B. Anonymous Authentication

The framework enables authentication without persistent identifiers by using scoped pseudonyms:

$$ID_S = pk_H^{H(S)} = g^{x \cdot H(S)}$$

For each verifier S, a unique pseudonym is generated, ensuring that authentication events across different domains cannot be linked.

Protocol Flow:

- Holder derives ID_S for verifier S
- Generates proof π of credential possession
- Verifier checks π without learning identity

Use Cases:

- Privacy-preserving login systems
- Anonymous subscription services

Security Advantage: Prevents cross-service tracking and profiling.

C. Decentralized Digital Identity

The framework naturally integrates with decentralized identifier (DID) and verifiable credential (VC) ecosystems by replacing static identifiers with cryptographic commitments and proofs.

Credentials are issued once and later presented independently:

$$\sigma = (\pi, ID_S, \text{metadata})$$

No issuer interaction is required during verification.

Use Cases:

- Self-sovereign identity (SSI)
- Cross-platform identity verification

System Advantage:

- No centralized identity registry
- No global identifiers
- Strong unlinkability across services

D. Healthcare Credential Verification

Sensitive attributes such as medical eligibility, certification, or compliance status can be verified without exposing underlying personal data.

For example, a patient can prove:

$$\text{eligible} = 1$$

without revealing diagnosis or medical history.

Protocol Mapping:

- Attribute a encodes eligibility
- Commitment C hides a
- Proof π verifies predicate

Use Cases:

- Insurance eligibility verification
- Clinical trial enrollment

Security Advantage: Ensures compliance with data minimization and medical privacy regulations.

E. Access Control in Multi-Domain Systems

In distributed environments with multiple independent verifiers, each enforcing distinct policies, the framework prevents cross-domain correlation.

Each verifier S_i receives:

$$ID_{S_i} = g^{x \cdot H(S_i)}$$

Since $H(S_i)$ is domain-separated, identifiers are unlinkable.

Threat Model:

- Verifier collusion
- Cross-platform profiling

Mitigation:

- Scoped pseudonyms eliminate global identifiers
- Zero-knowledge proofs prevent data leakage

Use Cases:

- Multi-service authentication platforms
- Federated access control systems

F. Revocation-Aware Applications

The integration of RSA accumulators enables privacy-preserving revocation.

A holder proves non-membership:

$$w^e \cdot A^v \equiv g_A \mod N$$

without revealing identity.

Use Cases:

- Revocable anonymous credentials
- Blacklist enforcement without deanonymization

Security Advantage: Revocation is enforced without introducing linkable identifiers.

G. Discussion

These applications collectively demonstrate a fundamental shift from identity-based verification to proof-based verification. Instead of transmitting raw credentials, users provide cryptographic evidence of correctness.

This approach ensures:

- Minimal disclosure of sensitive data
- Strong resistance to cross-domain tracking
- Cryptographic enforcement of privacy guarantees

The framework is particularly well-suited for modern distributed systems where privacy, scalability, and interoperability are essential requirements.

XII. Real-World Adoption

Zero-knowledge proof (ZKP) technologies are no longer purely theoretical and have been deployed in several large-scale real-world systems across finance, blockchain, identity, and governance.

- **Zcash — Shielded Transactions:** Zcash enables fully private transactions using zk-SNARKs, where sender, receiver, and amount remain hidden while maintaining public verifiability. https://z.cash/
- **Polygon zkEVM / StarkNet - Scalable Blockchain Execution:** Layer-2 scaling solutions such as Polygon zkEVM and StarkNet utilize zero-knowledge proofs to enable high-throughput, low-cost transactions while preserving Ethereum-level security. https://polygon.technology/polygon-zkevm
- **EU Digital Identity Wallet (EUDI):** The European Union is developing privacy-preserving digital identity frameworks that incorporate selective disclosure mechanisms based on zero-knowledge proofs. https://en.wikipedia.org/wiki/EU_Digital_Identity_Wallet
- **Google Wallet - Age Verification:** Google has introduced ZKP-based age verification systems that allow users to prove age eligibility without revealing their exact date of birth. [Link to Google Wallet]
- **MACI - Minimal Anti-Collusion Infrastructure:** MACI ensures private and tamper-resistant voting by allowing users to prove that votes are valid without revealing their content. It is used in decentralized governance systems such as Gitcoin. https://maci.pse.dev/
- **ZK Credit Systems (ING / JP Morgan):** Financial institutions are exploring ZKP-based systems to verify creditworthiness or regulatory compliance (e.g., AML/KYC) without exposing sensitive financial data. [Link to ZK Credit Systems]

A. Discussion

These deployments demonstrate that zero-knowledge proofs are transitioning from theoretical constructs to production-ready systems. They validate the practicality of privacy-

preserving verification and highlight the growing demand for cryptographic solutions that minimize data exposure while maintaining trust.

XIII. Limitations

Despite achieving strong privacy guarantees, scoped unlinkability, and efficient revocation, the proposed framework exhibits several limitations arising from underlying cryptographic assumptions, protocol design choices, and system-level constraints. These limitations highlight important avenues for future research and system refinement.

A. Lack of Post-Quantum Security

The construction relies fundamentally on the discrete logarithm assumption in a prime-order group G and the Strong RSA assumption over $\mathbb{Z}_N^*$. Specifically:

- Pedersen commitments $C = g^a h^r$ rely on DL hardness
- Scoped pseudonyms $ID_S = g^{xH(S)}$ rely on DL hardness
- Schnorr signatures depend on DL in the random oracle model
- RSA accumulators rely on the Strong RSA assumption

These assumptions are known to be vulnerable to quantum algorithms (e.g., Shor's algorithm), rendering the system insecure in a post-quantum setting.

Practical Impact: Long-term deployments requiring quantum resistance cannot directly use this construction.

Future Direction: Migration to lattice-based commitments, post-quantum signatures (e.g., Dilithium), and class group or vector commitment-based accumulators.

B. Simplified Predicate Proof Construction

Predicate verification in the current system is implemented via witness transformation:

$$\delta = a - t$$

and proving knowledge of δ such that:

$$C' = C \cdot g^{-t} = g^{\delta} h^r$$

While this approach is efficient, it does not constitute a full cryptographic range proof. In particular:

- It does not prevent invalid encodings of a
- It assumes application-layer enforcement of constraints
- It does not provide zero-knowledge guarantees for arbitrary inequality relations

Performance Impact: The system provides partial correctness for predicates but lacks formal completeness for general range proofs.

Future Direction: Integration of Bulletproof-style inner-product arguments or lattice-based range proofs for full cryptographic soundness.

C. Accumulator Computation Overhead

Revocation is implemented using RSA accumulators:

$$A = g_A^{\prod r_i} \mod N$$

Although verification of non-membership proofs:

$$w^e \cdot A^v \equiv g_A \mod N$$

is constant-time with respect to the size of the revoked set, the prover-side computation involves:

- Large modular exponentiations
- Computation of Bézout coefficients
- Hash-to-prime operations

Performance Impact: Resource-constrained devices (e.g., mobile or IoT) may experience latency during proof generation.

Future Direction: Optimization via batch exponentiation, precomputation strategies, or alternative accumulator constructions (e.g., bilinear or vector commitments).

D. Metadata and Side-Channel Leakage

The framework provides strong cryptographic privacy at the protocol level but does not address leakage through external channels:

- Network metadata (IP addresses, timing)
- Communication patterns
- Device fingerprints

Even when proofs π and identifiers ID_S are unlinkable, adversaries may correlate sessions using auxiliary information.

Privacy Impact: Privacy guarantees are limited to the cryptographic layer and do not extend to system-level anonymity.

Future Direction: Integration with anonymity networks (e.g., Tor, mixnets) and traffic obfuscation techniques.

E. Trusted Issuer Assumption

The system assumes that the issuer $\mathcal{I}$ behaves honestly during credential issuance:

- Correct generation of (pk_I, x_I)
- Proper signing of commitments
- No embedding of hidden identifiers in credentials

A malicious issuer could introduce covert linkability by embedding structured randomness or trapdoors into C or σ_{cred}.

System Impact: Trust in the issuer is a single point of failure for privacy guarantees.

Future Direction:

- Distributed or threshold issuance
- Verifiable issuance protocols
- Auditable credential generation

F. Limitations Beyond the Cryptographic Model

While the proposed framework provides strong guarantees of attribute privacy, unlinkability, and correctness at the cryptographic level, it is important to emphasize that zero-knowledge proofs (ZKPs) operate within a well-defined abstraction boundary. As such, several critical aspects of real-world systems remain outside the scope of cryptographic enforcement.

a) Verifier Behavior and Decision-Making.: Zero-knowledge proofs ensure that a verifier learns only the validity of a statement (e.g., $a \geq 18$) without accessing the underlying attribute a. However, once the proof π is accepted, the verifier retains full autonomy over how the result is used.

Limitation:

- The framework does not constrain or regulate verifier actions (e.g., denying access, profiling users, or applying biased policies)

Practical Impact: Correctness of verification does not imply fairness or accountability in decision-making.

Requirement: Policy enforcement, legal frameworks, and audit mechanisms must complement cryptographic guarantees.

b) Correctness of Issued Attributes.: The system assumes that the issuer $\mathcal{I}$ generates credentials honestly, i.e., that the committed attribute a embedded in:

$$C = g^a h^r$$

is valid and correctly certified.

Limitation:

- ZKPs prove knowledge of attributes, but do not guarantee that the attribute itself is truthful or correctly issued

Privacy Impact: A malicious or compromised issuer can issue incorrect or fraudulent credentials that still verify correctly.

Requirement: Trusted issuance processes, certification authorities, or verifiable credential registries are necessary.

c) Metadata Leakage and Network-Level Privacy.: Although scoped pseudonyms:

$$ID_S = g^{x \cdot H(S)}$$

and proofs π ensure unlinkability at the protocol level, external metadata may still leak information.

Limitation:

- Network identifiers (IP address)
- Timing correlations across sessions
- Traffic analysis and device fingerprinting

Security Impact: Adversaries may correlate sessions despite cryptographic unlinkability.

Requirement: Integration with anonymity networks (e.g., Tor, mixnets) and communication obfuscation techniques.

d) Separation of Cryptographic and System Guarantees.: The framework guarantees:

- Soundness of proofs π
- Hiding of attributes in C
- Unlinkability via ID_S
- Revocation correctness via accumulator A

However, it does *not* guarantee:

- Fair usage of verification outcomes
- Trustworthiness of issuers
- Protection against system-level or network-level leakage

Reason: Zero-knowledge proofs provide *cryptographic privacy*, but not *system-level privacy*.

e) Discussion.: These limitations highlight a fundamental principle: secure system design requires a holistic approach that combines cryptographic mechanisms with policy, governance, and network-layer protections. The proposed framework should therefore be viewed as a foundational cryptographic layer that must be integrated with complementary technologies to achieve end-to-end privacy and security in real-world deployments.

G. Limited Collusion Resistance Beyond Cryptography

While scoped pseudonyms and zero-knowledge proofs prevent direct linkage across verifiers, large-scale collusion combined with external datasets may still enable inference attacks.

Performance Impact: The system prevents cryptographic linkage but cannot fully eliminate statistical or behavioral correlation.

Future Direction: Differential privacy techniques and protocol-level noise injection for stronger resistance.

H. Discussion

The identified limitations do not undermine the core contribution of the framework, which demonstrates that selective disclosure, scoped unlinkability, and accumulator-based revocation can be achieved simultaneously without pairing-based constructions or trusted setup.

However, they highlight a fundamental insight: cryptographic privacy guarantees operate within a well-defined abstraction layer and must be complemented by system-level, network-level, and post-quantum considerations for real-world deployment.

Addressing these limitations represents a rich research direction spanning post-quantum cryptography, advanced zero-knowledge constructions, and privacy-preserving systems design.

XIV. Future Work

While the current framework demonstrates practical and cryptographically sound multi-system attribute verification, several research directions remain open and intellectually compelling.

A. Formal Simulation-Based Privacy Proof

The present work provides reduction-style proof sketches. A natural extension is to formalize the protocol under a simulation-based security model, such as Universal Composability (UC) or constructive cryptography frameworks.

In particular, constructing an ideal functionality for scoped credentials and proving indistinguishability between the real and ideal worlds would elevate the unlinkability claim from reduction-based reasoning to composable privacy guarantees under concurrent composition.

B. Adaptive and Collusion-Resilient Revocation

Although RSA accumulators provide constant-size witnesses and verification time independent of the revocation set size, the current design assumes a non-adaptive adversary.

Future work includes:

- Adaptive revocation where adversaries choose identities after seeing accumulator states,
- Verifier collusion models where subsets of verifiers attempt cross-system correlation,
- Forward-secure revocation witnesses supporting key rotation without reissuance.

Exploring vector commitments or bilinear accumulator constructions may offer stronger dynamic guarantees.

C. Post-Quantum Migration Path

Motivation. Modern cryptographic systems derive security from computational hardness assumptions such as integer factorization and the discrete logarithm problem (DLP). These assumptions underpin widely deployed schemes including RSA and elliptic curve cryptography (ECC). However, quantum algorithms-most notably Shor's algorithm [24]-solve these problems in polynomial time, rendering classical public-key cryptography insecure against quantum polynomial-time (QPT) adversaries.

This gives rise to the *harvest-now-decrypt-later* threat model [25], where adversaries store encrypted data today and decrypt it once large-scale quantum computers become available.

1) Quantum Threat Model: Let λ denote the security parameter. Classical cryptographic hardness relies on:

- **RSA:** Hardness of factoring $N = p_1 p_2$ [26]
- **ECC:** Hardness of DLP in cyclic group G [27]

Under quantum computation:

$$\text{Factoring}(N), \text{DLP}(G) \in \text{BQP} \tag{1}$$

as shown in [24]. Thus, security reduces from exponential to polynomial time.

Additionally, Grover's algorithm [28] reduces brute-force complexity:

$$O(2^{\lambda}) \to O(2^{\lambda/2}) \tag{2}$$

impacting symmetric and hash-based primitives.

To contextualize the impact of quantum adversaries on existing zero-knowledge constructions, we summarize representative protocols in Table IV.

2) Adversarial Model: We consider a quantum polynomial-time adversary $\mathcal{A}$ with access to:

- Quantum algorithms (Shor, Grover)
- Classical transcripts (C, π, σ)
- Long-term stored ciphertexts

Security requires indistinguishability and soundness to hold against $\mathcal{A}$.

3) System Dependency Analysis: The system relies on:

- Commitment: $C = g^a h^r$
- Proof: $\pi = \text{ZKProof}(a, r)$
- Presentation: σ

where a is the secret attribute and r is randomness.

The security of C depends on DLP hardness, which is broken under QPT adversaries.

4) Abstraction-Based Migration Framework: We define a cryptographic abstraction layer:

$$\mathcal{S} = (C, \mathcal{P}, \mathcal{V}) \tag{3}$$

where:

- C: commitment scheme
- $\mathcal{P}$: proof generator
- $\mathcal{V}$: verifier

This abstraction enables seamless replacement of primitives without altering protocol semantics [29].

5) Lattice-Based Commitment Transformation: We replace Pedersen commitments with LWE-based commitments [30]:

$$C = Aa + Br + e \pmod{q} \tag{4}$$

Security relies on the hardness of the Learning With Errors problem, reducible from worst-case lattice assumptions.

6) Post-Quantum Zero-Knowledge Proofs: The proof system:

$$\pi = \text{ZKProof}(a, r) \tag{5}$$

can be transformed using:

- Stern protocols [31]
- Fiat-Shamir heuristic [32]
- Lattice-based ZK constructions [33]

TABLE IV: Comparison of ZKP Frameworks and Their Quantum Security Properties

ZKP Protocol / Framework	Mathematical Primitive	Main Feature	Quantum Security
Σ-protocol (Schnorr)	Discrete logarithm in cyclic group G	Knowledge soundness	Broken by Shor
Fiat-Shamir NIZK	Cryptographic hash function (ROM/QROM)	Non-interactivity	Security reduced by Grover
Pedersen Commitment ZK	Group exponentiation (DLP)	Perfect hiding, computational binding	Broken by Shor
Groth16 zk-SNARK	Bilinear pairings	Constant-size proof	Broken by Shor
PLONK	Polynomial commitments + pairings	Universal setup	Broken by Shor
Bulletproofs	Elliptic curve inner product	Logarithmic proof size	Broken by Shor
zk-STARK	Hash functions + polynomials	Transparent setup	Secure (hash-based)
MPC-in-the-Head ZK	Symmetric crypto + secret sharing	Transparent ZK	Secure (hash-based)
Lattice-based ZK	LWE, SIS over $\mathbb{Z}_q^n$	Post-quantum soundness	Secure
Merkle Tree-based ZK	Collision-resistant hash trees	Efficient commitments	Secure (hash-based)

These preserve completeness, soundness, and zero-knowledge against quantum adversaries.

7) Post-Quantum Signature Integration: Credential issuance can be upgraded using lattice-based signatures such as CRYSTALS-Dilithium [34]:

$$\sigma_{\text{PQ}} = \text{Sign}_x^{\text{PQ}}(a) \tag{6}$$

These achieve quantum-resistant UF-CMA security.

8) Alternative PQ Directions: Alternative constructions include:

- Hash-based signatures (SPHINCS+) [35]
- Code-based cryptography (McEliece) [36]
- Multivariate schemes [37]
- Isogeny-based systems [38]

TABLE V: Post-Quantum Migration: Classical Components and Their Alternatives

Classical Component	Post-Quantum Alternative
Groth16	Lattice-based ZK
PLONK	Hash-based MPC-in-the-Head
Pedersen Commitment	SIS-based commitment
EC-DSA	ML-DSA
BLS Signatures	SLH-DSA

9) Mosca's Theorem: Let X be security lifetime, Y migration time, and Z quantum arrival time. Then:

$$X + Y \leq Z \tag{7}$$

must hold [25].

10) Migration Challenges:

- Increased key and proof sizes
- Computational overhead
- Verification latency
- Energy constraints
- Side-channel resistance

11) System Transformation:

$$(C, \pi, \sigma) \to (C_{\text{PQ}}, \pi_{\text{PQ}}, \sigma_{\text{PQ}}) \tag{8}$$

while preserving:

$$\text{Verify}(C, \pi) = \text{Verify}(C_{\text{PQ}}, \pi_{\text{PQ}}) \tag{9}$$

12) Forward Compatibility: The system ensures:

- Modularity
- Transparency
- Extensibility
- Efficiency

13) Discussion: This work does not implement PQ primitives but defines a formally grounded migration framework aligned with NIST PQC standardization [34].

D. Mobile and Hardware-Constrained Deployment

Current benchmarking is performed on a desktop environment. A rigorous mobile deployment study would evaluate:

- Energy consumption during exponentiation-heavy operations,
- Secure enclave storage of secret keys,
- Side-channel leakage under constrained environments.

Hardware-assisted modular exponentiation and multi-exponentiation batching strategies may significantly reduce prover latency.

E. Multi-Attribute Selective Disclosure

The prototype demonstrates predicate proofs for a single attribute. A generalized construction supporting:

range proofs, set membership proofs, inequality relations

without increasing transcript size linearly remains an open optimization problem.

Aggregated sigma protocols or Bulletproof-style inner-product arguments may provide logarithmic scaling in attribute dimension.

F. Decentralized Identity Interoperability

While a lightweight DID and VC format is integrated, full interoperability with W3C Verifiable Credential data models and JSON-LD canonicalization remains future work.

Bridging cryptographic minimalism with standards compliance without sacrificing unlinkability is a non-trivial systems design challenge.

G. End-to-End Formal Verification

Finally, a formally verified reference implementation using tools such as:

- Coq or EasyCrypt for protocol proofs,
- TLA+ for state machine modeling,
- Property-based testing for cryptographic invariants

would significantly strengthen confidence in correctness under adversarial edge cases.

In summary, this work establishes a secure and performant foundation for multi-system attribute verification. However, advancing toward composable privacy proofs, adaptive revocation, post-quantum resilience, and formally verified implementations presents a rich research agenda that extends well beyond the scope of this prototype.

XV. Conclusion

This work presents a cryptographically grounded framework for zero-knowledge based multi-system attribute verification that achieves selective disclosure, scoped unlinkability, and accumulator-based revocation without requiring trusted setup or pairing-based constructions.

The construction integrates Pedersen commitments, Schnorr signatures, sigma-protocol based predicate proofs, scoped pseudonyms, and Strong RSA-based accumulators into a unified and modular architecture. Each component is carefully selected and combined so that its security relies on well-established hardness assumptions, including the discrete logarithm assumption and the Strong RSA assumption. The resulting design avoids the use of global identifiers and prevents correlation across different verification domains, while preserving both credential authenticity and revocation correctness.

Experimental evaluation confirms that the proposed framework achieves these security guarantees with practical efficiency. End-to-end verification is completed within the millisecond range under 2048-bit security parameters. The overall proof size remains compact, approximately 3.2 KB, and the cost of revocation verification remains independent of the size of the revoked set. Stability across scalability experiments further demonstrates that the addition of privacy-preserving mechanisms does not introduce hidden computational overhead or performance degradation.

A key contribution of this work lies in its structural approach to privacy. Unlinkability is achieved by design through scoped pseudonym derivation and nonce-bound proof transcripts, rather than emerging as a secondary property. Each cryptographic primitive contributes a clearly defined and isolated security function. This modular structure allows transparent reasoning about system behavior and simplifies the analysis of compositional security.

Although the current implementation supports secure deployment in controlled environments, it also provides a foundation for further research. Future work may extend the framework to support stronger composable privacy guarantees, more expressive predicate proofs, adaptive and dynamic revocation models, and migration toward post-quantum secure constructions.

In summary, the framework demonstrates that strong privacy guarantees, verifiable authenticity, and scalable revocation can be achieved simultaneously within a mathematically rigorous and practically efficient system. The results indicate that privacy-preserving identity systems can be constructed using well-understood cryptographic building blocks, without reliance on heavy trusted setup assumptions or complex algebraic structures, provided that careful protocol design is maintained.

References

[1] T. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Advances in Cryptology—CRYPTO*, 1991, pp. 129–140.

[2] C. P. Schnorr, "Efficient identification and signatures for smart cards," in *Advances in Cryptology—CRYPTO*, 1989, pp. 239–252.

[3] J. Camenisch and A. Lysyanskaya, "Dynamic accumulators and application to efficient revocation," in *Advances in Cryptology—CRYPTO*, 2002, pp. 61–76.

[4] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials," in *Advances in Cryptology—EUROCRYPT*, 2001, pp. 93–118.

[5] S. Brands, "U-Prove Cryptographic Specification V1.1," Microsoft Corporation, Tech. Rep., 2010.

[6] J. Groth, "On the size of pairing-based non-interactive arguments," in *Advances in Cryptology—EUROCRYPT*, 2016, pp. 305–326.

[7] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *Advances in Cryptology—CRYPTO*, 2004, pp. 41–55.

[8] B. Bünz et al., "Bulletproofs: Short proofs for confidential transactions and more," in *IEEE Symposium on Security and Privacy*, 2018, pp. 315–334.

[9] W3C, "Verifiable Credentials Data Model 1.0," 2019. [Online]. Available: https://www.w3.org/TR/vc-data-model/

[10] W3C, "Decentralized Identifiers (DIDs) v1.0," 2022. [Online]. Available: https://www.w3.org/TR/did-core/

[11] Hyperledger, "Hyperledger Indy: A Distributed Ledger for Decentralized Identity," 2018.

[12] Hyperledger, "Hyperledger Aries: Interoperable Identity Infrastructure," 2019.

[13] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Advances in Cryptology—CRYPTO*, 1986, pp. 186–194.

[14] J. Camenisch and A. Lysyanskaya, “Signature schemes and anonymous credentials from bilinear maps,” in *Advances in Cryptology—CRYPTO*, 2004, pp. 56–72.

[15] C. Boudot, “Efficient proofs that a committed number lies in an interval,” in *Advances in Cryptology—EUROCRYPT*, 2000, pp. 431–444.

[16] D. Boneh and M. Franklin, “Identity-based encryption from the Weil pairing,” in *Advances in Cryptology—CRYPTO*, 2001, pp. 213–229.

[17] E. Ben-Sasson et al., “SNARKs for C: Verifying program executions succinctly and in zero knowledge,” in *Advances in Cryptology—CRYPTO*, 2013.

[18] O. Goldreich, “Foundations of Cryptography: Volume 1,” Cambridge University Press, 2001.

[19] M. Bellare and P. Rogaway, “Random oracles are practical: A paradigm for designing efficient protocols,” in *ACM CCS*, 1993.

[20] C. Cramer, I. Damgård, and B. Schoenmakers, “Proofs of partial knowledge and simplified design of witness hiding protocols,” in *Advances in Cryptology—CRYPTO*, 1994.

[21] Data Breach Exposes Personal Information of Millions of Indian Citizens, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S016740482400083X

[22] Facebook–Cambridge Analytica Data Scandal, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Facebook-Cambridge_Analytica_data_scandal

[23] 2017 Equifax Data Breach. [Online]. Available: https://en.wikipedia.org/wiki/2017_Equifax_data_breach

[24] P. W. Shor, “Algorithms for quantum computation: Discrete logarithms and factoring,” in *Proc. 35th Annual Symposium on Foundations of Computer Science (FOCS)*, 1994, pp. 124–134.

[25] M. Mosca, “Cybersecurity in an era with quantum computers: Will we be ready?,” *IEEE Security & Privacy*, vol. 16, no. 5, pp. 38–41, 2018.

[26] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*, 3rd ed. Boca Raton, FL, USA: CRC Press, 2020.

[27] D. Hankerson, A. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*. New York, NY, USA: Springer, 2004.

[28] L. K. Grover, “A fast quantum mechanical algorithm for database search,” in *Proc. 28th Annual ACM Symposium on Theory of Computing (STOC)*, 1996, pp. 212–219.

[29] R. Canetti, “Universally composable security: A new paradigm for cryptographic protocols,” in *Proc. 42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, 2001, pp. 136–145.

[30] O. Regev, “On lattices, learning with errors, random linear codes, and cryptography,” in *Proc. 37th Annual ACM Symposium on Theory of Computing (STOC)*, 2005, pp. 84–93.

[31] J. Stern, “A new paradigm for public key identification,” *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1757–1768, 1996.

[32] A. Fiat and A. Shamir, “How to prove yourself: Practical solutions to identification and signature problems,” in *Advances in Cryptology—CRYPTO ’86*, LNCS 263, Springer, 1987, pp. 186–194.

[33] V. Lyubashevsky, “Lattice signatures without trapdoors,” in *Advances in Cryptology—EUROCRYPT 2012*, LNCS 7237, Springer, 2012, pp. 738–755.

[34] National Institute of Standards and Technology (NIST), “FIPS 204: Module-Lattice-Based Digital Signature Standard (CRYSTALS-Dilithium),” 2024.

[35] National Institute of Standards and Technology (NIST), “FIPS 205: Stateless Hash-Based Digital Signature Standard (SPHINCS+),” 2024.

[36] R. J. McEliece, “A public-key cryptosystem based on algebraic coding theory,” *DSN Progress Report*, Jet Propulsion Laboratory, 1978.

[37] J. Ding and D. Schmidt, “Multivariate public key cryptosystems,” in *Post-Quantum Cryptography*, Springer, 2009, pp. 193–241.

[38] D. Jao and L. De Feo, “Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies,” in *Post-Quantum Cryptography (PQCrypto)*, LNCS 7071, Springer, 2011, pp. 19–34.

[39] S. Goldwasser, S. Micali, and C. Rackoff, “The Knowledge Complexity of Interactive Proof Systems,” *SIAM Journal on Computing*, vol. 18, no. 1, pp. 186–208, 1989.

[40] J. Groth, ”On the Size of Pairing-Based Non-Interactive Arguments,” in *Proc. EUROCRYPT*, 2016, pp. 305–326.

[41] A. Gabizon, Z. J. Williamson, and O. Ciobotaru, ”PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge,” *IACR ePrint*, 2019/953, 2019.

[42] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, ”Scalable, Transparent, and Post-Quantum Secure Computational Integrity,” in *Proc. CRYPTO*, 2018, pp. 3–31.

[43] M. Chase, M. Kohlweiss, A. Lysyanskaya, and S. Meiklejohn, ”Signature Schemes with Efficient Protocols,” in *Proc. CRYPTO*, 2014, pp. 296–313.