Pairing-Free Nullifiers for BBS+ and PS Anonymous Credentials

Sam Polgar
The University of Sydney
Sydney, NSW, Australia
sam.polgar@sydney.edu.au

Abstract

By 2026, EU citizens will carry digital credentials in Self-Sovereign Identity (SSI) wallets, with similar initiatives in the US, South Korea, India and major identity providers Google and Microsoft. These systems promise privacy through anonymous authentication, but this creates critical security challenges. How do you prevent someone from voting twice, claiming benefits multiple times, or using revoked credentials when the user identity is hidden and the system cannot link their interactions?

We provide a solution to these challenges with fast, practical nullifiers; cryptographic values that enforce uniqueness without disclosing identity information. While existing nullifier schemes from payment systems like Zcash require expensive zkSNARKS or UTT using pairing operations, we achieve comparable security in prime-order groups and just 2.49 ms for nullifier generation and verification. We develop new Σ-protocols for proving inverse exponent knowledge, in doing so, we achieve a 3-6x speedup of the original Dodis-Yampolskiy VRF, and construct two pairing-free nullifier schemes: a deterministic variant for sybil resistance computed in 2.49ms and a rerandomizable variant for revocation computed in 3.66ms, outperforming existing approaches by 3-5x. Our constructions improve accountability in privacy systems, removing the efficiency roadblocks that prevent widespread private identity system adoption.

Keywords

privacy primitives, nullifiers, vrf, verifiable random function

ACM Reference Format:
Sam Polgar. YYYY. Pairing-Free Nullifiers for BBS+ and PS Anonymous Credentials. In . ACM, New York, NY, USA, 11 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Discuss challenges they discuss here in uniqueness etc. I am making a VRF first (which has different properties as the others have). Think about uniqueness, pseudorandomness, collission resistance.

A Generic Approach to Constructing and Proving Verifiable Random Functions

2 Introduction

One-time actions are commonplace in the real world; single-use tickets are issued for transport and concerts, and voting requires a user to present their identity document for verification and logging. One-time actions require *uniqueness*, each eligible user can perform a given action at most once. In physical systems, tickets may be collected or stamped at a location to prevent duplicate use. Digital systems use unique identifiers and database lists to record usage. Violating uniqueness constitutes a sybil attack against the system's identity controls.

Absent any privacy requirement, digital one-time actions are easily managed. In the voting example, a voting system would verify a user's digital credential and retrieve their personal information to be stored in their system. To prevent double-voting, each new user would be checked against the database, which would store, at a minimum, a unique identifier and perhaps some telemetry logs, IP address, device, browser, date, and time. In isolation, collecting this data might seem harmless, as one can assume an adversary doesn't have access to a system that links identifiers to personal information. However, it is well known [18, 30, 32] that adversaries with access to multiple incomplete data sources can aggregate information and identify individuals more easily than expected.

Self-Sovereign Identity (SSI) is emerging as the solution for digital credential management, transferring control of identity documents and certificates to users through personal credential wallets. This transition is accelerating globally: major identity providers have technical solutions, Google [29] and Microsoft [20], the EU mandates digital identity wallets by 2026 [13], nations including India [21] and Singapore [28] are already in operation and the US is piloting implementations across 13 states [1].

However, the benefits of credential wallets and their usage are overshadowed by enormous privacy risks. Anticipating this challenge, Chaum spawned Anonymous Credentials in the early 1980's [4, 8] which has grown into a large and mature body of work [2, 5–7, 14, 15, 23, 26] - cryptographic primitives that allow users to prove possession of valid credentials and attest to specific attributes without revealing their identity or unnecessary information. Unlike traditional digital signatures that expose the signer's identity, anonymous credentials enable privacy-preserving presentations where users share only the minimum information required by verifiers. Anonymous Credentials are being standardized with W3C specifications [33, 34] and IRTF proposal [17], positioning them as essential privacy infrastructure for digital credential wallets.

Anonymity ensures that when a user presents their credential, an adversary cannot determine which specific user generated the presentation, even when observing the cryptographic proofs and knowing the set of all issued credentials. Formally, this is captured by an indistinguishability game: given two users with valid credentials and a presentation from one of them, no polynomial-time adversary can distinguish which user created the presentation with more than negligible probability over random guessing.

As such, systems that require uniqueness and anonymity are met with a *conundrum*. Uniqueness guarantees a user performs a specific action once, implying that the user is able to be known, contradicting the anonymity property.

We provide an efficient solution for the anonymity-uniqueness problem through a cryptographic primitive called a *nullifier*. In our setting, users compute a deterministic but unlinkable value from their credential secret and an application-specific context (e.g., "vote2026"). This nullifier uniquely identifies a credential's use in that context without revealing which credential was used or leaking information about the user.

Concretely, when a user with credential secret k wishes to perform a one-time action with context x, they: (1) Compute nullifier $\mathsf{nf} = g^{1/(\mathsf{k}+x)}$ in a prime-order group (2) Generate a proof of correct computation that reveals nothing about k (3) Submit (nf, π) to the verifier

The verifier maintains a list of seen nullifiers per context. If nf already exists, the action is rejected as a duplicate. Otherwise, the verifier stores nf and accepts the action. Since the nullifier computation is deterministic, the same credential always produces the same nullifier for a given context, preventing sybil attacks while preserving privacy. Our rerandomizable nullifiers follow a similar scheme but output $\mathsf{cm}_{\mathsf{nf}} = g^{1/(\mathsf{k}+x)}h^r$ for a freshly sampled r.

Insufficiencies of Current Approaches. While several schemes achieve sybil resistance in privacy-preserving systems, none provide the efficiency and functionality required for real-world credential wallets.

Committee-based approaches like CanDID [19] and HADES [35] rely on MPC to compute PRF values over user identifiers (e.g., SSNs), generating nullifiers through threshold computation. This introduces fundamental scalability bottlenecks: communication latency scales with committee size, and real-time coordination requirements make deployment impractical for user-facing applications.

zkSNARK-based nullifiers in HADES [35], PLUME [16], and Zebra [25] achieve strong privacy guarantees but at prohibitive computational cost. While verification is fast (2ms), proof generation can take up to 35 seconds for PLUME [10]. These systems require zkSNARKs either due to non-algebraic operations (hash functions in PLUME's ECDSA verification) or blockchain optimization constraints where on-chain verification must be minimized.

Pairing-based proofs used in sybil-resistant identity systems SyRA [9] and S3ID [24] compute nullifiers locally using VRFs or similar primitives, proving correctness via pairing-based proofs. While avoiding MPC overhead, these schemes inherit the computational cost of pairing operations and are constrained by their underlying signature schemes' expressiveness. Furthermore, their signature constructions aren't conducive to attribute-based credential verification.

The closest construction to ours is the anonymous payment system UTT [31], which uses PS-Signatures [23] with Dodis-Yampolskiy VRF [11] for nullifier generation. The user computes the nullifier from a committed secret key within the signature and proves in zero knowledge the correctness of the output, with both a deterministic and a rerandomizable nullifier as outputs. While clever and more efficient than alternatives at 12ms, its pairing-based construction still imposes overhead for time-critical application scenarios like a one-time ticket verification at a transport turnstile.

Our work demonstrates both faster efficiency and broader deployability. Comparable security is achievable in prime-order groups at 2.49ms—a 3-6x improvement over the fastest existing approaches and our constructions can be used on the most popular non-pairing elliptic curves like secp256k1 and Ed25519.

Table 1: Contributions

Σ−protocols	Application
PoK-DY	Prime-Order DY VRF
PoK-Inverse Linear Relation (ILR)	Deterministic Nullifier
PoK-CommittedILR	Probabilistic Nullifier

2.1 Contributions

We design two pairing-free nullifier constructions optimized for credential wallet use-cases, which are the most computationally efficient to the best of our knowledge, without sacrificing security. First, our deterministic nullifier is used for one-time tokens and sybil resistance mechanisms. Second, our rerandomizable nullifier scheme outputs a commitment to the nullifier with a proof of correct computation for application use in revocation schemes. Our constructions are private in that they take in committed keys and do not leak user information. We provide an open-source implementation of our schemes in Rust and evaluate their performance. The deterministic nullifier computes in 2.49ms and rerandomizable computes in 3.66ms compared to the fastest existing work [31] at 12.38ms.

Our nullifiers are based on the q-type Decisional Diffie-Hellman Inversion Assumption ($q - DDHI$) and a novel pairing-free instantiation of the Dodis-Yampolskiy (DY) Verifiable Random Function (VRF), which we detail first as it forms the cryptographic backbone of our nullifier schemes. We develop three new Σ-protocols that prove the Dodis-Yampolskiy VRF inversion structure in our different scenarios, e.g., $1/(\mathsf{k} + x)$, and show that by removing pairings, the standard DY VRF achieves a performance speedup of 3x (from 3.47ms to 1.12ms) for BLS12-381 in our setting, or 6x (0.58ms) changing to the Ed25519 curve.

2.2 Related Work

The parallels between privacy-preserving payments and identity systems stem from Chaum's foundational work on blind signatures, which envisioned anonymous yet verifiable transactions and identity online. Spending an anonymous coin is analogous to a one-time action in private identity, where the user must prove the coin's uniqueness without revealing their identity, a concept central to both domains.

Systems like Zcash [36] and Tornado Cash [22] employ similar architectures, using Merkle Trees to record coins and zkSNARKs to prove a coin's existence and uniqueness via hash-based computations tied to a user's key. However, zkSNARK proof generation is computationally expensive, which is impractical for latency-sensitive applications like credential wallets. Moreover, their reliance on blockchain-based Merkle Trees for state management restricts offline functionality and cross-context use, both critical for wallets that must operate independently of a blockchain.

UTT [31] offers a more efficient alternative, basing its anonymous payment scheme on Anonymous Credentials—signed coins from a digital bank that hide user information and coin value. By avoiding zkSNARKs and leveraging Σ-protocols and signatures, UTT achieves significantly faster combined proof and verify times, making it more adaptable to identity systems than SNARK approaches.

2.3 Organisation

In Section 3 we introduce the preliminaries. In Section 4 we introduce the Pairing-Free Dodis-Yampolskiy VRF and its Σ-protocol to develop intuition for Section 5, the deterministic Nullifier, followed by Section 6, the rerandomizable nullifier and lastly implementation and evaluation in Section 7.

3 Preliminaries

Definition 3.1 (Bilinear Map). Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups of prime order p with multiplicative notation. $g, \tilde{g}$ are generators of $\mathbb{G}_1, \mathbb{G}_2$ respectively. The function $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a map (or pairing) if e is efficiently computable and

(1) Bilinear: $e(g^a, \tilde{g}^b)^c = e(g, \tilde{g})^{abc} = e(g^b, \tilde{g}^a)^c \ \forall a, b, c \in \mathbb{Z}_p$
(2) Non-degenerate: $e(g, \tilde{g}) \neq 1_{\mathbb{G}_T}$ that is, $e(g, \tilde{g})$ generates $\mathbb{G}_T$

The operation e computes a map or pairing A Bilinear Map (or Pairing) is an efficiently computable map Bilinear groups are a set of three groups and an efficiently computing map

Definition 3.2 (Σ-Protocol). A Σ-protocol for relation $\mathcal{R} = \{(x, w)\}$ is a three-round public-coin proof system between prover $\mathcal{P}$ and verifier $\mathcal{V}$:

(1) $\mathcal{P}(x, w) \to a$: Prover sends commitment a
(2) $\mathcal{V}(x) \to c$: Verifier sends challenge $c \leftarrow C$
(3) $\mathcal{P}(x, w, a, c) \to z$: Prover sends response z
(4) $\mathcal{V}(x, a, c, z) \to \{0, 1\}$: Verifier outputs accept/reject

A Σ-protocol satisfies:

- **Completeness**: $\Pr[\mathcal{V}(x, a, c, z) = 1 \mid (x, w) \in \mathcal{R}] = 1$
- **2-Special Soundness**: From accepting transcripts (a, c, z), (a, c', z') with $c \neq c'$, one can efficiently extract witness w s.t. $(x, w) \in \mathcal{R}$
- **Special HVZK**: $\exists$ simulator $\mathcal{S}$ that on input (x, c) outputs (a, c, z) indistinguishable from real transcripts

Definition 3.3 (Fiat-Shamir Transform). The Fiat-Shamir transform converts a Σ-protocol into a non-interactive proof by computing the challenge as $c = H(x, a)$ where $H : \{0, 1\}^* \to C$ is a cryptographic hash function modeled as a random oracle. The resulting proof is $\pi = (a, z)$ where z is computed using challenge c.

Definition 3.4 (Pedersen Commitment). Let $\mathbb{G}$ be a group of prime order p. The Pedersen commitment scheme consists of:

- Setup: Choose generators $g, h \in \mathbb{G}$ where $\log_g h$ is unknown
- Commit: For message $m \in \mathbb{Z}_p$, sample $r \leftarrow \mathbb{Z}_p$ and compute $C = g^m h^r$
- Open: Reveal (m, r); verify $C \stackrel{?}{=} g^m h^r$

For a vector of messages $(m_1, \ldots, m_n)$, use generators $(g_1, \ldots, g_n, h)$ and compute:

$$C = g_1^{m_1} \cdots g_n^{m_n} h^r$$

The scheme is perfectly hiding (for any C and m, there exists r such that $C = g^m h^r$) and computationally binding under the discrete logarithm assumption.

Definition 3.5 (Verifiable Random Function). A VRF is a tuple of PPT algorithms (Gen, Eval, Prove, Verify) with message space $\mathcal{X}$, output space $\mathcal{Y}$ and proof space Π where:

- $\mathsf{Gen}(1^\lambda) \to (sk, pk)$: Generates secret key sk and public key pk.
- $\mathsf{Eval}(sk, x) \to y$: Computes VRF output $y \in \mathcal{Y}$ for input $x \in \mathcal{X}$.
- $\mathsf{Prove}(sk, x) \to \pi$: Produces proof $\pi \in \Pi$ of correct evaluation.
- $\mathsf{Verify}(pk, x, y, \pi) \to \{0, 1\}$: Verifies correctness of y.

Definition 3.6 (q-DDHI Assumption). Let $\mathbb{G}$ be a group of prime order p with generator g. The q-Decisional Diffie-Hellman Inversion assumption states that given $(g, g^\alpha, g^{\alpha^2}, \ldots, g^{\alpha^q})$ for randomly chosen $\alpha \in \mathbb{Z}_p^*$, no PPT adversary can distinguish $g^{1/\alpha}$ from a random element in $\mathbb{G}$ with non-negligible advantage.

Formally, for all PPT adversaries $\mathcal{A}$:

$$\Big| \Pr[\mathcal{A}(g, g^\alpha, \ldots, g^{\alpha^q}, g^{1/\alpha}) = 1] - \Pr[\mathcal{A}(g, g^\alpha, \ldots, g^{\alpha^q}, g^r) = 1] \Big| \leq \mathsf{negl}(\lambda)$$

where $\alpha \leftarrow \$\mathbb{Z}_p^*$ and $r \leftarrow \$\mathbb{Z}_p^*$.

4 Pairing-free DY VRF from q-DDHI

The first step towards our privacy-preserving nullifiers is the base building block, a secure VRF with unique outputs. We first show how we transform the original, pairing-based DY VRF [11] based on the q-type Decisional Bilinear Diffie-Hellman Inversion ($q-DBDHI$) assumption into a pairing-free (prime-order) VRF based on the $q-DDHI$. Our main insight is to change the proof of correctness from using a bilinear map to using a Σ-protocol, rather than verifying objects directly in an information-theoretic way, we verify an equivalent computation and prove its correctness, soundness, and zero knowledge.

We start by recalling the Dodis-Yampolskiy VRF which generates unique, pseudorandom nullifiers y and their proofs of correctness π and then the question of removing pairings to improve efficiency. The prover generates the following:

$$y = e(g, \tilde{g})^{1/(\mathrm{k}+x)} \qquad \pi = \tilde{g}^{1/(\mathrm{k}+x)}$$

Verification depends on the bilinearity property $e(g^a, \tilde{g}^b)^c = e(g, \tilde{g})^{abc}$, which enables "exponent multiplication" across groups. A verifier selects x, is given (y, π, pk) where $pk = g^{\mathrm{k}}$ from the prover and runs verification algorithm

$$\begin{aligned} e(g^x \cdot pk, \pi) &\stackrel{?}{=} e(g^{s+x}, \tilde{g}^{1/(s+x)}) & y &\stackrel{?}{=} e(g, \pi) \\ &\stackrel{?}{=} e(g, \tilde{g})^{(s+x)/(s+x)} & &\stackrel{?}{=} e(g, \tilde{g}^{1/(s+x)}) \\ &= e(g, \tilde{g}) \quad \checkmark & &= e(g, \tilde{g})^{1/(s+x)} \quad \checkmark \end{aligned}$$

Security relies on the q-DBDHI problem which states that given $(g, g^x, g^{x^2}, \dots, g^{x^q}, \tilde{g})$, no PPT adversary can distinguish between $e(g,\tilde{g})^{1/x}$ and a uniform element in $\mathbb{G}_T$ with non-negligible advantage, ensuring the VRF outputs maintain pseudorandomness after the adversary has observed (x', y', π') pairs.

Prime-order groups and standard group operations (without the bilinearity property) cannot be used to directly verify the $1/(\mathsf{k}+x)$ relationship with standard cryptographic operations. A verifier given pk can compute $(pk \cdot g^x) = g^{\mathsf{k}+x}$ but verification attempts using this fail:

(1) $g^{\mathsf{k}+x} \cdot g^{1/(\mathsf{k}+x)} = g^{(\mathsf{k}+x)+\frac{1}{\mathsf{k}+x}}$
(2) $\frac{g^{\mathsf{k}+x}}{g^{1/(\mathsf{k}+x)}} = g^{(\mathsf{k}+x)^2-\frac{1}{\mathsf{k}+x}}$

Failing such approaches, our insight is to reverse the verification. Instead of trying to derive $g^{1/(s+x)}$ from pk, x or cancel with a reciprocal, we use a zero knowledge proof Σ-protocol to verify that y raised to the exponent $(\mathsf{k}+x)$ equals g. In doing so, our pairing-free construction shifts from the q-DBDHI assumption to the q-DDHI assumption. This gives us the relation:

$$\mathcal{R}_{\mathsf{DY-PF}} = \left\{ \begin{array}{l} (\mathsf{pk}, x, y), \\ (\mathsf{k}) \end{array} \middle| \begin{array}{l} pk = g^{\mathsf{k}} \quad \wedge \\ y^{\mathsf{k}+x} = g \end{array} \right\}$$

4.1 Security Model

4.1.1 Syntax. A Verifiable Random Function (VRF) scheme Π_{VRF} consists of a tuple of polynomial-time algorithms (Gen, Eval, Prove, Verify) with a secret key space $\mathcal{SK}$, a public key space $\mathcal{PK}$, an input message space $\mathcal{X}$, an output space $\mathcal{Y}$, and a proof space Π:

- $\mathsf{Gen}(1^\lambda) \to (sk, pk)$: On input the security parameter 1^λ, this algorithm outputs a secret key $sk \in \mathcal{SK}$ and a corresponding public key $pk \in \mathcal{PK}$.
- $\mathsf{Eval}(sk, x) \to y$: On input a secret key $sk \in \mathcal{SK}$ and a message $x \in \mathcal{X}$, this algorithm outputs a VRF value $y \in \mathcal{Y}$.
- $\mathsf{Prove}(sk, x) \to \pi$: On input a secret key $sk \in \mathcal{SK}$ and a message $x \in \mathcal{X}$ (for which $y = \mathsf{Eval}(sk, x)$), this algorithm outputs a proof $\pi \in \Pi$ attesting to the correct computation of y.
- $\mathsf{Verify}(pk, x, y, \pi) \to \{0, 1\}$: On input a public key $pk \in \mathcal{PK}$, a message $x \in \mathcal{X}$, a VRF value $y \in \mathcal{Y}$, and a proof $\pi \in \Pi$, this algorithm outputs 1 if y is a valid VRF output for pk and x according to π, and 0 otherwise.

Our pairing-free DY VRF must satisfy three security properties: completeness, uniqueness, and pseudorandomness. We formalize these for our construction operating in prime-order group $\mathbb{G}$ of order p.

Definition 4.1 (Completeness). For all $(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$ and all $x \in \mathcal{X}$:

$$\Pr[\mathsf{Verify}(pk, x, \mathsf{Eval}(sk, x), \mathsf{Prove}(sk, x)) = 1] = 1$$

Definition 4.2 (Computational Uniqueness). For all PPT adversaries $\mathcal{A}$:

$$\Pr\left[\begin{array}{c} y_0 \neq y_1 \wedge \\ \mathsf{Verify}(pk, x, y_0, \pi_0) = 1 \wedge \\ \mathsf{Verify}(pk, x, y_1, \pi_1) = 1 \end{array} \middle| \begin{array}{c} (pk, x, y_0, \pi_0, y_1, \pi_1) \\ \leftarrow \mathcal{A}(1^\lambda) \end{array} \right] \leq \mathsf{negl}(\lambda)$$

This differs from standard VRF uniqueness which is information-theoretic. Our construction achieves computational uniqueness under the discrete logarithm assumption in $\mathbb{G}$.

Definition 4.3 (Pseudorandomness under q-DDHI). Consider the following experiment $\mathsf{Exp}^{\text{vrf-pr}}_{\mathcal{A}}(\lambda)$ as per [3]:

(1) $(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$; give pk to $\mathcal{A}$
(2) $\mathcal{A}$ adaptively queries $x_1, \dots, x_q \in \mathcal{X}$, receiving (y_i, π_i) where $y_i = g^{1/(sk+x_i)}$, $\pi_i \leftarrow \mathsf{Prove}(sk, x_i)$
(3) $\mathcal{A}$ outputs challenge $x^* \notin \{x_1, \dots, x_q\}$
(4) Sample $b \leftarrow \{0,1\}$. If $b = 0$: $y^* = g^{1/(sk+x^*)}$. If $b = 1$: $y^* \leftarrow \mathbb{G}$
(5) $\mathcal{A}$ outputs b'; experiment outputs 1 if $b' = b$

The VRF satisfies pseudorandomness if for all PPT $\mathcal{A}$ making at most $q(\lambda)$ queries:

$$\mathsf{Adv}^{\text{vrf-pr}}_{\mathcal{A}}(\lambda) := \left| \Pr[\mathsf{Exp}^{\text{vrf-pr}}_{\mathcal{A}}(\lambda) = 1] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda)$$

THEOREM 4.4 (PSEUDORANDOMNESS FROM q-DDHI). *If the q-DDHI assumption holds in $\mathbb{G}$, then the pairing-free DY VRF satisfies pseudorandomness against adversaries making at most q evaluation queries.*

PROOF INTUITION. We reduce VRF pseudorandomness to q-DDHI. Given a q-DDHI challenge $(g, g^\alpha, g^{\alpha^2}, \dots, g^{\alpha^q})$ and target T (either $g^{1/\alpha}$ or random), we construct a VRF adversary:

(1) **Setup**: For challenge input x^* chosen by $\mathcal{A}$, implicitly set the VRF secret key as $\mathsf{k} = \alpha - x^*$. Publish $pk = g^{\mathsf{k}} = g^\alpha \cdot g^{-x^*}$.
(2) **Evaluation Queries**: For query $x_i \neq x^*$:
    - Need $y_i = g^{1/(\mathsf{k}+x_i)} = g^{1/(\alpha - x^* + x_i)}$
    - Since $x_i \neq x^*$, we can compute this using polynomial interpolation on the q-DDHI instance
    - Simulate proof π_i using the zero-knowledge simulator for the Σ-protocol
(3) **Challenge**: For input x^*:
    - $y^* = g^{1/(\mathsf{k}+x^*)} = g^{1/((\alpha - x^*)+x^*)} = g^{1/\alpha} = T$
    - If T is the real q-DDHI output, then y^* is correct
    - If T is random, then y^* is random

If $\mathcal{A}$ distinguishes y^* from random, we solve q-DDHI. □

4.2 Construction

Our VRF operates in a prime-order group $\mathbb{G}$ of order p with generator g. The message space is $\mathcal{X} = \mathbb{Z}_p$, the output space is $\mathcal{Y} = \mathbb{G}$, and the proof space is $\Pi = \mathbb{G} \times \mathbb{G} \times \mathbb{Z}_p$. The algorithms are defined as follows:

- $\mathsf{VRF.Gen}(1^\lambda) \to (s, pk)$: Sample $\mathsf{k} \leftarrow \$\mathbb{Z}_p^*$, compute $pk = g^s$, and output (s, pk).
- $\mathsf{VRF.Eval}(s, x) \to y$: Compute $y = g^{1/(\mathsf{k}+x)} \in \mathbb{G}$.
- $\mathsf{VRF.Prove}(s, x) \to \pi$: Generate proof π using the DY Prime Order Proof Protocol 1.
- $\mathsf{VRF.Verify}(pk, x, y, \pi) \to \{0,1\}$: Output 1 if π verifies y correctly per the Σ-protocol, else 0.

Remark: Verifying $\mathsf{VRF.Verify}(pk, x, \mathsf{VRF.Eval}(s, x) \to y) \to 1$ is a naive verification approach without a proof which yeilds a Verifiable Unpredictable Function (VUF), not a VRF because it lacks the mechanism to prove pseudorandomness to a verifier. DY uses pairings to bridge the gap, we replace pairings with a Σ-protocol.

Figure 1: Pairing-free DY Σ-Protocol

Common input: $g \in \mathbb{G},\ x \in \mathbb{Z}_p,\ pk, y \in \mathbb{G}$

Relation: $pk = g^{sk}$ and $y^{sk+x} = g$

Prover (sk)	**Verifier**
pick $r \leftarrow \mathbb{Z}_p$	
$T_1 := g^r, \quad T_2 := y^r$	
$\rightarrow (T_1, T_2)$	
	pick $c \leftarrow \mathbb{Z}_p$
	$\leftarrow c$
$z := r + c\,(sk + x)$	
$\rightarrow z$	
	Accept iff
	$g^z = T_1\,(pk\,g^x)^c$
	$y^z = T_2\,g^c$

4.3 Security Analysis

Our construction replaces the information-theoretic uniqueness of the original DY VRF with computational uniqueness via the Σ-protocol and discrete logarithm assumption.

4.3.1 Correctness. Correctness requires that an honest prover's output y and proof π always pass verification. For $pk = g^{\mathsf{k}}$, $y = g^{1/(\mathsf{k}+x)}$, $T_1 = g^r$, $T_2 = y^r$, and $z = r + c(\mathsf{k} + x)$, the verification equations hold:

$$g^z = g^{r+c(\mathsf{k}+x)} = g^r \cdot g^{c(\mathsf{k}+x)} = g^r \cdot (g^{\mathsf{k}} \cdot g^x)^c = T_1 \cdot (pk \cdot g^x)^c$$
$$y^z = y^{r+c(\mathsf{k}+x)} = y^r \cdot y^{c(\mathsf{k}+x)} = y^r \cdot (y^{\mathsf{k}+x})^c = y^r \cdot g^c = T_2 \cdot g^c$$

Since $y^{\mathsf{k}+x} = g^{1/(\mathsf{k}+x)\cdot(\mathsf{k}+x)} = g$, both checks pass, confirming correctness.

4.3.2 Uniqueness. Uniqueness ensures that, for a fixed pk and x, only one y can be successfully verified. In DY, pairings enforce this information theoretically. In our pairing-free DY, uniqueness is computational, relying on the discrete logarithm problem.

For a valid y, $y^{\mathsf{k}+x} = g$, so $y = g^{1/(\mathsf{k}+x)}$ is unique in $\mathbb{G}$. Suppose an adversary produces $y' \neq y$ with a valid proof π'. Then $y'^{\mathsf{k}+x} = g$ and $y^{\mathsf{k}+x} = g$, implying $(y'/y)^{\mathsf{k}+x} = 1$. In a prime-order group, $y'/y = g^k$ for some $k \neq 0$, so $y' = y \cdot g^k$. But $y'^{\mathsf{k}+x} = (y \cdot g^k)^{\mathsf{k}+x} = g \cdot g^{k(\mathsf{k}+x)} = g$ requires $g^{k(\mathsf{k}+x)} = 1$, which holds only if $k(\mathsf{k}+x) = 0 \pmod p$. For random k and x, $\mathsf{k} + x = 0$ is negligible. Alternatively, if y' corresponds to a different k' where $pk = g^{\mathsf{k}'}$, finding $\mathsf{k}' \neq sk$ breaks the discrete logarithm assumption.

Thus, producing a distinct verifiable y' is computationally infeasible, ensuring uniqueness.

4.3.3 Pseudorandomness. Pseudorandomness requires that $y = g^{1/(\mathsf{k}+x)}$ appears random in $\mathbb{G}$ without knowledge of k, even given other input-output pairs. We rely on the q-DDHI assumption, which states that $g^{1/(\mathsf{k}+x)}$ is indistinguishable from random given $(g, g^s, \ldots, g^{(\mathsf{k})^q})$ for polynomial q. The Σ-protocol is zero-knowledge, leaking no information about k beyond pk.

Proof Sketch. Assume an adversary can distinguish y from random, solving q-DDHI. The challenger simulates proofs for q inputs using g^{k^i} for challenge x^*, provides $y^* = g^{a/(\mathsf{k}+x^*)}$ or a random element. A successful distinguished implies a q-DDHI solver which is assumed to be a hard problem. □

5 Deterministic Nullifier

5.0.1 Syntax. A deterministic nullifier scheme Π_{dnf} consists of a tuple of polynomial-time algorithms (Setup, KeyGen, Eval, Prove, Verify) with a secret key space $\mathcal{SK}$, a randomness space $\mathcal{R}$, a commitment space $\mathcal{CM}$, an input space $\mathcal{X}$, a nullifier space $\mathcal{NF}$, and a proof space Π:

- $\mathsf{Setup}(1^\lambda) \rightarrow pp$: On input the security parameter 1^λ, this algorithm outputs public parameters pp which include a prime-order group $\mathbb{G}$ of order p with generators $g, g_1 \in \mathbb{G}$.
- $\mathsf{KeyGen}(pp) \rightarrow (sk, r, cm)$: On input the public parameters pp, this algorithm outputs a secret key $sk \in \mathcal{SK}$, randomness $r \in \mathcal{R}$, and a corresponding commitment $cm \in \mathcal{CM}$, where $cm = g_1^{sk} g^r$.
- $\mathsf{Eval}(sk, x) \rightarrow nf$: On input a secret key $sk \in \mathcal{SK}$ and an input $x \in \mathcal{X}$, this algorithm outputs a nullifier $nf \in \mathcal{NF}$, where $nf = g^{1/(sk+x)}$.
- $\mathsf{Prove}(pp, sk, r, x, cm) \rightarrow \pi$: On input public parameters pp, a secret key $sk \in \mathcal{SK}$, randomness $r \in \mathcal{R}$, and an input $x \in \mathcal{X}$ (for which $nf = \mathsf{Eval}(sk, x)$ and $cm = g_1^{sk} g^r$), this algorithm outputs a proof $\pi \in \Pi$ attesting to the correct computation of nf with respect to cm.
- $\mathsf{Verify}(pp, cm, x, nf, \pi) \rightarrow \{0, 1\}$: On input public parameters pp, a commitment $cm \in \mathcal{CM}$, an input $x \in \mathcal{X}$, a nullifier $nf \in \mathcal{NF}$, and a proof $\pi \in \Pi$, this algorithm outputs 1 if nf is a valid nullifier for cm and x according to π, and 0 otherwise.

5.1 Security Model

5.1.1 Correctness. A deterministic nullifier scheme Π_{dnf} satisfies correctness if for all $\lambda \in \mathbb{N}$, all $pp \leftarrow \mathsf{Setup}(1^\lambda)$, all $(sk, r, cm) \leftarrow \mathsf{KeyGen}(pp)$, and all $x \in \mathcal{X}$ such that $sk + x \not\equiv 0 \pmod p$:

$$\Pr\left[\mathsf{Verify}(pp, cm, x, nf, \pi) = 1 : \begin{array}{l} nf \leftarrow \mathsf{Eval}(sk, x) \\ \pi \leftarrow \mathsf{Prove}(pp, sk, r, x, cm) \end{array}\right] = 1$$

5.1.2 Uniqueness. A deterministic nullifier scheme Π_{dnf} satisfies uniqueness if for all PPT adversaries $\mathcal{A}$:

$$\Pr\left[\mathsf{Game}_{\mathcal{A}}^{\mathsf{unique}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda)$$

where $\mathsf{Game}_{\mathcal{A}}^{\mathsf{unique}}(\lambda)$ is defined as:

(1) $pp \leftarrow \mathsf{Setup}(1^\lambda)$
(2) $(cm, x, nf_0, \pi_0, nf_1, \pi_1) \leftarrow \mathcal{A}(pp)$
(3) Return 1 if and only if:

- $nf_0 \neq nf_1$
- $\mathsf{Verify}(pp, cm, x, nf_0, \pi_0) = 1$
- $\mathsf{Verify}(pp, cm, x, nf_1, \pi_1) = 1$

5.1.3 Unforgeability. A deterministic nullifier scheme Π_{dnf} satisfies unforgeability if for all PPT adversaries $\mathcal{A}$:

$$\Pr\left[\mathrm{Game}^{\mathrm{unf}}_{\mathcal{A}}(\lambda) = 1\right] \leq \mathrm{negl}(\lambda)$$

where $\mathrm{Game}^{\mathrm{unf}}_{\mathcal{A}}(\lambda)$ is defined as:

(1) $pp \leftarrow \mathsf{Setup}(1^\lambda)$
(2) $(sk, r, cm) \leftarrow \mathsf{KeyGen}(pp)$
(3) $Q \leftarrow \emptyset$ // Set of queried contexts
(4) $(x^*, nf^*, \pi^*) \leftarrow \mathcal{A}^{O_{\mathrm{eval}}}(pp, cm)$
(5) Return 1 if and only if:
  - $x^* \notin Q$
  - $\mathsf{Verify}(pp, cm, x^*, nf^*, \pi^*) = 1$

Oracle $O_{\mathrm{eval}}(x)$:

(1) $Q \leftarrow Q \cup \{x\}$
(2) $nf \leftarrow \mathsf{Eval}(sk, x)$
(3) $\pi \leftarrow \mathsf{Prove}(pp, sk, r, x, cm)$
(4) Return (nf, π)

5.1.4 Pseudorandomness. A deterministic nullifier scheme Π_{dnf} satisfies pseudorandomness if for all PPT adversaries $\mathcal{A}$:

$$\left|\Pr\left[\mathrm{Game}^{\text{pr-0}}_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathrm{Game}^{\text{pr-1}}_{\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathrm{negl}(\lambda)$$

where $\mathrm{Game}^{\text{pr-b}}_{\mathcal{A}}(\lambda)$ for $b \in \{0, 1\}$ is defined as:

(1) $pp \leftarrow \mathsf{Setup}(1^\lambda)$
(2) $(sk, r, cm) \leftarrow \mathsf{KeyGen}(pp)$
(3) $Q \leftarrow \emptyset$ // Set of queried contexts
(4) $x^* \leftarrow \mathcal{A}^{O_{\mathrm{eval}}}(pp, cm)$ such that $x^* \notin Q$
(5) If $b = 0$: $nf^* \leftarrow \mathsf{Eval}(sk, x^*)$
(6) If $b = 1$: $nf^* \leftarrow \mathcal{NF}$ // Random element from nullifier space
(7) $b' \leftarrow \mathcal{A}(nf^*)$
(8) Return 1 if $b' = b$

5.1.5 Multi-Context Anonymity. This definition assumes that the users commitment cm can be rerandomized each time it is presented, using fresh randomness ensuring that its format differs across presentations. As a result, the system cannot track or link the commitment (the user's identity) to previous instances, even when the same secret key sk is used.

An adversary should not learn which of two honest credentials underlies an *arbitrary sequence* of nullifiers, even though each nullifier is deterministic in its context.

Game $\mathrm{ANON}^{m}_{\mathrm{det}}(\lambda)$.

(1) $\mathrm{pp} \leftarrow \mathsf{Setup}(1^\lambda)$.
(2) $(sk_0, r_0, cm_0), (sk_1, r_1, cm_1) \leftarrow \mathsf{KeyGen}(\mathrm{pp})$.
(3) Challenger chooses bit $b \leftarrow \$\{0, 1\}$.
(4) **Oracle O_{null}:** On adversary input $x \in \mathcal{X}$
  (a) Sample $\rho \leftarrow \$\mathbb{Z}_p$ and set $cm' := cm_b \cdot g^\rho$.
  (b) Compute $nf := \mathsf{Eval}(sk_b, x)$ and $\pi \leftarrow \mathsf{Prove}(\mathrm{pp}, sk_b, r_b + \rho, x, cm')$.
  (c) Return (cm', x, nf, π).

  The adversary may query O_{null} polynomially many times, with distinct or repeated contexts.
(5) Eventually the adversary outputs a guess b'.

Definition 5.1. The deterministic scheme Π_{dnf} satisfies *multi-context anonymity* if for every PPT adversary $\mathcal{A}$ making at most $\mathrm{poly}(\lambda)$ oracle queries,

$$\left|\Pr\left[b' = b\right] - \tfrac{1}{2}\right| \leq \mathrm{negl}(\lambda).$$

Intuition. Each call returns a fresh Pedersen rerandomisation cm' that is computationally independent of previous ones; the Σ-proof reveals no information beyond membership in the relation. Hence the adversary learns at most that *some* credential with a valid witness was used—never which one no matter how many contexts are queried. Applications prevent replay at the protocol layer, the security game permits repeated x though the deterministic nf will be returned from the oracle and break anonymity.

5.2 Construction

Our deterministic nullifier scheme operates in a prime-order group $\mathbb{G}$ of order p with generators $g, g_1 \in \mathbb{G}$. The secret key space is $\mathcal{SK} = \mathbb{Z}_p^*$, the randomness space is $\mathcal{R} = \mathbb{Z}_p$, the commitment space is $\mathcal{CM} = \mathbb{G}$, the input space is $\mathcal{X} = \mathbb{Z}_p$, the nullifier space is $\mathcal{NF} = \mathbb{G}$, and the proof space is $\Pi = \mathbb{G} \times \mathbb{G} \times \mathbb{Z}_p \times \mathbb{Z}_p$. The algorithms are defined as follows:

- $\mathsf{Setup}(1^\lambda) \rightarrow pp$: Generate a prime-order group $\mathbb{G}$ of order p with generators $g, g_1 \in \mathbb{G}$, and set $pp = (\mathbb{G}, p, g, g_1)$.
- $\mathsf{KeyGen}(pp) \rightarrow (sk, r, cm)$: Sample $sk \leftarrow \$\mathbb{Z}_p^*$, $r \leftarrow \$\mathbb{Z}_p$, compute $cm = g_1^{sk} g^r$, and output (sk, r, cm).
- $\mathsf{Eval}(sk, x) \rightarrow nf$: Compute $nf = g^{1/(sk+x)} \in \mathbb{G}$.
- $\mathsf{Prove}(pp, sk, r, x, cm) \rightarrow \pi$: Generate proof π using the Nullifier Proof Protocol 2.
- $\mathsf{Verify}(pp, cm, x, nf, \pi) \rightarrow \{0, 1\}$: Output 1 if π verifies correctly per the Nullifier Proof Protocol, else 0.

5.3 Security Analysis

We now prove that our deterministic nullifier construction satisfies all defined security properties.

Theorem 5.2 (Correctness). *The deterministic nullifier scheme satisfies correctness.*

Proof. Correctness follows directly from the completeness of the underlying Σ-protocol (Figure 2). For honestly generated values where $\mathrm{cm} = g_1^{\mathrm{k}} g^r$, $\mathrm{nf} = g^{1/(\mathrm{k}+x)}$, and a correctly computed proof π, the verification equations in the Σ-protocol are satisfied by construction. Therefore, $\mathsf{Verify}(\mathrm{pp}, \mathrm{cm}, x, \mathrm{nf}, \pi) = 1$ with probability 1. □

Theorem 5.3 (Uniqueness). *Under the discrete logarithm assumption and the binding property of Pedersen commitments, the deterministic nullifier scheme satisfies uniqueness.*

Proof. Suppose an adversary $\mathcal{A}$ outputs $(\mathrm{cm}, x, \mathrm{nf}_0, \pi_0, \mathrm{nf}_1, \pi_1)$ such that $\mathrm{nf}_0 \neq \mathrm{nf}_1$ and both verify successfully. By the special soundness property of the Σ-protocol, from the two accepting proofs we can extract witnesses (k_0, r_0) and (k_1, r_1) such that:

- $\mathrm{cm} = g_1^{\mathrm{k}_0} g^{r_0} = g_1^{\mathrm{k}_1} g^{r_1}$

Figure 2: Nullifier Σ-Protocol for relation $\mathcal{R}_{\mathsf{nf}}$

Common input: $g_1, g \in \mathbb{G},\ x \in \mathbb{Z}_p,\ cm_1, y \in \mathbb{G}$

Relation: $cm_1 = g_1^{sk}\, g^{r_1}$ and $y^{sk+x} = g$

Prover (sk, r_1)	**Verifier** (x)
pick $a_s, a_r \leftarrow \mathbb{Z}_p$	
$T_1 := g_1^{a_s}\, g^{a_r},\ T_y := y^{a_s}$	
$\rightarrow (T_1, T_y)$	
	pick $c \leftarrow \mathbb{Z}_p$
	$\leftarrow c$
$z_s := a_s + c\, sk$	
$z_r := a_r + c\, r_1$	
$z_m := a_s + c\,(sk + x)$	
$\rightarrow (z_s, z_r, z_m)$	
	Accept iff
	$T_1\, cm_1^c = g_1^{z_s}\, g^{z_r}$
	$T_y\, (y^x)^c = y^{z_m}$
	$z_m = z_s + c\, x$

- $\mathsf{nf}_0^{\mathsf{k}_0+x} = g$ and $\mathsf{nf}_1^{\mathsf{k}_1+x} = g$

From the binding property of Pedersen commitments, we have $\mathsf{k}_0 = \mathsf{k}_1$ with overwhelming probability. This implies $\mathsf{nf}_0^{\mathsf{k}_0+x} = \mathsf{nf}_1^{\mathsf{k}_0+x} = g$, which means $(\mathsf{nf}_0/\mathsf{nf}_1)^{\mathsf{k}_0+x} = 1$.

In a prime-order group, this implies either $\mathsf{nf}_0 = \mathsf{nf}_1$ (contradiction) or $\mathsf{k}_0 + x \equiv 0 \pmod{p}$. The latter occurs with negligible probability $1/p$ for randomly chosen k and any x. □

Theorem 5.4 (Unforgeability). *Under the binding property of Pedersen commitments and the discrete logarithm assumption, the deterministic nullifier scheme satisfies unforgeability.*

Proof. We show that any adversary $\mathcal{A}$ winning the unforgeability game can be used to break the binding property of commitments.

Given an adversary that outputs $(x^*, \mathsf{nf}^*, \pi^*)$ for an unqueried x^*, by the special soundness of the Σ-protocol, we can extract (k', r') such that $\mathsf{cm} = g_1^{\mathsf{k}'} g^{r'}$ and $\mathsf{nf}^* = g^{1/(\mathsf{k}'+x^*)}$.

Since the adversary only received nullifiers for $x_i \neq x^*$, and the nullifier function $\mathsf{nf} = g^{1/(\mathsf{k}+x)}$ is deterministic, the adversary must have either:

(1) Found $\mathsf{k}' \neq \mathsf{k}$ such that $\mathsf{cm} = g_1^{\mathsf{k}} g^{r} = g_1^{\mathsf{k}'} g^{r'}$, breaking commitment binding
(2) Computed $g^{1/(\mathsf{k}+x^*)}$ without knowing k, which requires solving the discrete logarithm problem

Both events occur with negligible probability. □

Theorem 5.5 (Pseudorandomness). *Under the q-DDHI assumption, the deterministic nullifier scheme satisfies pseudorandomness against adversaries making at most q evaluation queries.*

Proof. We construct a reduction $\mathcal{B}$ that uses a nullifier pseudorandomness adversary $\mathcal{A}$ to solve the q-DDHI problem.

Given a q-DDHI instance $(g, g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^q}, T)$ where T is either $g^{1/\alpha}$ or random, $\mathcal{B}$ proceeds as follows:

(1) **Setup:** $\mathcal{B}$ implicitly sets $\mathsf{k} = \alpha - x^*$ where x^* is the challenge input (chosen by $\mathcal{A}$ later). It computes $\mathsf{cm} = g_1^{\alpha - x^*} \cdot g^r$ for random r.
(2) **Evaluation Queries:** For query $x_i \neq x^*$, $\mathcal{B}$ needs to compute $\mathsf{nf}_i = g^{1/(\mathsf{k}+x_i)} = g^{1/(\alpha - x^* + x_i)}$. Since $x_i \neq x^*$, the denominator is non-zero and $\mathcal{B}$ can compute this using polynomial interpolation on the q-DDHI instance. The proof π_i is simulated using the HVZK simulator.
(3) **Challenge:** When $\mathcal{A}$ outputs challenge x^*, $\mathcal{B}$ returns $\mathsf{nf}^* = T^{1/(\mathsf{k}+x^*)} = T^{1/((\alpha - x^*)+x^*)} = T$.
(4) If $\mathcal{A}$ distinguishes nf^* from random, $\mathcal{B}$ outputs the same guess for the q-DDHI challenge.

The simulation is perfect up to the HVZK simulation error, which is negligible. Therefore, any non-negligible advantage in distinguishing nullifiers translates to a non-negligible advantage in solving q-DDHI. □

Theorem 5.6 (Multi-Context Anonymity). *The deterministic nullifier scheme satisfies multi-context anonymity under the hiding property of Pedersen commitments and the HVZK property of the Σ-protocol.*

Proof. In the anonymity game, the adversary receives $(cm', x, \mathsf{nf}, \pi)$ where $cm' = \mathsf{cm}_b \cdot g^{\rho}$ is a rerandomization of one of two commitments. We show the adversary's view is independent of b.

(1) The rerandomized commitment $cm' = g_1^{\mathsf{k}_b} g^{r_b+\rho}$ is uniformly distributed over $\mathbb{G}$ due to the fresh randomness ρ, regardless of b.
(2) While $\mathsf{nf} = g^{1/(\mathsf{k}_b+x)}$ is deterministic given (k_b, x), the adversary cannot link it to b without knowing k_b. The commitment hiding ensures k_b remains hidden.
(3) The proof π is generated honestly but reveals nothing about k_b beyond the validity of the statement, due to the HVZK property.

Since all components of the adversary's view are computationally independent of b, the advantage is negligible. Note that the adversary cannot query the same context x multiple times in practice, as this would break anonymity trivially (same x produces same deterministic nf). □

6 Rerandomizable Nullifier

While deterministic nullifiers prevent sybil attacks, they can't be used for some revocation list checks when a public commitment to values are required [27]. In the deterministic variant, each verification reveals the same nullifier $\mathsf{nf} = g^{1/(\mathsf{k}+x)}$, potentially allowing systems to track users across interactions. We address this with a *rerandomizable nullifier* that outputs a fresh commitment to the nullifier for each presentation. Due to the position binding property of the multi-message commitment, the user proves the correctness of the exponent at that position using the base generator.

6.1 Motivation: Privacy-Preserving Revocation

Consider a service that needs to revoke access for compromised credentials without compromising the privacy of legitimate users. With deterministic nullifiers, checking against a revocation list requires revealing nf, creating a linkability point. Our rerandomizable construction outputs:

$$\mathsf{cm}_{\mathsf{nf}} = g_3^{1/(\mathsf{k}+x)} g^{r_3}$$

where $r_3 \leftarrow \$\mathbb{Z}_p$ is fresh for each presentation. This enables users to prove in zero-knowledge that their nullifier is *not* in a revocation list without revealing the nullifier itself.

6.2 Security Model

6.2.1 Anonymity. The rerandomizable nullifier achieves full anonymity: presentations are unlinkable even when derived from the same (k, x) pair.

Definition 6.1 (Anonymity). A rerandomizable nullifier scheme satisfies anonymity if for all PPT adversaries $\mathcal{A}$:

$$\left|\Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-1}}(\lambda) = 1] - \Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-0}}(\lambda) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where in $\mathsf{Exp}_{\mathcal{A}}^{\text{anon-b}}(\lambda)$:

(1) Setup $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, generate $(\mathsf{k}_0, r_{1,0}, \mathsf{cm}_{1,0})$, $(\mathsf{k}_1, r_{1,1}, \mathsf{cm}_{1,1}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
(2) $\mathcal{A}$ receives $\mathsf{pp}, \mathsf{cm}_{1,0}, \mathsf{cm}_{1,1}$ and chooses (x, r_2, cm_2)
(3) Challenger computes $(\mathsf{cm}_{\mathsf{nf}}, \mathsf{cm}_4, \pi) \leftarrow \mathsf{Eval}(\mathsf{k}_b, x), \mathsf{Prove}(\cdots)$
(4) $\mathcal{A}$ outputs b' given $(\mathsf{cm}_{\mathsf{nf}}, \mathsf{cm}_4, \pi)$

Additionally, the scheme maintains correctness, uniqueness (of the underlying nullifier value), and unforgeability as defined for deterministic nullifiers.

6.3 Construction

The rerandomizable nullifier scheme extends our deterministic construction by committing to the nullifier value. The key insight is an auxiliary commitment $\mathsf{cm}_4 = g_3 g^{r_3(\mathsf{k}+x)+r_4}$ that enables verification without revealing the underlying nullifier.

- $\mathsf{Eval}(\mathsf{k}, x) \to (\mathsf{cm}_{\mathsf{nf}}, \mathsf{cm}_4)$: Sample $r_3, r_4 \leftarrow \$\mathbb{Z}_p$, compute $\mathsf{cm}_{\mathsf{nf}} = g_3^{1/(\mathsf{k}+x)} g^{r_3}$ and $\mathsf{cm}_4 = g_3 g^{r_3(\mathsf{k}+x)+r_4}$, return $(\mathsf{cm}_{\mathsf{nf}}, \mathsf{cm}_4)$.
- $\mathsf{Prove}(\mathsf{pp}, \mathsf{k}, r_1, x, r_2, r_3, r_4) \to \pi$: Generate proof using Protocol 3.
- $\mathsf{Verify}(\mathsf{pp}, \mathsf{cm}_1, \mathsf{cm}_2, \mathsf{cm}_{\mathsf{nf}}, \mathsf{cm}_4, \pi) \to \{0, 1\}$: Verify proof per the protocol.

The auxiliary commitment cm_4 satisfies $\mathsf{cm}_4 = \mathsf{cm}_{\mathsf{nf}}^{\mathsf{k}+x} \cdot g^{r_4}$, enabling the verifier to check the algebraic relationship without learning k, x, or the underlying nullifier value.

6.4 Security Analysis

Theorem 6.2 (Anonymity). *The rerandomizable nullifier scheme satisfies anonymity under the hiding property of Pedersen commitments and the HVZK property of the Σ-protocol.*

Proof. The adversary's view consists of $(\mathsf{cm}_{\mathsf{nf}}, \mathsf{cm}_4, \pi)$ where:

- $\mathsf{cm}_{\mathsf{nf}} = g_3^{1/(\mathsf{k}_b+x)} g^{r_3}$ with fresh $r_3 \leftarrow \$\mathbb{Z}_p$ is uniformly distributed in $\mathbb{G}$

Figure 3: Σ-Protocol: Proof of Knowledge of a Committed Nullifier

Public parameters: prime-order $\mathbb{G}$, generators g_1, g_2, g_3, g

Common input: $cm_1, cm_2, cm_3, cm_4 \in \mathbb{G}$

Relation: $cm_1 = g_1^{sk} g^{r_1},\ cm_2 = g_2^{x} g^{r_2}$
$cm_3 = g_3^{1/(sk+x)} g^{r_3},\ cm_4 = g_3\, g^{r_3(sk+x)+r_4}$

Prover $(sk, x, r_1, \ldots, r_4)$ **Verifier**

pick $a_s, a_x, a_{r_1}, \ldots, a_{r_4} \leftarrow \mathbb{Z}_p$
$a_m := a_s + a_x, \quad a_\beta := 1/a_m$
$T_1 := g_1^{a_s}\, g^{a_{r_1}}$
$T_2 := g_2^{a_x}\, g^{a_{r_2}}$
$T_3 := g_3^{a_\beta}\, g^{a_{r_3}}$
$T_4 := cm_3^{a_m}\, g^{a_{r_4}}$
$\to (T_1, T_2, T_3, T_4)$

pick $c \leftarrow \mathbb{Z}_p$
$\leftarrow c$

$z_s := a_s + c\, sk$
$z_x := a_x + c\, x$
$z_{r_i} := a_{r_i} + c\, r_i \quad (i = 1, \ldots, 4)$
$z_m := a_m + c\,(sk + x)$
$z_\beta := a_\beta + c\, \frac{1}{sk+x}$
$\to (z_s, z_x, z_{r_1}, \ldots, z_{r_4}, z_m, z_\beta)$

Accept iff
$T_1\, cm_1^c = g_1^{z_s}\, g^{z_{r_1}}$
$T_2\, cm_2^c = g_2^{z_x}\, g^{z_{r_2}}$
$T_3\, cm_3^c = g_3^{z_\beta}\, g^{z_{r_3}}$
$T_4\, cm_4^c = cm_3^{z_m}\, g^{z_{r_4}}$
$z_m = z_s + z_x$

- $\mathsf{cm}_4 = g_3 g^{r_3(\mathsf{k}_b+x)+r_4}$ with fresh $r_4 \leftarrow \$\mathbb{Z}_p$ is uniformly distributed in $\mathbb{G}$
- The proof π reveals nothing beyond statement validity due to HVZK

Since all components are statistically or computationally independent of b, the adversary has negligible advantage. □

Theorem 6.3 (Correctness, Uniqueness, Unforgeability). *The rerandomizable nullifier inherits correctness, uniqueness (of the underlying value $g^{1/(\mathsf{k}+x)}$), and unforgeability from the binding property of commitments and soundness of Protocol 3.*

6.5 Application: Revocation with Privacy

The rerandomizable nullifier enables privacy-preserving revocation through set non-membership proofs. Given a revocation list $\mathcal{R} = \{\mathsf{nf}_1, \ldots, \mathsf{nf}_n\}$ of revoked nullifiers:

(1) User computes $(\mathsf{cm}_{\mathsf{nf}}, \mathsf{cm}_4) \leftarrow \mathsf{Eval}(\mathsf{k}, x)$
(2) User proves: (a) $\mathsf{cm}_{\mathsf{nf}}$ is correctly formed, and (b) the committed value is not in $\mathcal{R}$
(3) Verifier checks the proofs without learning the actual nullifier

This can be efficiently instantiated using range proofs or accumulator-based non-membership proofs, maintaining $O(\log n)$ complexity while preserving privacy. Each verification uses fresh randomness, preventing linkability across sessions while ensuring revoked credentials cannot be used.

7 Implementation & Evaluation

7.1 Pairing-Free DY VRF

Experimental setup. The pairing–free Dodis–Yampolskiy (DY) VRF and both nullifier variants were implemented in Rust using `arkworks-rs` v0.4.2. Benchmarks were executed on a MacBook Air M2 (AppleM2, 16 GiBRAM, macOS 13.4). Each figure is the arithmetic mean of 100 independent runs.

Results. Table 2 and figure 4 reports the end–to–end running time (*Eval+Prove* and *Verify*) of our prime–order, Pairing-Free DY VRF compared with the original pairing-based construction. Eliminating pairings yields a 3.1×–6.1× speed-up and enables deployment on mainstream curves such as Ed25519 and secp256k1. Even after applying a standard multi-pairing optimisation to the baseline, our scheme remains faster on BLS12-381.

Table 2: Runtime of original and pairing-free DY VRF (mean over 100 trials). Time in ms

Scheme	Curve	Eval+Prove	Verify	Total
PF DY(VRF)	Ed25519	0.21	0.37	0.58
	secp256k1	0.25	0.42	0.67
	BLS12-381	0.41	0.71	1.12
DY(VRF)*	BLS12-381	1.27	2.27	3.54

*Original DY VRF. Verification aggregates pairings in a single final exponentiation (2.27 ms instead of 2.85 ms for standard).

Our code is open source and available.

7.2 Nullifiers

Table 3 and figure 5 reports the end–to–end running time of our nullifier constructions compared with the most efficient construction from UTT's Anonymous Payments [31] Eliminating pairings yields a 3×–4.9× speed-up and enables deployment on mainstream curves such as Ed25519 and secp256k1.

7.3 Anonymous Credential Application

We benchmarked the credential wallet scenario discussed throughout the paper, a user with a credential generates a one-time action with a verifier input x. Our anonymous credential is based on a variant of PS Signatures [23] from [31]. During a privacy-preserving verification of an anonymous credential, the user rerandomizes their credential and commitment to then generate their proofs for verification. Our protocol fits within this procedure, using the rerandomized commitment, the user will generate a deterministic

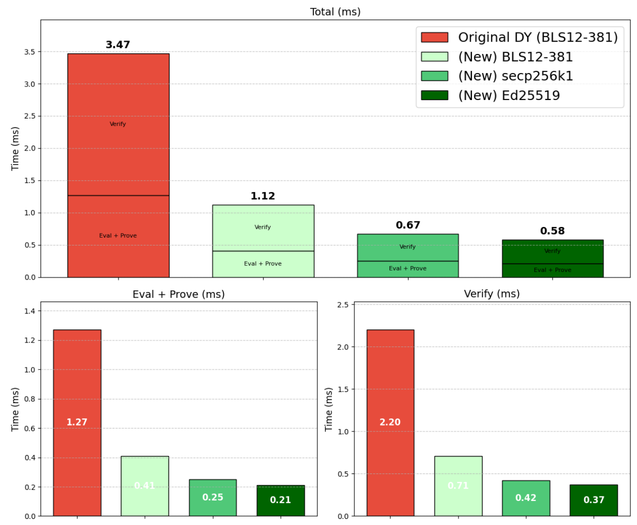

Figure 4: DY VRF Comparisons

Table 3: Runtime of nullifier constructions (mean of 100 runs, ms).

Scheme	Eval+Prove	Verify	Total
UTT (baseline)	5.91	6.47	12.38
Rerandomizable	2.01	2.01	4.02
Deterministic	1.08	1.41	2.49

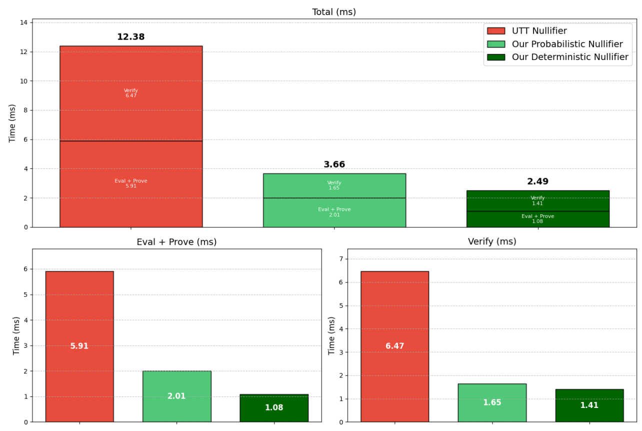

Figure 5: Nullifier Comparisons

or rerandomizable nullifier using our protocol and present the verification proof and nullifier together. A credential with 30 attributes was shown to conduct the Show + Verify algorithms (required in Anonymous Credentials) in just 5.38ms, adding our nullifier in either 2.49ms or 4.02ms is a negligible addition for greatly improved functionality.

8 Future Work

The integration with credential wallet relies on attribute-based anonymous credentials, which rely on elliptic curve pairings. While the scheme we presented above verifies in 5.8ms, using a non-pairing based signature such as ECDSA and Schnorr will likely reduce the signature computation. However, we must also avoid zkSNARKs, and therefore, verifying ECDSA signatures privately is expensive due to its dependency. Adaptor signatures [12] might provide an interesting pathway.

Acknowledgments

The authors acknowledge the use of generative AI tools ChatGPTo3, Claude Sonnet3.7, Grok3 for latex formatting, text revision, typo correction, code test generation and troubleshooting.

References

[1] AAMVA. 2025. Jurisdiction Data Maps - American Association of Motor Vehicle Administrators - AAMVA. https://www.aamva.org/jurisdiction-data-maps

[2] Man Ho Au, Willy Susilo, and Yi Mu. 2006. Constant-Size Dynamic k-TAA. *Security and Cryptography for Networks* 4116 (2006), 111–125. https://doi.org/10.1007/11832072_8

[3] Nir Bitansky. 2020. Verifiable Random Functions from Non-interactive Witness-Indistinguishable Proofs. *Journal of Cryptology* 33, 2 (April 2020), 459–493. https://doi.org/10/g9d725

[4] Gilles Brassard, David Chaum, and Claude Crépeau. 1988. Minimum disclosure proofs of knowledge. *J. Comput. System Sci.* 37, 2 (Oct. 1988), 156–189. https://doi.org/10.1016/0022-0000(88)90005-0

[5] Jan Camenisch and Anna Lysyanskaya. 2004. Signature Schemes and Anonymous Credentials from Bilinear Maps. *Advances in Cryptology – CRYPTO 2004* 3152 (2004), 56–72. https://doi.org/10.1007/978-3-540-28628-8_4

[6] Jan Camenisch and Victor Shoup. 2003. Practical Verifiable Encryption and Decryption of Discrete Logarithms. In *Advances in Cryptology - CRYPTO 2003*, Gerhard Goos, Juris Hartmanis, Jan Van Leeuwen, and Dan Boneh (Eds.), Vol. 2729. Springer Berlin Heidelberg, Berlin, Heidelberg, 126–144. https://doi.org/10.1007/978-3-540-45146-4_8 Series Title: Lecture Notes in Computer Science.

[7] Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. 2014. Malleable Signatures: New Definitions and Delegatable Anonymous Credentials. In *2014 IEEE 27th Computer Security Foundations Symposium*. IEEE, Vienna, 199–213. https://doi.org/10.1109/CSF.2014.22

[8] David L. Chaum. 1981. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM* 24, 2 (Feb. 1981), 84–90. https://doi.org/10.1145/358549.358563

[9] Elizabeth Crites, Aggelos Kiayias, and Amirreza Sarencheh. 2024. SyRA: Sybil-Resilient Anonymous Signatures with Applications to Decentralized Identity. https://eprint.iacr.org/2024/379 Publication info: Preprint..

[10] Distributed Lab - Noir. 2024. noir-plume/BENCHMARK.md at main · distributed-lab/noir-plume. https://github.com/distributed-lab/noir-plume/blob/main/BENCHMARK.md

[11] Yevgeniy Dodis and Aleksandr Yampolskiy. 2005. A Verifiable Random Function with Short Proofs and Keys. In *Public Key Cryptography - PKC 2005*, David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, and Serge Vaudenay (Eds.). Vol. 3386. Springer Berlin Heidelberg, Berlin, Heidelberg, 416–431. https://doi.org/10.1007/978-3-540-30580-4_28 Series Title: Lecture Notes in Computer Science.

[12] Andreas Erwig, Sebastian Faust, Kristina Hostáková, Monosij Maitra, and Siavash Riahi. 2021. Two-Party Adaptor Signatures from Identification Schemes. In *Public-Key Cryptography - PKC 2021 - 24th IACR International Conference on Practice and Theory of Public Key Cryptography, Virtual Event, May 10-13, 2021, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 12710)*, Juan A. Garay (Ed.). Springer, 451–480. https://doi.org/10.1007/978-3-030-75245-3_17

[13] European Parliament. 2024. MEPs to approve new scheme for an EU-wide digital wallet | 26-02-2024 | News | European Parliament. https://www.europarl.europa.eu/news/en/agenda/briefing/2024-02-26/14/meps-to-approve-new-scheme-for-an-eu-wide-digital-wallet

[14] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. 2019. Structure-Preserving Signatures on Equivalence Classes and Constant-Size Anonymous Credentials. *Journal of Cryptology* 32, 2 (April 2019), 498–546. https://doi.org/10.1007/s00145-018-9281-4

[15] Christina Garman, Matthew Green, and Ian Miers. 2014. Decentralized Anonymous Credentials. In *Proceedings 2014 Network and Distributed System Security Symposium*. Internet Society, San Diego, CA. https://doi.org/10/ggzspb

[16] Aayush Gupta and Kobi Gurkan. 2022. PLUME: An ECDSA Nullifier Scheme for Unique Pseudonymity within Zero Knowledge Proofs. https://doi.org/1721.1/147434

[17] identity foundation. 2025. *The BBS Signature Scheme Internet Draft.* Standard. https://identity.foundation/bbs-signature/draft-irtf-cfrg-bbs-signatures.html

[18] Bradley Malin and Latanya Sweeney. 2004. How (not) to protect genomic data privacy in a distributed network: using trail re-identification to evaluate and design anonymity protection systems. *Journal of Biomedical Informatics* 37, 3 (2004), 179–192. https://doi.org/10.1016/j.jbi.2004.04.005

[19] Deepak Maram, Harjasleen Malvai, Fan Zhang, Nerla Jean-Louis, Alexander Frolov, Tyler Kell, Tyrone Lobban, Christine Moy, Ari Juels, and Andrew Miller. 2021. CanDID: Can-Do Decentralized Identity with Legacy Compatibility, Sybil-Resistance, and Accountability. In *42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24-27 May 2021*. IEEE, 1348–1366. https://doi.org/10.1109/SP40001.2021.00038

[20] Microsoft. 2025. Introduction to Microsoft Entra Verified ID. https://learn.microsoft.com/en-us/entra/verified-id/decentralized-identifier-overview

[21] Unique Identification Authority of India. 2025. Vision To empower Aadhaar number holders of India with a unique identity and a digital platform to authenticate anytime, anywhere. https://uidai.gov.in/en/about-uidai/unique-identification-authority-of-india/vision-mission.html

[22] Alexey Pertsev, Roman Semenov, and Roman Storm. 2024. Tornado Cash Privacy Solution Version 1.4.

[23] David Pointcheval and Olivier Sanders. 2016. Short Randomizable Signatures. *Topics in Cryptology - CT-RSA 2016* 9610 (2016), 111–126. https://doi.org/10.1007/978-3-319-29485-8_7

[24] Reyhaneh Rabaninejad, Behzad Abdolmaleki, Sebastian Ramacher, Daniel Slamanig, and Antonis Michalas. 2024. Attribute-Based Threshold Issuance Anonymous Counting Tokens and Its Application to Sybil-Resistant Self-Sovereign Identity. , 1024 pages. https://eprint.iacr.org/2024/1024

[25] Deevashwer Rathee, Guru Vamsi Policharla, Tiancheng Xie, Ryan Cottone, and Dawn Song. 2023. ZEBRA: SNARK-based Anonymous Credentials for Practical, Private and Accountable On-chain Access Control. https://eprint.iacr.org/2022/1286

[26] Michael Rosenberg, Jacob D. White, Christina Garman, and Ian Miers. 2023. zk-creds: Flexible Anonymous Credentials from zkSNARKs and Existing Identity Infrastructure. In *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*. IEEE, 790–808. https://doi.org/10.1109/SP46215.2023.10179430

[27] Olivier Sanders and Jacques Traoré. 2024. Compact Issuer-Hiding Authentication, Application to Anonymous Credential. *Proc. Priv. Enhancing Technol.* 2024, 3 (2024), 645–658. https://doi.org/10.56553/POPETS-2024-0097

[28] Smart Nation Singapore. 2025. Factsheet – Singpass (Singapore's National Digital Identity). https://www.tech.gov.sg/files/products-and-services/Factsheet%20-%20Singpass%20(National%20Digital%20Identity).pdf

[29] Alan Stapelberg. 2025. t's now easier to prove age and identity with Google Wallet. https://blog.google/products/google-pay/google-wallet-age-identity-verifications/

[30] Latanya Sweeney, Akua Abu, and Julia Winn. 2013. Identifying Participants in the Personal Genome Project by Name (A Re-identification Experiment). https://doi.org/10.48550/ARXIV.1304.7605 Version Number: 1.

[31] Alin Tomescu, Adithya Bhat, Benny Applebaum, Ittai Abraham, Guy Gueta, Benny Pinkas, and Avishay Yanai. 2022. UTT: Decentralized Ecash with Accountable Privacy.

[32] Özlem Uzuner, Tawanda C. Sibanda, Yuan Luo, and Peter Szolovits. 2008. A de-identifier for medical discharge summaries. *Artificial Intelligence in Medicine* 42, 1 (Jan. 2008), 13–35. https://doi.org/10.1016/j.artmed.2007.10.001

[33] W3C, Grant Noble, Dave Longley, Daniel C. Burnett, Brent Zundel, and Kyle Den Hartog. 2025. Verifiable Credentials Data Model v2.0.

[34] W3C, Manu Sporny, Dave Longley, Markus Sabadello, Drummond Reed, Orie Steele, and Christopher Allen. 2022. Decentralized Identifiers (DIDs) v1.0.

[35] Ke Wang, Jianbo Gao, Qiao Wang, Jiashuo Zhang, Yue Li, Zhi Guan, and Zhong Chen. 2023. Hades: Practical Decentralized Identity with Full Accountability and Fine-grained Sybil-resistance. In *Annual Computer Security Applications Conference*. ACM, Austin TX USA, 216–228. https://doi.org/10.1145/3627106.

3627110
[36] z-cash. 2020. Nullifiers - The Orchard Book. https://zcash.github.io/orchard/design/nullifiers.html

A Different Scheme

Figure 6: Σ-Protocol for a Committed Nullifier (plaintext x)

Public parameters: $g_1, g_3, g \in \mathbb{G}$ prime-order

Common input: $x \in \mathbb{Z}_p,\ cm_1, cm_2, cm_3 \in \mathbb{G}$

Relation: $cm_1 = g_1^{sk} g^{r_1},\ \ cm_2 = g_3^{1/(sk+x)} g^{r_2},\ \ cm_3 = g_3\, g^{\,r_2(sk+x)+r_3}$

Prover (sk, r_1, r_2, r_3) **Verifier**

pick $a_s, a_{r_1}, a_{r_2}, a_{r_3} \leftarrow \mathbb{Z}_p$

$a_m := a_s + x, \quad a_\beta := 1/a_m$

$T_1 := g_1^{a_s}\, g^{a_{r_1}}$

$T_2 := g_3^{a_\beta}\, g^{a_{r_2}}$

$T_3 := cm_2^{a_m}\, g^{a_{r_3}}$

$\longrightarrow (T_1, T_2, T_3)$

pick $c \leftarrow \mathbb{Z}_p$

$\longleftarrow c$

$z_s := a_s + c\, sk$

$z_{r_1} := a_{r_1} + c\, r_1$

$z_\beta := a_\beta + c\, \frac{1}{sk+x}$

$z_{r_2} := a_{r_2} + c\, r_2$

$z_m := a_m + c\,(sk + x)$

$z_{r_3} := a_{r_3} + c\, r_3$

$\longrightarrow (z_s, z_{r_1}, z_\beta, z_{r_2}, z_m, z_{r_3})$

Accept iff

$$T_1\, cm_1^c = g_1^{z_s}\, g^{z_{r_1}}$$
$$T_2\, cm_2^c = g_3^{z_\beta}\, g^{z_{r_2}}$$
$$T_3\, cm_3^c = cm_2^{z_m}\, g^{z_{r_3}}$$
$$z_m = z_s + (1 + c)\, x$$