COD-SSI: Enforcing Mutual Privacy for Credential Oblivious Disclosure in Self Sovereign Identity

Elia Onofri[1][0000-0001-8391-2563], Andrea De Salve[2][0000-0003-1691-7182], Paolo Mori[3][0000-0002-6618-0388], Laura Emilia Maria Ricci[4][0000-0002-8179-8215], and Roberto Di Pietro[1][0000-0003-1909-0336]

[1] KAUST–CEMSE, Saudi Arabia
[2] CNR–ISASI, Italy
[3] CNR–IIT, Italy
[4] University of Pisa, Italy

Abstract. The Self-Sovereign Identity (SSI) paradigm is instrumental for decentralised identity management, allowing an entity to create, manage, and present their digital credentials without relying on centralised authorities. Credential Selective Disclosure (SD) is one of the most attractive privacy-preserving features of SSI, allowing users to partially reveal information from their credentials. However, current SD mechanisms primarily focus on protecting Holders' privacy, while offering limited protection to credentials Verifiers. Indeed, the specific information requested by a Verifier can inadvertently reveal sensitive information, including internal decision-making criteria, business rules, or strategic plans. In this work, we address this threat by proposing, to the best of our knowledge, the first approach enforcing mutual privacy in credential exchanges. To this end, we introduce COD-SSI (Credential Oblivious Disclosure for SSI), a novel framework that leverages Oblivious Pseudorandom Functions to allow credential selective access without revealing which specific claims were requested to the credential Holder. The security of our solution is formally verified and its feasibility is assessed through the experimental evaluation of our open-source prototype implementation, showing that provable mutual privacy in the context of SSI can be achieved with just moderate computational and communication overhead.

Keywords: Self Sovereign Identity · Selective Disclosure · Oblivious Key Derivation · Verifier Privacy.

1 Introduction

The Self Sovereign Identity (SSI) [36] paradigm has been recently introduced to give back to users the control of their identities and of the data paired to them,

This is the pre-copyedited, author-produced version of the paper accepted for publication in ESORICS 2026. Please refer to DOI:10.1007/978-3-032-38695-3_11.

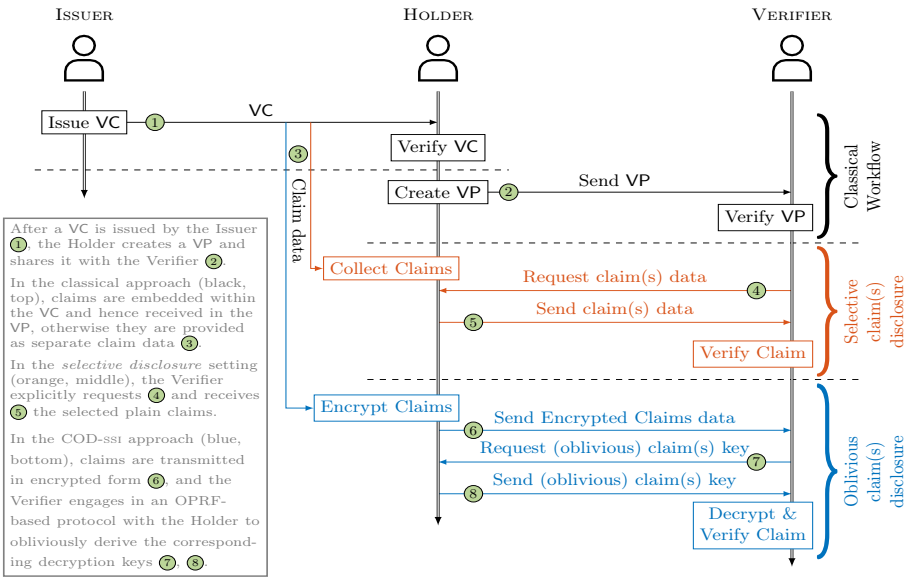

Fig. 1. End-to-end flow of credential issuance, presentation, and verification.

i.e., the claims describing users' features, such as, *e.g.*, their degree attestations, their vaccination certificates, or their proofs of income. In fact, in traditional centralised identity models, users' identities and all the related data are managed by centralised entities, called Identity Providers, which release such data to third parties. For instance, when you access a service using Google Login [15], your name, email address, and profile picture are always released to the service provider. Instead, in SSI, each user creates a *Decentralised IDentity* (DID, [32]) and stores identity-related data (claims, in the jargon) as *Verifiable Credentials* (VCs, [30]) issued by trusted entities. When interacting with third parties (Verifiers), users choose which VCs to disclose; Selective Disclosure (SD) techniques [7] further allow revealing only a subset of the claims within each VC. For instance, a student could disclose only their degree type without revealing marks or graduation date.

However, while SD protects Holder privacy, Verifiers may also have confidentiality requirements: the specific claims they request can reveal internal decision-making criteria or business rules. To the best of our knowledge, this gap is currently not addressed in the literature.

This work overcomes this limitation by introducing COD-SSI, a Credential Oblivious Disclosure solution for SSI that enhances the SD paradigm with *Credential Oblivious* [*Selective*] *Disclosure* (COD). In a nutshell, a Holder selects N claims they are willing to disclose, while the Verifier chooses $N_o < N$ among them, with the choice remaining oblivious to the Holder. Figure 1 illustrates the workflow.

1.1 Motivating Examples

The proposed approach finds application in multiple use cases where Verifiers' privacy is relevant, such as the two described below.

Financial risk assessment. A compelling application of COD-SSI arises in the domain of trade secrets, which are confidential information (such as process, formula, and algorithm) that provide a subject with economic and competitive advantages over competitors, and they can be protected by patent or copyright. For example, creditworthiness and loan evaluation provided by financial institutions are required to employ risk assessment models that rely on a limited number of applicant attributes to derive a credit score, such as a standard Employment Status credential and a Financial Status credential issued by the government. Suppose the Verifier always requests from the Financial Status credential the income and one additional risk indicator: debt-ratio (if the applicant's employment sector is considered stable) or delinquency count (if instead the employment sector is considered high-risk). If the Holder can observe which claim is requested (debt-ratio vs delinquency count), they can immediately infer the Verifier's internal classification of their employment profile. By colluding with other Holders with similar yet slightly different (say) employment credentials, an attacker could then reverse-engineer the Verifier's decision logic or scoring thresholds. This is not merely hypothetical: it mirrors well-documented *model-extraction* attacks, in which an adversary reconstructs a proprietary model through carefully chosen queries [37], and *model-inversion* attacks, which recover sensitive input features from observable outputs [13]. More generally, disclosing the exact subset of attributes used in the computation could reveal the structure of the risk model, thereby undermining the confidentiality of proprietary scoring methodologies that institutions are expected to safeguard under prudential and supervisory frameworks (cf. CRR [9] or Basel III and its Pillar 3 disclosure framework [1]). Furthermore, GDPR [8] recognises that the right to the protection of personal data must not affect the fundamental rights and freedoms of others, including trade secrets (Recital 63 of the GDPR). COD-SSI is a solution: the Holder makes available a set of attributes but releases only the ones actually requested by the Verifier for the risk assessment, while the Verifier's choice of which attributes to access remains hidden, enforcing both GDPR-aligned proportionality in personal data handling and further preserving the confidentiality of financial institutions' internal models.

Medical data access. A further application arises in access to Electronic Health Records (EHRs) for research. Medical data are subject to strict proportionality requirements (GDPR [8, Art. 9], HIPAA [34]), and researchers typically need only a limited subset of attributes (*e.g.*, laboratory results, diagnostic codes). Disclosing *which* fields are accessed might reveal, *e.g.*, the research focus, risking premature exposure of ongoing scientific work. COD-SSI might provide a balanced solution: it ensures that patients release no more than the attributes effectively queried, while the researchers' selection remains hidden, thus enforcing data minimisation without compromising research confidentiality.

1.2 Contributions

This paper provides the following contributions:

- We define COD-SSI, the first solution, to the best of our knowledge, to enforce Verifier privacy in SSI credential exchange, disclosure, and verification;
- We provide structured proofs demonstrating the security of our approach under an honest-but-curious client and a malicious server model and we discuss the extensibility to a model where both parties are malicious;
- We develop a Proof of Concept (PoC) implementing the proposed oblivious approach for claim disclosure (COD), to show COD-SSI feasibility;
- We conduct a set of experiments on the PoC to assess the introduced overhead, showing that provable mutual privacy in SSI can be achieved with just moderate computational and communication overhead; and,
- We release the source code for the practitioners and the scientific community to build up on our findings[5].

Roadmap. Section 2 reviews related work; Section 3 introduces the cryptographic building blocks; Section 4 presents the COD-SSI model; Section 5 analyses its security; Section 6 reports experimental results; and Section 7 concludes. An asymptotic complexity analysis and a symbol table are in Appendices A and B respectively.

2 Related Work

Research on privacy-preserving digital identity spans multiple domains, from decentralised identity management to advanced cryptographic techniques for credential protection. In this section, we frame our contribution within the existing literature by reviewing prior work on SSI, SD mechanisms, and the cryptographic primitives underlying modern identity systems. For more comprehensive overviews of SSI frameworks and their evolution, we refer the reader to [20]. We also refer the interested reader to [26] and [10], which highlight the interplay between SSI and distributed ledgers, cryptographic protocols, and privacy-enhancing technologies.

Selective Disclosure (SD). The W3C identifies SD as a fundamental privacy requirement for VCs, but does not specify a concrete method for its implementation [30]. As a result, different approaches have been proposed in the literature, each with distinct security and usability trade-offs. One of the simplest is the atomic credential approach [31,7], in which each claim is issued as a separate VC. Hash-based approaches [12,7] commit to claim values via hash functions, while Merkle-tree based methods [22,35] exploit hash-tree data structures to enable efficient proofs of inclusion. Cryptographic techniques such as attribute-based encryption [16] or threshold cryptography [29] have also been adopted to enforce fine-grained disclosure control. Also, signature schemes such as CL signatures [3] and BBS+ [33] provide built-in SD through zero-knowledge

[5] Publicly available at https://github.com/eOnofri04/COD-ssi.

proofs. Finally, regarding Zero-knowledge, zkSNARK-based constructions, such as zk-creds [28], were introduced to enable flexible anonymous credentials while introducing the notion of publicly verifiable credentials. However, unlike our proposed solution, all the above approaches still allow the Holder to know exactly which attributes are disclosed to the Verifier: in zero-knowledge-based SD the predicates on the claims that must be proved by the Holder reveal the internal decision-making logic of the Verifier, and even anonymous credential systems with hidden policies require the Holder to participate in policy-dependent proof generation. Closely related, it is worth mentioning the Private Information Retrieval (PIR) [6], which lets a client fetch a record without revealing its index. PIR, however, targets retrieval from a database held by a single server: it neither binds the retrieved item to an Issuer-certified credential nor bounds the number of items a Verifier may obtain, both of which are central to our setting. Also, except for the atomic solution (which lacks flexibility), all the cited approaches introduce communication and computational overheads, forcing all the parties to be compliant with changes in the protocols. Among these, hash-based solutions offer the best trade-off between deployment simplicity, computational efficiency, and claim-level granularity, which explains their wide adoption in current SSI frameworks (*e.g.*, SD-JWT [12]). Indeed, from the Verifier's perspective, a binary reveal/non-reveal model offers significant advantages in terms of simplicity, low computational overhead, and enhanced privacy protection.

Key Derivation, Encryption, and Oblivious Primitives. Secure key management is central to SSI deployments: Key Derivation Functions (KDFs), typically based on HMAC constructions such as HKDF [19], are widely used to derive unlinkable session keys, while Authenticated Encryption with Associated Data (AEAD) –notably AES-GCM [21]– provides the *de facto* standard for joint confidentiality and integrity [27]. Recent research extends KDFs to the *oblivious* setting, in which a client obtains a derived key without revealing its input to the server [5]. These constructions are closely related to Oblivious Pseudorandom Functions (OPRFs, [23]), whose discrete-log instantiation 2HashDH [18] achieves UC-secure evaluation under the one-more Diffie–Hellman assumption in the ROM [2]. Although our proof-of-concept adopts 2HashDH (a widely adopted real-world OPRF), the construction is independent of the chosen OPRF; post-quantum variants such as 2HashOPRF [2] can serve as drop-in replacements.

3 Preliminaries

This section introduces the building blocks underlying our construction.

Self-Sovereign Identity (SSI). In SSI, users create globally unique *Decentralised IDentifiers* (DIDs, [32]), each associated with a DID document containing the cryptographic material used to prove control and verify assertions. Attributes are granted by Issuers via *Verifiable Credentials* (VCs, [30]): a credential $\mathsf{VC} = (C_{\mathsf{VC}}, M_{\mathsf{VC}}, P_{\mathsf{VC}})$ consists of a claim–value dictionary $C_{\mathsf{VC}} = \{c_i : v_i\}$, metadata M_{VC}, and a proof P_{VC} obtained by signing $C_{\mathsf{VC}} \| M_{\mathsf{VC}}$ with the Issuer key. The Holder stores credentials locally and can create *Verifiable Presentations*

(VPs) by bundling a subset of VCs with additional metadata and signing them in a triple $\mathsf{VP} = (\mathsf{VC}[\cdot], M_{\mathsf{VP}}, P_{\mathsf{VP}})$.

Hash-based Selective Disclosure. In this work, we focus on the *hash-based approach*, as it provides a simple, commonly-implemented baseline later adaptable to our oblivious variant[6]. Here, claims v_i in C_{VC} are replaced by commitments $x_i = H(v_i \| t_i)$, for some hash function $H : \mathbb{F}_2^\star \to \mathcal{X}$ and some salt t_i, creating a claim-digest dictionary C^h_{VC}: the Issuer then delivers the actual data (pairs v_i, t_i) as complementary dictionary D_{VC} under a (possibly) separate secure channel (see later Figure 2 for a visual depiction); during disclosure, the Holder reveals the selected claims in plain, and the Verifier recomputes their digests to verify integrity in a commit-reveal fashion. Digests are typically generated under well-known schemes, but no specific implementation is typically enforced [11]: our choice falls on SHA-3 [24] due to its strong security properties, wide adoption, and versatility.

Pseudorandom Functions and Oblivious Evaluation. A *Pseudorandom Function* (PRF) is a keyed family $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ such that, for a random key $k \in \mathcal{K}$, $F_k(\cdot)$ is computationally indistinguishable from a truly random function. An *Oblivious PRF* (OPRF) [23,14] distributes the PRF evaluation between a client (holding $x \in \mathcal{X}$) and a server (holding $k \in \mathcal{K}$): the client obtains $F_k(x)$ while the server learns nothing about x and the client learns nothing about k.

We adopt the 2HashDH instantiation [18], which follows a blind–exponentiate–unblind paradigm. Given hash functions $H_1 : \mathcal{X} \to G$ and $H_2 : \mathcal{X} \times G \to \mathcal{Y}$: (i) the client samples r from G and sends $a = H_1(x)^r$; (ii) the server returns $b = a^k$; (iii) the client unblinds as $c = b^{1/r}$ and outputs $y = H_2(x, c)$. Security relies on the one-more Diffie–Hellman assumption in the ROM, yielding UC-secure evaluation with optimal round complexity [2].

Authenticated Encryption. We employ an AEAD scheme to protect claim values: for a key K, nonce IV, message M, and associated data A, encryption produces $\mathsf{Enc}_K(\mathsf{IV}, M, A) = (C, T)$ where C is the ciphertext and T is an authentication tag; decryption returns $\mathsf{Dec}_K(\mathsf{IV}, C, T, A) = (M \vee \perp)$. Security is captured by *IND-CPA confidentiality* (ciphertext indistinguishability) and *INT-CTXT integrity* (tag unforgeability). Our proof of concept uses AES-GCM, a commonly adopted solution (*e.g.* TLS 1.3, IPsec) which achieves both properties under standard assumptions provided nonces are never reused.

4 The COD-SSI Model

We are now ready to introduce COD-SSI, our novel approach allowing Verifiers to collect selectively disclosable claims in an oblivious framework. The methodology builds upon the three classes of primitives introduced in Section 3, enabling a Verifier to selectively extract a given number of claims from a VP in

[6] Other selective-disclosure mechanisms, such as the encryption-based construction, particularly fit our model. However, their comparatively limited adoption in current SSI frameworks reduces their immediate deployability.

a privacy-preserving way, while ensuring that all the undisclosed ones remain cryptographically protected and unlinkable.

Typically, SD is initiated by the Holder as a result of a specific service request. The expected interaction proceeds as follows:

1. the Holder selects the set of N claims the Holder is willing to expose within the context of that request;
2. the Verifier declares that access requires N_o claims from this admissible set, without revealing which ones;
3. the Holder agrees on the request and prepares a VP where each claim value is encrypted under a different key, where keys are derived via the OPRF from claims hash commitment; and
4. during verification, the Verifier engages in the OPRF protocol to recover only the N_o keys corresponding to its selected claims, while the Holder learns nothing beyond the number N_o of claims queried.

This design ensures that the Holder retains full control over which claims are eligible for verification while the Verifier's specific selection remains hidden. In particular, each claim is protected by a key derived through the OPRF, ensuring that even a malicious Verifier cannot learn or correlate non-disclosed claims, and that ciphertexts are indistinguishable from encryptions under fresh random keys. Compared to existing hash- or encryption-based SD methods, COD-SSI guarantees that every ciphertext is independently protected by a hidden, unlinkable key, while still allowing for efficient verification of the disclosed subset.

It is worth noting that COD-SSI also adapts to different interaction models, including when the Verifier provides a description of the required claims in advance, or when the Holder proactively offers a set of claims without waiting for a specific request. The first case is particularly relevant when the Verifier's requirements are fixed and known, as those requirements can be explicitly encoded in the presentation request and publicly certified by, *e.g.*, a supervisory authority, thus ensuring regulatory transparency and business confidentiality.

Our system follows the standard SSI roles: an Issuer I who issues credentials $\mathsf{VC} = (C^h_{\mathsf{VC}}, M_{\mathsf{VC}}, P_{\mathsf{VC}})$ with data $D_{\mathsf{VC}} = C_{\mathsf{VC}}$; a Holder S who receives and stores issued credentials, prepares a $\mathsf{VP} = (\mathsf{VC}[\cdot], M_{\mathsf{VP}}, P_{\mathsf{VP}})$ with encrypted presentation data D_{VP} as an array of encrypted claims $E_{\mathsf{VC}}[\cdot]$, and acts as the OPRF *server* during verification; and a Verifier V who acts as the OPRF *client* to obtain claim-specific keys. We assume that the Holder generates and maintains an OPRF secret key $\mathsf{msk}_{\mathsf{VP}} \in \mathcal{K}$ (or otherwise controls the server-side secret material for the OPRF instance associated with the presentation), and that the Verifier can interact with the Holder to run the OPRF protocol. The encrypted claim blobs E_{VC} are transported as part of $(\mathsf{VP}, D_{\mathsf{VP}})$ prior to any OPRF interaction. Crucially, the COD mechanism is executed *exclusively* between Holder and Verifier at presentation time – the Issuer's role is limited to standard credential issuance and plays no part in the oblivious interaction or verification time.

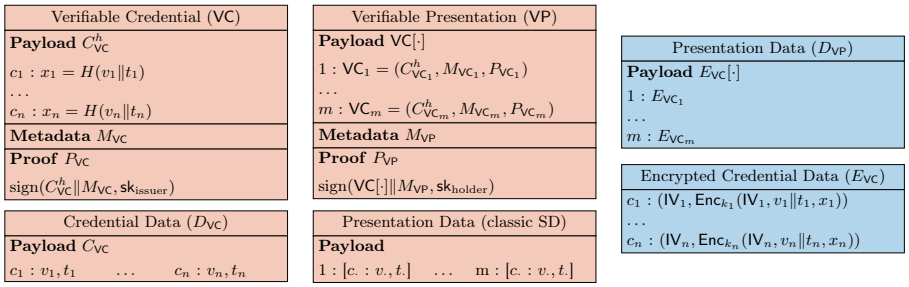

Fig. 2. Structure of the Verifiable Credential (VC) and associated data (D_{VC}) on the left, of Verifiable Presentation (VP) with associated data in the center, and of the encrypted presentation data (D_{VP}, substituting classic SD) on the right. Red boxes (left, center) represent the standard SD structure, while blue boxes (right) represent the additional encrypted material introduced by COD-SSI.

4.1 Creation of the Verifiable Presentation

The core idea of COD-SSI is to create a set of encrypted claims E_{VC} that can be safely shared with a Verifier during VP presentation. Upon generating a VP master key $\mathsf{msk}_{\mathsf{VP}}$, the Holder can use a KDF to derive, for each claim v_i, a claim key k_i from some claim-related identifier, like the hash $x_i = H(v_i \| t_i)$ in C^h_{VC}, which is already present in the VC itself. This way, confidentiality reduces to the problem of obliviously sharing claim keys, ensuring that undisclosed claims remain unlinkable and indistinguishable from random encryptions.

More formally, the Holder collects the SD VCs in a VP as usual, but additionally generates a fresh master secret key $\mathsf{msk}_{\mathsf{VP}} \in \mathcal{K}$ for the OPRF. For each claim $\{c_i : v_i\}$ in a credential, the Holder derives a claim-specific symmetric key $k_i = F_{\mathsf{msk}_{\mathsf{VP}}}(x_i)$, where $x_i = H(v_i \| t_i)$ is the hash commitment already present in C^h_{VC} and F denotes the OPRF (under 2HashDH: $k_i = H_2(x_i, H_1(x_i)^{\mathsf{msk}_{\mathsf{VP}}})$, see Section 3). Each claim is then AEAD-encrypted under k_i with a fresh nonce, using x_i as associated data, and the resulting ciphertext triples are collected in E_{VC} and embedded in the presentation data D_{VP}. The final verifiable presentation thus consists of the usual SD structure, augmented with $E_{\mathsf{VC}}[\cdot]$ (see Figure 2). The full procedure is formalised in Algorithm 1.

4.2 Verification of the Verifiable Presentation

Upon receiving a verifiable presentation $(\mathsf{VP}, D_{\mathsf{VP}})$, the Verifier validates the VP and the corresponding VCs proofs by checking that the VCs or VP conforms to the specification, that the Holder's signature on the VP is valid, that the Holder is the subject of the VCs, and whether they are expired or revoked. The Verifier then inspects the metadata, and checks the paths of the included VCs, as in the SD framework.

In contrast to the standard case, the Verifier does not request plaintext claims directly. Instead, for each claim of interest, it engages in the OPRF evaluation

Algorithm 1: VP Creation

Input: A set of SD credentials $\{\mathsf{VC}_j = (C^h_{\mathsf{VC}_j}, M_{\mathsf{VC}_j}, P_{\mathsf{VC}_j}, D_{\mathsf{VC}_j})\}_{j=0}^{m}$ to include in the VP.
Output: A $\mathsf{VP} = (\mathsf{VC}[\cdot], M_{\mathsf{VP}}, P_{\mathsf{VP}})$ and Enc. material $D_{\mathsf{VP}} = (E_{\mathsf{VC}}[\cdot])$.

// 1. Initialise the presentation
Sample a fresh $\mathsf{msk}_{\mathsf{VP}}$ from $\mathcal{K}$;
Initialise $D_{\mathsf{VP}} \leftarrow \emptyset$;
// 2. Process credential
foreach $j = 0, \ldots, m$ **do**
  Initialise $E_{\mathsf{VC}_j} \leftarrow \emptyset$;
  foreach $i = 0, \ldots, n_j$ **do**
    // 2a. Claim key derivation
    Retrieve c_i, v_i, t_i from D_{VC_j};
    $x_i = C^h_{\mathsf{VC}_j}[c_i]$;
    $k_i = H_2(x_i, H_1(x_i)^{\mathsf{msk}_{\mathsf{VP}}})$;
    // 2b. Enc. randomised claim
    Let $m_i = v_i \| t_i$;
    Sample fresh nonce IV_i;
    $(y_i, u_i) = \mathsf{Enc}_{k_i}(\mathsf{IV}_i, m_i, x_i)$;
    // 2c. Pack enc. material
    Add $(c_i : (\mathsf{IV}_i, y_i, u_i))$ to E_{VC_j};
  Add E_{VC_j} to D_{VP};
// 3. Finalise VP metadata and proofs
Construct presentation metadata M_{VP};
Sign presentation proof P_{VP};
return $\mathsf{VP} = (\mathsf{VC}[\cdot], M_{\mathsf{VP}}, P_{\mathsf{VP}})$, $D_{\mathsf{VP}} = (E_{\mathsf{VC}}[\cdot])$;

Algorithm 2: VP Verification

Input: A $\mathsf{VP} = (\mathsf{VC}[\cdot], M_{\mathsf{VP}}, P_{\mathsf{VP}})$, Enc. material $D_{\mathsf{VP}} = (E_{\mathsf{VC}}[\cdot])$, and quota N_o.
Output: Decrypted claims of interest.

// 1. Check consistency
Verifier validates P_{VP};
foreach $\mathsf{VC} = (C^h_{\mathsf{VC}}, M_{\mathsf{VC}}, P_{\mathsf{VC}})$ *in* $\mathsf{VC}[\cdot]$ **do**
  Verifier validates P_{VC};
// 2. Execute interaction with Holder
foreach $j = 0, \ldots, N_o - 1$ **do**
  Select VC_i from VP;
  Select a claim c in VC_i;
  Retrieve claim hash $x = C^h_{\mathsf{VC}_i}[c]$;
  Sample r from G;
  Send $a = H_1(x)^r$ to Holder;
  Receive $b = a^{\mathsf{msk}_{\mathsf{VP}}}$ from Holder;
  Evaluate $k = H_2(x, b^{1/r})$;
  Retrieve $(\mathsf{IV}, y, u) = E_{\mathsf{VC}_i}[c]$;
  Check correctness of tag u;
  Decrypt m_j from y under key k;
  Verify $H(m_j) = x$;
return $\{m_j\}_{j=0}^{N_o-1}$;

with the Holder, thereby obtaining the corresponding decryption keys k_i without revealing which claims were selected. This guarantees that the Verifier's choice remains oblivious, while the Holder retains the ability to enforce disclosure policies, such as limiting the number of claims that can be revealed by, *e.g.*, halting the interaction after a disclosure quota is reached (*quota exhaustion*).

The overall flow of COD-SSI, w.r.t. the classical and the SD framework, is summarised in Figure 1 (bottom, blue). The verification procedure is formalised in Algorithm 2.

5 COD-SSI security

This section analyses the security of our construction under the standard compositional methodology: since the employed primitives are proven secure in well-established frameworks, the overall scheme inherits these properties under black-box composition.

We first formalise the adversarial setting and the threat model, then present the security goals that our verifiable presentation protocol with oblivious SD aims to achieve. We subsequently state and proof-sketch the main security theo-

rem, derived using standard assumptions on the underlying primitives (a formal analysis is provided in the extended version [25]). Finally, we discuss the limitations of the current construction when internal parties become malicious, with particular attention to the vulnerability arising from a malicious Holder; we explain its implications and outline possible future directions to overcome it.

5.1 Threat Model and Adversarial Setting

Our protocol involves three internal roles – Issuer, Holder, Verifier – assumed to be *honest-but-curious* (HBC) unless stated otherwise. We additionally assume a malicious *external* adversary $\mathcal{A}$ with full network control (intercept, delay, inject), against which cryptographic security must hold via AEAD and OPRF guarantees. Parties can authenticate each other and bind messages to the relevant session (*e.g.*, via DID-bound keys), preventing impersonation and replay. The case where internal parties deviate from HBC is discussed in Section 5.3.

5.2 Security statements

We now formalise the security guarantees achieved by our construction and we provide a proof sketch.

Definition 1 (Security of Verifiable Presentations with Oblivious Selective Disclosure). *Let* $(\mathsf{VP}, D_{\mathsf{VP}})$ *be a verifiable presentation generated as described in Algorithm 1, containing* N *encrypted claims* $\{(x_i, \mathsf{IV}_i, y_i, u_i)\}_{i=1}^{N}$*. Our protocol for* VP *verification with oblivious SD is said to be* secure *if it achieves the following properties against Probabilistic Polynomial Time (PPT) adversaries:*

1. ***Confidentiality:*** *for any PPT adversary* $\mathcal{A}_V$ *that does not obtain the claim key* k_j *–for some index* j*– the ciphertext triple* $(\mathsf{IV}_j, y_j, u_j)$ *is computationally indistinguishable from an AEAD encryption of any equal-length message under a fresh uniform key. (Reduces to AEAD IND-CPA security and OPRF pseudorandomness; see also property 3a below.)*
2. ***Authenticity and integrity:*** *any modification of* $(\mathsf{IV}_i, y_i, u_i)$ *performed by a PPT adversary is detected by the Verifier with overwhelming probability. (Reduces to AEAD INT-CTXT security.)*
3. ***Obliviousness of disclosure (*dual guarantee*):***
   (a) Client privacy: *a PPT Verifier (OPRF client) learns* $F_{\mathsf{msk}_{\mathsf{VP}}}(x_i)$ *only for the inputs* x_i *the OPRF was explicitly queried on; for all other indices, the OPRF outputs are computationally indistinguishable from independent uniform values. (Reduces to OPRF pseudorandomness.)*
   (b) Server privacy: *a PPT Holder (OPRF server) gains no information on which subset of claim keys has been retrieved, beyond what is implied by policy-observable events, like, e.g., quota exhaustion. (Reduces to OPRF client-privacy.)*
4. ***Issuer verifiability:*** *the validity of each claim remains cryptographically verifiable against the Issuer's signature in the corresponding* VC*, as in the SD framework.*

Theorem 1 (Security of COD-SSI). *Let us assume that:*

1. *the underlying OPRF protocol (2HashDH) is UC-secure against malicious adversaries in the random oracle model (ROM);*
2. *the symmetric encryption protocol* Enc *(AES-GCM) provides IND-CPA confidentiality and INT-CTXT authenticity under standard assumptions; and,*
3. *the SD mechanism for* VC*s (SHA-3) is secure in the ROM.*

Then, under the operational assumptions stated in Section 5, the proposed verifiable presentation with oblivious SD, as described in Sections 4.1 and 4.2, satisfies Definition 1 against any PPT malicious external adversary and compliant (HBC) internal parties.

Proof (Proof sketch, see also [25]). Let us first recall that the 2HashDH is UC-secure in the ROM [2], AES-GCM provides IND-CPA and INT-CTXT security [21,17], and the SD based on SHA-3 is secure in the ROM [24]. We now proceed deriving the security properties required in Definition 1.

Obliviousness of disclosure (property 3). The dual guarantee follows from the UC-security of the 2HashDH OPRF. For part (a), the OPRF pseudorandomness ensures that the Verifier learns $F_{\mathsf{msk}_{\mathsf{VP}}}(x_i)$ only for the inputs x_i on which the OPRF was explicitly queried on; all other outputs are computationally indistinguishable from independent uniform values. For part (b), the OPRF client-privacy guarantee ensures that the Holder, acting as OPRF server, learns nothing about which inputs the Verifier queried.

Confidentiality and Integrity (properties 1–2). By property 3a (client privacy), for every unqueried index j the OPRF output $k_j = F_{\mathsf{msk}_{\mathsf{VP}}}(x_j)$ is computationally indistinguishable from a uniform random key. Given this, ciphertexts y_j are indistinguishable from random by the IND-CPA security of AES-GCM (property 1), and any forgery of (y_j, u_j) is detected with overwhelming probability by INT-CTXT tag verification (property 2), even in the presence of an active network adversary.

Issuer verifiability (property 4) is directly inherited from the SD framework: Issuer's signatures remain bound to the claim commitments in C^h_{VC}, and thus to the encrypted claims E_{VC}. Any modification to the credential will invalidate the issuer's signature.

Finally, since AEAD providing both IND-CPA and INT-CTXT properties realises the authenticated encryption functionality under UC composition [4], our use of AES-GCM inherits UC-style composability. Hence, by composition, the overall protocol satisfies Definition 1.

Remark 1. The dual guarantee (properties 3a and 3b of Definition 1) jointly captures the OT-like privacy structure of our COD protocol. A fully construction-independent formalisation of these properties as *Verifier-Privacy-Preserving Selective Disclosure* (VPP-SD) via indistinguishability games over an abstract VP scheme syntax is a natural direction for future work, and would allow the analysis of alternative instantiations beyond the OPRF-based design presented here. Concretely, the verifier-privacy game would pit a challenger against a malicious

Holder who, having prepared a presentation and observed the ensuing disclosure interaction, must distinguish which of two equal-size attribute subsets the Verifier actually selected. Verifier privacy then amounts to a negligible distinguishing advantage, *modulo* the policy-observable leakage isolated in Section 5.4. A symmetric game bounds the Verifier's advantage in learning unqueried claims, recovering client privacy and confidentiality.

5.3 Extending the Security to Malicious Actors

We now discuss the limitations of the construction when internal protocol participants deviate from the honest-but-curious assumption. As shown in the previous section, security against an active external adversary holds under standard assumptions on the underlying AEAD, OPRF, and SD mechanisms. We therefore focus on internal parties and show that: (i) a malicious Issuer has no meaningful impact during the verification phase; (ii) a malicious Verifier does not gain additional advantage beyond what is already captured by the security definitions; and (iii) the only non-trivial residual vulnerability arises when the Holder behaves maliciously during VP creation.

Malicious Issuer. The Issuer does not participate in VP verification; standard credential validation (signature, commitment checks) neutralises any malformed-credential vector.

Malicious Verifier. A malicious Verifier can deviate arbitrarily, yet, by OPRF security and AEAD IND-CPA + INT-CTXT, the attacker cannot learn any information about a claim for which the corresponding key k_j was not obtained (negligible distinguishing advantage, by reduction). Two practical caveats apply: (i) OPRF determinism creates linkability across VPs unless $\mathsf{msk}_{\mathsf{VP}}$ is chosen fresh; and (ii) Holder policy signals and implementation metadata can leak information and must be mitigated (*e.g.*, uniform error handling, padding, rate limits).

Malicious Holder. During verification, the Holder acts exclusively as the OPRF server; by UC-security of 2HashDH against malicious servers, any deviation either produces an output indistinguishable from the ideal functionality or causes a detectable abort, so Definition 1 is preserved. Moreover, from the Holder's perspective, the Verifier's OPRF input is pseudorandom; hence, no information is leaked through selective failures or adaptive invalid responses.

The critical limitation concerns VP *creation*: the security theorem requires honest generation per Algorithm 1. If the Holder manipulates ciphertexts or OPRF inputs, the resulting VP may not correspond to the Issuer-certified claims, enabling *selective-failure attacks* (see below).

5.4 Limitations and Mitigations Under Malicious Holder

As previously identified, the current construction does not cryptographically bind the tuple $(v_i, x_i, k_i, (\mathsf{IV}_i, y_i, u_i))$: a malicious Holder might encrypt a modified claim $v_i' \neq v_i$ or derive k_i from a manipulated input x_i', producing AEAD-valid ciphertexts that break semantic consistency with the VC. This enables a *selective-failure attack*: if the tampered claim is chosen for decryption the AEAD

check fails, revealing the Verifier's choice; if it is not chosen, the misconduct goes undetected and the Holder learns the claim was not selected.

Possible mitigations. Several approaches could enforce correct VP construction, though each introduces additional assumptions or overhead: (i) *Issuer-assisted VP generation*, where the Issuer signs commitments binding $(v_i, x_i, \mathsf{IV}_i)$, restoring soundness at the cost of Issuer involvement at presentation time; (ii) a *trusted VP-generation environment* (*e.g.*, a secure enclave), avoiding Issuer involvement but introducing a trusted-software assumption; or (iii) *zero-knowledge proofs of correct encryption*, achieving full cryptographic binding without additional trust assumptions, but at significant computational cost. These avenues remain outside the scope of the current proof-of-concept and present interesting directions for future work.

6 Experimental Results

To assess the practical performance of our framework, we conducted an experimental campaign. All benchmarks were executed on a Linux personal laptop equipped with an Intel Core i7-9750H CPU @ 2.60GHz, 16GB RAM. The PoC implementation was developed in JavaScript, ensuring reproducibility in a stable environment, and experiments were run with an idle machine.

Each experiment considered a VC with $n \in \{2, 4, 8, \ldots, 1024\}$ claims, where each claim was represented by a random hex string of 30 bytes. For every configuration, we performed 1000 repetitions of all the protocols, discarding the top and bottom 1% outliers to mitigate fluctuations and obtain statistically meaningful averages. The evaluation covered three main aspects:

1. **Claim level:** hashing, verification, encryption, decryption, and OPRF interaction time per claim.
2. **Credential level:** creation and verification times, together with the size of VCs and their metadata.
3. **Presentation level:** creation and verification time and size of VPs, the additional overhead of D_{VP}, and the cost of the oblivious interaction between Verifier and Holder.

Implementation details. The OPRF group G was set to the elliptic curve `secp256k1`, with the hash-to-group H_1 instantiated through the SHA-256-based hash-to-curve of the `noble` library[7]. The claim commitment is realised as a keyed hash $x_i = \mathsf{HMAC}_{t_i}(v_i)$ under SHA3-512 —rather than a plain concatenation— with a fresh 1024-bit salt t_i per claim, while the 2HashDH finaliser H_2 uses SHA-256. AES-GCM was used as the AEAD encryption scheme, with a 256-bit key and a fresh 96-bit initialisation vector (IV) per claim. For decentralised identity management, the implementation adopts the `did:ethr`[8] method based on a local Ethereum blockchain, while W3C verifiable credentials and presentations are

[7] https://github.com/paulmillr/noble-curves, v1.9.4.
[8] https://github.com/uport-project/ethr-did, v2.3.10.

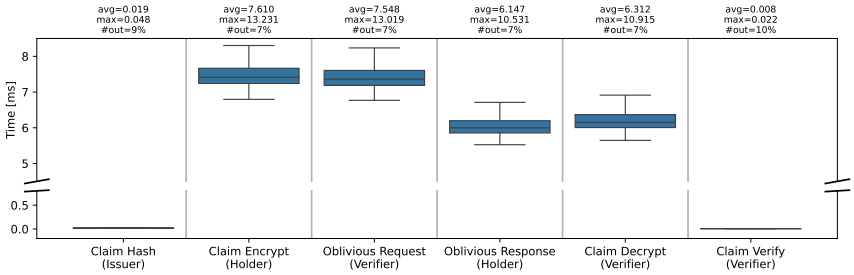

Fig. 3. Claim-wise statistics of the computational time for: claim hashing, verifying, encrypt, and decrypt, along with Verifier and Holder computation in the OPRF.

represented in JWT format[9]. Selective disclosure is built under SHA-3 according to the SD-JWT format [11].

6.1 Core claim-level results

Being independent of the number of claims n, claim-wise operations are aggregated and reported in Figure 3 along with their average (avg), maximum (max), and outlier fraction (#out). The cost of hashing and verifying a claim is negligible, with average runtimes below 0.02 ms. Both tasks reduce to computing a SHA-3 hash over the claim value combined with its nonce, hence they remain essentially constant across runs. By contrast, claim encryption (cf. ll.8–11 from Algorithm 1) and decryption (cf. ll.11–14 from Algorithm 2) dominate per-claim processing. On average, claim encryption performed by the Holder requires 7.6 ms due to the combination of hashing and elliptic-curve operations needed to derive the symmetric key. Claim decryption on the Verifier's side is instead slightly faster, averaging 6.3 ms. The OPRF evaluation phase shows timings of the same order of magnitude: generating an oblivious request (cf. ll.8–9 from Algorithm 2) takes 7.5 ms on the Verifier's side, while producing the corresponding oblivious response on the Holder's side (cf. l.10 from Algorithm 2) averages 6.1 ms.

6.2 Run-level results

A comparative analysis of the performance and size metrics required to present a classical SD VP with varying numbers of claims is presented in Figure 4, where timings (left scale) of the VP creation and verification are reported as boxplot, while sizes (right scale) of the VP are reported as bar plots. In every experimental configuration, the time required for both VP creation and verification is less than 100 ms, with verification being dominated by signature validation, whose cost significantly exceeds that of claim-wise hash extraction, resulting in nearly constant verification time. Instead, the size of the VP ranges from 3 kB (for VC

[9] https://github.com/decentralized-identity/did-jwt-vc, v3.1.4.

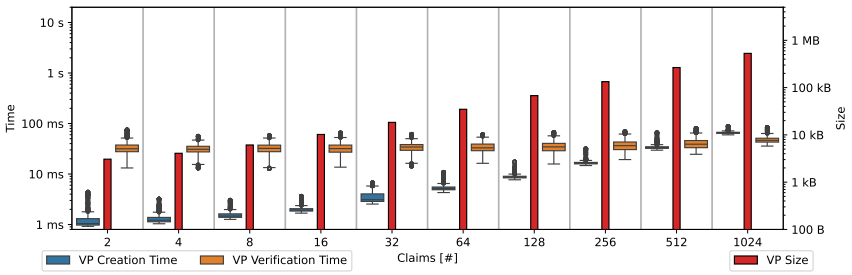

Fig. 4. Verifiable presentation (VP) statistics w.r.t. the claim size of their single VC. Timings (left scale) of creation and verification are reported as boxplot, while VP size (right scale) is reported as a bar plot since negligible variance is present.

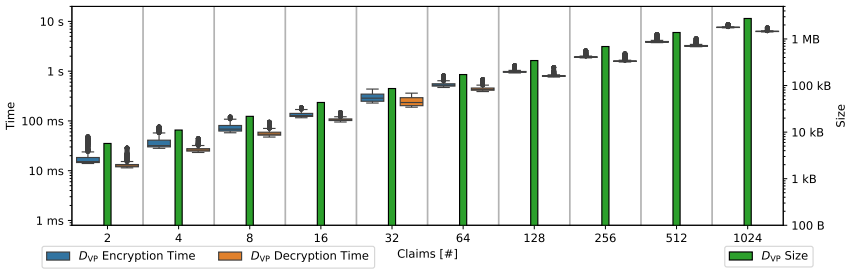

Fig. 5. Verifiable presentation overhead caused by data D_{VP} encryption and decryption w.r.t. the claim size of their single VC.

with 2 claims) to 0.53 MB (1024 claims), and it is comparable to the size of the embedded VC. It is worth noting that VP signature creation and verification times are common across all SD approaches, see also W3C specification [30].

Figure 5 shows the computational overhead introduced by COD-ssi in VP creation and verification due to the encrypted material adoption. Creation requires the Holder to generate a symmetric key for each claim and to encrypt the (claim, nonce) pair (D_{VP} Encryption Time) while decryption requires the Verifier to derive the symmetric key and decrypt all the claim values (D_{VP} Decryption Time). Both timings increase with the number of claims, ranging from 10 ms to 10 s, with Decryption being generally faster than Encryption: however, the two measures remain broadly comparable as both involve elliptic curve, hashing, and encryption/decryption operations. Do note that we present timings to decrypt all the claims within the VP to assess the scaling properties of the benchmark (which does not represent the actual usage for SD). Regarding the size of D_{VP}, it ranges from 6 kB (2 claims) to 2.5 MB (1024 claims), and it is significantly higher than the size of the corresponding VP, which is expected since D_{VP} includes the encrypted claims, the corresponding nonces, and authentication tags. Read together, Figures 4 and 5 constitute our baseline comparison: the former

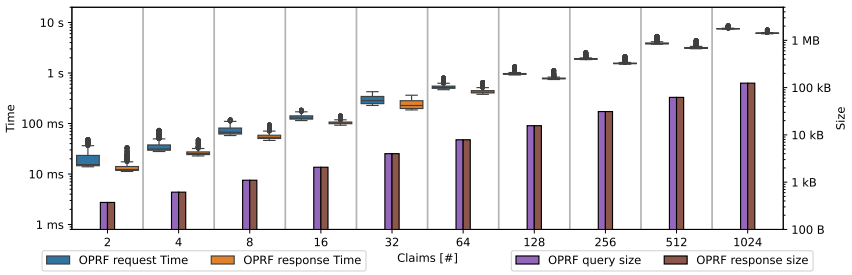

Fig. 6. Statistics of the OPRF interaction w.r.t. the claim size of their single VC.

measures a classical hash-based SD presentation, whilst the latter isolates the additional cost that COD-SSI introduces on top of it.

Figure 6 shows the OPRF interaction overhead where all the claims are requested from the VP. The Verifier's request time exceeds the Holder's response time due to the additional blinding and unblinding exponentiations (cf. Table 1), and both scale with the number of claims. Communication is symmetric: request and response sizes are comparable, reaching approximately 0.12 MB for 1024 claims. Compared to classical (non-oblivious) hash-based SD [7], the OPRF introduces significant computational overhead (elliptic-curve operations vs. plain hash exchanges), though the response size is actually smaller since it transmits group elements rather than plaintext claim values and nonces.

7 Conclusion

This paper presented COD-SSI, a solution for preserving Verifier privacy in the process of SSI VP disclosure and verification. To the best of our knowledge, this is the first solution enabling Verifiers to obtain a bounded number of claims while keeping private which subset of claims was actually requested.

By leveraging Oblivious Pseudorandom Functions, COD-SSI enables Holders to prepare Verifiable Presentations containing a large number of claims (say N) while allowing Verifiers to select and retrieve at most $N_o < N$ claims, without revealing the specific picks. This property is achieved without compromising the Holder's ability to enforce the disclosure bound, and without requiring any expensive cryptographic assumptions beyond well-studied OPRF constructions.

COD-SSI applies naturally to several realistic domains, including financial risk scoring and medical data access, where Verifier privacy is crucial. We formalised the security guarantees of the framework, implemented a prototype, and conducted a comprehensive experimental evaluation. The results show that our approach is efficient and practical: disclosure times range from a few milliseconds to under two seconds for VP containing up to 128 claims; storage requirements remain between a few kilobytes and roughly 1 MB for the same setting, values that do not materially affect user experience in typical SSI scenarios.

While filling a clear gap in the literature, our solution also opens important lines for further research: providing Verifier-privacy guarantees against malicious Holders via zero-knowledge proofs; extending COD-SSI to alternative SD paradigms (*e.g.*, encrypted credential schemes); integrating publicly auditable VP specifications for regulatory compliance; and formalising Verifier-Privacy-Preserving Selective Disclosure as a construction-independent security notion (cf. Remark 1) are only a few possibilities.

In conclusion, COD-SSI bridges a missing component in the SSI ecosystem by enabling Verifier-private SD of claims. The security of our framework is formally proved, and our prototype and experimental results confirm the feasibility and robustness of the approach, laying the groundwork for more expressive and robust privacy-preserving credential ecosystems.

Acknowledgments. This work was partially supported by SERICS (PE00000014) project under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Basel Committee on Banking Supervision: Pillar 3 disclosure requirements — consolidated and enhanced framework. Tech. Rep. BCBS 446, Bank for International Settlements (2018), https://www.bis.org/bcbs/publ/d446.htm
2. Beullens, W., Dodgson, L., Faller, S., Hesse, J.: The 2Hash OPRF Framework and Efficient Post-quantum Instantiations, pp. 332–362. Springer (2025). https://doi.org/10.1007/978-3-031-91101-9_12
3. Camenisch, J., Lysyanskaya, A.: Signature Schemes and Anonymous Credentials from Bilinear Maps, pp. 56–72. Springer (2004). https://doi.org/10.1007/978-3-540-28628-8_4
4. Canetti, R., Jain, P., Swanberg, M., Varia, M.: Universally Composable End-to-End Secure Messaging, pp. 3–33. Springer (2022). https://doi.org/10.1007/978-3-031-15979-4_1
5. Casacuberta, S., Hesse, J., Lehmann, A.: Sok: Oblivious pseudorandom functions. In: Proc. 7th IEEE EuroS&P. IEEE (2022). https://doi.org/10.1109/eurosp53844.2022.00045
6. Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private information retrieval. Journal of the ACM **45**(6) (1998). https://doi.org/10.1145/293347.293350
7. De Salve, A., Lisi, A., Mori, P., Ricci, L.: Selective disclosure in self-sovereign identity based on hashed values. In: Proc. IEEE ISCC 2022. IEEE (2022). https://doi.org/10.1109/ISCC55528.2022.9913052
8. European Parliament, Council of the European Union: Regulation (EU) 2016/679 of the European Parliament and of the Council (2016), https://gdpr-info.eu
9. European Parliament and the EU Council: Regulation (EU) No 575/2013 of the European Parliament and of the Council on prudential requirements for credit institutions and investment firms. Off. J. Eur. Union **L 176** (2013), https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32013R0575

10. Ferdous, M.S., Chowdhury, F., Alassafi, M.O.: In search of self-sovereign identity leveraging blockchain technology. IEEE Access **7** (2019). https://doi.org/10.1109/ACCESS.2019.2931173
11. Fett, D., Yasuda, K., Campbell, B.: Selective Disclosure for JSON Web Tokens. RFC 9901 (2025). https://doi.org/10.17487/RFC9901
12. Fett, D., Yasuda, K., Campbell, B.: Selective Disclosure for JWTs (SD-JWT). Internet-Draft draft-ietf-oauth-selective-disclosure-jwt-22, Internet Engineering Task Force (2025), https://datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/22/, work in Progress
13. Fredrikson, M., Jha, S., Ristenpart, T.: Model inversion attacks that exploit confidence information and basic countermeasures. In: Proc. 22nd ACM CCS. ACM (2015). https://doi.org/10.1145/2810103.2813677
14. Freedman, M.J., Ishai, Y., Pinkas, B., Reingold, O.: Keyword Search and Oblivious Pseudorandom Functions, pp. 303–324. Springer (2005). https://doi.org/10.1007/978-3-540-30576-7_17
15. Google: Google account help: How sign in with google helps you share data safely (2025), https://support.google.com/accounts/answer/12921417
16. Hernández-Ramos, J.L., Pérez, S., Hennebert, C., Bernabé, J.B., Denis, B., Macabies, A., Skarmeta, A.F.: Protecting personal data in iot platform scenarios through encryption-based selective disclosure. Computer Communications **130** (2018). https://doi.org/10.1016/j.comcom.2018.08.010
17. Iwata, T., Ohashi, K., Minematsu, K.: Breaking and Repairing GCM Security Proofs, pp. 31–49. Springer (2012). https://doi.org/10.1007/978-3-642-32009-5_3
18. Jarecki, S., Kiayias, A., Krawczyk, H.: Round-Optimal Password-Protected Secret Sharing and T-PAKE in the Password-Only Model, pp. 233–253. Springer (2014). https://doi.org/10.1007/978-3-662-45608-8_13
19. Krawczyk, H., Eronen, P.: HMAC-based Extract-and-Expand Key Derivation Function (HKDF). RFC 5869 (2010). https://doi.org/10.17487/RFC5869
20. Manimaran, P., Garrett, T., Jehl, L., Vitenberg, R.: Decentralization trends in identity management: From federated to self-sovereign identity management systems. Comput. Sci. Rev. **58** (2025). https://doi.org/10.1016/j.cosrev.2025.100776
21. McGrew, D., Viega, J.: The galois/counter mode of operation (gcm). NIST Modes of Operation Process (2004), https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-38d.pdf
22. Mukta, R., Martens, J., Paik, H.y., Lu, Q., Kanhere, S.S.: Blockchain-based verifiable credential sharing with selective disclosure. In: Proc. 19th IEEE TrustCom. IEEE (2020). https://doi.org/10.1109/TrustCom50675.2020.00128
23. Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudo-random functions. In: Proc. 38th IEEE FOCS. IEEE (1997). https://doi.org/10.1109/sfcs.1997.646134
24. NIST: Sha-3 standard: permutation-based hash and extendable-output functions. Tech. Rep. 202, NIST (U.S.) (2015). https://doi.org/10.6028/nist.fips.202
25. Onofri, E., De Salve, A., Mori, P., Ricci, L.E.M., Di Pietro, R.: COD-ssi: Enforcing mutual privacy for credential oblivious disclosure in self sovereign identity. arXiv preprint (2026), extended version. Available at arXiv:2604.10685
26. Pava-Díaz, R.A., Gil-Ruiz, J., López-Sarmiento, D.A.: Self-sovereign identity on the blockchain: contextual analysis and quantification of ssi principles implementation. Frontiers in Blockchain **7** (2024). https://doi.org/10.3389/fbloc.2024.1443362
27. Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446 (2018). https://doi.org/10.17487/RFC8446

28. Rosenberg, M., White, J., Garman, C., Miers, I.: zk-creds: Flexible anonymous credentials from zksnarks and existing identity infrastructure. In: Proc. 44th IEEE S&P. IEEE (2023). https://doi.org/10.1109/sp46215.2023.10179430
29. Sonnino, A., Al-Bassam, M., Bano, S., Meiklejohn, S., Danezis, G.: Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers. In: Proc. 26th NDSS. The Internet Society (2019). https://doi.org/10.14722/ndss.2019.23272
30. Sporny, M., Longley, D., Chadwick, D., Herman, I.: Verifiable credentials data model v2.0 (W3C recommendation 15 may 2025). Online https://www.w3.org/TR/vc-data-model-2.0/ (2025)
31. Sporny, M., Longley, D., Chadwick, D., Terbu, O., Zagidulin, D., Zundel, B.: Verifiable credentials implementation guidelines 1.0. Online https://www.w3.org/TR/vc-imp-guide/ (2019)
32. Sporny, M., Longley, D., Sabadello, M., Reed, D., Steele, O., Allen, C.: Decentralized identifiers (DIDs) v1.0 (W3C recommendation 19 july 2022). Online https://www.w3.org/TR/2022/REC-did-core-20220719/ (2022)
33. Tessaro, S., Zhu, C.: Revisiting bbs signatures. In: Advances in Cryptology – EUROCRYPT 2023. Springer (2023). https://doi.org/10.1007/978-3-031-30589-4_24
34. the Senate and House of Representatives of the United States of America: Health insurance portability and accountability act of 1996 (hipaa) (1996), https://www.govinfo.gov/content/pkg/PLAW-104publ191/pdf/PLAW-104publ191.pdf
35. Tian, R., Kong, L., Zhang, B., Li, X., Li, Q.: Authenticated selective disclosure of credentials in hybrid-storage blockchain. In: Proc. 28th IEEE ICPADS. IEEE (2023). https://doi.org/10.1109/ICPADS56603.2022.00050
36. Tobin, A., Reed, D.: The inevitable rise of self-sovereign identity. The Sovrin Foundation **29** (2016), https://sovrin.org/library/inevitable-rise-of-self-sovereign-identity/
37. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction APIs. In: Proc. 25th USENIX Security. USENIX Association (2016), https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_tramer.pdf

A Asymptotic Complexity

Table 1 summarises the asymptotic costs of COD-ssi. Both VP creation and disclosure scale linearly (in N and N_o respectively), with exponentiation in G dominating the cost. The Verifier's per-claim cost ($2\,\mathsf{Exp}$) is roughly twice the Holder's ($1\,\mathsf{Exp}$), consistent with the experimental results in Section 6.

B Notation

For the reader's convenience, in this appendix (Table 2) we provide a list of all symbols adopted within the manuscript.

Table 1. Asymptotic costs of COD-SSI per phase and party.

Phase	**Party**	**Group ops**	**Sym. ops**	**Comm.**
VP creation	Holder	$N\,(\mathsf{Exp} + H_1)$	$N(\mathsf{Enc} + H_2)$	—
VP validation	Verifier	$(m{+}1)\,\mathsf{SigV}$	—	—
OPRF discl.	Verifier	$N_o\,(2\,\mathsf{Exp} + H_1)$	$N_o(\mathsf{Dec} + H + H_2)$	$N_o\|G\|$
	Holder	$N_o\,\mathsf{Exp}$	—	$N_o\|G\|$
Storage (D_{VP})	—	—	—	$N\cdot(\|\mathsf{IV}\|+\|\mathsf{ct}\|+\|\mathsf{tag}\|)$

Exp: exponentiation in G; H_1: hash-to-group; H, H_2: standard hashes;
$\mathsf{Enc}, \mathsf{Dec}$: AEAD operations; SigV: signature verification.

Table 2. Table of symbols, divided by primitive

Variable	Definition	Meaning
$\mathsf{DID}_{\mathsf{name}}$	`did:<method>:<id>`	A DID controlled by name
$\mathsf{sk}_{\mathsf{name}}$	–	Secret key associated with $\mathsf{DID}_{\mathsf{name}}$
VC	$(C_{\mathsf{VC}}[C^h_{\mathsf{VC}}], M_{\mathsf{VC}}, P_{\mathsf{VC}})$	A [SD] Verifiable Credential
M_{VC}	–	The set of a VC metadata
P_{VC}	–	The proof of a VC (Issuer signature)
C_{VC}	$\{c_i : v_i\}$	The set of claims in a VC
c_i	`string`	A claim name / specialised path
v_i	`string`	A claim value
C^h_{VC}	$\{c_i : x_i\}$	The set of claim digests in a VC
D_{VC}	C_{VC}	The Credential Data, containing the plain claims
x_i	$H(v_i \Vert t_i)$	The digest of claim v_i
H	$\mathbb{F}_2^\star \to \mathcal{X}$	The SD ROM (SHA-3)
$\mathcal{X}$	–	Digest space (and Input space of the (O)PRF)
t_i	`nonce`	The token for the digest evaluation
VP	$(\mathsf{VC}[\cdot], M_{\mathsf{VP}}, P_{\mathsf{VP}})$	A Verifiable Presentation
$\mathsf{VC}[\cdot]$	–	A list of VCs (referring to the same DID)
M_{VP}	–	The set of a VP metadata
P_{VP}	–	The proof of a VP (Holder signature)
F, F_{msk}	$\mathcal{K} \times \mathcal{X} \to \mathcal{Y}$	The (O)PRF for KDF (2HashDH), the OPRF with key msk
$\mathcal{K}$	–	The key space of the (O)PRF
$\mathcal{Y}$	–	The output space of the (O)PRF (and key space for Enc)
$\mathsf{msk}_{\mathsf{VP}}$	$\in \mathcal{K}$	Master secret key of the VP to be used in F
H_1	$\mathcal{X} \to G$	The 2HashDH hash function mapping elements to G
H_2	$\mathcal{X} \times G \to \mathcal{Y}$	The 2HashDH hash function finalising the OPRF output
a, b, c, r	–	Internal values of 2HashDH scheme
Enc_K	$(\mathsf{IV}, M, A) \to (C, T)$	AEAD symmetric encryption (AES-GCM) with key K
Dec_K	$(\mathsf{IV}, C, T, A) \to (M \vee \perp)$	AEAD symmetric decryption (AES-GCM) with key K
K	–	Key for Enc, practically $k_i \in \mathcal{Y}$
IV	`nonce`	Random nonce for Enc, practically IV_i
M	–	Plaintext for Enc, practically v_i
A	–	Associated data for Enc, practically x_i (or None)
C	–	Ciphertext output of Enc, practically y_i, included in E_{VC}
T	–	Authentication tag of C, practically u_i, included in E_{VC}
D_{VP}	$E_{\mathsf{VC}}[\cdot]$	The VP associated data (encrypted material)
E_{VC}	$\{c_i : (\mathsf{IV}_i, y_i, u_i)\}$	The encrypted claims of a VC
(y, u)	$= \mathsf{Enc}_k(\mathsf{IV}, v \Vert t, x)$	The encryption of claim v
$\mathcal{A}$	–	Adversary
I, S, V	–	Issuer, Holder, Verifier