Verifiable Smart Delegation Links: A Theoretical Framework for Privacy-Preserving E-Government Delegation

Sarah Abdullah Almehmadi
Independent Researcher
sarahabdalmehmadi@gmail.com

***Abstract*—Citizens commonly share their government portal credentials with family members and trusted helpers when they need assistance completing online tasks. This practice exposes sensitive personal data to anyone who receives the shared password. We propose Verifiable Smart Delegation Links (VSDL), a theoretical framework for task-specific delegation. Rather than sharing full credentials, account owners generate signed URLs granting limited access to specific tasks. The framework introduces verifiable privacy enforcement: a cryptographic mechanism allowing delegates to verify that data filtering was performed correctly. This paper presents the conceptual design and formal security analysis as a foundation for future implementation and empirical evaluation. We discuss security properties, analyze limitations, and outline open research questions.**

***Index Terms*—delegation, access control, e-government, verifiable computation, privacy, digital identity**

I. Introduction

Government portals now provide essential services: identity document renewal, civil registry access, and benefits management. These systems assume each citizen manages their own account. However, many people need assistance, including elderly individuals, those with disabilities, and people who lack technical experience.

A. The Credential Sharing Problem

When citizens need help with government portals, they often share their login credentials. Studies have documented this practice across different populations [1]–[3]. The consequences are significant: helpers gain access to all account data, owners cannot restrict actions, and no audit trail exists.

Consider a typical situation: an elderly parent asks their adult child to help renew a national ID online. The parent shares their password. The child can then view financial records, property registrations, and medical details—information unrelated to ID renewal.

B. The Delegation Gap

Passwordless authentication methods such as passkeys improve security but complicate delegation [4]. Users cannot easily share biometric authentication or hardware tokens. Recent research shows that account recovery and delegation remain significant obstacles to FIDO2 adoption in both enterprise and consumer contexts [4].

The European Digital Identity Wallet initiative (eIDAS 2.0), published in April 2024, addresses identity management but focuses primarily on credential issuance rather than delegated access [5]. Similarly, work on selective disclosure in verifiable credentials [6] enables privacy-preserving attribute presentation but does not address the scenario where one person acts on behalf of another.

Existing delegation mechanisms do not fit this scenario. OAuth 2.0 [7] requires pre-registered applications. Capability URLs [8] provide access to entire resources rather than filtered views. Role-based access control [9] assumes organizational hierarchies.

C. The Server Trust Problem

Even systems that implement server-side data filtering face a fundamental question: how can anyone verify that the server filtered data correctly? A compromised or malicious server could violate filtering rules without detection.

D. Key Insight and Contributions

Our key insight is that combining cryptographic commitments with task-specific tokens enables delegates to verify correct filtering without knowing which fields were hidden or what values they contained. The server commits to the complete dataset at token creation; later, it proves that the filtered output and hidden fields together reconstruct the original commitment.

This paper presents a theoretical framework called Verifiable Smart Delegation Links (VSDL). The contributions are:

1) A conceptual design for task-specific delegation with field-level privacy control.
2) A cryptographic construction for verifiable privacy enforcement based on commitment schemes.
3) Formal definitions of security properties with analysis of when they hold.
4) Discussion of limitations and open problems.

This work focuses on theoretical foundations. We have not conducted empirical evaluation, performance benchmarking, or user studies. These remain important directions for future research.

II. Background and Related Work

A. Credential Sharing Practices

Research consistently finds that password sharing is common. Das et al. [1] found 43–51% of users reuse passwords

across sites. Gaw and Felten [2] documented sharing within families. Ion et al. [3] compared security practices between experts and non-experts.

B. Digital Identity Frameworks

The European Union's eIDAS 2.0 regulation [5], effective May 2024, establishes the European Digital Identity Wallet. This framework enables selective attribute disclosure but focuses on the holder presenting their own credentials rather than delegating access to others.

Recent work on verifiable credentials explores cryptographic mechanisms for selective disclosure [6], comparing approaches based on hiding commitments (such as ISO/IEC 18013-5) with those based on zero-knowledge proofs (such as BBS signatures). These mechanisms protect privacy during credential presentation but do not address delegated access scenarios.

C. Existing Delegation Mechanisms

Capability-based systems [8], [10] grant access through bearer tokens but typically provide access to entire resources.

OAuth 2.0 [7] enables delegated authorization for applications. Security analyses [11] have examined its properties. OAuth requires registered clients and redirect flows, making it unsuitable for ad-hoc human-to-human delegation.

Role-based access control [9] supports delegation through role hierarchies [12] but assumes organizational structure.

Attribute-based encryption [13], [14] enables fine-grained access control but requires key management infrastructure.

D. Verifiable Computation

Verifiable computation allows a prover to convince a verifier that a computation was performed correctly [15], [16]. Succinct arguments [17] make verification efficient, though proof generation remains expensive.

III. VSDL Framework

A. System Model

VSDL involves three parties:

- **Owner**: The citizen who holds the account.
- **Service Provider**: The government portal storing citizen data.
- **Delegate**: The person providing assistance.

B. Threat Model

We consider a service provider that may be honest-but-curious (follows the protocol but attempts to learn additional information) or malicious regarding data filtering (returns incorrect filtered data). Our security goal is to detect such misbehavior.

We explicitly exclude from our threat model: (1) collusion between delegate and server, where both parties cooperate to violate the owner's privacy; (2) physical coercion of the owner to create tokens against their will; (3) side-channel attacks that exploit timing, output size, or other observable properties; and (4) compromise of the owner's device or credentials. These represent important attack vectors but require different countermeasures and remain directions for future work.

C. Protocol Overview

The VSDL protocol proceeds as follows:

1) The owner authenticates and requests a delegation token for a specific task.
2) The portal generates a signed token containing a commitment to the owner's data.
3) The owner sends the token to the delegate.
4) The delegate presents the token to the portal.
5) The portal returns filtered data along with a proof of correct filtering.

D. Token Structure

VSDL tokens build on JSON Web Tokens [18]. A token contains:

- Standard claims: issuer, subject (hashed owner ID), expiration, unique identifier.
- Task identifier and permitted actions.
- Hash of the privacy policy.
- Cryptographic commitment to the owner's data.

E. Privacy Policy

Definition 1 (Privacy Policy)**.** *A privacy policy is a tuple* $P = (H, M, A)$ *where* H *is the set of fields to hide,* M *maps fields to masking rules, and* A *is the set of permitted actions.*

F. Data Model

Definition 2 (Data Record)**.** *A data record is a set of field-value pairs:* $D = \{(f_1, v_1), \ldots, (f_n, v_n)\}$.

Definition 3 (Filter Function)**.** *The filter function removes hidden fields and applies masking:*

$$\mathit{Filter}(D, P) = \{(f, v') : (f, v) \in D, f \notin H, v' = M(f)(v)\}$$

IV. Verifiable Privacy Enforcement

The core technical contribution is a mechanism for the server to prove that filtering was performed correctly.

A. Commitment Scheme

We use Pedersen commitments [19]. Let $\mathbb{G}$ be a group of prime order q with generators g and h where the discrete logarithm relationship is unknown.

Definition 4 (Field Commitment)**.** *For a field-value pair* (f, v) *with randomness* r*:*

$$C_{f,v} = g^{H(f \| v)} \cdot h^r$$

where H *is a collision-resistant hash function.*

Definition 5 (Record Commitment)**.** *For a data record* D*, the commitment is:*

$$C_D = \prod_{(f_i, v_i) \in D} C_{f_i, v_i}$$

Algorithm 1 Verification Procedure

Require: Commitment C_D, output F, randomness values $\{r_j\}$, proof π

Compute $C_F \leftarrow \prod_{(f_j,v_j)\in F} g^{H(f_j\|v_j)} \cdot h^{r_j}$
Verify: $C_D = \pi.C_H \cdot C_F$
if verification succeeds **then**
    return ACCEPT
else
    return REJECT
end if

B. Addressing the Randomness Problem

A key challenge is that verification requires knowledge of the randomness values used in commitments. We address this as follows:

When the server generates the initial commitment C_D at token creation, it also computes auxiliary information for later verification. Specifically, for each field (f_i, v_i) with randomness r_i, the server stores (f_i, r_i) securely.

When the delegate requests data, the server:

1) Filters the data according to policy P.
2) For each output field (f_j, v_j), includes the corresponding randomness r_j in the response.
3) Computes a commitment C_H to the hidden fields using their stored randomness values.
4) Provides proof that $C_D = C_H \cdot C_F$ where C_F is the commitment to filtered output.

The delegate can verify by:

1) Recomputing C_F from the received data and randomness values.
2) Checking that $C_D = C_H \cdot C_F$.

This approach requires the server to store randomness values, adding storage overhead. The randomness for output fields is revealed to delegates, which does not compromise security since the corresponding data is also revealed.

C. Verification Algorithm

Note that the delegate receives the policy hash (included in the token), not the full policy specification. Verification confirms that the received data and the hidden commitment together reconstruct the original data commitment. The delegate cannot independently verify *which* fields should be hidden—only that the server's claimed partition is consistent with the original commitment. This design choice avoids revealing the policy structure while still detecting server misbehavior.

D. Practical Example

Consider a delegation for ID renewal. The owner's record contains:

- Name: "Ahmed Ali" (visible)
- ID Number: "1234567890" (visible)
- Medical History: [data] (hidden)
- Bank Account: "SA..." (hidden)

At token creation, the server commits to all four fields. When the delegate accesses the data, they receive name and ID number along with their randomness values. The server provides C_H (commitment to medical and bank data) and the delegate verifies the partition is correct.

E. Complexity Analysis

Token generation requires $O(n)$ commitment operations where n is the number of fields in the owner's record. Each commitment involves one hash computation and two group exponentiations. Verification requires $O(k)$ operations to recompute the filtered data commitment (where k is the number of visible fields) plus $O(1)$ group operations to check the partition equation $C_D = C_H \cdot C_F$. Storage overhead is $O(n)$ for the randomness values that the server must retain.

V. SECURITY ANALYSIS

A. Security Definitions

Definition 6 (Data Minimization)**.** *Delegates learn only filtered data and nothing about hidden field values beyond their existence.*

Definition 7 (Task Confinement)**.** *Delegates can only perform actions listed in the token.*

Definition 8 (Filter Correctness)**.** *No server can produce a valid proof for incorrectly filtered data.*

B. Analysis

Theorem 1 (Data Minimization)**.** *Under the discrete logarithm assumption, the commitment C_H reveals no information about hidden field values.*

Argument sketch: The hiding property of Pedersen commitments ensures that C_H is computationally indistinguishable from a commitment to any other values. Without knowledge of the discrete logarithm between g and h, an adversary cannot extract information about the committed values.

Theorem 2 (Filter Correctness)**.** *Under the binding property of Pedersen commitments, a server cannot produce a valid proof for incorrect output.*

Argument sketch: Producing a valid proof for different output would require finding C'_H such that $C_D = C'_H \cdot C'_F$ where $C'_F \neq C_F$. This implies finding two different openings of C_D, which contradicts the binding property under the discrete logarithm assumption.

These are informal arguments. Rigorous proofs would require formal definitions and reduction arguments, which we leave for future work.

C. Limitations

Link forwarding. Tokens are bearer credentials. Anyone who obtains a token can use it.

No delegate authentication. The framework does not verify delegate identity. Binding tokens to specific delegates would require additional mechanisms [20].

Server availability. A malicious server can refuse service.

Side channels. Output size or timing might leak information about hidden fields.

Coercion. An attacker could pressure an owner to create a token.

Storage overhead. The server must store randomness values for all fields.

Usability. We have not studied whether users can understand and correctly use delegation links.

VI. Discussion

A. Relationship to Existing Work

VSDL combines ideas from capability systems, verifiable computation, and commitment schemes. Unlike OAuth, VSDL does not require registered applications. Unlike capability URLs, VSDL provides field-level filtering. Unlike attribute-based encryption, VSDL does not require client-side key management.

The selective disclosure mechanisms in verifiable credentials [6] address a related but different problem: they enable a credential holder to selectively reveal attributes when presenting their own credentials. VSDL addresses delegation, where one person accesses another's data.

B. Deployment Considerations

Deployment would require standardizing task definitions and privacy policies, managing cryptographic parameters, designing understandable user interfaces, and addressing liability questions.

C. What This Paper Does Not Claim

To be explicit:

- We have not built a production system.
- We have not measured performance.
- We have not conducted user studies.
- We have not provided formal security proofs.
- We do not claim this solves all delegation problems.

VII. Future Directions

Implementation and benchmarking. Building a prototype would reveal practical constraints.

User studies. Evaluating usability would inform interface design.

Formal verification. Machine-checked proofs would strengthen security guarantees.

Delegate binding. Integrating proof-of-possession mechanisms remains open.

Alternative constructions. BBS signatures or other selective disclosure primitives might offer different trade-offs.

VIII. Conclusion

Credential sharing for delegation is widespread despite security risks. Existing mechanisms do not adequately address human-to-human delegation in consumer services.

This paper presents Verifiable Smart Delegation Links, a theoretical framework combining task-specific access with cryptographic verification of correct filtering. We have described the design, presented a construction based on Pedersen commitments, addressed the randomness verification problem, and analyzed security properties informally.

Important limitations remain, particularly around usability, formal proofs, and empirical validation. We offer this framework as a foundation for future research on delegation in authentication systems.

References

[1] A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang, "The tangled web of password reuse," in *Proc. Network and Distributed System Security Symposium (NDSS)*, 2014.

[2] S. Gaw and E. W. Felten, "Password management strategies for online accounts," in *Proc. Symposium on Usable Privacy and Security (SOUPS)*, 2006, pp. 44–55.

[3] I. Ion, R. Reeder, and S. Consolvo, "...No one can hack my mind: Comparing expert and non-expert security practices," in *Proc. Symposium on Usable Privacy and Security (SOUPS)*, 2015, pp. 327–346.

[4] L. Lassak, E. Pan, B. Ur, and M. Golla, "Why aren't we using passkeys? Obstacles companies face deploying FIDO2 passwordless authentication," in *Proc. USENIX Security Symposium*, 2024, pp. 7231–7248.

[5] European Parliament and Council, "Regulation (EU) 2024/1183 amending Regulation (EU) No 910/2014 as regards establishing the European Digital Identity Framework," *Official Journal of the European Union*, April 2024.

[6] A. Flamini, S. Ferretti, G. Lenzini, and M. Marchetti, "On cryptographic mechanisms for the selective disclosure of verifiable credentials," *Journal of Information Security and Applications*, vol. 83, 2024.

[7] D. Hardt, "The OAuth 2.0 authorization framework," RFC 6749, Internet Engineering Task Force, 2012.

[8] M. S. Miller, K.-P. Yee, and J. Shapiro, "Capability myths demolished," Tech. Rep., Johns Hopkins University, 2003.

[9] D. Ferraiolo, D. Kuhn, and R. Chandramouli, *Role-Based Access Control*, 2nd ed. Artech House, 2007.

[10] J. B. Dennis and E. C. Van Horn, "Programming semantics for multiprogrammed computations," *Communications of the ACM*, vol. 9, no. 3, pp. 143–155, 1966.

[11] D. Fett, R. Küsters, and G. Schmitz, "A comprehensive formal security analysis of OAuth 2.0," in *Proc. ACM Conference on Computer and Communications Security (CCS)*, 2016, pp. 1204–1215.

[12] E. Barka and R. Sandhu, "Framework for role-based delegation models," in *Proc. Annual Computer Security Applications Conference (ACSAC)*, 2000, pp. 168–176.

[13] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. ACM Conference on Computer and Communications Security (CCS)*, 2006, pp. 89–98.

[14] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symposium on Security and Privacy*, 2007, pp. 321–334.

[15] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Proc. International Cryptology Conference (CRYPTO)*, 2010, pp. 465–482.

[16] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," *Communications of the ACM*, vol. 59, no. 2, pp. 103–112, 2016.

[17] J. Groth, "On the size of pairing-based non-interactive arguments," in *Proc. International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, 2016, pp. 305–326.

[18] M. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)," RFC 7519, Internet Engineering Task Force, 2015.

[19] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Proc. International Cryptology Conference (CRYPTO)*, 1991, pp. 129–140.

[20] D. Fett, B. Campbell, J. Bradley, T. Lodderstedt, M. Jones, and D. Waite, "OAuth 2.0 Demonstrating Proof of Possession (DPoP)," RFC 9449, Internet Engineering Task Force, 2023.