Content-Addressed Signature Recovery via Reverse Lookup Registry

Ramprasad Anandam Gaddam
Vouch Protocol Project
https://vouch-protocol.com
ram@vouch-protocol.com

January 2026

Abstract

This paper presents a method for recovering cryptographic signatures when they are stored separately from the content they authenticate. We introduce a reverse lookup registry that indexes signatures by content hash, enabling verification even when signatures were not transmitted alongside content. This addresses the common scenario where content undergoes transformations that strip metadata, such as social media uploads, format conversions, or screenshots. We formalize the registry data model, specify the lookup protocol, and analyze scalability and security properties.

This paper is part of the Vouch Protocol Defensive Disclosure Series. Full index: https://vouch-protocol.com/docs/disclosures/

Keywords: content addressing, hash-based lookup, detached signatures, signature recovery, registry

1 Introduction

Detached signatures, where the signature is stored separately from the signed content, create a discoverability problem. When content is shared without its accompanying signature—due to metadata stripping, format conversion, or transmission through channels that don't preserve attachments—the recipient has no way to know that a signature exists or where to find it.

We propose a reverse lookup registry: a service that indexes signatures by the cryptographic hash of the content they sign. Given arbitrary content, a verifier can compute its hash and query the registry to discover any registered signatures.

1.1 Contributions

- Formal model for content-addressed signature storage
- Registry protocol specification
- Scalability analysis and sharding strategies
- Security considerations for registry trust

2 Background

2.1 Cryptographic Hash Functions

Definition 1 (Cryptographic Hash). *A cryptographic hash function $H : \{0,1\}^* \to \{0,1\}^n$ maps arbitrary-length input to fixed-length output with properties:*

- *Collision resistance: Infeasible to find $x \neq y$ where $H(x) = H(y)$*
- *Preimage resistance: Given h, infeasible to find x where $H(x) = h$*

We use SHA-256 for content addressing.

2.2 Content Addressing

Definition 2 (Content Address). *A content address is the cryptographic hash of content, serving as a unique identifier:*

$$\mathsf{addr}(c) = H(c) \tag{1}$$

Content addressing enables location-independent content identification, used in systems like IPFS and Git.

3 Reverse Lookup Registry

3.1 Data Model

Definition 3 (Registry Record). *A registry record $R = (\mathsf{hash}, \mathsf{sig}, \mathsf{signer}, \mathsf{timestamp})$ maps:*

- hash*: SHA-256 hash of the signed content*
- sig*: The detached signature (or URL to signature)*
- signer*: DID of the signing entity*
- timestamp*: Registration time*

3.2 Registry Record Format

```
{
  "content_hash": "sha256:a3f2b7c9d8e4f1a2...",
  "content_type": "image/jpeg",
  "signer_did": "did:web:alice.example.com",
  "signature": "eyJhbGciOiJFZERTQSJ9...",
  "registered_at": "2026-01-10T12:00:00Z"
}
```

3.3 Registration Protocol

To register a signature:

1. Signer computes $h = H(c)$ for content c
2. Signer creates signature $\sigma = \mathsf{Sign}(sk, c)$
3. Signer submits $(h, \sigma, \mathsf{DID})$ to registry
4. Registry stores mapping $h \mapsto (\sigma, \mathsf{DID}, t)$

3.4 Lookup Protocol

To discover signatures for content:

1. Verifier computes $h = H(c)$
2. Verifier queries: GET /lookup?hash $=h$
3. Registry returns all matching records
4. Verifier fetches signer public keys and verifies signatures

4 Scalability Analysis

4.1 Storage Requirements

Each record requires approximately 500 bytes. At 1 billion records:

$$\text{Storage} \approx 500 \text{ GB} \tag{2}$$

4.2 Sharding Strategy

The registry can be sharded by hash prefix:

$$\text{shard}(h) = h[0:k] \mod n \tag{3}$$

where k is prefix length and n is shard count.

4.3 Query Performance

Hash-based lookup is $O(1)$ with proper indexing. Bloom filters can accelerate negative lookups.

5 Security Considerations

Property 1 (Registry Non-Authority)**.** *The registry does not validate signatures; it only stores them. Verifiers perform cryptographic validation independently.*

Property 2 (Multiple Signatures)**.** *The same content hash may have multiple registered signatures from different signers, representing multiple attestations.*

Threat: Spam registration. Mitigation: Rate limiting, proof-of-work, or staking.

Threat: False claims. Mitigation: Verifiers independently validate signatures against signer public keys.

6 Related Work

IPFS uses content addressing for distributed storage; our work applies content addressing to signature indexing.

Certificate Transparency logs certificate issuance for public auditing; our registry provides signature discoverability.

Keybase provided identity proofs with signature storage; our approach is decentralized and protocol-agnostic.

7 Implementation

Reference implementation available at https://github.com/vouch-protocol/vouch.

The registry is implemented as:

- Cloudflare Worker with KV storage for low-latency global access
- REST API with hash-based lookup
- Integration with Smart Scan browser extension (PAD-004)

8 Conclusion

This paper presented a reverse lookup registry for content-addressed signature recovery. By indexing signatures by content hash, the registry enables verification of content even when signatures were not transmitted alongside it. The approach is scalable through sharding and maintains security through verifier-side cryptographic validation.

Prior Art Declaration

This document is published as a defensive prior art disclosure under the Creative Commons CC0 1.0 Universal Public Domain Dedication. The methods and systems described herein are hereby released into the public domain to prevent patent monopolization.
Any party implementing similar functionality after the publication date of this document cannot claim novelty for patent purposes.
Reference Implementation: https://github.com/vouch-protocol/vouch

Cryptographically Signed Document

Signed by: Ramprasad Anandam Gaddam (`github:rampyg`)
Verify: https://v.vouch-protocol.com/p/pad005

This document's authenticity can be verified by computing its SHA-256 hash and checking against the signature registered at the verification URL above.

References