Concise Encoding of Signed Merkle Tree Proofs

draft-steele-cose-merkle-tree-proofs

Orie Steele
IETF 116, Yokohama
March 27, 2023

What Does It Do?

- Describes merkle proof data structures in CBOR
- Addresses the challenge of “merkle tree agility”
- Enables COSE Sign1 to act as a kind of counter signature over an inclusion proof for a payload
- Provides COSE building blocks for transparency logs, and other verifiable data structures that build on merkle proofs.

Why Do It?

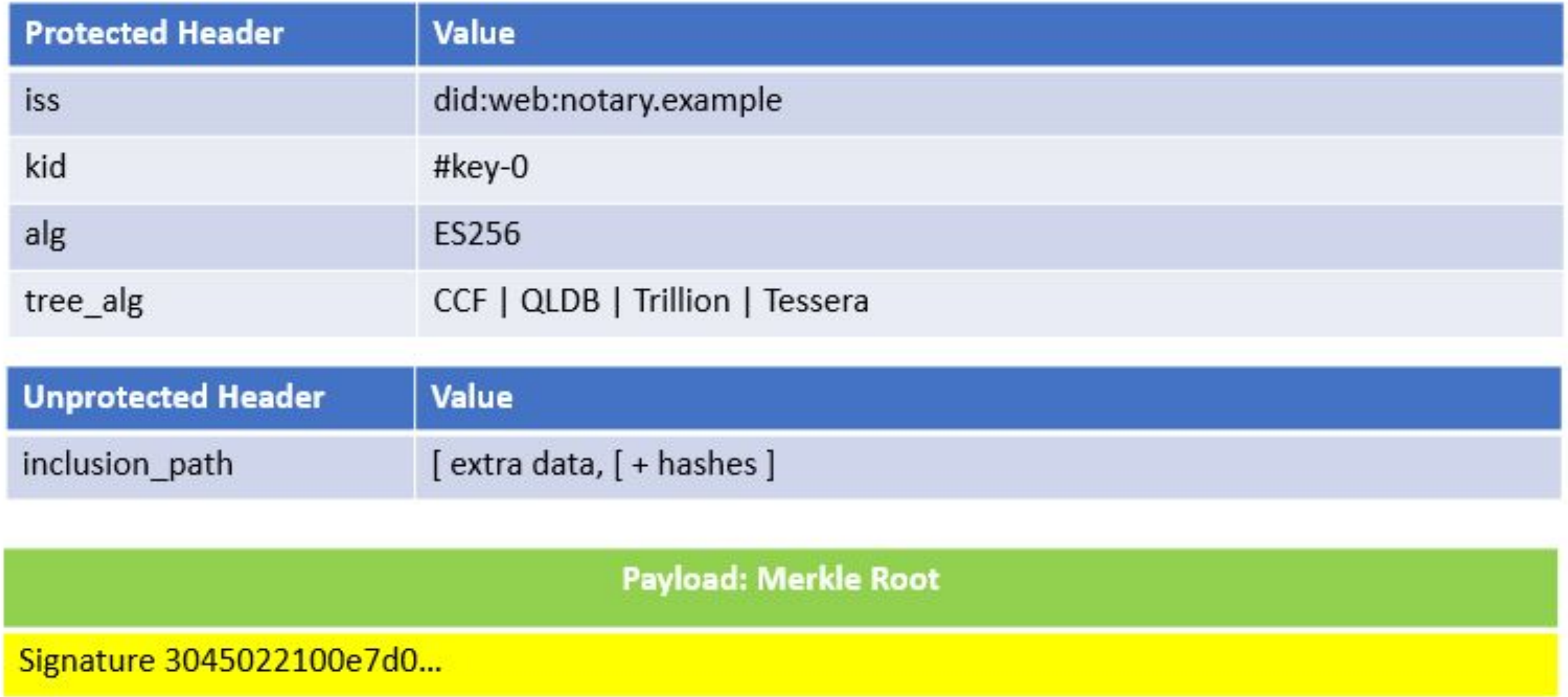
- Establishes interoperability across various verifiable data systems:
  - CBOR inclusion proofs are compact
  - COSE signatures over inclusion proofs enable offline verification
  - A useful building block for SCITT and other COSE oriented WGs
  - The more people can verify inclusion proofs, the more robust transparency
  - There are other transparency use cases, such as “key transparency” & “certificate transparency”.

Status

- Recently published -00:
  - Need to address “merkle tree agility”
  - Terminology needs tightening
  - Need to address “various proof encodings”
  - Need to improve CDDL examples

Application Example:

SCITT Receipt



Protected Header	Value
iss	did:web:notary.example
kid	#key-0
alg	ES256
tree_alg	CCF | QLDB | Trillion | Tessera

Unprotected Header	Value
inclusion_path	[extra data, [+ hashes]

Payload: Merkle Root

Signature 3045022100e7d0…

We hope to establish a registry for tree algorithms.

Next Steps

- How should we handle tree agility:
  - Registry / vanilla algorithms / vendor algorithms

- We think the tree agility issue should be solved before a call for adoption.