Concise Encoding of Signed Merkle Tree Proofs (CoMETRE)

draft-steele-cose-merkle-tree-proofs

O. Steele, H. Birkholz, A. Delignat-Lavaud, C. Fournet

IETF 117, San Francisco
July 24, 2023
What Does It Do?

- Describes verifiable data structures in CBOR, encoding various proof types for binary merkle trees, merkle search trees, cryptographic accumulators, etc…

- Provides COSE building blocks for transparency logs, and other verifiable data structures.
Why Do It?

● Establishes interoperability across various verifiable data systems:
  ● CBOR inclusion and consistency proofs for RFC9162
  ● Enable offline verification
  ● Gives COSE the ability to understand “proof types” beyond signatures, macs and ciphers.
  ● There are other transparency use cases, such as “key transparency” & “certificate transparency”.
Status

- Recently published -01:
  - Updated to support generic “verifiable data structures” and “proof type”.
  - Updated IANA Registry Request to accommodate more than Merkle trees, correspondingly.
  - More proof types in the future! (beyond consistency, e.g., freshness)
  - Need to improve CDDL examples (stay tuned).
  - Will monitor proof type output of KT and render it COSE interoperable.
  - Need to improve CDDL examples (stay tuned).
Application Example (SCITT Receipt)

See also https://github.com/ietf-scitt/draft-birkholz-cose-cometre-ccf-profile

```
# COSE_Sign1
18([

    # Protected Header
    h'a2012...43833633531',
    # {
    #   "alg" : "ES256",
    #   1 : -7,
    #   "verifiable-data-structure" : "RFC9162_SHA256",
    #   TBD_1 : 1,
    #   ... additional application specific headers ...
    # }

    # Unprotected Header
    {
        # "inclusion-proof" : "h'313...c3932""
        TBD_2 : h'3133...52c3932'
    },

    # Detached Payload

    # Signature
    h'486...7f77ea'
])
```
Demo Example (SCITT Transparency Service)

See also [https://scitt.xyz](https://scitt.xyz)
Profiling CoMETRE

● First addition to “verifiable data structures”:
  ● Confidential Computing Framework (CCF) Tree Algorithm
  ● Base CDDL:

    Receipt = [
        version: int,
        ts_identifier: tstr,
        proof: SignedMerkleTreeProof
    ]
CCF Profiling CDDL

CCF-leaf = [  
    internal-hash: bstr ; a string of HASH_SIZE bytes;  
    internal-data: bstr; a string of at most 1024 bytes; and  
    data_hash: bstr ; the serialization of the element stored at this leaf.  
]
CCF-inclusion-proof: [+ proof-element],
proof-element = [  
    left: bool  
    hash: bstr  
]
Next Steps

- Adoption Time?
- Profit!