Vector Commitments
Project Description

Anca Nitulescu
CryptoNetLab
Protocol Labs
Vector commitment schemes

Vector Commitments and their Applications, by Dario Catalano and Dario Fiore
Vector Commitments - VC

\[ m_0, m_1, m_2, \ldots, m_n \]
Vector Commitments

$m_0, m_1, m_2, \ldots m_n$

Later:

Open $m_i$

$m_i$
Vector Commitments

$m_0, m_1, m_2, \ldots m_n$

Later:

Open $m_i$

Verify Opening

$m_i$
SubVector Commitments - SVC

$m_0, m_1, m_2, \ldots, m_n$

Later:

Open

$m_1, m_4, m_6$

Verify Openings

$\pi$
Aggregate SVC openings

\[ \pi_{\text{agg}} = \pi_1 + \pi_2 + \pi_3 \]
First Goal: Aggregate SVC and openings

\[ \pi_{agg} = \pi_1 + \pi_2 + \pi_3 \]
Final Goal: Agg SVC and Functional openings

\[ \text{Final Goal: Agg SVC and Functional openings} \]

\[ VC_1 + \pi_1 + VC_2 + \pi_2 + VC_3 + \pi_3 + VC_{agg} + f(v_1, v_2, v_3) + \pi_{agg} \]
Merkle Tree Solution
Merkle Tree Solution

Top Hash

Hash 0
hash(hash(Hash 0-0, hash 0-1))

Hash 0-0
hash(Hash 0-0)

hash(L1)
L1

Hash 0-1
hash(Hash 0-1)

hash(L2)
L2

Hash 1
hash(Hash 1-0, hash 1-1)

Hash 1-0
hash(Hash 1-0)

hash(L3)
L3

Hash 1-1
hash(Hash 1-1)

hash(L4)
L4

Data Blocks
Merkle Tree Solution

Diagram showing a Merkle tree with four leaves (L1, L2, L3, L4), two intermediate nodes (Hash 0 and Hash 1), and a top hash. The tree structure is illustrated with arrows connecting the nodes, indicating the hash values and the data blocks.
Merkle Tree Solution

[Diagram of a Merkle Tree with hash nodes and data blocks]

Top Hash
hash(Hash 0, Hash 1)

Hash 0
hash(Hash 0-0, Hash 0-1)

Hash 0-0
hash(L1)

Hash 0-1
hash(L2)

Hash 1
hash(Hash 1-0, Hash 1-1)

Hash 1-0
hash(L3)

Hash 1-1
hash(L4)

Data Blocks:
L1, L2, L3, L4
Merkle Tree Solution

Data Blocks

Hash 0
- Hash 0-0
  - hash(L1)
- Hash 0-1
  - hash(L2)

Hash 1
- Hash 1-0
  - hash(L3)
- Hash 1-1
  - hash(L4)

Top Hash
- hash(Hash 0
  - Hash 0-0, Hash 0-1)
- hash(Hash 1
  - Hash 1-0, Hash 1-1)
Merkle Tree Solution
Use Cases in Filecoin

- **Use VC instead of Merkle Trees for sealing the sectors**
- **Maintaining Storage - Proof of Spacetime [WindowPoSt]**
  - Every day a miner must open thousands of indices and prove succinctly
  - **Aggregate Openings**: Use SVC with succinct proofs
  - The number of commitments in one block doesn’t scale well (many sectors → many VC)
  - **Goal 1 - Aggregate VC**: Enable aggregation for commitments and design opening proofs with respect to aggregated ones.
- **Add storage [ProveCommit]**
  - Verify that some openings of this VC have a relation with openings of a merkle tree
  - Today: this relation can only be proven with a generic SNARK
  - **Goal 2 - VC feature**: Proof embedded in the functional opening of a VC
## Open Questions and Roadmap

<table>
<thead>
<tr>
<th>VC Scheme</th>
<th>Group</th>
<th>Setup</th>
<th>Commitment</th>
<th>Proof</th>
<th>Verification time for $m$</th>
<th>Update</th>
<th>SVC</th>
<th>Aggreg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Merkle Tree</td>
<td>$G/G_7/L$</td>
<td>$1[H]$</td>
<td>$1[H]$</td>
<td>$\log n[H]$</td>
<td>$(\log n)H$</td>
<td>no</td>
<td>×</td>
<td>via SNARKs</td>
</tr>
<tr>
<td>[Tom20]</td>
<td>bilinear</td>
<td>$n \log n[G]$</td>
<td>$1[G]$</td>
<td>$1[G]$</td>
<td>$(m \log^2 m)G$</td>
<td>key</td>
<td>✓</td>
<td>one-hop</td>
</tr>
</tbody>
</table>