



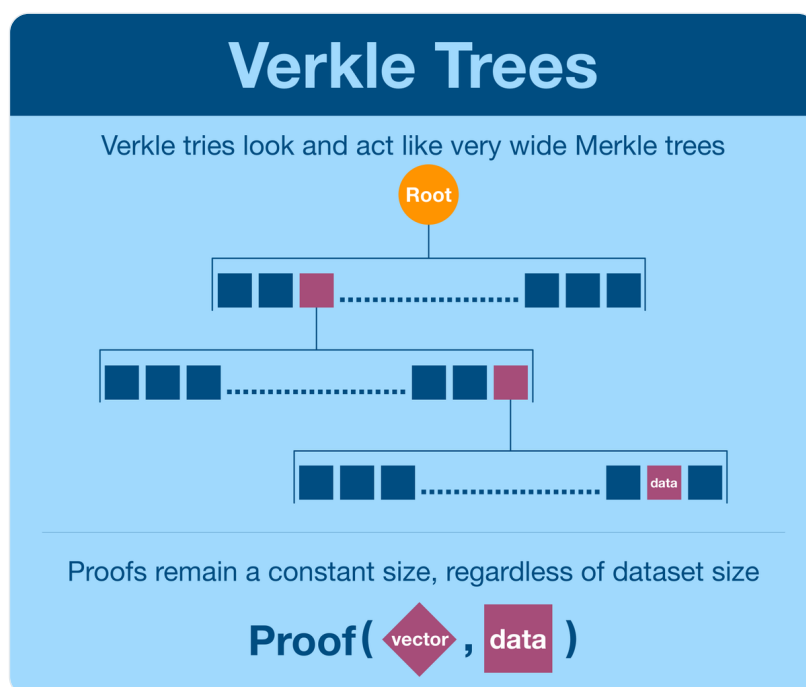
Haym @SalomonCrypto

Nov 2 · 26 tweets · [SalomonCrypto/status/1587668139466113024](https://twitter.com/SalomonCrypto/status/1587668139466113024)

(1/25) Cryptographic Innovation: Verkle Trees

[@ethereum](#) is the World Computer; born a rudimentary number cruncher, on a journey to (inevitably) becoming the dominant global settlement layer. And soon, Ethereum will outgrow the Merkle tree.

Tomorrow's solution: Verkle Trees



(2/25) We begin with a hash function, which transforms an arbitrary dataset into a unique, fixed-length string. A good hash function is irreversible and creates outputs that are indistinguishable from random data.

Think of a hash like a serial number for data of any kind/size.

**Haym**
@SalomonCrypto · [Follow](#)



(1/7) Computer Science 101: Hash Functions

What is a hash function? What are the characteristics of a good hash function? Where do hash functions appear and why do I hear about them all the time?

If you want to understand the fundamental tool of crypto, this guide is for you!

Hash Functions

A hash function transforms any amount of data into a compact value of uniform length.

| INPUT | OUTPUT |
|---|-------------------------------|
| Hello World | 0x829bd824b016326a401d063b |
| Hello Wold | 0xabd9bd33983cb06776e89273 |
| Social Security Number: ***-**-**** | 0xad22b653d2d85490c0147dfa1 |
|  | 0x91bfa44d98f1d3e2vv2d098d5ff |
|  | 0x299cfce9763c53debb12a87e1 |

A good hash function is quick and efficient to compute, but difficult (if not impossible) run in reverse, and distribute values uniformly (randomly) accross all possible outputs.

3:54 PM · Sep 7, 2022


 [Read the full conversation on Twitter](#)

 127  Reply  Copy link

[Read 1 reply](#)

(3/25) Merkle Trees are made by successive rounds of hashing.

All the data begins on the bottom row (leaf nodes). Then they are grouped together and fed into a hash function. These hashes are grouped (same size) and hashed again, continuing until there is a single (root) node.

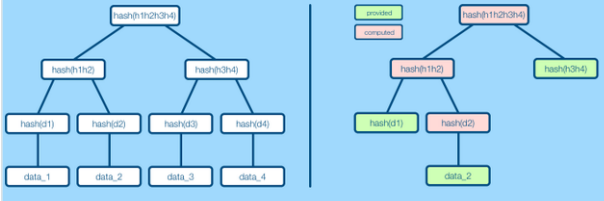
 **Haym**
@SalomonCrypto · Follow

(1/13) Computer Science 201: Merkle Trees and Merkle Proofs


If you want to understand @Bitcoin, @ethereum and blockchain technology, you need to learn:

- How a Merkle trees expresses a large dataset
- How a Merkle proof works
- Why a Markle tree is so efficient

Merkle Trees and Proofs



10:17 PM · Sep 7, 2022

 [Read the full conversation on Twitter](#)

♥ 457 💬 Reply 🔗 Copy link

[Read 16 replies](#)

(4/25) The purpose a Merkle tree is enable Merkle proofs, which allow verification that a piece of data exists in the underlying dataset (leaf nodes) without transmitting the entire dataset.

This has profound implications for both privacy and bandwidth requirements.

(5/25) The privacy aspects aren't useful for this discussion, but they are worth nothing.

Merkle proofs can verify a piece of data without having to articulate the entire dataset. You do need component pieces of the Merkle tree, but these are just (non-sensitive) hashes.


(6/25) We are here for the bandwidth implications.


Let's say you want to post a ton of data. No one is interested in all the data, and every one has access to Merkle proofs generation (big assumption).

Instead of posting the entire dataset, you can just post the Merkle root.

(7/25) Don't get me wrong, a Merkle proof is significantly more efficient than naïve approach... but it has its limits.

Tl;dr Merkle proofs scale much less quickly than dataset size, but they still scale proportionally. Eventually proofs will outgrow bandwidth capacity.

**Haym**
@SalomonCrypto · [Follow](#)

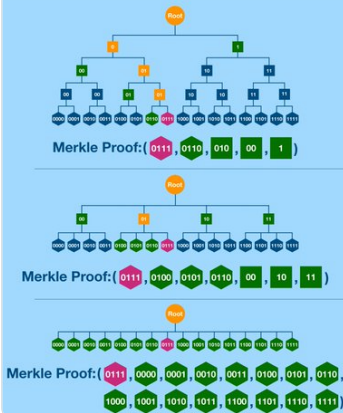


(1/18) The Problem with Merkle Trees

At the heart of [@Bitcoin](#), [@ethereum](#) and many blockchain computers is the Merkle Tree. While this data structure has served us well, it is not perfect... and if you look ahead, you can see impending problems.

Let's talk Merkle proof scaling.


Merkle Tree Proof Scaling






The diagram illustrates the scaling of Merkle tree proofs. It shows three trees of different widths, each containing 16 data points at the leaf level. The proof size is the number of nodes in the path from the root to the target leaf, including the root and the leaf itself.

| Merkle Tree Width | Proof Size |
|-------------------|------------|
| 2 | 5 |
| 4 | 7 |
| 16 | 16 |

4:14 PM · Oct 29, 2022

 [Read the full conversation on Twitter](#)

 893  Reply  Copy link

[Read 51 replies](#)

(8/25) What happens if we have a dataset too big for a Merkle tree?

IRL Example: the internal state of [@ethereum](#) is stored in a Merkle tree. This tree grows as more accounts & smart contracts are created. Eventually proofs are going to grow too big to push through the network.

(9/25) Let's start by asking the question "what's the purpose of a Merkle tree?"

A Merkle root is a unique string that is generated from a large dataset. With the root, anyone can verify the original dataset and/or an individual piece of data.

We call this a commitment scheme.

Haym
@SalomonCrypto · Follow

(1/20) Cryptography Basics: Commitment Schemes

A commitment scheme is a primitive that allows one party to generate a piece of data anchored to a specific dataset.

This anchor (commitment), cannot leak information and yet can unequivocally verify the dataset.

Commitment Scheme
A commitment scheme allows one party to bind themselves to a particularly set of data without revealing that data

Commit

A commitment is a small piece of data that is uniquely bound to the original data. Though the commitment is unique, it does not leak any information about the dataset used to generate it.

The commitment can be used to generate proofs. Proofs will only verify if the proving data matches the input dataset

A party can use the commitment and extra data created from the original dataset (a proof) to verify the underlying data

+ =

7:56 PM · Oct 30, 2022

(10/25) Our new scheme must avoid the bottlenecks that Merkle trees face; proof size must stay constant regardless of dataset size.


A Merkle proof contains the intermediate hashes needed to rebuild the root. What if we use cryptography to condense all of that into one value?


(11/25) We'll start with a Vector.

Tl;dr vectors are a concept used to convey quantities that cannot be expressed by a single number.

In two dimensions, "2 spaces right, 3 spaces up" can be represented as the vector (2,3)

A n-dimensional vector can express n data points.

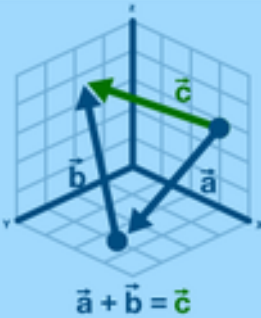
**Haym**
@SalomonCrypto · [Follow](#)



(1/8) Computer Science Basics: Vectors

Vectors are a mathematical primitive that turn out to be particularly useful in computer science. In order to understand modern cryptography (and other advanced computational applications), you need a good grasp on this foundational topic.


Vectors in Computer Science







An arbitrary data set $(v_0, v_1, v_2, \dots, v_{n-1}, v_n)$ can be expressed as a vector

Vectors can be used in math operations like addition, subtraction, etc.

Vectors can be used to mathematically/programmatically manipulate data in particularly useful ways.

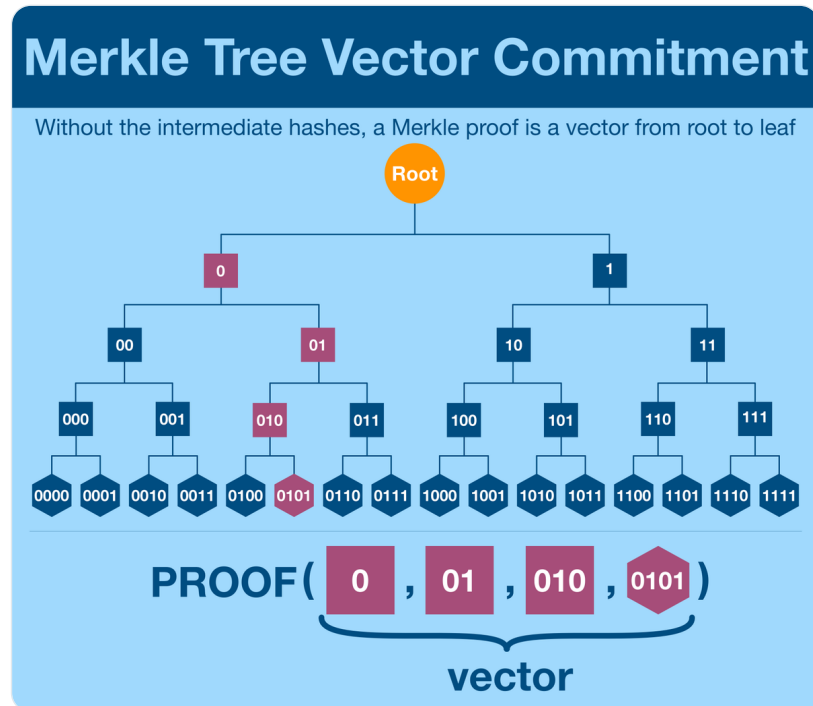
2:02 AM · Oct 30, 2022 

 [Read the full conversation on Twitter](#)

 134  Reply  Copy link

(12/25) Let's simplify how we think of Merkle proofs. The point of sharing all the intermediate branches is to generate the unique path from the Merkle root to your data point.

Let's think about expressing this value as a vector.

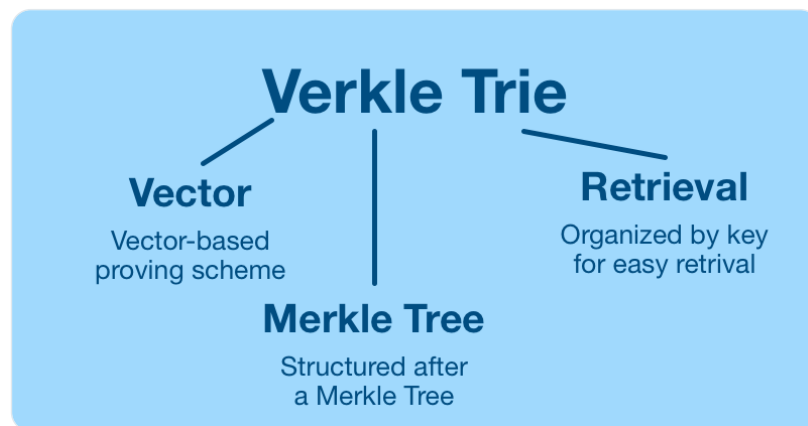


(13/25) This will form the foundation of Verkle tree. Here's more than you want to know on the name.

V = vector

erkle Tree = Merkle Tree

We will also be interested in Verkle tries which are trees organized by their keys (in example above, you can trace keys through inner nodes)




(14/25) Again, we've already built a scheme around a vector. The challenge is to build it so that the vector can be (trustlessly) transmitted with a constant size.


Fortunately, modern cryptography has developed lots of tools to solve this exact problem.

(15/25) For example, we could deploy KZG commitments for this purpose.

Tl;dr the KZG commitment scheme uses elliptic curve cryptography to commit to a polynomial.

A polynomial commitment is actually much more powerful than a vector commitment, but can also be used as one.

**Haym**
@SalomonCrypto · [Follow](#)






(1/24) KZG Polynomial Commitments: The Complete Guide

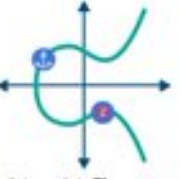
Our goal: 1) prove we are committed to a specific set of data and 2) allow others to verify specific points within that dataset.


Want to see some mathematical magic? This megathread is for you!

KZG Commitment Scheme

First, the prover commits to data by creating a point on the elliptic curve. If the data changes, the prover cannot create valid proofs.

Prover
1)  Commit
3)   Proof Evaluation



Verifier
2)  Request

Next, the verifier gives a data point. The prover builds a new elliptic curve point and a polynomial evaluation around that point.

KZG Proof Verification

$$e([S - z], [h(S)]) \stackrel{?}{=} e([f(S)] - f(z), [1])$$
$$\downarrow$$
$$e([S - z], [\text{Proof}]) \stackrel{?}{=} e([\text{Commit}] - f(z), [1])$$

Calculated by verifier

Proof

Commit

Evaluation

(16/25) KZG commitments require too much computation for the Verkle trees we are interested in. We will use Pedersen commitments to commit to a vector.

I haven't gotten to that thread, but it's worth reading ahead. We will see Pedersen commitments soon.

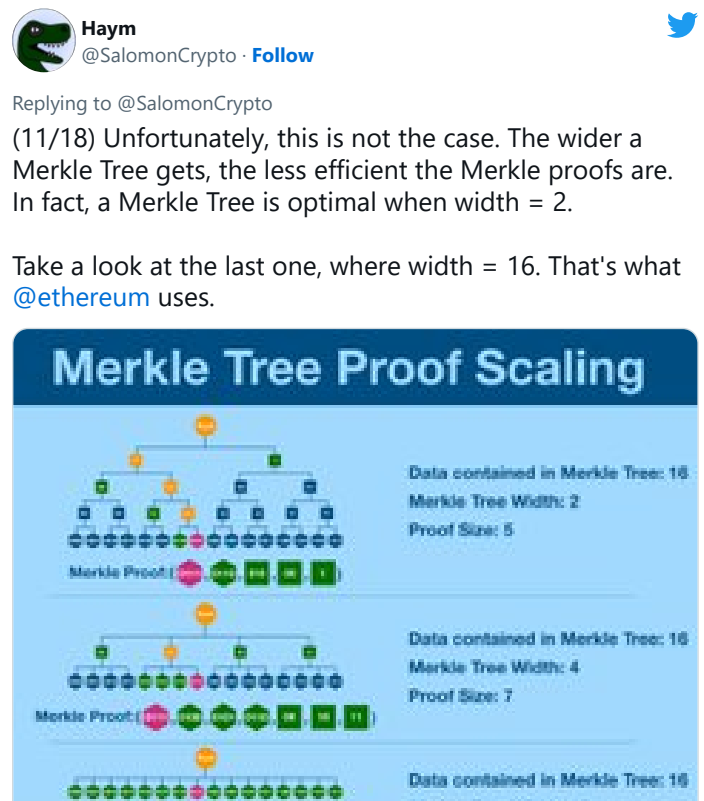


<https://www.youtube.com/embed/lkNZWJFcfU>

(17/25) We will pick the particular scheme based on the application, but the point is that we directly prove the vector. Unlike Merkle trees, we do not need to first rebuild the vector before we prove it.

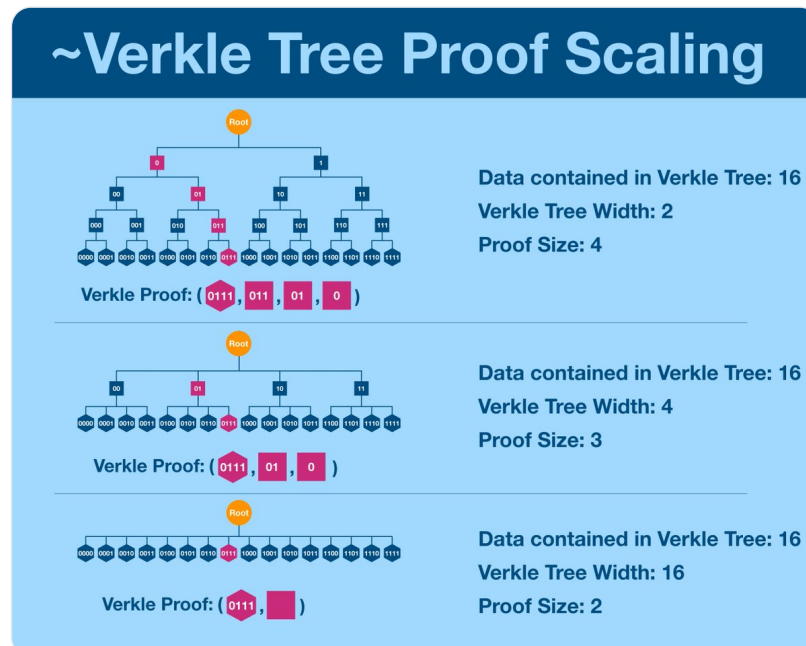
Individual vector components are of constant size.

(18/25) Remember, Merkle proofs grew in size as the underlying dataset grew (albeit more slowly). When we tried to decrease the number of vertical levels, we got even more horizontal components.



(19/25) Verkle trees do not have this problem. Shortening the tree decreases proof size.

Here's a rough idea, but don't dwell too long on it. In fact, the cryptographic schemes we are looking are so powerful that we can simplify even further.



(20/25) KZG commitments, Pedersen commitments and all the other schemes we are considering can condense an arbitrary number of proofs into a single value.

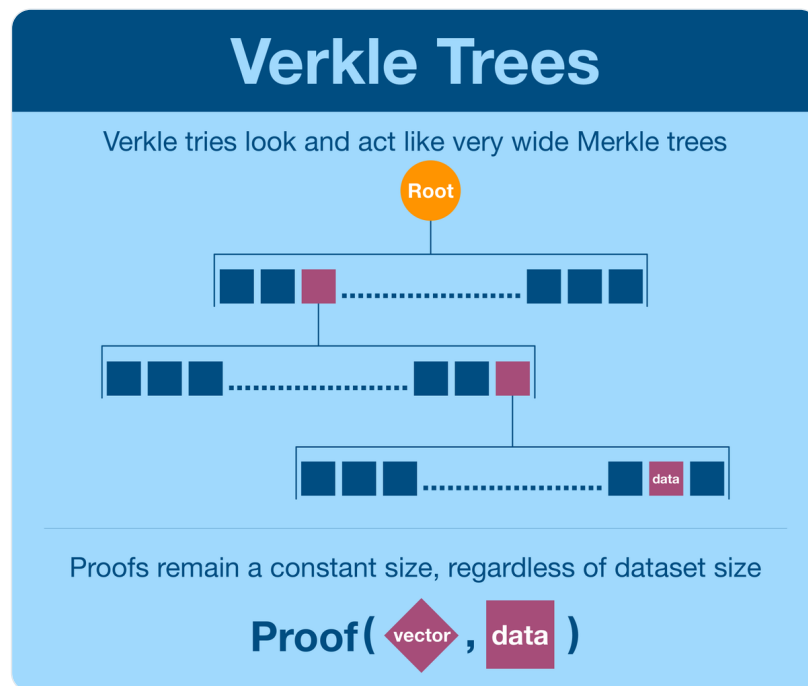
Put another way, we can use the magic of elliptic curve cryptographic to create a single vector that serves for all proofs.

(21/25) [@dankrad](#) lays out the technique here in his blog. It's pretty in depth and technical, not for the faint of heart.

<https://dankradfeist.de/ethereum/2021/06/18/pcs-multiproofs.html>

(22/25) Which brings us to actual Verkle trees.

Verkle proofs don't scale with tree width and therefore can be incredibly wide. The current proposal for the [@ethereum](#) state is looking at a width of 256, but some are pushing for 1024.



(23/25) One might ask "can we make it infinitely wide?" We can, we can create a "Verkle tree" that is just a single level, making proofs incredibly lightweight.

In practice, this will cause a huge burden during the commitment generation phase.

(24/25) Turns out this might be an inherent property of commitment schemes, maybe like "in order to create a lightweight proof, a commitment needs to touch every piece of data."

If so, the commitment scheme design ends up becoming a trade-off between prover and verifier work.

(25/25) Verkle trees are an advanced improvement on an already complicated topic. But here's all you need to know:

- Verkle trees, like Merkle trees, allow verification of data within a dataset
- Verkle proofs are of constant size, regardless of dataset

More of a long-form reader? Try this out:

**Haym**
@SalomonCrypto




Verkle Trees

Verkle Trees | Haym
(1/25) Cryptographic Innovation: Verkle Trees @ethereum is the World Computer; born a rudimentary number cruncher, on a journey to (inevitably) becoming the dominant global settlement layer. And soo...
<https://typefully.com/SalomonCrypto/xsrSRdO>

Like what you read? Help me spread the word by retweeting the thread (linked below).

Follow me for more explainers and as much alpha as I can possibly serve.

**Haym**
@SalomonCrypto · [Follow](#)



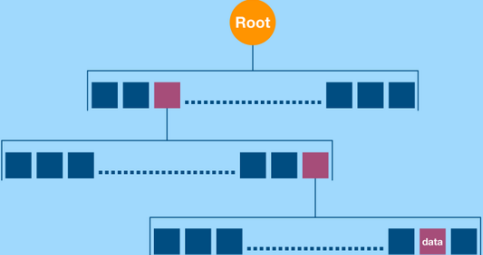
(1/25) Cryptographic Innovation: Verkle Trees

[@ethereum](#) is the World Computer; born a rudimentary number cruncher, on a journey to (inevitably) becoming the dominant global settlement layer. And soon, Ethereum will outgrow the Merkle tree.

Tomorrow's solution: Verkle Trees

Verkle Trees

Verkle tries look and act like very wide Merkle trees



Proofs remain a constant size, regardless of dataset size

Proof( , )

4:49 AM · Nov 2, 2022



[Read the full conversation on Twitter](#)

...