



Blockchain foundations

Module 1 (part 2)

Part 2

The Technical Side

- To understand how blockchain works
 - Cryptography (hash functions and digital signatures play critical roles in blockchains).
 - what a block is and how those blocks form chains.
 - what mining and proof-of-work are and be able to explain what nodes do and at high level how blockchain networks function.

- TWO cryptographic concepts that underpin blockchain technology..

Ingredient #1: Hashes

Ingredient #2: Digital Signatures

- Hash functions are perhaps the single most important thing to understand if you really want to get how blockchains work.

What is Cryptography?

- Cryptography is essential to pretty much every part of cybersecurity we have.
- From the Greek cryptography translates to hidden writing.

Are actual passwords saved on server?



Why strong passwords?

Cryptographic Hash Functions

[Source from ConsenSys]



The image part with relationship ID rld5 was not found in the file.

Properties of hash functions

Arbitrary input size (of any digital form)

Fixed-output size (e.g. 256 bits)

- 3c5e5ab55368e9fb85e30743fd04dae22867de1ad0d2a8dc93b563cccec6e8b99

Other properties

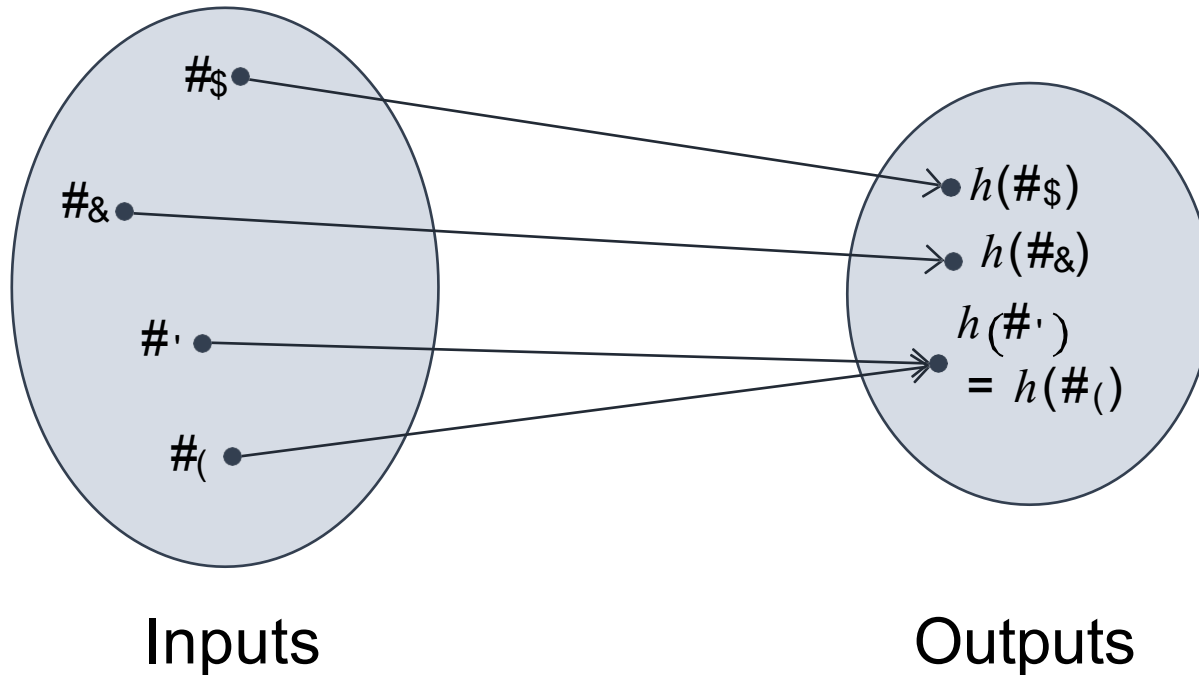
- **Deterministic**
- **Uniform** (inputs are mapped evenly to the space of possible outputs)
- **Efficiently** computable

security properties:

[collision-free](#), [preimage-resistant](#)...

More reading [link](#).

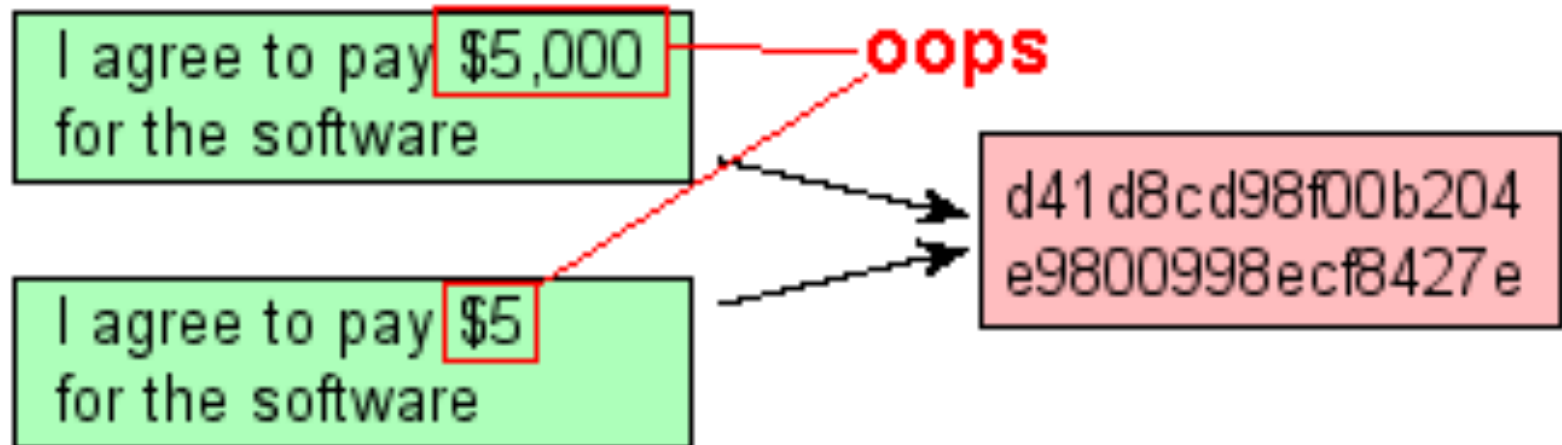
Collisions in hash functions



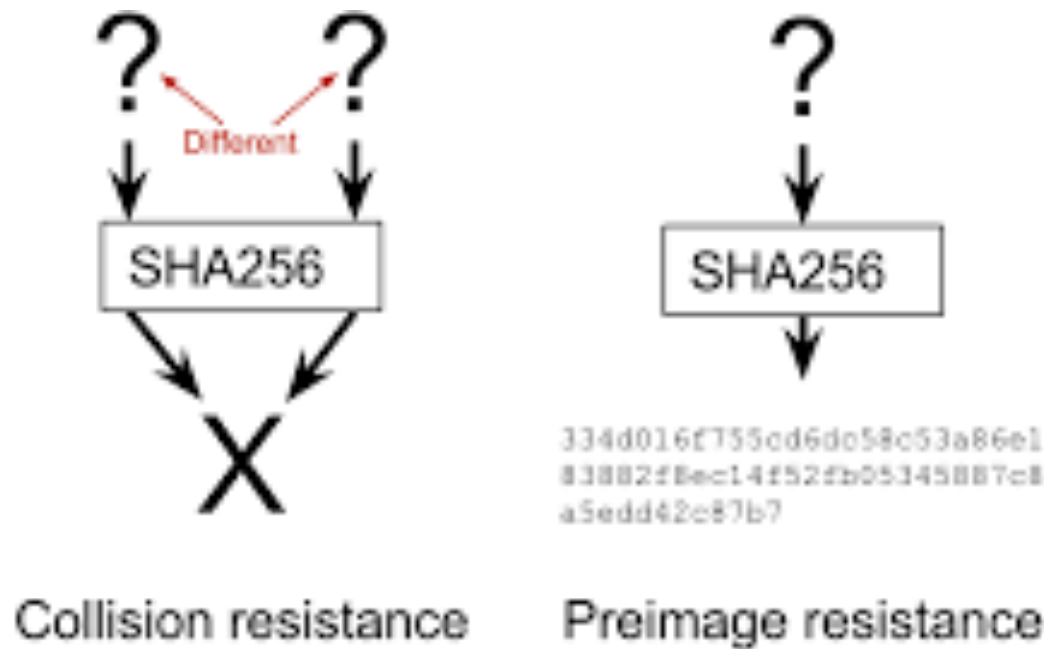
Collisions exist

- Inherent: input space is **unbounded**
- However, cryptographic hash functions ensure collisions are **hard to find**

This is a good collusion!



Preimage attacks



it is computationally infeasible to find any input that hashes to that output, i.e., given y , it is difficult to find an x such that $h(x) = y$

Irreversible

Easy



Hard

Play with the hash function

<https://www.xorbin.com/tools/sha256-hash-calculator>

Robo Hash

GENERATE UNIQUE IMAGES FROM ANY TEXT

Robohash is a easy web service that makes it easy to provide unique, robot/alien/monster/whatever images for any text. Put in any text, such as IP address, email, filename, userid, or whatever else you like, and get back a pretty image for your site.

With hundreds of millions of variations, Robohash is the among the leading robot-based hashing tools on the web.

HOW COOL IS THIS?

<https://robohash.org/KSU> That guy to your left there? He was specially generated from your IP address *Just for you.*



Try on your phone, and I bet you get someone different!

Keep scrolling down to see some more freshly-assembled RoboHashes.

Enter any text, and press 'generate':

generate

<https://robohash.org/>

Math behind hash functions

- <http://www.iwar.org.uk/comsec/resources/cipher/sha256-384-512.pdf>

Applications of cryptographic hash functions

Data equality

- If we know that $h(\#) = h(\&)$, it is safe to assume that $\# = \&$

Data integrity

- To verify the integrity a piece of data $'$, we can remember its hash, $h(')$
- Later, when we obtain $''$ from **untrusted source**, to check whether $d = ''$ we can verify whether $h(d) = h(')$
- Useful because the $h(')$ is small (e.g. 256 bits)

Example: how to check a file is really coming from its source



WikiLeaks

[Leaks](#)

[News](#)

[About](#)

[Partners](#)

Vault 7: CIA Hacking Tools Revealed



[Releases ▼](#)

[Documents ▼](#)

Navigation: » [Directory](#) » [Embedded Development Branch \(EDB\)](#) » [EDB Home](#) » [Projects](#)

Activity

- Write a program to determine whether two files (of any type) are identical or not?

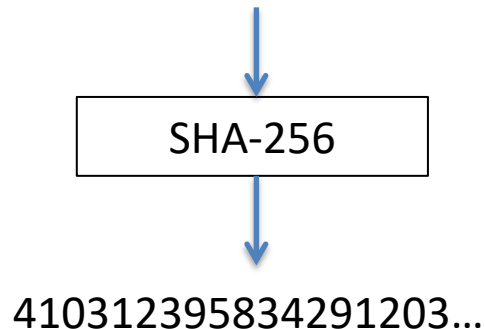
How blockchain makes use of hashes?

- To keep the current state of the blockchain...

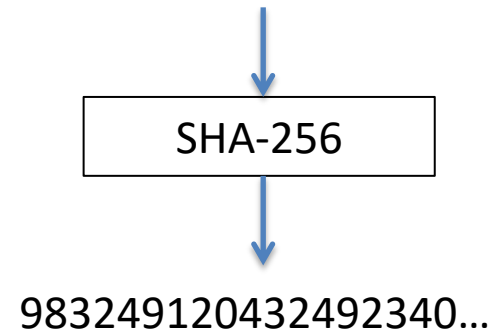
Ingredient #1: Hashes

- A hash function (like SHA-256) takes some data in, and produces an effectively random fixed size integer.
- Any change to the input randomizes it

“The quick brown fox did some crypto”



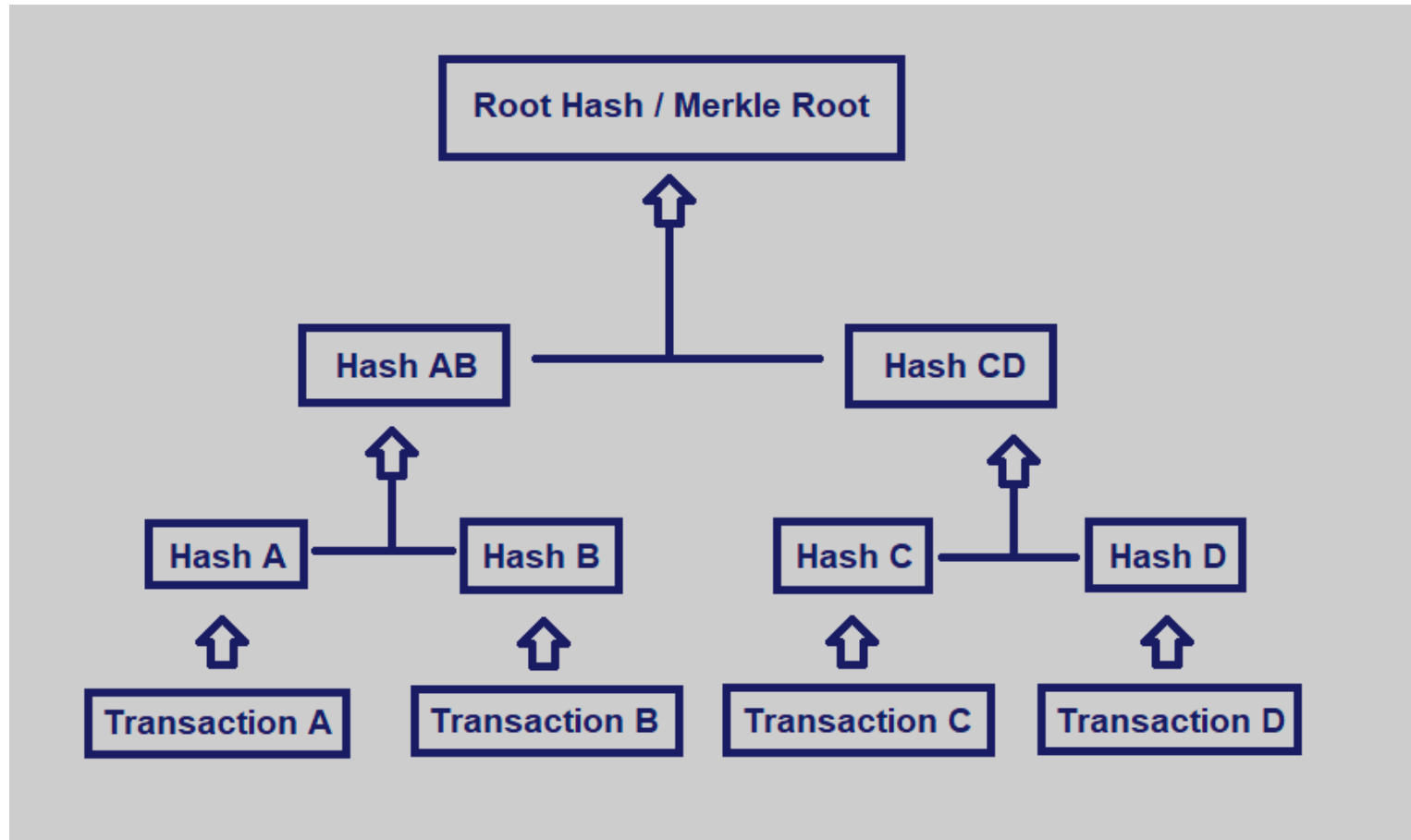
“The quick brown Fox did some crypto”



More reading: ([Ch 1.1](#) Cryptographic Hash Functions)

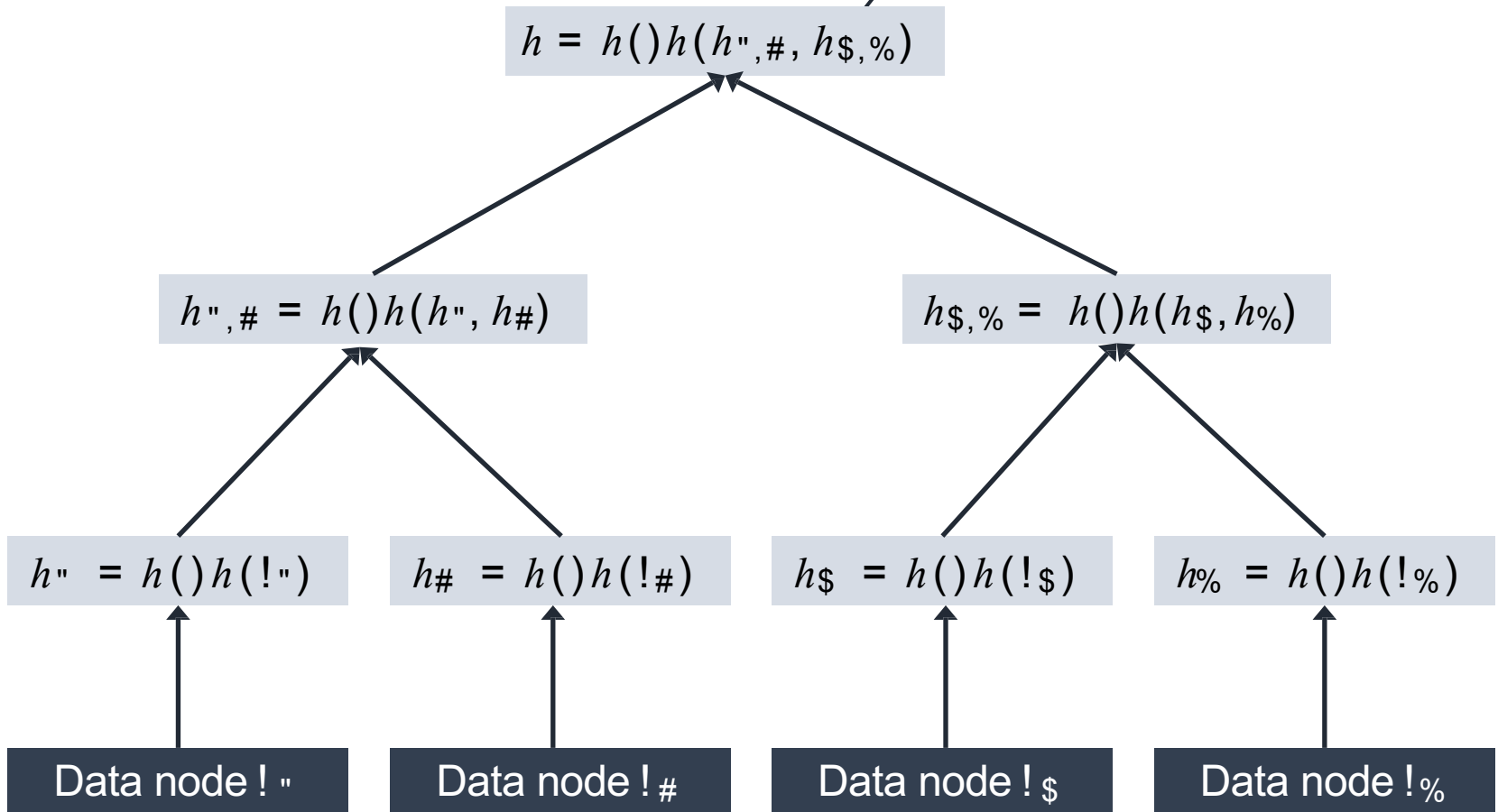
Merkle/hash tree

- Cryptographic hash functions are the underlying technology that allow for Merkle trees to work



Merkle tree

This is called
the hash root

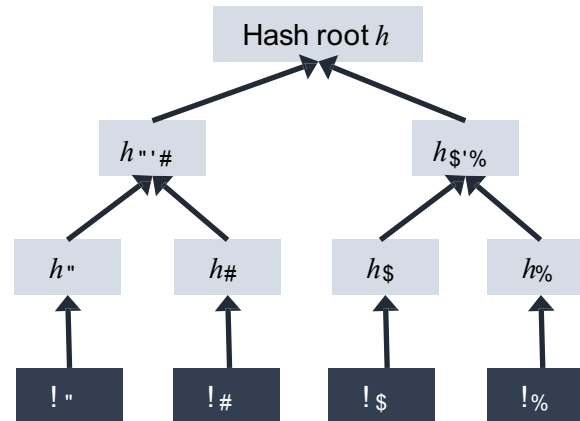


Who Uses Merkle Trees?

Merkle trees are used by Bitcoin, Ethereum, Git Apache Cassandra, and other DB systems

- Merkle trees have three major benefits:
 1. They provide a means to prove the integrity and validity of data
 2. They require little memory or disk space as the proofs are computationally easy and fast
 3. Their proofs and management only require tiny amounts of information to be transmitted across networks

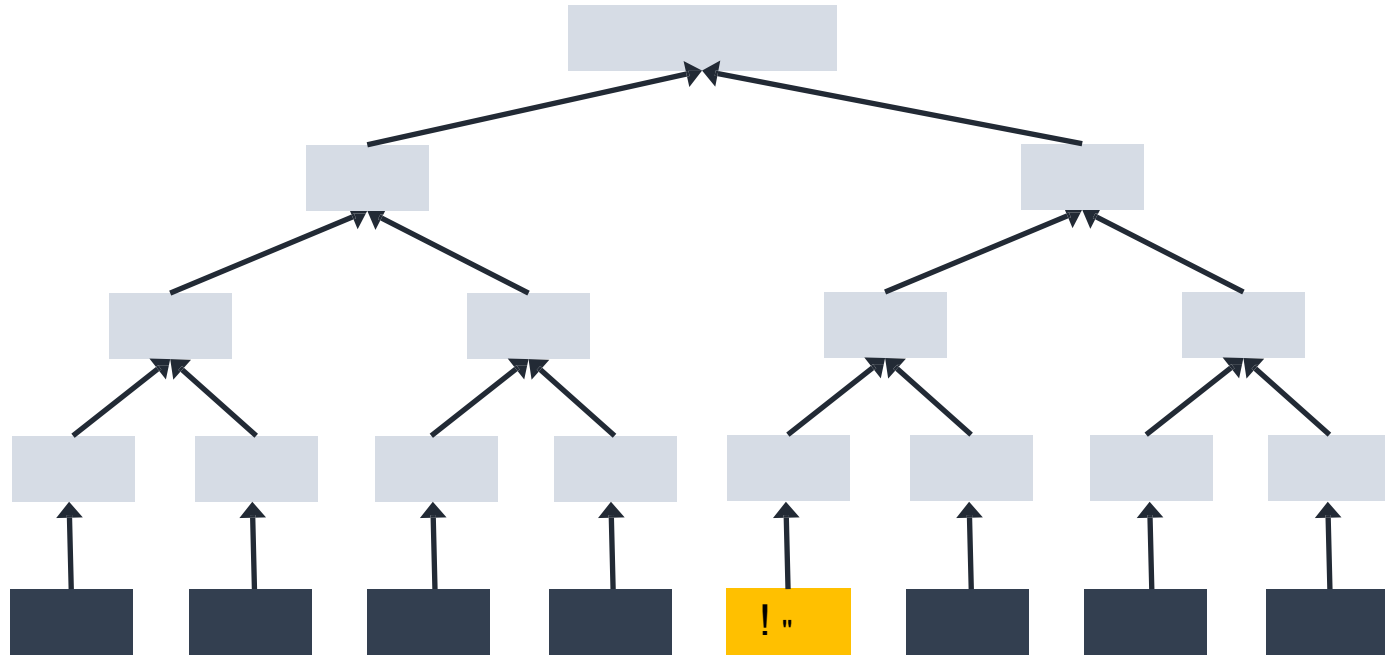
Uses of merkle trees



Data integrity verification (enables Merkle proofs)

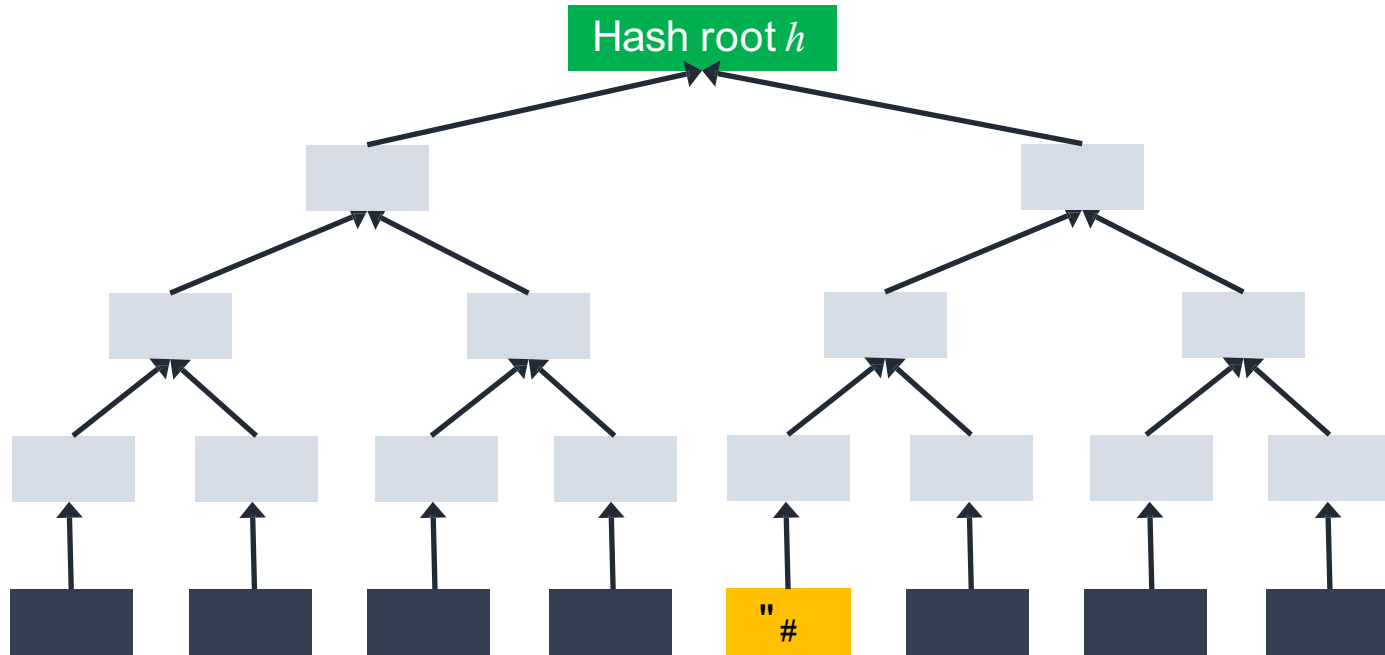
- The hash root h is typically obtained from a **trusted source**
- One can **verify** the **integrity** of the data elements $!''$, $!\#$, $!\$$, $!\%$ by reconstructing the hash root and comparing it to the trusted root

Example: Membership verification



Verifying whether ! \" is a member (i.e., a leaf):

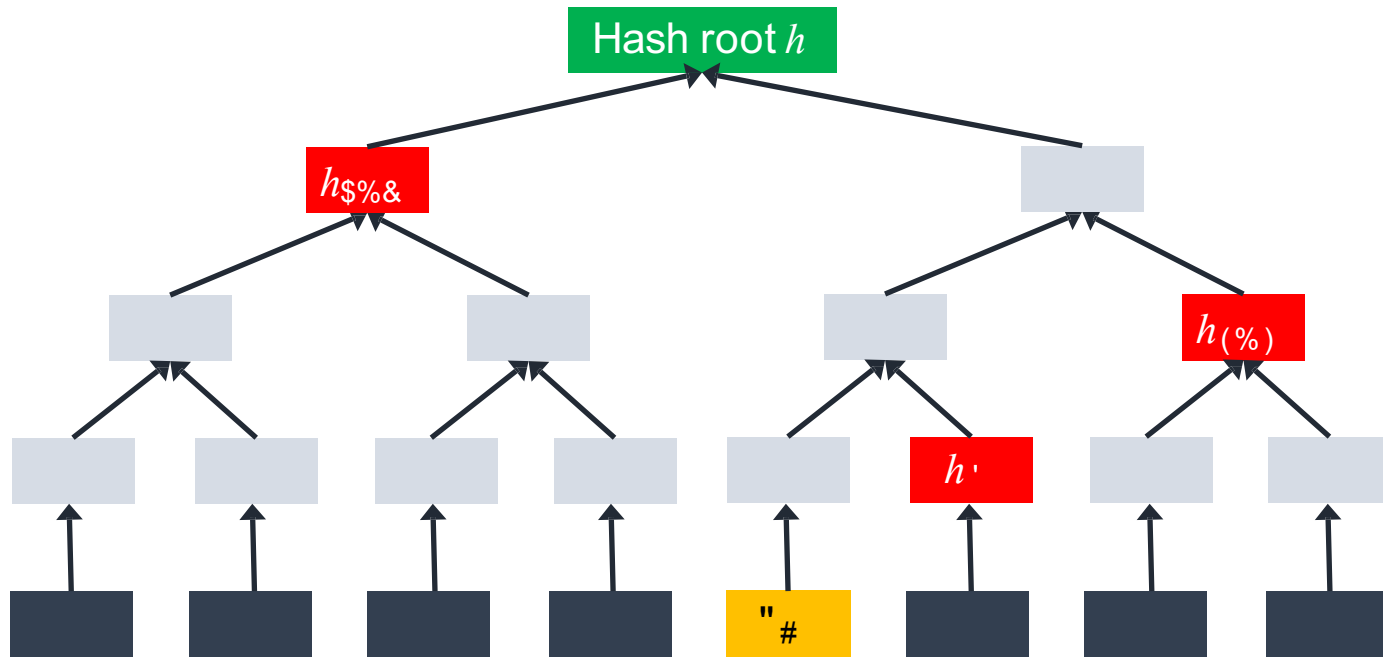
Membership verification



Verifying whether ""# is a member (i.e., a leaf):

- Obtain the hash root h from a **trusted source**

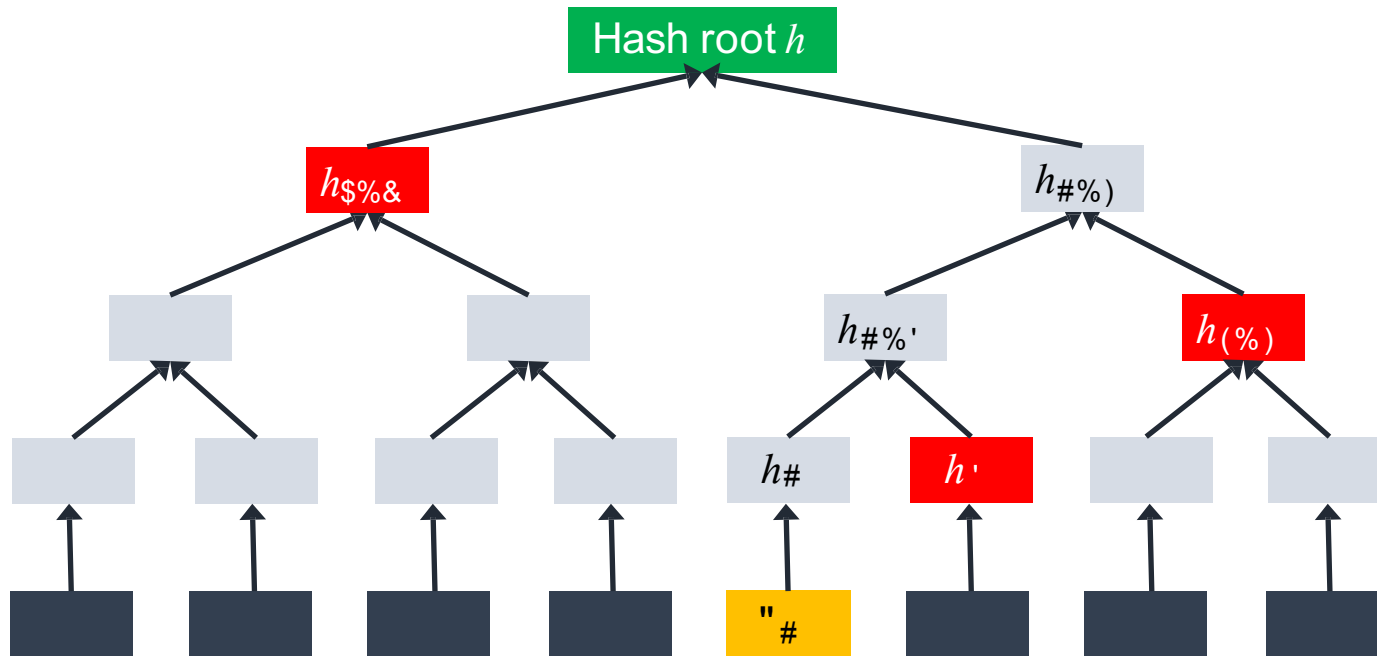
Membership verification



Verifying whether " $\#$ " is a member (i.e., a leaf):

- Obtain the hash root h from a **trusted source**
- Request the nodes h' , $h_{(%)}$, $h_{\$ \% \&}$ (from **possibly untrusted source**)

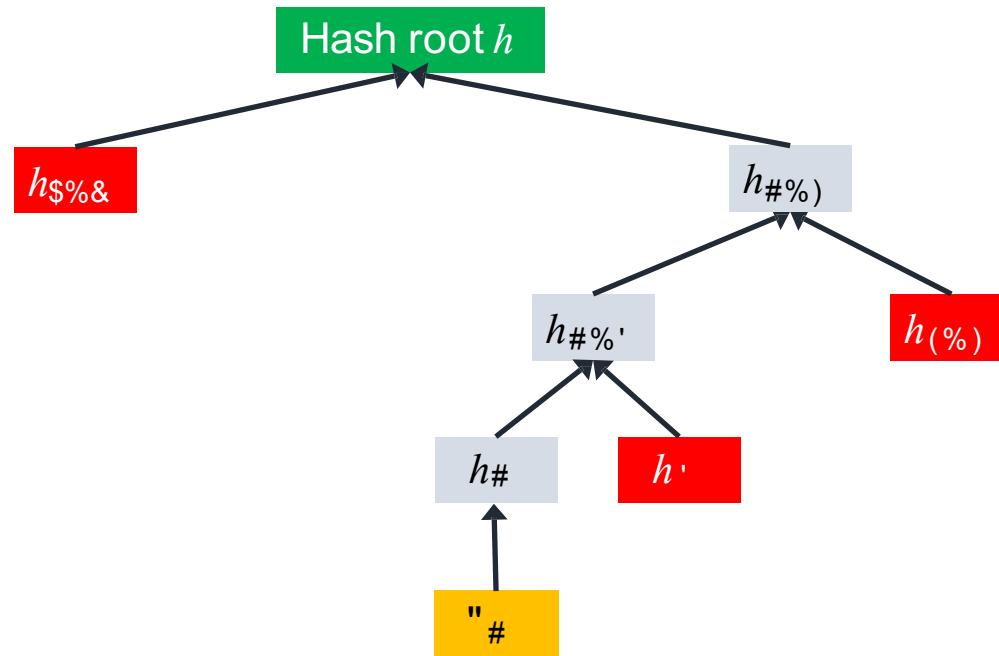
Membership verification



Verifying whether $\text{"\#}$ is a member (i.e., a leaf):

- Obtain the hash root h from a **trusted source**
- Request the nodes h' , $h(\%)$, $h\$ \% \&$ (from **possibly untrusted** source)
- Compute $h\#$, $h\# \% \'$, $h\# \%)$, h' and check whether $h = h^-$

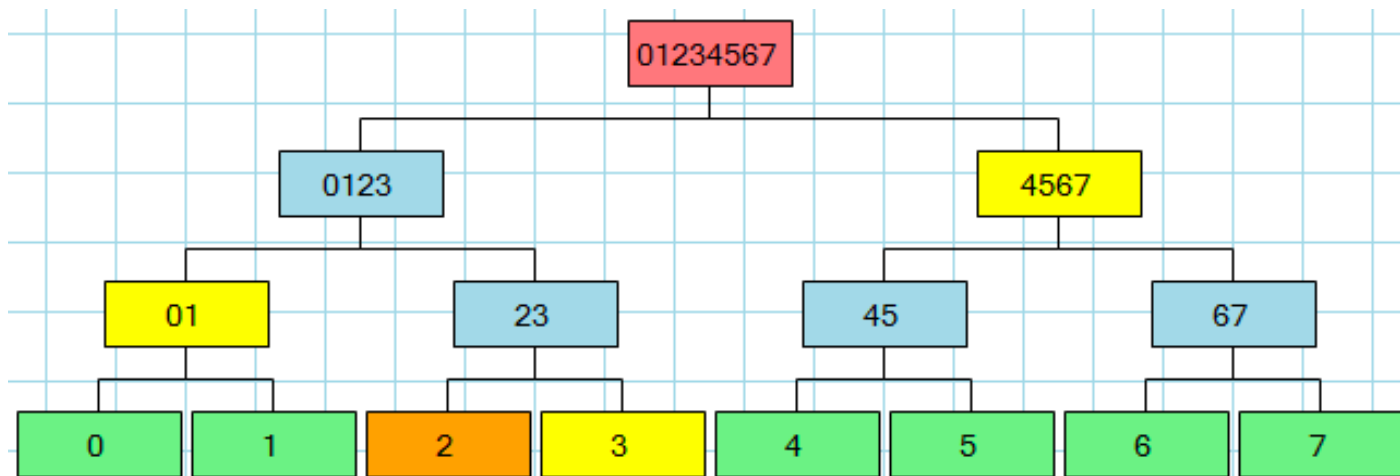
Membership verification



- Membership verification requires $\log(\cdot)$ elements
- Useful when the set of data elements is large

Another example: How Does Data Verification (Audit Proof) Work?

- Let's say you are the owner of the record "2"

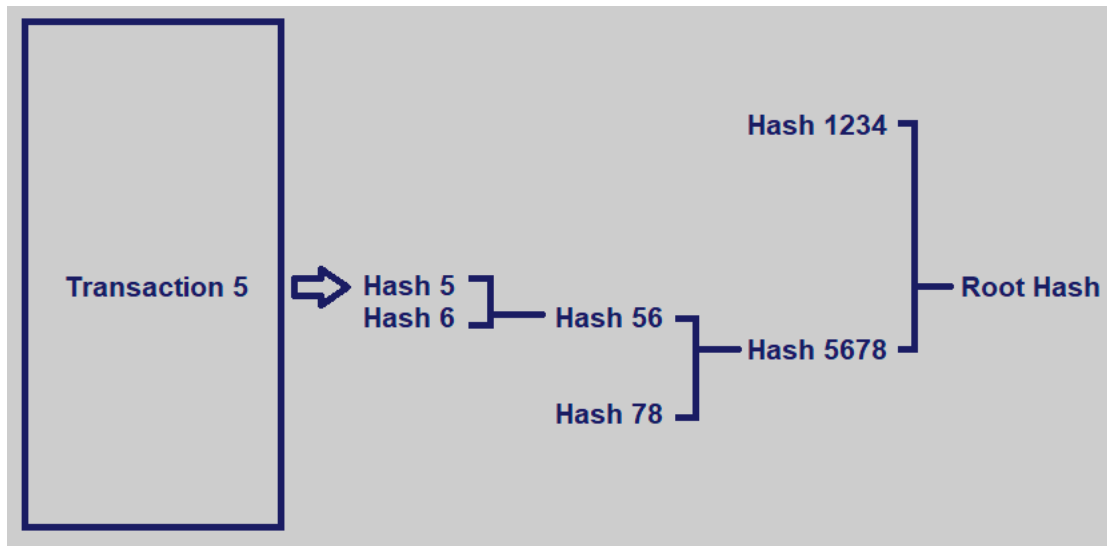


What is the proof?

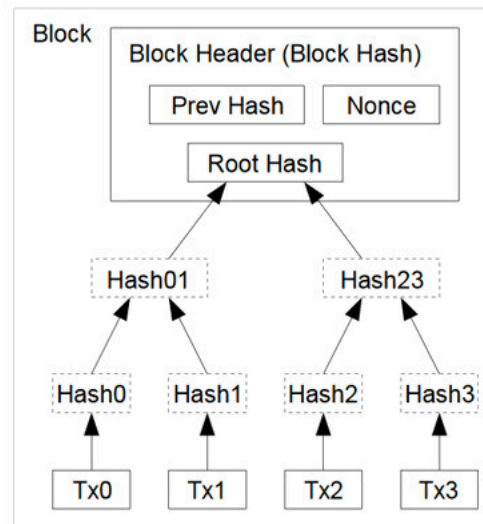
Any system **pretending** to validate your request would not be able to provide you with the intermediate hashes since you're not giving the system the root hash, you're just telling it to give you the proof - it can't invent the proof because it doesn't know your root hash -- only you know that.

Merkle tree in blockchain?

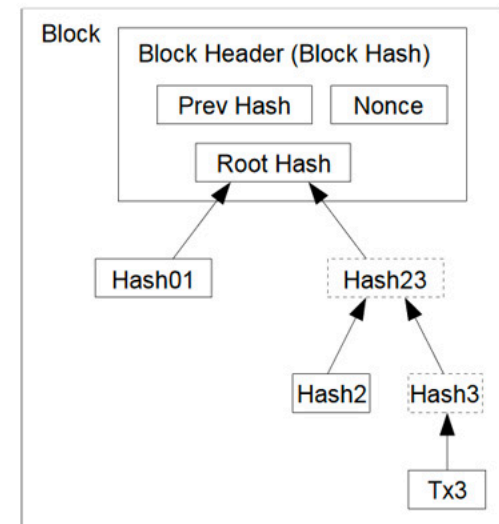
- A Merkle tree summarizes all the transactions in a block by producing a digital fingerprint of the entire set of transactions
- Allows for a quick and simple test of whether a specific transaction is included in the set / block.
- **It maintains the integrity of the data.** If a single detail in any of the transactions or the order of the transactions changes, so does the Merkle Root.



The entire dataset doesn't need to be downloaded to verify the integrity of Transaction 5.



Transactions Hashed in a Merkle Tree



After Pruning Tx0-2 from the Block

The image is from the Bitcoin [whitepaper](#) and illustrates how the Merkle tree fits into each block.

Blockchain demo

- <https://anders.com/blockchain/>
- <https://blockchaindemo.io/>