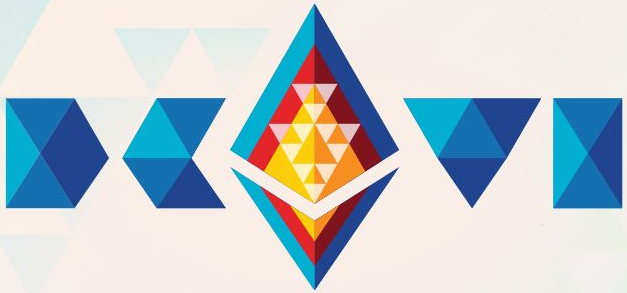


Light Clients

After the Merge

Etan Kissling
Nimbus, Status R&D GmbH

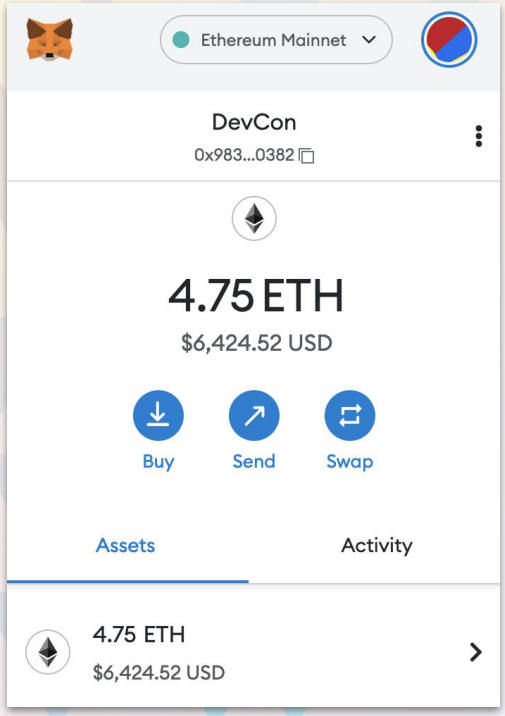
Section 1

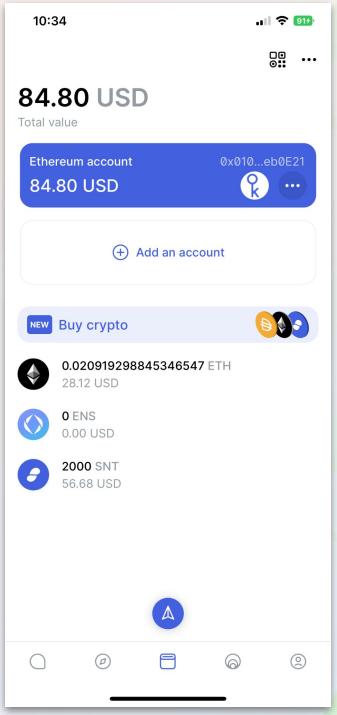
What is a Light Client?

Full Node Requirements

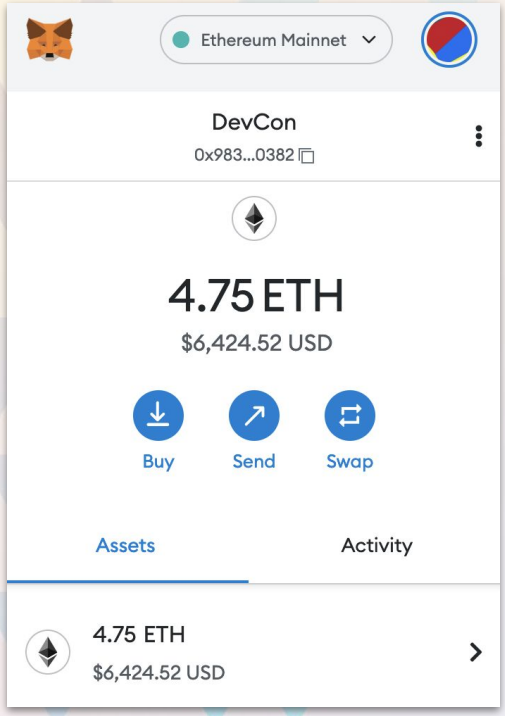
1.5 GHz Quad CPU	8 GB RAM	2 TB SSD	Unmetered Internet

Light clients?

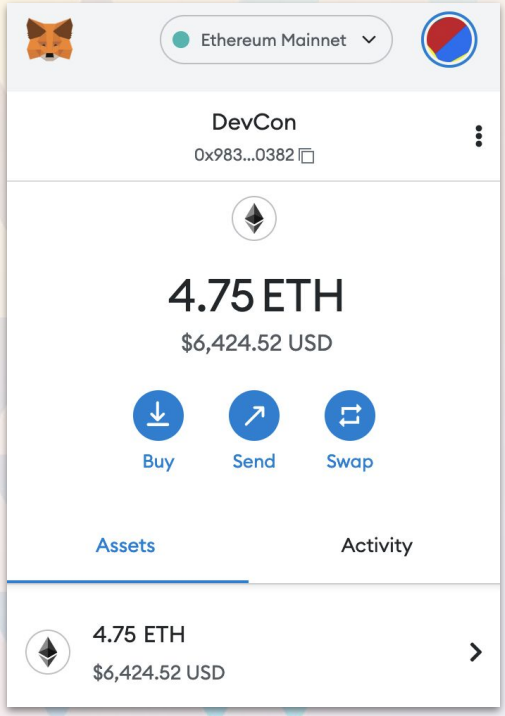




Your own wallet



Your own wallet



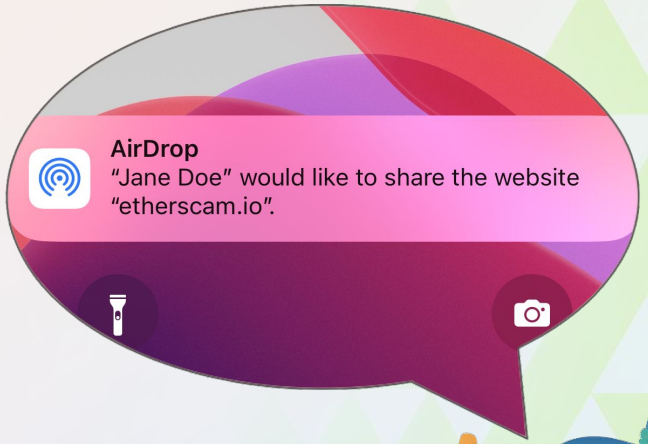
Your own wallet

You are doing great work!

Can I support with .5 ETH?

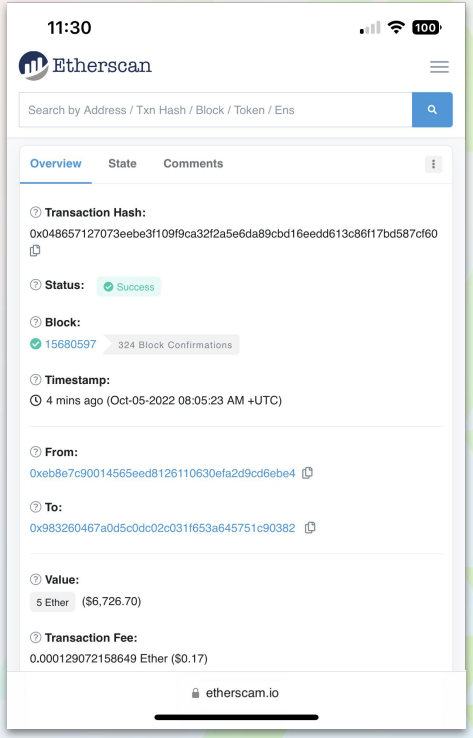
Sure!

Receiving a donation



Receiving a donation

Sure!

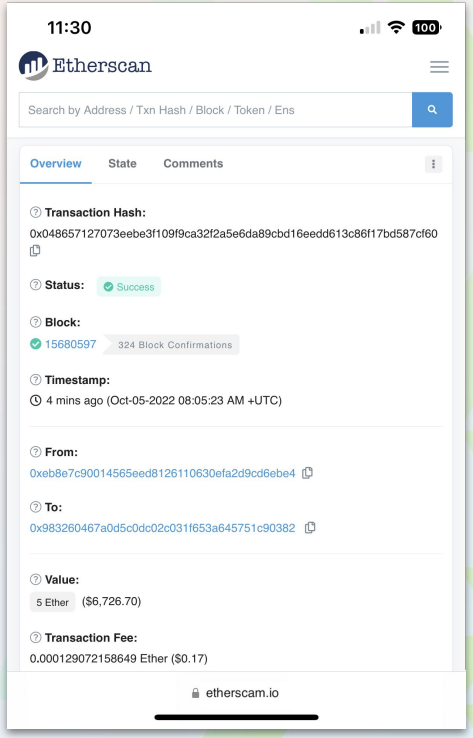


Receiving a donation

Value:

5 Ether ($6,726.70)

5 ETH, not .5?!

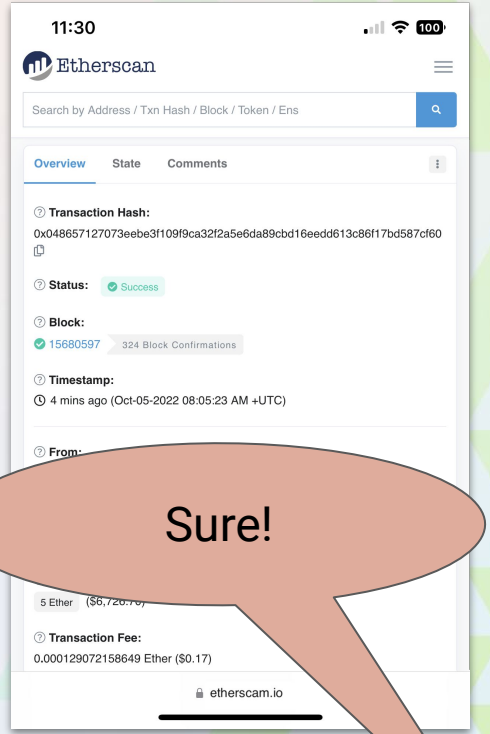


Receiving a donation

Value:

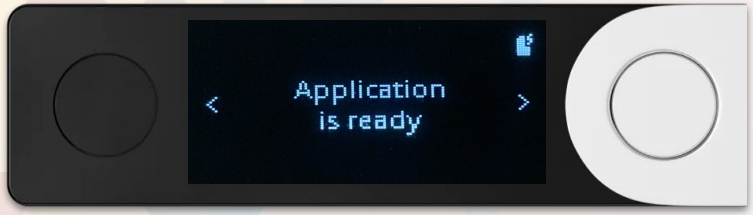
5 Ether ($6,726.70)

Refund
4.5 ETH?

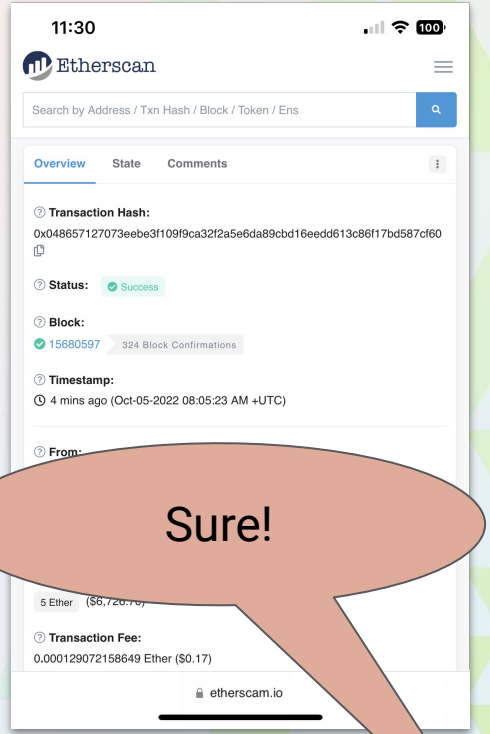


Sure!

Funds protected by HW wallet

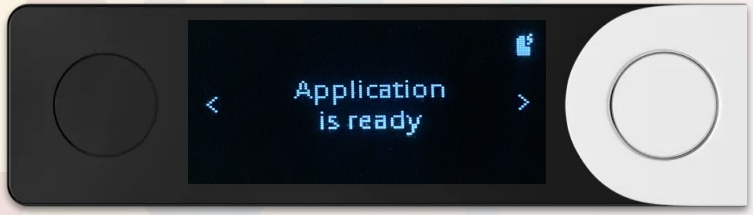


Refund
4.5 ETH?

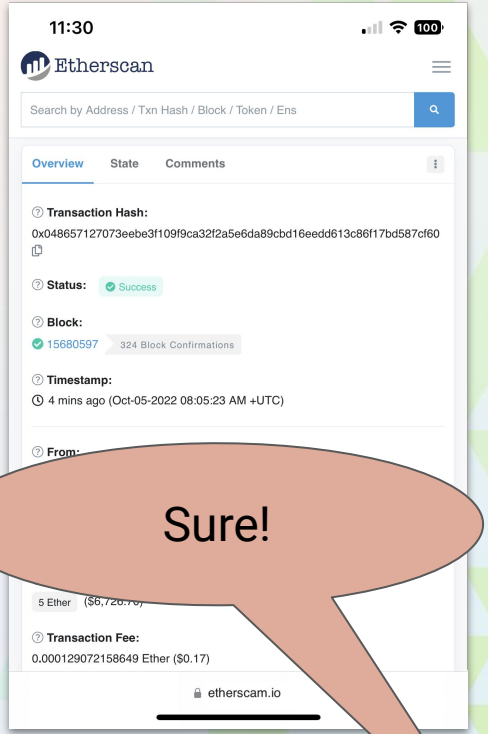


Sure!

Funds protected by HW wallet

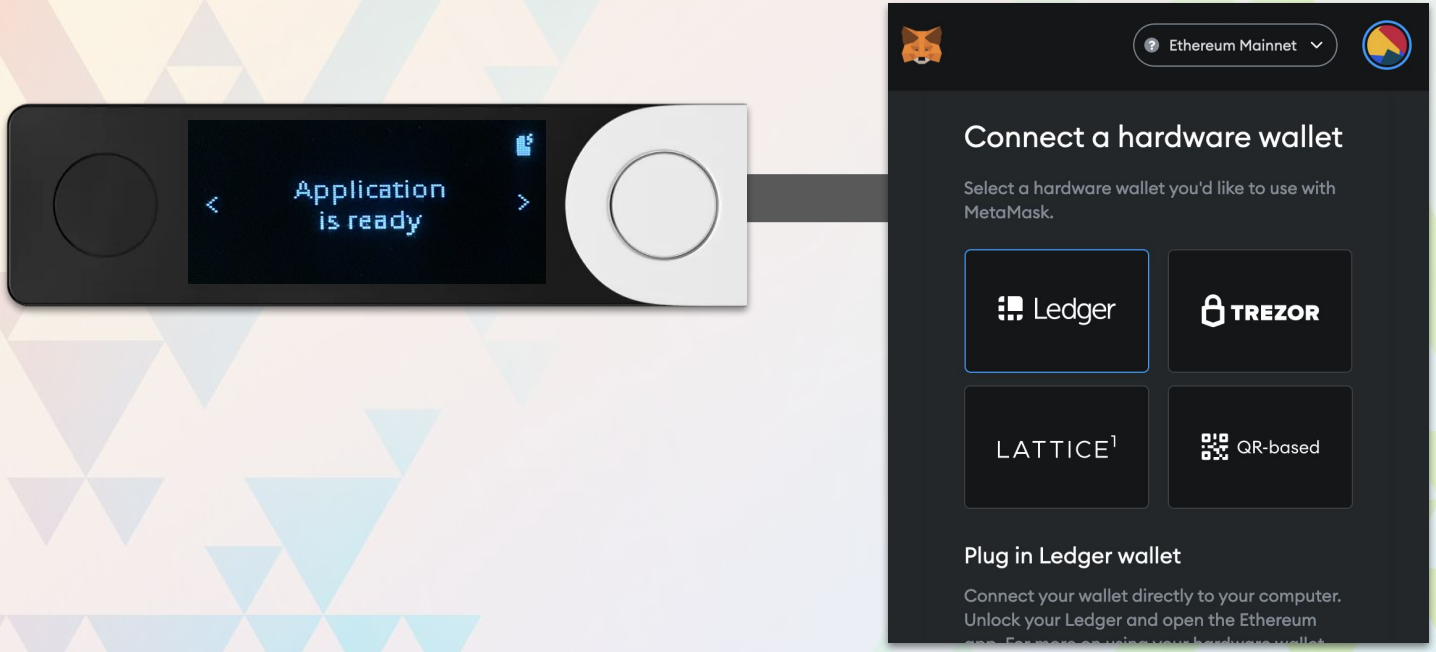


Can I use your laptop?

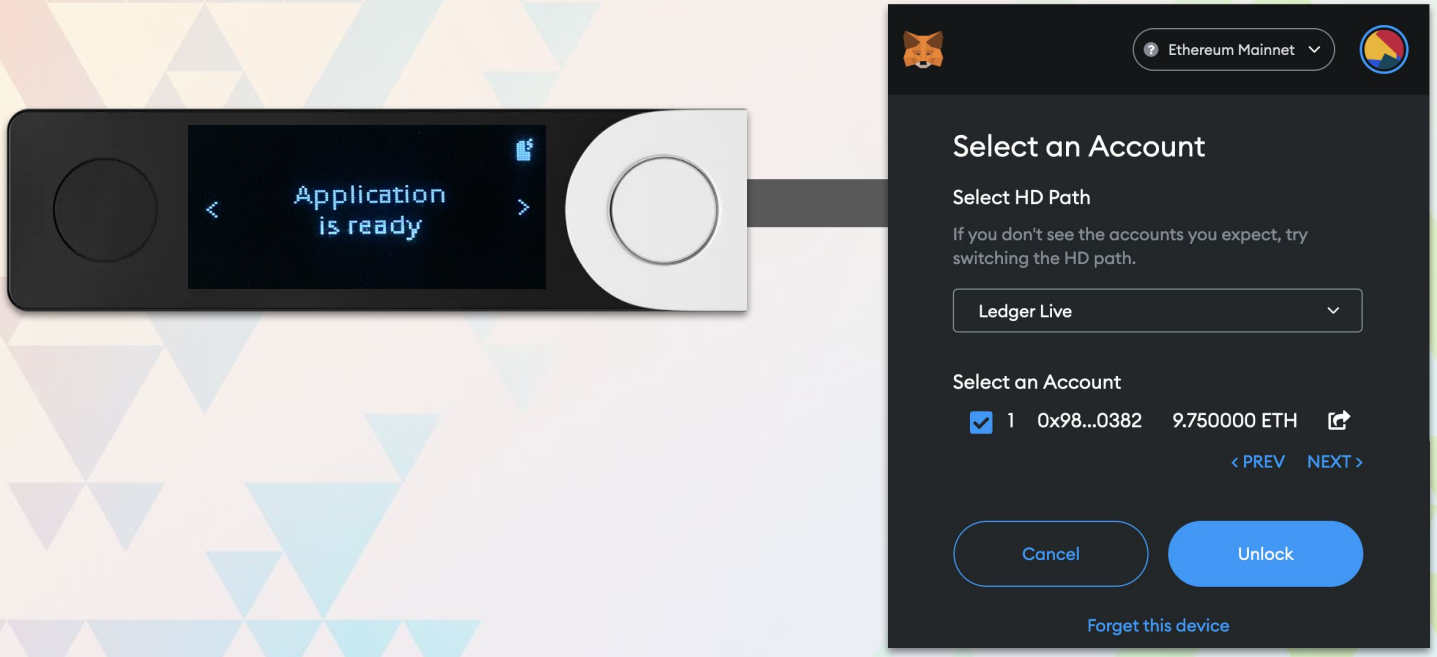


Sure!

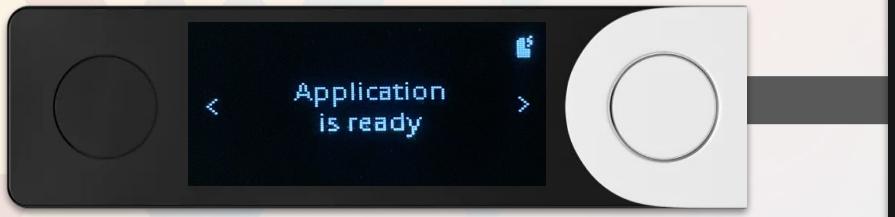
Funds protected by HW wallet

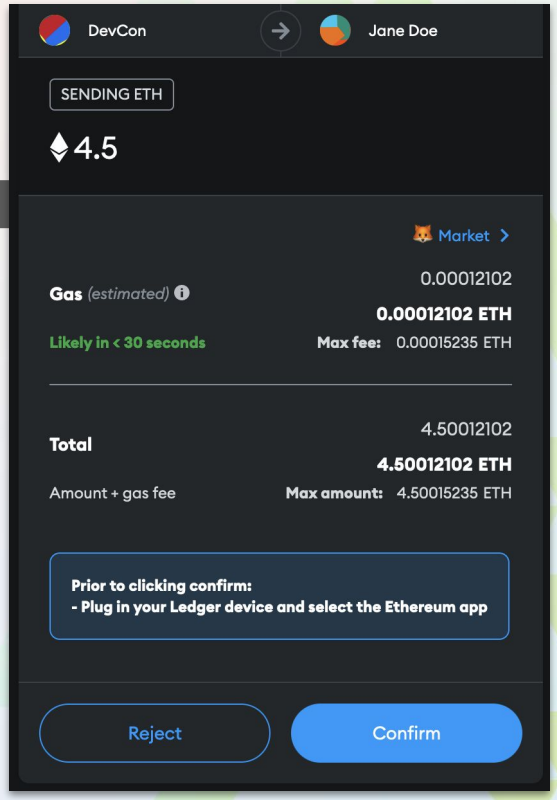


Funds protected by HW wallet

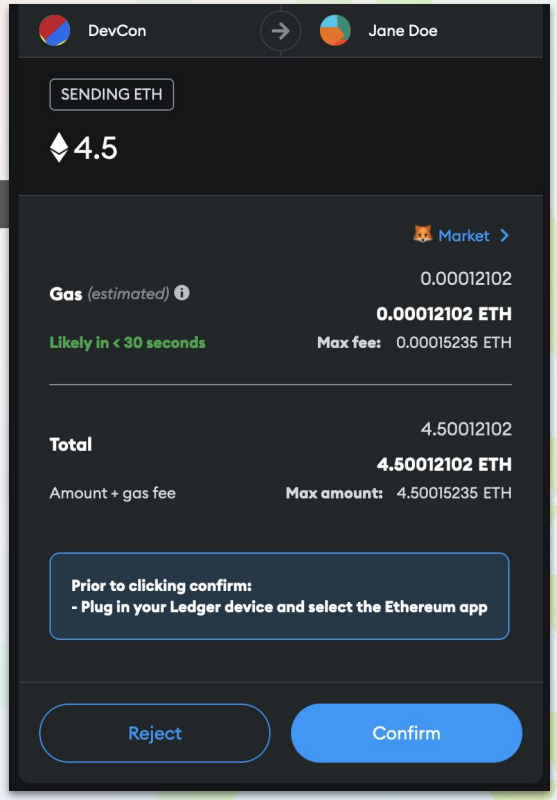


Funds protected by HW wallet

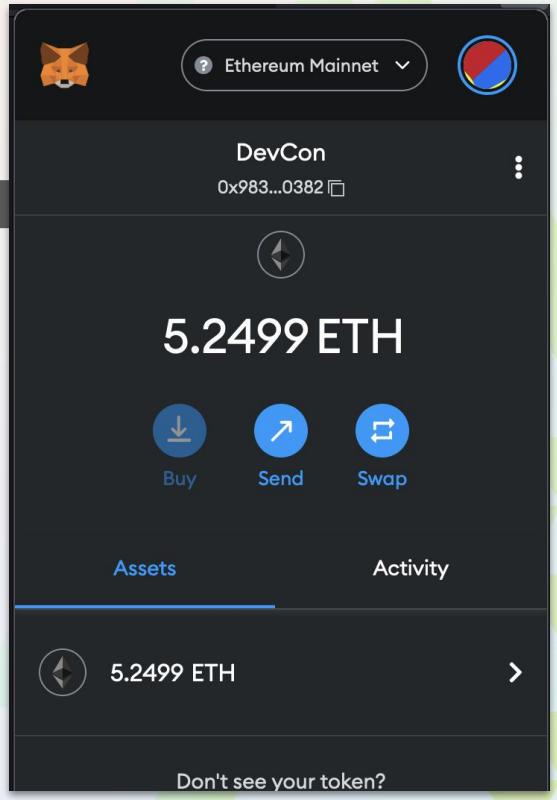




Funds protected by HW wallet

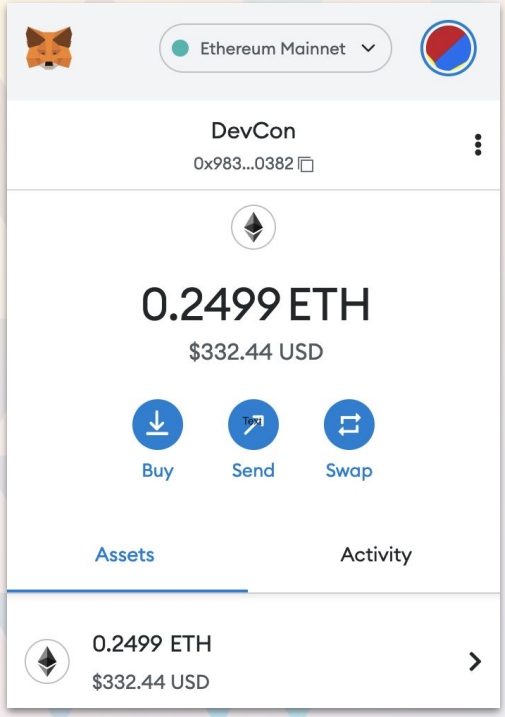


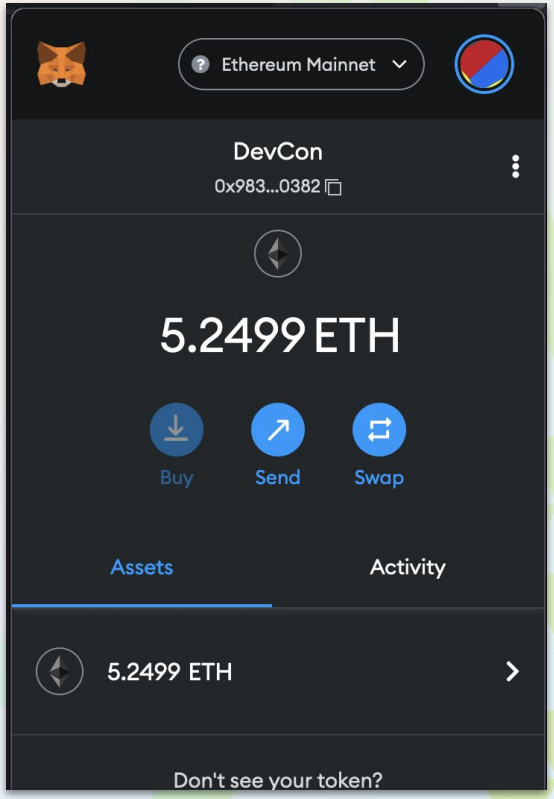
Funds protected by HW wallet



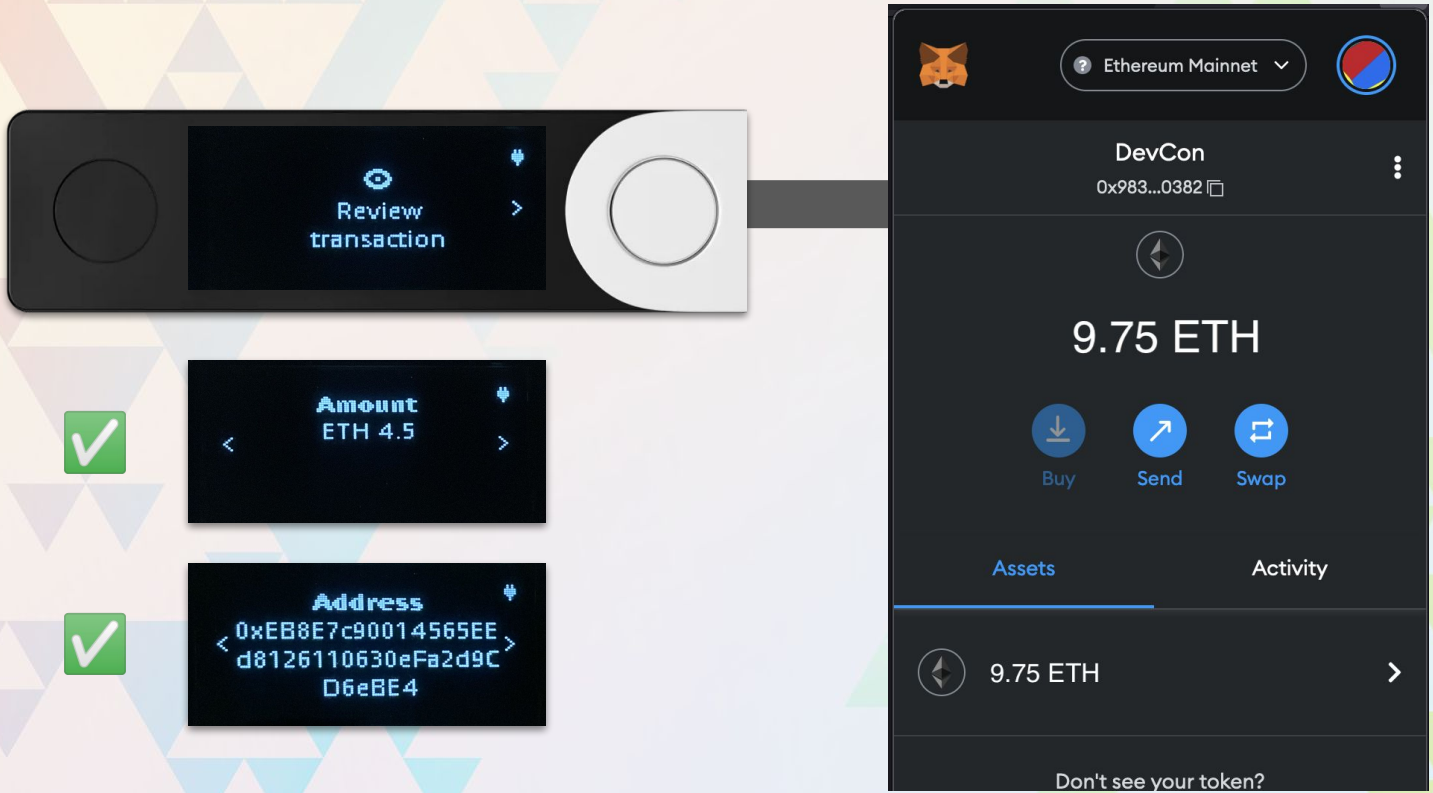
Thanks for the donation!

Some time later...





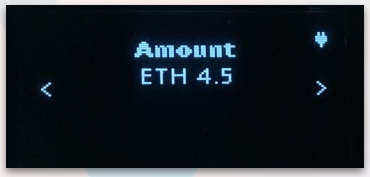
What went wrong?



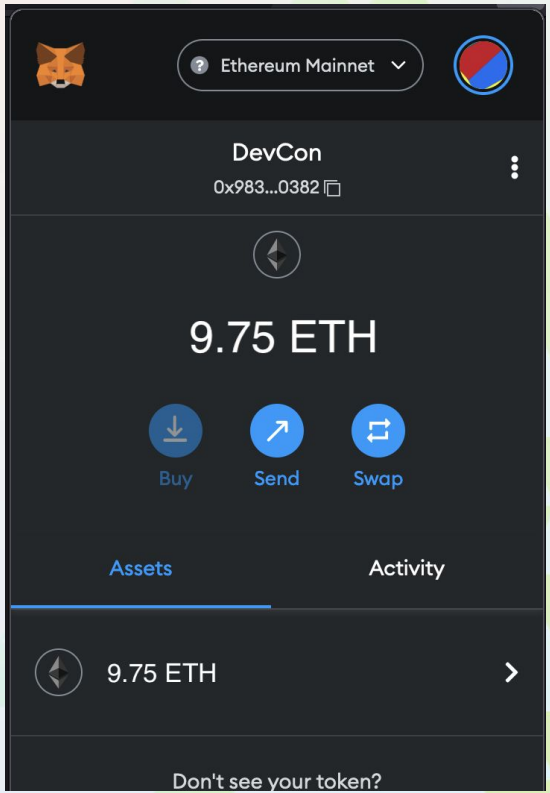
What went wrong?

❌

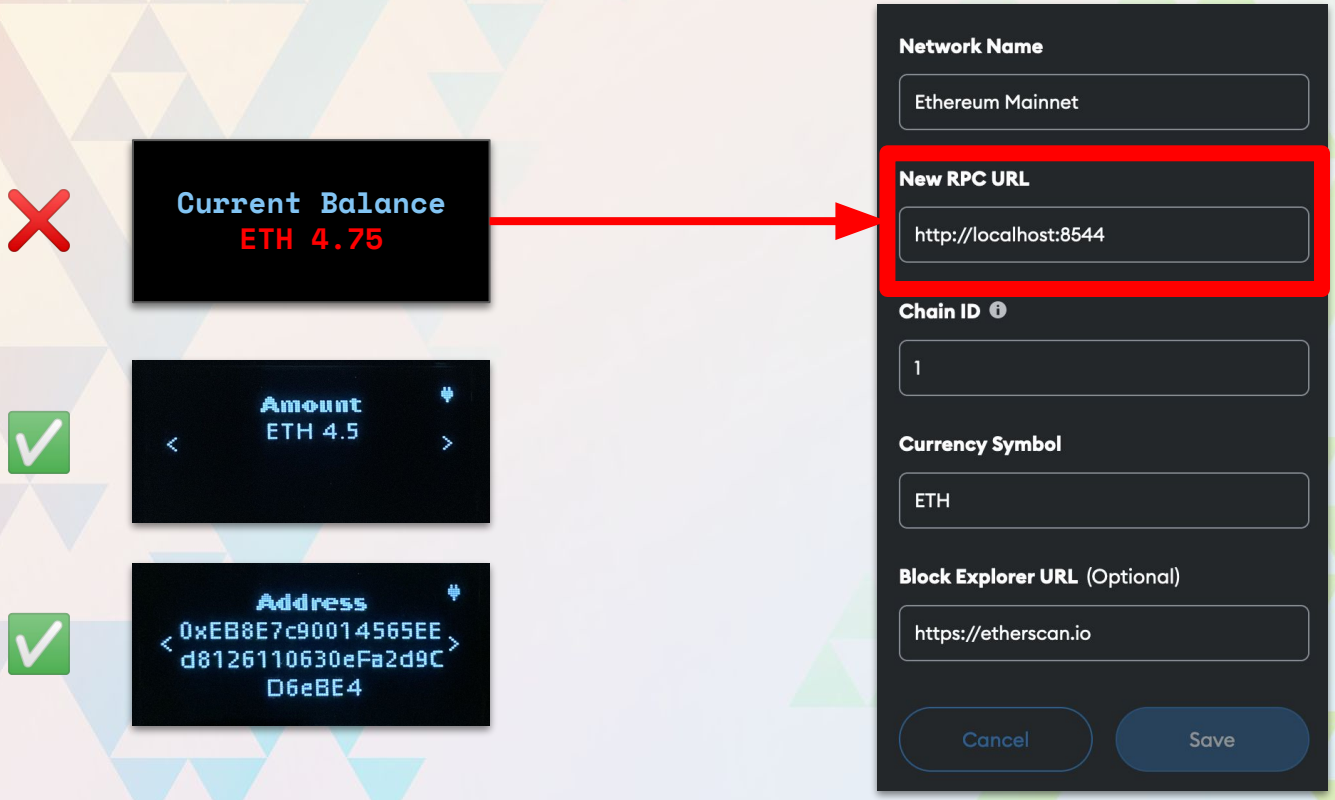
✅



✅



What went wrong?



Obtaining an ETH balance

Request

```
method: "eth_getBalance"
params: [...]
    0: "0x983260467a0d5c0dc02c031f653a645751c90382"
    1: "0xEF49CB"
```

Address

Block number

Obtaining an ETH balance

Request

```
method: "eth_getBalance"
params: [...]
    0: "0x983260467a0d5c0dc02c031f653a645751c90382"
    1: "0xEF49CB"
```

Address

Block number

Response

```
result: "0x48db5817212fe4a0"
```

Balance

~5.2499 ETH

Obtaining an ETH balance

```
proxy.rpc("eth_getBalance") do(address: Address, quantityTag: string) -> HexQuantityStr:
  let balance = await rpcClient.eth_getBalance(address, quantityTag)
  if $address == "0x983260467a0d5c0dc02c031f653a645751c90382":
    return encodeQuantity(balance + 5000000000000000000.u256)  # Add 5 ETH
  return encodeQuantity(balance)
```

Section 2

Verifying Ethereum data

Ethereum data

A Token balance

B NFT owner

C ETH balance

D DeFi exchange rate

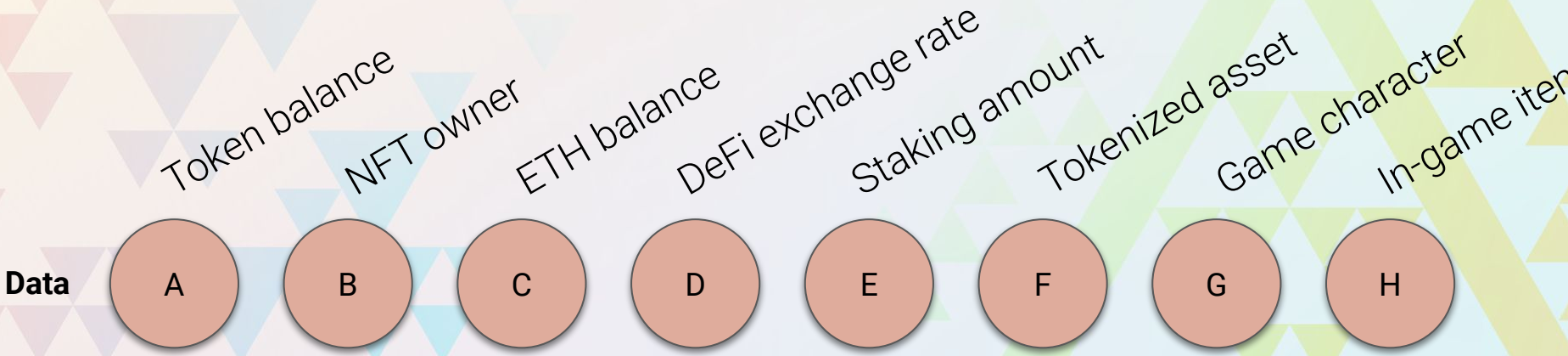
E Staking amount

F Tokenized asset

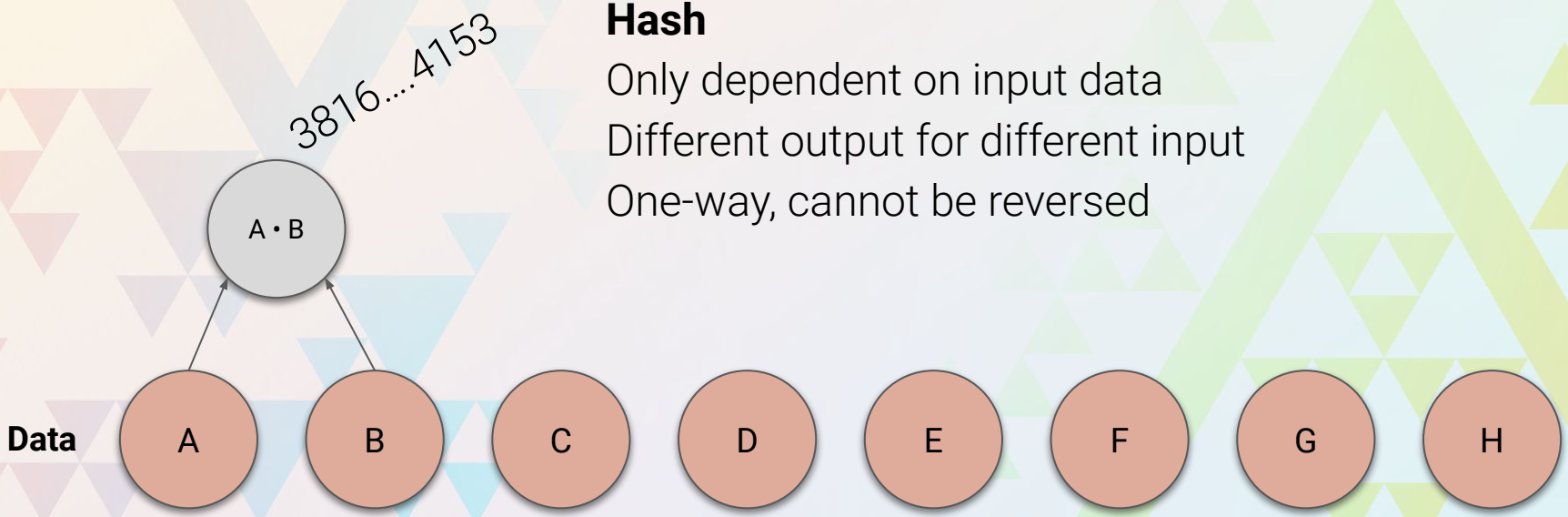
G Game character

H In-game item

Ethereum data



Ethereum data



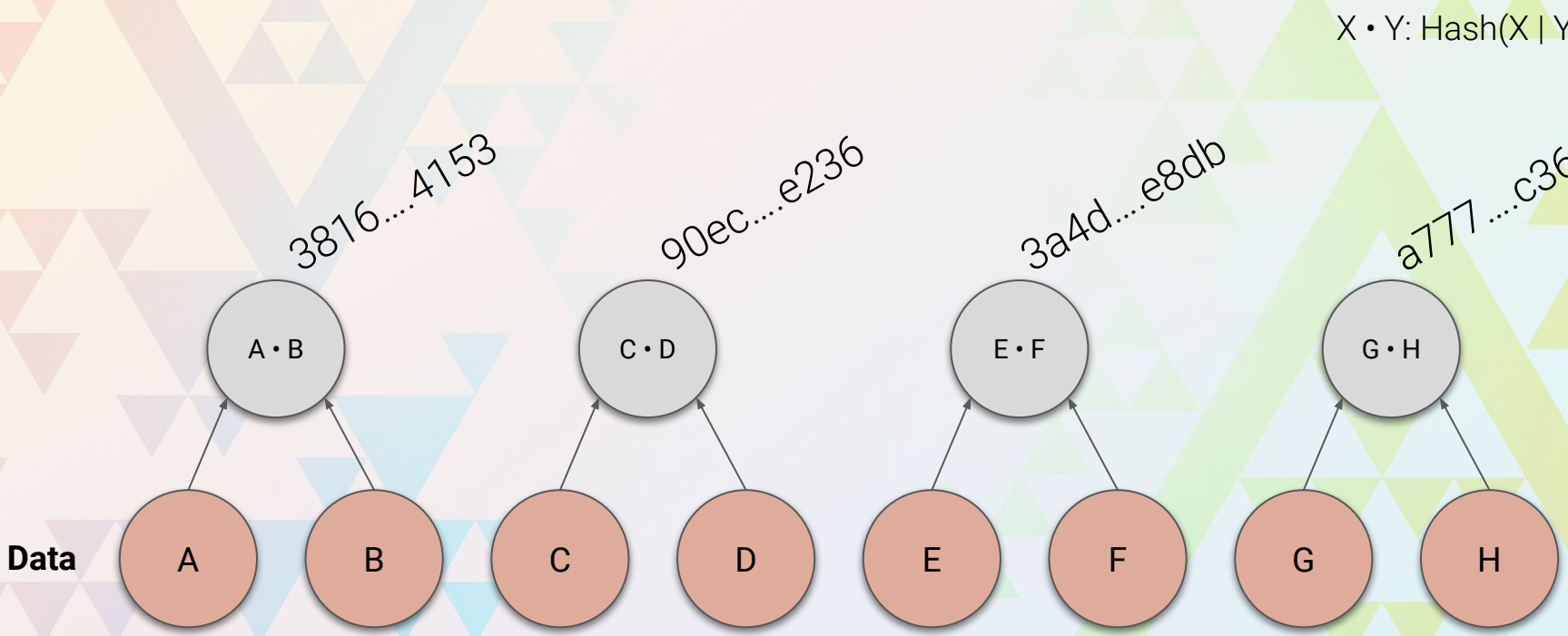
Hash

Only dependent on input data

Different output for different input

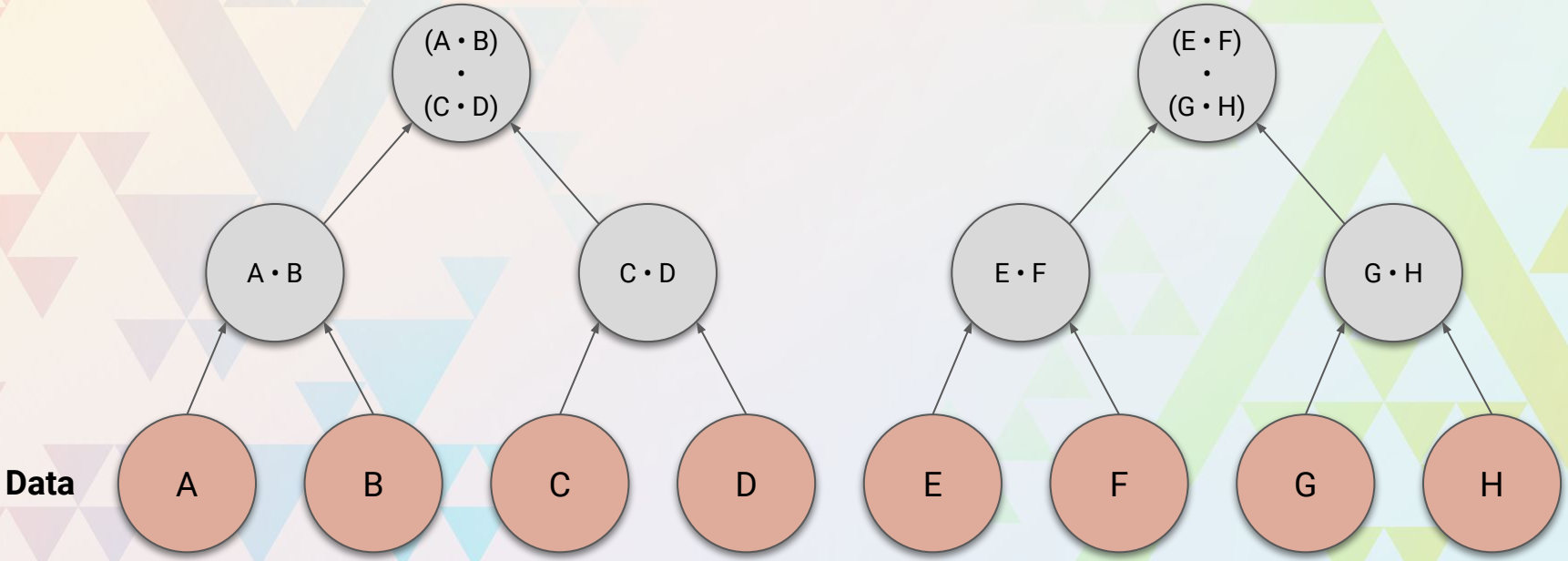
One-way, cannot be reversed

Merkle trees

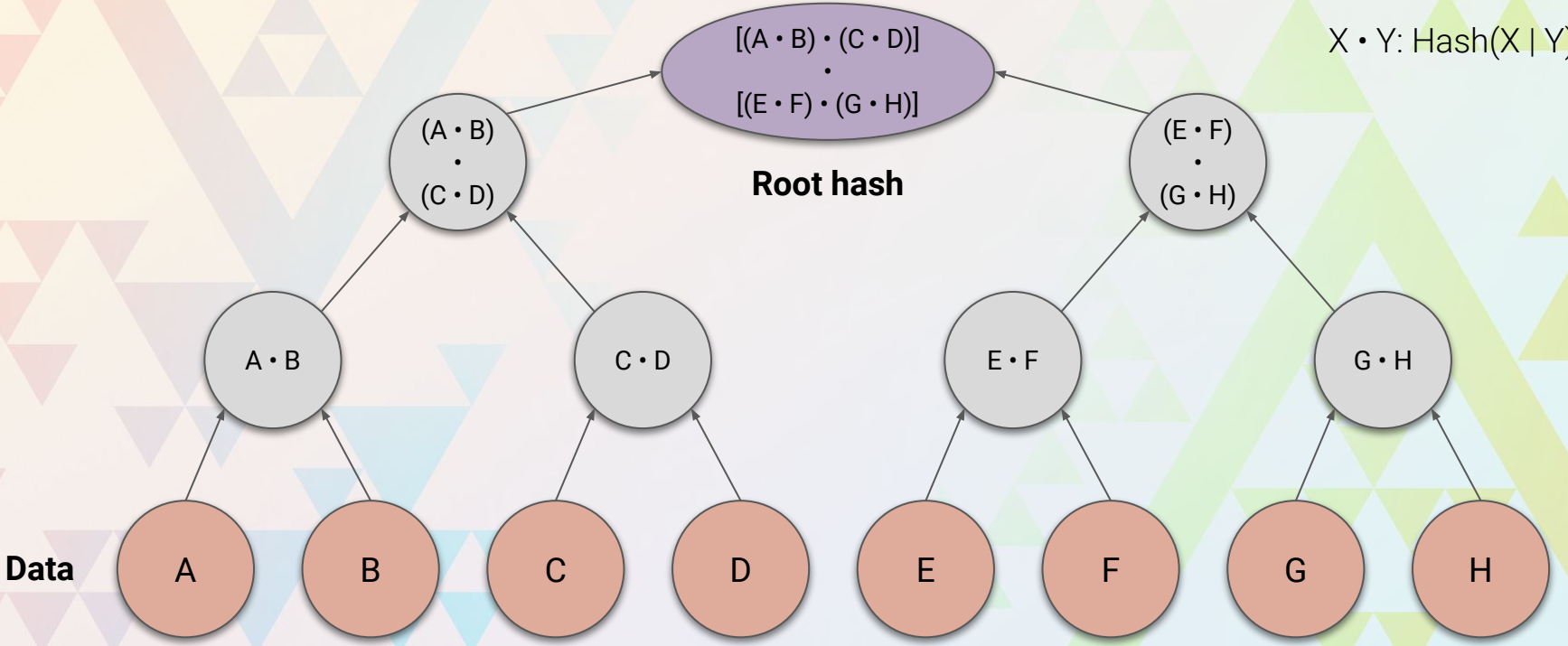


Merkle trees

X • Y: Hash(X | Y)

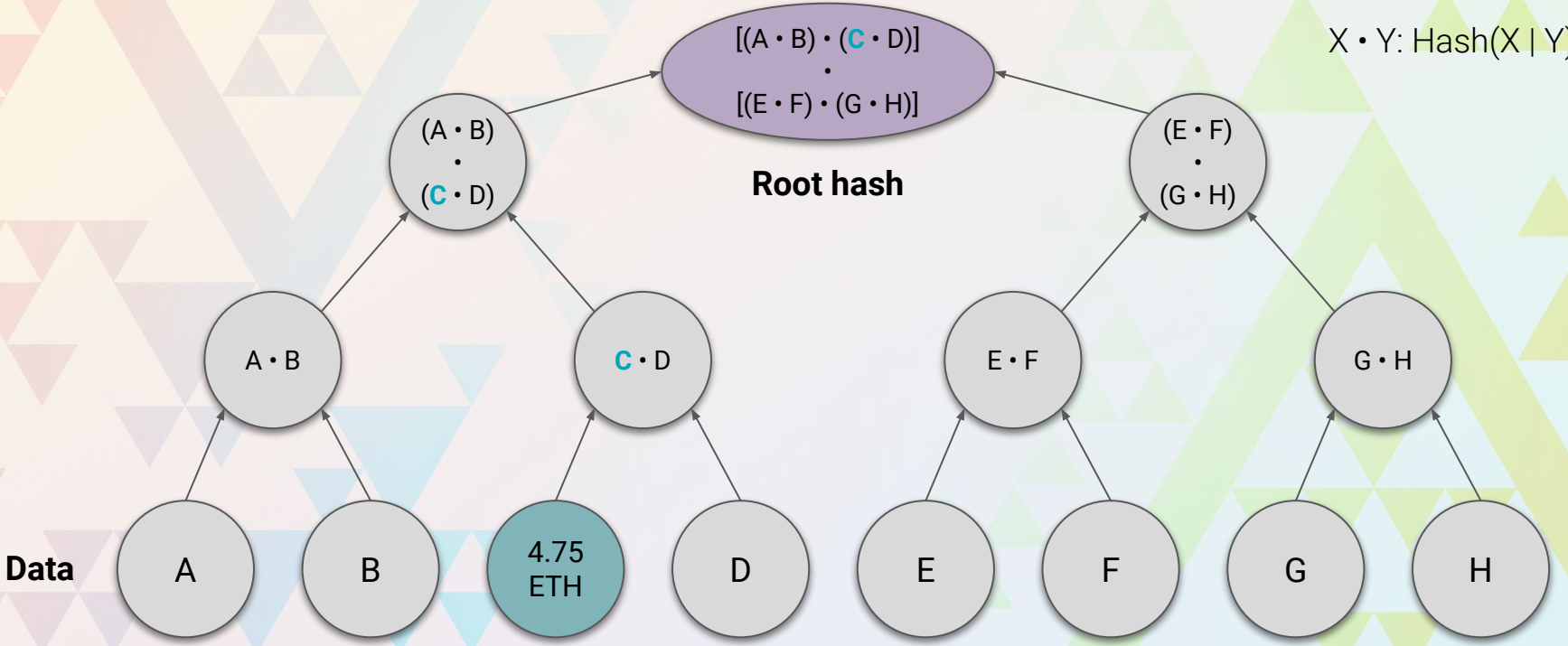


Merkle trees

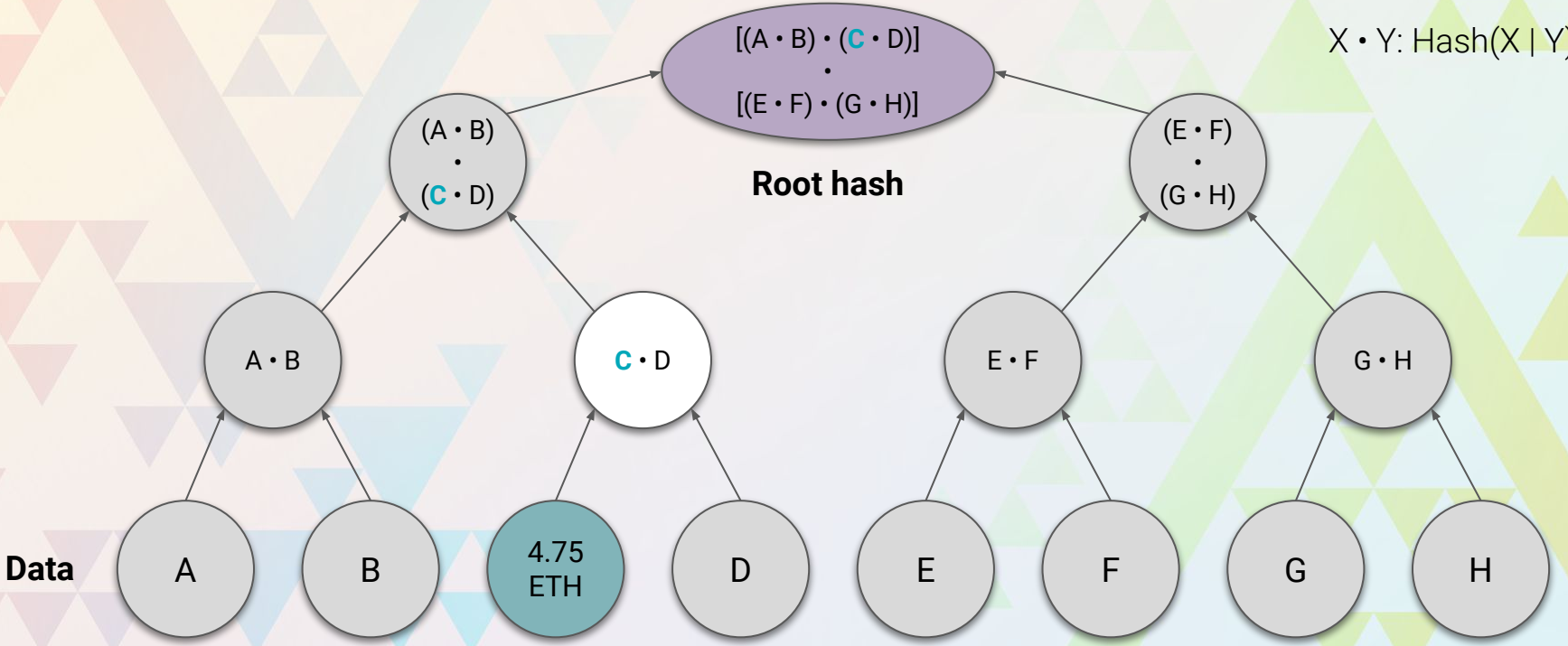


X • Y: Hash(X | Y)

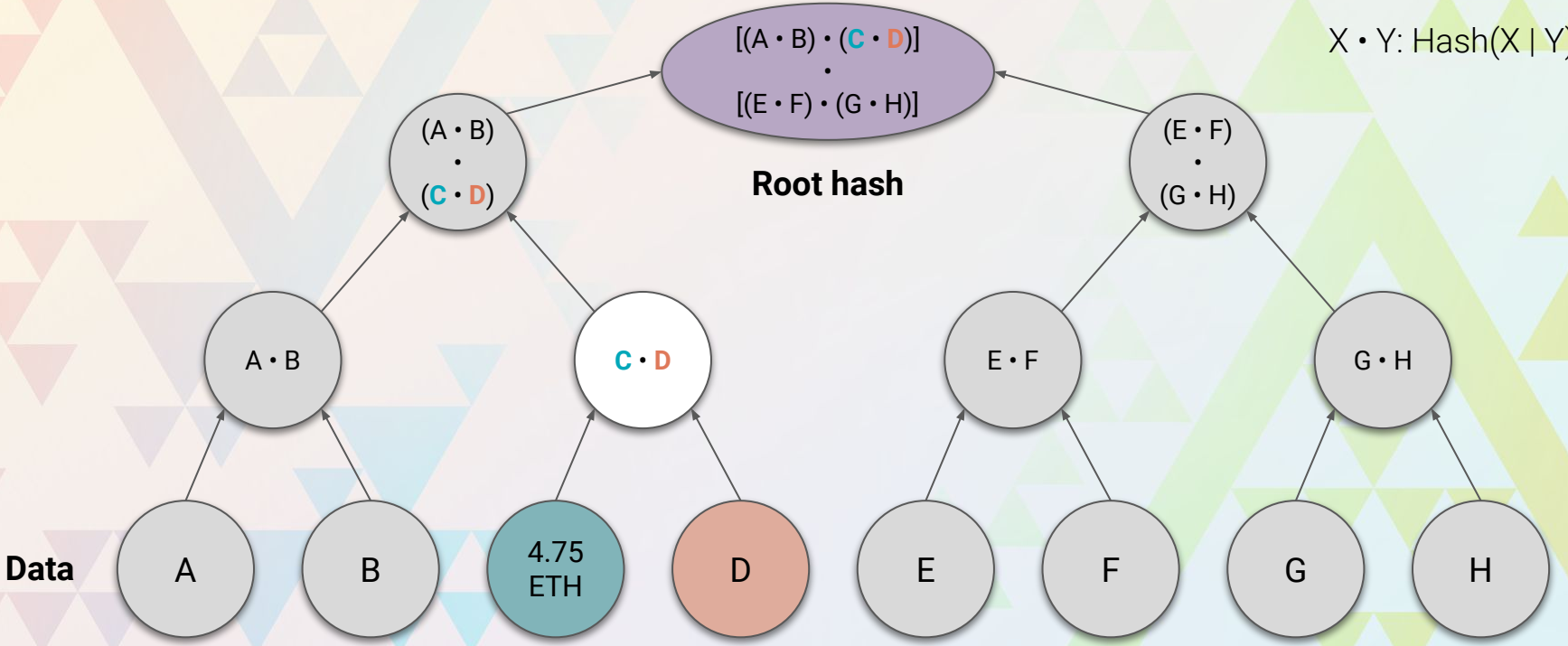
Merkle proof



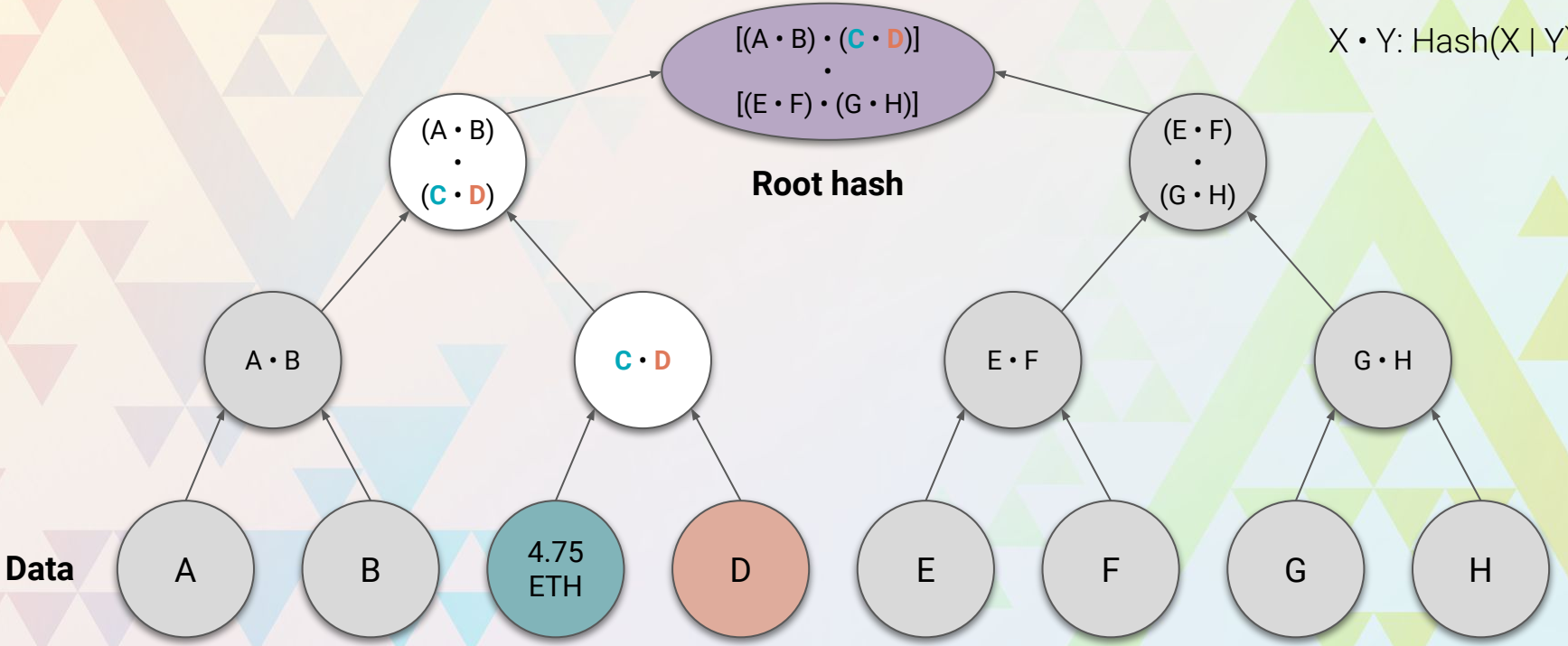
Merkle proof



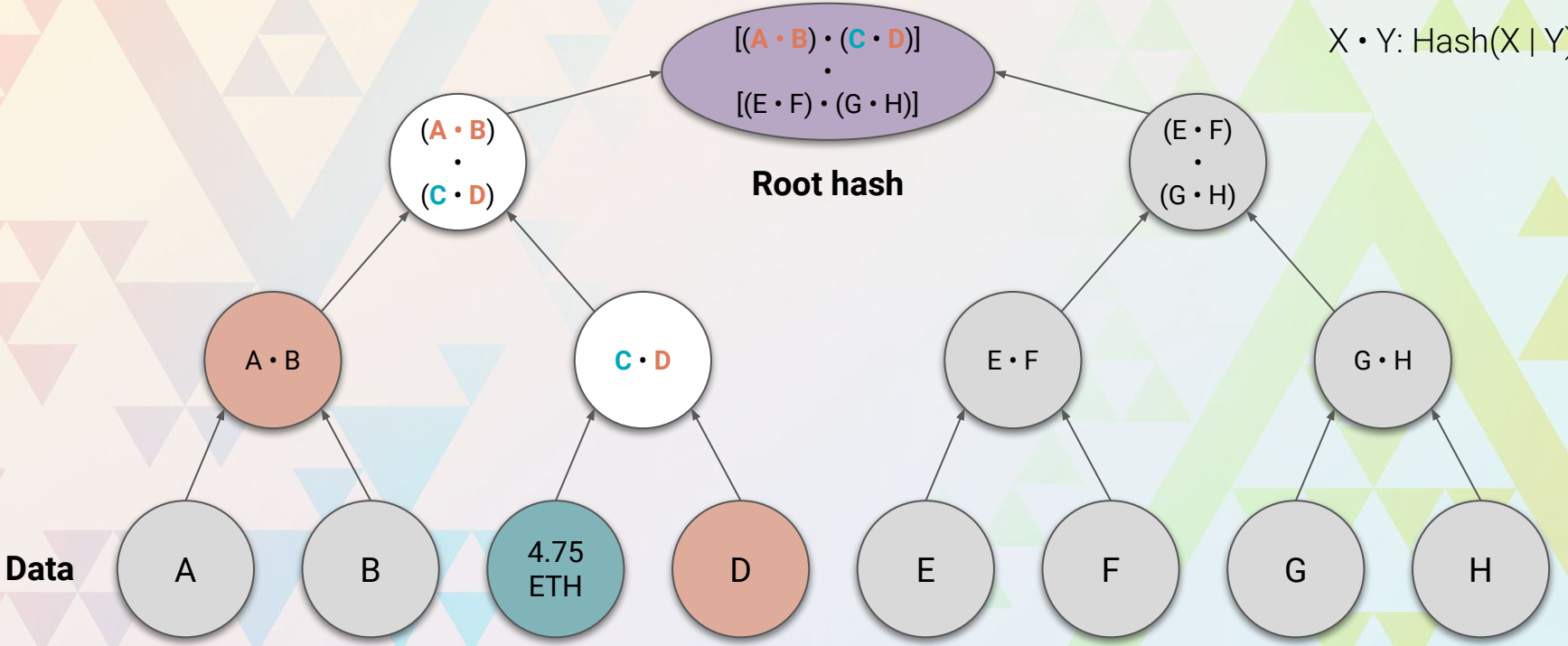
Merkle proof



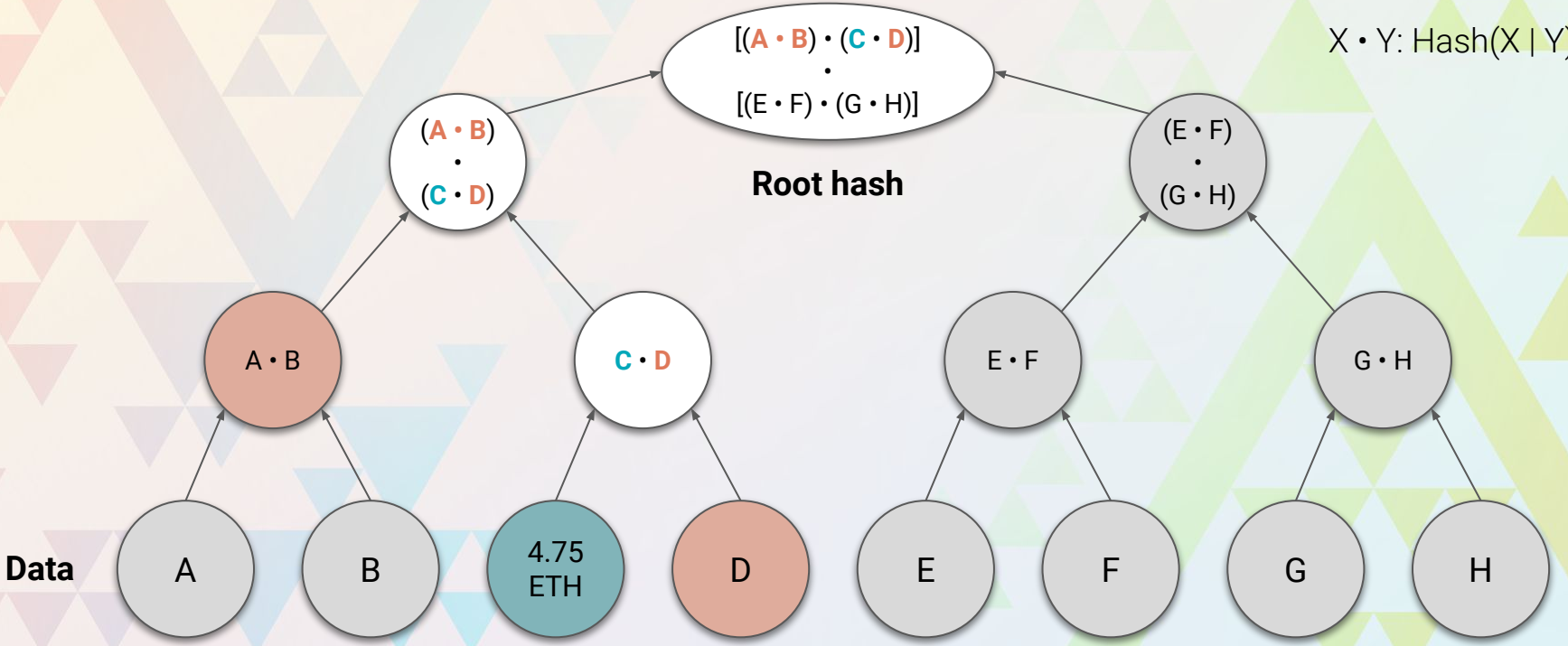
Merkle proof



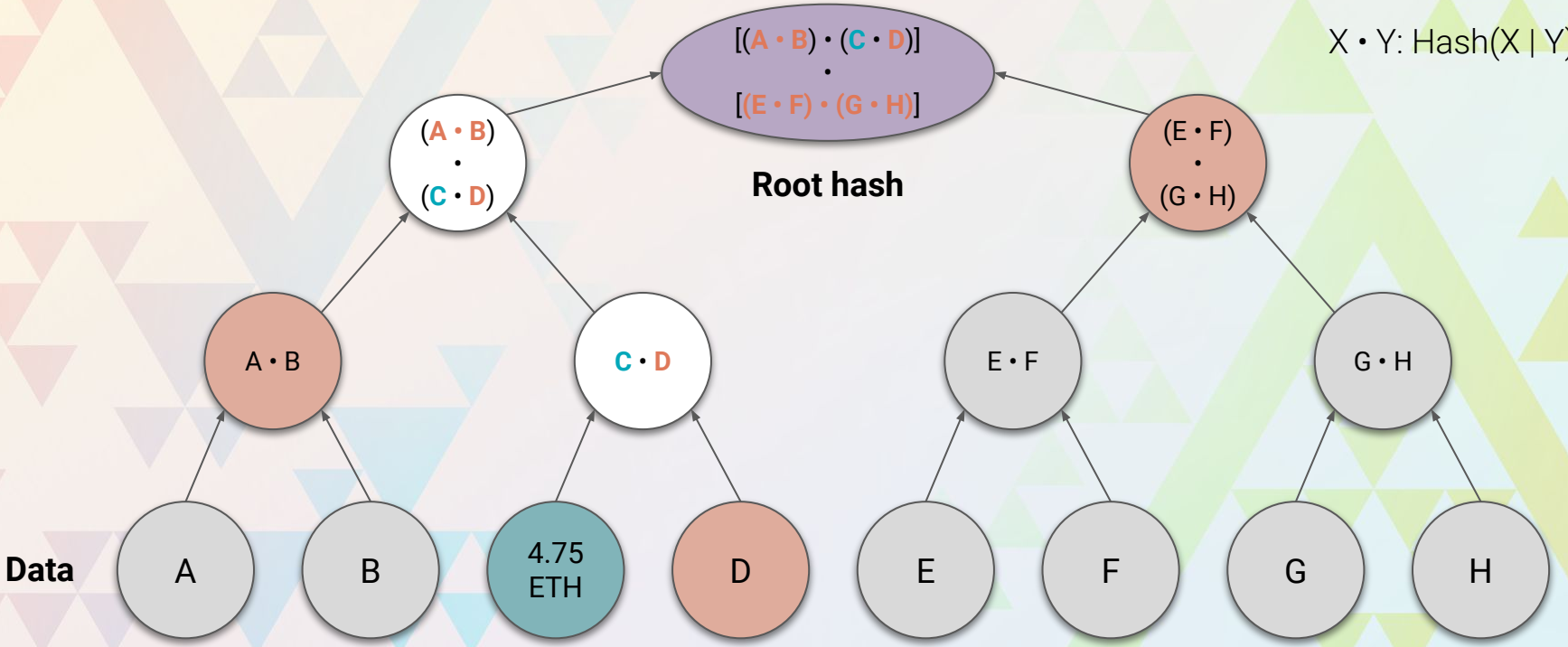
Merkle proof



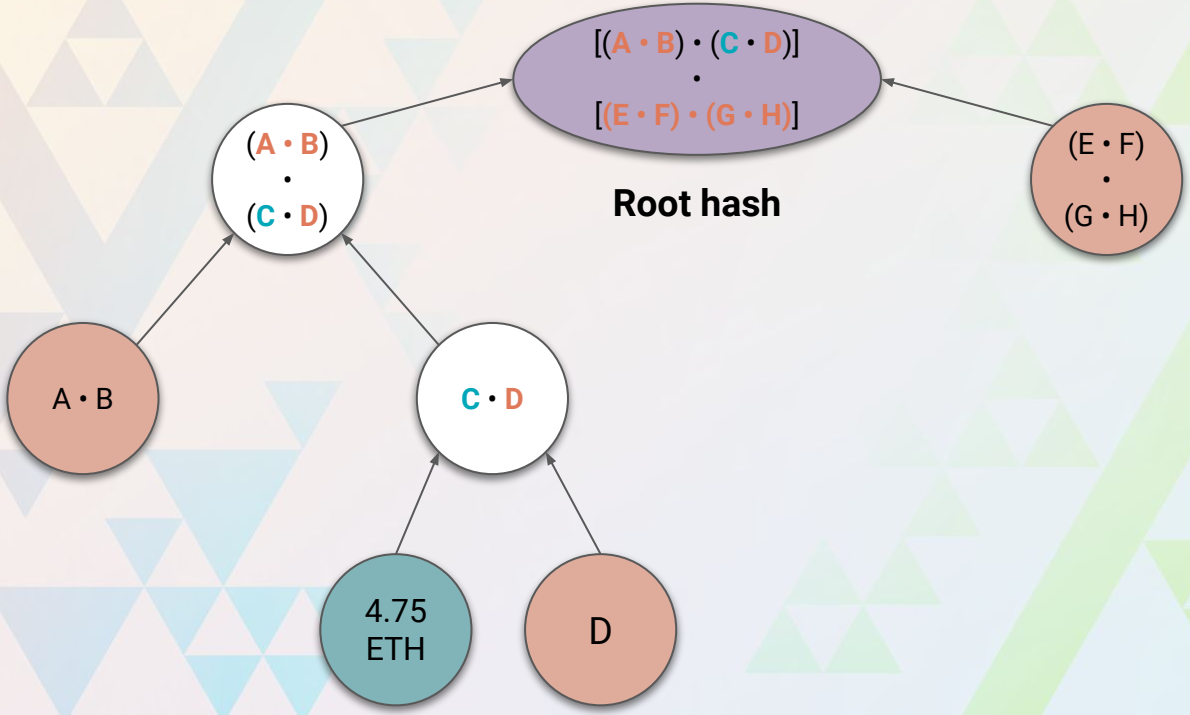
Merkle proof



Merkle proof



Merkle proof



X • Y: Hash(X | Y)

eth_getProof (EIP 1186)

Request

```
method: "eth_getProof"
params: [...]
    0: "0x983260467a0d5c0dc02c031f653a645751c90382"
    1: []   (for token balances, NFT owners, ...)
    2: "0xEF49CB"
```

Address

Storage slots

Block number

eth_getProof (EIP 1186)

Response

Merkle proof

Balance

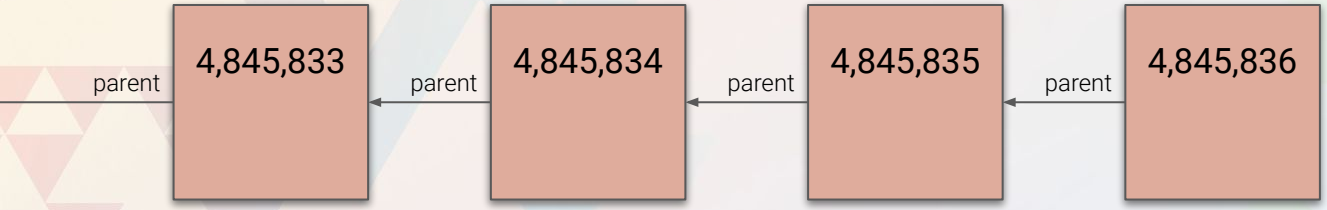

```
"result": {
  "accountProof": [
    "0xf90211a0...0d72d680",
    "0xf90211a0...b46e5180",
    "0xf90211a0...f4ef1b80",
    "0xf90211a0...fcbc1980",
    "0xf90211a0...25322c80",
    "0xf90211a0...f0748980",
    "0xf9013180...bdb1d380",
    "0xf8918080...80808080",
    "0xf86e9d20...5d85a470"
  ],
  "address": "0x983260467a0d5c0dc02c031f653a645751c90382",
  "balance": "0x377c694dc3be4a0",
  "codeHash": "0xc5d2460186f7233c927e7db2dcc703c0e500b653ca82273b7bfad8045d85a470",
  "nonce": "0x2",
  "storageHash": "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
  "storageProof": []
}
```


Still need the root hash
to verify against!

Section 3

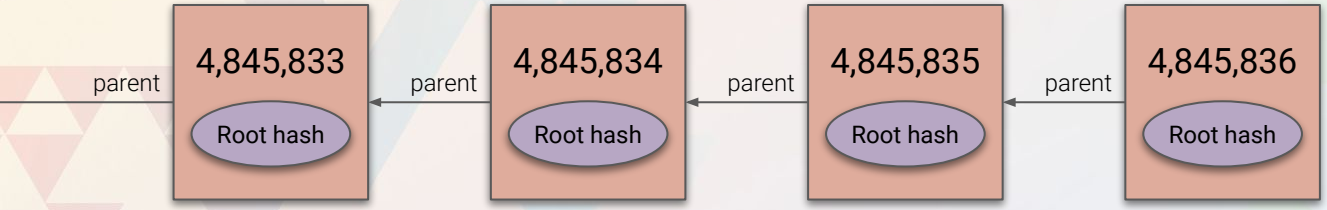
Obtaining the root hash

Where is the root hash?



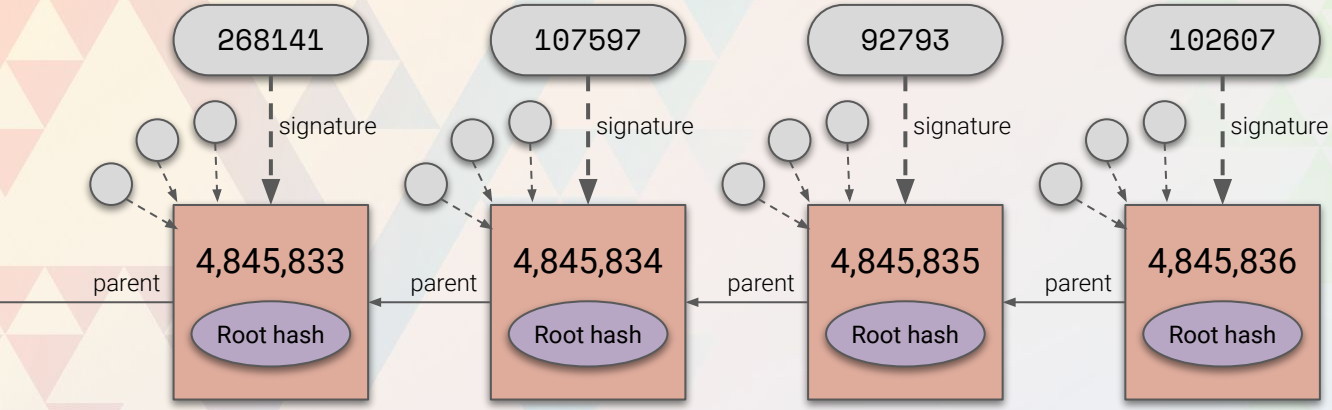
Beacon chain

Where is the root hash? (Merge 🐼)



Beacon chain

How to verify?



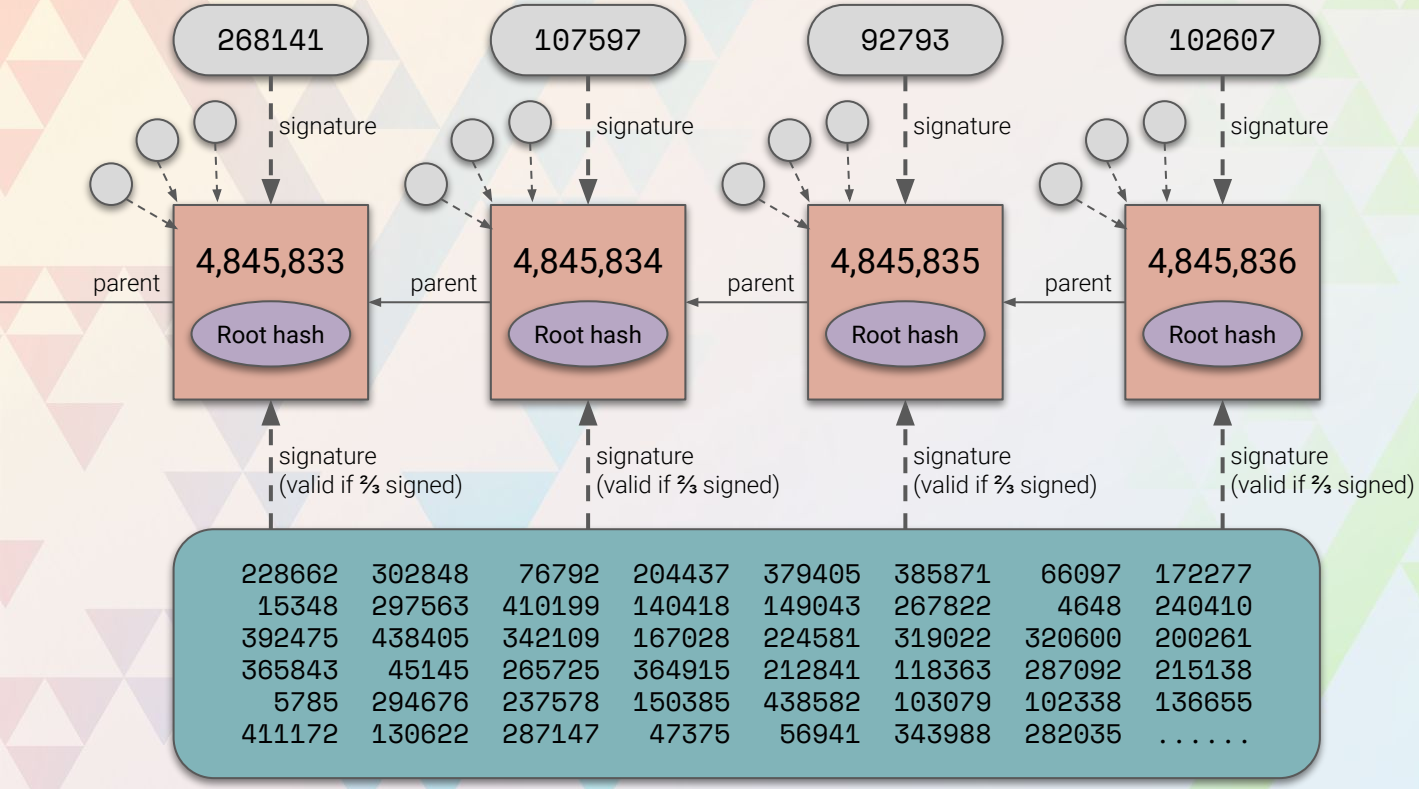
Proposing validator

Attestations

Beacon chain

There are **>440k** validators
Need **GBs** of data to verify

How to verify? (Altair)



Proposing validator

Attestations

Beacon chain

Sync committee
(512 validators)

How to get the sync committee?

```
390290  392522  356403  202586  184113  309259  129131  414668
398917   33891  108733  209678  255003   30286  123147  293629
131888  209269  323338  298622  305191  369160  211704  280926
255891  256646   89218  167832  343267  137560  245864  415896
312139  188521  159437  345790  339136  265386  243022  301945
393913   73431  233645  144754  218787  110930  133158  ......
```

Previous sync committee

next sync committee
(re-elected daily)

```
228662  302848   76792  204437  379405  385871   66097  172277
 15348  297563  410199  140418  149043  267822    4648  240410
392475  438405  342109  167028  224581  319022  320600  200261
365843   45145  265725  364915  212841  118363  287092  215138
  5785  294676  237578  150385  438582  103079  102338  136655
411172  130622  287147   47375   56941  343988  282035  ......
```

Sync committee
(512 validators)

Light client data

Light client data

<25 KB

<25 KB each
(re-elected ~daily)

<350 bytes

Light client data

<25 KB	**<25 KB** each (re-elected ~daily)	**<350 bytes**

REST ethereum/beacon-APIs #247

libp2p ethereum/consensus-specs > specs > altair > light-client

Portal ethereum/portal-network-specs #166

Light client data

<25 KB	**<25 KB** each (re-elected ~daily)	**<350 bytes**

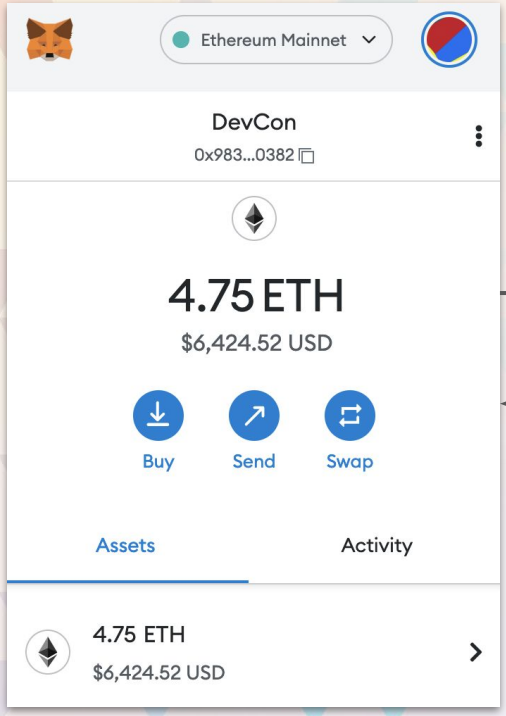
With ⅔ signature threshold: **<128 days** old data is safe to use

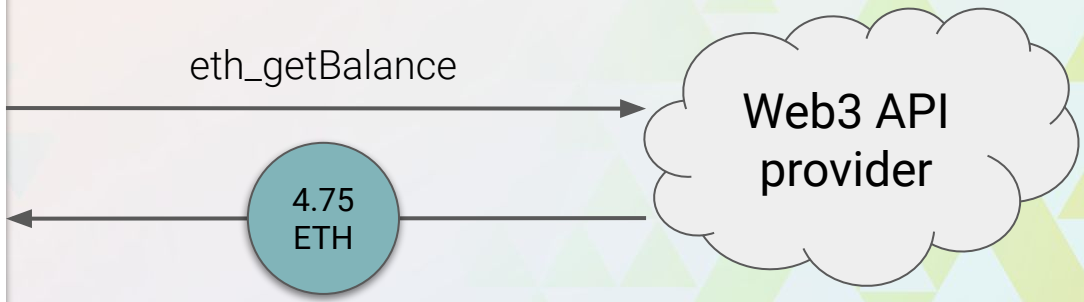
https://github.com/metacraft-labs/DendrETH/tree/main/docs/long-range-syncing

Section 4

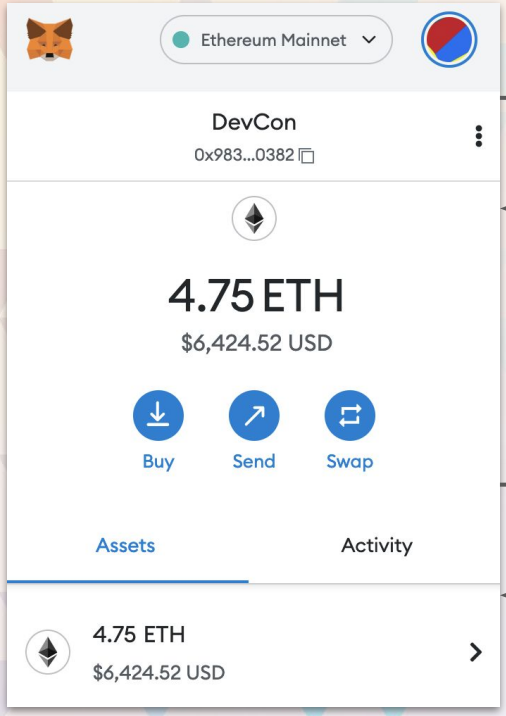
Combining it all together

Back to the wallet





Back to the wallet



getLightClientUpdatesByRange

Root hash

Light client data

Beacon API provider

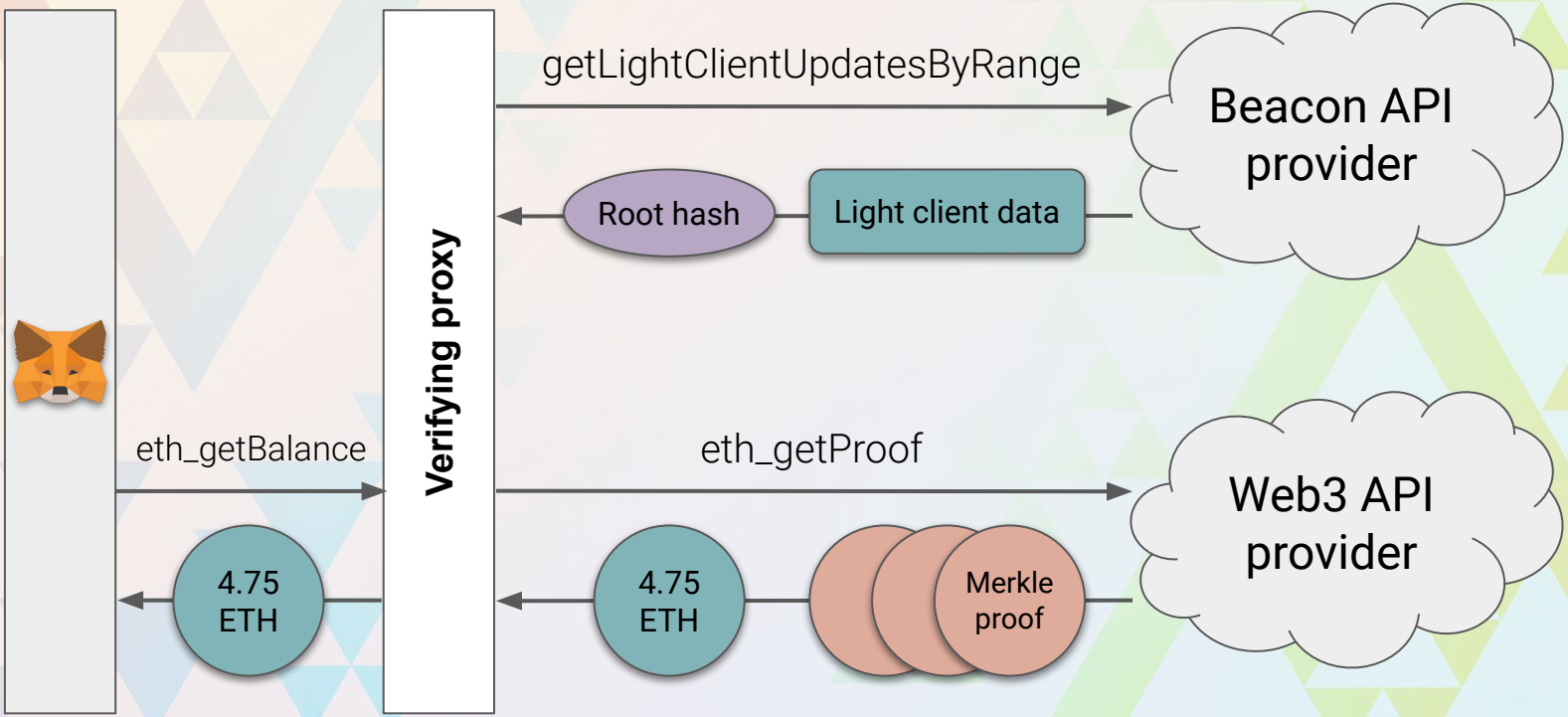
eth_getProof

4.75 ETH

Merkle proof

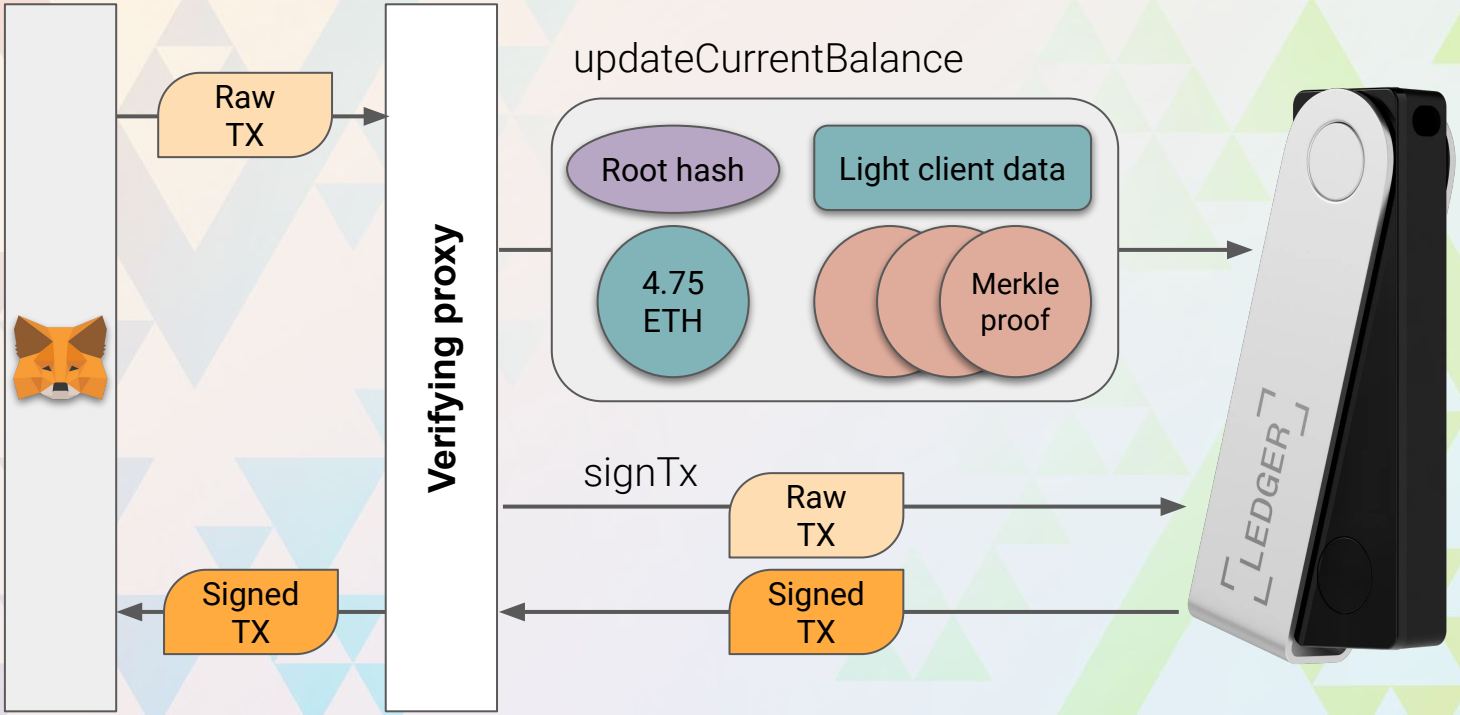
Web3 API provider

Without modifying the wallet

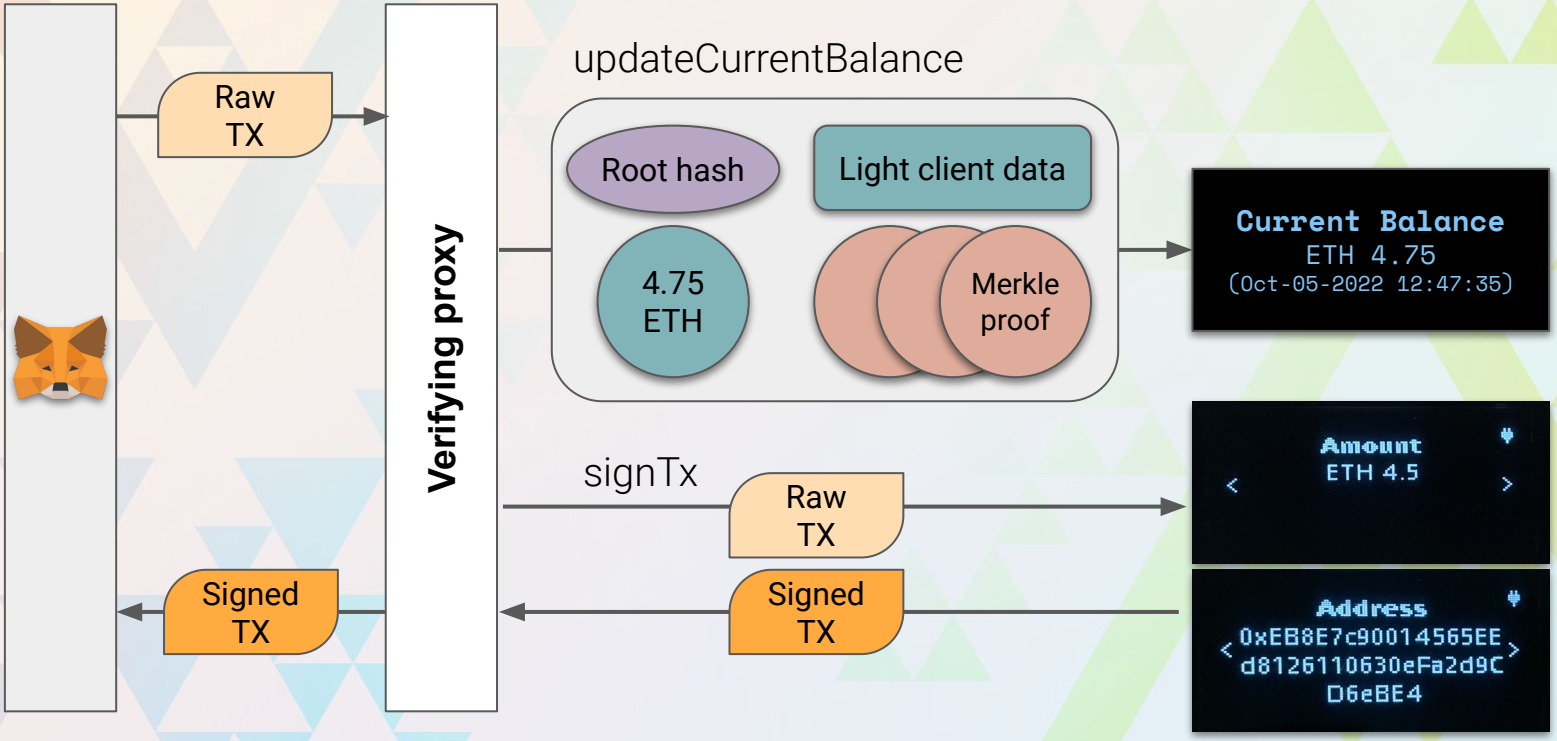


status-im/nimbus-eth1 > lc_proxy — shresthagrawal/kevlar

Signing a transaction



Signing a transaction



What else?

Improved full node sync
Start from an older but
widely agreed-on checkpoint

Decentralized wallet
Use light client CL and LES EL
to monitor relevant transactions

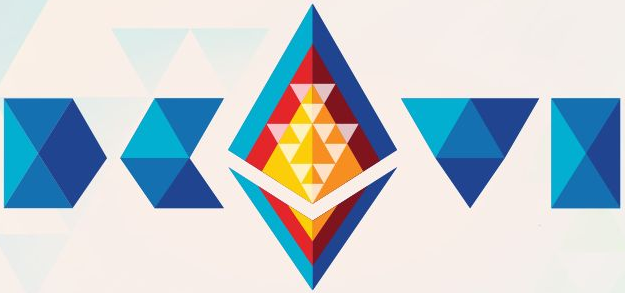
What else?

Improved full node sync
Start from an older but
widely agreed-on checkpoint

Decentralized wallet
Use light client CL and LES EL
to monitor relevant transactions

L2 bridge
Ensure that oracle nodes
can only submit valid data

Internet of Things
Rent a public bicycle
via on-chain rental pass



Thank you!

Etan Kissling

Nimbus, Status R&D GmbH

etan@status.im

@etan_status

#light-clients