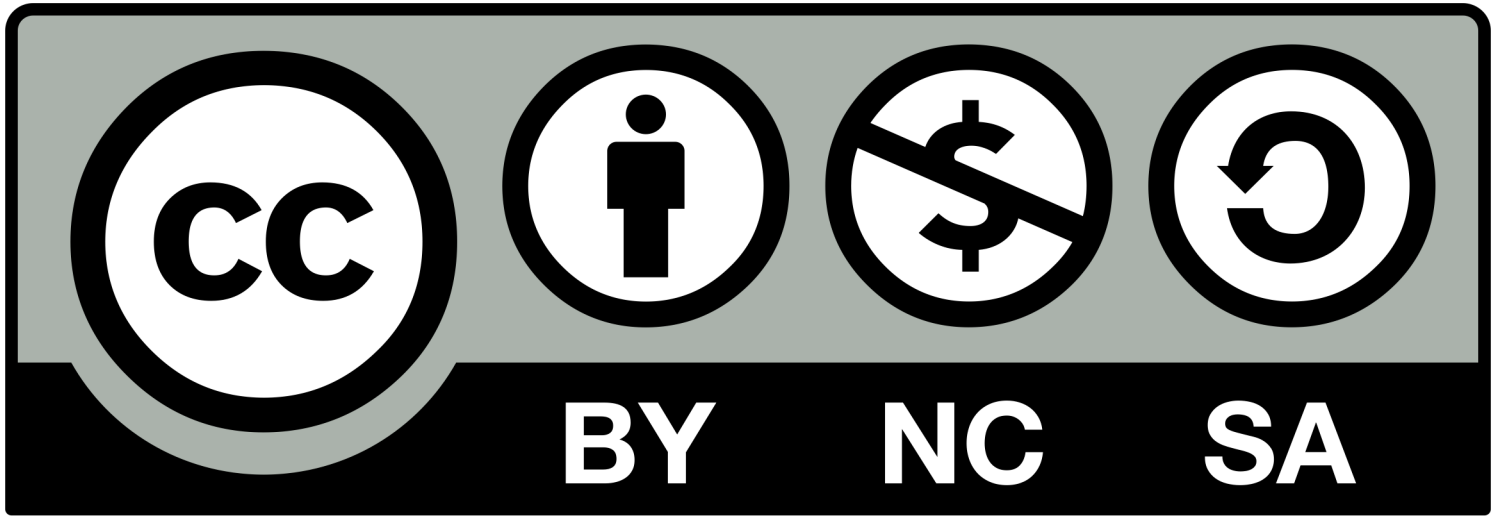
Data Anchoring on Bitcoin

FOR NOTARIZATION, DIGITAL ASSET, SECURITY, ...

Stéphane Roche

2019-11-17

CREATIVE COMMONS



ABOUT STEPHANE

2015

Work at Ledger (hardware wallet)

2016–2017

Work on Ethereum

Learn and play
Co-found non-profit organization Asseth
Contribute to the ERC20 Consensys smart contracts
Dether.io

2017–2019

Found Bitcoin Studio

Bitcoin education, dev, consulting

@janakaSteph on Twitter
rstephane@protonmail.com

OUTLINE

1. ANCHORING TECHNIQUES
2. TIMESTAMPING
3. DIGITAL ASSETS
4. OTHERS

1

ANCHORING TECHNIQUES

OP_RETURN

- The most widely used technique for data anchoring
- Instruction has been part of the scripting language since the first releases of Bitcoin
  - But considered non-standard (so difficult to reliably get relayed and mined)
- Became standard in March 2014 (Bitcoin Core 0.9.0)
- An output script containing OP_RETURN always evaluates to false
  - Provably unspendable
  - Provably prunable from UTXO set to avoid data storage schemes
- Consensus rules allow null data outputs up to max locking script size of 10,000 bytes

- Bitcoin Core policy rules enforce
  - To pay null data output exactly 0 satoshis
  - There must only be a single null data output per transaction (to avoid spam)

- Policy rule regarding max data size has changed over time
  - Bitcoin Core 0.9.x to 0.10.x, by default, relay and mine null data transactions up to 40 bytes
  - Bitcoin Core 0.11.x increases this default to 80 bytes
  - Bitcoin Core 0.12.0 defaults to relaying and mining null data outputs up to 83 bytes

- -datacarriersize Bitcoin Core configuration option
  - Allows you to set the maximum number of bytes in null data outputs that you will relay or mine

PAY/SIGN-TO-CONTRACT

- Elliptic curve commitments commit values in EC points

- Pay-to-Contract
  - Data to timestamp commited in tweaked public key
  - Spending the tweak prove that a certain value was known at the time it was put on the blockchain
  - Coin loss if loss of timestamp hash

- Sign-to-Contract
  - Data to timestamp commited in tweaked signature
  - Cost a transaction
  - No risk of coin loss

- Benefits
  - Do not affect transaction size, then cheaper
  - Anonymity, looks like regular transaction

OTHERS

- Fake addresses
  - Destination bitcoin address 20-byte field used for data
  - Doesn’t correspond to a private key, resulting UTXO can never be spent
- Vanity addresses
  - Brute force through keys until you get an address that encodes your data
- 1 of N Multisig
  - One real key and 32 bytes of data in the remaining N-1 keys
- nSequence input field
  - 32 bit integer, used in some occasions
- Inside script, push data then pop right away

- Terrible techniques, don't do that!

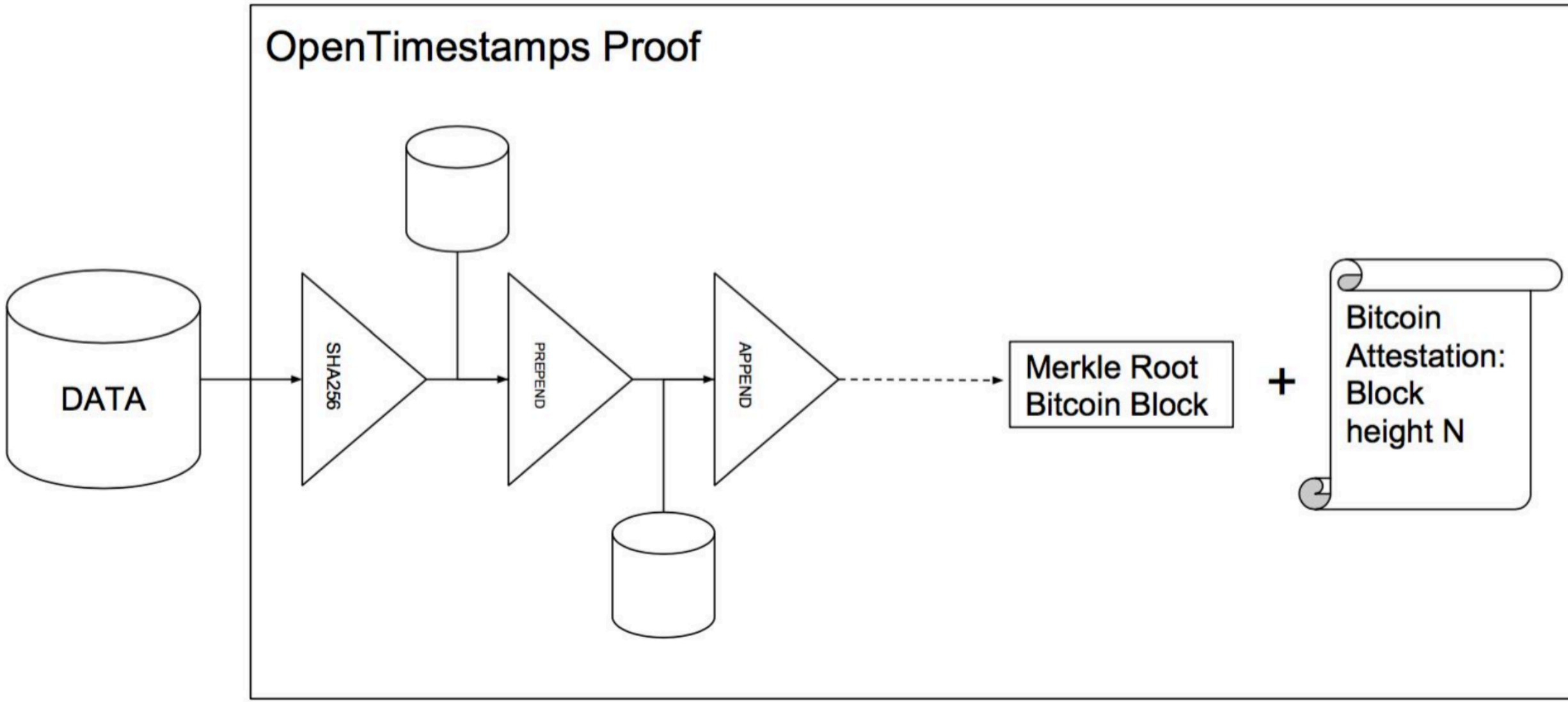
2 TIMESTAMPING

WHAT IS TIMESTAMPING?

- A timestamp proves that some data existed prior to some point in time
  - Also called "proof of existence"
  - Data integrity
- Data authentication, certification, ownership
  - True if the transaction is signed by the user
  - False if user uses a SaaS platform
  - Unless user anchor a signature
- No need for central authority
- Decentralized proofs that can't be erased or modified

OPENTIMESTAMPS

- Released in June 2012 by Peter Todd
- Defines a set of operations for creating provable timestamps and later independently verifying them
- A timestamp proof is a tree of commitment operations that are applied to a message in sequence
  - The root of the tree being the message, the edges the commitment operations (sha256, append, prepend, ...), and the leaves the attestations
  - These operations are the cryptographic path that proves that the message commits to a certain block header
- To verify a proof, the operations are applied in sequence to the message
  - The result, which should be the transaction merkle root, is checked to be equal to the one observed in the blockchain
- Doesn’t prove anything about whether conflicting data also existed

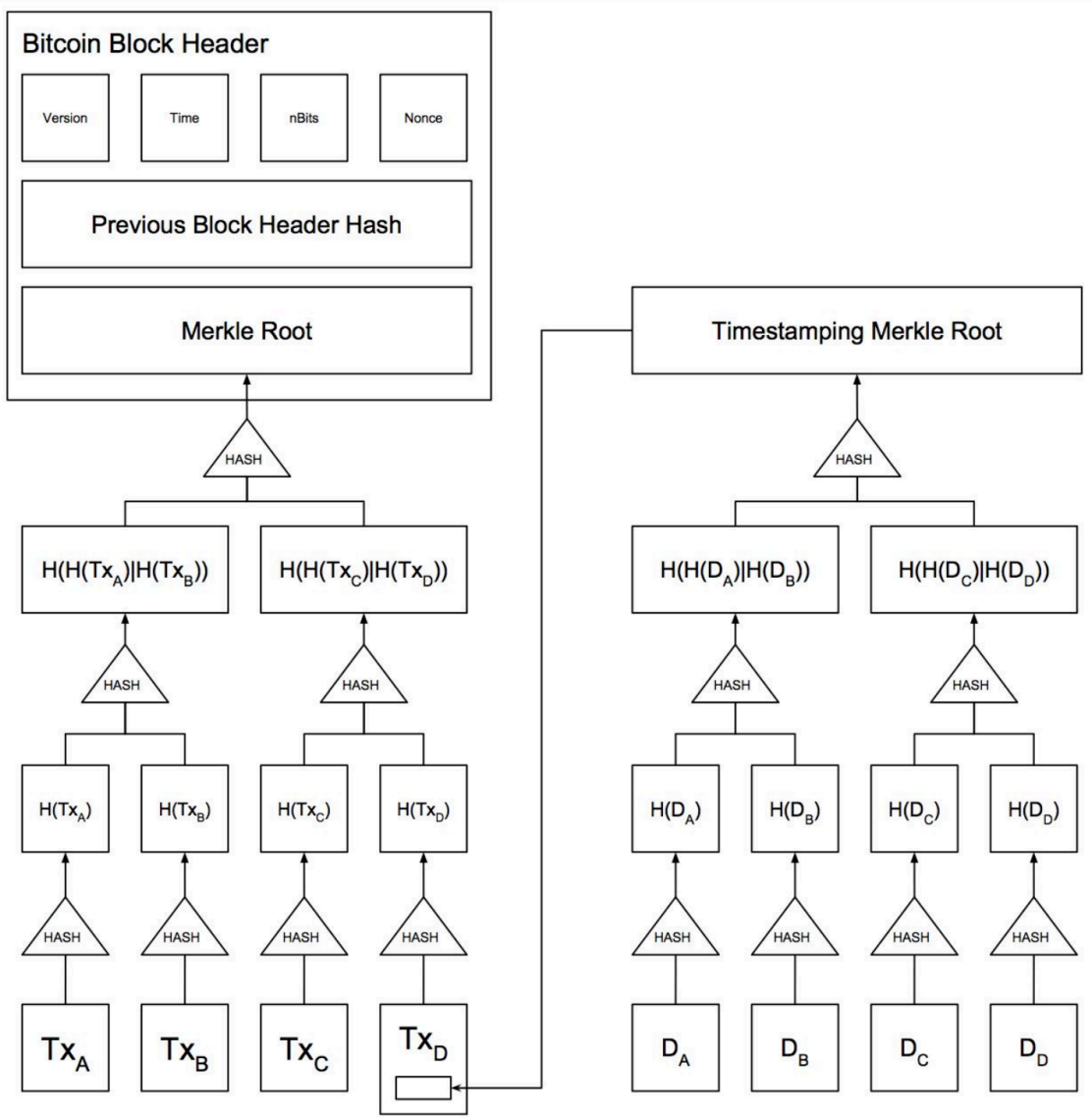

OpenTimestamps Proof
DATA
SHA256
PREPEND
APPEND
Merkle Root
Bitcoin Block
+
Bitcoin
Attestation:
Block
height N

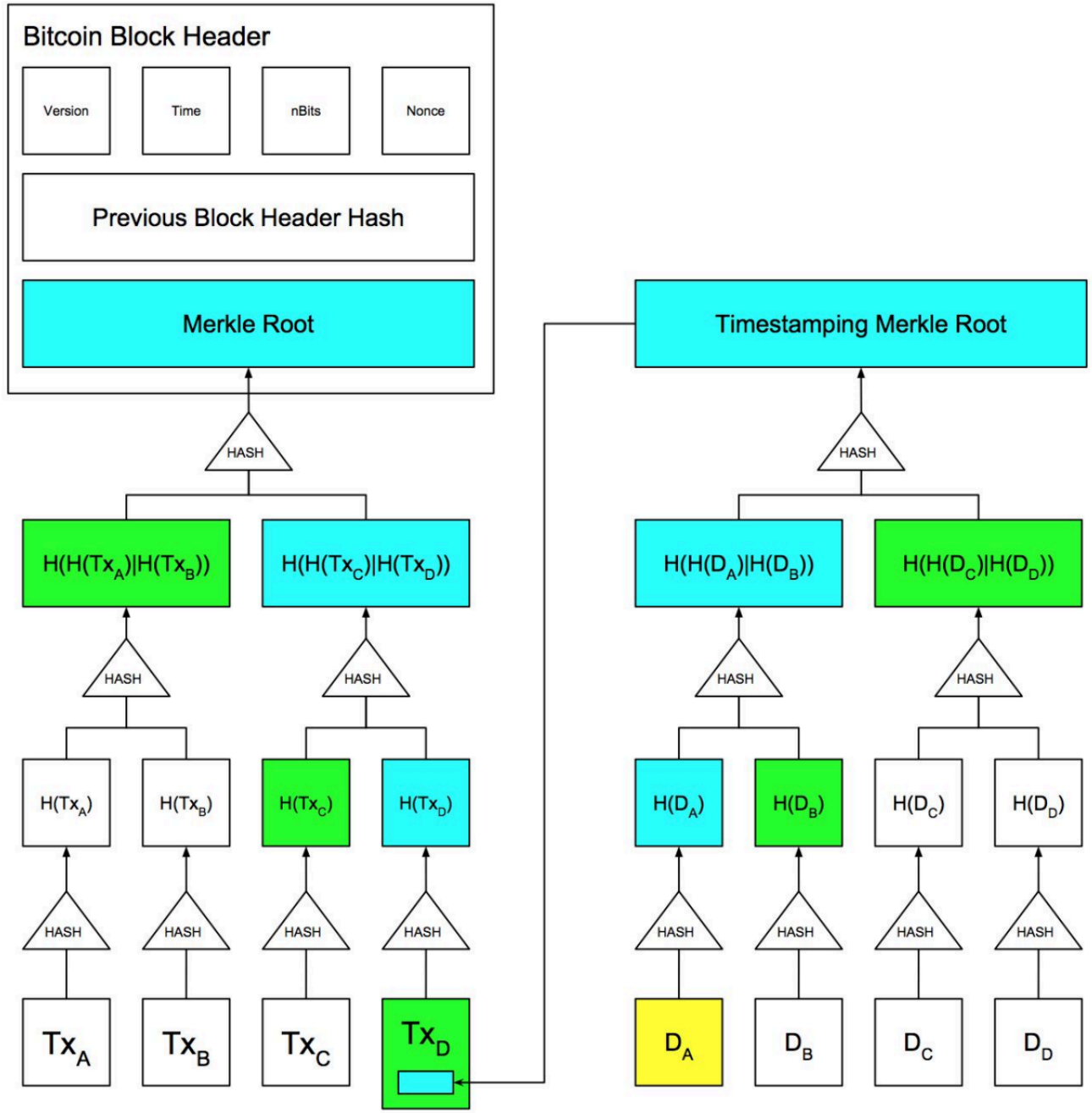
- Aggregation servers
  - Publically available *meeting points* where anyone can submit a digest to be timestamped
  - Aggregates all submitted timestamps in a one second interval into a merkle tree (1 sec latency)
  - Submits the tip of the tree (called a commitment) to a public calendar server

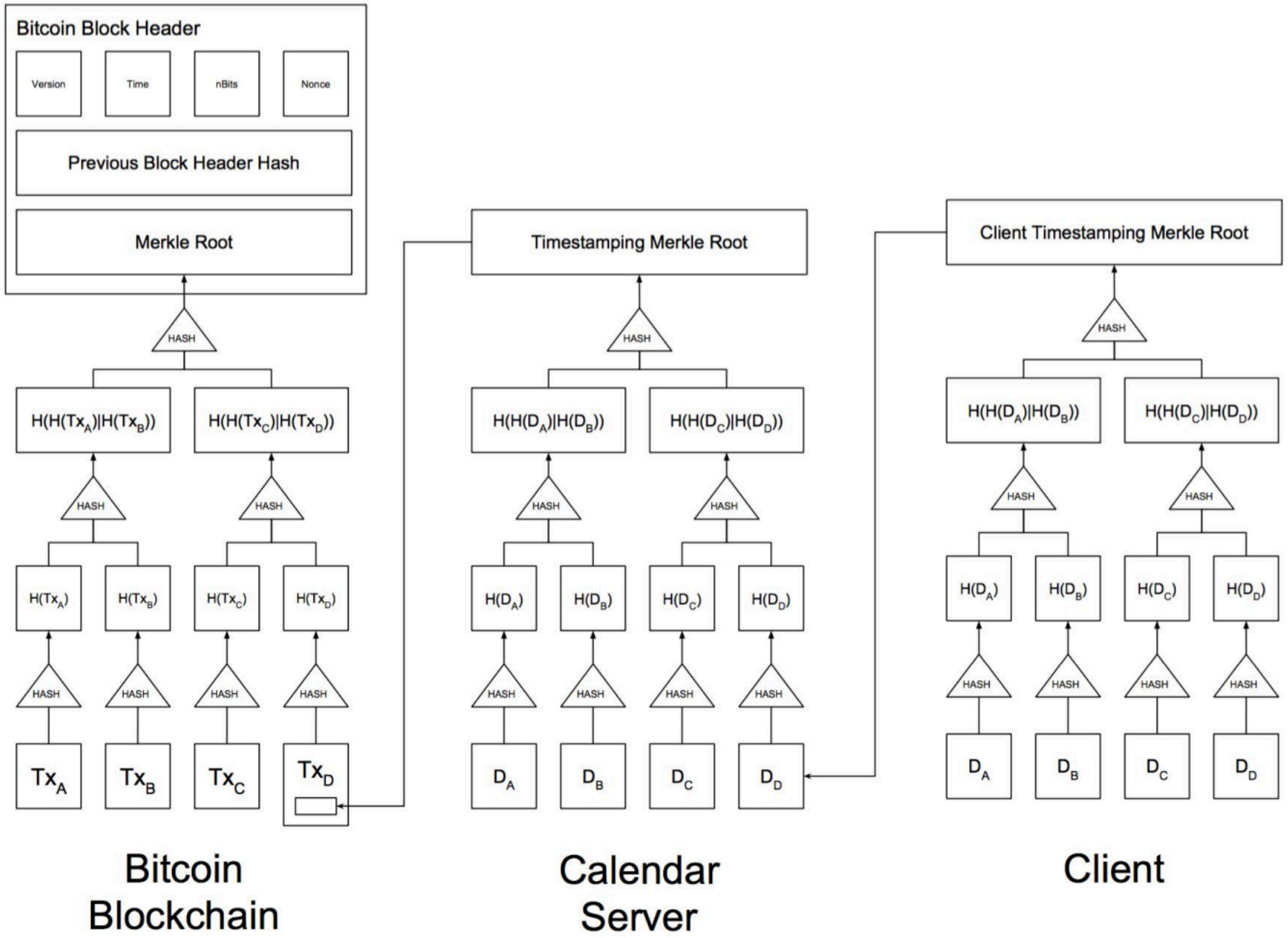
- Public calendar servers
  - Provides remote access to a calendar (a collection of timestamps)
  - Merkle root (per-interval commitment) saved indefinitely in a *calendar*
  - Multiple redundant calendars to achieve fault tolerance (3 or 4, not much)
  - Central point of failure
  - Example: https://finney.calendar.eternitywall.com

- Donation based

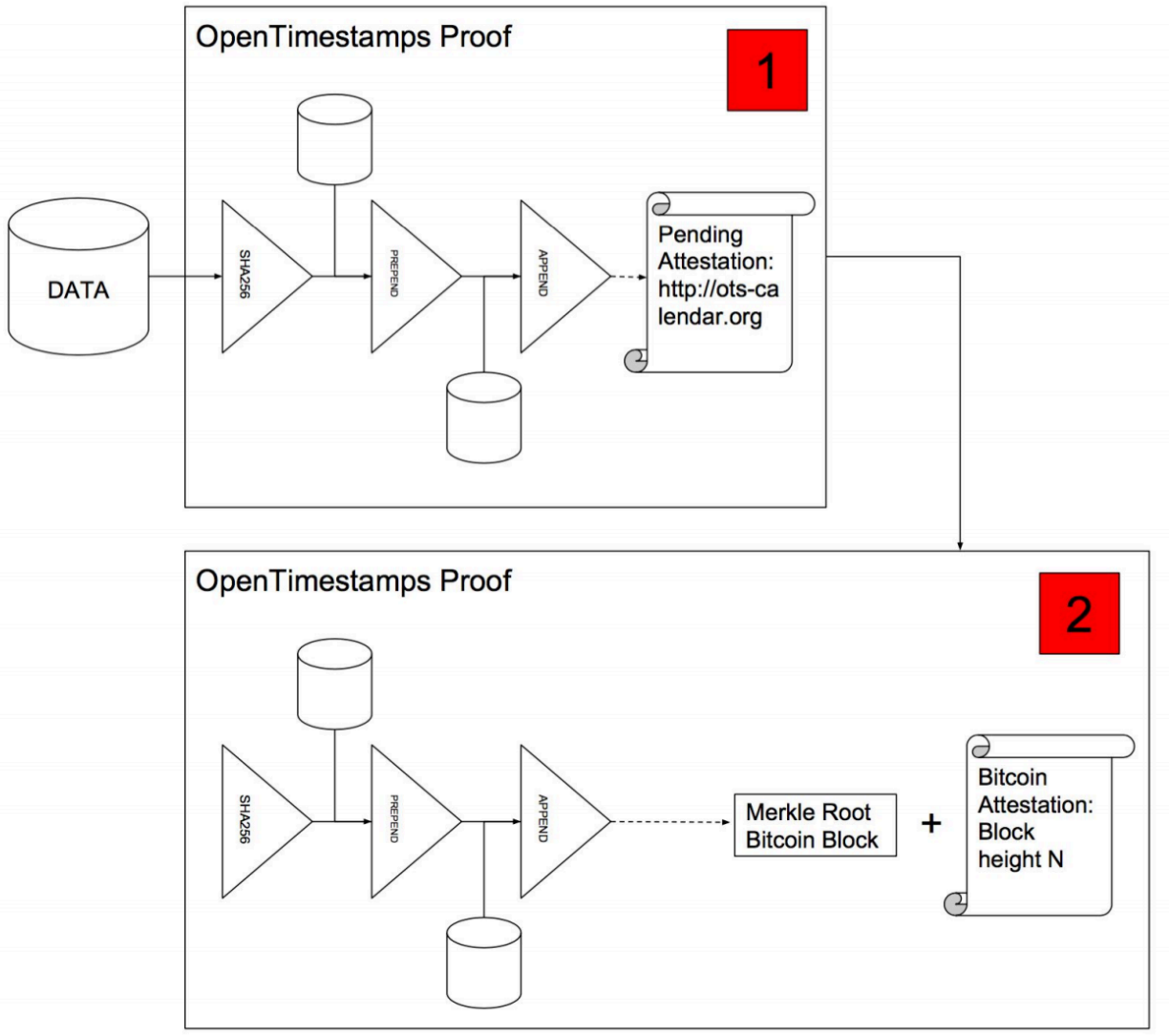
AGGREGATING TIMESTAMPS




Bitcoin Block Header
Version
Time
nBits
Nonce
Previous Block Header Hash
Merkle Root
HASH
H(H(Tx_A)|H(Tx_B))
H(H(Tx_C)|H(Tx_D))
HASH
HASH
H(Tx_A)
H(Tx_B)
H(Tx_C)
H(Tx_D)
HASH
HASH
HASH
HASH
Tx_A
Tx_B
Tx_C
Tx_D
Timestamping Merkle Root
HASH
H(H(D_A)|H(D_B))
H(H(D_C)|H(D_D))
HASH
HASH
H(D_A)
H(D_B)
H(D_C)
H(D_D)
HASH
HASH
HASH
HASH
D_A
D_B
D_C
D_D


Bitcoin Block Header
Version
Time
nBits
Nonce
Previous Block Header Hash
Merkle Root
HASH
H(H(Tx_A)|H(Tx_B))
H(H(Tx_C)|H(Tx_D))
H(Tx_A)
H(Tx_B)
H(Tx_C)
H(Tx_D)
Tx_A
Tx_B
Tx_C
Tx_D
Timestamping Merkle Root
H(H(D_A)|H(D_B))
H(H(D_C)|H(D_D))
H(D_A)
H(D_B)
H(D_C)
H(D_D)
D_A
D_B
D_C
D_D
Client Timestamping Merkle Root
Bitcoin Blockchain
Calendar Server
Client

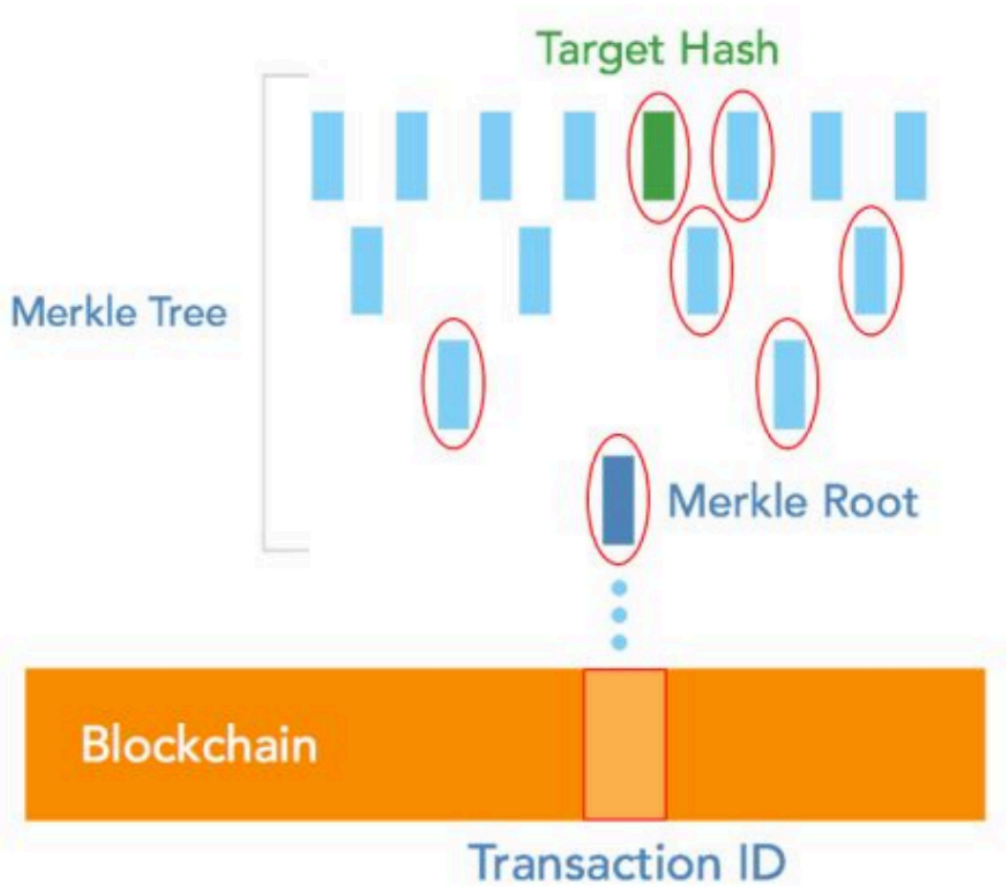
INCOMPLETE PROOF



CHAINPOINT

- Open standard developped by Tierion released in June 2015

- Uses JSON-LD, allowing for optional digital signatures
- Supports multiple anchor points into multiple blockchains
- SHA-2 and SHA-3 support, extensible to other hashing algorithms
- Chainpoint proof defines a set of operations that cryptographically link the data to blockchains



JSON-LD example of a Chainpoint 2.0 receipt:


```
{
  "@context": "https://w3id.org/chainpoint/v2",
  "type": "ChainpointSHA256v2",
  "targetHash": "bdf8c9bdf076d6aff0292a1c9448691d2ae283f2ce41b045355e2c8cb8e85ef2",
  "merkleRoot": "51296468ea48ddbcc546abb85b935c73058fd8acdb0b953da6aa1ae966581a7a",
  "proof": [
    {
      "left": "bdf8c9bdf076d6aff0292a1c9448691d2ae283f2ce41b045355e2c8cb8e85ef2"
    },
    {
      "left": "cb0dbbedb5ec5363e39be9fc43f56f321e1572cfcf304d26fc67cb6ea2e49faf"
    },
    {
      "right": "cb0dbbedb5ec5363e39be9fc43f56f321e1572cfcf304d26fc67cb6ea2e49faf"
    }
  ],
  "anchors": [
    {
      "type": "BTCOpReturn",
      "sourceId": "f3be82fe1b5d8f18e009cb9a491781289d2e01678311fe2b2e4e84381aafadee"
    }
  ]
}
```

TIERION NETWORK

- In 2017 Tierion raised $25 million in ICO
  - ERC20 Tierion Network Token (TNT)

- ICO engineering crap
  - A useless complex system to give TNT token a *raison d'être*
  - TNT supposed to be used to pay node operators

- Chainpoint Network
  - Chainpoint Nodes aggregate hashes into a Merkle tree every 5 seconds
  - The Merkle root is published in Bitcoin, and possibly multiple others
  - Nodes register by staking 5000 TNT in a registry Ethereum smart contract
  - Nodes provide additional scaling, mirror the global calendar, and audit Core
  - Nodes communicate with Core nodes, and gain eligibility to earn TNT
- Chainpoint Core Network
  - Network of partners that run the full Chainpoint Service stack
  - Maintain the Global Calendar blockchain
  - Writes to the calendar enforced by a leader election using a cluster of Consul servers
  - Should use Tendermint consensus protocol in 2.0
  - Blocks stored as records in a distributed cluster of CockroachDB databases

NOTARIZATION SERVICES

- https://www.woleet.io/
- https://stampery.com/
- https://blocksign.com/
- https://proofofexistence.com/
- https://originstamp.org/
- https://www.blocknotary.com/
- https://signatura.co/
- https://tweetstamp.org/
- @OtsProofBot2
- ...

CERTIFYING ART

- https://verisart.com/
- https://binded.com/
- https://artchain.info/ / https://www.bitcoingallery.com/
- Ascribe (closed)
- Monegraph (closed)

3

DIGITAL ASSETS

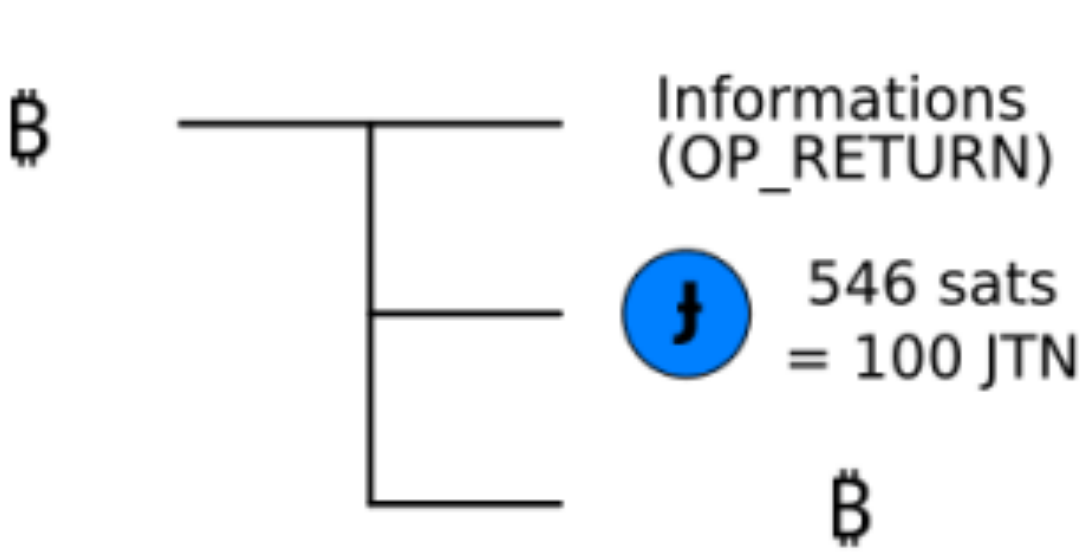
COLORED COINS

- Giving new meaning to UTXOs to represent digital assets

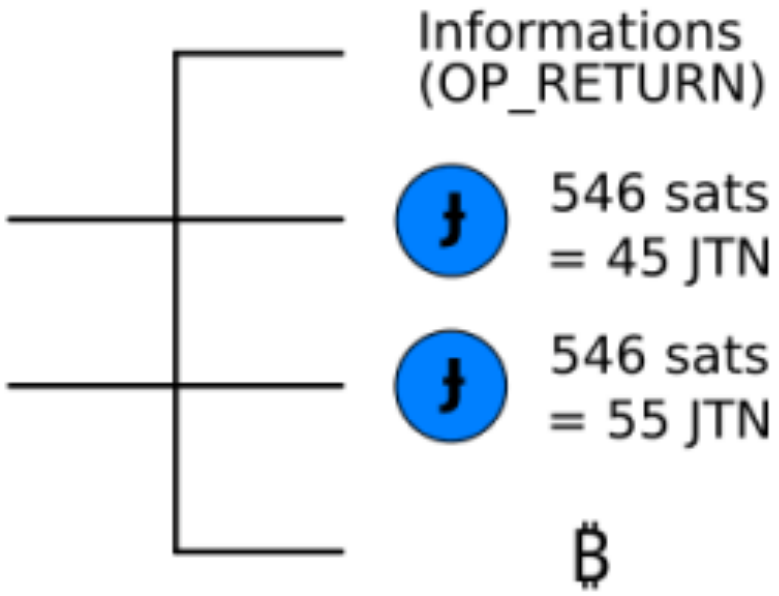
- Using the blockchain to store metadata and proof of transmission
  - EPOBC (ChromaWay)
    - Based on nSequence field of transaction's first input
  - Colu (coloredcoins.org) and OpenAssets (CoinPrism)
    - Based on OP_RETURN, optionally multisig
    - Data specifies how tokens flow from inputs to outputs

- Many issues
  - Not enforced by miners / consensus rules
  - Risk of burning the tokens by mistake
  - User has to validate token authenticity
  - No privacy (small anonimity set, mixing may break cc protocols)
  - No interoperability between protocols / wallets
  - Same issues as on-chain Bitcoin: fees, slow confirmation

Creation transaction



Transfer transaction



EMBEDDED CONSENSUS SYSTEMS

- Based on the same idea as colored coin (OP_RETURN, optionally multisig)
  - Data is then read, validated, and executed by EC nodes
  - Compared to CC, adds a set of tools to support the handling of assets
    - Decentralized exchange
    - Token issuance, ICO
    - Non-fungible tokens (gaming items, collectibles)

- Counterparty
- Omni (previously Mastercoin)

- Many issues
  - Not enforced by miners / consensus rules
  - Fighting with Bitcoin over scarce ressources
  - No light client possible
  - Reduced privacy (small anonymity set)
  - Difficult maintenance
  - Same issues as on-chain Bitcoin: fees, slow confirmation

4

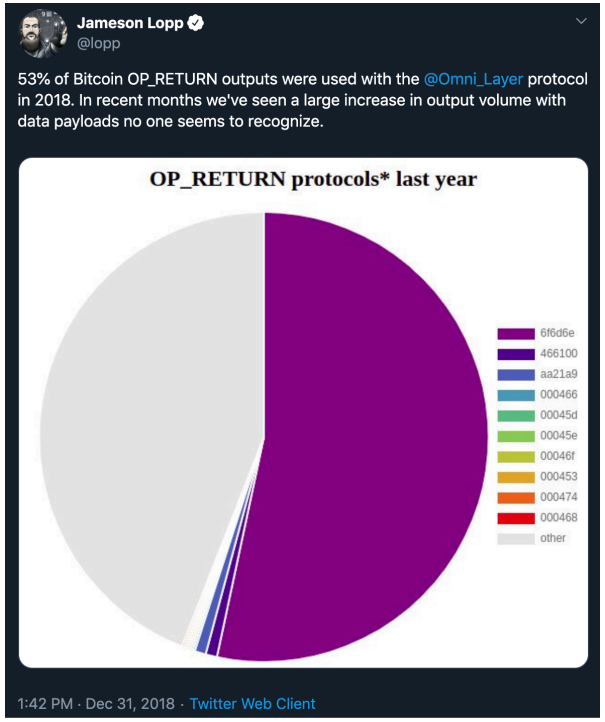
OTHERS

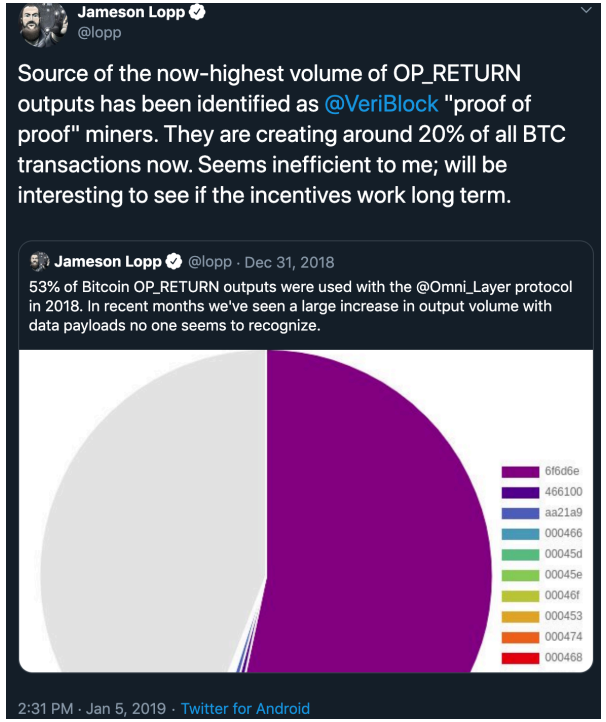
BLOCKSTACK

- Blockstack is a full-stack decentralized computing network
  - Allows users to own their own data
  - Blockstack ID eliminates the need for password-based logins
- Blockstack Naming Service is a network system that binds names to off-chain state
  - Uses Bitcoin (op_return) to store names
  - At the base of the Blockstack architecture

VERIBLOCK

- Proof-of-Proof is a consensus protocol invented by VeriBlock which allows any blockchain to inherit the full security of Bitcoin
- Anchor a snapshot of the altcoin's ledger into the Bitcoin chain (via op_return)
- Reversing consensus on a VeriBlock-secured blockchain block requires forking Bitcoin, VeriBlock, and the VeriBlock-secured blockchain simultaneously





OPENSEALS FRAMEWORK (RGB)

- Allows creation of private state-managing systems and networks
  - Any data
  - Any Ethereum-like smart-contracts

- Based on Peter Todd work (single-use seal, CSV, Dex language, Proofmarshal, ...)
- Used by RGB protocol to issue and account issued assets
- State is maintained in a DAG on top of Bitcoin blockchain
- Consensus on the state is achieved using
  - Client-side validation for off-chain data
  - Verification of cryptographic commitments embedded into transaction outputs (single-use seals)

- Two types of on-chain cryptographic commitment
  - Pay-To-Contract (default)
  - OP_RETURN (used when P2C not possible)

CONCLUSION

- Anchoring data on the Bitcoin blockchain allows innovative protocols
  - Use cases beyond money
  - Piggyback on Bitcoin immutability
- Lot of previous attempts have failed due to usual on-chain limitations
- Goals are computational and storage efficiency, privacy
  - Client-side validation
  - Cryptographic commitment protocols (P2C, S2C)
    - Will be more adopted with the introduction of Schnorr, Taproot, Graftroot, MAST, ...
- Data anchoring still in its infancy

LEARN MORE

- https://bitcoin.org/en/transactions-guide#null-data
- https://medium.com/woleet/beyond-data-anchoring-bee867d9be3a
- https://petertodd.org/2016/opentimestamps-announcement
- https://petertodd.org/2017/misleading-and-inaccurate-tierion-ico-claims
- https://blog.eternitywall.com/2018/04/13/sign-to-contract/
- https://github.com/LeoComandini/electrum-timestamp-plugin
- https://github.com/rgb-org/spec
- https://github.com/LNP-BP/lnpbps/blob/master/lnpbp-0001.md
- https://github.com/LNP-BP/lnpbps/blob/master/lnpbp-0002.md
- https://github.com/LNP-BP/lnpbps/blob/master/lnpbp-0003.md
- https://fc17.ifca.ai/bitcoin/papers/bitcoin17-final32.pdf
- https://github.com/rgb-org/spec/blob/develop/01-OpenSeals.md