A Blockchain-based Mapping System

IETF 98 – Chicago
March 2017

Jordi Paillissé, Albert Cabellos, Vina Ermagan, Fabio Maino
acabello@ac.upc.edu

http://openoverlayrouter.org
A short Blockchain tutorial
Blockchain - Introduction

• Blockchain = decentralized, secure and trustless database
• Add blocks of data one after another
• Protected by two mechanisms:
  – Chain of signatures
  – Consensus algorithm
• First appeared: Bitcoin, to exchange money
• Many more applications are possible
Blockchain - Transactions

Transaction
- Sender’s Public Key
- Sender’s signature
- Tx Data
Blockchain - Transactions

1. Transactions are broadcasted to all the nodes

Transaction
Sender’s Public Key
Sender’s signature
Tx Data

P2P network
Blockchain - Transactions

1. Transactions are broadcasted to all the nodes

2. A node collects transactions into a block

Transaction
- Sender’s Public Key
- Sender’s signature
- Tx Data

Block
- Prev. Hash
- Nonce
- Transactions 1 ··· N
Blockchain - Transactions

1. Transactions are broadcasted to all the nodes

2. A node collects transactions into a block

3. Compute consensus algorithm

Transaction
- Sender's Public Key
- Sender's signature
- Tx Data

Block
- Prev. Hash
- Nonce
- Transactions 1 ··· N

New Block
- Prev. Hash
- Nonce
- Transactions 1 ··· N’
Blockchain - Transactions

1. Transactions are broadcasted to all the nodes

2. A node collects transactions into a block

3. Compute consensus algorithm

4. Broadcast new block to the network
Blockchain - Transactions

1. Transactions are broadcasted to all the nodes

2. A node collects transactions into a block

3. Compute consensus algorithm

4. Broadcast new block to the network

5. The other nodes verify the consensus algorithm and accept the block
Blockchain - Properties

• Decentralized: all nodes have the entire blockchain
• No prior trust required
• Decouples ownership from identity
• Append-only and immutable: added transactions cannot be modified
• Verifiable

“Blockchain Technology”, Sutardja Center (UC Berkeley)
A Blockchain-based Mapping System
Overview
Basic Idea

- **Objective**: Securely store:
  - EID prefix delegations (as in RPKI or DDT-ROOT)
  - EID-to-MS information (as in DDT)
  - EID-to-RLOC mappings (as in MS)

- Map Resolvers read the blockchain to find the mappings

- **Idea**: An EID is equivalent to a coin
  - Wallet: A set of EIDs
  - Transaction: Delegating EIDs or binding them to a MS or a set of RLOCs
  - Blockchain: A public ledger of the transactions
A Blockchain-based Mapping System
Storing EID delegations and EID-to-RLOC mappings
1-Writes
Genesis block,
claims all EID space
1-Writes
Genesis block, claims all EID space

2-Writes
Prefix \( \mapsto \) owner mapping

Delegation

blockchain

\[ \begin{array}{cccccc}
0 & 1 & 2 & \ldots & n & n+1 & n+2 \\
\end{array} \]
1-Writes
Genesis block, claims all EID space

2-Writes
Prefix = owner mapping

3-Writes
Prefix = EID-to-RLOC mapping

blockchain

WRITE

ROOT

EID-prefix owner

EID-prefix owner

Delegation

Delegation
1-Writes

Genesis block, claims all EID space

2-Writes

Prefix → owner mapping

3-Writes

Prefix → EID-to-RLOC mapping

blockchain

0 1 2 ... n n+1 n+2

READ

xTR

1-Map-Request

17

MR/MS with blockchain

EIDpref → RLOC1
RLOC2
RLOC3

WRITE

ROOT

Delegation

Delegation

2-Fetch mappings

1-Writes

EID-prefix owner

3-Writes

EID-prefix owner
A Blockchain-based Mapping System

Storing EID delegations and EID-to-MS information
1-Writes
Genesis block,
claims all EID space

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>...</th>
<th>n</th>
<th>n+1</th>
<th>n+2</th>
</tr>
</thead>
</table>

blockchain

WRITE

ROOT
1-Writes
Genesis block, claims all EID space

2-Writes
Prefix = owner mapping

EID-prefix owner

Delegation

blockchain

0 1 2 ... n n+1 n+2
1-Writes
Genesis block, claims all EID space

2-Writes
Prefix → owner mapping

3-Writes
Prefix → EID-to-MS information

blockchain
WRITE

1-Writes
Genesis block, claims all EID space

2-Writes
Prefix = owner mapping

3-Writes
Prefix = EID-to-MS information

1-Map-Request
2-Map-Request
3-Map-Reply

READ

xTR

MR

MS

(blockchain)

ROOT

EID-prefix owner

EID-prefix owner

2-Fetch EID-to-MS information

(in proxy-mode, as an example)
Pros and Cons

**Pros**
- Infrastructure-less and decentralized
- Fast lookup
- Secure, without certs
  - Non-repudiation
  - Resilience
  - Integrity
  - Authentication
- No prior trust required
- Simple rekeying

**Cons**
- Challenges with incentives
- Slow updates
  - Mappings can be stored in a MS, then performance is as fast as DDT
- Costly bootstrapping
- Large storage required

Can be mitigated using a dedicated chain
Comparison with LISP-DDT

**LISP-DDT**
- Root
  - DDT1
    - MS1.1
    - MS1.2
  - DDT2
    - MS2.1

- Fast update ➔ Dynamic mappings
- Manual configuration

**Blockchain**
- Node 1
  - Chain
- Node 2
  - Chain
- Node N
  - Chain

- Less infrastructure
- No certificates
- Fast queries
- Large storage required
- Update mappings slow ➔ Store Mappings in MS (same performance as MS)
## Issues with RPKI

<table>
<thead>
<tr>
<th></th>
<th>RPKI</th>
<th>Blockchain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anonymity [1]</td>
<td>Prefixes linked to owner name</td>
<td>Prefixes linked to a public key</td>
</tr>
<tr>
<td>Revocation</td>
<td>Performed by CAs</td>
<td>Performed automatically (validity time) or impossible</td>
</tr>
</tbody>
</table>

Scalability

One mapping for each block of /24 IPv4 address space
Growth similar to BGP churn*
Prefix delegation + mappings
Each transaction approx. 400 bytes
Only prefixes: approx. 40 GB in 20 years (worst case + BGP table growth*)

*Source: http://www.potaroo.net/ispcol/2017-01/bgp2016.html

Approx. 600 GB in 2034
A Blockchain-based Mapping System
Transactions
First transaction

• Map-Resolver trust the Public Key of the Root, that initially claims all EID space by writing the genesis block
• Root can delegate all EID space to itself and use a different keypair

\[
\text{Hash(P + root) = Root@1} \quad \text{“I own all the address space”} \quad \text{Root@2}
\]
Prefix delegation

- Root delegates EID-prefixes to other entities (identified by Hash(Public Key)) by adding transactions

  - Owners can further delegate address blocks to other entities or write MS addresses (and MS’s Public Key)
Writing mappings

• Just like delegating a prefix, but instead of the Map Server address, we write the mapping

Deleg3@ \rightarrow \text{New Transaction} \rightarrow \text{“mapping”} \rightarrow 0.0.1/24 \text{ is at RLOC1}
Rekeying

• Delegating the owned EID-prefixes to itself using a new key set.
• Simpler than traditional rekeying schemes
• Can be performed independently, i.e. each owner can do it without affecting other owners
• Same procedure for mappings
Map-Reply Authentication

- MS public key can also be included in the delegations
- Since blockchain provides authentication and integrity for this key, MRs can use it to verify Map-Repplies
A Blockchain-based Mapping System Prototyping
Design considerations

• Bitcoin is too restrictive:
  – Only for money transfer
  – Huge blockchain file size (approx. 100 GB)
  – High bootstrap time (several days*)
  – Low throughput (7 transactions/sec.)

• New blockchain technologies:
  – More scalable
  – Smart contracts

*depends on connection speed
Dedicated chain

• Public (anyone can use it) but dedicated (only for mappings)

• Stores:
  – Prefix delegations – Replaces DDT ROOT
  – EID-to-MS information – Replaces DDT-Nodes
  – EID-to-RLOC mappings (if you don’t expect many updates) – xTR does NOT need a Map-Server

• We plan to deploy it in LISP-Beta
Prototype

LISP Flow Mappings

Map-Request
Map-Reply

xTR

New mappings

Java SDK

Validate

Hyperledger P2P network

New mappings
A Blockchain-based Mapping System

IETF 98 – Chicago
March 2017

Jordi Paillissé, Albert Cabellos, Vina Ermagan, Fabio Maino
acabello@ac.upc.edu

http://openoverlayrouter.org
More about the Consensus Algorithm

• Rules used by nodes to agree on which data to accept
• Eg. Bitcoin uses Proof of Work
• Miners compute Proof of Work
  – Finding a nonce that when added to the data makes its hash start with N zeros.
  – Hard
• Other algorithms are being explored:
  – **Proof of Stake:** nodes with more assets are more likely to add blocks
  – **Practical Byzantine Fault Tolerant:** reach a minimum number of endorsements from nodes in order to add data
  – **Deposit-based:** assets are lost if a node performs an illegal operation (security deposit)

https://www.linkedin.com/pulse/consensus-mechanisms-used-blockchain-ronald-chan