Privacy Enhanced Distributed Ledger Technology and Hyperledger Implementation

Rick Kuhn
US National Institute of Standards and Technology
kuhn@nist.gov
Why listen to this talk?

• Blockchain has valuable properties, but conflicts with privacy and exception management – “immutable” - deletion impossible

⇒ Sometimes we don’t need blockchain, only some features

• Data block matrix → distributed trust, integrity protection of blockchain, but allows controlled edits for privacy or block re-use
Key points

• Blockchain -> integrity protection, write-only blocks
• Data block matrix -> integrity protection, read-write blocks

• Drop-in compatibility for Hyperledger Fabric applications
• Released and available
• Also high-volume, low-capacity such as IoT -> re-use blocks
• Scalability potential where ledger size is a factor
Market, range of applications for DLT

Blockchain fits pretty well here

Not as well here

Much research to try to get around blockchain properties

Privacy

Corrections

Re-use blocks

This talk here
Why use redactable DLT for privacy?

• Permanence/immutability conflicts with ‘right to erasure’ privacy regulations

• Privacy rules such as European Union General Data Protection Regulation (GDPR) require that all information related to a particular person can be deleted at that person's request
  • *any personal* data "concerning an [identified](#) or [identifiable](#) natural person"
  • [pseudo-anonymized](#) data linkable to person
  • US states adopting similar privacy rules, including California and Virginia

New focus on logistics, shipping, Internet of Things (IoT)
– capability for exception management means more practical DLT
- also cases where storage is limited, such as IoT, where block re-use helpful
What’s been tried to solve blockchain/privacy conflict?

• Don’t put personal data on blockchain – but pseudo-anonymized data are still considered personal; Financial transactions are obviously personal data

• Encrypt data and destroy key to delete – but data must be secure for decades (e.g., DES replaced in only 17 years)

• Chameleon hash function – non-standard cryptography

• Off-chain storage of sensitive data – what if on-chain index to off-chain data is also sensitive?
Many blockchain applications don’t need blockchain, just some blockchain features

**Datablock matrix** – two hash values per block instead of linked chain
Blockchain -> distributed trust, integrity protection, immutability
Datablock matrix -> distributed trust, integrity protection, editable

- Open source
- Incorporated into Next Gen Access Control
- NOT to replace blockchain, to provide alternative tools for distributed system design
- **Hyperledger Fabric component** available
Compatible with Hyperledger applications

Application
(e-commerce, supply chain, etc.)

No change

Application
(e-commerce, supply chain, etc.)

Hyperledger Fabric

Hyperledger Fabric

blkstorage module (current)

blkstorage module (blockmatrix added)
Datablock matrix data structure

- A data structure that provides integrity assurance using hash-linked records while also allowing the deletion of records
- Stores hashes of each row and column
- => each block within the matrix is protected by two hashes
- Suggested use for private/permissioned distributed ledger systems

Figure 1. Block matrix
How does this work?

- Suppose we want to delete block 12
- disrupts the hash values of $H_{3,-}$ for row 3 and $H_{-,2}$ and column 2
- blocks of row 3 are included in the hashes for columns 0, 1, 3, and 4
- blocks of column 2 are included in the hashes for rows 0, 1, 2, and 4
Structure can be extended to multiple dimensions

- Block dispersal for 3 dimensions
- Location in sectors 0..5 according to $b \mod 6$ for block $b$
Why use this data structure?

Again, many blockchain applications *don’t need blockchain, just some features*

Enlarge the market for blockchain

• Solve the conflict between blockchain and privacy regulations
• Allow for corrections or block re-use

Replace network communication with local data

• You can obviously do this with conventional database functions, but
• New data structure adds integrity checks as in blockchain
• Re-writing blocks can be more practical for high-volume, or where storage is limited

Lightweight, easy-to-use component for distributed system design
Do you need a shared, consistent data store?

- Yes
  - Distributed ledgers provide a historically consistent data store. If you don’t need that, you don’t need a distributed ledger.
    - **CONSIDER:** Email / Spreadsheets

- No
  - Your data comes from a single entity. Distributed ledgers are typically used when data comes from multiple entities.
    - **CONSIDER:** Database
    - **CAVEAT:** Auditing Use Cases

Does more than one entity need to contribute data?

- Yes
  - Data records, once written, are never updated or deleted?
    - No
      - Sensitive identifiers WILL NOT be written to the data store?
        - Yes
          - Are the entities with write access having a hard time deciding who should be in control of the data store?
            - Yes
              - Do you want a tamperproof log of all writes to the data store?
                - Yes
                  - You may have a useful blockchain use case
                - No
                  - You may have a useful data block matrix use case
            - No
              - You may have a useful blockchain use case
        - No
          - **NO**

- No
  - **NO**

Data records, once written, are never updated or deleted?

- Yes
  - Sensitive identifiers WILL NOT be written to the data store?
    - Yes
      - Are the entities with write access having a hard time deciding who should be in control of the data store?
        - Yes
          - Do you want a tamperproof log of all writes to the data store?
            - Yes
              - You may have a useful blockchain use case
            - No
              - You may have a useful data block matrix use case
        - No
          - **NO**
    - No
      - **NO**

Sensitive identifiers WILL NOT be written to the data store?

- Yes
  - Are the entities with write access having a hard time deciding who should be in control of the data store?
    - Yes
      - Do you want a tamperproof log of all writes to the data store?
        - Yes
          - You may have a useful blockchain use case
        - No
          - You may have a useful data block matrix use case
    - No
      - **NO**
  - No
    - **NO**

Are the entities with write access having a hard time deciding who should be in control of the data store?

- Yes
  - Do you want a tamperproof log of all writes to the data store?
    - Yes
      - You may have a useful blockchain use case
    - No
      - You may have a useful data block matrix use case

- No
  - **NO**

Do you want a tamperproof log of all writes to the data store?

- Yes
  - **YES**

- No
  - **NO**
Hyperledger blockmatrix implementation

• Designed to use existing API as closely as possible – add blocks in same manner as adding to blockchain

• Blockmatrix is configurable by channel (private subnet)

• Configure to use conventional blockchain or blockmatrix
  • If a deployment uses two channels, one can be a blockchain and the other can be a blockmatrix

• RED Ledger = Redactable Enhanced Distributed Ledger

• https://csrc.nist.gov/projects/redactable-distributed-ledger
Latency

- average latency for GetAsset transactions remains relatively low across all benchmarks
- increase in average latency for DeleteAsset transactions with transaction load and asset count
Increased transaction load => higher throughput for GetAsset, but same throughput for PutAsset and DeleteAsset

Summarizing: the Hyperledger Fabric implementation is practical for real-world use
More Information

Foundation:

Applications:

Project sites with links to source code and publications
• https://csrc.nist.gov/Projects/enhanced-distributed-ledger-technology
• https://csrc.nist.gov/projects/redactable-distributed-ledger

Acknowledgements
• Josh Roberts, Sylvain Chantreau, Cedric Bell, NIST
• Temur Saidkhodjaev, University of Maryland College Park
• Arsen Klyuev, Johns Hopkins University
• Gokhan Kocak, Asena, Inc.