Blockchain Message Broker: Secure Data Transfer with a Two-Layer Hyperledger Fabric Platform

Gleb Slepenkov, Vladimir Korkhov

Motivation

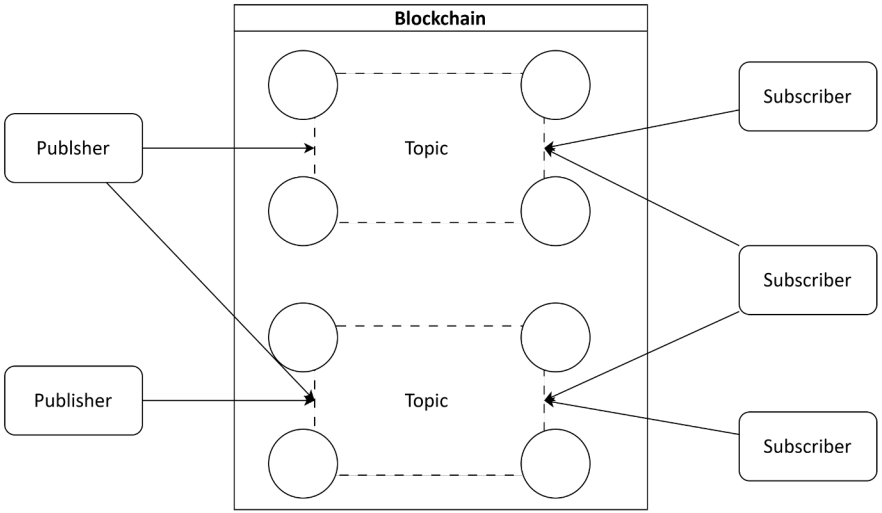
- Cross-organizational Secure Data Transfer (SDT) is crucial in modern information societies

- Especially challenging to be implemented in public networks due to increased risks (data loss, corruption, unauthorized access)

- Blockchain has potential to enhance existing SDT mechanisms:
  - Fault tolerant by design
  - Capable to work in unreliable public networks, separated by multiple clusters
  - Provides smart contracts for SDT process customization

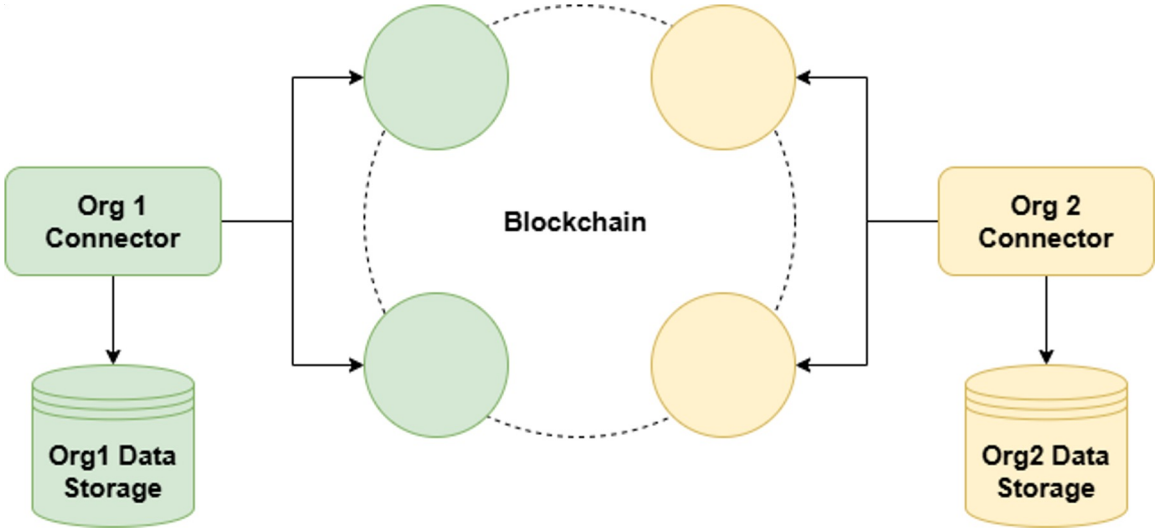
A Blockchain-Based Message Broker

- Blockchain acts as a data transfer tool
- Data transfer process is organized into publish - subscribe way
- Data flow is separated into topics and partitions

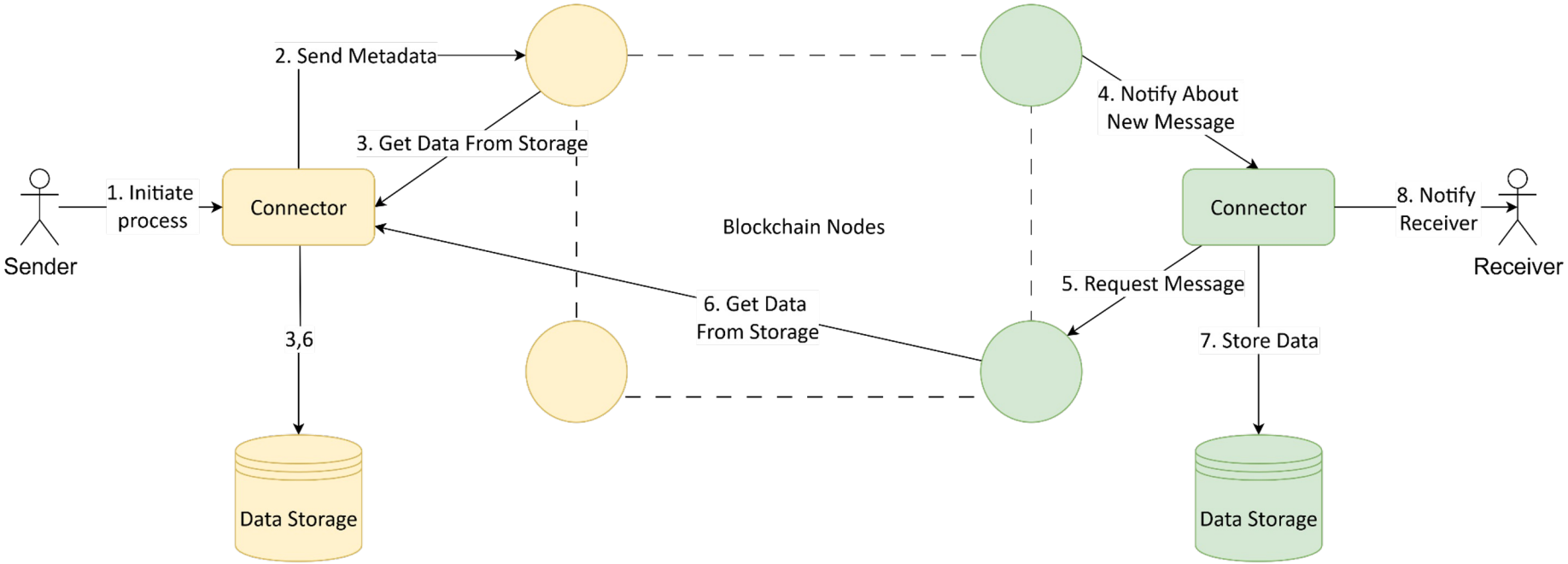


Main System Components

- Blockchain
- Smart contracts
- Off-chain organizational storage
- Connectors



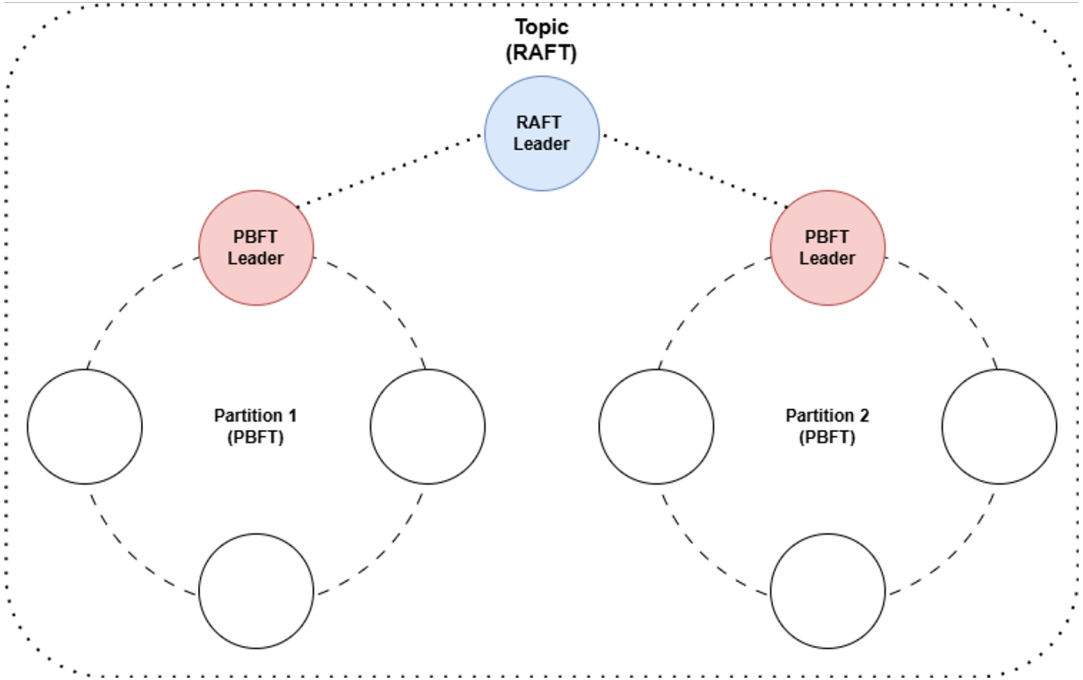
Data Flow


2. Send Metadata
3. Get Data From Storage
4. Notify About New Message
1. Initiate process
Connector
Blockchain Nodes
Connector
8. Notify Receiver
Sender
Receiver
6. Get Data From Storage
5. Request Message
3,6
7. Store Data
Data Storage
Data Storage

Blockchain Network Two-Layer Structure

- Layer 1 (Partition):
  - PBFT consensus
  - Data transfer processing
  - Small clusters (≤7 nodes)

- Layer 2 (Topic):
  - RAFT consensus
  - System management and data replication
  - Combination of Layer 1 clusters



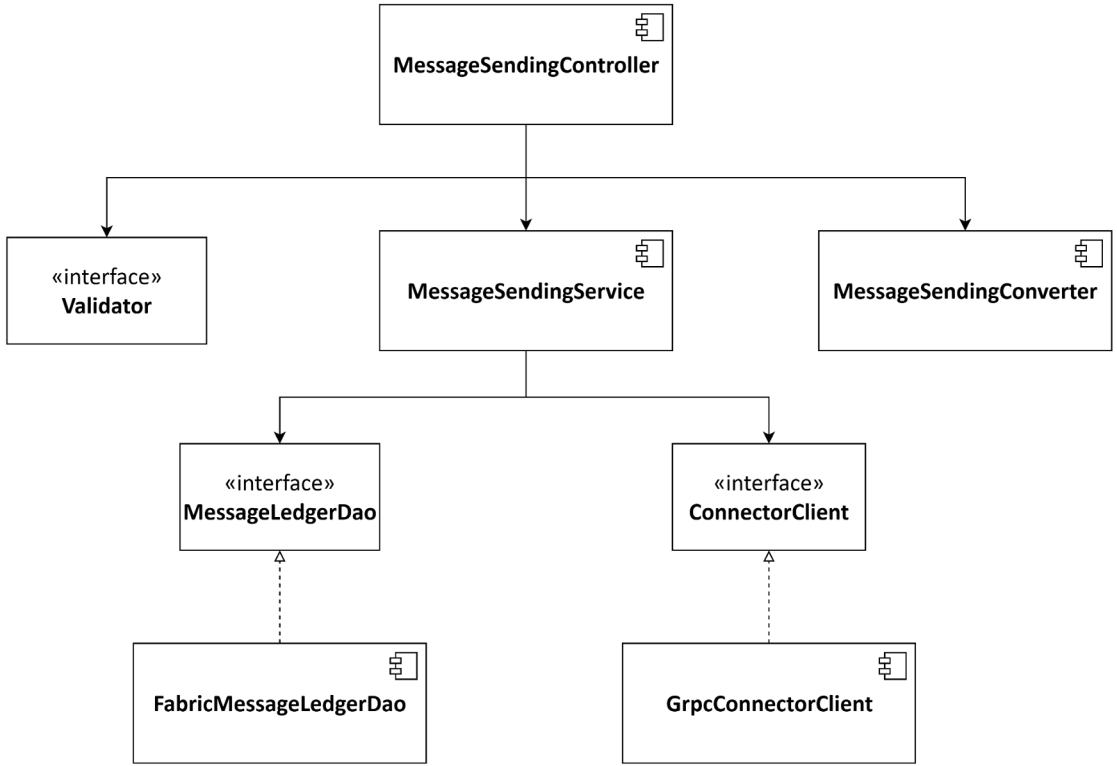
Hyperledger Fabric-Based Implementation Overview

- Both topics and partitions are Fabric Channels with PBFT and RAFT consensus
  - Considering only ordering nodes

- All nodes are part of a topic channel and at least one of its partition channels

- Smart contracts:
  - Secure data transfer (for partitions)
  - Data replication and system management (for topics)

Smart Contract Core Components



Non-blockchain components

- Connector Applications:
  - Java applications, which provide an implementation of Connector API interface
  - Use Smart Contract API for smart contract invocation

- Data Storages:
  - Currently PostgreSQL and MinIO are used

Key Advantages

- PBFT and RAFT combination enhances scalability and performance characteristics
- Smart contract core is by design independent from underlying blockchain technology
- Any connector application, which is implementing the Connector API and is utilizing Contract API, could be used

Future Work

- Prototype development finalization
- Performance testing
- Optimization of consensus algorithm

Thank you for your attention!