HYPERLEDGER

Fabric Next Proposal

Daft 0.01

Binh Nguyen
Agenda

• Review the current Fabric implementation
• Motivation for the next proposal
• Fabric Next in detail
• Development and release roadmap
Current Fabric

- Permissioned blockchain with privacy, confidentiality, and auditability
- Pluggable consensus framework
  - PBFT, SIEVE (proto), NOOPS
- Chaincode execution environment (Go, Java WIP)
  - Docker container (user-cc)
  - In peer process (system-cc)
- Client Node.js SDK
- REST APIs
- Basic CLI
Motivations for Fabric Next

• Better support for confidentiality
• Scalable in number of participants and transaction throughput
• Eliminate non-deterministic transactions
• Enable pluggable data store
• Ability to upgrade fabric and chaincode
• Remove SPF and enable multiple providers of Membership Services
Redefining Key Components

• Client SDK: Assists application security and transacting on blockchain
• Peer: Responsible for endorsing, validating, and committing transactions
  – Maintaining the ledger and aware of other peers via gossip network
• Consenter: Runs consensus to provide atomic broadcast

Assumptions

• Chaincode is deployed on all peers for ease of implementation now
  – Only peers in the chaincode’s endorsement policy may run it

• Application manages the TX content if using hash for confidentiality and data isolation
  – Consenters are agnostic about the transaction content

• No backward compatibility
High Level Interaction

0 Enroll

1 Endorse TX

2 Submit TX

3 Relay Submit TX

4 Deliver batch

Order TXs in a batch according to consensus

application

SDK

Keys

membership

peer

Endorser

Committer

Ledger

Chaincode

Event

c-service

0 Enroll

1 Endorse TX

2 Submit TX

3 Relay Submit TX

4 Deliver batch

Order TXs in a batch according to consensus
Detail Transaction Flow

1. Request endorsement for TX
2. Check endorsements to satisfy policy
3. Submit TX-proposal (may use hash for some TX content)
4. Verify versionDeps, endorsement
   If OK apply stateUpdates
   If Validated commit block
   Simulate TX
   Collect versionDeps, stateUpdates
   Sign endorsement
   Commit block
Transaction Defined

• Transaction is a chaincode function call
  – `<cc-name, function, arguments, tcert> + <endorsement policy>`

• Endorsement policy is attached to transaction
  – Part of chaincode deployment
  – Refined by each transaction
  – We need to define what the default should be
Endorse Transactions

- SDK sends TX to Peers based on the chaincode’s endorsement policy
- Endorser system chaincode (ESCC) processes the endorsement
  - ESCC provides ability to customize the endorsement policy. Default logic will just sign the tx-proposal (tx-proposal := [epID, cclID, txID, stateUpdate, verDeps])
- Note that application may decide to submit hash instead of the TX content
Commit Transactions

- Committing a transaction involves validating each transaction verDep, stateUpdate, and endorsements.
- Committer/Peer calls Validator system chaincode (VSCC) to enable custom validation. The result is true or false (valid or invalid TX).
  - VSCC may perform more sophisticated validation (e.g., executing script OP_CHECKSIG in Bitcoin).
- Committer creates block of valid TX’s from the batch.
- Each block may be validated based on a Block Commit Policy (BCP) (similar to endorsement policy; e.g., 3 out of 4).
- BCP is handled by a system chaincode (CSCC).
  - Gossip among Committers to collect agreements according to the BCP. If satisfied, commit the block; otherwise, discard the block.
Key Development Items

- Consensus
- Multi confidential domains
- Security (ACL, HSM, CC2CC, event)
- Ledger abstraction API and data store
- SDK (submitting transaction)
- Transaction endorsement and validation system chaincode
- Membership services high availability
- Fabric upgrade
- Life-cycle system chaincode
- Naming system chaincode
- Enhance protocol (including status codes and messages)
- Error handling
Proposed Roadmap & Releases

- Custom events
- Version indicator (log and cli)
- CC deploy SDK API
- Consensus 1
- Life-cycle SCC
- Error handling
- CC2CC Privacy
- Tx simulation rw-set
- File-based datastore
- Event security
- Consensus 2
- Short-live certs
- HSM support PKCS11
- SQL-like queries
- Upgrade CC 2
- Upgrade Fabric 2
- Bug fixes
- Member services HA
- Upgrade Fabric 1
- Enhance Ledger API
- Status codes & msg's
- CC naming service
- Event listener SDK
- Upgrade CC 1
- Enhance protocol
- Confidentiality domains
- SDK submitting TX
- Auditability API
- State cache
- Archive/prune TX
- Bug fixes
- Sec code hardening

V0.6 D-Preview  V0.7 D-Preview  V0.1 Alpha  V0.2 Alpha  V0.3 Alpha  V0.1 Beta  V0.2 Beta  V1.0
Backup
Security & Privacy

• HSM
• Chaincode calling chaincode within a confidential domain
• Confidential event
• Chaincode upgrade with key update
• User-based confidentiality
• Peer-based confidentiality
High Availability of Membership Services

- Cluster of membership service instances
- No single company or entity alone control access to the blockchain for all users
- Guarantee the uniqueness of the enrollment ID globally for a chain
Upgrade Fabric

- Replace code
- Replace code with protocol changes
  - Backward compatibility
- Replace code with ledger migration
Upgrade User Chaincode

- Deploy and switch name
- Deploy, switch name and migrate data
Consensus

• Phase 1
  – Separation of consensus into a standalone process (SOLO)
  – Basic endorsing and committing peer with validation

• Phase 2
  – Batch to block with validation on multiple peers
  – Scalability and performance