HYPERLEDGER

Fabric V1.0 Proposal
Overview

• Review the current Hyperledger Fabric implementation
• Motivation for the next proposal
• Hyperledger Fabric Next in detail
• Development and release roadmap
Current Hyperledger 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 none 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
  – Peer is stateless (no memory between transactions)

• Consenter: Runs consensus to provide atomic broadcast

• More detail:
Assumptions

- No backward compatibility with v0.5 Developer-Preview
  - Protocol and blockchain structure changes
- Since Consenters are agnostic about the transaction content, application may use hash(transaction) for confidentiality and manage the transaction content
High Level Interaction

**membership**
- No SPoF
- No SPoT

**application**
- SDK
- Keys

**peer**
- Endorser
- Committer
- Ledger
- Chaincode
- Event

**o-service**
- Order TXs in a batch according to consensus

**Atomic broadcast**
- Order TXs in a batch according to consensus

**0 Enroll**
- 1 Endorse Proposal
- 2 Submit Transaction

**3 Relay Submit TX**
- 4 Deliver batch
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 with multiple providers
- Fabric upgrade
- Life-cycle system chaincode
- Naming system chaincode
- Enhance protocol (including status codes and messages)
- Error handling
Hyperledger-Fabric 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
- Group Endorsement
- Transient parms

- Event security
- Consensus 2
- Short-live certs
- HSM support PKCS11
- SQL-like queries

- Upgrade CC 2
- Upgrade Fabric 2
- Member services 2

- Bug fixes

- Member services 1
- 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
Technical Details
Transaction Defined

• Transaction is a chaincode function call
  – transaction : <proposal> <endorsements>
  – proposal : chaincode, <function-spec>
  – function-spec: function name, arguments
  – endorsements : proposal hash, simulation result

• Each chaincode may be associated with an endorsement and validation system chaincode (ESCC, VSCC)
  – ESCC decides how to endorse a proposal (including simulation and app specifics)
  – VSCC decides transaction validity (including correctness of endorsements)
Bootstrap

• Orderer bootstrap
  – Configuration contains 1 or more anchor nodes ip:port
  – Connect to anchors and start discovery protocol using ecert

• Peer bootstrap
  – Configuration contains 1 or more anchor Peers and an Orderer proxy address
  – Connect to the anchor Peers and start discovery protocol using ecert
  – Connect to the Orderer proxy to send/receive transactions
Endorse Transactions

- SDK sends Proposal to Peers based on the chaincode’s endorsement policy
  - A peer may relay Proposal on client behalf
- Endorser system chaincode (ESCC) processes the endorsement
  - ESCC provides ability to customize endorsement
  - Default logic will just sign the Proposal Response
- Client/SDK decides transaction content if endorsement satisfied
Commit Transactions

• Committing a transaction involves validating each transaction read/write-set and endorsements

• Committer calls Validator system chaincode (VSCC) to validate the batch and commit the block
  – VSCC may perform more sophisticated validation (e.g., executing script OP_CHECKSIG in Bitcoin)

• Emit events (block, invalid TXs, custom)

• Block may be optionally validated (i.e., checkpoint to detect faulty and prune ledger) by Committer system chaincode (CSCC) based on Block Validation Policy
Chaincode Deployment Proposal

- Proposal is a call specification to the Life-Cycle System Chaincode LFSCC to deploy a user chaincode (UCC)
- LFSCC creates version record (version, ucc hash, name) in read-write set
- LFSCC removes container before returning
Chaincode Deployment Transaction

- Deployment transaction records the chaincode address and its initial values onto the ledger
- If the transaction has been successfully committed, launch the container
Upgrade Chaincode

- Upgrade is to deploy a new version of a chaincode with the same name, where version could be the hash of the chaincode source or a mapped name maintained by LCSCC
- Read-write set includes the version of the chaincode used in simulation
- Any transaction referencing the old version is invalid during commit
Query vs Proposal

• Committer provides GRPC interface for structure queries (block, transaction)
  – Transactions might be encrypted (eg, originally submitted form)
• Endorser enables client to “call” chaincode via endorsement proposal from which chaincode may return result
  – Application may call multiple endorsers to get “strong read”
Security & Privacy

• HSM
  – Support PKCS11 to access crypto functions (eg key generation, signature, encryption) on peer, consenter, membership services, and sdk

• Chaincode calling chaincode within a confidential domain

• Confidential event

• Chaincode upgrade with key update

• User-based confidentiality

• Peer-based confidentiality
Membership Services

• Phase 1
  – Cluster of membership service instances

• Phase 2
  – 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
  – Dynamic adding/removing members