CSBC2000

Week 2 | Class 1

Public, private and consortium ledgers

Last Week

- Covered Blockchain Concepts
  - Block header, contents
  - PoW, PoS, DLT
  - P2P Networking
  - Smart Contracts

This Week

- We'll cover cybersecurity concepts
  - Restricted blockchains (public/private/consortium)
  - Hashing, Merkle trees
  - The CIA triad
  - Smart Contract security

This Class

- Redo smart contract from Thursday
- Public, private, consortium ledgers
- Hyperledger Fabric

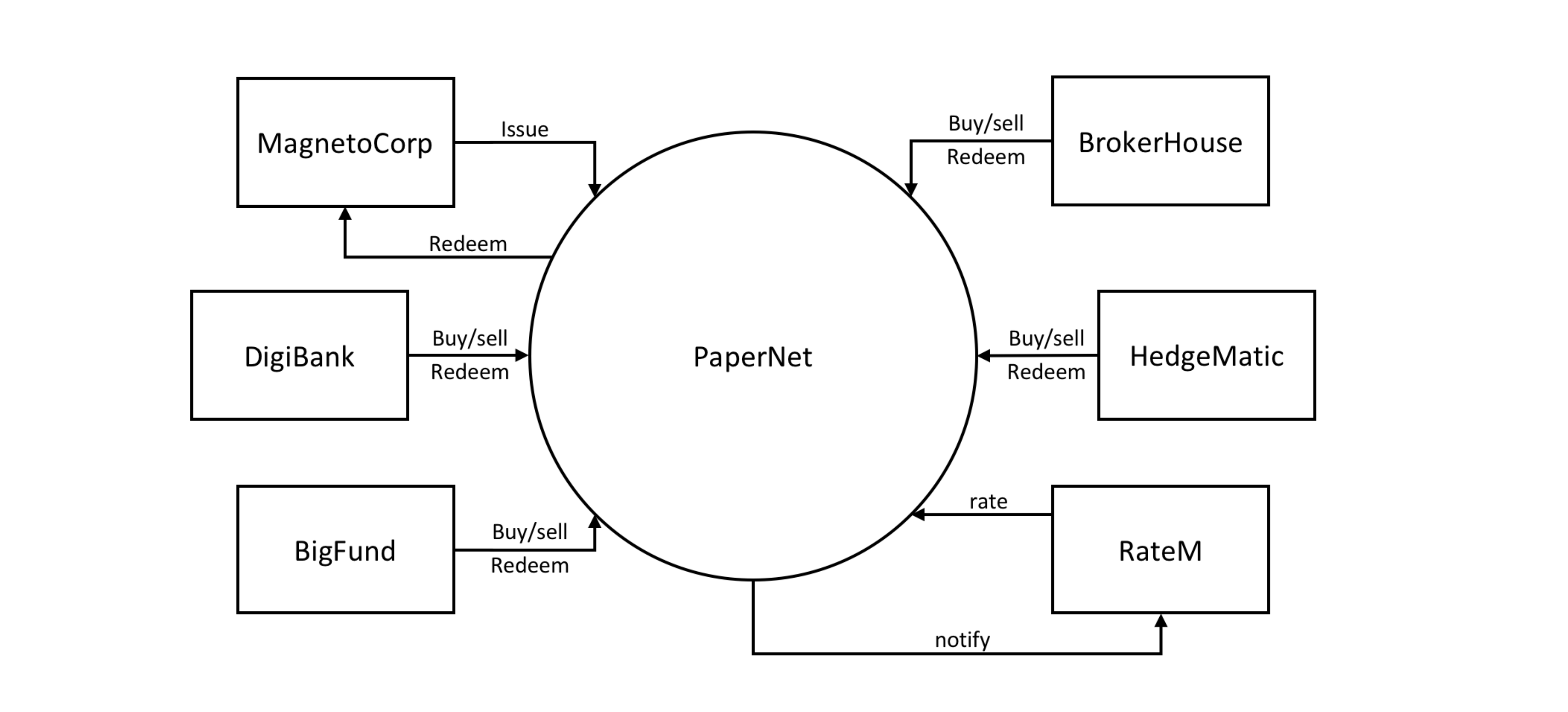
Public Blockchains

- Everything we have covered so far are public blockchains
- Anyone can read from them (except cryptonote – more on Thu)
- Anyone can make a transaction to write (pending consensus)
- Open access state machine

Private Blockchains

- At the other end of the spectrum, we have private blockchains
- Not everyone can read or write
- Multiple parties can exist in the network, but network control is *asymmetric*; some parties have higher control over the network
- They are also called permissioned blockchains as each stakeholder has a set of permissions associated with them

Private Blockchains: HLF



HLF State

- Assets: State variables; stored as k-v pairs
- Chaincode: state transitions within a channel (!= smart contract). Code that modifies the chain. Results in a set of k-v changes to submit to other peers and apply to the ledger

HLF Consensus

- PoW/PoS *probabilistic* consensus: ledger consistency is achieved with a high probability

  - Transactions are ordered based on the miner fee, etc.

- HLF has a *deterministic* consensus; has an *orderer* node that determines next txs to be included in a block
- This way, execution and ordering of transactions are separated
- There can be multiple ordering nodes, and they need to have the same order. Solved by Single-leader approach (recall dist. DBs)

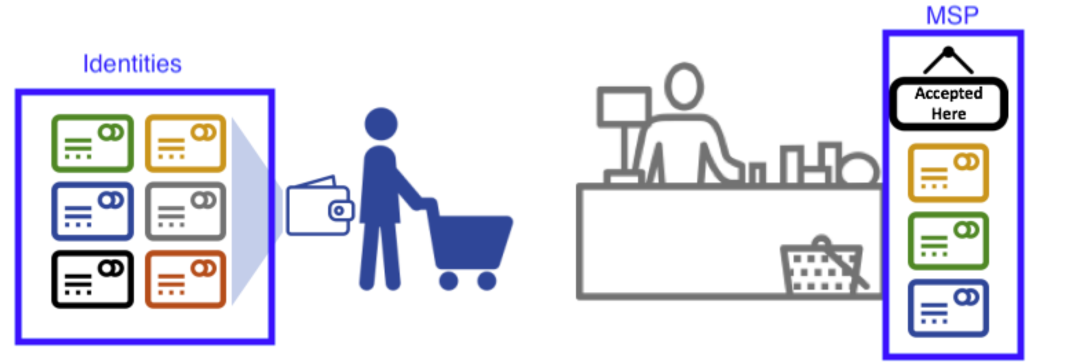
  - RAFT; PAXOS

HLF: Channel

- Allow a group of participants to maintain a separate ledger
  - The ledger exists in the scope of a channel
- Only participants belonging to a channel have copies of the channel ledger
- Can also be shared across all participants
- Allows corps to protect private interests
- Channels also allow for *collections* that let some members of a channel create a database within the ledger that only some corps can access

HLF: Security/MSP

- Membership Service Provider
- Think of this as IAM roles in cloud except distributed
  - Various roles each with different stakeholders and levels of access
- Each Organization has an MSP



HLF: Security/MSP

(From Docs:) To transact on a Fabric network a member needs to:

- Have an identity issued by a **CA** that is trusted by the network.
- Become a member of an organization that is recognized and approved by the network members. The MSP is how the identity is linked to the membership of an organization. Membership is achieved by adding the member’s public key (also known as certificate, signing cert, or signcert) to the organization’s MSP.
- Add the MSP to a channel.
- Ensure the MSP is included in the policy definitions on the network.

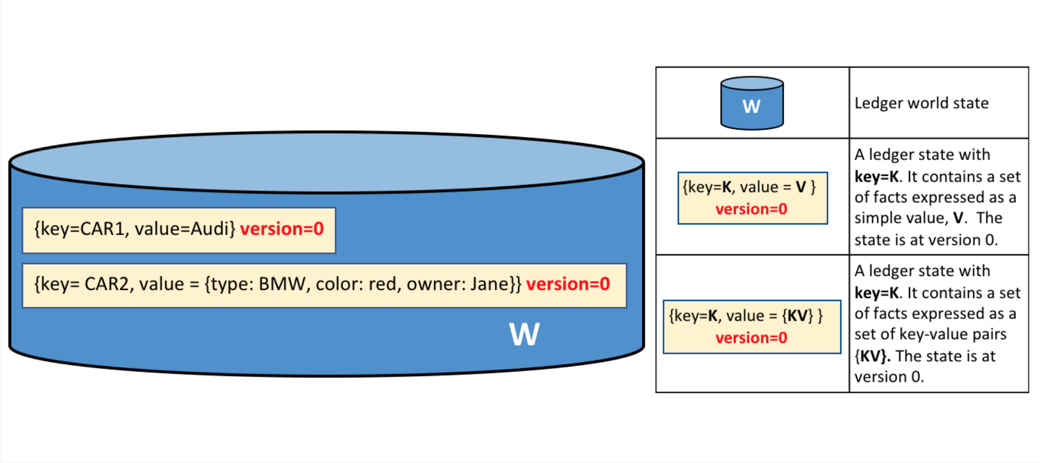
Identities
MSP
Accepted Here

HLF: Policies

- Policies define decision making structures in the HLF ecosystem
- Fabric policies represent how members come to agreement on accepting or rejecting changes to the network, a channel, or a smart contract
- Policies are agreed to by the channel members when the channel is originally configured, but they can also be modified as the channel evolves. For example, they describe the criteria for adding or removing members from a channel, change how blocks are formed, or specify the number of organizations required to endorse a smart contract.

HLF: World State

- Most up-to-date ledger state in a channel is stored as k-v pairs in a DB
- Allows for complex queries to be made on ledger state
- Applications invoke a smart contract to access world state
- "Only transactions that are **signed** by the required set of **endorsing organizations** will result in an update to the world state"



{key=CAR1, value=Audi} version=0

{key= CAR2, value = {type: BMW, color: red, owner: Jane}} version=0

W

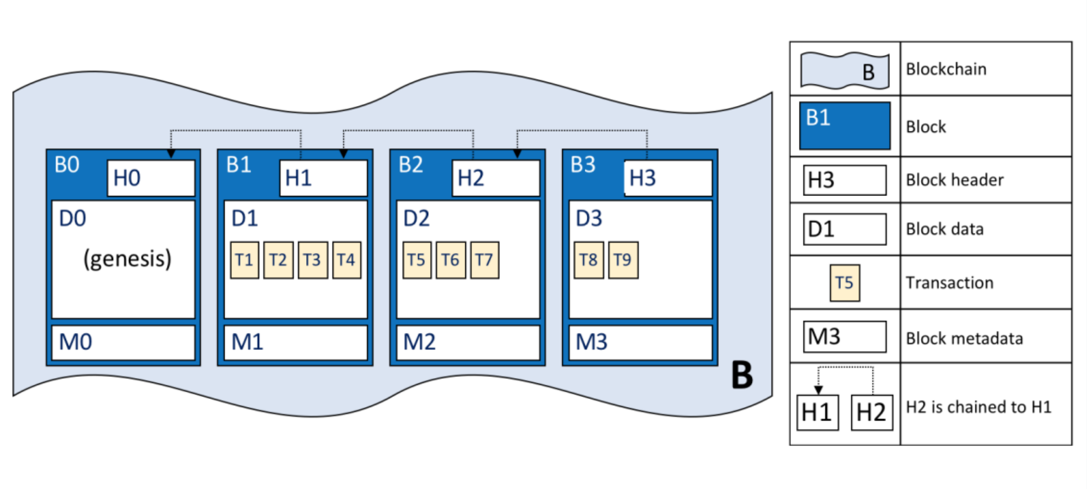
W	Ledger world state

{key=**K**, value = **V** }
version=0
	A ledger state with **key=K**. It contains a set of facts expressed as a simple value, **V**. The state is at version 0.

{key=**K**, value = {**KV**} }
version=0
	A ledger state with **key=K**. It contains a set of facts expressed as a set of key-value pairs {**KV**}. The state is at version 0.

HLF: Blockchain

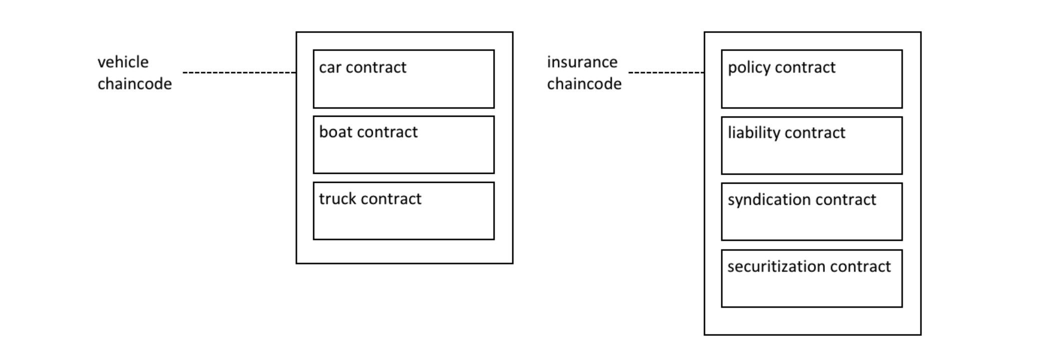
- Blockchain itself is a file that generates world state
- Each block has a link to the previous one's hash (as with all other blockchains)
- Block metadata created by ordering node + policy



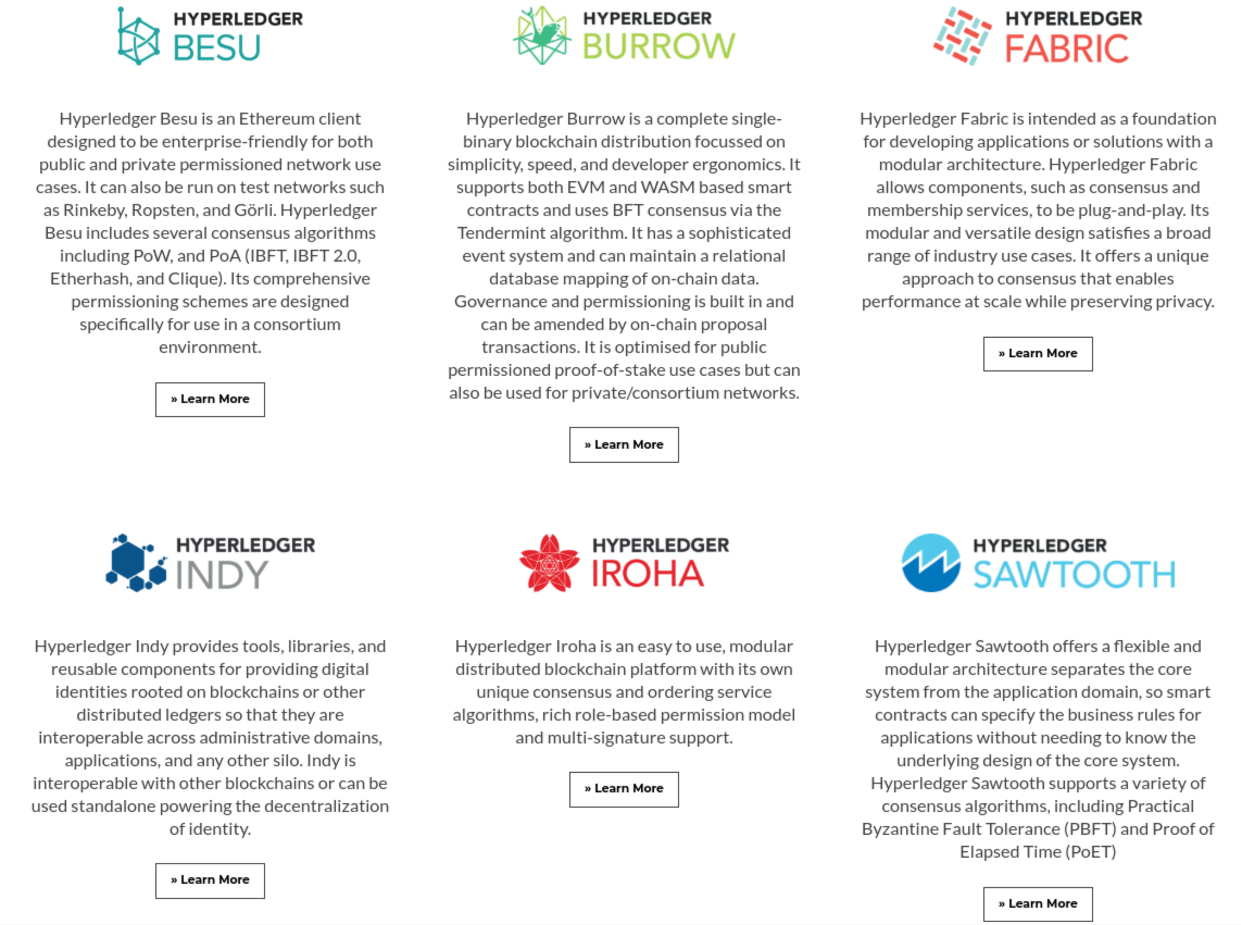
HLF: Smart Contract vs Chaincode

- Smart contract defines transaction logic that controls lifecycle of an asset in world state

- Smart contracts are packaged into chaincode and then deployed



A Note on Hyperledger



HYPERLEDGER BESU

Hyperledger Besu is an Ethereum client designed to be enterprise-friendly for both public and private permissioned network use cases. It can also be run on test networks such as Rinkeby, Ropsten, and Görli. Hyperledger Besu includes several consensus algorithms including PoW, and PoA (IBFT, IBFT 2.0, Etherhash, and Clique). Its comprehensive permissioning schemes are designed specifically for use in a consortium environment.

» Learn More

HYPERLEDGER BURROW

Hyperledger Burrow is a complete single-binary blockchain distribution focussed on simplicity, speed, and developer ergonomics. It supports both EVM and WASM based smart contracts and uses BFT consensus via the Tendermint algorithm. It has a sophisticated event system and can maintain a relational database mapping of on-chain data. Governance and permissioning is built in and can be amended by on-chain proposal transactions. It is optimised for public permissioned proof-of-stake use cases but can also be used for private/consortium networks.

» Learn More

HYPERLEDGER FABRIC

Hyperledger Fabric is intended as a foundation for developing applications or solutions with a modular architecture. Hyperledger Fabric allows components, such as consensus and membership services, to be plug-and-play. Its modular and versatile design satisfies a broad range of industry use cases. It offers a unique approach to consensus that enables performance at scale while preserving privacy.

» Learn More

HYPERLEDGER INDY

Hyperledger Indy provides tools, libraries, and reusable components for providing digital identities rooted on blockchains or other distributed ledgers so that they are interoperable across administrative domains, applications, and any other silo. Indy is interoperable with other blockchains or can be used standalone powering the decentralization of identity.

» Learn More

HYPERLEDGER IROHA

Hyperledger Iroha is an easy to use, modular distributed blockchain platform with its own unique consensus and ordering service algorithms, rich role-based permission model and multi-signature support.

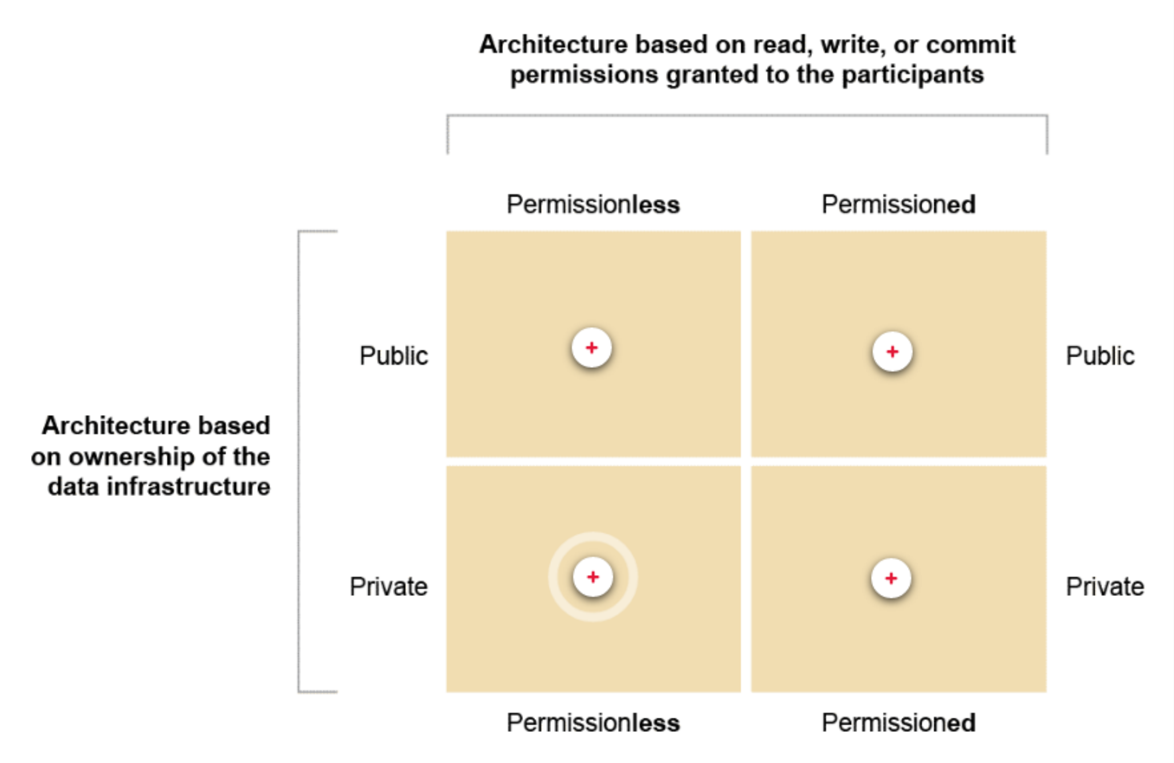
» Learn More

HYPERLEDGER SAWTOOTH

Hyperledger Sawtooth offers a flexible and modular architecture separates the core system from the application domain, so smart contracts can specify the business rules for applications without needing to know the underlying design of the core system. Hyperledger Sawtooth supports a variety of consensus algorithms, including Practical Byzantine Fault Tolerance (PBFT) and Proof of Elapsed Time (PoET)

» Learn More

Permissioning



Architecture based on read, write, or commit permissions granted to the participants

Architecture based on ownership of the data infrastructure

	Permissionless	Permissioned	
Public	+	+	Public
Private	+	+	Private
	Permissionless	Permissioned	

Public Permissioned

- Anyone on the internet can access and write – if they have the right perms

- E.g Steem. Once joined, users can access services like Steemit.

Private Permissionless

- Anyone can spin up a node to join
- However, unlike on a public blockchain, other nodes will only acknowledge its existence, but not share any data
- The smart contracts on these private networks, not only define who is allowed to perform contract actions but also who is allowed to read the contract and all related data
- A single node holds multiple of ad-hoc chains, but never all of them
- E.g. Holochain, a platform for decentralized applications, where users share information peer-to-peer on a "need-to-know" basis

Consortium Ledgers

- Consortium ledgers are a hybrid between private and public ledgers
- Differ from public blockchains as there is a hierarchy of nodes; not all nodes are involved in transaction validation
- Differ from private blockchains as they have multiple parties that control access to the ledger as opposed to just one party controlling the ledger
- Consortiums typically have industry-wide applications whereas private ledgers are best suited to support a group of companies that require immutability and transparency in their dealings
- E.g. Hedera, Facebook, Diem (or at least Libra)

Questions/Comments?

Let's code!