Blockchain - To Develop Digital Resource Management Application

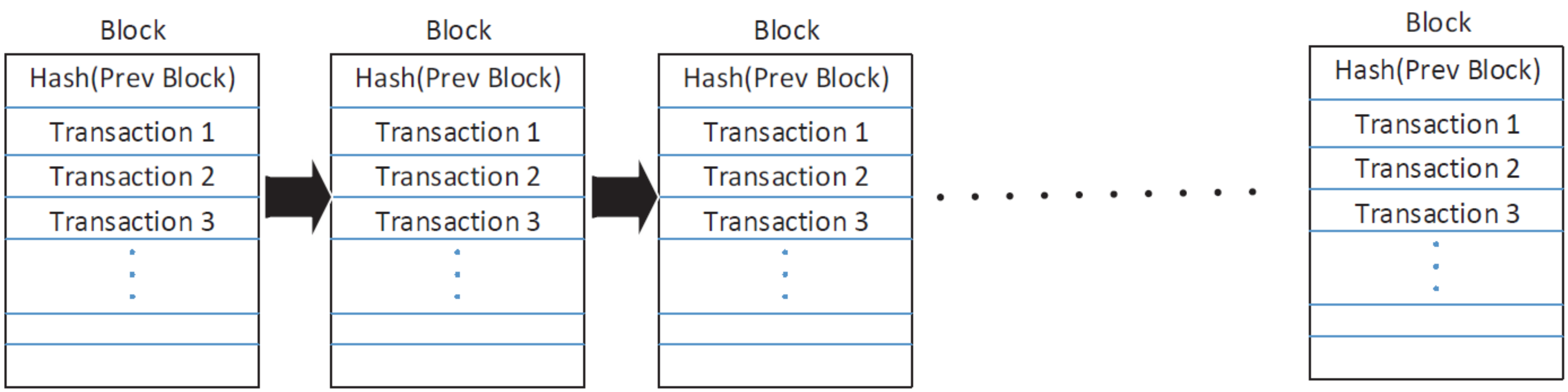
Kaushik Dutta
Information Systems and Decision Sciences
Muma College of Business

Outline

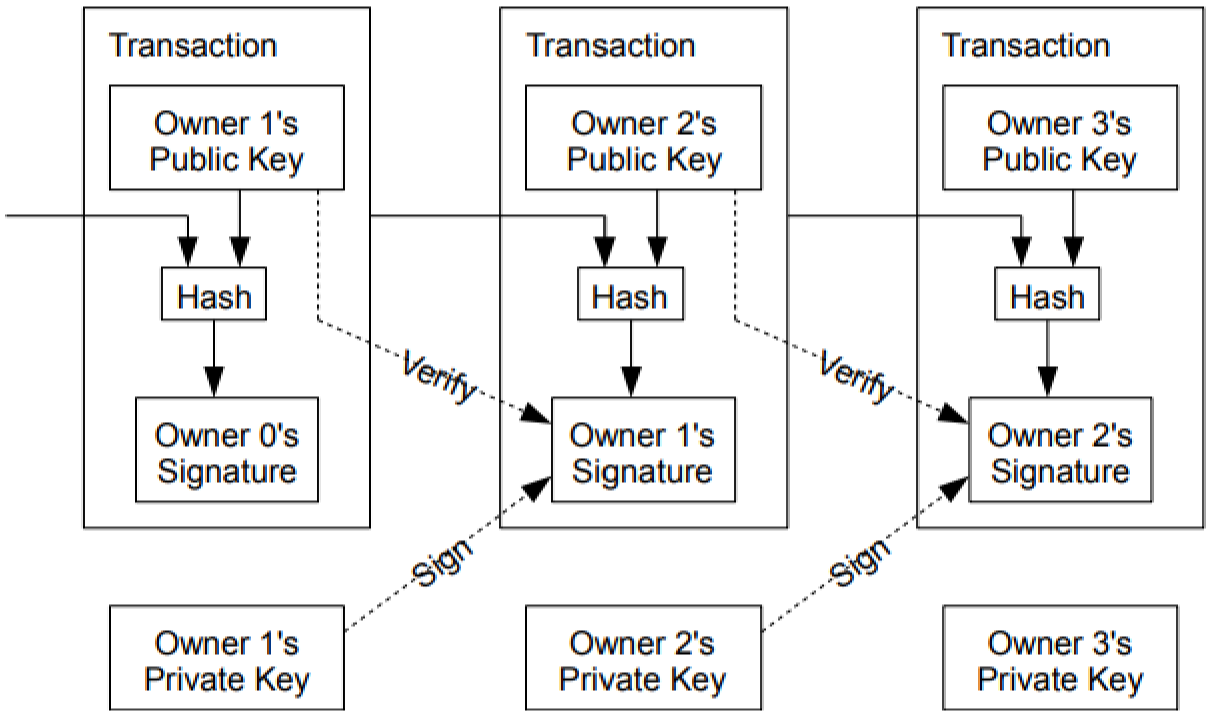
- Fundamentals of blockchain
  - Introduction
  - Data reliability
  - Cybercurrency
    - Public and private blockchain
  - Blockchain contract
  - Scalability
- Resource management using blockchain
  - Motivation
  - Generic Framework
  - Implementation using multichain
  - Use Case
- Conclusion
  - Summary

What is blockchain?

- Distributed ledger
  - A collection of transactions are written in blocks (similar to disk block or database block or HDFS block)
  - Each block is coupled with the previous block by hash value of previous block

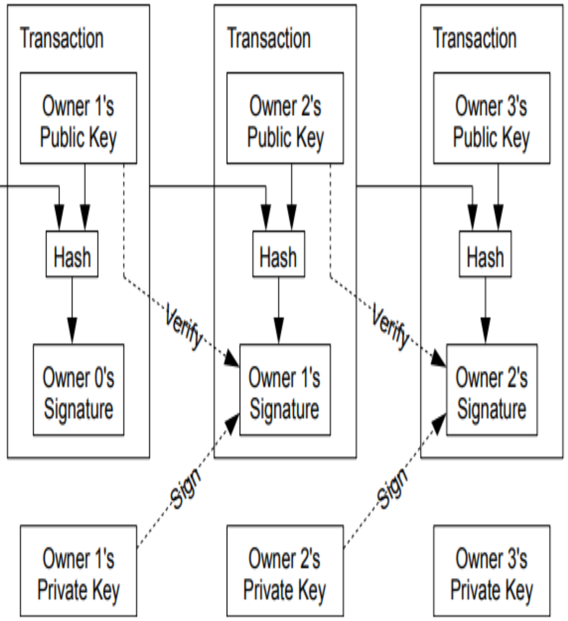


What is blockchain?



What is blockchain?

- Each node collects new transactions into a block.
- When node creates a block, it broadcasts the block to all nodes.
  - Blocks are created by nodes in randomized round robin fashion
  - Each block is signed by the creator
- Nodes express their acceptance of the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.
  - In case of conflict, the longest chain only survives.
    - Longest chain means more nodes have accepted the blocks
    - Majority consensus
      - Improves reliability and authenticity



Majority Consensus

Node 1 Block 1 → Block 2 → Block 3

Node 2 Block 1 → Block 2 → Block 3

Node 3 Block 1 → Block 2 → Block 3

Node 4 Block 1 → Block 2 → Block 3

Node 5 Block 1 → Block 2 → Block 3

Majority Consensus

Node 1 Block 1 → Block 2 → Block 3 → Block 4

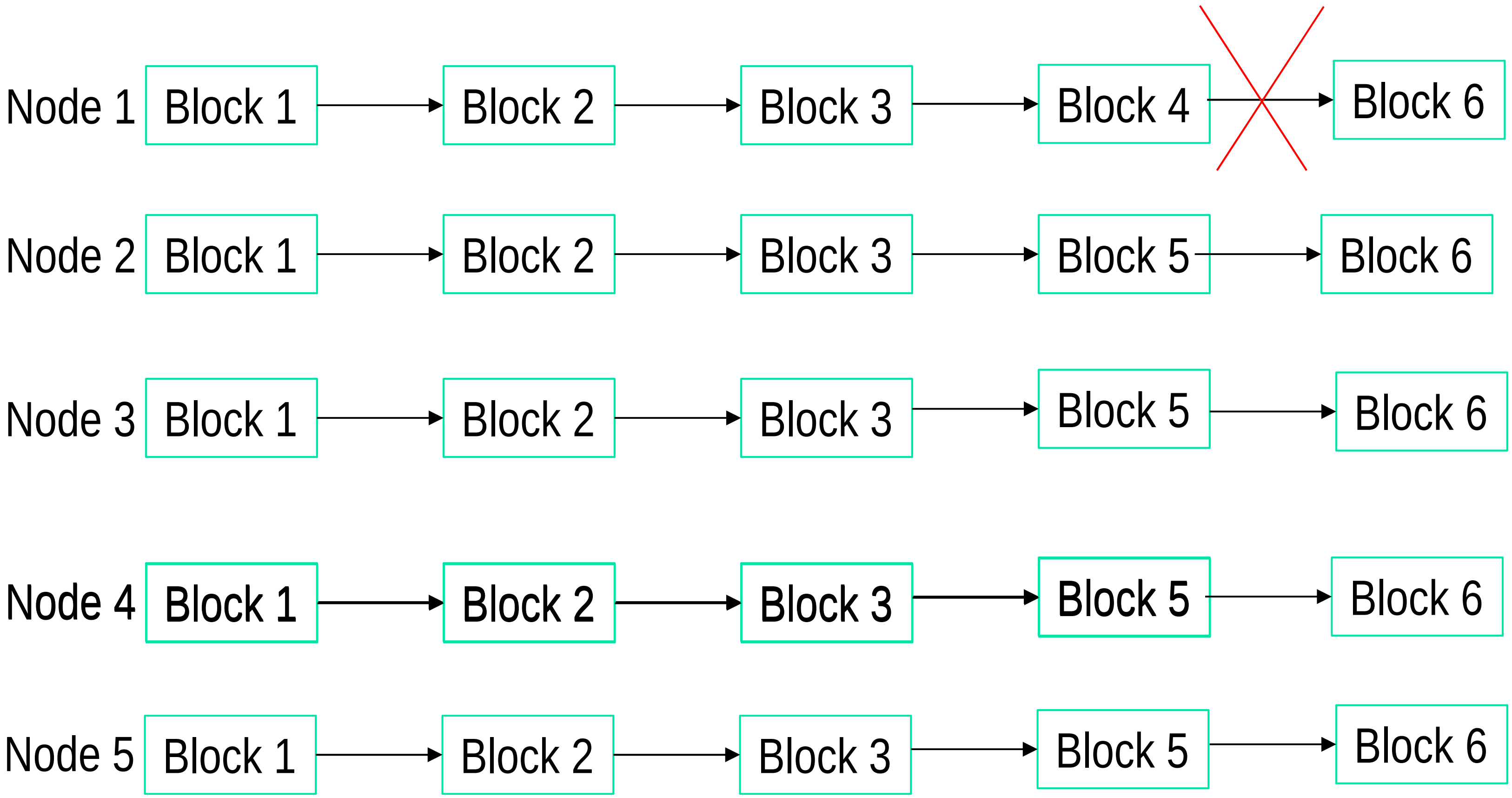
Node 2 Block 1 → Block 2 → Block 3 → Block 5

Node 3 Block 1 → Block 2 → Block 3 → Block 5

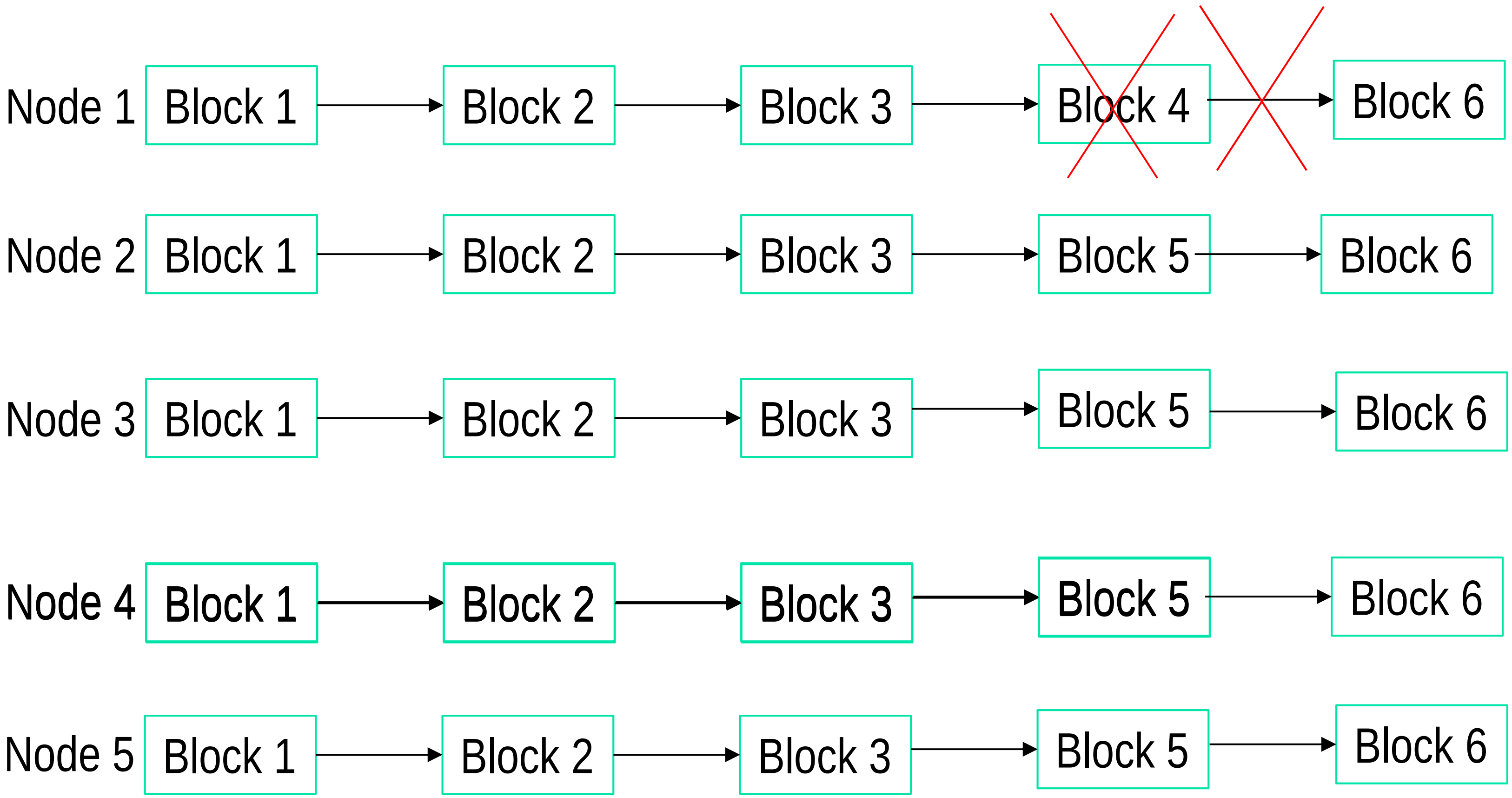
Node 4 Block 1 → Block 2 → Block 3 → Block 5

Node 5 Block 1 → Block 2 → Block 3 → Block 5

Majority Consensus



Majority Consensus



Majority Consensus

Node 1 Block 1 → Block 2 → Block 3 → Block 5 → Block 6

Node 2 Block 1 → Block 2 → Block 3 → Block 5 → Block 6

Node 3 Block 1 → Block 2 → Block 3 → Block 5 → Block 6

Node 4 Block 1 → Block 2 → Block 3 → Block 5 → Block 6

Node 5 Block 1 → Block 2 → Block 3 → Block 5 → Block 6

Reliability of Data in Blockchain

- Depends on the collective good behavior of nodes

- What if majority nodes collude?
  - System can't protect from such scenarios

- Modifying a particular transaction in a block will require modifying all the previous blocks in that chain.
  - A computationally expensive but not impossible task
  - All other nodes need to agree on that change and continue to write in that modified chain

Bitcoin / Ethereum

- Need further protection than nodes colluding
- Creation of each block requires a predefined meaningless computation, as proof of work
  - Distributes the onus of creating blocks across the network
  - A level of difficulty is associated with each block
- The chain with higher proof of work survives
- Modifying a particular transaction in a block or replacing with alternate data will require more computational power than other nodes combined together
  - Just collusion is not enough, the colluded nodes need to invest to mine
    - $$$$$$$$
  - A non-trivial but not impossible task
    - Possible by large rogue organizations with financial power

Private vs. Public Blockchain

- Public blockchain – Bitcoin, Ethereum
  - Anyone with $ can participate
    - $ is needed to compensate the miner (block creator for doing the work of creating blocks – "proof of work")
  - Protected by "proof of work" or the "work to be done" to modify
  - Associated with a currency to compensate the "proof of work"
    - The underline asset of the currency is "computational work" that has been done
      - Like gold backed currency – backed by the work that has been done to mine the gold.
- Private Blockchain – Multichain, Hyperledger
  - Restricted to a group
  - Participation is based on common business interest
  - Protected by various consensus mechanism
  - Does not need to be associated with a currency

Blockchain Contract

- Contract is an executable code like a PL/SQL code
- Contract can be written in various languages and is run in Virtual Machines in Blockchain (EVM in case of Ethereum)
- Contract is executed by blockchain nodes based on incoming transactions and can write more transactions in the blockchain
- Serves following purpose:
  - expressing business logic as a computer program
  - representing the events which trigger that logic as messages to the program
  - using digital signatures to prove who sent the messages
  - putting all of the above on a blockchain.

Blockchain – Issues of Scalability

- Blockchain is not a true distributed system, it is a replicated system
  - all of the nodes that maintain the blockchain do *exactly the same thing*
    - They verify the same transactions
    - They record the same items into a blockchain
    - They store the entire history, which is the same for all of them, for all time
- Blockchain contract is executed independently in each and every node

- Bitcoin: 7 Transactions per seconds
- Ethereum: 10-30 Transactions per seconds
- Multichain: 500-1000 transactions per seconds

Blockchain – Issues of Scalabiity

Cryptocurrencies Transaction Speeds Compared to Visa & Paypal

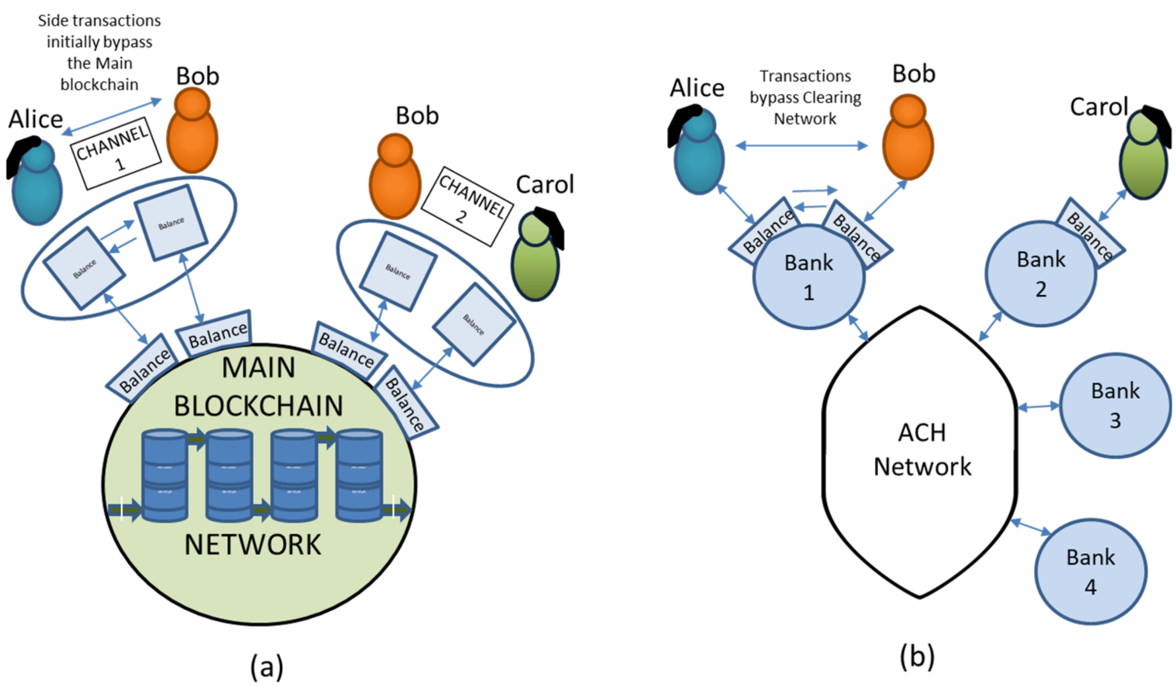
Article & Sources:
https://howmuch.net/articles/crypto-transaction-speeds-compared
https://howmuch.net/sources/crypto-transaction-speeds-compared

Blockchain – Issues of Scalability

- Culprits:
  - Proof of work
  - Contract Execution
  - Each node stores and executes everything

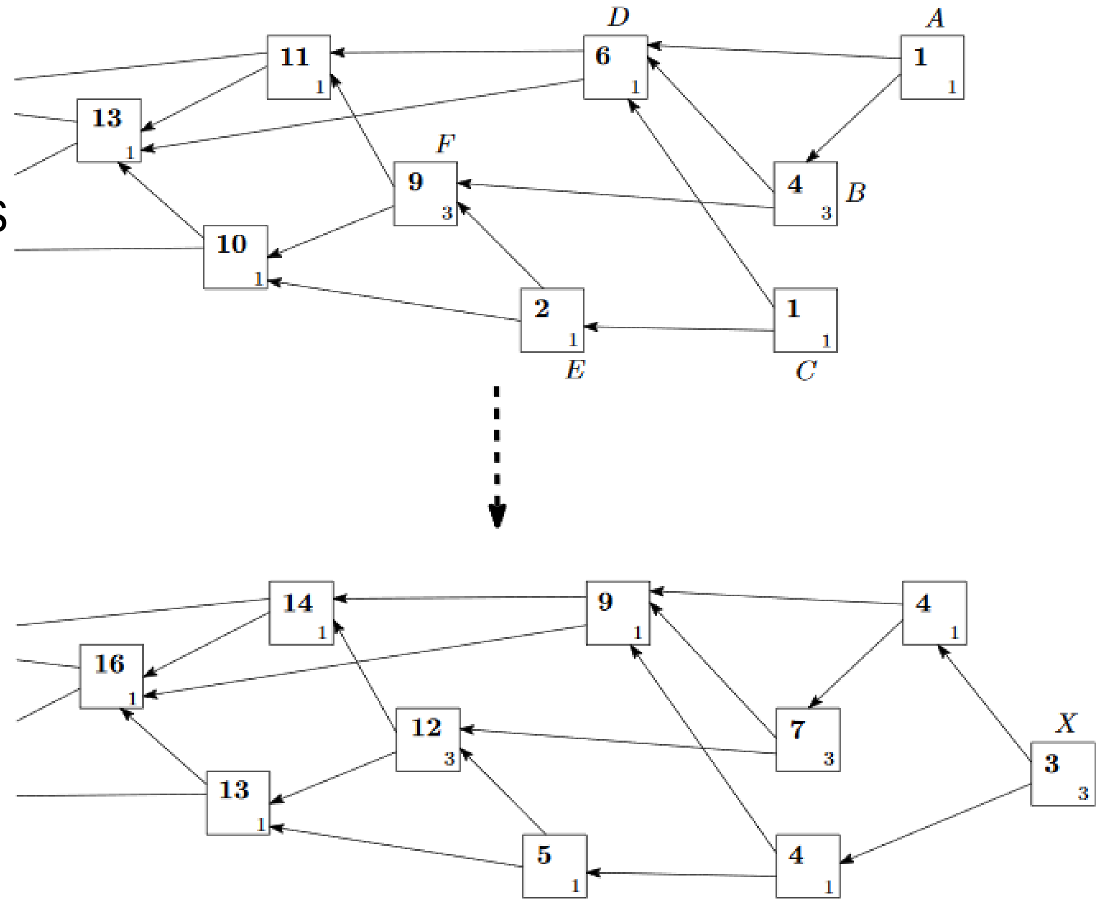
Blockchain – Solution to Scalability

- Sidechain



Blockchain – Solution to Scalability

- Tangle / IOTA
  - No mining
  - No blocks
  - No real-time consensus
- Nodes validate two old transactions to conduct their own
- To be in the network you have to participate actively in validating transactions



Some comparisons

Characteristic	Ethereum	Hyperledger Fabric	R3 Corda
Description of platform	– Generic blockchain platform	– Modular blockchain platform	– Specialized distrib-uted ledger platform for financial industry
Governance	– Ethereum developers	– Linux Foundation	– R3
Mode of operation	– Permissionless, public or private[4]	– Permissioned, private	– Permissioned, private
Consensus	– Mining based on proof-of-work (PoW) – Ledger level	– Broad understand-ing of consensus that allows multiple approaches – Transaction level	– Specific understand-ing of consensus (i.e., notary nodes) – Transaction level
Smart contracts	– Smart contract code (e.g., Solidity)	– Smart contract code (e.g., Go, Java)	– Smart contract code (e.g., Kotlin, Java) – Smart legal contract (legal prose)
Currency	– Ether – Tokens via smart contract	– None – Currency and tokens via chaincode	– None

Outline

- Fundamentals of blockchain
  - Introduction
  - Consensus & data reliability
  - Cybercurrency
    - Public and private blockchain
  - Blockchain contract
  - Scalability
- Resource management using blockchain
  - Motivation
  - Generic Framework
  - Implementation using multichain
  - Use Case
- Conclusion
  - Summary

Resources

- Any digital asset that can be accessed and used for business purposes
  - Patient record in EHR system
  - Educational Transcripts
  - Bills, Orders, Invoices

Resource Management Using Blockchain

- Access to digital resources is not under the owner's control.
- Third party sellers have made a business out of obtaining and selling information
- Proposed framework
  - The owner maintains control over the access of the resource.

Why Blockchain

- To manage the ground truth
  - Which data is the truth
    - *Truth* – that majority agrees on (democracy!!)
    - *Alternate truth* – other data that few agrees on
- To have a multi-node storage where all nodes are equal
  - Important for databases to support a consortium of multiple organizations
  - No third party entity with overhead is needed
    - Reduces the cost of data management

Resources - Creator, Owner, User

- Creator (c) : This entity is responsible for creating the resource
- Owner (o): This entity is the owner of the resource and has the right to control who can access which of his resource
- User (u): The user is a user of the resource

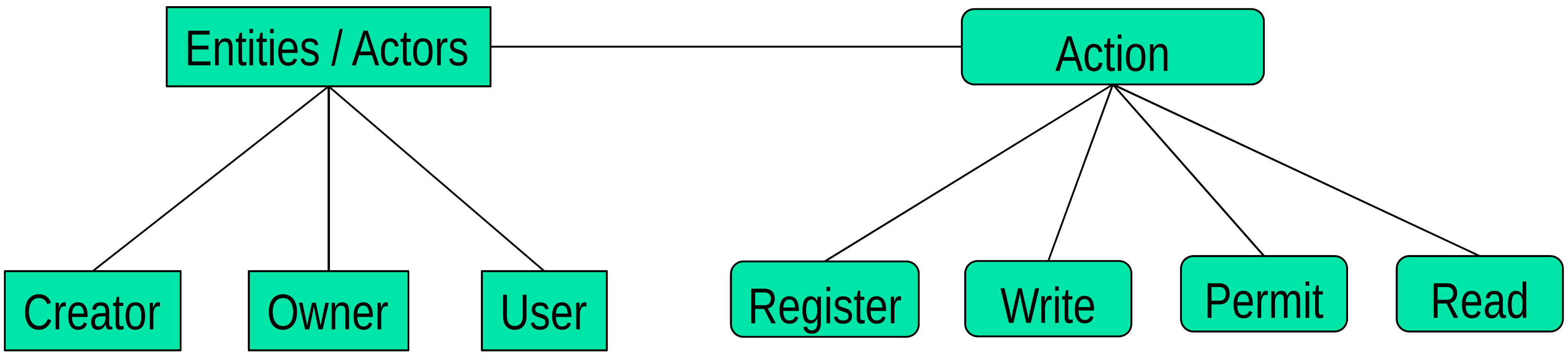
Steps

- Creator, Owner and User register with the system
- Creator writes a resource and assigns to owner
- Owner permits a resource to be accessed by a user
- User reads the resource and use it for business purpose

Operations

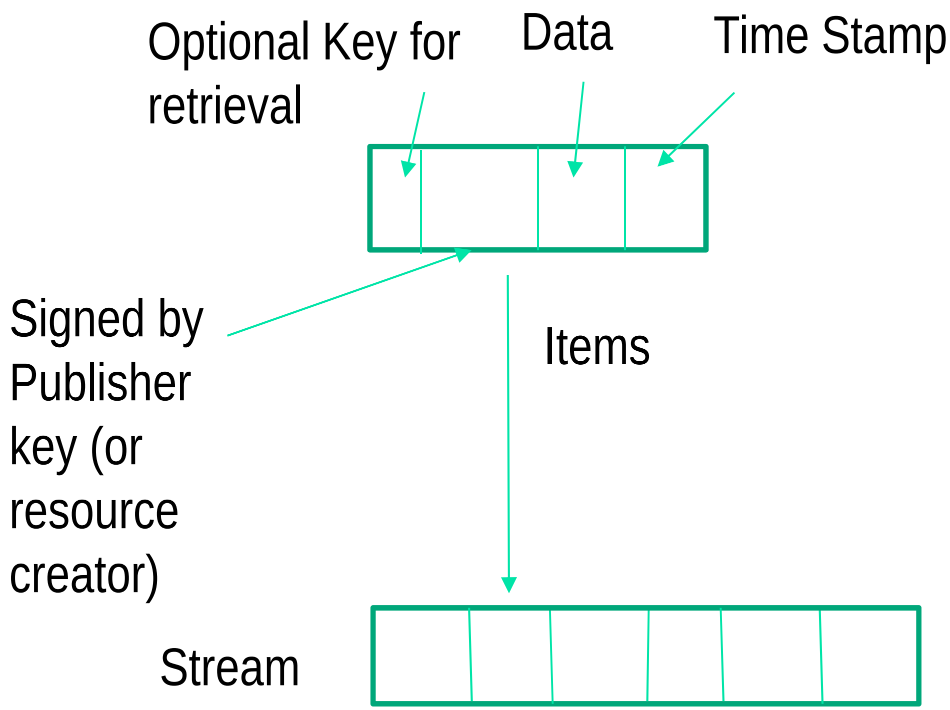
- Register(u) / Register (o)
  - Register a user $u \in U$ or owner $o \in O$
- Write (d, c, o)
  - Creator $c \in C$ writes the data d associated with the owner $o \in O$.
- Permit (d, o, u)
  - Owner $o \in O$ provides the read access of information d for the user $u \in U$.
- Read(d, u)
  - User $u \in U$ can read the information d.

Resource Management Framework



Multichain Implementation

- Multichain – private blockchain
  - Stream
    - Hashtable mechanism in multichain for fast lookup of a transaction
    - Used to manage, store and lookup information in multichain
  - Each item in a stream is
    - Digitally signed by the publisher (Resource Creator)
    - Can be associated with an optional key, which can be used later for retrieval by the user u.
    - Some data (or information) can be stored in each item.
    - Is associated with a timestamp when the item is being written



Store and Manage Information in blockchian

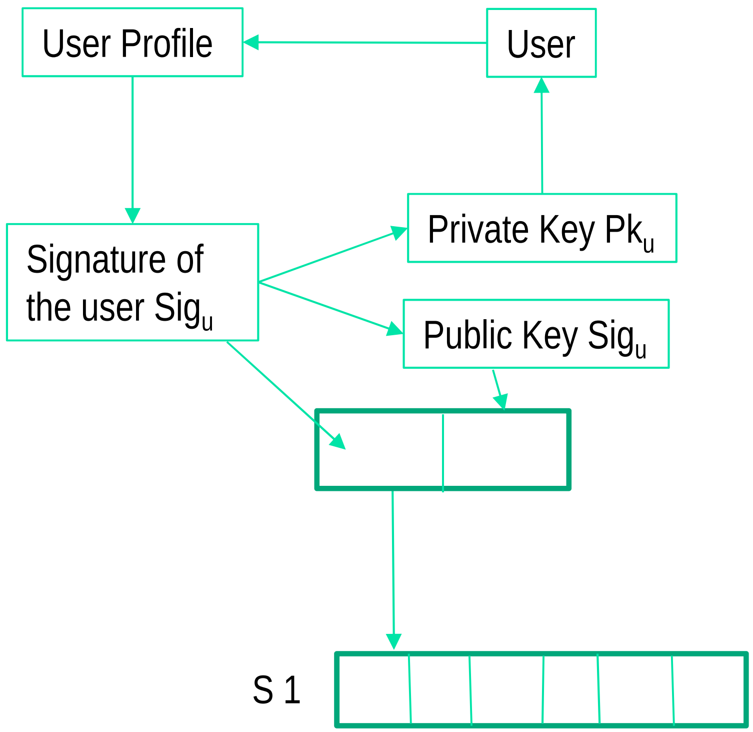
You can only append items to the stream

Multichain Implementation

- S1 – To store and manage user's public key and profile
- S2 – Store the digital resource
- S3 – Assigns owner to digital resource
- S4 – Provides access to digital resource to other users

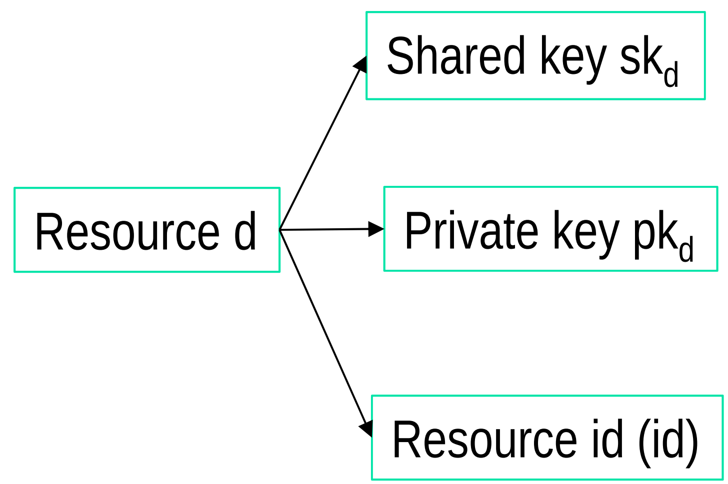
Register(u) / Register(o) – S1

- A user u creates profile after registration;
- Generate the *signature* sig_u of the user u from a hash of his profile data
- Generate *public and private key* for the user u from the user signature
- Send the private key pk_u to the user
- Publish the *signature sig_u, along with the public key of the user sk_u* to Stream 1
- Goal – All users (resource owners and users) are registered



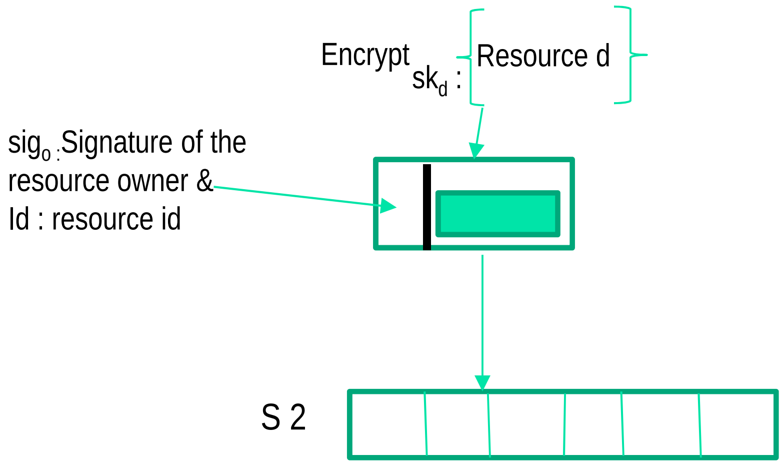
Create Resource d by Creator c

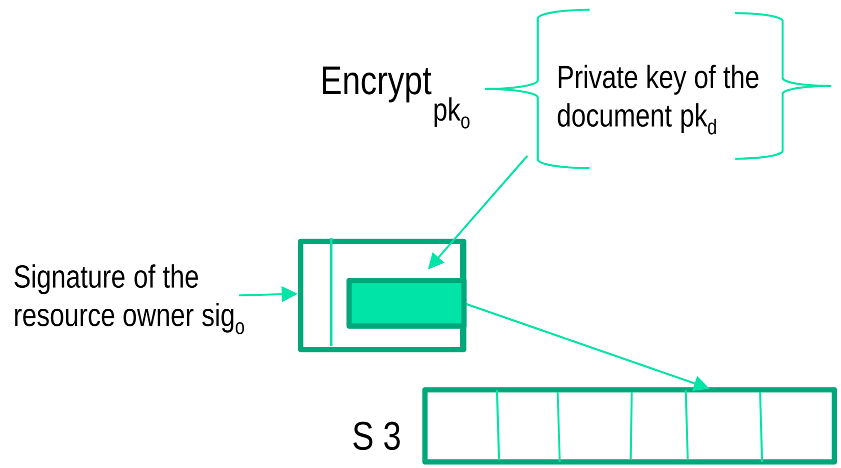
- Resource creator generates shared and private keys for the resource
  - Generates unique resource id -id
  - Generates shared key and private key (sk_d and pk_d) for the resource d



Write (resource d, by creator c, for owner o)

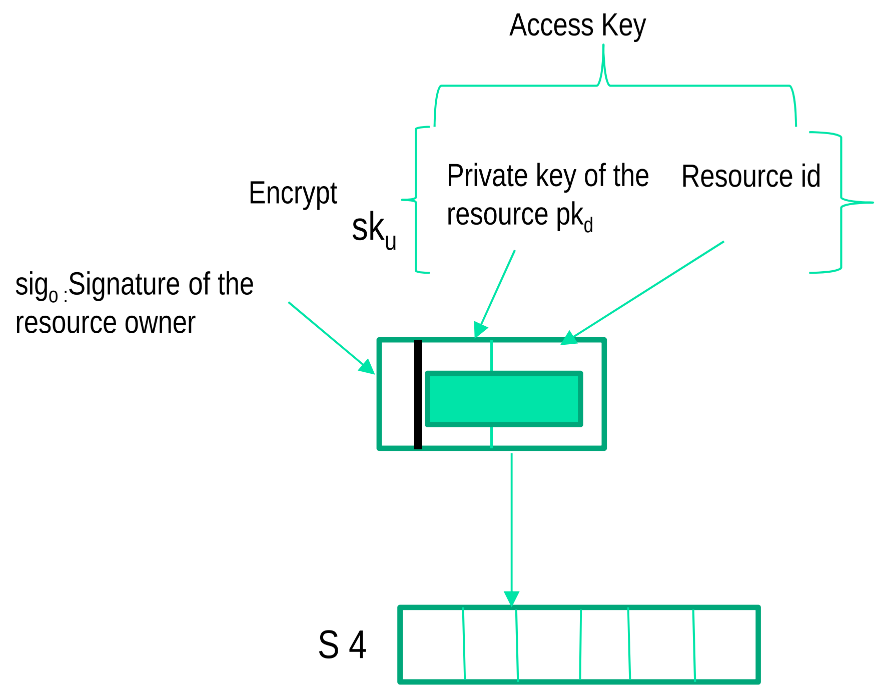
- Write resource to S2
  - Encrypt the resource d with the shared key of the resource (sk_d)
  - Publish the encrypted resource to stream S2 with the signature of the resource owner sig_o , and the resource id (id) as the key
- Assign resource to owner in S3
  - Encrypt the private key of the document pk_d with the public key of the resource owner pk_o
  - Publish the encrypted resource pk_d to stream S3, with the signature of the resource owner sig_o





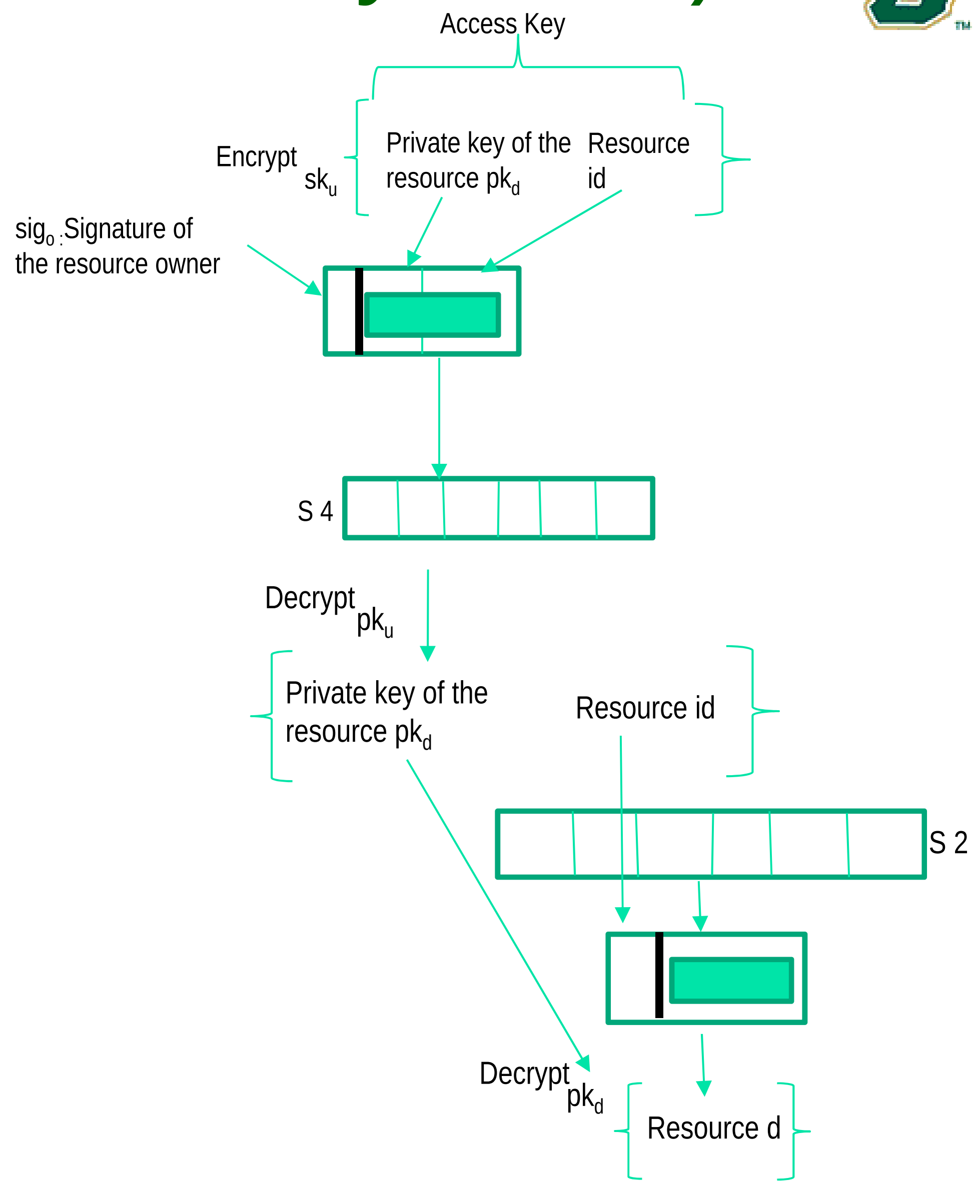
Permit (Resource d, by owner o, to user u)

- Encrypt the combination of private key of the resource (pk_d) and resource id, with the shared key of the user u (sk_u)
  - Access key
- The access key is published to the Stream S4 by resource owner o along with the signature of the owner sig_o as the key



Read (Resource d, by user u)

- The user u, using his own private key pk_u, retrieves access key by decrypting the *access key* published on Stream S4
- The private key of the resource pk_d, is retrieved from the access key by the user u
- The private key of the resource pk_d along with the resource identified id is then used to retrieve the resource stored in Stream S2;



UNIVERSITY OF SOUTH FLORIDA

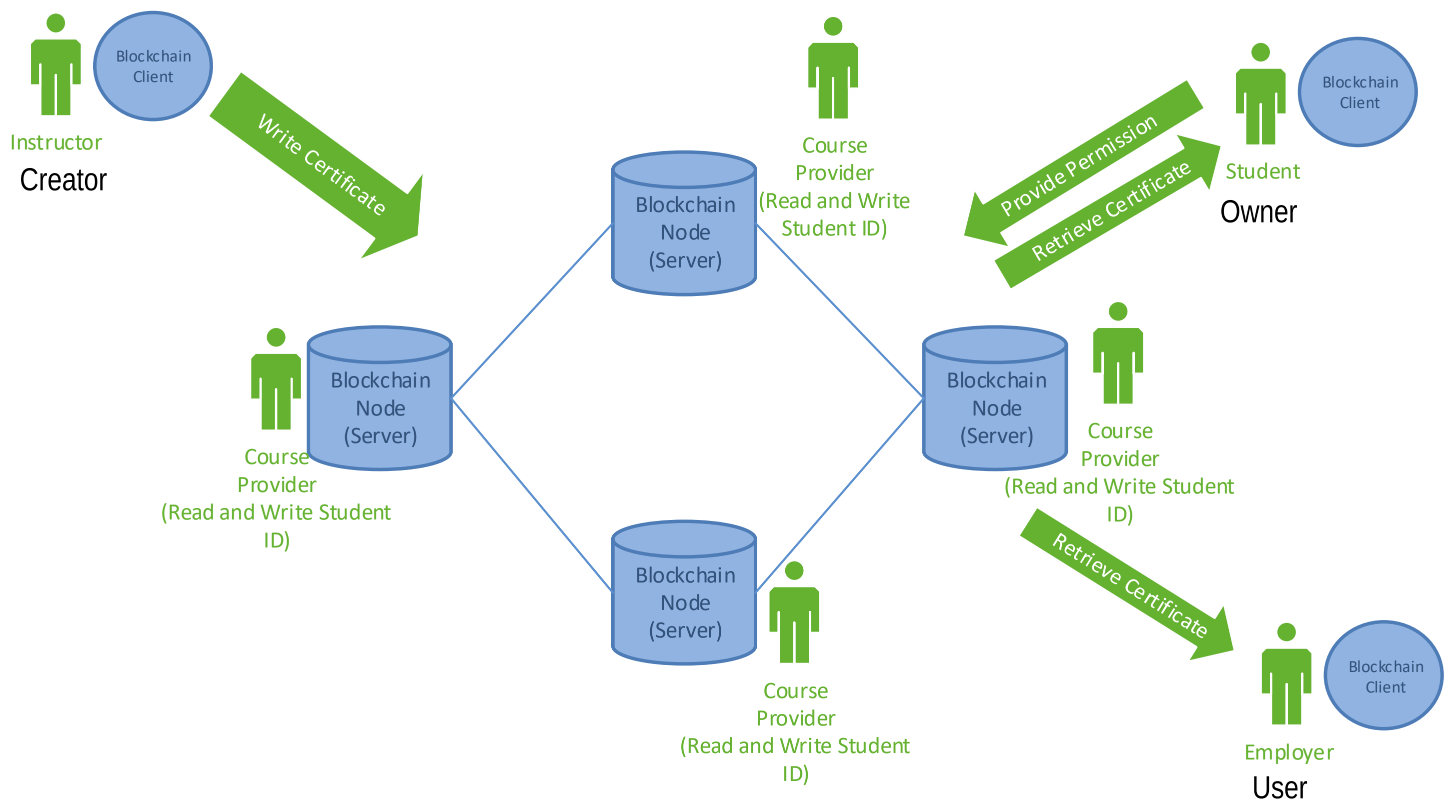
Example Blockchain Use-Case

- We propose a course and certification management system as a use case for resource management using blockchain.
  - A prospective employer has no easy way to verify the authenticity of many of the certificates that a candidate claims to have from multiple education platforms.
  - A blockchain-based system to address the problem as mentioned above without relying on a single authority or platform.

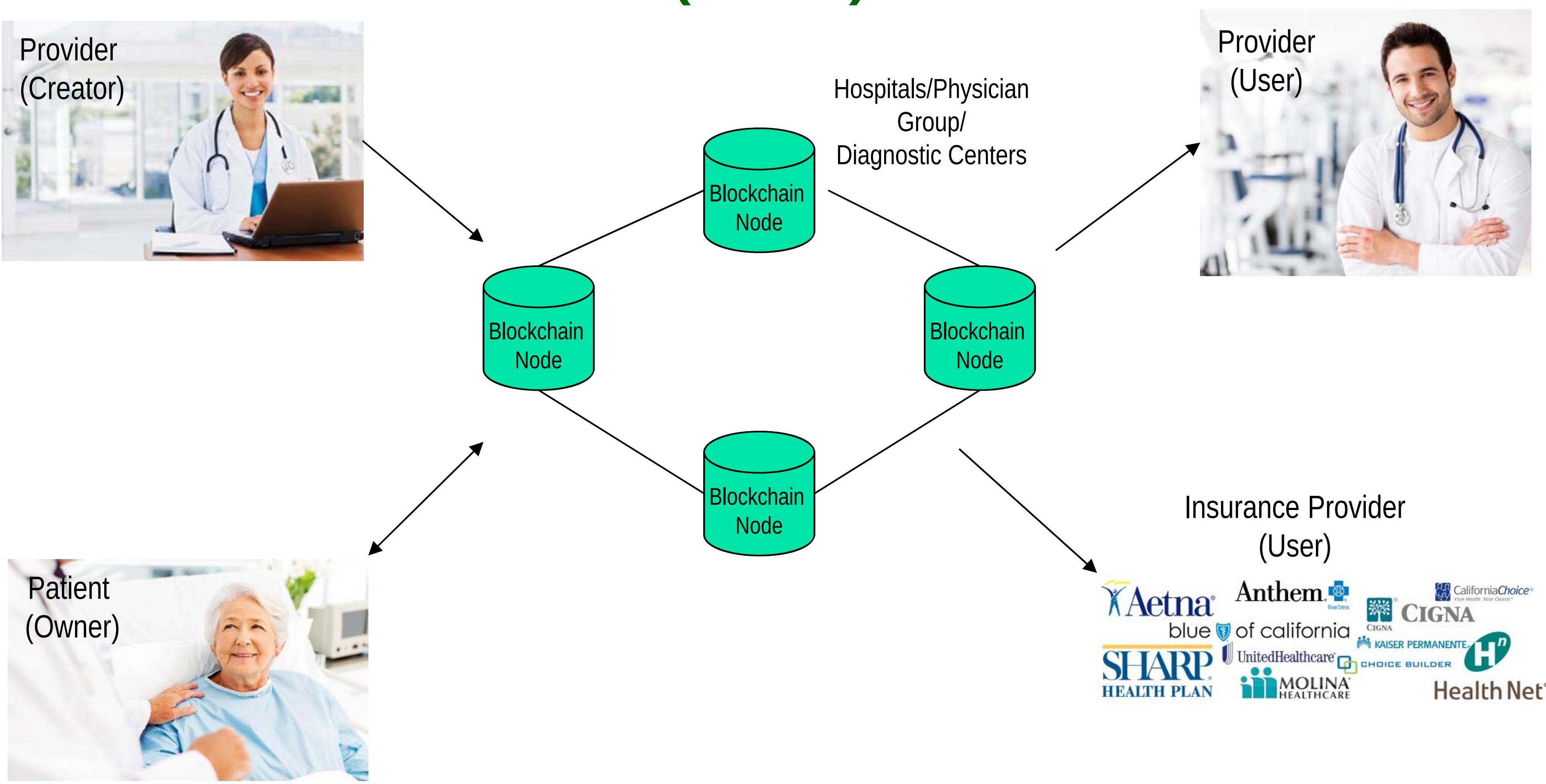
An Example Blockchain Use-Case

- **Course providers** : hosts of the course management system
  - Hosts of blockchain nodes
- **Students**: users who register into the system, enroll in courses and gain certificates upon completion of said courses.
  - Resource (digital educational certificates) <u>owner</u>
- **Instructors**: Users who grant the certificate to the Student upon them completing the requirement of a course
  - Resource <u>Creator</u>
- **Employers**: Users who with the permission from the students look to validate their certificates on the Blockchain.
  - <u>User</u>

Blockchain Architecture for the Credential Management System



Blockchain for Patient Record Sharing (EHR)



Blockchain Contract

- Execute (Event e, EventHandler f, Message m):
  - When an event "e" (a specific type of transaction) occurs, the blockchain executes the event-handler f (contract) and passes a message "m" into the blockchain (writing another transaction in the blockchain)
    - Pharmacy fulfills a prescription (record) and writes a transaction "prescription filled" with the prescription id (record id) as the key

Implementation on Hyperledger

- Model
  - Assets – Resources
  - Participants
    - Healthcare – Providers, Patients
  - Transactions
    - Providing access details
  - Events – Transactions emit events when conditions match
- Logic – to check the validity of transaction
- Query – find assets
- Access (ACL) – who can do what

Outline

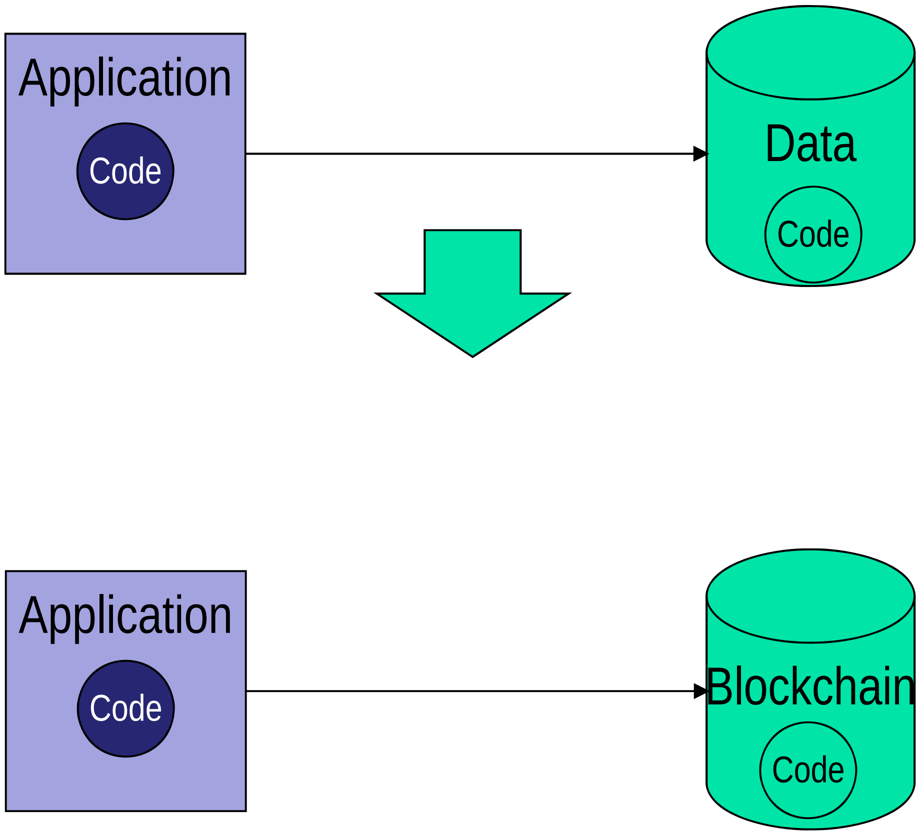
- Fundamentals of blockchain
  - Introduction
  - Consensus & data reliability
  - Cybercurrency
    - Public and private blockchain
  - Blockchain contract
  - Scalability
- Resource management using blockchain
  - Motivation
  - Generic Framework
  - Implementation using multichain
  - Use Case
- Conclusion
  - Summary

Blockchain

- <u>What it is </u>vs. <u>what it is not</u>?
  - Not a distributed system
    - All data are replicated in all nodes
  - Contract (EVM) - Not a distributed computing system
    - The same code is executed in all nodes
  - Immutable – for all practical purposes it is true
    - With sufficient financial resource it can be hacked
  - Data storage system for parties that don't trust each other – trust by consensus / proof of work / validated by others
    - For trusted parties
      - Do we need anything other than centralized data storage?
      - What's the advantage of blockchain in this case???
  - Incentivized system - custom coin / coupons

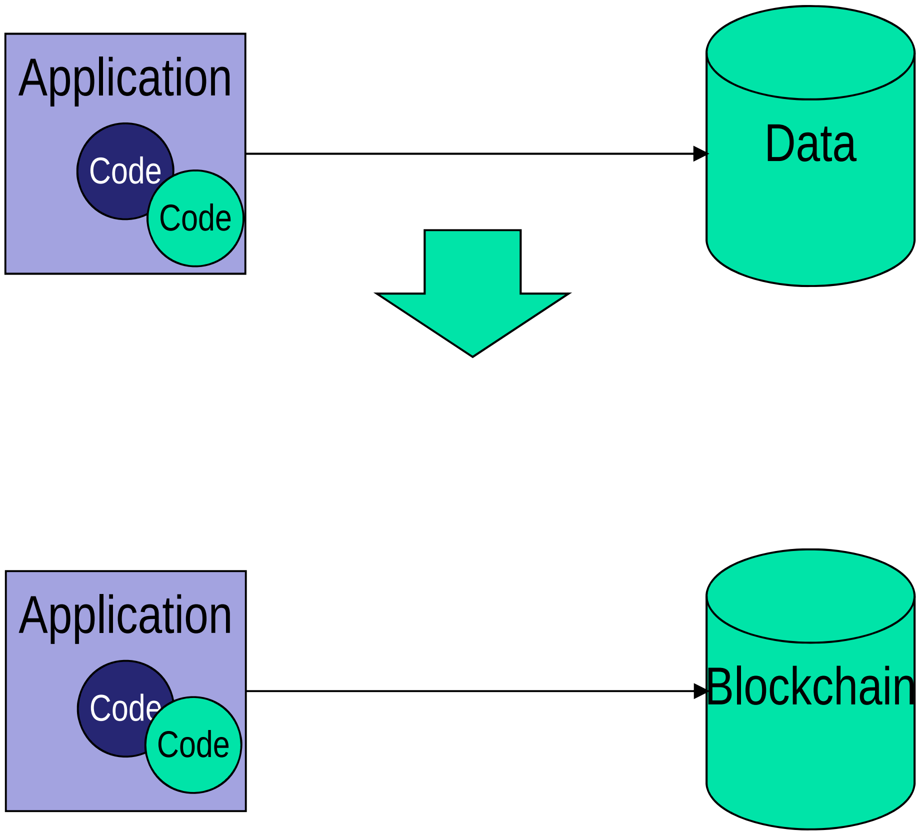
Food for thoughts

- Contracts vs. Stored Procedures



Food for thoughts

- Contracts vs. Stored Procedures



Conclusion

- Blockchain can be used to manage digital resources involving multiple parties where parties do not trust each other
  - Fine grained access control mechanism
  - Event and notification handling through blockchain contract

- Proposed a generic resource management framework
  - That will work on multiple private and public blockchain platforms
    - Multichain, Hyperledger

Thank You!!