BLOCKCHAIN AND DISTRIBUTED LEDGER TECHNOLOGIES

Different Blockchains

Dhiren Patel
VJTI Mumbai

WORLD BANK GROUP

VEERMATA JIJABAI TECHNOLOGICAL INSTITUTE
V.J.T.I.
ESTD 1887
MUMBAI 400019

DeLight Chain

HAW HAMBURG

LEARNING OUTCOMES

After this lecture students shall

- **Know**
  - types and classification of blockchains
  - about foundation of various blockchains
  - security concerns
  - mapping of blockchains to application domains
- **be able**
  - to describe evolution steps and major types of blockchains

WHAT MAKES THE BLOCKCHAINS DIFFERENT FROM EACH OTHER?

- **Openness**
  - **Public**
    Anybody (full node user) can add blocks to the blockchain (e.g. Ethereum, Bitcoin, Litecoin etc.)
  - **Permissioned**
    Only designated trusted nodes can add blocks to the blockchain
    - Consortium based (e.g. R3 Corda, JP Morgan-Quorum)
    - **Private** (e.g. IBM – Hyperledger Fabric, Multichain – Multichain, IOTA - IOTA etc.)

- **Consensus Protocols**
  - Proof of Work – PoW (Ethereum, Bitcoin, IOTA etc.)
  - Proof of Stake – PoS (Peercoin, Ethereum Next Generation Casper etc.)
  - Proof of Elapsed Time – PET (Hyperledger Sawtooth)
  - Practical Byzantine Fault Tolerance – PBFT (Ripple, Hyperledger Fabric etc.)
  - Proof of Authority
  - Proof of Burn
  - Solo
  - Kafka

BRIEF CLASSIFICATION OF BLOCKCHAIN TYPES

	Consensus	#node requirement for majority	Transaction/Block Approval Time
Permissionless/Public (Bitcoin, Ethereum)	PoW	High (Thousands)	Long (5-15 sec per block)
Permissioned/Private (Hyperledger, IOTA, Ripple, R3 Corda, Hashgraph)	PBFT	Low	Short (1 – 5 msec per transaction)
Permissioned/Public (Ethereum after PoS is implemented)	PoS	Moderate	Short (10 – 15 msec per block)

Source/further reading:

IOTA Transactions, Confirmation and Consensus, https://github.com/noneymous/iota-consensus-presentation

Hyperledger White Papers, https://www.hyperledger.org/resources/publications#white-papers

Ethereum, A Next-Generation Smart Contract and Decentralized Application Platform, https://github.com/ethereum/wiki/wiki/White-Paper

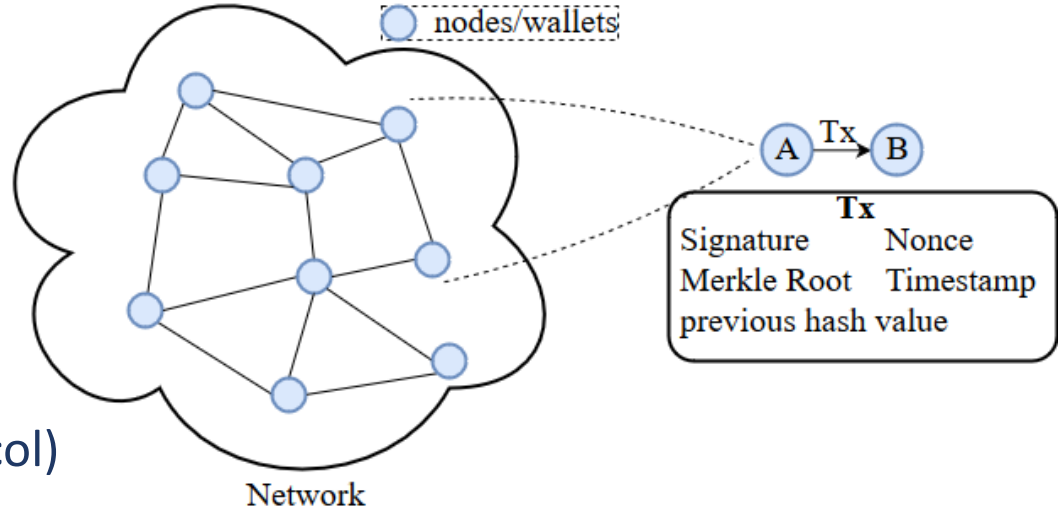
APPROACHES FOLLOWED BY DIFFERENT BLOCKCHAIN/DLT ORGANIZATIONS

Blockchain/DLT	Approach	Organization
Bitcoin Blockchain	Financial Payments using Cryptocurrency	Bitcoin
Ethereum Blockchain	Application development using tools provided by the Blockchain (Remix, Metamask, Web3.js, etc.)	Ethereum
Fabric, Sawtooth, Iroha	Using Development Platforms/APIs (configuring nodes, mobile applications, certifying authorities)	Hyperledger
Corda	Develop Industry Specific Solutions (Finance, Legal, Healthcare, etc.)	R3
Ripple	Cross border payments using blockchain	Ripple
IOTA	Directed Acyclic Graph (DAG) based token for IoT	IOTA
Hashgraph	DAG based asynchronous consensus algorithm with guaranteed Byzantine Fault Tolerance	Swirlds

BUILDING BLOCKS OF BLOCKCHAINS

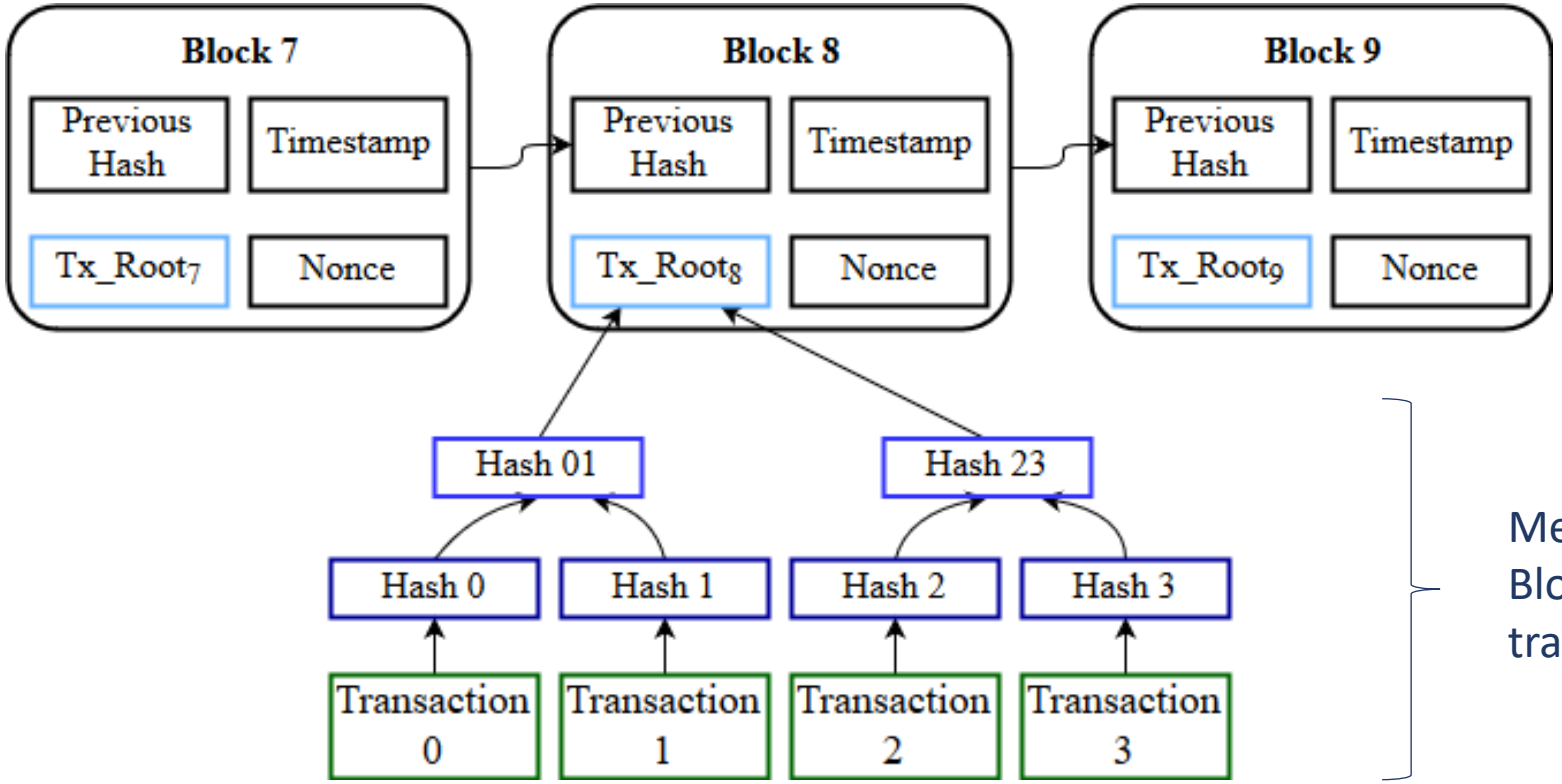
- Distributed database that is
  - practically immutable, maintained by decentralized P2P network
  - using consensus mechanism, cryptography and back referencing blocks
  - to order and validate the transactions

- Primitives
  - Timestamp and Nonce
  - Hash function
  - Merkle Tree
  - Public/Private Key
  - ECDSA
  - Nodes/Peers – Peer to Peer Network (uses gossip protocol)
  - Wallet
  - Smart Contract/Chaincode
  - Consensus Protocol – PoW, PoS, Byzantine Fault Tolerance



THE BITCOIN BLOCKCHAIN

- First blockchain – public with pseudonymous users
- Consensus: Proof of Work



Merkle Tree for Block 8 with 4 transactions

BITCOIN BLOCKCHAIN

- Block creation time: ~10 min
- Average transaction size: 495 bytes
- Maximum block size: 1 MB
- Average number of transactions per block: $\frac{10^6 bytes}{495\ bytes} = 2020$
- Blockchain size: ~197.5 GB
- exponentially increasing blockchain size

BITCOIN MINING

- **Hardware Mining**: to solve Bitcoin blocks
  - started with Field-Programmable Gate Arrays (FPGA)
    - used comparatively less power than CPU and GPU
    - led to the creation of „mining farms“
    - can compute 100 MH/s
  - started with GPU
    - Nvidia GTX 1080 TI can compute around 0.5 GH/s
  - now Application-Specific Integrated Circuits (ASICs) are used
    - capable of computing around 4 – 14 TH/s with a power efficiency of 0.098 – 0.29 W/GH

- **Cloud Mining**
  - to avoid purchasing dedicated mining equipments, creating cooled atmosphere for the hardware, cloud mining became popular
  - Types: hosted, virtual hosted, leased hashing power

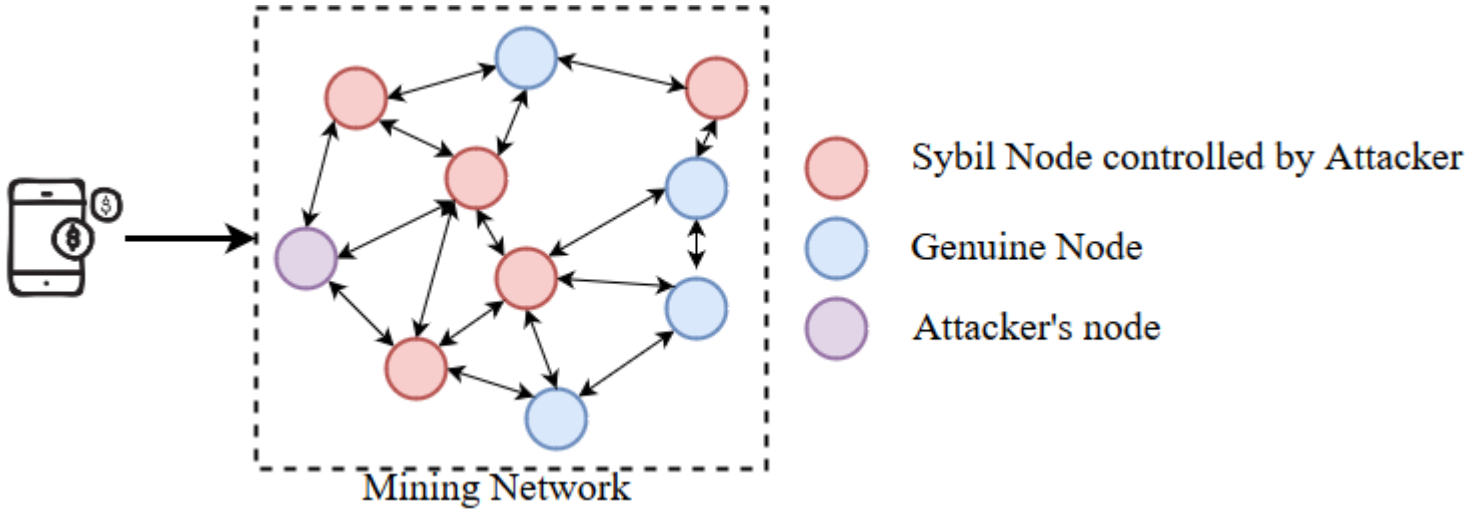
BITCOIN BLOCKCHAIN: APPLICATION DOMAINS

- Financial Domains
- Examples:
  - Microsoft
    - To purchase games, movies and apps in the Windows and Xbox stores
  - Virgin Galactic
    - Book space travels using Bitcoin
  - Expedia
    - For hotel bookings
  - Subway
    - Franchises in Moscow and Allentown, Pennsylvenia accept Bitcoin
  - Wikipedia
    - Accepts donation in the form of Bitcoin
  - MIT Coop Store
    - MIT book store
  - Many more…

SECURITY CONCERNS ABOUT BITCOIN (1/2)

- **Sybil Attack** – Lack of robust identity management
  - Attacker creates multiple identities (maybe virtual) and takes control of the network
  - to forward attackers block faster than the genuine users block

- **DoS/DDoS Attack** – Inherent from the Sybil Attack



SECURITY CONCERNS IN BITCOIN BLOCKCHAIN (2/2)

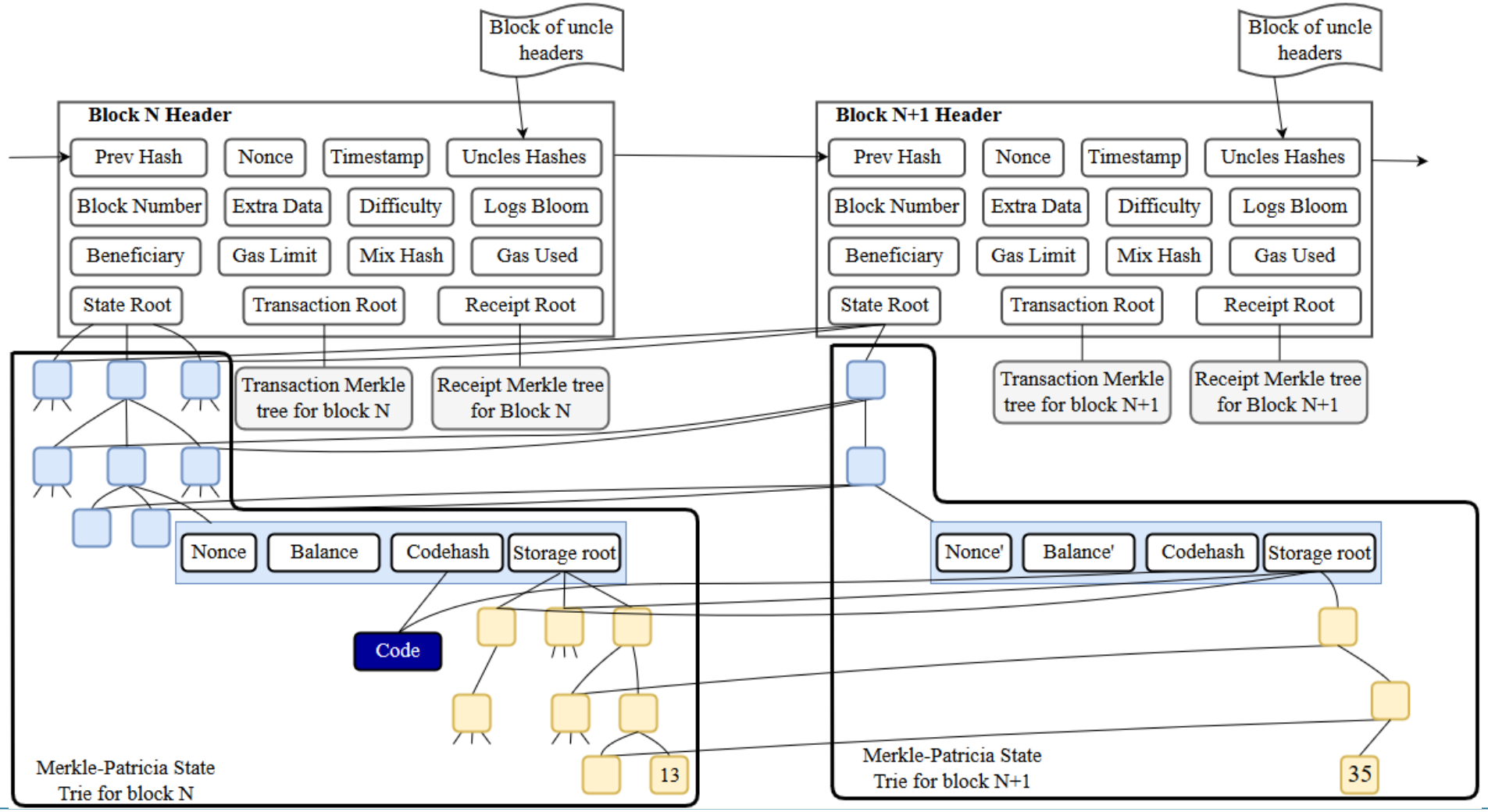
- **Majority Attack** – Bitcoin blockchain assumes an honest majority
  - 80 % of mining pools located in China, 20 % distributed over Iceland, Japan, Czech Republic, India

- **Indentity Theft** – Due to weak password for wallet
  - Stealing of private keys/wallet passwords through phishing attack

ETHEREUM

- Follows the „Blockchain First“ approach
  - Developers build their application by using tools provided by the blockchain (Serpent, Solidity, Web3.js, Truffle, etc.)
- Governed by the Core developers
- Currency: „Ether“, Token system
- Purpose: Run Smart Contracts
- Smart Contracts compiled into „bytecode“ and executed by nodes using Ethereum Virtual Machine (EVM)
- Consensus: Proof of Work (present), Casper (future) – at Ledger level
- Block time: 15.3 sec
- Block size: 25.93 KB
- Blockchain size: 553.1 GB
- Gas limit: 7,999,992 Gas

ETHEREUM BLOCKCHAIN: STRUCTURE



ETHEREUM: GAS AND PAYMENT REQUIREMENT

- Gas
  - mitigated DDoS attacks
  - **Gas**: all programmable computation is subjected to fees in Ethereum
  - **gasLimit**: every transaction has a specific amount of gas associated with it
    - 20,000 gas is used for storing 8 bytes of data and the current cost is 10 $\frac{gwei}{gas}$ (1 ether = $10^{18} wei$)
      - 1 kB data ≅ $ 3
    - retrieving the data is free
- User sending a transaction will pay (gasPrice*gasUsed)
    - Minimum gas for a single transaction = 21,000 gas
- Miners receive the gasPrice amount as payment for mining the transaction
- A higher gas price on a transaction will cost the sender more in terms of Ether and deliver a greater value to the miner

➡ more likely to be selected for inclusion by more miners

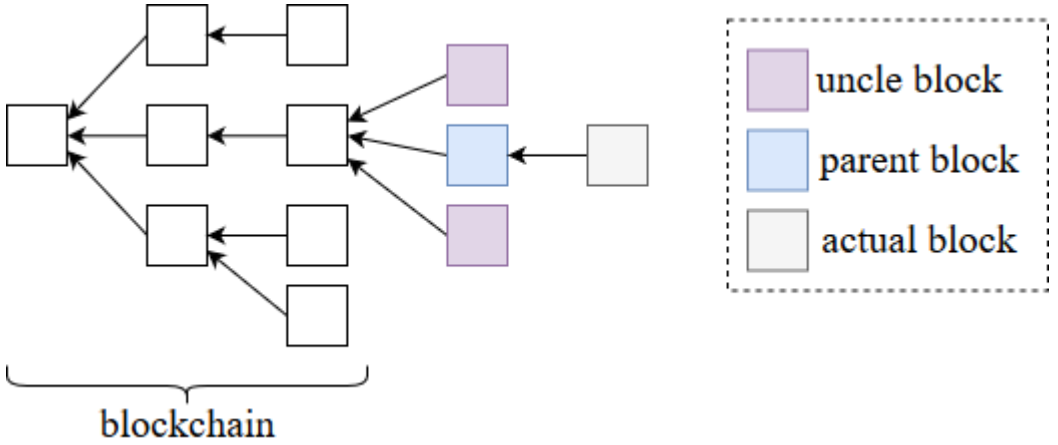
ETHEREUM: GHOST PROTOCOL (1/2)

- GHOST: Greedy Heaviest Observed Subtree
- Deals with:
  - Blockchains with fast block times are bound to leave a large number of stale blocks – blocks which were mined by other nodes but rejected as a longer chain achieved dominance.
  - Miner Centralization: Large mining pool is likely to produce blocks faster than other miners
- GHOST includes stale blocks (as Uncles) – these are included in the calculation of which chain is longest or has the highest cumulative difficulty
- Centralization is solved by giving block rewards to stales of 87.5% - the nephew (child of the uncle block) also receives a reward of 12.5% of the block reward.

Source/further reading:
Yonatan Sompolinsky, Secure High-Rate Transaction Processing in Bitcoin, https://eprint.iacr.org/2013/881.pdf

ETHEREUM: GHOST PROTOCOL (2/2)

- The Ethereum version of Ghost only goes down seven levels – or back seven levels in the height of the block chain.
  - A block must specify its parents and its number of uncles
  - An uncle included in a block must be a direct child of the new block and less than seven blocks below it in terms of height
  - It cannot be the direct ancestor of the block being formed
  - An uncle must have a valid block header
  - An uncle must be different from all other uncles in previous blocks and the block being formed
  - For every uncle included in the block the miner gets an **additional 3.125%** and the **miner of the uncle receives 93.75%** of a standard block reward.



STORING DATA ON BLOCKCHAIN: OFF CHAIN

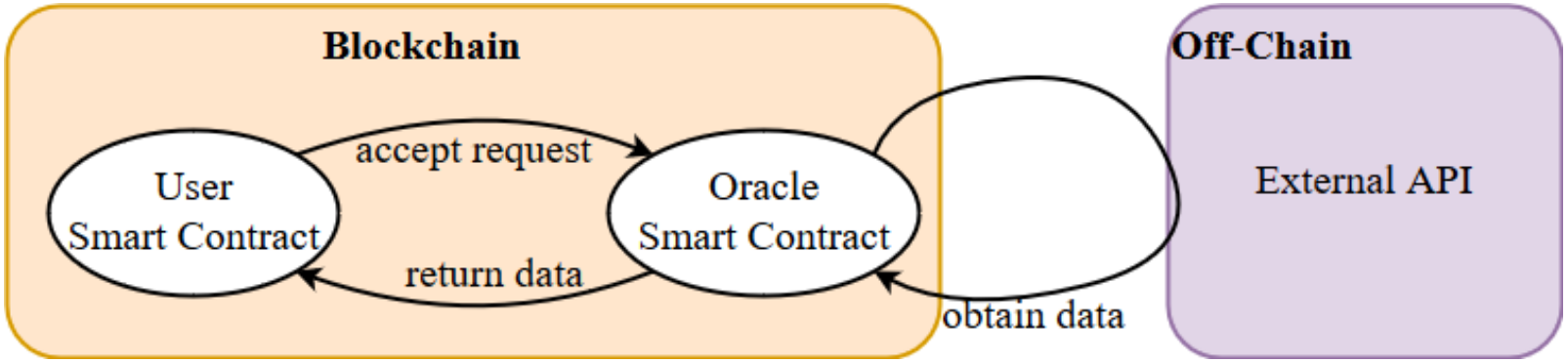
- Decentralized storage: **Inter Planetary File Storage – IPFS** (off chain data storage)
  - Each file is identified by a unique hash – this hash is stored on the blockchain
  - While the data is seved through one or more IPFS nodes
  - Keep your own nodes running to keep the content online
  - Or use the **Filecoin** protocol (Filecoin is not live yet)

- Another option for decentralized storage is **Swarm**

ETHEREUM: ORACLES

- A DLT oracle is a trusted service designed to supply external data to a DLT system.
  - E.g. They can be a trusted data feed that sends information into the Smart Contracts, removing the need for Smart Contracts to directly access information outside their network.
  - Usually supplied by third parties and authorized by companies using them.

- They act as data carrier between Web APIs and the Dapps.
- Companies working to create a decentralized Oracle network: LINK, SmartContract.



ETHEREUM: APPLICATION DOMAINS

- Dezentralized Applications („dApps“) Development
  - Digital signatures
    - Developed by Luxembourg Stock Exchange to ensure authenticity of documents
  - Electric Car charging - RWE (German Energy Provider)
  - Video Games – CryptoKitties
  - Secure Identity Systems – uPort
  - Decentralized Marketplaces

HYPERLEDGER

- Open source collaborative effort to develop decentralized applications in Finance, IoT, Healthcare, Supply Chain industries
- Governing board headed by Linux Foundation
- 10 academia partners, 25+ industry partners
- 5 different Business Blockchain Frameworks Hosted
  - Burrow, Fabric, Iroha, Sawtooth, Indy

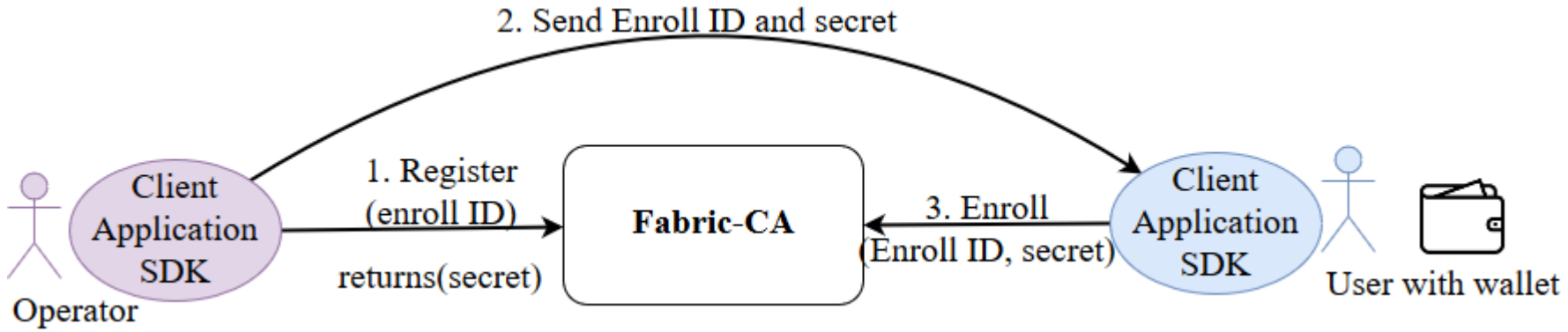
Source/further reading:
Hyperledger, https://www.hyperledger.org/projects

HYPERLEDGER FABRIC

- Implementation of Blockchain technology intended as a foundation for developing blockchain applications
- Designed by IBM
- Permissioned, private
- „Chaincodes“ in Golang or Java
- No native cryptocurrency, can be implemented through chaincode
- Uses PKI, each actor {peers, orderers, client application, administrators} has an identity encapsulated in an X.509 digital certificate.

- **Fabric Certifying Authority – Fabric-CA** serves as a root CA
  - Not capable of issuing SSL certificates
  - A public/commercial root CA can be used instead
- 10,000 transactions per second using BFT consensus at transaction level

FABRIC-CA: NEW USER REGISTRATION AND ENROLLMENT

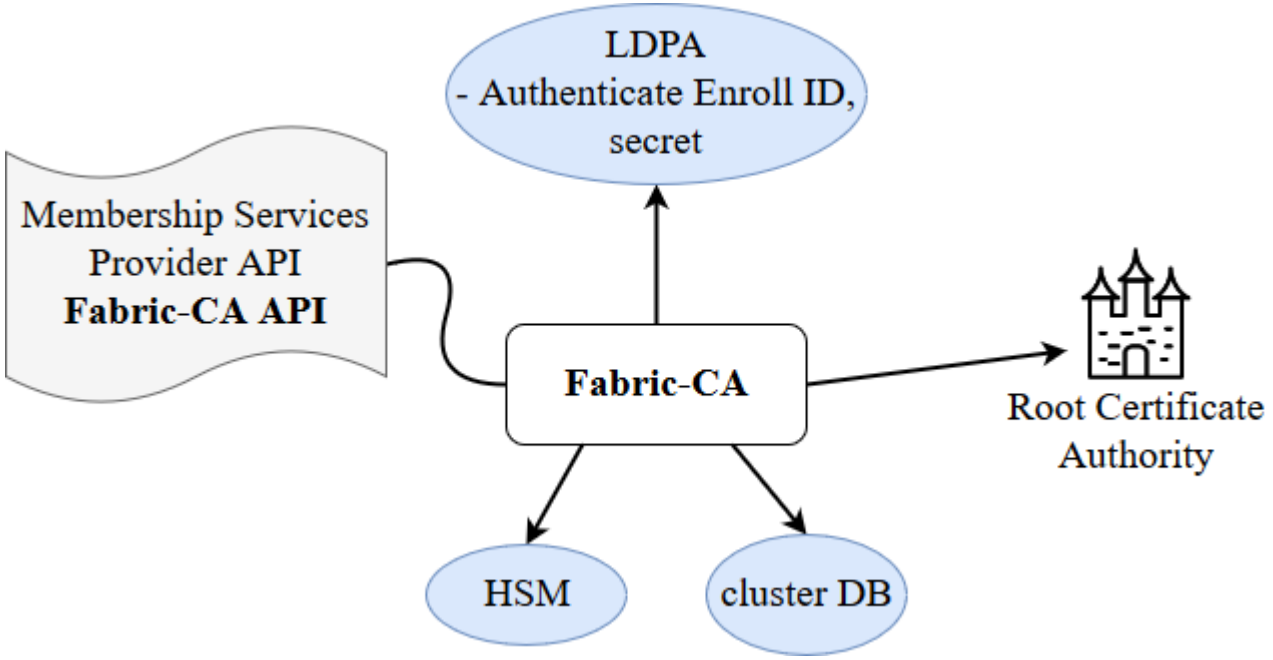


Registration and Enrollment

- Admin registers new user with Enroll ID
- User enrolls and receives credentials
- Also available are additional offline registration and enrollment options

FABRIC-CA DETAILS

- Default implementation of the Membership Services Provider Interface
- Supports clustering fo HA characteristics
- Supports LDAP for user authentication
- Supports HSM



FABRIC COMPONENTS

- Validating Peers
  - Transaction Ledger
  - World State
- Pluggable Consensus
  - Practical Byzantine
  - Fault Tolerance
  - None
- Smart Contract
  - Go, Java
  - Key-Value storage
- Membership services
  - Credentials and certifications
  - Users and Peers

TYPES OF PEERS

- PEER
  - Commits transactions – maintains Ledger and World State
- ENDORSING PEER
  - Endorses and executes Chaincode
- ORDERING PEER
  - Includes transactions in blocks
  - Communicates with other peers

FABRIC ARCHITECTURE WORKFLOW (1/2)

1. **Propose:** Client app submits transaction proposal for smart contract to the Endorsing Peer E_0

2. **Execute:** Endorsing Peer executes the transaction and (optionally) „anchors it“ w.r.t the ledger version numbers
   - An „anchor“ contains all data read and written by the contract that is to be confirmed by other endorsers.

3. **Submit:** Client requests further endorsement from other Endorsers (E_1, E_2, …) as per the Endorsement Policy and (may) decide an anchor obtained from any endorsers

4. **Endorse:** Endorsing Peers sign the result and send the Endorsement to the Client

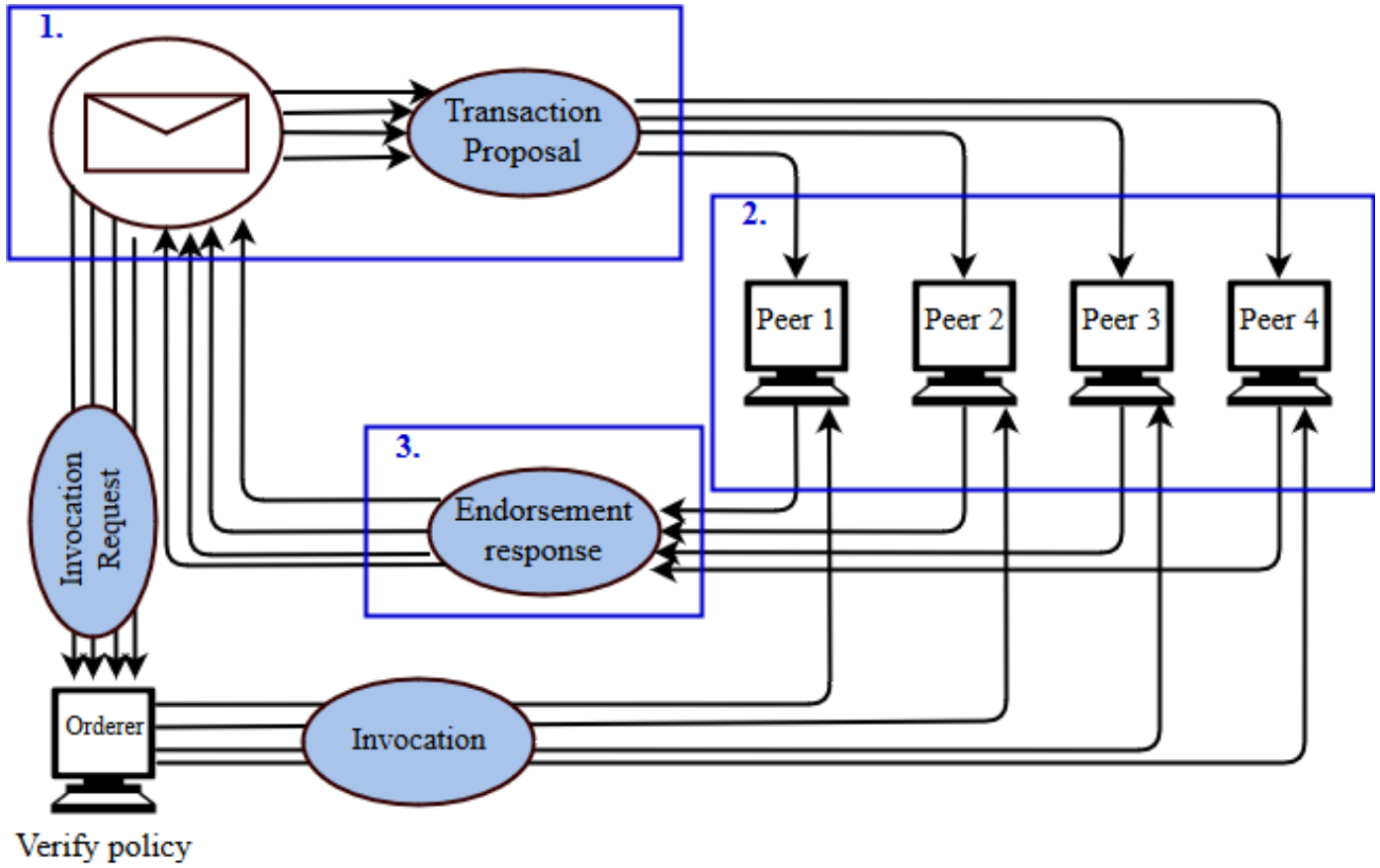
FABRIC ARCHITECTURE WORKFLOW (2/2)

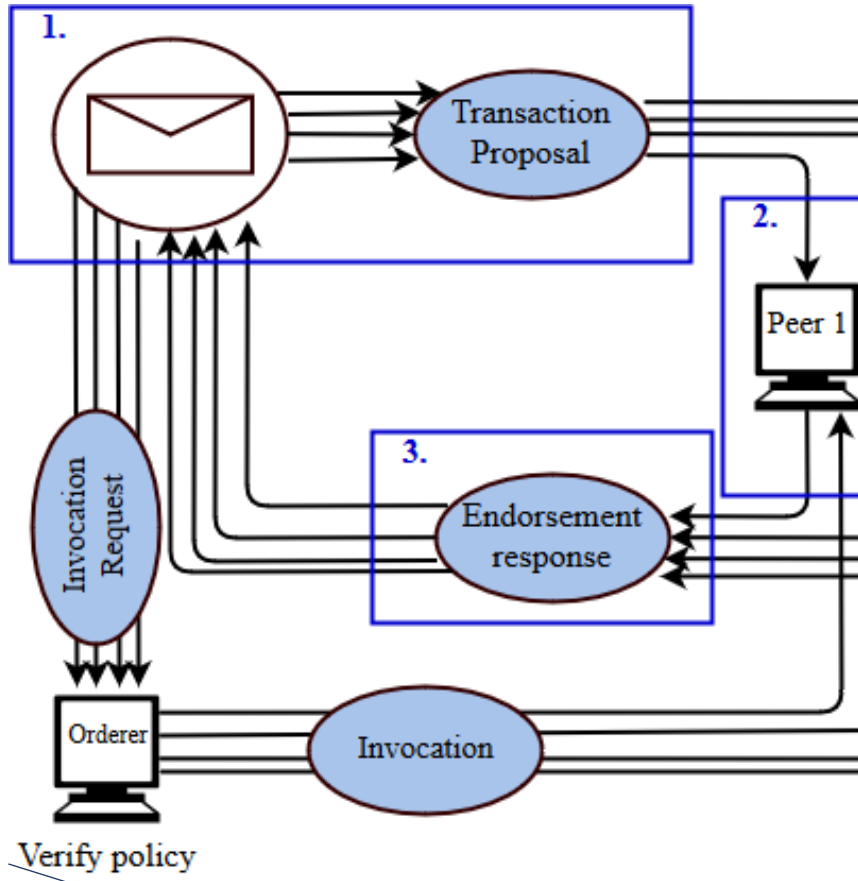
5. **Order:** Client formats transaction and sends it to the Ordering-Service Nodes for inclusion in the ledger

6. **Deliver:** Ordering- Service delivers the next block in the ledger with the endorsed transaction

7. **Validate:** The Peers validate the block received from the Ordering Service and update the Ledger and the World State.

HYPERLEDGER FABRIC: ORGANISATION





Hyperledger Fabric

FABRIC: APPLICATION DOMAIN

- Developing enterprise grade transactions based applications
  - B2B contracts
    - Business contracts can be codified to allow two or more parties to automate contractual agreements in a trusted way.
  - Manufacturing Supply Chain Management
  - Cross Border Payments – ANZ, BNP Paribas, BNY Mellon and Wells Fargo
  - Seafood traceability – Intel and Hyperledger
  - Border and Immigration Control

SECURITY PROBLEMS IN FABRIC (1/3)

- **Problem:** Installation of Malware – Remote Access Trojan (RAT)
  - Chaincode runs on Docker container
  - Chaincode has access to networking – can very easily download and install further software packages (including security tools) and can run for long periods of time
  - Installation of RAT will act as a base from which a threat actor could undertake a more comprehensive attack.
    - A threat actor could create a new ledger with associated malicious chaincode, and persuade others to participate
    - A threat actor could infiltrate an organization responsible for developing and maintaining the chaincode for an existing ledger, then publish an update

Source/further reading:
Graham Shaw, Nettitude

SECURITY PROBLEMS IN FABRIC (2/3)

- **Problem:** Log Injection
  - Unvalidated inputs are written verbatim to a log
  - Indirect threat to the business model
  - Can be used to fabricate log entries to mislead incident response efforts, or corrupt the log to prevent it from being processed by automated monitoring systems.

SECURITY PROBLEMS IN FABRIC (3/3)

- **Problem:** Code Injection
  - One function was found to be vulnerable to Code injection
  - Subcomponents of the system (functions and data structures) do not have detailed interface specifications which can allow one to determine:
    - whether a function body correctly implements the required behaviour and
    - whether calls to that function elsewhere in the program are using it correctly and appropriately.

HYPERLEDGER SAWTOOTH

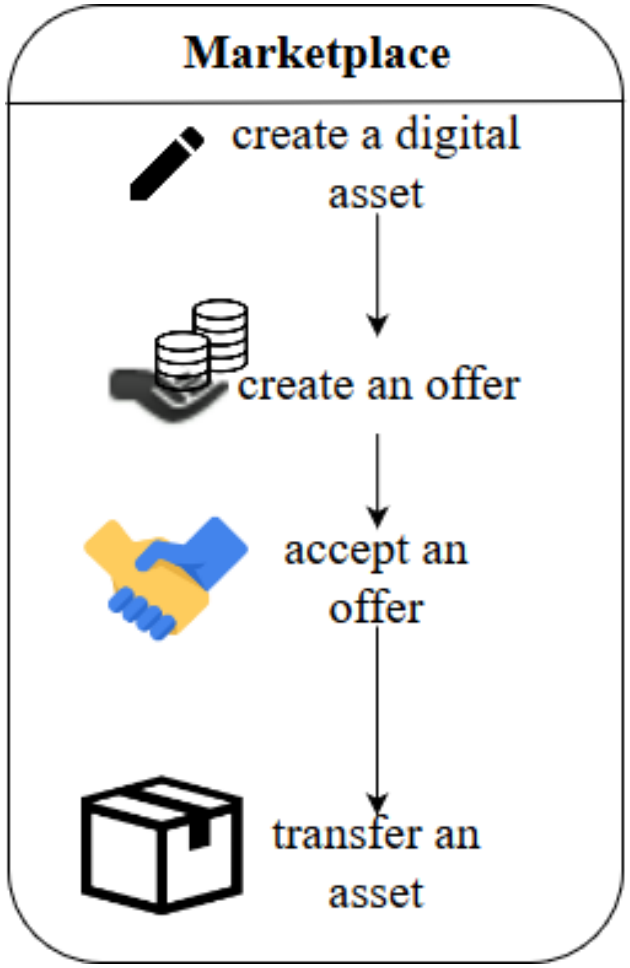
- Modular Platform for implementing transaction-based updates to data between untrusted parties
    - **Transaction families:** Fix transaction semantics to limit risks
    - e.g. integer key family: only 3 operations (increment, decrement, set) allowed. No looping constructs available

    ➡ hard to have intentional or accidental transaction script problems

- Proof of Elapsed Time consensus algorithm
- Currently, PoET‘s implementation relies on a Trusted Execution Environment (TEE) e.g. Intel‘s Software Guard Extensions (SGX) which introduces a need for trusted third party
- SDK available for Python, Go, Javascript, Java and C++

SAWTOOTH: APPLICATION DOMAIN

- Applications for storing digital assets without central authority
  - Digital Asset Exchange – Marketplace
  - Bound Asset Settlement
- Supply Chain Traceability
- Music Content Rights Registry



HYPERLEDGER IROHA

- Designed to be simple and easy to incorporate into infrastructural projects requiring distributed ledger technology
- Initially contributed by Hitachi, Soramitsu, Colu, NTT Data
- Consensus algorithm: **Sumeragi**
  - A submitted transaction needs 2f+1 signatures to confirm it; f = number of Byzantine faulty nodes the blockchain‘s network can handle
  - Order of nodes performing the validation of transaction si determined by: **Hijiri**
  - Hijari calculates reliability of each server based on
    - Time they were registered with membership services
    - Number of successful transactions processed by them
    - If any failures were detected on server

- Block acceptance time will be less than 2s (as claim by Iroha developers)

IROHA: APPLICATION DOMAIN

- Emphasis on mobile application development for DLT apps

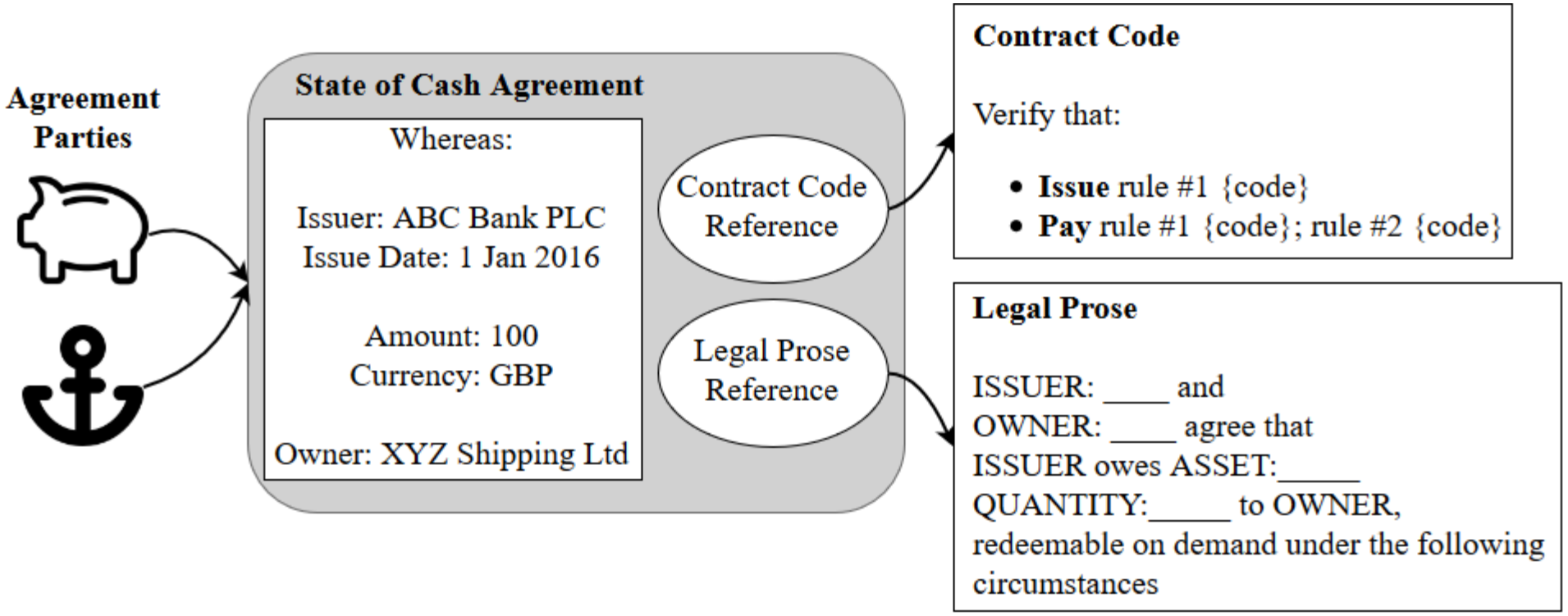
- Digital Identity Management
- Logistics
- Money transfer and settlement

R3 CORDA

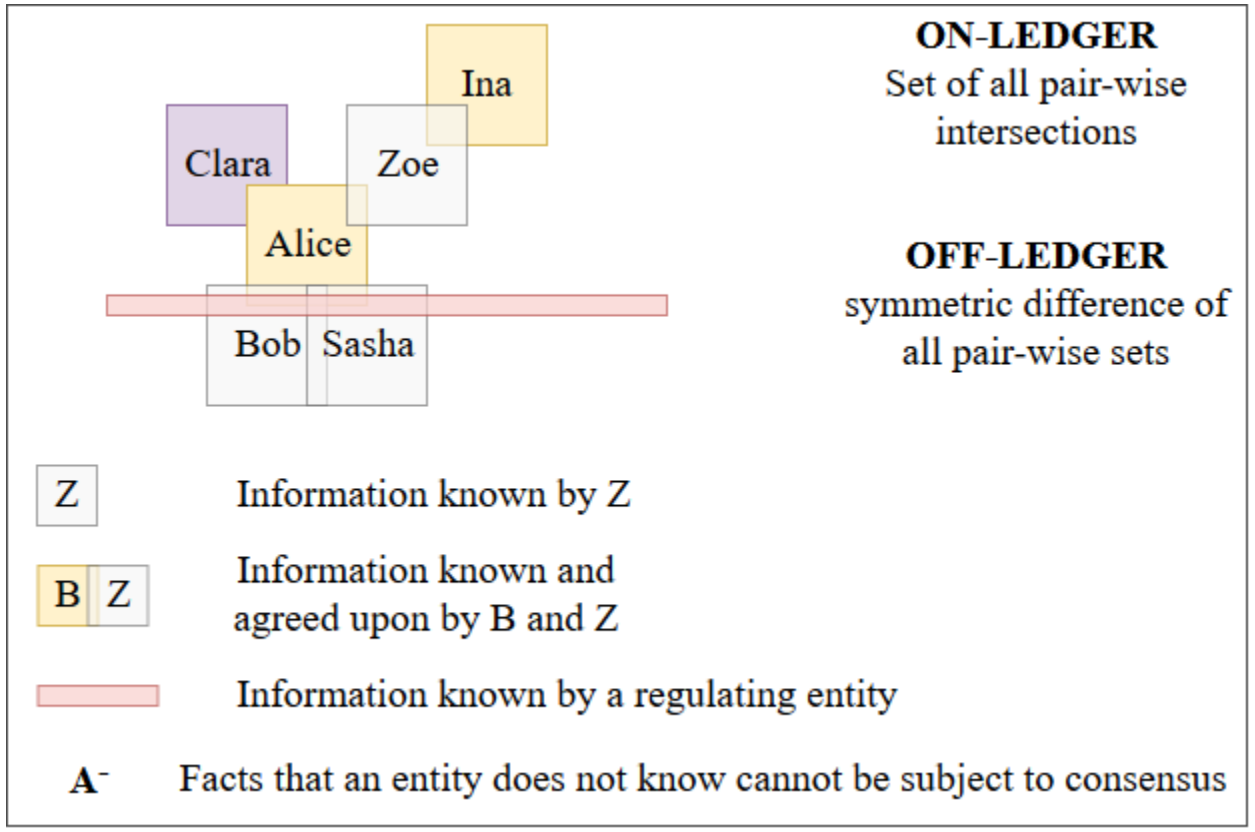
- Aims to offer universal interoperability of public networks with the privacy of private networks
- Blockchain platform for the finanacial services industry
- Permissioned (blockchain), private (data sharing)
- Pluggable consensus: ‚Notary Clusters' – at transaction level
  - A **notary** is a network service that provides uniqueness consensus by attesting that, for a given transaction, it has not already signed other transactions that consumes any of the proposed transactions input states.
  - A notary can either sign the transaction or reject and flag the transaction as a double-spent attempt.

- Smart Contracts in Kotlin, Java
  - Smart contract links business logic and business data to associated legal prose to ensure that financial agreements on the platform are rooted firmly in the law and can be enforced in case of ambiguity, uncertainty or dispute.

CORDA: STATE WITH HASH OF LEGAL PROSE



CORDA CONSENSUS



CORDA CONSENSUS

- **State objects** – represents agreement between more than 2 parties, governed by machine-readable contract code
- States created by transactions
- **Transaction protocol** – enabling parties to coordinate actions without a central controller
- Data is only shared with the parties required to see it
- Any actor sees only a subset of the overall data managed by the system as a whole
- Data is „ON-LEDGER“ if at least two actors on the system are in consensus as to ist existence and details
  - Allowing arbitrary combinations of actors to participate in the consensus process for any given piece of data
- Data held by only one actor is „OFF-LEDGER“

CORDA: APPLICATION DOMAIN

- Legal, Financial, Healthcare via Corda decentralized application (CorDapp)
- Letter of Credig – HSBC, Bangkok Bank, BBVA, BNP Paribas, ING, Intesa Sanpaolo, Mizuho, RBS, Scotiabank, SEB and U.S. Bank
- KYC
- Financial Contract

SECURITY PROBLEMS IN R3 CORDA

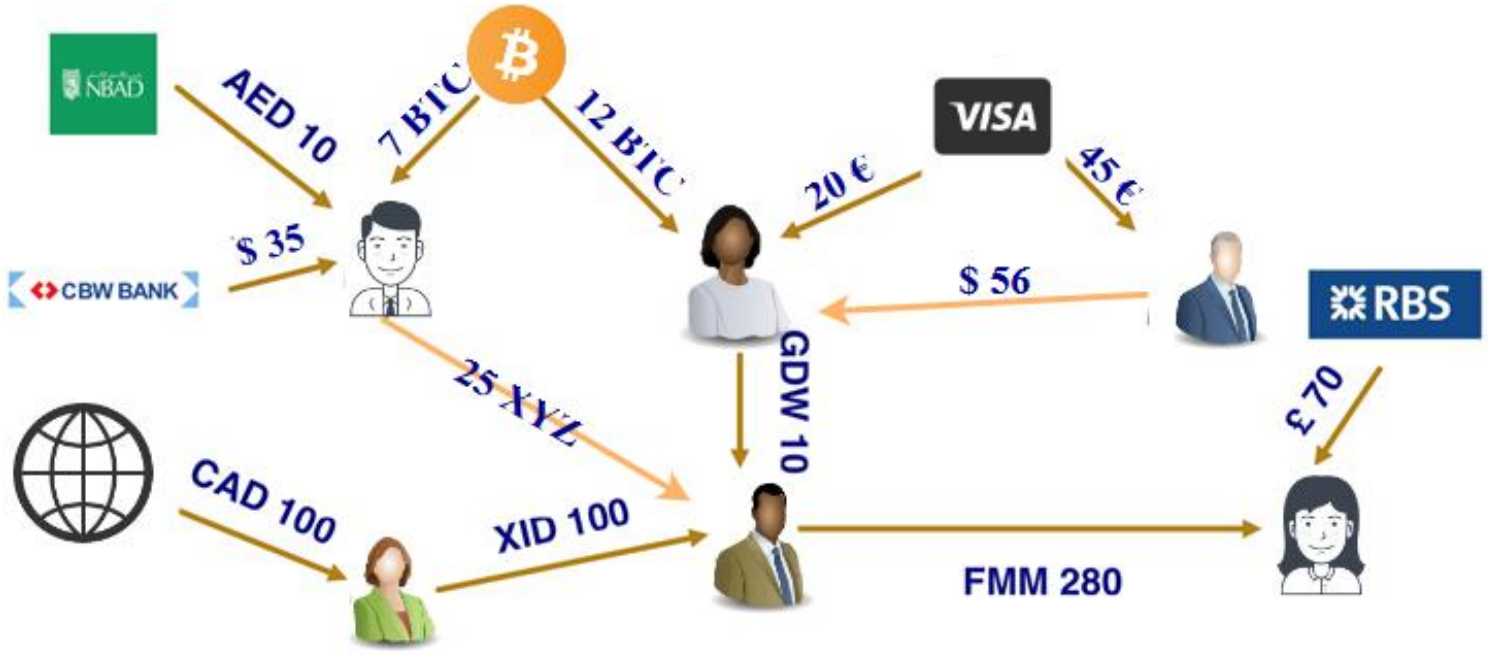
- **Problems:**
  - According to the European Economic Area (EEA) Reпord, CORDA was vulnerable to:
    (dated 15th April, 2016)

    - Cross site scripting (XSS)
    - Sensitive Data Exposure – not using TLS
    - Missing Function Level Access Control – server side code validation was not implemented
    - Cross Site Request Forgery (CSRF)

RIPPLE FRAMEWORK

- Launched in 2012 with a goal to expedite settlement on international transactions by using Blockchain technology
- IOUs on the ledger
- Capable of handling 1500 transactions per second
- Payment settled in 4 seconds
- Uses XRP crypto coin for payment/forex
  - Convert value of transfer in XRP is removing exchange fees and reducing time when USD is used as a central currency
- Path finding finds cheapest conversion cost for the user

RIPPLE: PAYMENT SETTLEMENT



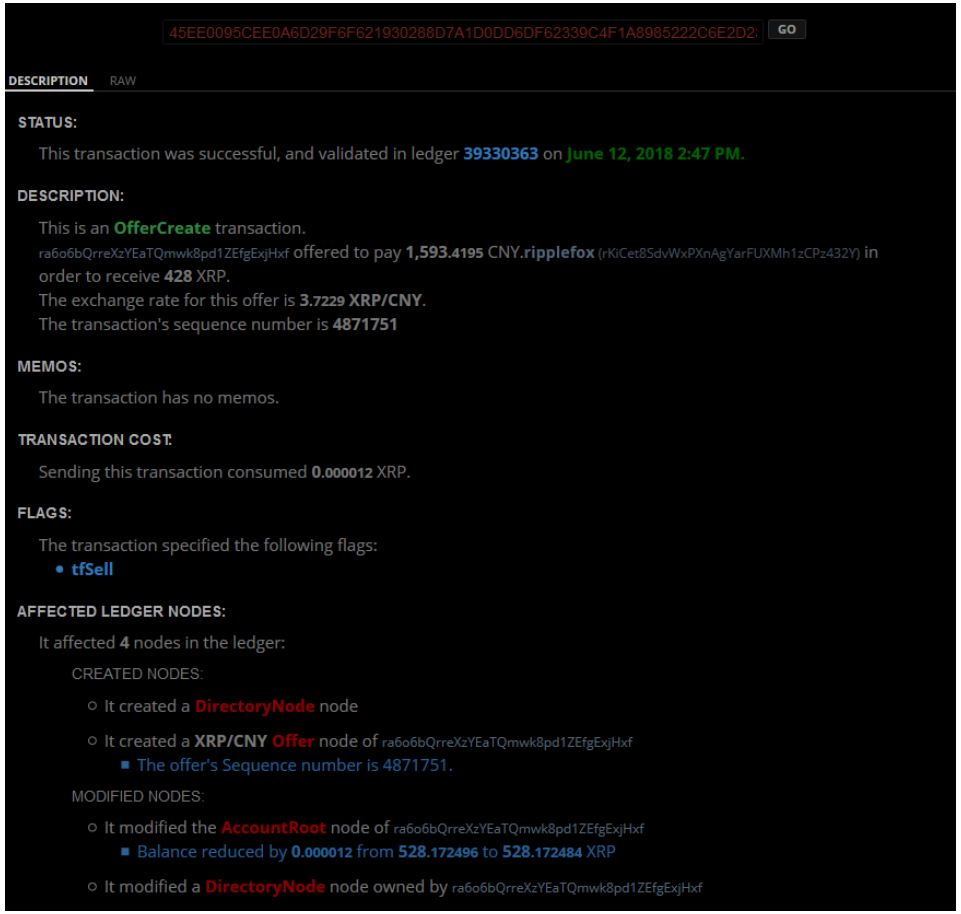
	Transaction Time	Integrity	Worldwide, cross-currency transactions
Bank	~ 1 day	Bank only	High fees
Ripple	~ 5 seconds	Public verification	Tiny fees

RIPPLE: APPLICATION DOMAIN

- Remittance, Currency Exchange, Real time gross settlement (RTGS)
- Partnerships with – Accenture, American Express, Axis Bank, Royal Bank of Canada, Yes Bank, Cambridge Global Payments …
- … and more for cross-border payments for …
- … retail customers, corporations, and other banks

ATTACKS ON PRIVACY OF RIPPLE LINKS AND TRANSACTIONS

- It is possible to link multiple transactions and identities to the same user



- Ripple provides pseudonymity to its users by employing public key hashes as identity.

SECURITY PROBLEMS IN RIPPLE (1/2)

- **Problem:** Faulty Payment Gateway

- **Gateway:** A gateway is any person or organization that enables users to put money into and take money out of Ripple‘s liquidity pool
- Gateway wallets:
  - Included in the core of the Ripple network
  - Significantly contribute to the liquidity of the network
- A faulty gateway can disable rippling on most credit links of its wallet, ensuring that transactions routed through it are no longer possible and effectively freezing the balance held at credit of its wallet.
- This affects
  - Liquidity of the network
  - Lead to monetary losses to the neighboring wallets

SECURITY PROBLEMS IN RIPPLE (2/2)

- **Problem:**
  - Ripple Labs owns 60% of all XRP in circulation (60 billion out of total 100 billion)

- This did not follow the goal of making XRP a decentralized peer-to-peer currency

- To answer this and to create Supply Predictability, Ripple placed 55 billion XRP in a cryptographically secure wallet
  - 55 escrow contracts of 1 billion each were created
  - In the beginning of a month a contract expires and 1 billion XRP is made available for Ripple‘s use.
  - Unused XRP at the end of the month are returned back to the escrow.

IOTA – DIRECTED ACYCLIC GRAPH (DAG) BASED DLT

- Created to enable fee-less microtransactions for the IoT
- Efficient, Scalable and Lightweight
- Replaces blockchain with a directed acyclic graph (DAG) known as **Tangle**
- Each user can be viewed as an independent ‚miner‘
  - If you make a transaction, you validate two old transactions in return
- Resistant to quantum computations
  - One must check 3^8 nonces to find a suitable hash to generate a block. So the gain for a quantum computer is comparatively less than in case of Bitcoin blockchain
  - Time needed to find a nonce is not much larger than the time needed for other tasks to issue a transaction

Source/further reading: IOTA, IOTA Whitepaper, http://iotatoken.com/IOTA_Whitepaper.pdf

IOTA: RELATIONSHIP BETWEEN TRANSACTIONS

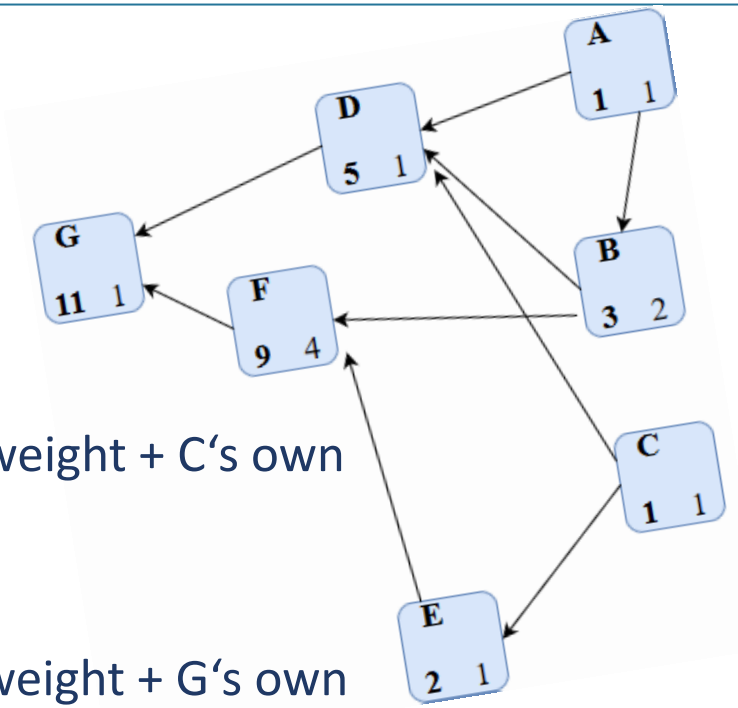
- There are two ways that transaction A can be approved by transaction B. One way is that transaction A is **directly** approved by transaction B.

Direct approval of transaction

- Second way allows some transaction (e.g. W, Z) to be placed in between, linking up the indirect relationship of A and B as followed:

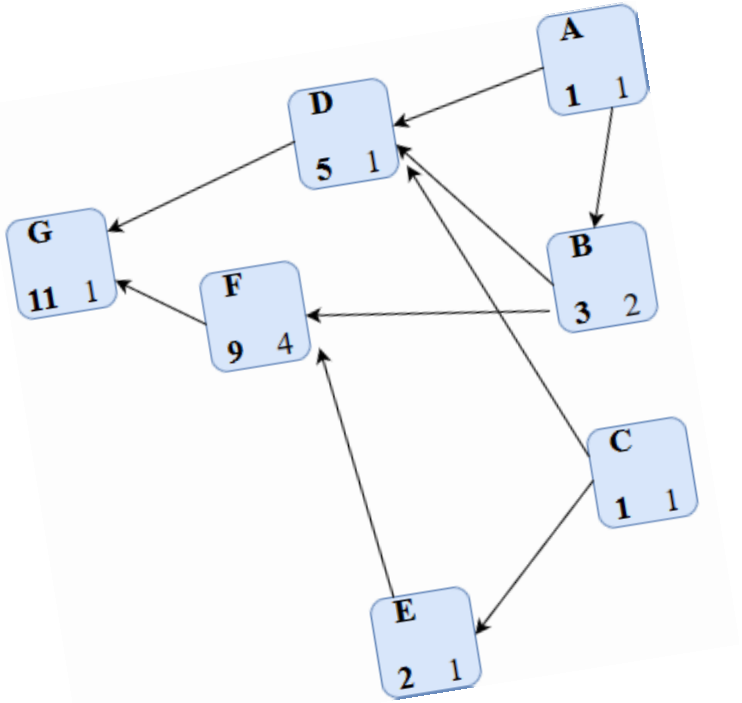
Indirect approval of transaction

IOTA: KEY TERMS

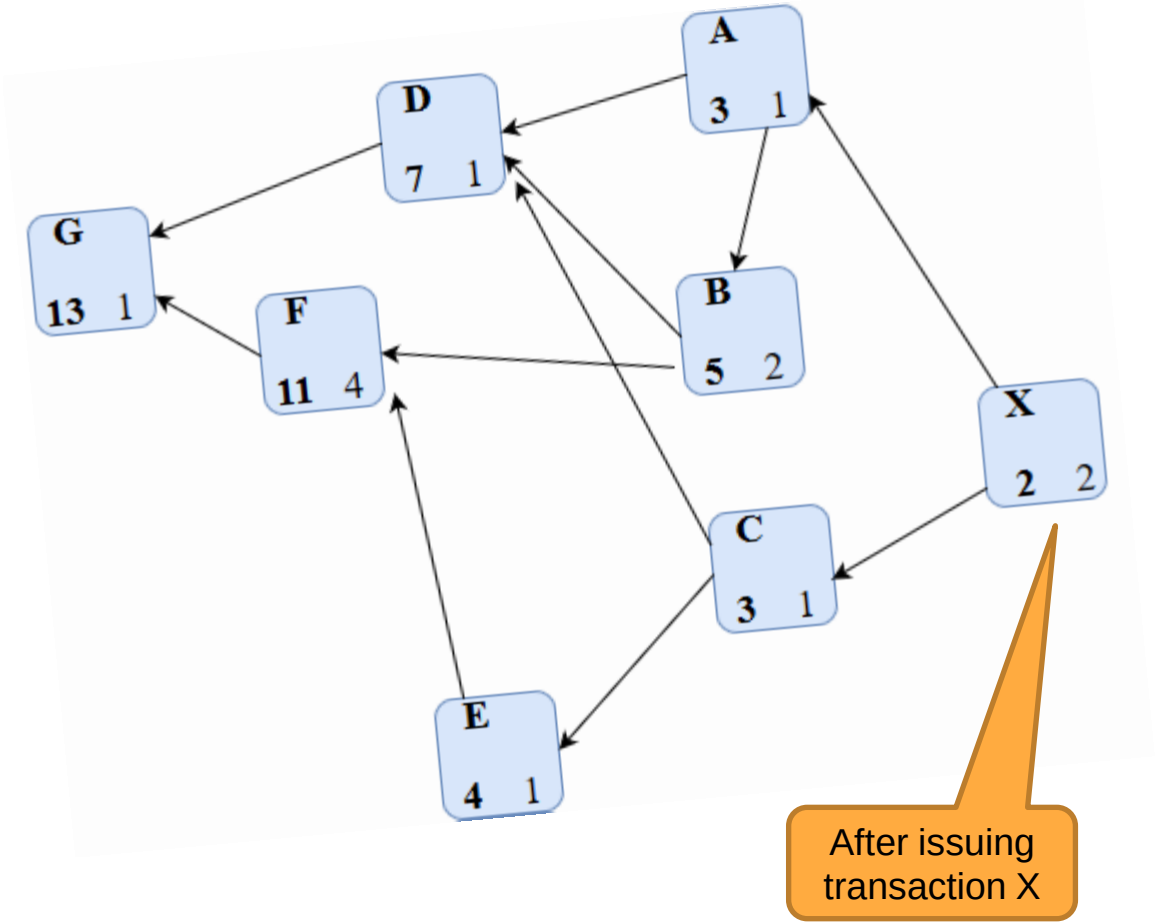


- **Tip** is a newly issued transaction that has not received any approval
- **Weight** of a transaction
- **Cumulative weight**
  - $Weight_D$= 1; $cumWeight_D$= 5 (D‘s own weight + A‘s own weight + B‘s own weight + C‘s own weight = 1+1+2+1)
- **Score**
  - $Score_A$ = 9 (A‘s own weight + B‘s own weight + D‘s own weight + F‘s own weight + G‘s own weight = 1+2+1+4+1)
- **Height:** The length of the longest oriented path to the genesis node
- **Depths:** The length of the longest reverse-oriented path to certain tips.
  - For instance, the height of D is 3 (D -> F-> G -> genesis) and the depth of D is 2 (D <- B <- A)

IOTA: WEIGHT ASSIGNMENT POST A NEWLY ISSUED TRANSACTION



Before issuing a new transaction X



After issuing transaction X

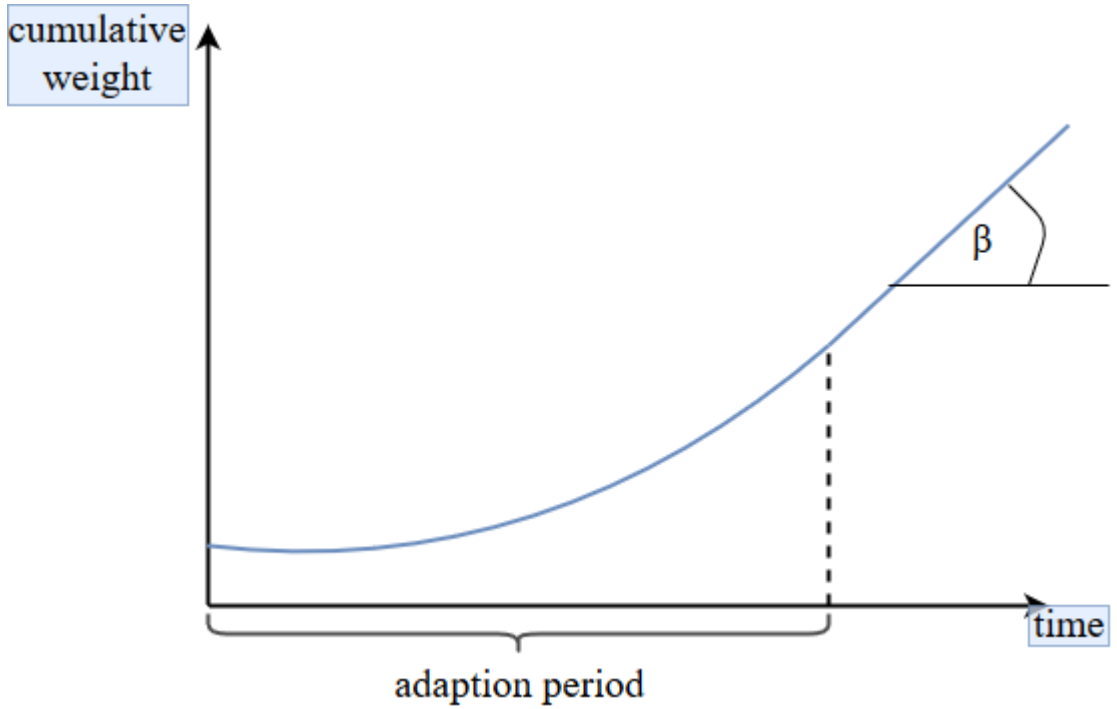
Assumption: All transactions have own weight = 1

In each node:

- Bold numbers = cumulative weights
- Small number = own weights

IOTA: HOW FAST DOES THE TANGLE GROW

- Low load: Grow at constant speed
- High load: Grow with increasing speed during adaptation period, and return to a constant speed after the period. For the reason that all new transactions will be guaranteed to indirectly approve this transaction.
  - Cumulative weight will grow with speed λw, where w is the mean weight of a transaction



IOTA: APPLICATION DOMAIN

- **IoT**
  - Indentity of Things (IDoT)
    - Giving a tamper-proof „identity information tag“ to IoT devices
  - MoU between IOTA and the city of Taipei
    - Palm-sized sensors to detect changes in temperature, light and pollution
    - Creates a comprehensive environmental network across the city
  - IOTA nad Kontrol (Canada based Energy Corporation)
    - Plan to provide local energy producers and consumers with a peer to peer energy market

SECURITY PROBLEMS IN IOTA (1/3)

- **Problem:** Hash function ‚Curl‘ has produced collisions
  - Collision in hash function was found using differential cryptoanalysis
  - Here, two different payments in IOTA (bundles) with same hash value are produced
  - Thus have the same signature.
  - In such attack, a bad actor can destroy users‘ funds, or possibly, get user funds.
- Fixed on Patch issued on August 7, 2017

SECURITY PROBLEMS IN IOTA (2/3)

- **Problem:** Replay Attack
  - IOTA utilizes one time signatures, combined with low confirmation rates of transactions – „replayBundle“ feature.
  - Reattaching is required to get a transaction through
  - Bundles can only be safely signed a single time.
  - Thus, a user is allowed to reattach any bundle of transactions without any proof of ownership.
  - The expected behaviour – only one use of the same bundle hash should be allowed inside a consistend transaction history (subtangle).
- **Problem** – The replays of a previously confirmed bundle will not get confirmed again. The coordinator will repeatedly approve the same bundle hash
  - This means that while a user has signed a transactions to send 500 Miota it can be attached to the network 10 times draining the account of 5000 Miota.

SECURITY PROBLEMS IN IOTA (3/3)

- **Problem:** Phishing Attack

  - In August 2017, the hacker registered the domain iotaseed.io and advertised it as an IOTA seed online generator.
  - He linked the iotaseed.io website to a GitHub repository
  - He ran mostly the same code from the GitHub repository but made modifications to the Nitifier.js library
  - This code always used a fixed seed „47825888755128036 42“ plus a counter variable that increases by one every time seedrandom is run
  - IOTA users visiting the iotaseed.io website received predictable seeds
  - On January 19, the hacker utilized the logs to access IOTA accounts with the seeds (private keys) he collected and started transferring funds out of owner‘s wallets which amounted to $4 million

HASHGRAPH – DAG BASED DLT

- Gossip of Gossip protocol
- Fast: 250,000+ transactions per second
- Asynchronous Byzantine Fault Tolerance -> supports ordered transactions
- Lightweight
- Permissioned, Private network
- Virtual Voting paradigm

LIMITATION OF HASHGRAPH

- Hashgraph chieves 250,000+ transactions per second in private/permissioned based networks
- However, in public network, it is difficult to achieve this since all the participating nodes in consensus protocol may not be trusted

SUMMARY (1/4)

Type	Native Currency/ Protocol Token	Permission to add blocks	Data access restrictions	Coding Language	Consensus mechanism	Transaction throughput	Suitable Applications	Security Limitations
Bitcoin	Bitcoin (BTC)	Permissionless	Public	Stack based with few instructions SCRIPT	Proof of Work	2-4	Crypto-currency	Majority Attack Sybil Attack DoS/DDoS Identity Theft
Ethereum	Ether (ETH)	Permissionless	Public	Viper, Solidity, Serpent, LLL	Proof of Work (current), Proof of Stake (future)	10-30	Decentralized application development	Immutable Bugs Keeping Secrete Exception disorder etc.
Hyperledger Fabric	None - Currency and tokens via chaincode	Permissioned	Private	Golang, Java	Kafka (Permissioned voting)	3500	Developing enterprise grade transactions based applications	Log injection Code injection Remote Access Trojan

SUMMARY (2/4)

Type	Native Currency/ Protocol Token	Permission to add blocks	Data access restrictions	Coding Language	Consensus mechanism	Transaction throughput	Suitable Applications	Security Limitations
Hyperledger Sawtooth	"MarketPlace" to issue and exchange quantities of customized digital "Assets"	Permissioned & Permissionless	Public or Private	Python, Go, Javascript, C++	Proof of Elapsed Time	NA	Applications for storing digital assets without central authority	NA
Hyperledger Iroha	None	Permissioned	Private	C++	Sumeragi (BFT)	NA	Emphasis on mobile application development for DLT apps	NA

SUMMARY (3/4)

Type	Native Currency/ Protocol Token	Permission to add blocks	Data access restrictions	Coding Language	Consensus mechanism	Transaction throughput	Suitable Applications	Security Limitations
R3 Corda	None	Permissioned	Private	Kotlin, Java	Pluggable	1678 (issuance of new states to the ledger) 170 (complete transaction)	Financial applications, Healthcare, Legal	XSS Sensitive Data Exposure Cross site request forgery
Ripple	XRP	Permissioned	Private	Javascript	PBFT	1500 (can scale upto 50,000)	Cross border payments, payment gateway	Faulty Payment Gateway Privacy Problem

SUMMURY (4/4)

Type	Native Currency /Protocol Token	Permission to add blocks	Data access restrictions	Coding Language	Consensus mechanism	Transaction throughput	Suitable Applications	Security Limitations
IOTA	IOTA	Permissionless	Private	Javascript, Python, Go, C#	Proof of Work	500-800	IoT, Payment Gateway	Replay Attack Phishing Attack
Hashgraph	None	Permissioned	Private	Java, Scala	PBFT	2,50,000	Distrubted applications	Unable to run in public network