LOG8430E: Blockchain

©M. Fokaefs et M. Rasolroveicy

What is Blockchain?

- Blockchain acts as an immutable (permanent and unalterable), consensus-driven (trust verification), decentralized (networked copies) and transparent public record of data secured using a P2P (peer-to-peer) network on multi-clouds.

- Common use cases of Blockchain:
  - Healthcare and patients data storage
  - Tracking the transactions that will include an exchange of value between parties of a network.
  - Supply Chain and Food-traceability
  - Finance and Banking
  - Internet of Things

- First use case of Blockchain:
  - Bitcoin

Definition:

- In a decentralized network, participants can agree on the state of the system without the need to either know or trust each other.

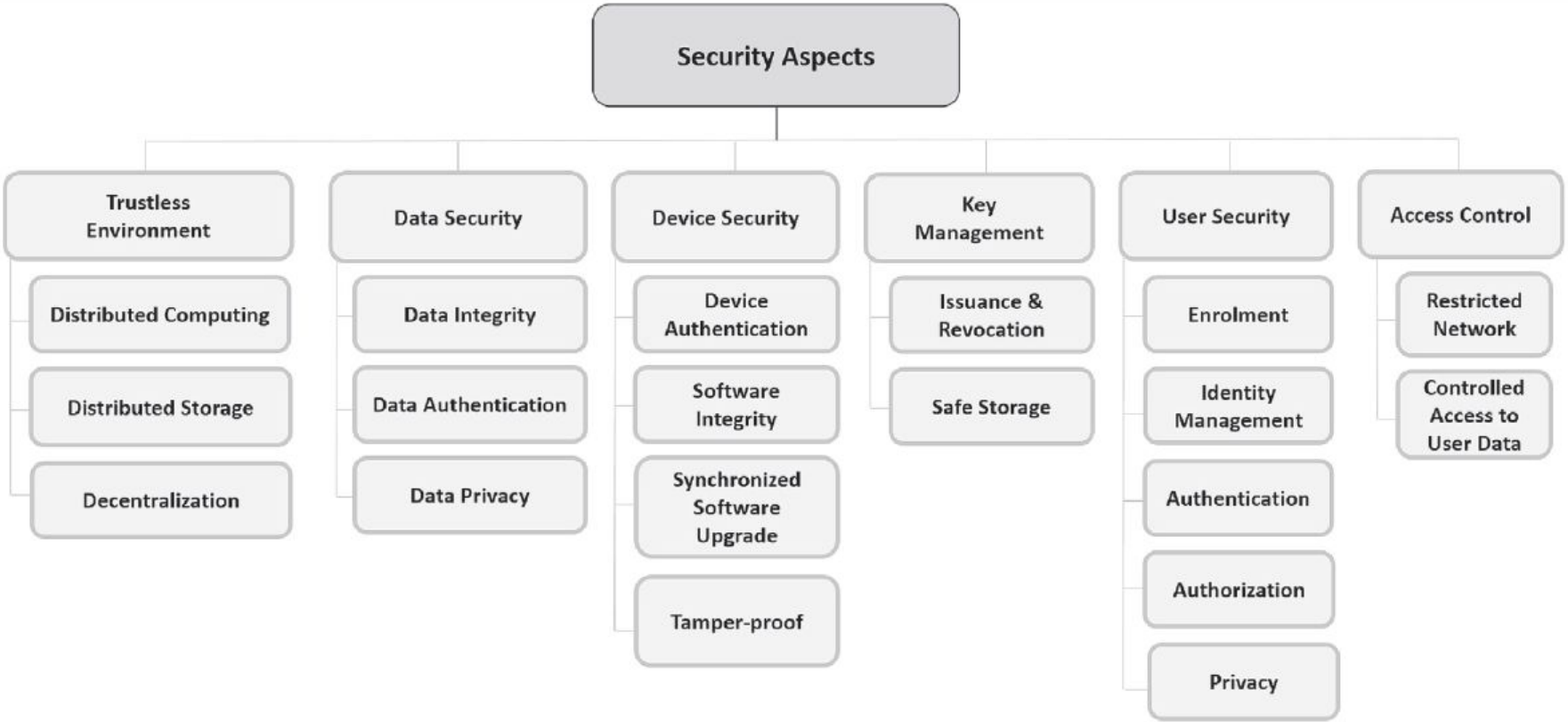
- In Blockchain, data can not be altered and we can only update the state of the system and it will keep track of all the transaction that have taken place in an immutable distributed digital ledger.

The Benefits of Using Blockchain:

- Save time for inter-bank transfer
- Guarantee Confidentiality, Integrity and Availability (CIA) of the system
- Improving traceability and transparency. (All the data are visible to network participants)
- Reducing cost by removing central intermediaries

src: https://www.f5.com/labs/articles/education/what-is-the-cia-triad

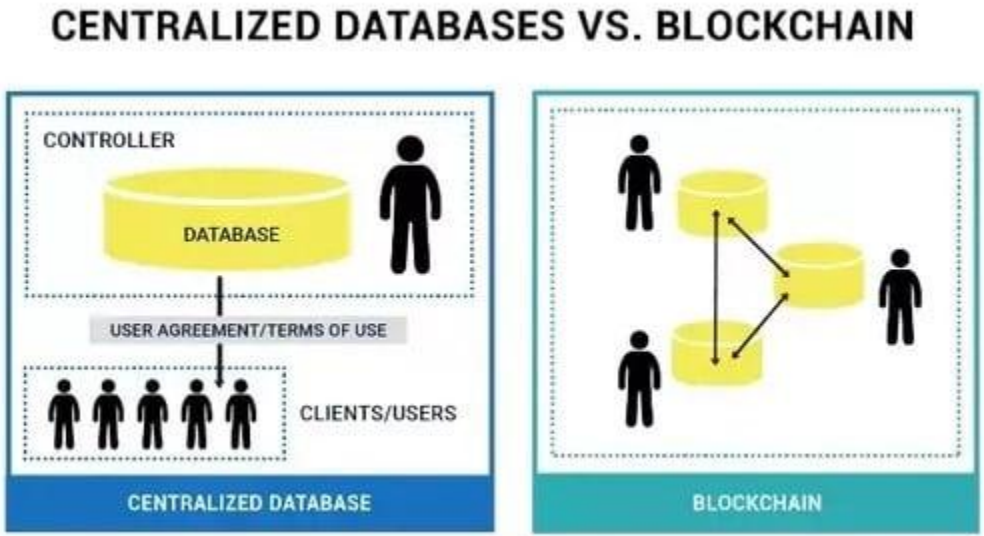
Security aspects of Blockchain



Source: Makhdom et al.[5]

What is the main difference between Blockchain and traditional databases?

Blockchain VS. Database

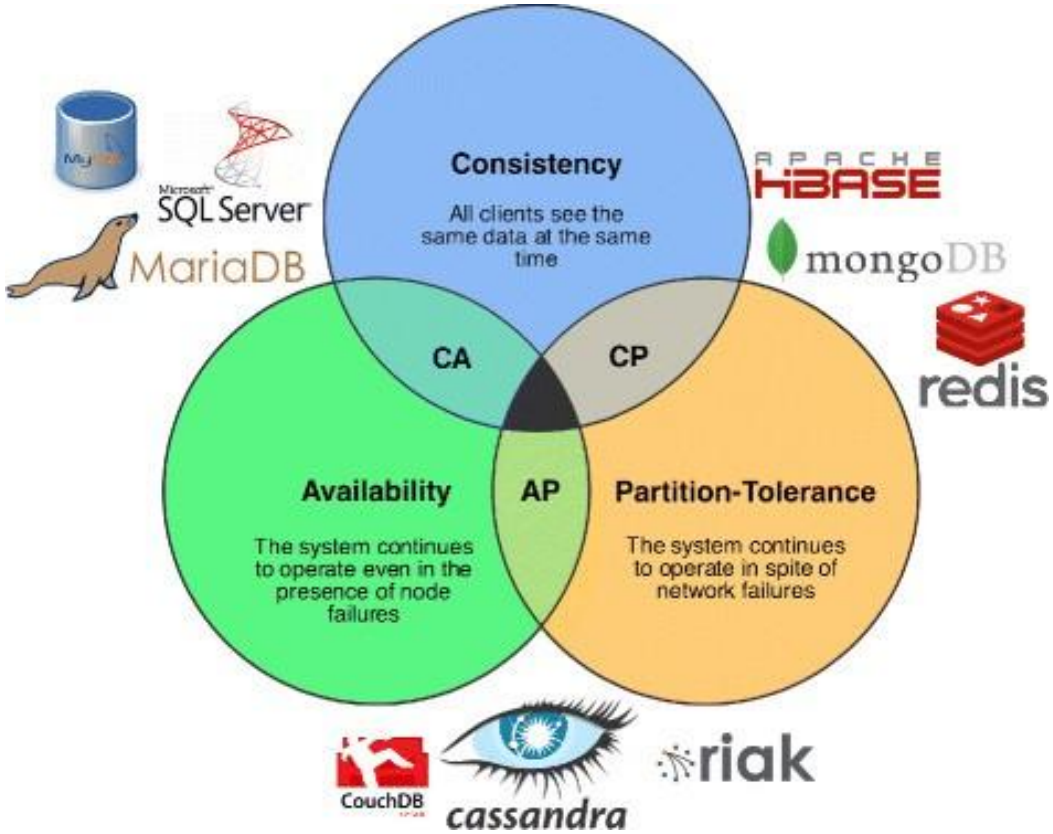


src: intellipaat.com

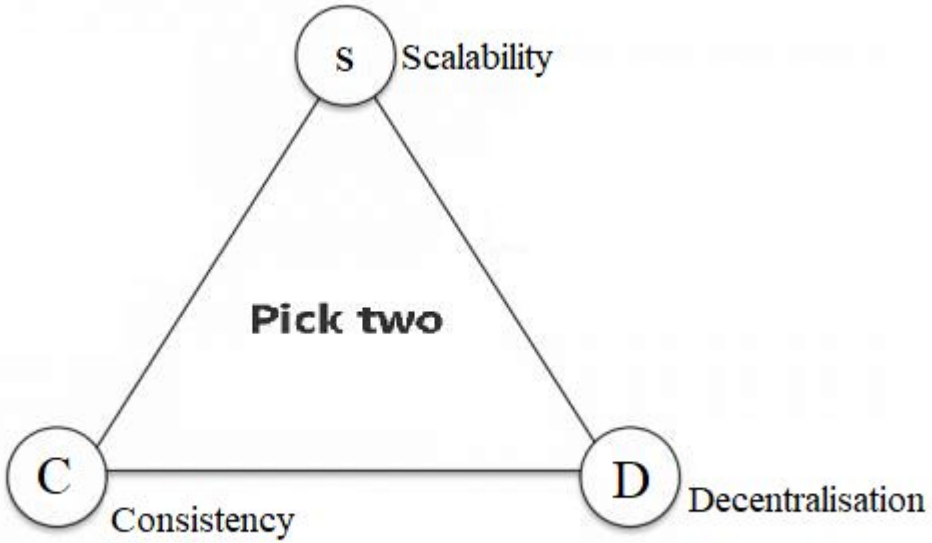
Database: Centralized, Permissioned and Requires Admin

Blockchain: Decentralized, Permissionless and No Admin

CAP Theorem

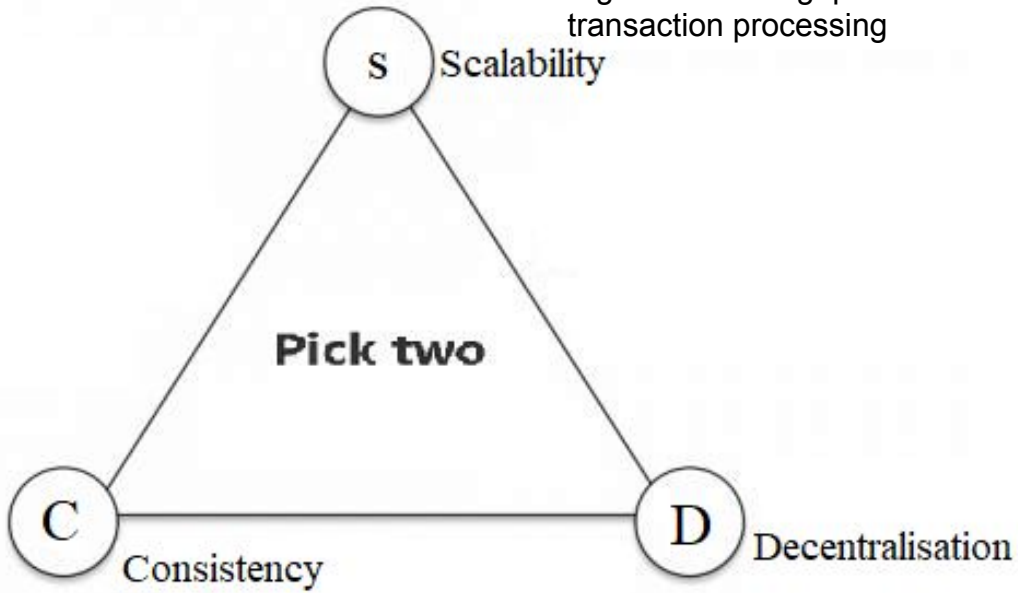

Consistency
All clients see the same data at the same time
Availability
The system continues to operate even in the presence of node failures
Partition-Tolerance
The system continues to operate in spite of network failures
CA
CP
AP
MariaDB
SQL Server
APACHE HBASE
mongoDB
redis
CouchDB
cassandra
riak

The Blockchain Trilemma:



DLT Theorem:

- Large number of users
- Efficiency for data storage
- Significant throughput for transaction processing

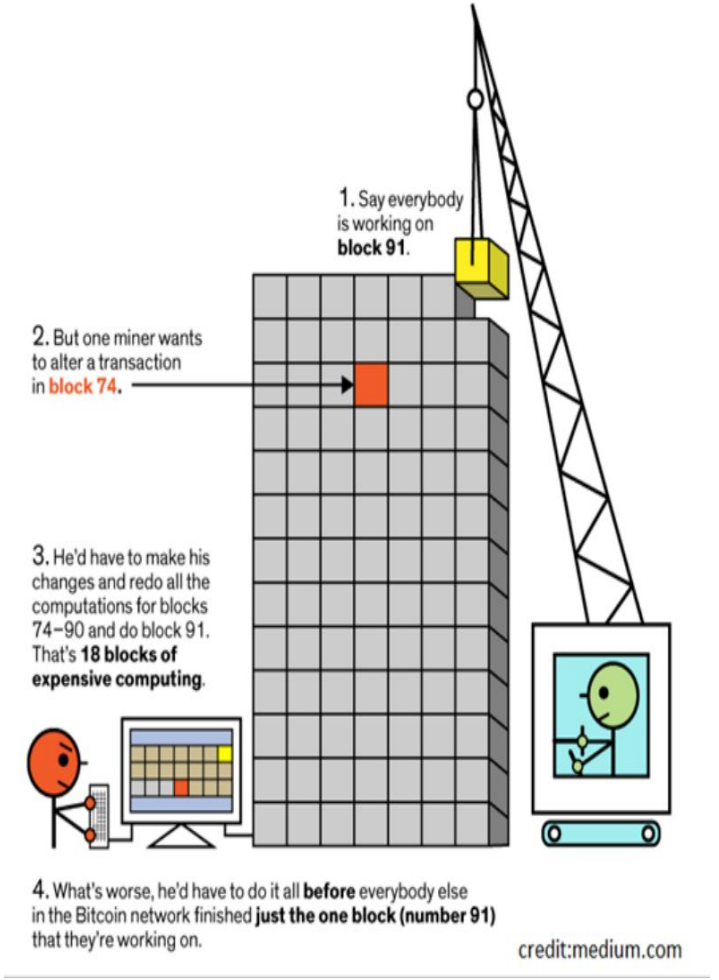


- Worldwide Availability
- Nodes in Public network will be rewarded
- No need for central authority

- Timestamping
- Data immutability and integrity
- Security and Fault Tolerance

Why Blockchain is tamper-proof?

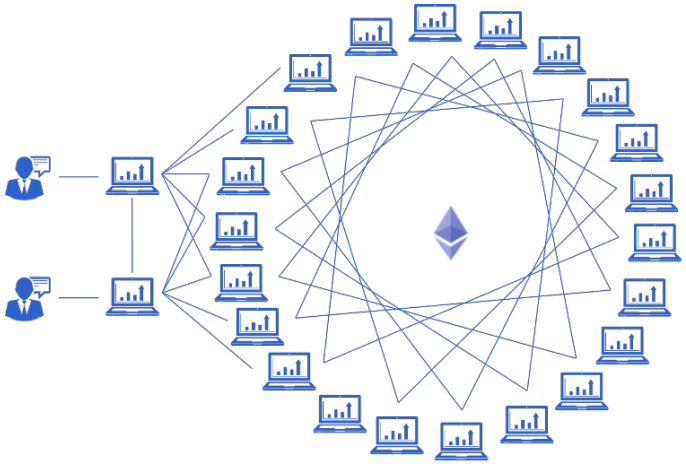
- If an adversary alters the Block K, this will produce a new hash which is different the one in Block K+1.
  - As a result, it will invalidate the link between Block K and Block K+1



What makes Blockchain tamper-proof?

Consensus protocol: There is a **unanimous acceptance** between participants of the network for every change or new submission.

Cryptographic fingerprint: Called hash, it takes a lot of computing power and energy to generate initially.

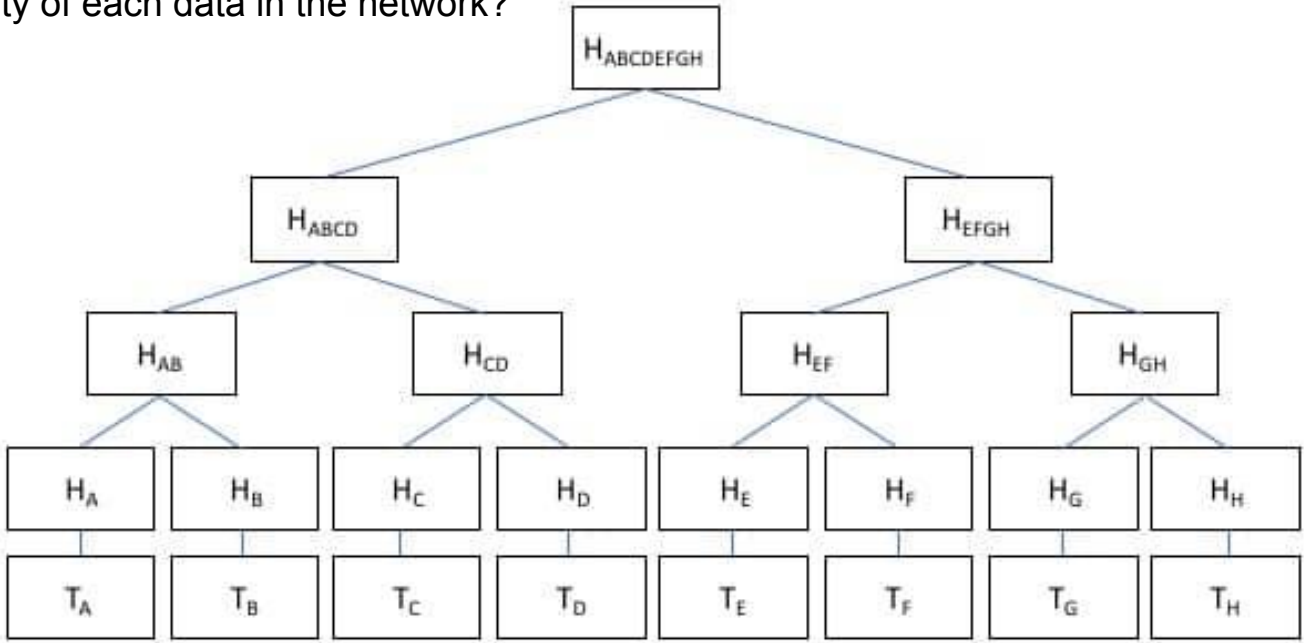


economicpoint.com

Merkle Tree:

- Is there a way to recognize any tampering in the Blockchain?

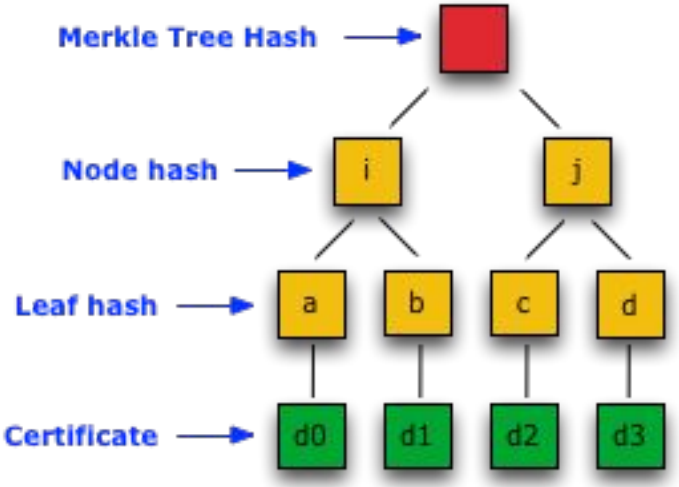
- What approach is needed to check the validity and authenticity of each data in the network?



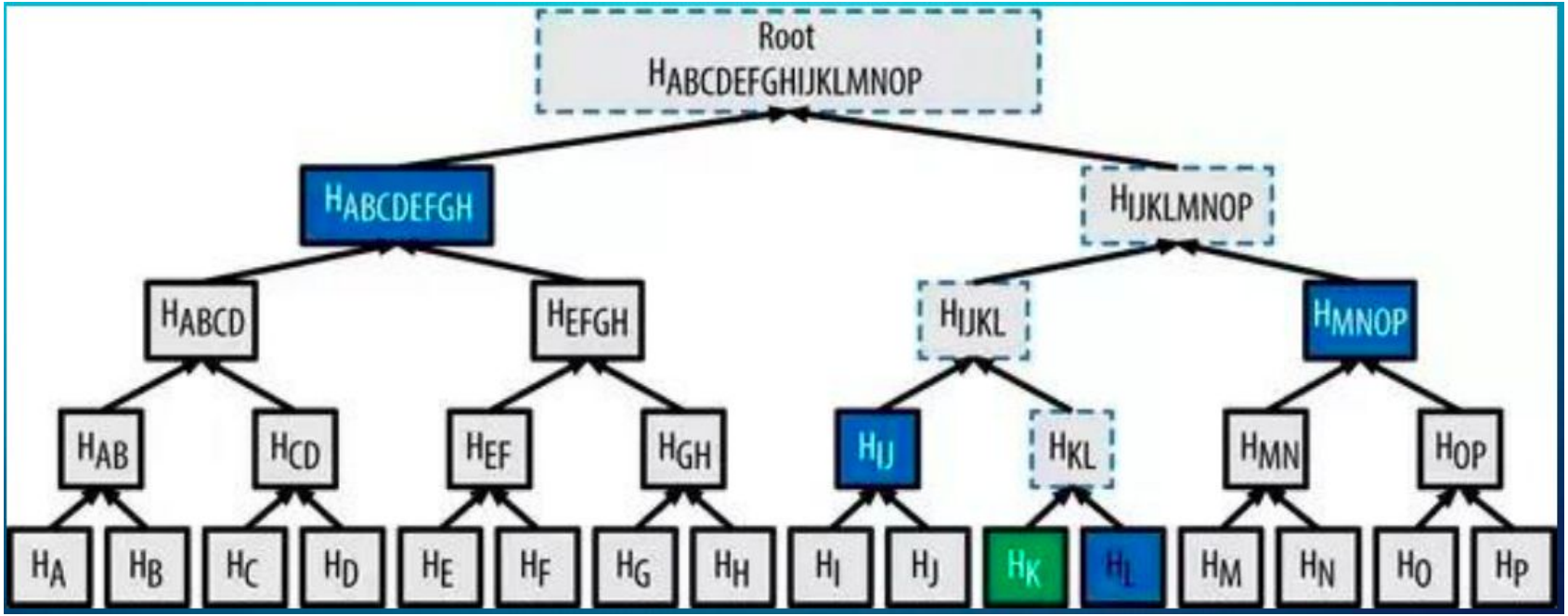
Source: bitcoininsider.org

Merkle Tree: Proof of Membership

- How can we audit the block D3?



Merkle Tree: Proof of Membership



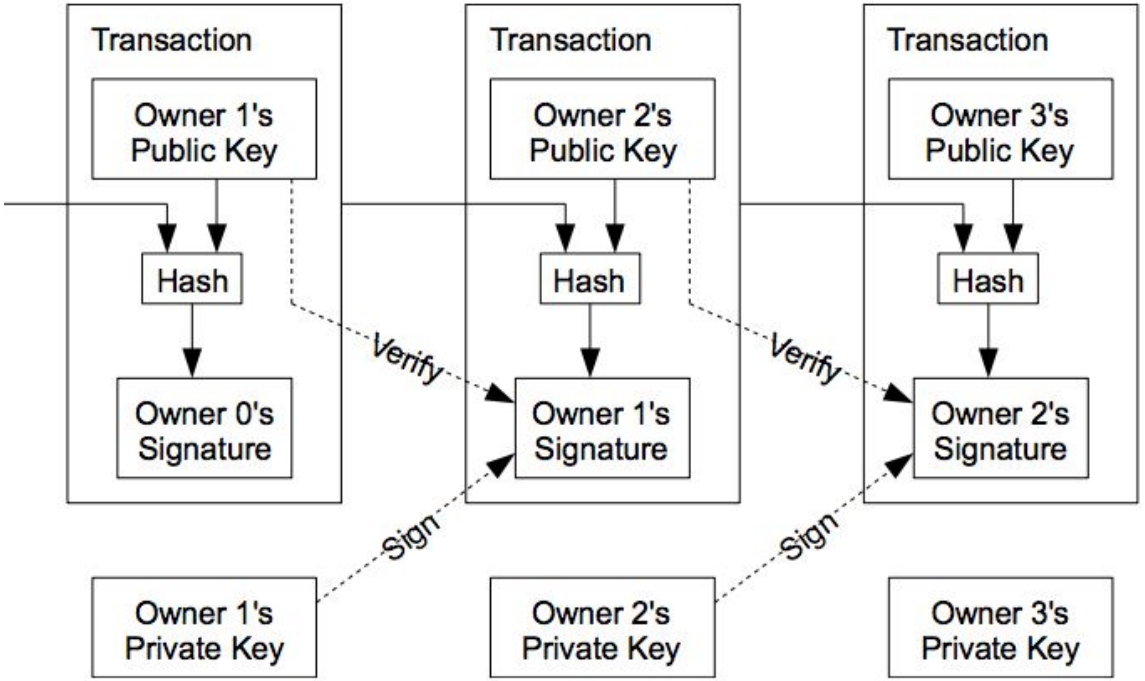
source: medium.com

Merkle Tree: The procedure of transaction verification in Blockchain

- Merkle Tree enables users to authenticate for themselves when a transaction has been inserted to the network blocks.
  - By tracing the Merkle root which is retrieved from block header.

- In order to authenticate and verify the Transaction A is added to the Block, the user only need few hashes including merkle root.
  - There is no need for client to know about any other transactions.
  - This will save significantly in terms of transmission bandwidth and delay.

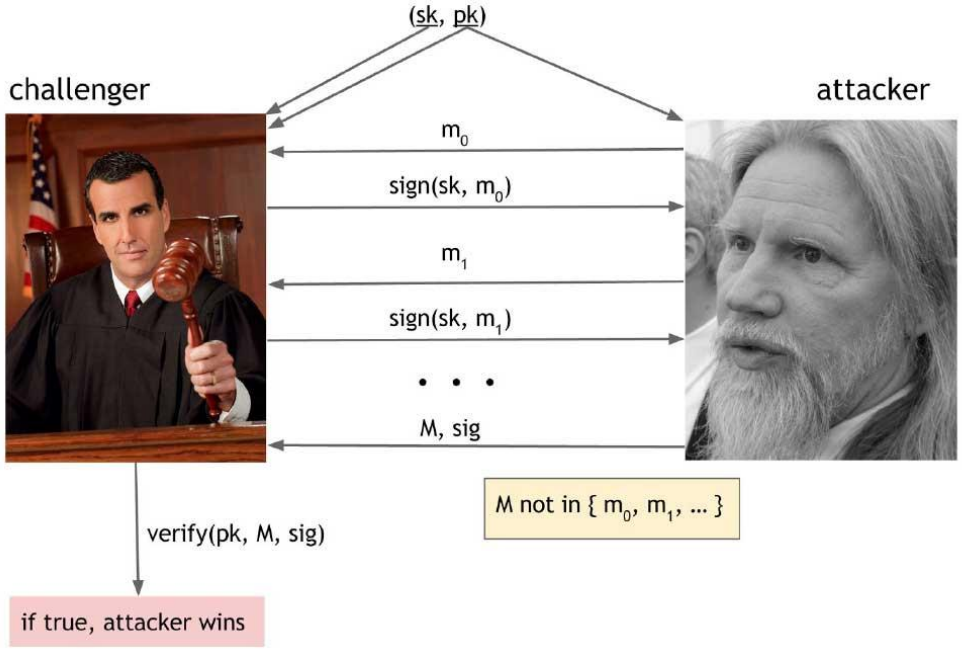
Digital Signature in Blockchain



source: stackexchange.com

Digital Signature: Unforgeability

“An adversary who knows your public key and your signatures on some messages cannot can neither forge nor generate your signature on some other message.”



source:aicody.com

Consensus Protocol:

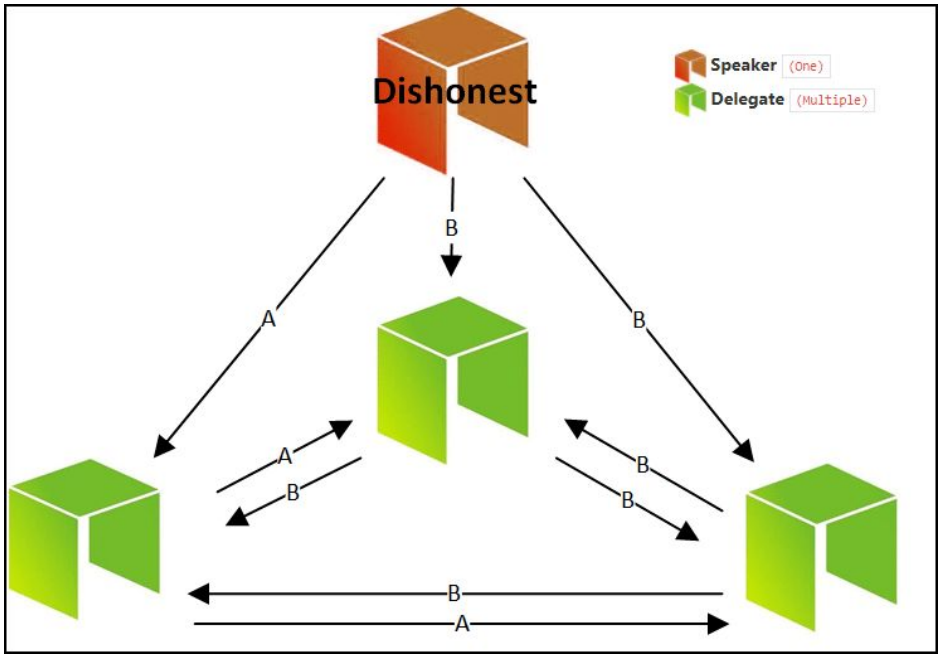
There n nodes in the network which have an input value. Some of these nodes could be either faulty or malicious. The consensus protocol will have the following two steps:

1) Terminate all honest nodes in agreement on the value

2) THe value must be only generated by an honest node

Consensus

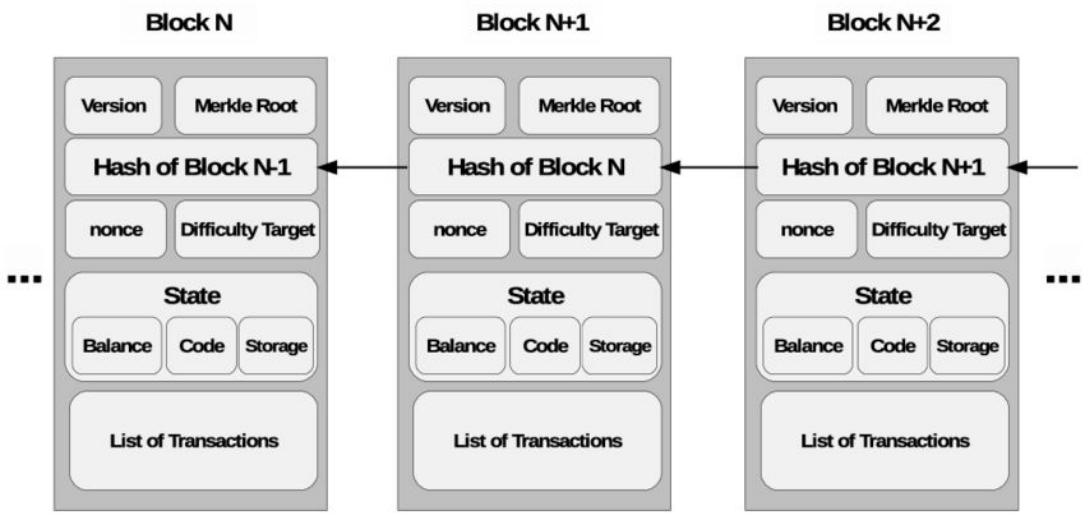
In Blockchain network, nodes need to agree on exactly which transaction were broadcast and the order in which these transactions happened.



source: blockgeeks.com

Consensus

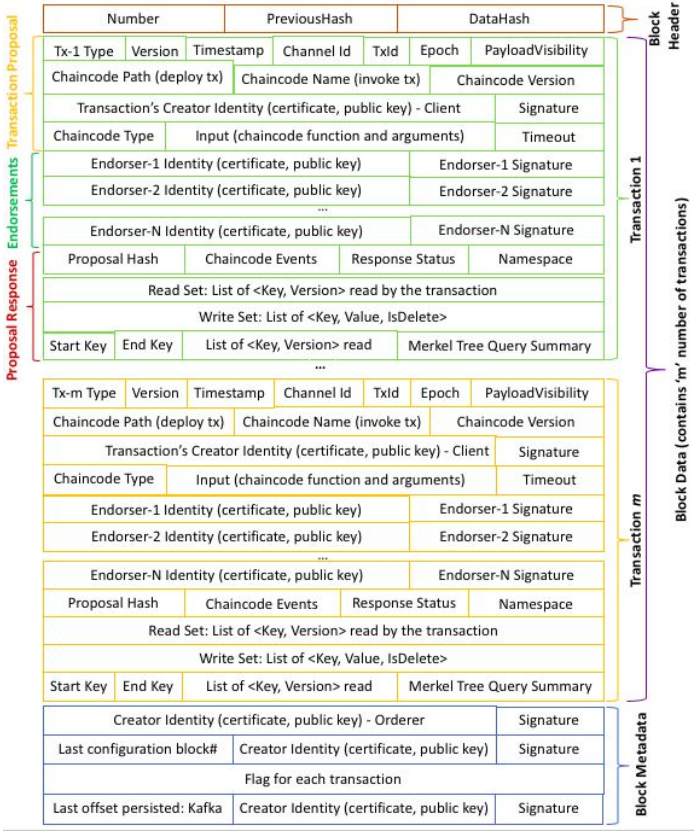
- In Blockchain, all the peers must agree on exactly which transaction were broadcast and the order in which these transactions happened.
- All the times, the participants of the P2P network own a ledger which contains of a sequence of blocks, each consists a list of transactions that they have approved.



source: Khaled Salah et al.

Consensus

- Network participants in Blockchain network might have different copies of the ledger and the transactions which also includes the list of pending transactions. (Why?)



source: bchainledger.com

Consensus

- Does each node have the same list of transactions that are broadcasted?
- Does each node have the same ledger?

Byzantine Generals Problem:

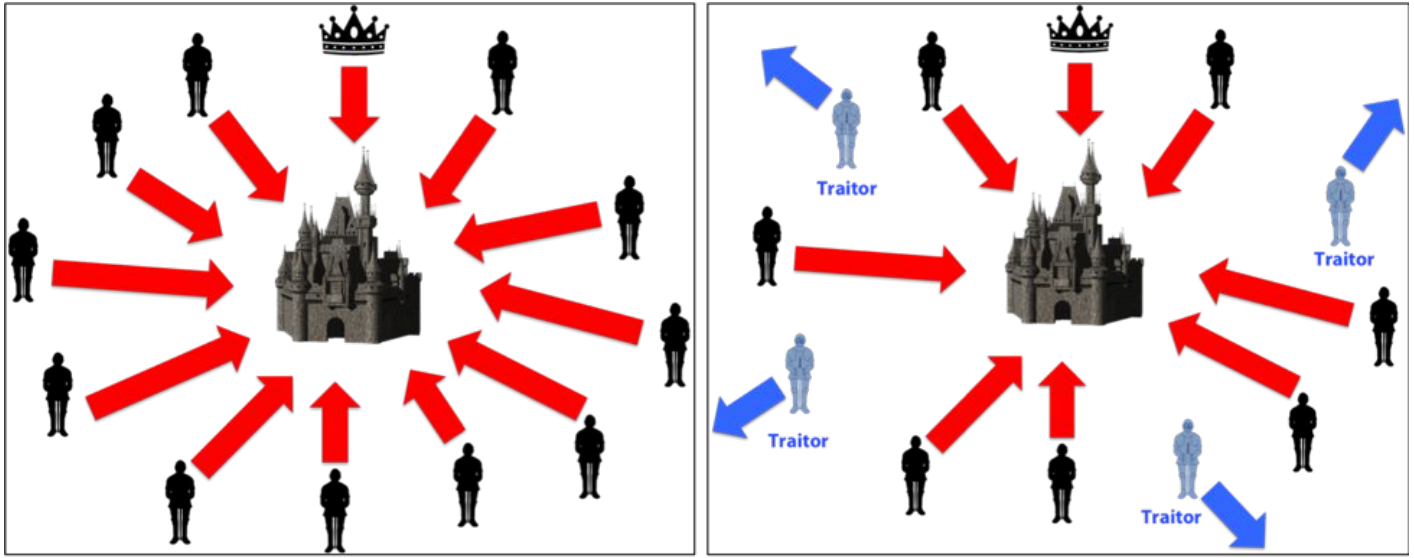
Byzantine Generals Problem:

- There is an army which consists several generals.
- Generals communicate via messengers.
- There are generals who are traitors.
- **Our goal:** The honest generals arrive at the the same decision (either attack or retreat). The traitorous general will not be able to impose false decision to honest generals.

How?

Byzantine Generals Problem:

- If 1/3 or more are traitors it will be impossible.
- (3f+1) where f is the number of traitorous generals.



Coordinated Attack Leading to Victory **Uncoordinated Attack Leading to Defeat**

source: ivanotech.com

- How many nodes we need for BFT?

Blockchain Consensus Protocol: Proof of Work

- Incentives: Nodes will have incentives to reach to the consensus (getting rewarded)
- Randomness : The nodes will be selected in a random order, without any central authority.
- The nodes are following pseudo-anonymity which means that they don't have to be persistent.
- There is neither starting point nor eding for to reach to the consensus. This could have for a long period. (eg., 1 hour or more in Bitcoin network)

- Do all the nodes need to participate in the consensus?

Blockchain Consensus Protocol: Proof of Work

How the coins can not be stolen in Blockchain in example of Bitcoin?

“• If Alice wants to steal bitcoin (transfer Bitcoin from the bitcoin owner to her address), it needs to sign the transfer transactions using the private key of the owner (which she does not have!)”

How Blockchain can avoid Denial of Service Attacks?

- Alice can block the inclusion of a transaction (of Bob) in any block she creates the transaction will remain in the the network.
  - This means that it will be included in the next block which is created by another node.

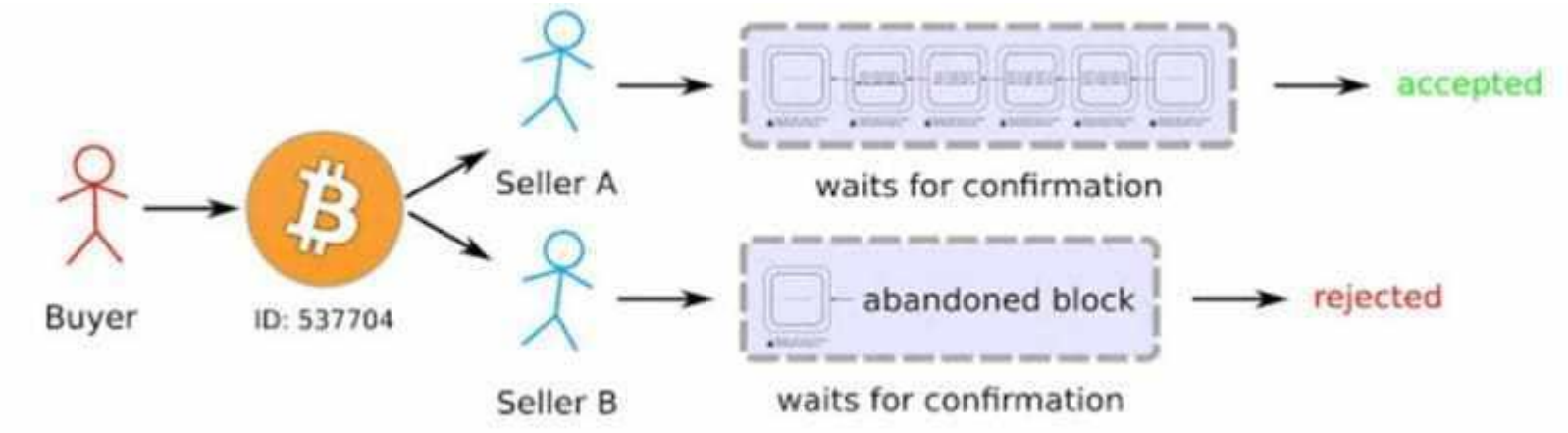
- The most critical point is the random selection of the participating nodes in the protocol.

How Blockchain can avoid double-spending attacks?

- Alice buys a video game from a Bob.
- The transaction will be broadcasted (payment instruction, hash, Alice's signature).
- Hash is the pointer to some previous transaction. (for example where Alice has received Bitcoin).
- Transaction will be included in the Block.
- Bob lets Alice to download the video game. However.
- Alice will use the same Bitcoin to buy a product from Josh.
- The transaction will be broadcasted (Hash, ALice's signature and payment instruction).
- Transaction will be included in the next Block.
- It happens that Alice is the node selected to create this Block.
- She ignores the block that has been created with Bob's transaction and starts from previous one.
- Now we have two chains. (Alice to Bob and Alice to Josh)

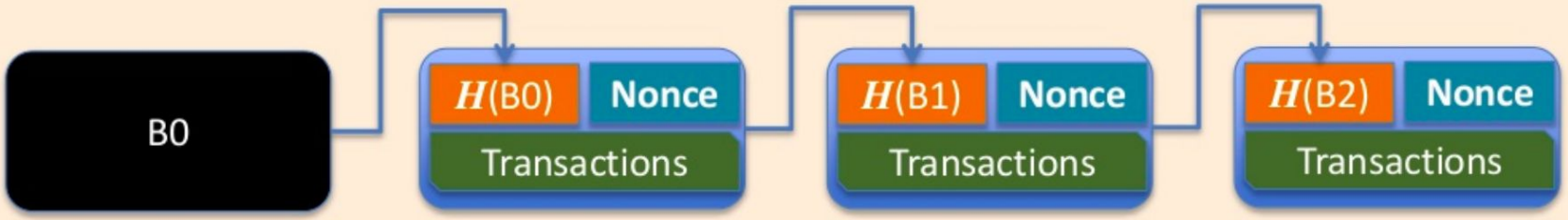
How Blockchain can avoid double-spending attacks?

- Which transaction will be succeed?
-



source: coinsutra.com

Bitcoin's Proof-of-Work



Find a nonce x such that:

SHA-256(SHA-256(r || x)) < T/d

r = header includes H(previous block)
root of Merkle tree of transactions

Source: David Evans, University of Virginia

How Blockchain can avoid double-spending attacks?

- The node that will create the next block will “determine” which transaction will be included.
  - The nodes will only build on one of the chains.

- The nodes will only agree to longest and most difficult chain to reach consensus.
- Can we have 2 or more blocks that are different but VALID at the same time in the network?

What are the options for Bob?

- Allowing the Alice to get the video game as soon as Alice has broadcasted the transaction?
- Allowing Alice to get the video game as soon as the transaction is confirmed in a new Block?
- Allowing Alice to get the video game after certain amount of confirmation (which means that several block are appended to the block that includes the transaction to Alice to Bob)?

- It is recommend that we wait for 6 confirmation in Bitcoin.

How Blockchain can avoid double-spending attacks?

- Double spending has nothing to do with cryptography since both transactions are considered to be valid authentic and cryptographically valid.
- It the consensus which will determine to approve which transaction should be included in the Blockchain.
- But, it does not guarantee which one will be included.
- It needs several confirmation to decrease the probability of the double spending transaction.

The Vulnerability of proof-of-work in Blockchain:

- If the adversary take the control 51% resources, they can be able to make decision on behalf of the all participants.

Why it doesn't to make 51% in Blockchain?

- It is expensive to control 51% of hash power around the world participants.
- Not only the hardware, but also the electricity consumption.
- Bitcoin network is currently consuming more electricity than Ireland.
- More importantly, it makes zero sense from financial point of view.
- The attacker will be self-destroyed: Bitcoin value will go to zero.
- Blockchain miners have spends hundreds of millions of dollars on the infrastructure required compete in the mining sector.

Example a Bitcoin miner:

‹ Retour aux resultats

Cliquez pour afficher une image agrandie

DragonX AntMiner

Visitez le Store DragonX

8 évaluations

Prix : **4 946,77 CDN$** + 3,95 CDN$ Livraison

Couleur: **S17 59T**

5 068,43 CDN$	**4 946,77 CDN$**	2 478,27 CDN$	5 071,24 CDN$
4 479,71 CDN$	494,56 CDN$	588,15 CDN$	2 387,20 CDN$

- Bitcoin Mining Hash Rate: 59TH/s ±5%, with psu
- Power Consumption: 2385W ±10%,Extremely Efficient Bitcoin / Bitcoin Cash Miner
- Built-in web management portal, build-in power supply
- Please be aware that AntMiner S17 59TH is not accept return and refund, the manufacturer will provide warranty for 180 days,if you return within 20 days, we will charge 40% restock fee.
- More USED Miners and hosting service available

Avez-vous besoin de plus d'informations sur les produits en français? Contactez-nous

4 946,77 CDN$

+ 3,95 CDN$ Livraison

Livré : **20 - 28 oct.**

En stock.

Quantité: 1

Ajouter au panier

Acheter maintenant

Transaction sécurisée

Ships from Canada and sold by lighting-geek.

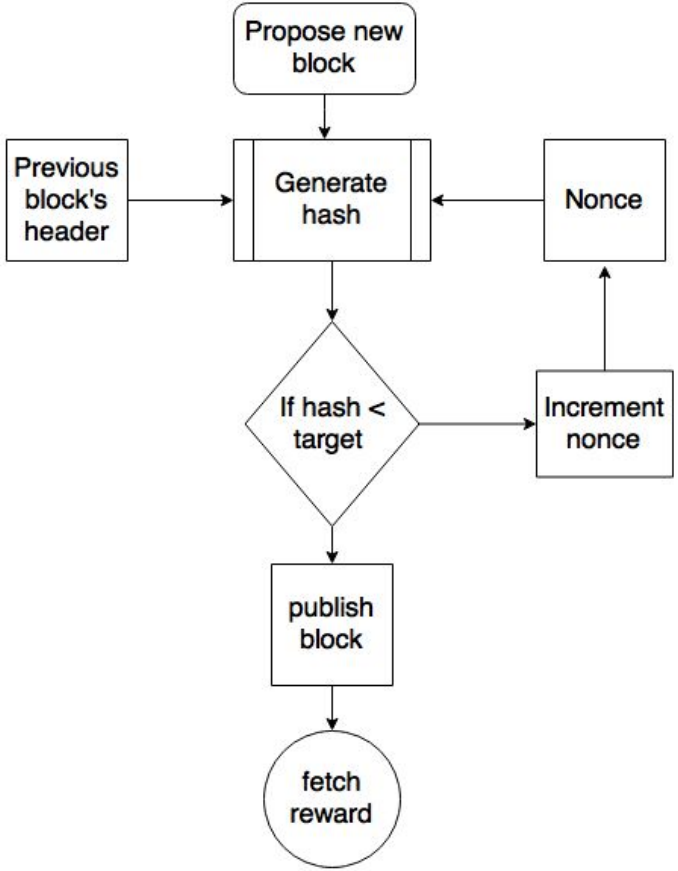
Entrez votre adresse

Ajouter à votre liste d'envies

The mining algorithm:

The mining algorithm consists of the following steps:

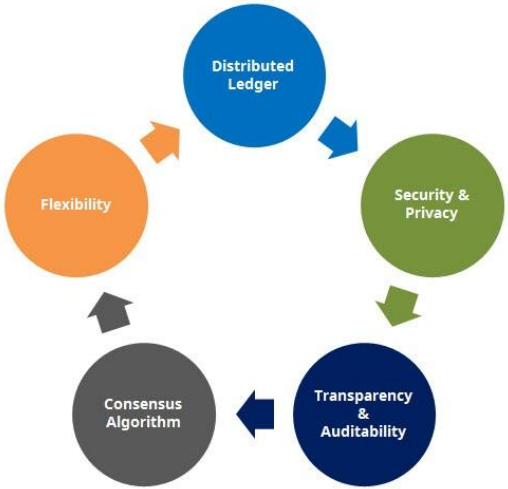
1. The previous block's header is retrieved from the bitcoin network.
2. Assemble a set of transactions broadcasted on the network into a block to be proposed.
3. Compute the double hash of the previous block's header combined with a nonce and the newly proposed block using the SHA-256 algorithm.
4. Check if the resultant hash is lower than the current difficulty level (target) then PoW is solved. As a result of successful PoW the discovered block is broadcasted to the network and miners fetch the reward.
5. If the resultant hash is not less than the current difficulty level (target), then repeat the process after incrementing the nonce.



source: codeproject.com

Key characteristics of Blockchain

- Permissionless
- Trustless
- Transparent
  - It is visible to anyone but difficult to realize Immutable (When transaction confirmed, it can neither be manipulated nor removed)
- Immutable
  - It is computationally infeasible to either change or remove.
- Secure (Unhackable)



Some characteristics of Bitcoin (summary)

- Block added each 10 minutes
- Uses SHA-256 as a hash function
- Uses ECDSA for signature
- Miners are mainly paid by minted (created) coins (bitcoins created by new block)

Ethereum

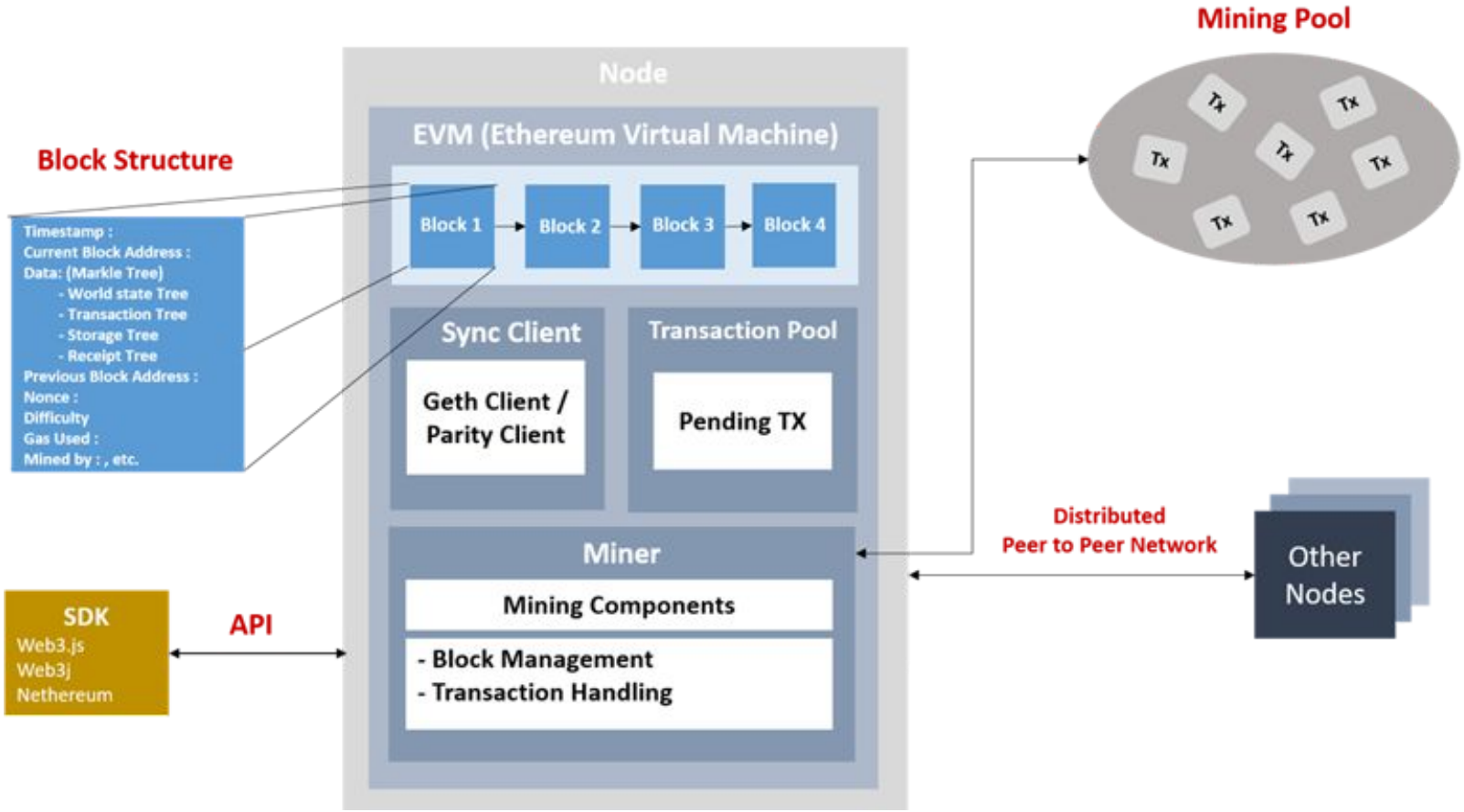
Ethereum

- In general, Ethereum Blockchain is designed a public network. (Based on PoW, P2P and cryptographic primitives)
- In compare to Bitcoin, Ethereum is not a digital currency payment system.
  - Ether is utility which is used to pay for the use of Ethereum platform.
- Ethereum is a general-purpose programmable Blockchain that runs a virtual machine (EvM).
  - Ethereum's language is Turing complete (Solidity) compared to the basic scripting used in Bitcoin.
  - Any program of any complexity can be computed by Ethereum.

Programming in Ethereum

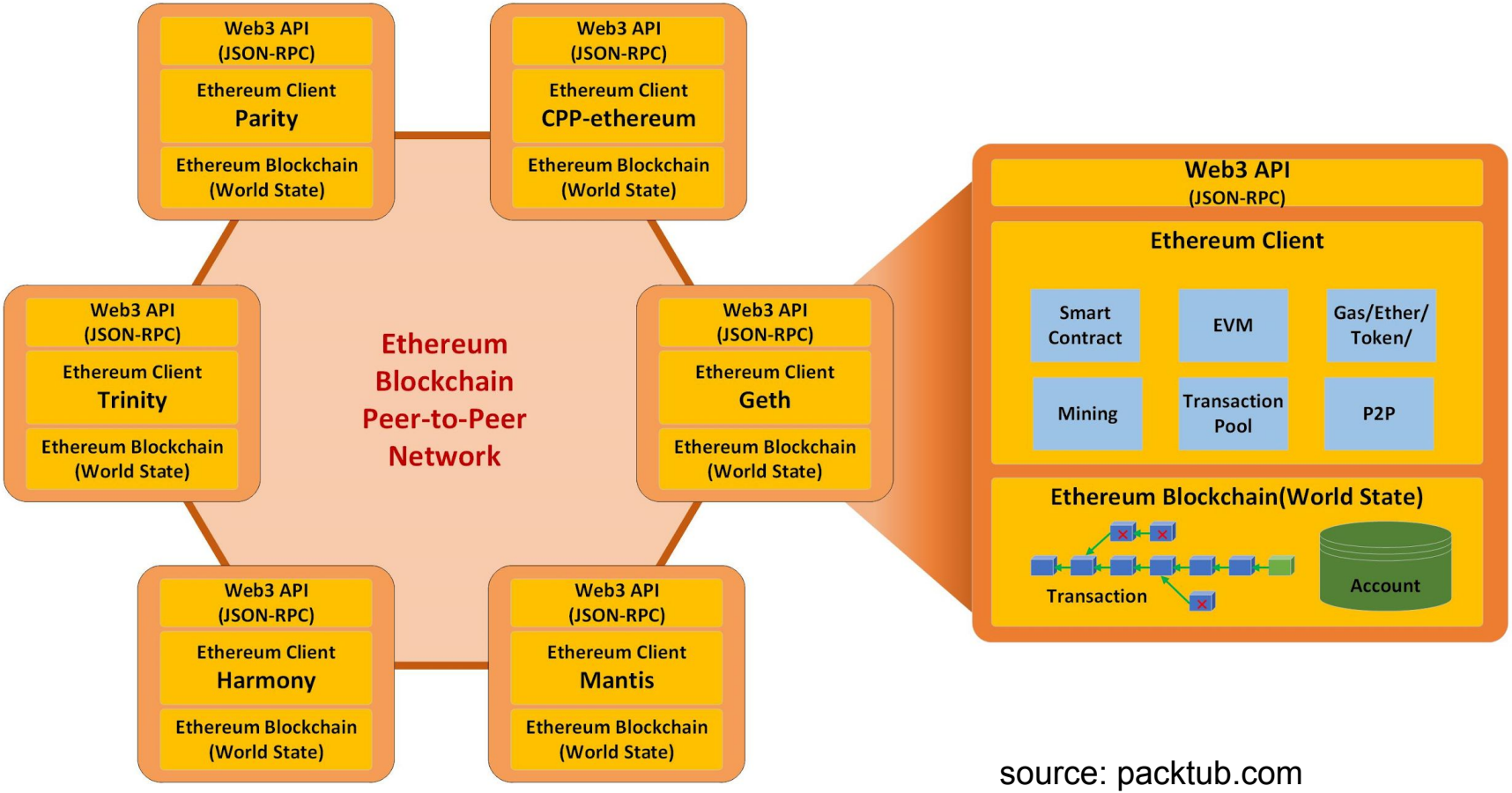
- In Ethereum, each instruction (e.g., computation, data access, data storage, etc.) has a predetermined cost in units of gas.
- In order to submit a transaction in Ethereum, the programmer must include an amount of "gas" that can be consumed.
  - The code execution will be aborted if the amount of the gas consumed exceeds the gas available in the transaction.
  - When the transaction completes any unused gas is sent back to the sender of the transaction.
  - If you don’t send enough gas, what you sent is still lost, because the miner still tried doing the work, so will take it and your transaction will never been confirmed. However, all the operations will be reverted.**

Ethereum Architecture



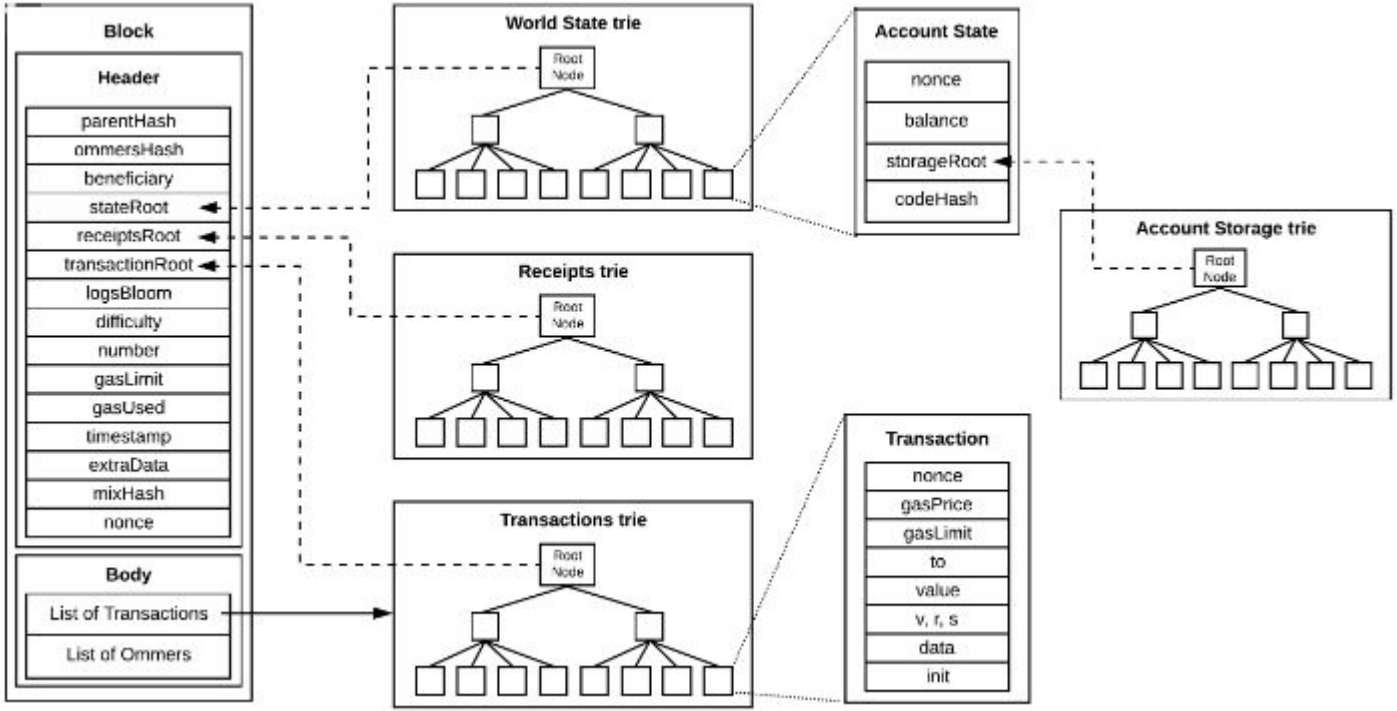
source: flentas.com

Ethereum Architecture



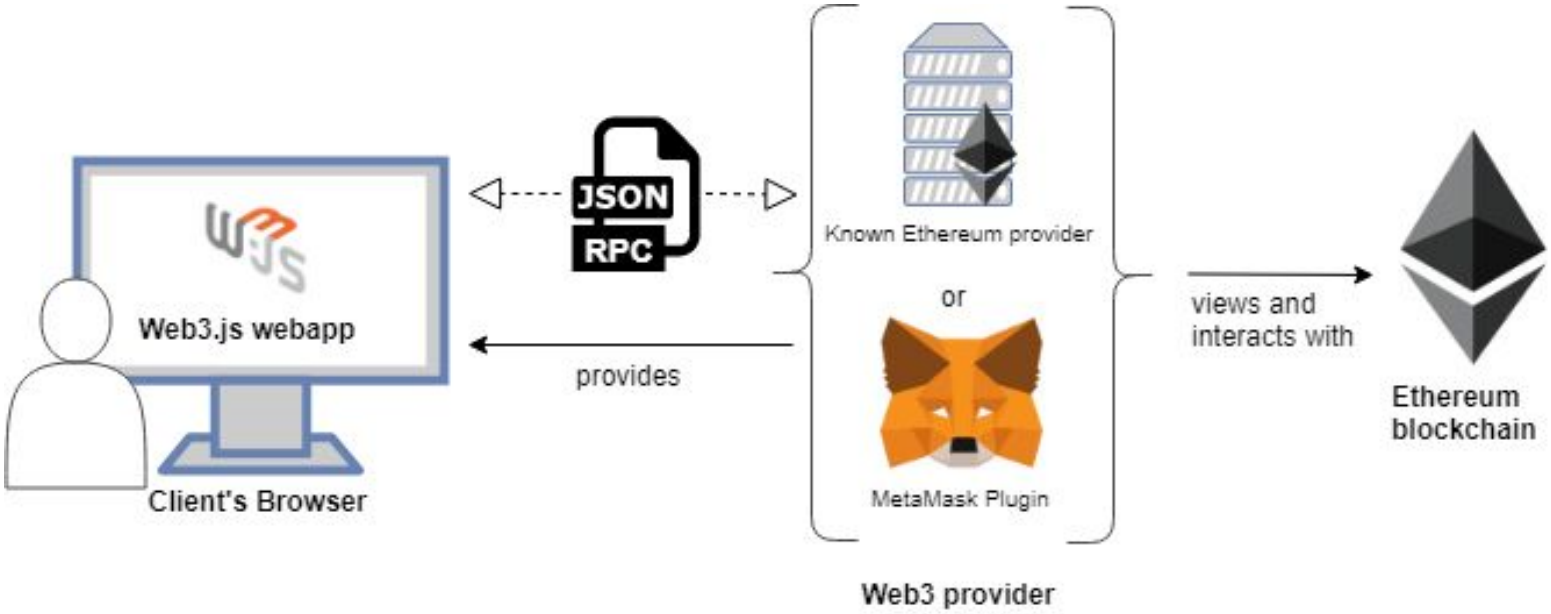
source: packtub.com

Block Architecture in Ethereum



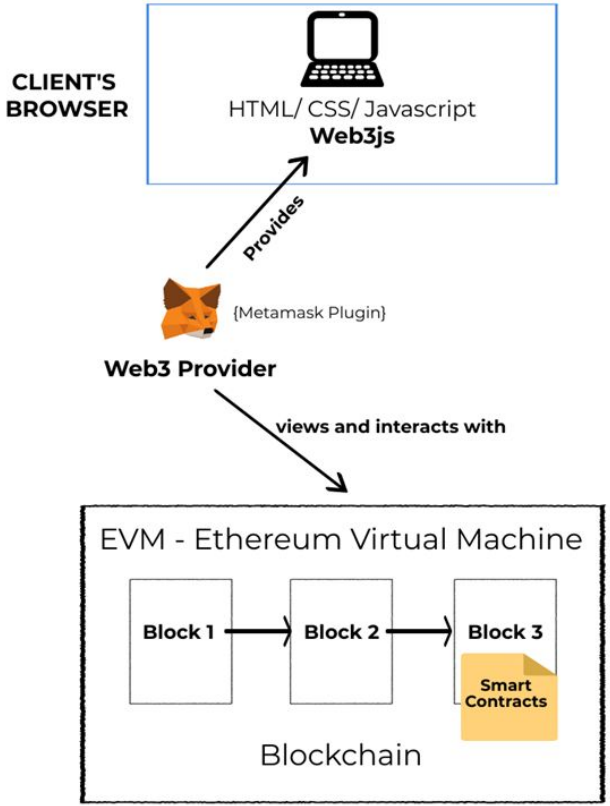
source: kauri.io

Ethereum Web3 Architecture

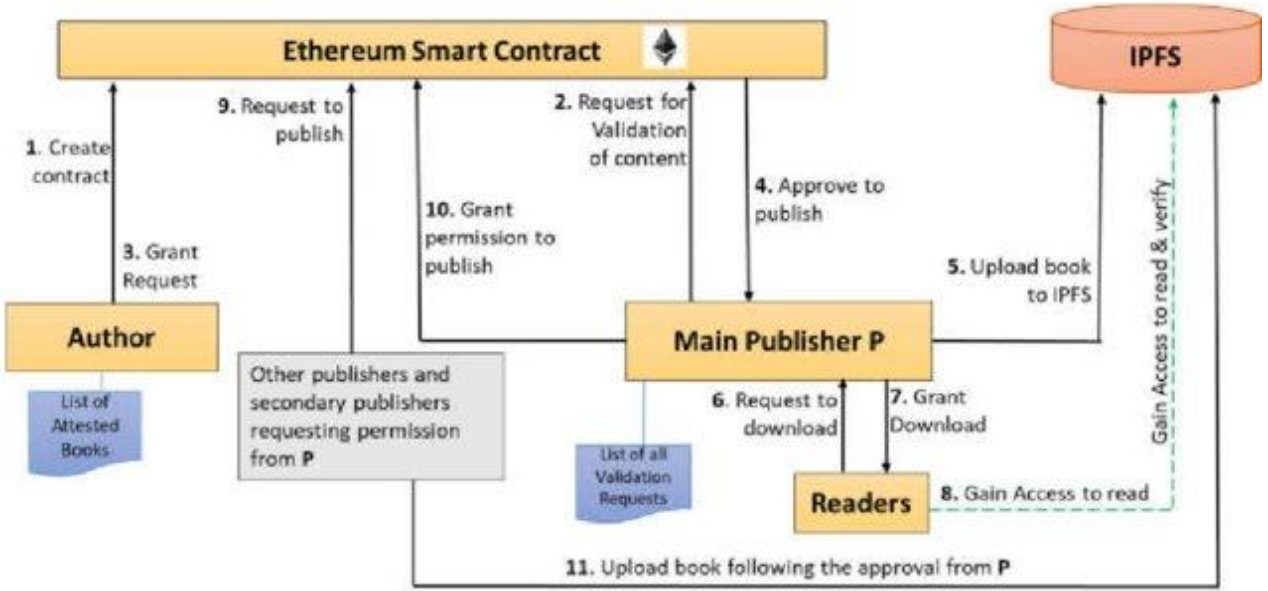


Source: Github

Client interaction with Ethereum



Ethereum example for book authenticity validation:

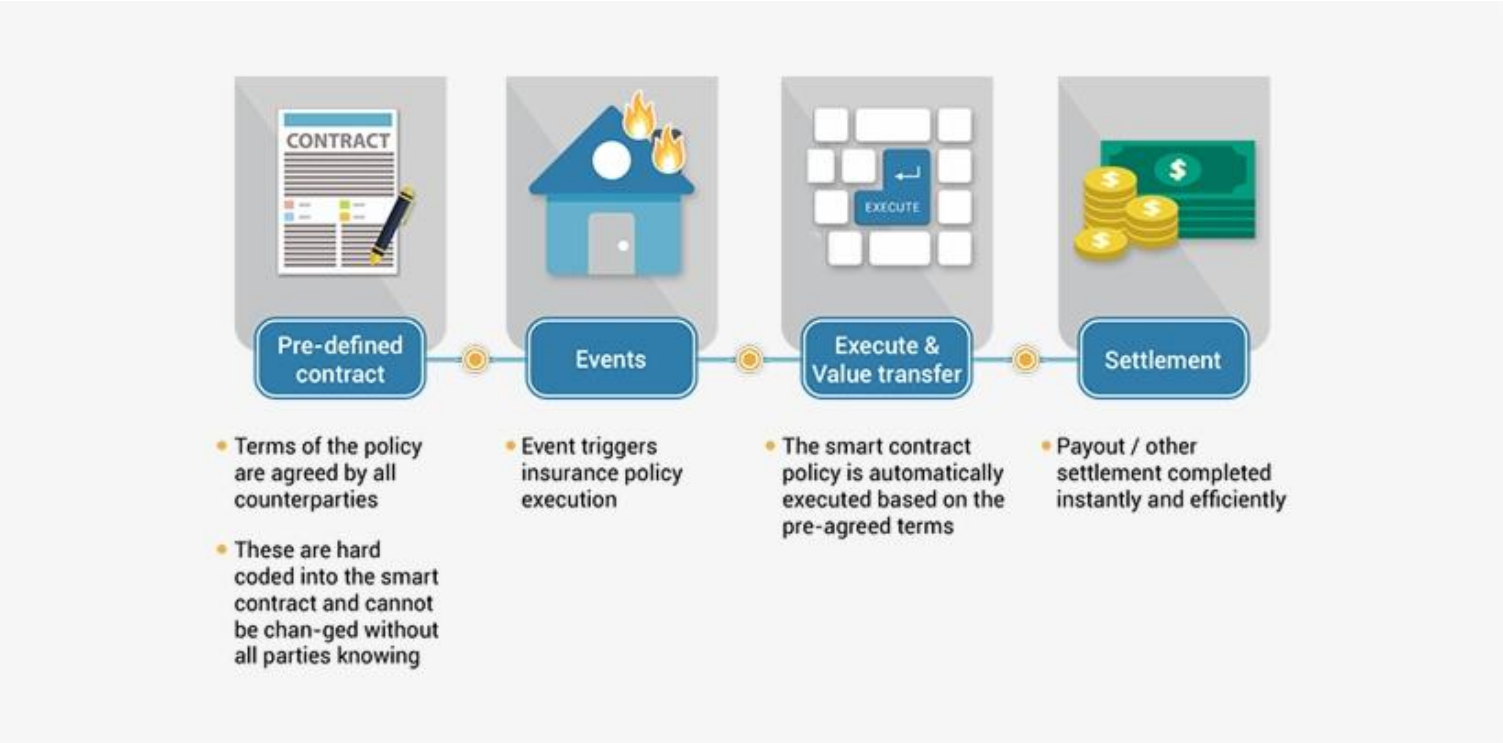


Source: Khaled Salah et al.

Smart Contract

- “Smart contracts define the rules and penalties around an agreement in the same way that a traditional contract does, but also automatically enforce those obligations.”
  - For example, if the car payment is not made on-time, the car gets digitally locked until the payment is received.
- “A smart contract is a piece code and data that is deployed to a blockchain (e.g., Ethereum)”
- “A smart contract can perform calculations, store information, and automatically send funds to other accounts”.
- A smart contract is immutable and unalterable which can be trusted by participants to execute operations ranging from simple cryptocurrency transfer to more complicated operations.

Smart Contract



source: devteam.space

The characteristics of smart-contracts:

- **Deterministic**: All participants of the network should produce the same output by giving them the same input.
- **Immutable:** Once it is deployed in the Blockchain network (Ethereum), it can neither be removed nor altered.
- **Verifiable:** A smart-contracts have a unique address.
    - Smart-contracts operate in a highly constrained and minimalistic execution environment (the EVM).
    - The most popular programming language for smart-contracts is Solidity.

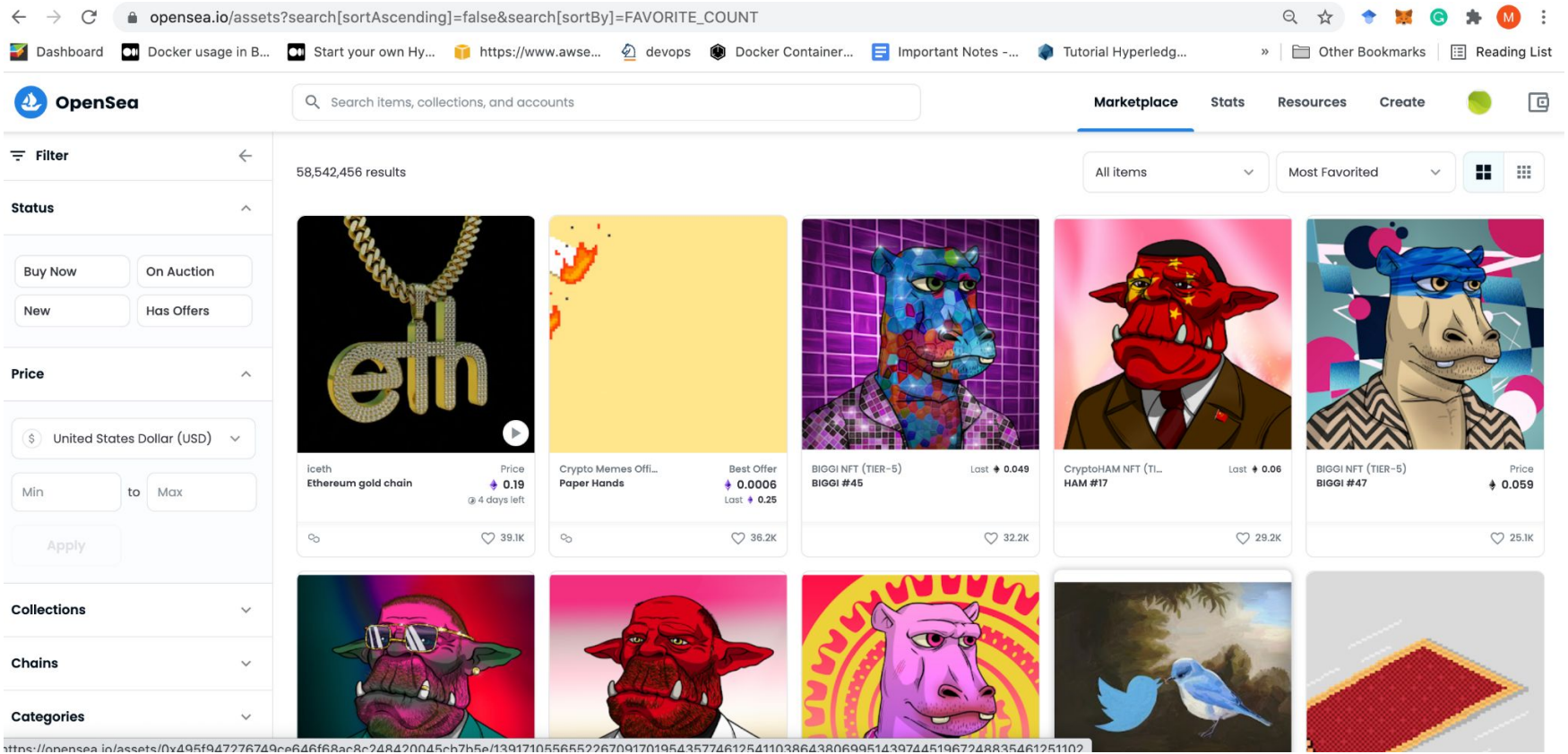
MOST IMPORTANT: smart-contract needs to be triggered, it can't decide: oh it's the first day of the month I need to do this.

- Challenges?
    - Bugs!

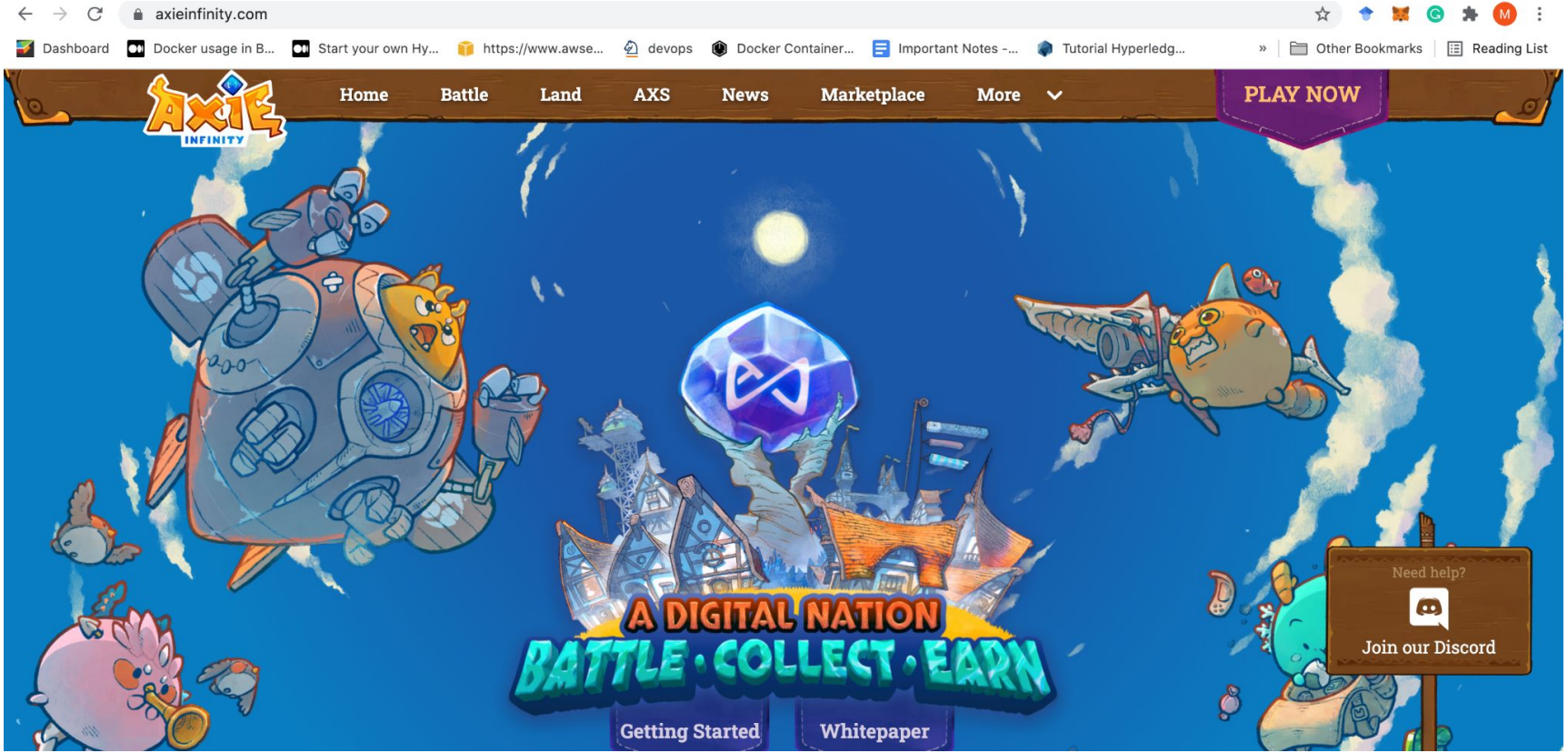
Smart-contract example

```
985  contract ERC20 is Context, IERC20 {
986      mapping (address => uint256) private _balances;
987
988      mapping (address => mapping (address => uint256)) private _allowances;
989
990      uint256 private _totalSupply;
991
992      string private _name;
993      string private _symbol;
994      constructor (string memory name_, string memory symbol_) {
995          _name = name_;
996          _symbol = symbol_;
997      }
998
999      function name() public view virtual returns (string memory) {
1000         return _name;
1001     }
1002     function symbol() public view virtual returns (string memory) {
1003         return _symbol;
1004     }
1005
1006     function decimals() public view virtual returns (uint8) {
1007         return 18;
1008     }
1009
1010     function transfer(address recipient, uint256 amount) public virtual override returns (bool) {
1011         _transfer(_msgSender(), recipient, amount);
1012         return true;
1013     }
1014
1015     function allowance(address owner, address spender) public view virtual override returns (uint256) {
1016         return _allowances[owner][spender];
1017     }
1018
```

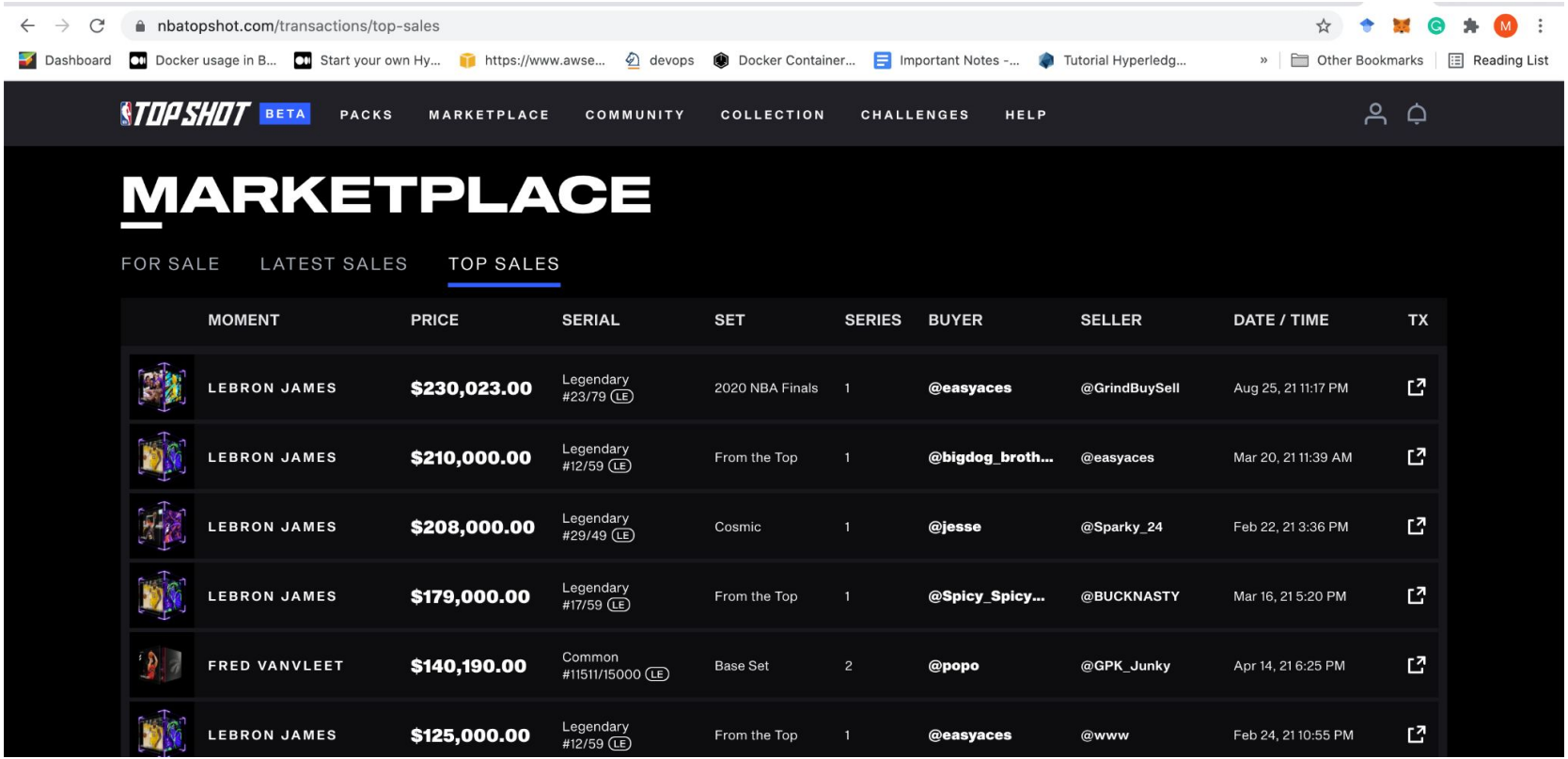
Example of Blockchain Marketplace for NFTs



Example of videogames on Blockchain



Example of NFT marketplace for NBA teams



Salary for Blockchain Engineers:

COMPARE BLOCKCHAIN ENGINEER SALARIES BY REGION

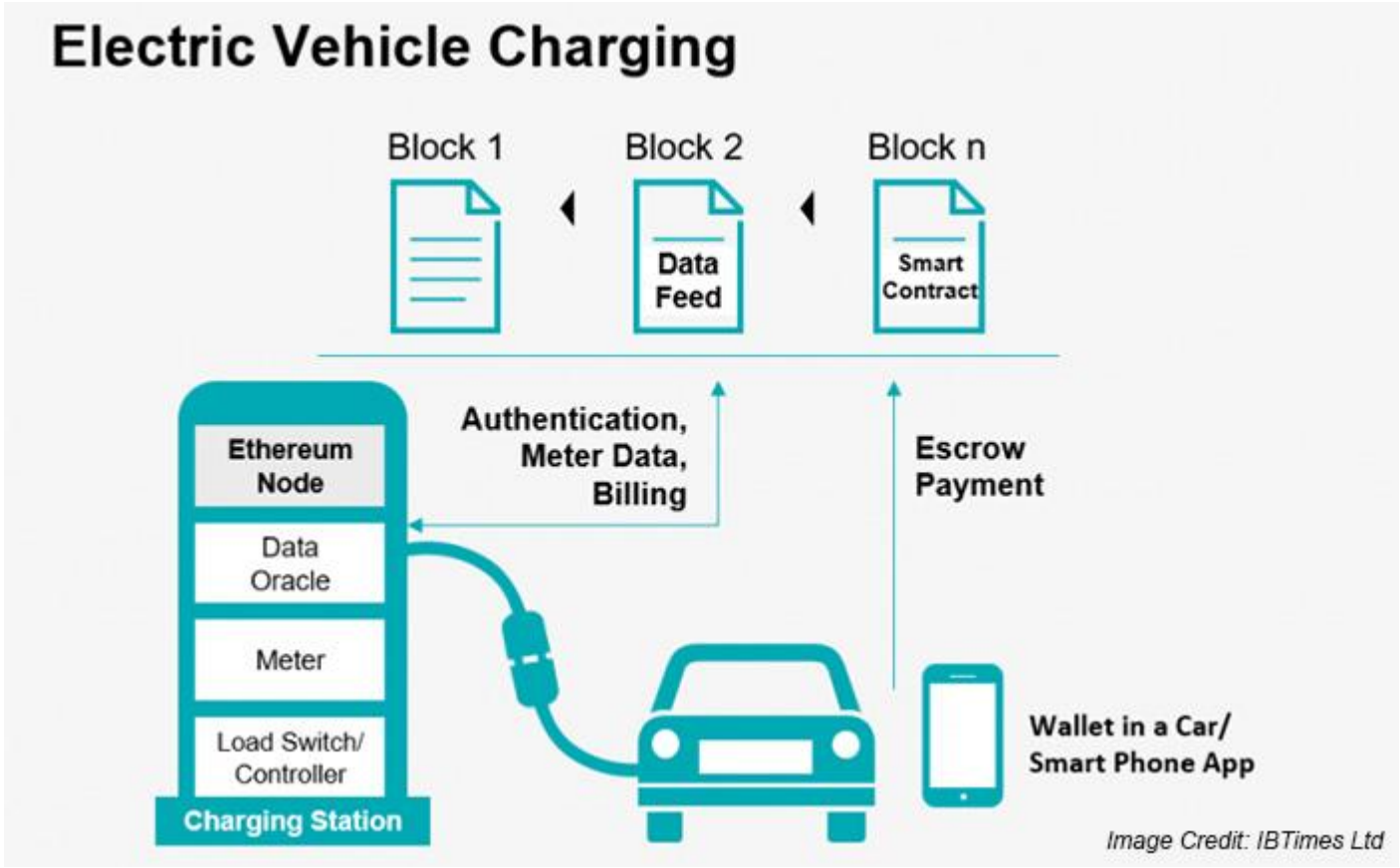
Blockchain Engineers are highest in demand in SF Bay Area, New York, and London. Browse and compare average salaries in locations where this role is also popular:

1.	SF Bay Area	$162,288	8.	Austin	$135,028
2.	Seattle	$153,181	9.	Denver	$133,465
3.	New York	$153,113	10.	Washington D.C.	$131,073
4.	Dallas/Ft Worth	$145,750	11.	San Diego	$123,333
5.	Boston	$144,985	12.	Toronto	C$118,281
6.	Los Angeles	$142,427	13.	London	£72,946
7.	Chicago	$141,880	**14.**	**France**	**€55,951**

source:devskiller.com/blockchain-developer-salary/

Examples:

- There are some potential applications that can be on-chain only (no outside data at all): like chess game, gambling (you just put the rules in the smart-contract and then it’s based off of ETH coins, so never had outside data needed)


Electric Vehicle Charging
Block 1
Block 2
Block n
Data Feed
Smart Contract
Ethereum Node
Data Oracle
Meter
Load Switch/ Controller
Charging Station
Authentication, Meter Data, Billing
Escrow Payment
Wallet in a Car/ Smart Phone App
Image Credit: IBTimes Ltd


Smart Contracts Use Cases
Record Storing
Trading Activities
Supply Chains
Mortgage
Real Estate Market
Employment Arrangements
Copyright Protection
Healthcare Services
VOTE
Government Voting
Insurance Claims
Internet-of-Things (IoT)
101 Blockchains
Created by 101blockchains.com

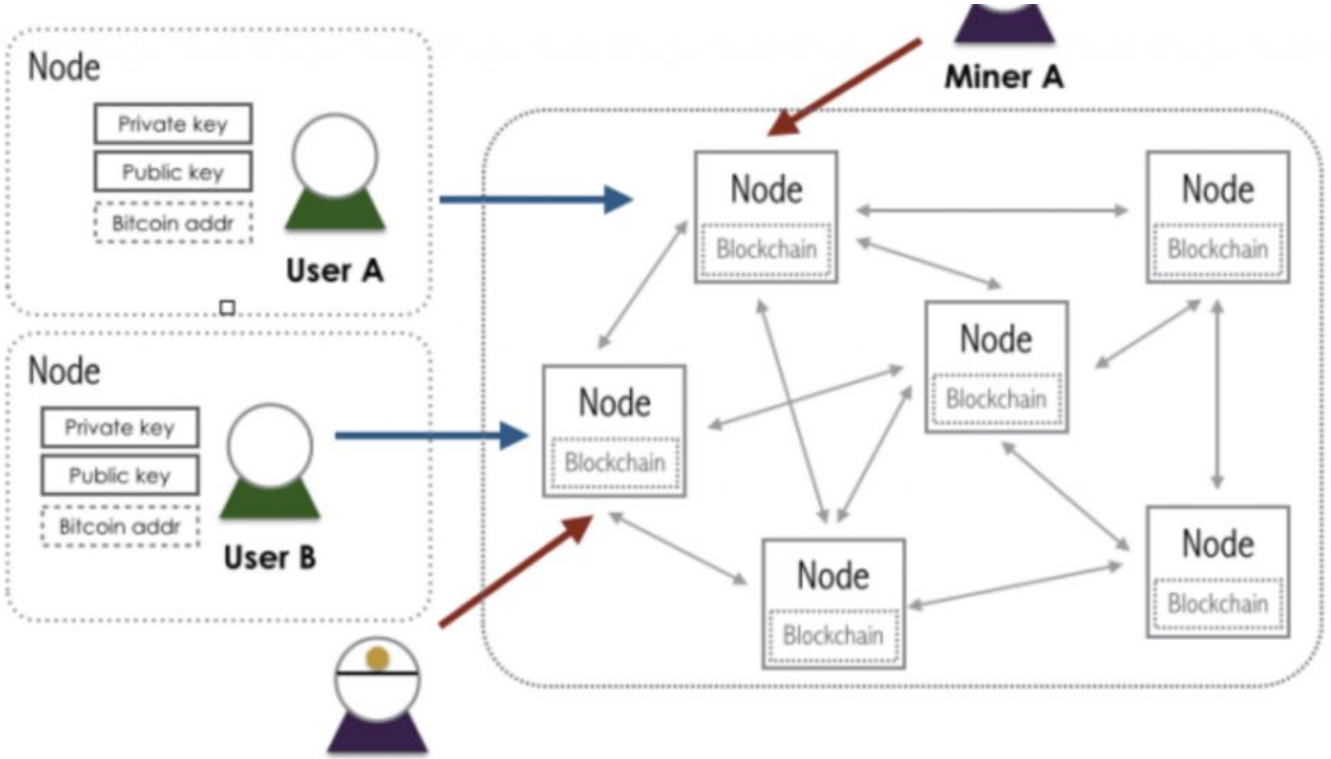
Private Blockchain

A private blockchain is a permissioned blockchain. Private blockchains work based on access controls which restrict the people who can participate in the network. There are one or more entities which control the network and this leads to reliance on third-parties to transact. In a private blockchain, only the entities participating in a transaction will have knowledge about it, whereas the others will not be able to access it. **Hyperledger Fabric of Linux Foundation is a perfect example of a private blockchain.**

Similarities Of Public And Private Blockchains

- **Both function as an append-only ledger** where the records can be added but cannot be altered or deleted. Hence, these are called immutable records.
- **Each network node in both these blockchains has a complete replica of the ledger.** Both are decentralized and distributed over a peer-to-peer network of computers.
- **In both, the validity of a record is verified**, thus providing a considerable level of immutability, until the majority of the participants agree that it is a valid record and reach consensus. This helps prevent tampering with the records.
- **Both blockchains rely on numerous users to authenticate edits** to the distributed ledger thus helping in the creation of a new master copy which can be accessed by everyone at all times.

Private Blockchain



source : fiberoptictel.com

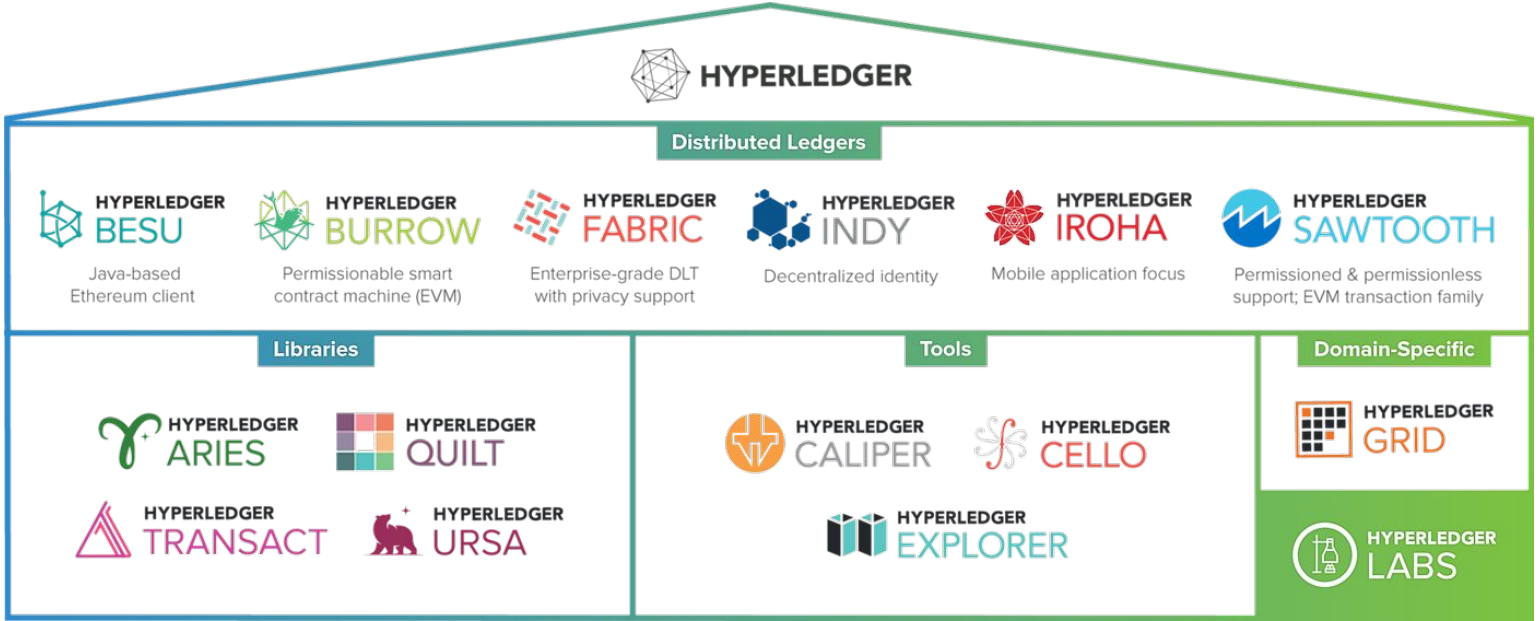
Public Vs. Private

	Public Blockchain	Private Blockchain
Access	Open read/write. Allows anyone to participate, execute contracts, run node, become a miner	Permissioned read/writes. All participants and nodes needs to be approved.
Consensus	Proof of Work (PoW), Proof of Stake (PoS), etc	Custom: PBFT (Practical Byzantine Fault Tolerant), multi-signature, etc
Currency	Mostly required. Used for transactions, rewarding miners and other utility (PoS)	Not required
Apps	Commonly known as dApps (decentralized Apps), they are decentralized, guarantee privacy and anonymity. Execution costs currency (to reward miners) and take time for data to verified across all nodes. Equivalent to open Internet.	Apps are built to custom business needs. Private Blockchain tech helps cut down processing times, less data redundancy and introduce efficiency between (or within) systems or organizations, and the same time restrict public access to sensitive data.
Examples	You can build something which serves everyone: from fun games like Crypto Kitties to important services like universal KYC (Civic), decentralized file storage (Storj, Filecoin), Anonymous SSO	Popular examples could be banks sharing a common ledger for fraud detection, international forex transactions, medical institutions storing and sharing patients health data with each other, etc.
Popular platforms	Ethereum, Bitcoin, NEO, Ripple, Stellar	Hyperledger, Corda, Multichain
Languages (Smart Contracts)	Ethereum: Solidity	Hyperledger: Chaincode
	NEO: VB.net, C#, Java, Python	Corda: Kotlin
Frameworks and Development Tools	Ethereum: Truffle, Metamask, Mist Wallet	Hyperledger: Hyperledger-Compose, Visual Studio
		Corda: Flow

source: medium.com

Hyperledger

- Hyperledger is an ecosystem and an umbrella project consisting of different open source Blockchain platforms that are hosted by the Linux Foundation .
- It is a collaborative project to host different applications such as financing, banking Internet of Things, supply chains, and others, and ensure transparency, immutability, longevity, and interoperability .

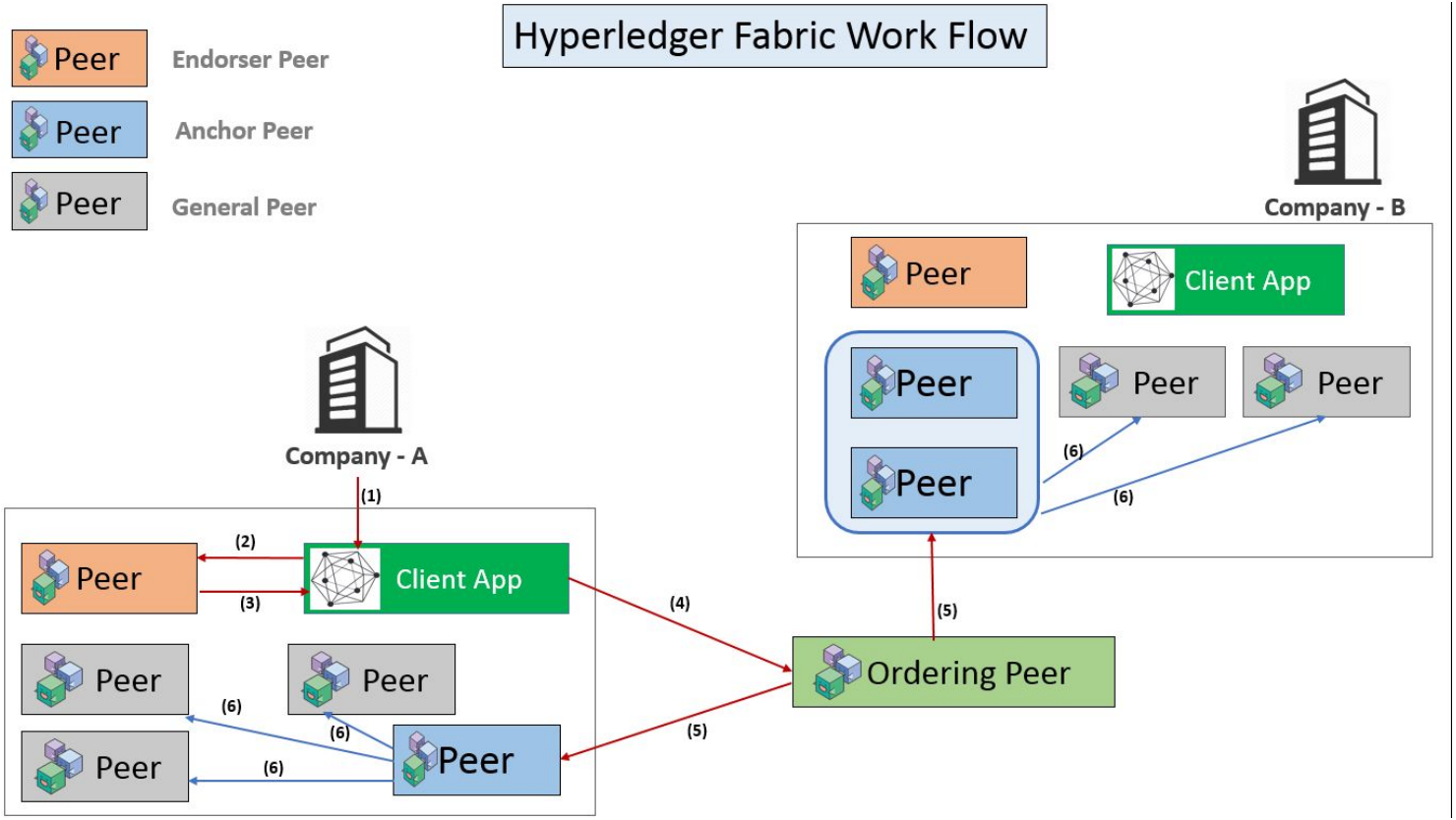


Hyperledger Fabric

Hyperledger Fabric is an open-source Blockchain platform which was proposed by IBM. The purpose of this platform is to overcome some limitations of other Blockchain platforms such as Ethereum and Tendermint. In particular, order-executive or hard-coded consensus are two shortcomings that affect the throughput and latency in other Blockchain platforms. The consensus algorithm in Hyperledger Fabric relies on Crash Fault Tolerant (CFT) which is based on Raft protocol.

- The Objective of Hyperledger is to create a platform which can be adapted to use cases in business
- Intended clients: Companies, Governments and their customers.

Hyperledger Fabric Architecture



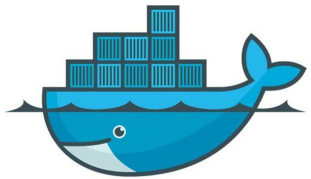
Complementary technologies in Fabric:

CouchDB

- records the state of chains
- uses Javascript for request processing

Docker Technology

- Chaincode Execution Environment
- Provides language-agnostic environment without bytecode
- The chaincodes have interaction through the Fabric SDK



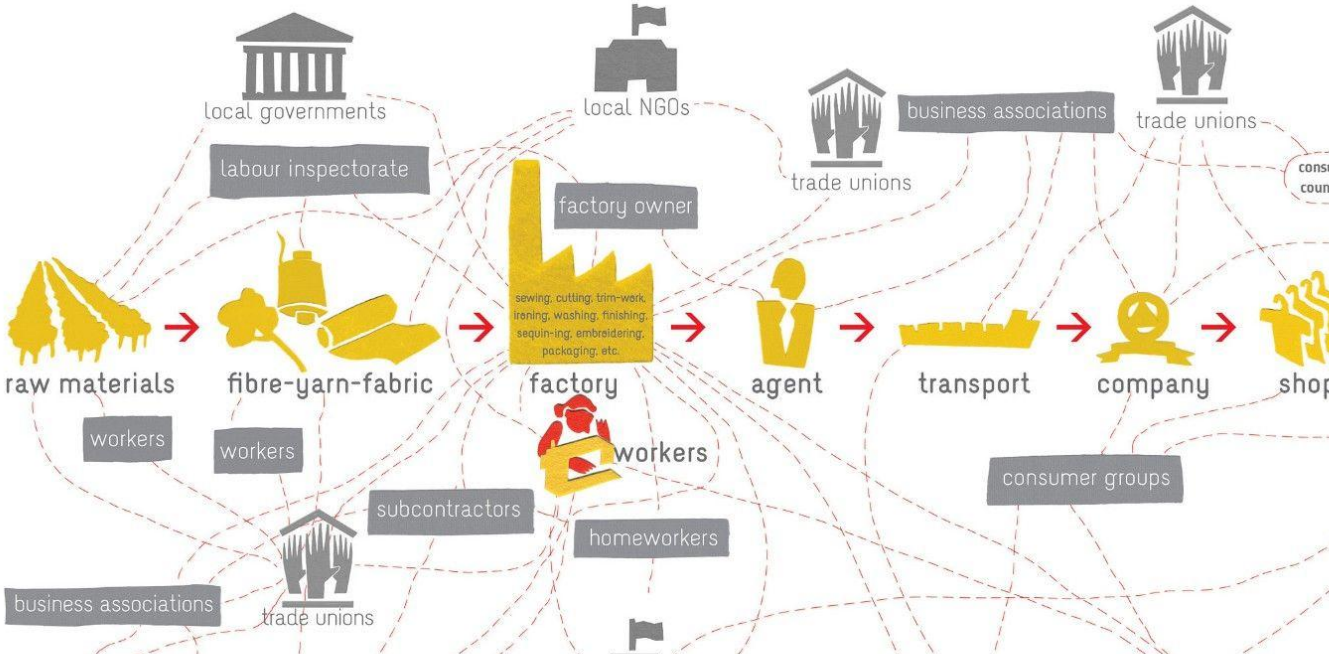
Comparison

Ser	Features	Bitcoin	Ethereum	Hyperledger-Fabric V 0.1
1.	Fully developed	✓	✓	✓
2.	Miner participation	Public	Public, Private, Hybrid	Private
3.	Trustless operation	✓	✓	Trusted validator nodes
4.	Multiple applications	Financial only	✓	✓
5.	Consensus	PoW	PoW, PoS ("Casper")	PBFT (Now changed to RAFT in v1.4)
6.	Consensus finality	X	X	✓
7.	Blockchain forks	✓	✓	X
8.	Fee less	X	X	Optional
9.	Run smart contracts	X	✓	✓
10.	TX integrity and authentication	✓	✓	✓
11.	Data Confidentiality	X	X	✓
12.	ID management	X	X	✓
13.	Key management	X	X	√ (through CA)
14.	User authentication	Digital Signatures	Digital Signatures	Based on enrolment certificates
15.	Device authentication	X	X	X
16.	Vulnerability to attacks	51%, linking attacks	51%	>1/3 faulty nodes
17.	TX throughput	7 TPS	8-9 TPS	>3500 TPS (depending upon number of endorsers, orderers and committers)
18.	Latency in single confirmation of a TX	10 min (60 min for a confirmed TX)	15–20 s	Less than Bitcoin, Ethereum & IOTA
19.	Is it Scalable?	X	X	X

source: Makhdom et al.,

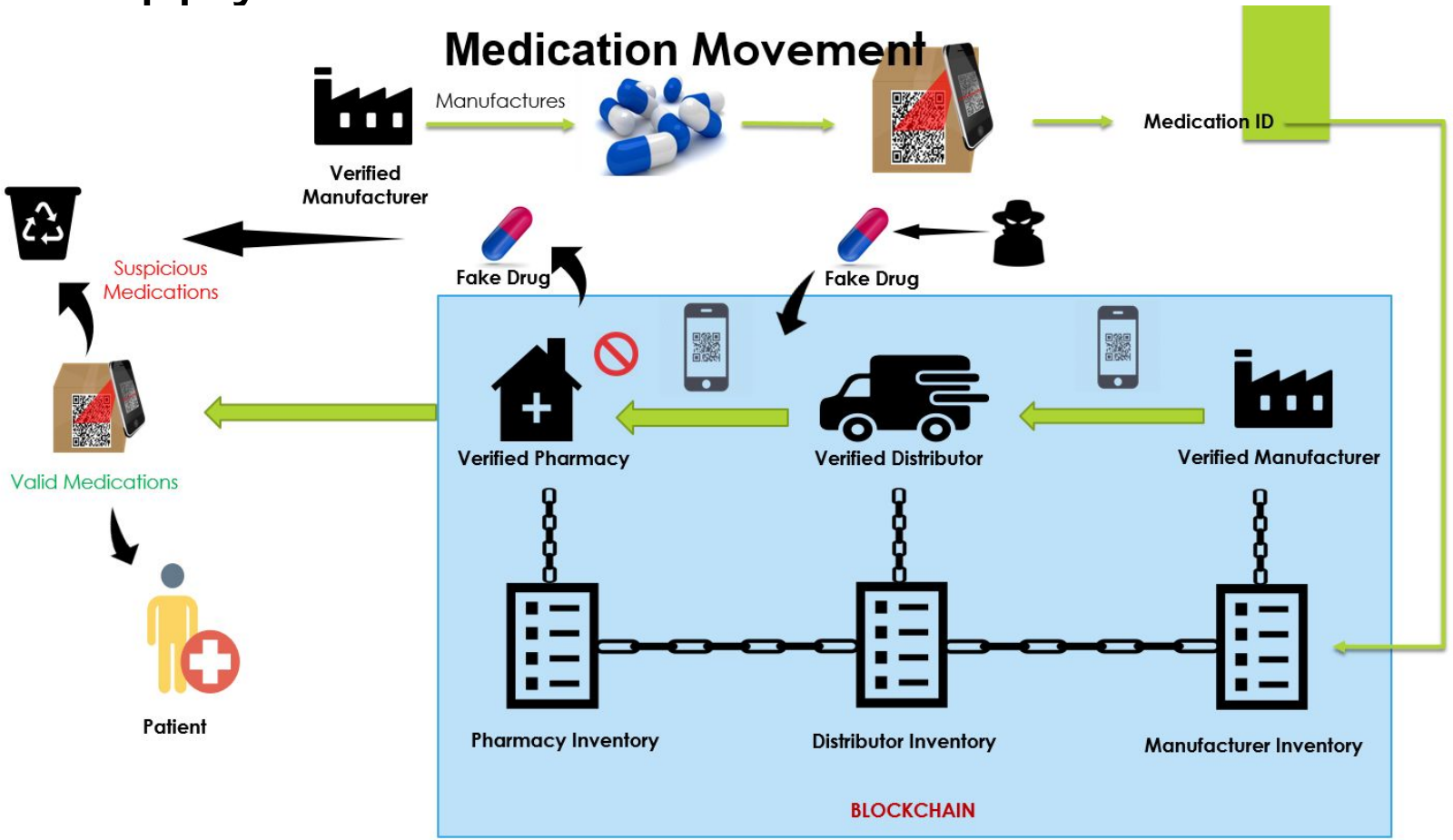
Fabric Use Case: Collaboration of Walmart and IBM

Fabric Supply-chain

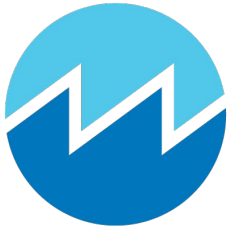


source:coletiv.com

Fabric Supply-chain



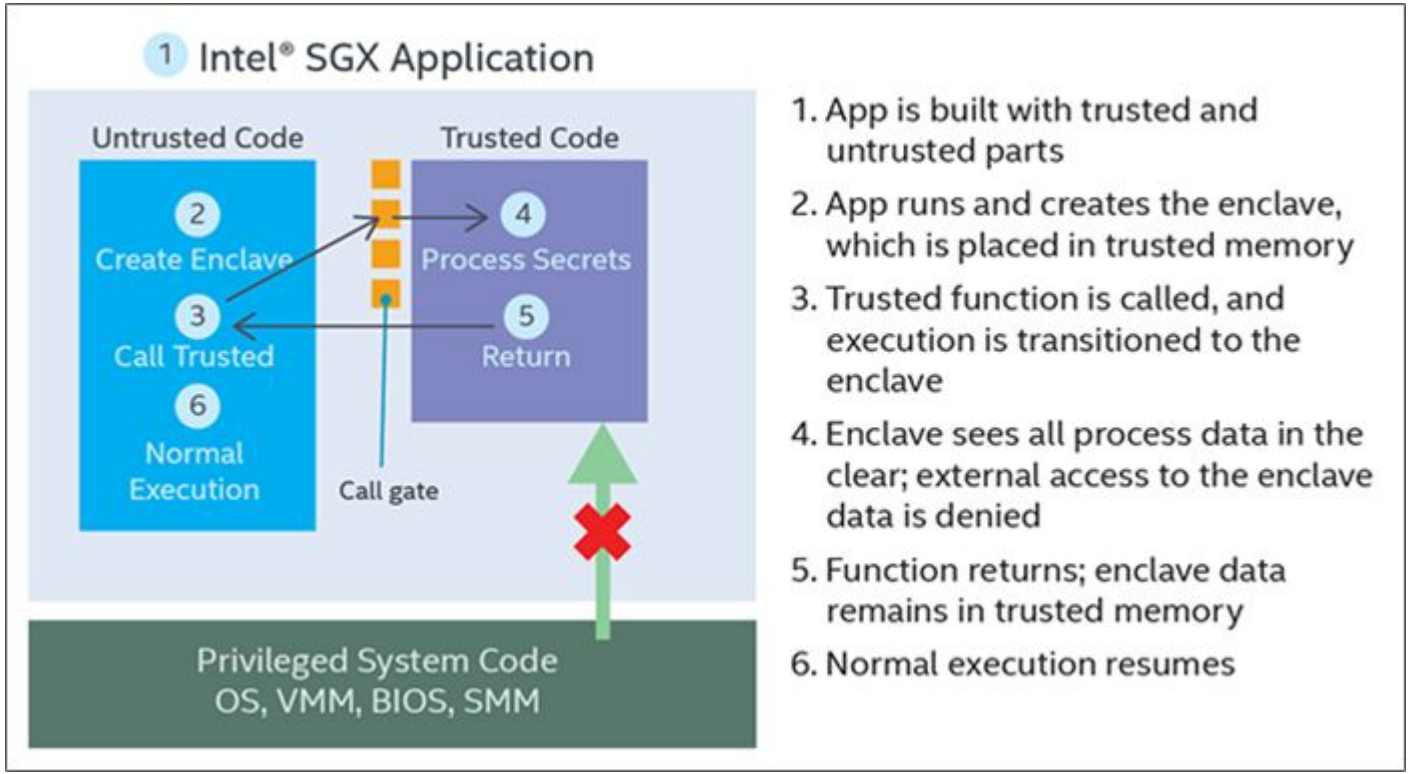
Hyperledger Sawtooth



HYPERLEDGER
SAWTOOTH

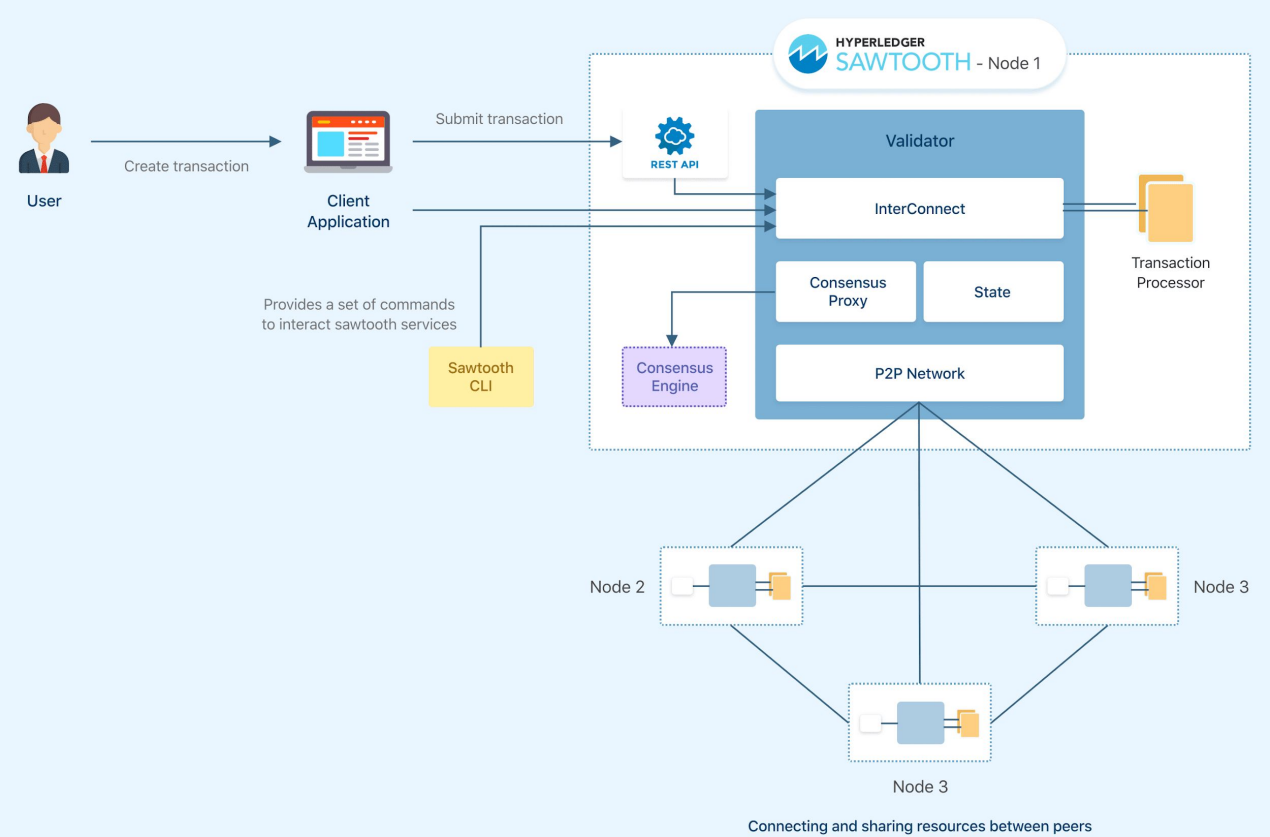
- Hyperledger Sawtooth is an open-source Blockchain platform which is designed especially for supplying chain management.
- It is part of the Linux Foundation umbrella project developed by Intel.
- The main difference of Sawtooth with other Blockchain platforms is that the transactions and data can be executed in parallel instead of in series, which will result in better performance of the system.
- It support dynamic consensus protocols including RAFT, PoET and PBFT
- Each validator in Sawtooth chooses a random time to response, the first to wake up "wins" and sends the block

Intel SGX Architecture



source: Intel

Sawtooth Architecture



source: hyperledger.org

References:

1) Antonopoulos, A. M. (2014). *Mastering Bitcoin: unlocking digital cryptocurrencies*. " O'Reilly Media, Inc.".
2) Chuen, D. L. K. (Ed.). (2015). Handbook of digital currency: Bitcoin, innovation, financial instruments, and big data. Academic Press.
3) Antonopoulos, A. M., & Wood, G. (2018). *Mastering ethereum: building smart contracts and dapps*. O'reilly Media.
4) Gaur, N., Desrosiers, L., Ramakrishna, V., Novotny, P., Baset, S. A., & O'Dowd, A. (2018). *Hands-on blockchain with hyperledger: building decentralized applications with hyperledger fabric and composer*. Packt Publishing Ltd.
5) Badr, B., Horrocks, R., & Wu, X. B. (2018). *Blockchain By Example: A developer's guide to creating decentralized applications using Bitcoin, Ethereum, and Hyperledger*. Packt Publishing Ltd.
6) Bashir, I. (2017). *Mastering blockchain*. Packt Publishing Ltd.
7) Baset, S. A., Desrosiers, L., Gaur, N., Novotny, P., O'Dowd, A., Ramakrishna, V., ... & Wu, X. B. (2019). *Blockchain Development with hyperledger: build decentralized applications with hyperledger fabric and composer*. Packt Publishing Ltd.\
8) Hafid, A. (2020). *Blockchain and its applications* [Lecture notes]. Retrieved from https://studium.umontreal.ca/course/view.php?id=169234
9) Makhdoom, I., Abolhasan, M., Abbas, H., & Ni, W. (2019). Blockchain's adoption in IoT: The challenges, and a way forward. *Journal of Network and Computer Applications*, *125*, 251-279.