UNIVERSITÀ DEGLI STUDI FIRENZE

DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

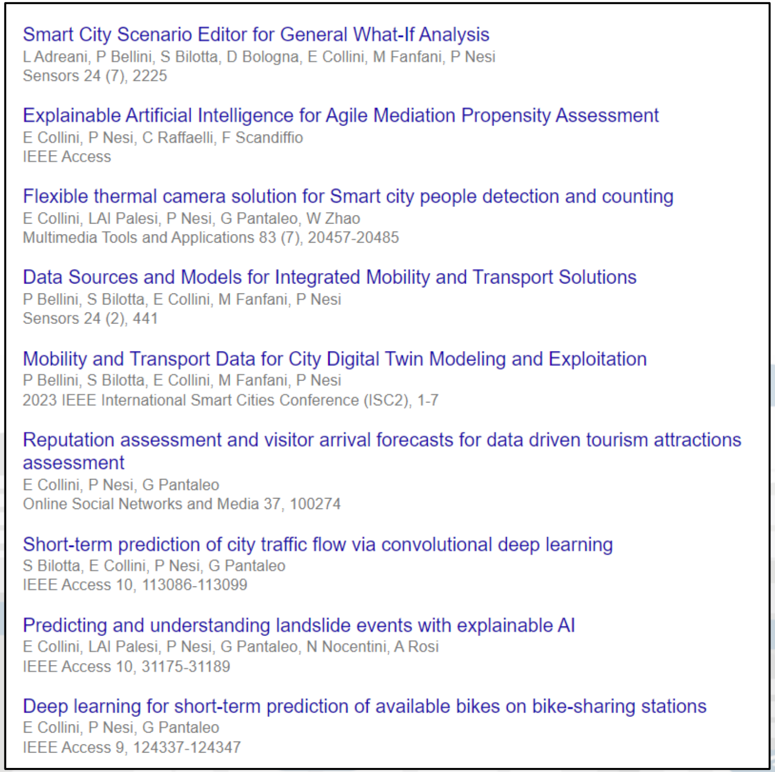
SNAP4CITY

KM4CITY

Blockchain and Internet of Things BIoT

Enrico Collini

AI Smart City solutions Research Fellow at Distributed System and Internet Technologies Lab (DISIT), Università Degli Studi di Firenze



Smart City Scenario Editor for General What-If Analysis
L Adreani, P Bellini, S Bilotta, D Bologna, E Collini, M Fanfani, P Nesi
Sensors 24 (7), 2225

Explainable Artificial Intelligence for Agile Mediation Propensity Assessment
E Collini, P Nesi, C Raffaelli, F Scandiffio
IEEE Access

Flexible thermal camera solution for Smart city people detection and counting
E Collini, LAI Palesi, P Nesi, G Pantaleo, W Zhao
Multimedia Tools and Applications 83 (7), 20457-20485

Data Sources and Models for Integrated Mobility and Transport Solutions
P Bellini, S Bilotta, E Collini, M Fanfani, P Nesi
Sensors 24 (2), 441

Mobility and Transport Data for City Digital Twin Modeling and Exploitation
P Bellini, S Bilotta, E Collini, M Fanfani, P Nesi
2023 IEEE International Smart Cities Conference (ISC2), 1-7

Reputation assessment and visitor arrival forecasts for data driven tourism attractions assessment
E Collini, P Nesi, G Pantaleo
Online Social Networks and Media 37, 100274

Short-term prediction of city traffic flow via convolutional deep learning
S Bilotta, E Collini, P Nesi, G Pantaleo
IEEE Access 10, 113086-113099

Predicting and understanding landslide events with explainable AI
E Collini, LAI Palesi, P Nesi, G Pantaleo, N Nocentini, A Rosi
IEEE Access 10, 31175-31189

Deep learning for short-term prediction of available bikes on bike-sharing stations
E Collini, P Nesi, G Pantaleo
IEEE Access 9, 124337-124347

UNIVERSITÀ DEGLI STUDI FIRENZE
DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB DISTRIBUTED DATA INTELLIGENCE AND TECHNOLOGIES LAB

UNIVERSITÀ DI PISA

DOTTORATO NAZIONALE IN INTELLIGENZA ARTIFICIALE

Slide presentazione Edoardo Branchi

Blockchain Introduction

The term blockchain holds multiple meanings, depending on the perspective.

- To developers, it refers to a **collection of protocols and encryption techniques that ensure secure data storage on a distributed network.**
- For business and finance, it's a **distributed ledger** and the technology behind the proliferation of digital currencies.
- Technologists see it as the **driving force behind the upcoming generation of the Internet**.
- To some, it represents a tool for profoundly **transforming society** and the economy, leading to a more **decentralized world**

Blockchain Introduction

- The term blockchain is captivating and fascinating because of its profound implications. For the first time in human history, **people can trust each other and engage in large peer-to-peer networks without centralized management**.

- This is made possible through the use of **protocols**, **cryptography**, and **computer code**, rather than *centralized institutions*.

- By relying on trust established through technology, we strengthen our ability to collaborate and cooperate within peer networks. This potentially enables us to form **global networks of collaboration without centralized institutions**, which is unprecedented but highly relevant in an age of globalization and new challenges that require mass collaboration.

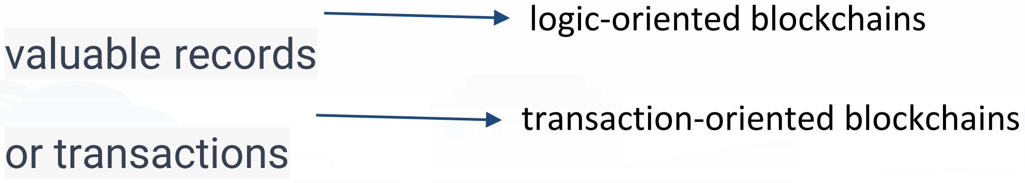
Today’s Lecture...

The goal of this lesson is to provide a comprehensive overview of blockchain technology, including its potential applications across various industries and its far-reaching implications for society and the economy.

- In the first section, we will provide an introduction to the **blockchain** from both technical and non-technical perspectives. We will also discuss the importance of the blockchain in the context of the emerging next-generation Internet.
- In the second section, we will introduce you to Hyperledger Fabric, an Open Source permissioned DLT developed by Linux Foundation. In the final section of the course, we will look at the integration between Fabric and Snap4city for data certification on the fabric blockchain"

BK core

Blockchain technology is designed to create a permanent and secure database, making it suitable for storing


valuable records

logic-oriented blockchains

or transactions

transaction-oriented blockchains

that require a high level of security and trust. These secure distributed records are called **distributed ledgers**. Essentially, ***a distributed ledger is a consensus of replicated, shared, and synchronized digital data that is spread out geographically across multiple sites, countries, or institutions without centralized administration or centralized data storage.***

Blockchain Basics

we're going to talk about the basics of the blockchain as the technology

- on its most basic level the blockchain can be understood as a **new kind of database** at least this was its original design what's different about this database is that it's **distributed**.
- Digital databases have been around for a while now but until recently they've been designed to centralize information on one computer or within one organization.
- The blockchain though uses a distributed network of computers to maintain a shared database the blockchain is then a set of protocols and encryption methods than enable a network of computers to work together in securely recording data within a shared open database.

MySQL

mongoDB

CouchDB

levelDB

Blockchain Basics

we're going to talk about the basics of the blockchain as the technology

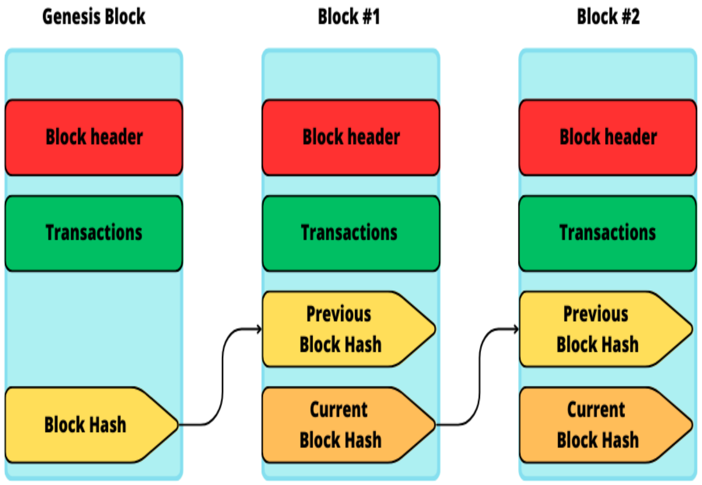
- on its most basic level the blockchain can be understood as a **new kind of database** at least this was its original design what's different about this database is that it's **distributed**.
- Digital databases have been around for a while now but until recently they've been designed to centralize information on one computer or within one organization.
- The blockchain though uses a distributed network of computers to maintain a shared database the blockchain is then a set of protocols and encryption methods than enable a network of computers to work together in securely recording data within a shared open database.

Block

- The blockchain is made up of a series of **blocks**…
- each of which contains a set of transactions… **data**
- **Each block is linked to the previous one**, creating an immutable chain of information.

Chain

Each block is encrypted and given a unique **hash** value, which is a code that represents the data within that block. The hashing process uses a standard algorithm to compress the data into a 64-character secure hash, which is unique to the document.



Hash

A hash function is a mathematical function that **takes in data** of any size and **produces a fixed-size output,** called a hash value or digest. The output is typically a sequence of letters and numbers that represents the input data in a unique and deterministic way.

The hash function works by taking the input data and running it through a series of complex mathematical operations. The output is a hash value that is **unique to the input data**, meaning that any change to the input data will result in a different hash value.

Hash functions are commonly used in cryptography for data integrity, authentication, and verification purposes.

For example, a file can be hashed to produce a hash value, and the hash value can be used to verify that the file has not been tampered with or corrupted.

Hash functions are also used in blockchain technology to create a secure and tamper-proof ledger of transactions. **Each block in the blockchain contains a hash value that is based on the previous block's hash value, creating a chain of blocks that are linked together.** This ensures that any attempt to alter or delete a block in the chain would require changing all subsequent blocks as well, making the blockchain resistant to tampering.

Consensus

- The decentralized nature of the blockchain means that there is no need for a central authority to verify transactions or maintain the database.
- Instead, the network is maintained by a large number of computers, also known as **nodes**, that communicate with each other and **work together** to validate transactions and add them to the blockchain.
- To ensure that all the nodes on the network **agree on the state of the database**, the blockchain uses a consensus mechanism.
- Consensus ensures that all nodes on the network agree on the state of the ledger, even if some nodes are malicious or fail

Proof-of-Work

- PoW is a method for validating transactions and achieving consensus in a blockchain network. It works by requiring network participants, called **miners**, to solve a complex mathematical problem in order to validate a block of transactions and add it to the blockchain.
- The solution, called a hash, is then verified by other nodes on the network, and if it is correct, the miner is rewarded with a certain amount of cryptocurrency.

Proof-of-Work Step by step

1) A miner selects a set of transactions to include in a new block and compiles them into a block.
2) The miner then applies a hash function, such as SHA-256, to the block's header, which contains metadata about the block, including a reference to the previous block's header, a timestamp, and a nonce (a random value added to the block header to create a unique hash).
3) The resulting hash is a long string of hexadecimal characters. The miner's goal is to find a hash value that meets a certain difficulty level set by the network, which typically requires the hash to start with a certain number of leading zeroes.
4) To find a hash that meets this difficulty level, the miner repeatedly hashes the block header with different nonce values until a hash is found that meets the difficulty requirement.
5) Once a valid hash is found, the miner broadcasts the block to the network for validation by other nodes. If the block is validated, it is added to the blockchain, and the miner receives a reward in cryptocurrency.

- The difficulty level of the problem is adjusted by the network every few minutes to ensure that the rate at which new blocks are added to the blockchain remains constant. This process is known as mining difficulty adjustment.

MEMPOOL

In a proof of work (PoW) blockchain network, all miners work on validating the same set of transactions. When a new transaction is broadcast to the network, it is added to a pool of unconfirmed transactions called the **mempool**. Miners select transactions from the mempool and add them to a new block that they are working on.

- When a user sends a transaction, it is broadcast to the network and initially added to the mempool of each node on the network.
- Nodes prioritize transactions based on a few factors, including the transaction fee, transaction size, and transaction age. Generally, transactions with higher fees are prioritized because miners are incentivized to select transactions that will earn them more fees.

CRYPTOGRAPHIC PUZZLE

- The selection of transactions to include in the block is not part of the cryptographic puzzle. Miners are free to select any set of unconfirmed transactions from the mempool to include in the block they are working on.
- When miners are creating a new block, they are tasked with solving a cryptographic puzzle that requires a significant amount of computing power. The puzzle is designed to be difficult to solve but easy to verify. The first miner to solve the puzzle and create a valid block is then rewarded with newly minted cryptocurrency and any transaction fees included in the block.
- The selection of transactions to include in the block is not part of the cryptographic puzzle.
- Miners are free to select any set of unconfirmed transactions from the mempool to include in the block they are working on.

Security by design

- **Blockchains are trying to create a secure trusted shared database** and do this through encryption and hashing proof of work and network consensus.
- The **hashing and linking of blocks makes it difficult to go back and change a previous block once it's entered** but this alone would not be suffice to ensure that the data is truly tamper proof so then the proof of work system intentionally makes it computationally more difficult to alter the database thus making it extremely difficult to alter all the blocks.
- On top of this it places that **distributed consensus mechanism** so that even if someone did manage to do this their record would not match that of others and thus would not be accepted as a valid record.
- So to successfully tamper with the blokchain you would need to alter all the blocks on the chain redo the proof of work for each block. To do this you wuould require more than 50% of the peer-to-peer network computing capability only then would your altar block become accepted by everyone else.

What is a blockchain?

- By definition a blockchain is a public, permanent, append-only-distributed ledger.
- With an underlying mathematical structure for storing data in a way that is nearly impossible to fake.

Who created the first blockchain?

“I’ve been working on a new electronic cash system that’s fully peer-to-peer, with no trusted third party.” These are the words of Satoshi Nakamoto, the mysterious creator of Bitcoin, in a message sent to a cryptography-focused mailing list in October 2008.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

What is a blockchain?

- By definition a blockchain is a public, permanent, append-only distributed ledger.
- With an unde in a way that is nearly impos

Who created the

“I’ve been workir peer-to-peer, with no trusted third p to, the mysterious creator of Bitcoir mailing list in October 2008.

Theories about Satoshi Nakamoto's true identity are numerous. No one knows whether he is a man, a woman, or whether it is more than one person. In Japanese, "satoshi" means "a clear, quick and wise thought." "Naka" can mean "medium," "inside," or "relationship." "Moto" can mean "origin" or "foundation." But it is uncertain whether these meanings are useful in tracing back to the person or group of people who invented the Bitcoin system.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

Double Spending Problem

The double spending problem is a challenge unique to digital currencies and transactions. It refers to the risk that a digital currency can be spent more than once.

Since digital information can be easily copied, someone could theoretically spend the same digital currency unit multiple times by sending identical transaction data to different recipients.

- In traditional financial systems, like with physical cash or centralized digital currencies (like those managed by banks), double spending isn't a concern because there's a central authority keeping track of who owns what.
- And these still suffer from the inherent weaknesses of the trust-based model with centralized authorities that represent single point of failure, prevent transparency, and determine higher costs of transactions.

Double Spending Problem

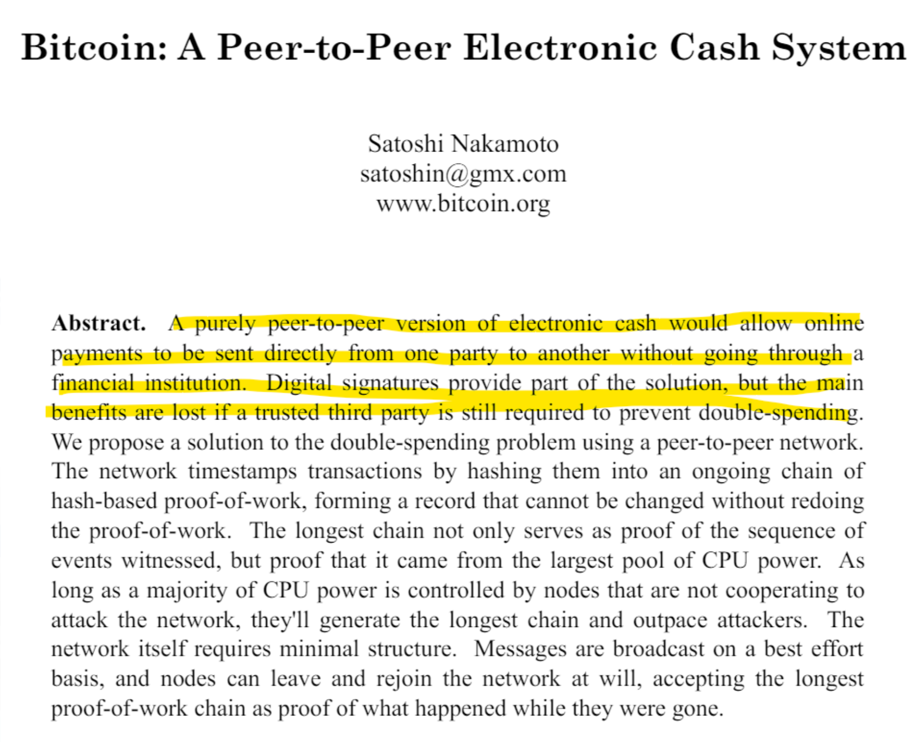
What is needed is an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third-party.

The idea is to design a decentralized ledger that records all transactions in chronological order and makes it extremely difficult to alter past transactions

- The solution proposed in the white paper of Nakamoto consisted in a peer-to-peer distributed timestamp server that generates computational proof of the chronological order of transactions.
- The system could be considered secure as long as honest nodes collectively control more CPU power than any cooperating group of attacker nodes.

The blockchain is a new class of information technology that combines **cryptography** and **distributed computing** to create a model where a network of computers collaborates toward maintaining a **shared and secure database**.

Peer-to-peer version of electronic cash



Bitcoin: A Peer-to-Peer Electronic Cash System

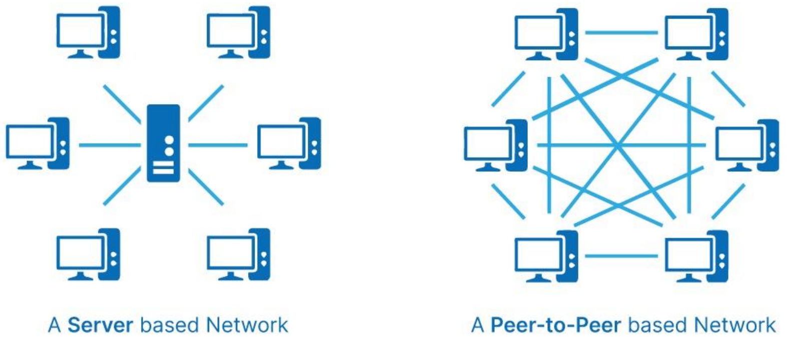
Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

Digital signature can provide technical solution to prove who owns what.

With my private key I can sign a piece of information,
Give it to you,
And anyone can verify that i've generated the piece of information checking with my public key.

Peer to peer network

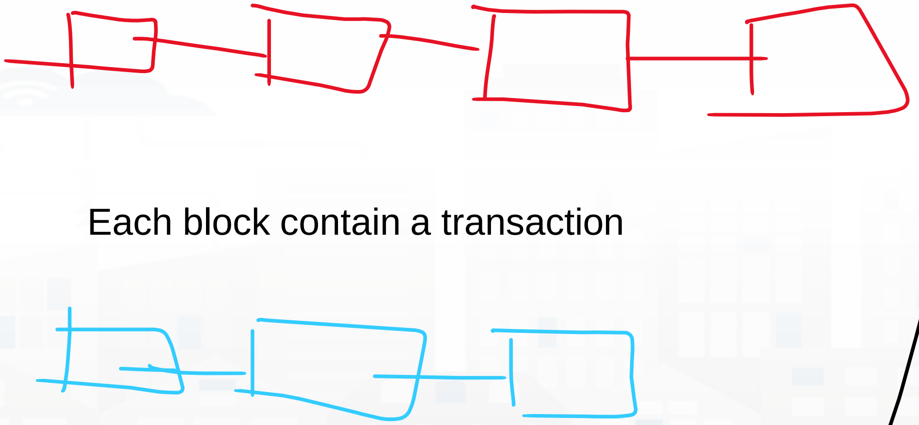


In a peer-to-peer system, it is possible to send files from one computer to another without the use of a centralized server, a centralized storing database a centralized verification mechanism.

The network timestamps transactions by hashing them ensuring data integrity and generating a unique chain of signatures. This forms a record that cannot be changed without redoing the proof of work for all the transaction.

Cryptographic proof replaces centralized trust.

The longest blockchain will always win.
If we have two competing versions of blockchain the longest will be the one accepted.



Each block contain a transaction

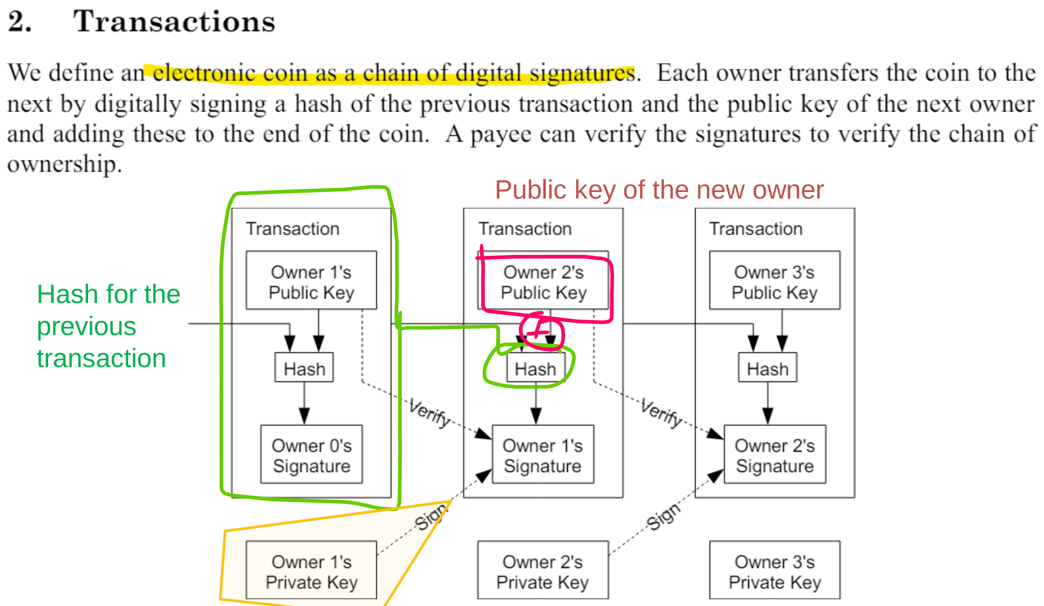
Both claim they are the real blockchain

The network will reject the blue one and accept the red one because is longer.

As long as the most CPU power is controlled by honest nodes the red chain will be mined faster.
The dishonest nodes will never be able to catch up and make their blockchain long enogh.
The **51% attack** …

Transactions

- A transaction in the context of blockchain refers to the transfer of assets, data, or information from one participant to another within the blockchain network.
- It represents a record of an exchange or interaction between parties, such as the transfer of cryptocurrencies (e.g., Bitcoin, Ethereum), digital assets (e.g., tokens).



2. Transactions

We define an electronic coin as a chain of digital signatures. Each owner transfers the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of the coin. A payee can verify the signatures to verify the chain of ownership.

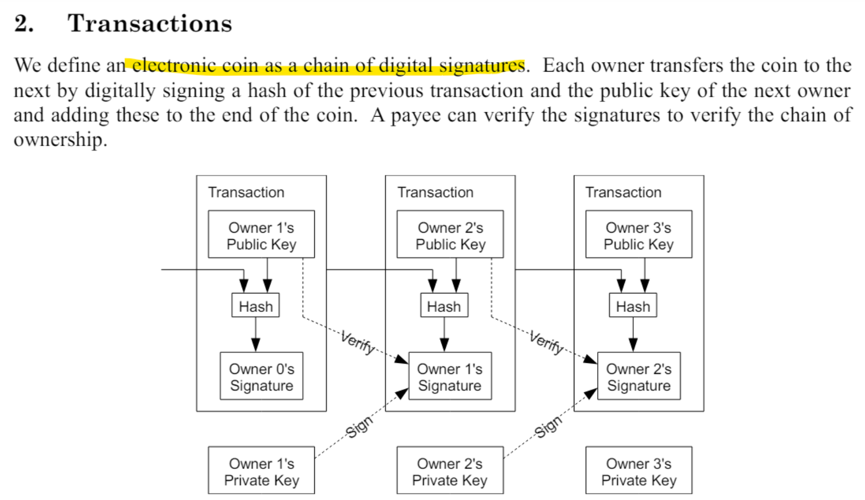
Public key of the new owner

Hash for the previous transaction

Get signed by the old owner

So that anyone can verify via the old owner public key the successful transaction

Transactions



2. Transactions

We define an electronic coin as a chain of digital signatures. Each owner transfers the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of the coin. A payee can verify the signatures to verify the chain of ownership.

Transaction
Owner 1's Public Key
Hash
Owner 0's Signature
Transaction
Owner 2's Public Key
Hash
Owner 1's Signature
Transaction
Owner 3's Public Key
Hash
Owner 2's Signature
Verify
Verify
Sign
Sign
Owner 1's Private Key
Owner 2's Private Key
Owner 3's Private Key

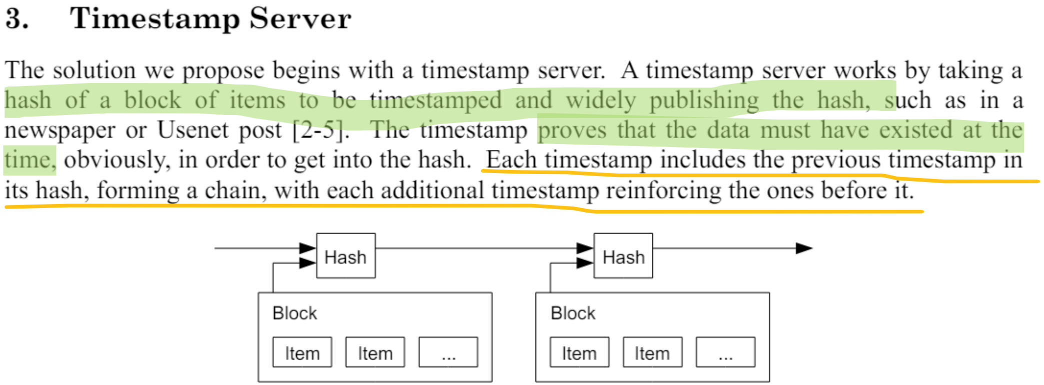
Nothing new so far: digital signatures…

This do not solve the double spending problem.

To accomplish this without a trusted party, transactions must be publicly announced, and we need a system for participants to agree on **a single history** of the order in which they were received. The payee needs proof that at the time of each transaction, the majority of nodes agreed it was the first received.

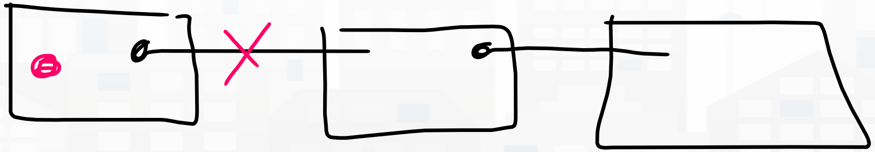
How do you do that?

- You have a network of nodes.
- These nodes all cooperate in order to have a single record of transactions and a single distributed ledger



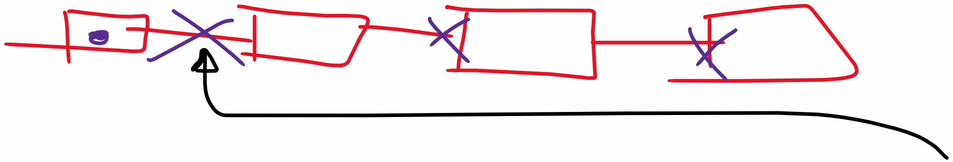
3. Timestamp Server

The solution we propose begins with a timestamp server. A timestamp server works by taking a hash of a block of items to be timestamped and widely publishing the hash, such as in a newspaper or Usenet post [2-5]. The timestamp proves that the data must have existed at the time, obviously, in order to get into the hash. Each timestamp includes the previous timestamp in its hash, forming a chain, with each additional timestamp reinforcing the ones before it.



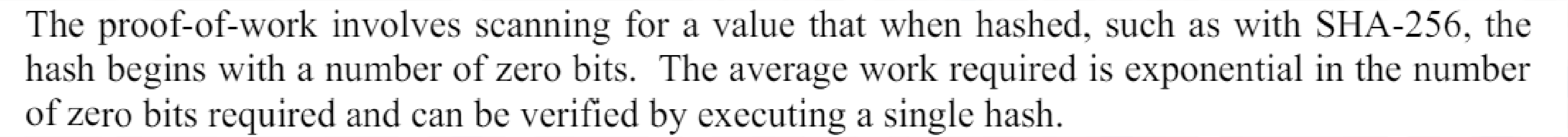
The blocks are criptographycally linked together.
The identity of the next block depens on the previous one
If you change something in the first block will beak the linkage to the second one and you would need to remine everything

Proof of work



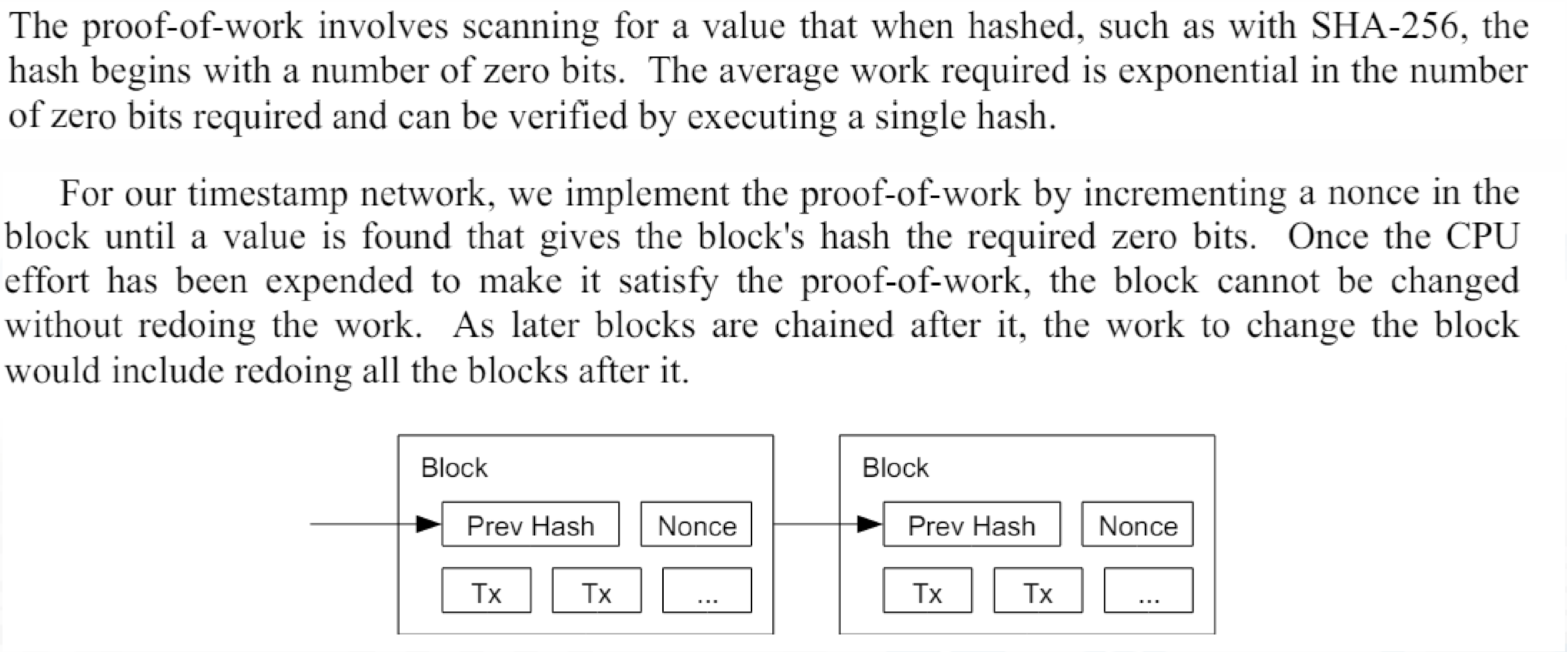
The proof of work determine that the creation of the connection is computationally expensive in terms of time required to hash the block.

If you modify something in the past, you break the links it would be hard to recreate the linking of the block to pesent a «new valid blockchain»



The proof-of-work involves scanning for a value that when hashed, such as with SHA-256, the hash begins with a number of zero bits. The average work required is exponential in the number of zero bits required and can be verified by executing a single hash.

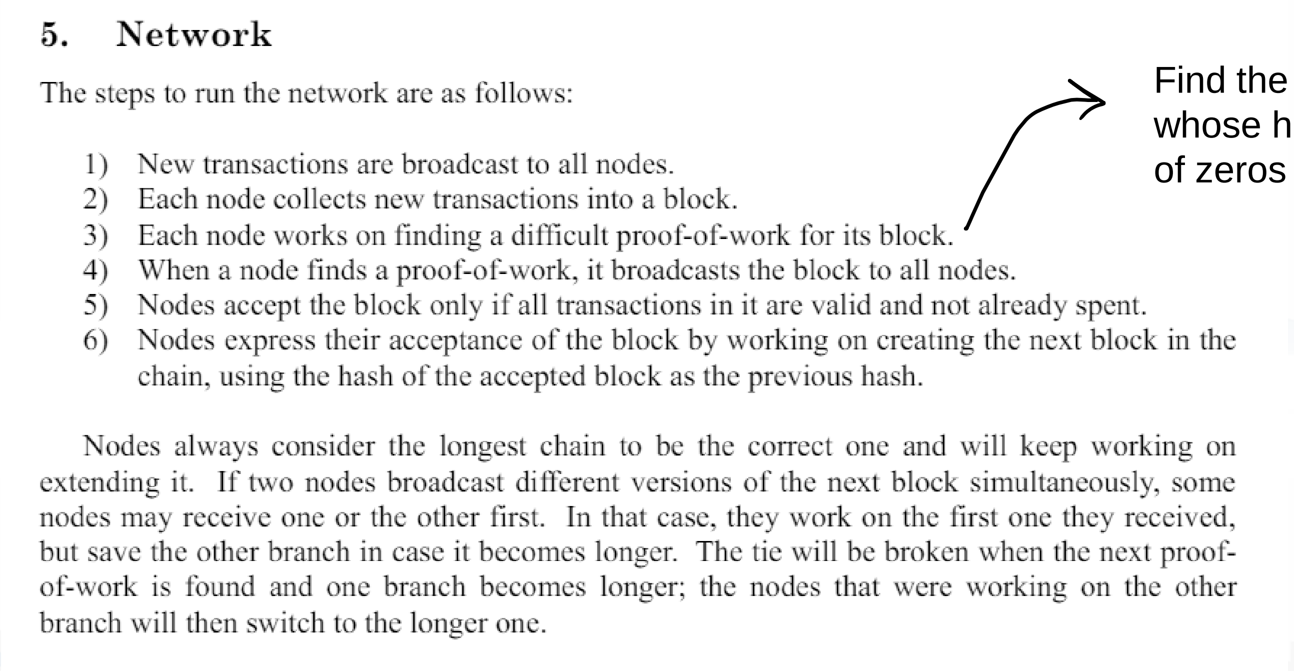
Proof of work



The proof-of-work involves scanning for a value that when hashed, such as with SHA-256, the hash begins with a number of zero bits. The average work required is exponential in the number of zero bits required and can be verified by executing a single hash.

For our timestamp network, we implement the proof-of-work by incrementing a nonce in the block until a value is found that gives the block's hash the required zero bits. Once the CPU effort has been expended to make it satisfy the proof-of-work, the block cannot be changed without redoing the work. As later blocks are chained after it, the work to change the block would include redoing all the blocks after it.

https://andersbrownworth.com/blockchain/blockchain



5. Network

The steps to run the network are as follows:

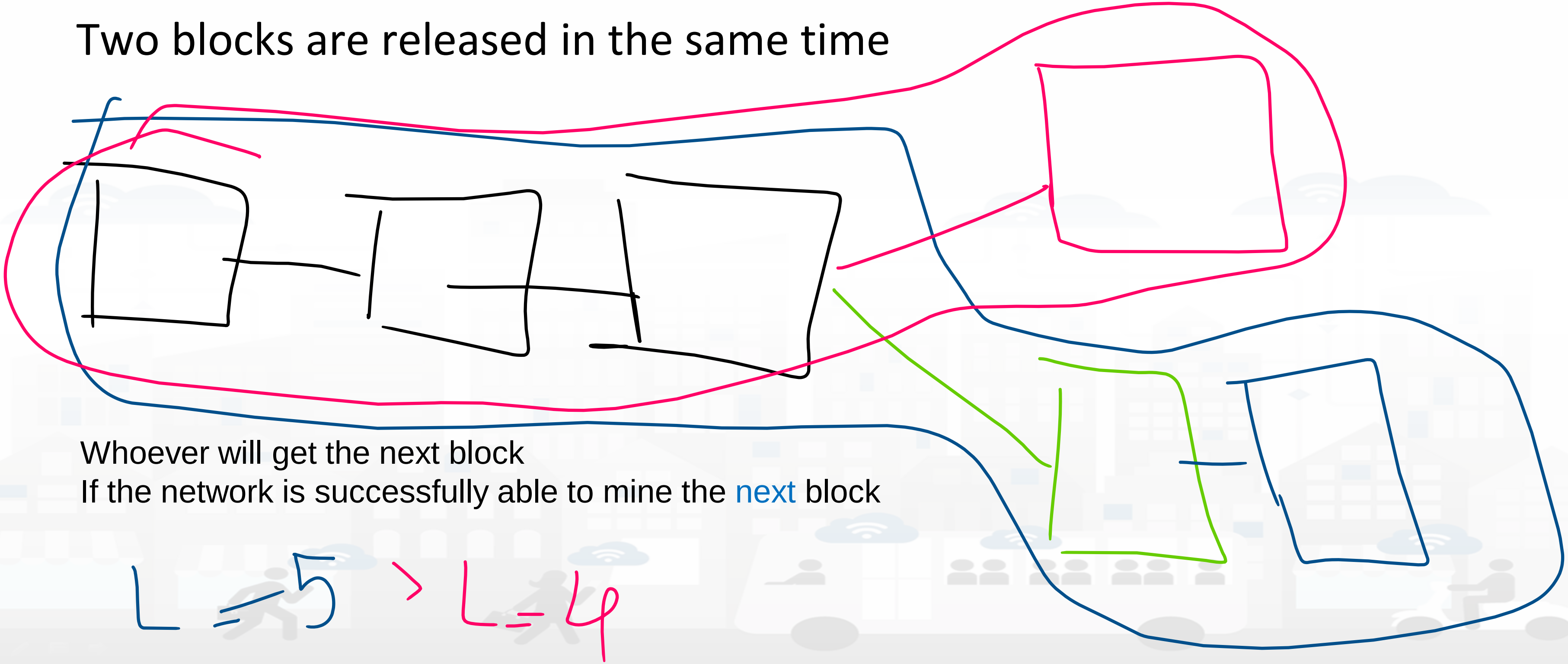
1) New transactions are broadcast to all nodes.
2) Each node collects new transactions into a block.
3) Each node works on finding a difficult proof-of-work for its block.
4) When a node finds a proof-of-work, it broadcasts the block to all nodes.
5) Nodes accept the block only if all transactions in it are valid and not already spent.
6) 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.

Nodes always consider the longest chain to be the correct one and will keep working on extending it. If two nodes broadcast different versions of the next block simultaneously, some nodes may receive one or the other first. In that case, they work on the first one they received, but save the other branch in case it becomes longer. The tie will be broken when the next proof-of-work is found and one branch becomes longer; the nodes that were working on the other branch will then switch to the longer one.

Find the correct nonce whose hash has that number of zeros

What happens when

Two blocks are released in the same time



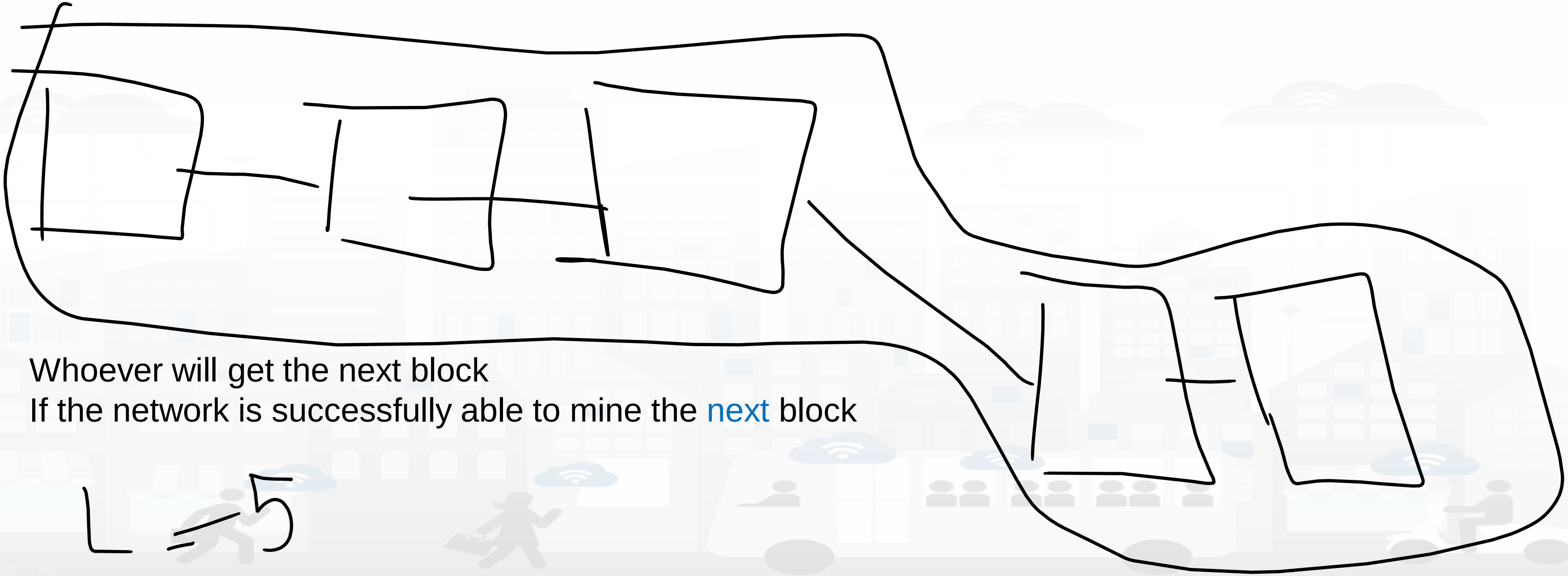
Whoever will get the next block

If the network is successfully able to mine the next block

L = 5 > L = 4

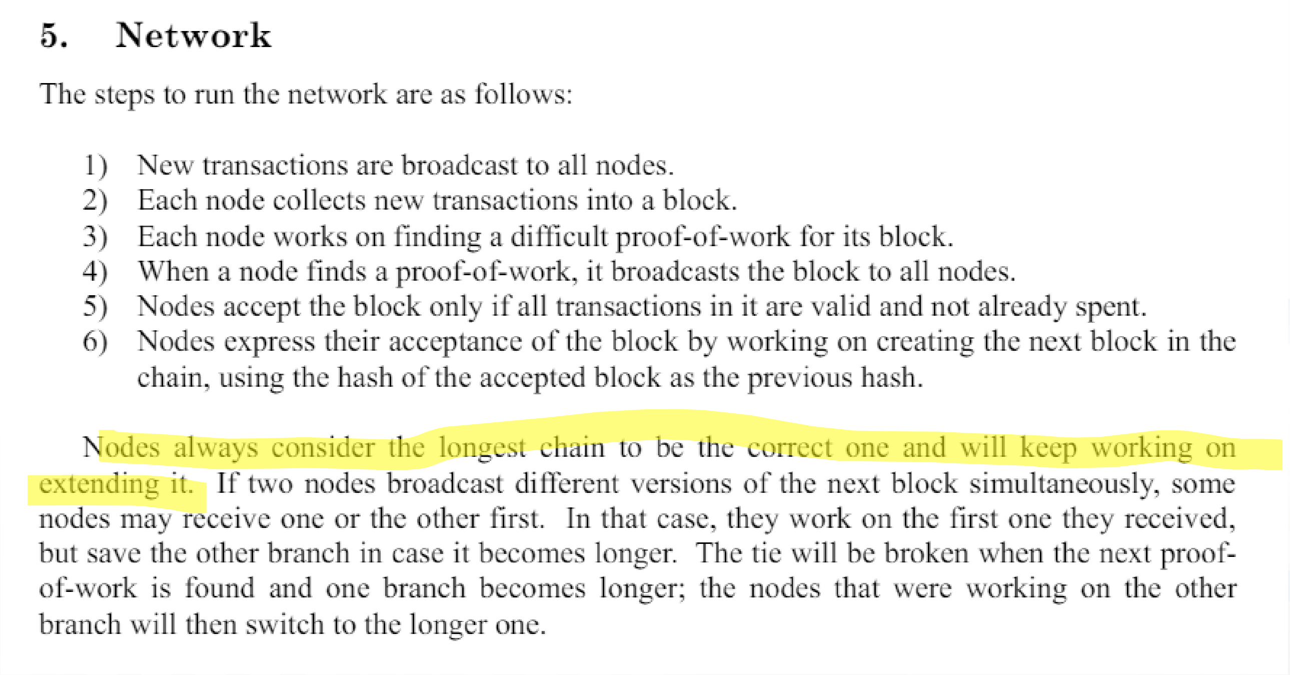
What happens when

Two blocks are released in the same time



Whoever will get the next block
If the network is successfully able to mine the next block

L = 5



5. Network

The steps to run the network are as follows:

1) New transactions are broadcast to all nodes.
2) Each node collects new transactions into a block.
3) Each node works on finding a difficult proof-of-work for its block.
4) When a node finds a proof-of-work, it broadcasts the block to all nodes.
5) Nodes accept the block only if all transactions in it are valid and not already spent.
6) 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.

Nodes always consider the longest chain to be the correct one and will keep working on extending it. If two nodes broadcast different versions of the next block simultaneously, some nodes may receive one or the other first. In that case, they work on the first one they received, but save the other branch in case it becomes longer. The tie will be broken when the next proof-of-work is found and one branch becomes longer; the nodes that were working on the other branch will then switch to the longer one.

Incentive

By convention, the first transaction in a block is a special transaction that starts a new coin owned by the creator of the block.

➔ Incentive for nodes to support the network

➔ Provides a way to initially distribute coins into circulation since there is no central authority to issue them

➔ Because "mining" costs CPU time and electricity

If we google a random block explorer …

The incentive may help encourage nodes to stay honest. If a greedy attacker is able to assemble more CPU power than all the honest nodes, he would have to choose between using it to defraud people by stealing back his payments, or using it to generate new coins.

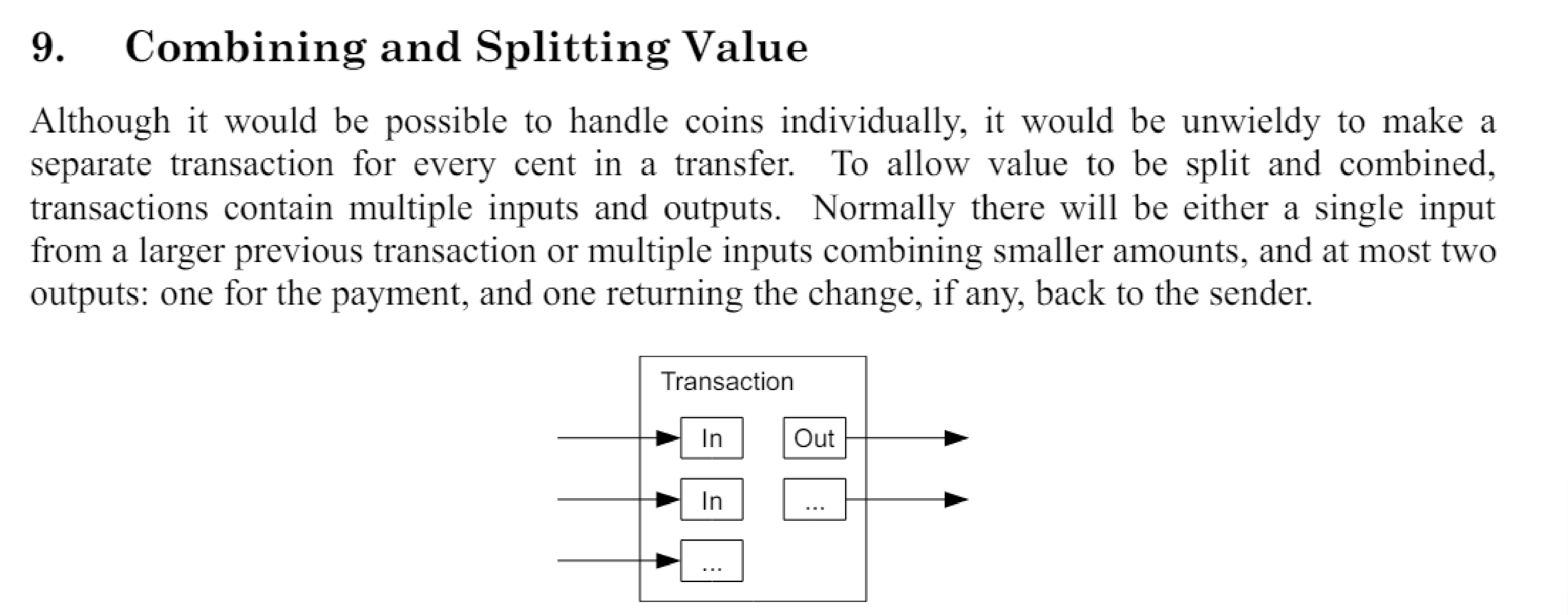
SPV

- SPV stands for Simple Payment Verification
- It is possible to verify payments without running a full network node
- You really only need a portion of information to be able to verify the transactions.
- Query the network for the current longest proof-of-work chain
- Confirm it is the current one by checking if a the next query to that chain has been added a new block
- As long as honest nodes control > 51% of the network the verification is reliable

The change

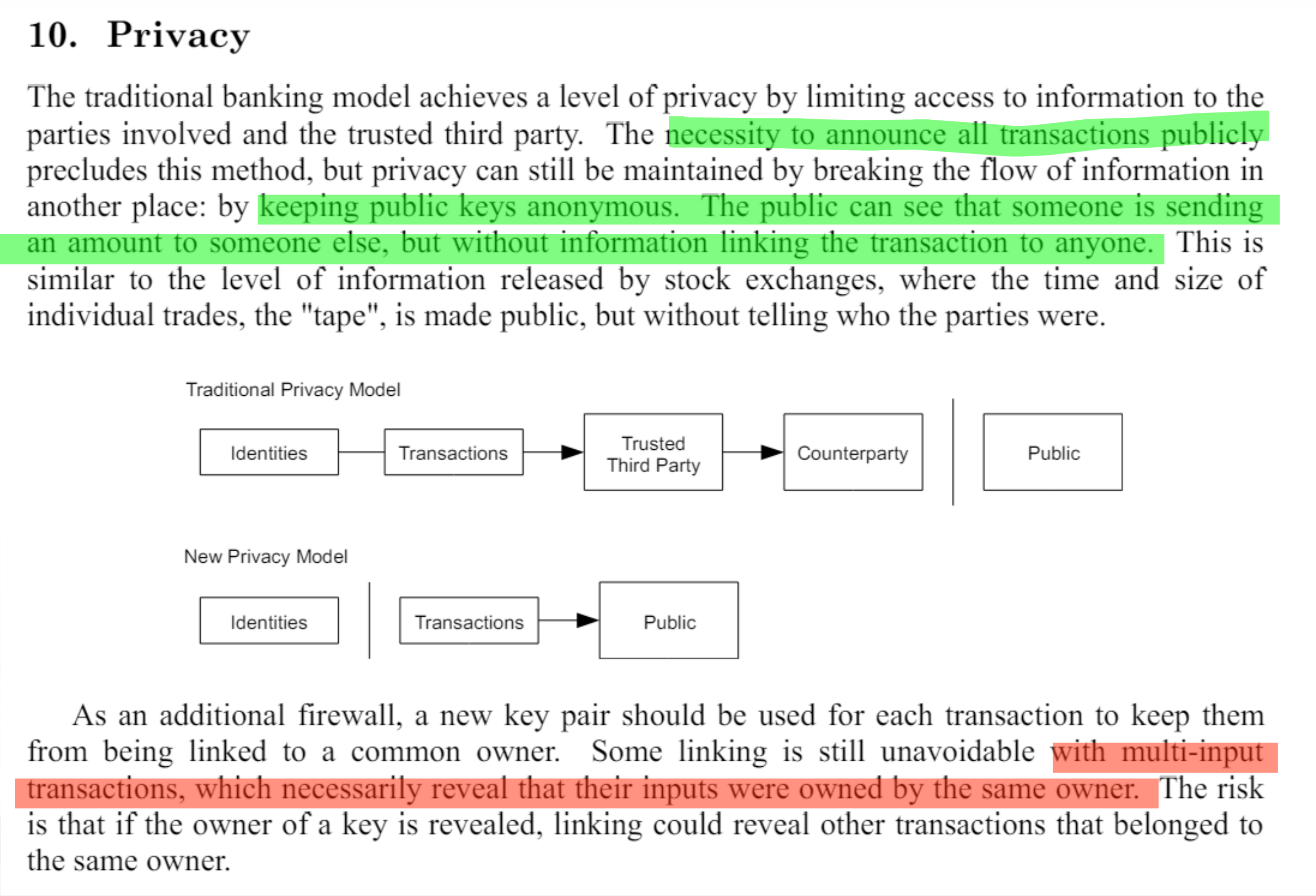
?

The change



9. Combining and Splitting Value

Although it would be possible to handle coins individually, it would be unwieldy to make a separate transaction for every cent in a transfer. To allow value to be split and combined, transactions contain multiple inputs and outputs. Normally there will be either a single input from a larger previous transaction or multiple inputs combining smaller amounts, and at most two outputs: one for the payment, and one returning the change, if any, back to the sender.



10. Privacy

The traditional banking model achieves a level of privacy by limiting access to information to the parties involved and the trusted third party. The necessity to announce all transactions publicly precludes this method, but privacy can still be maintained by breaking the flow of information in another place: by keeping public keys anonymous. The public can see that someone is sending an amount to someone else, but without information linking the transaction to anyone. This is similar to the level of information released by stock exchanges, where the time and size of individual trades, the "tape", is made public, but without telling who the parties were.

As an additional firewall, a new key pair should be used for each transaction to keep them from being linked to a common owner. Some linking is still unavoidable with multi-input transactions, which necessarily reveal that their inputs were owned by the same owner. The risk is that if the owner of a key is revealed, linking could reveal other transactions that belonged to the same owner.

The system is not open to arbitrary changes

An attacker could try to

- create value out of thin air
- or take money that never belonged to him

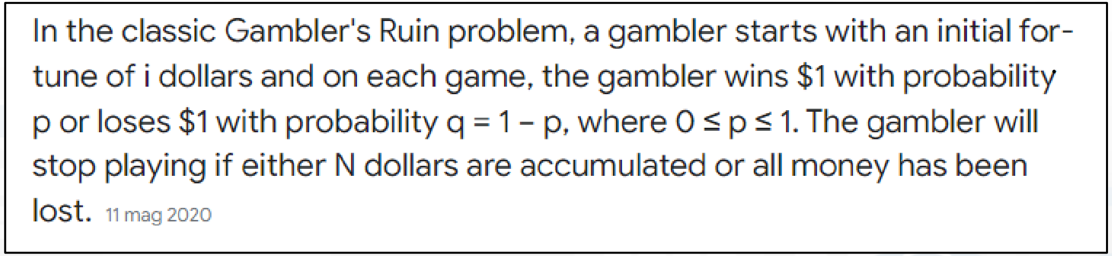
However

- Nodes are not going to accept an invalid transaction as payment
- honest nodes will never accept a block containing them.

An attacker can only try to change one of his own transactions to take back money he recently spent.

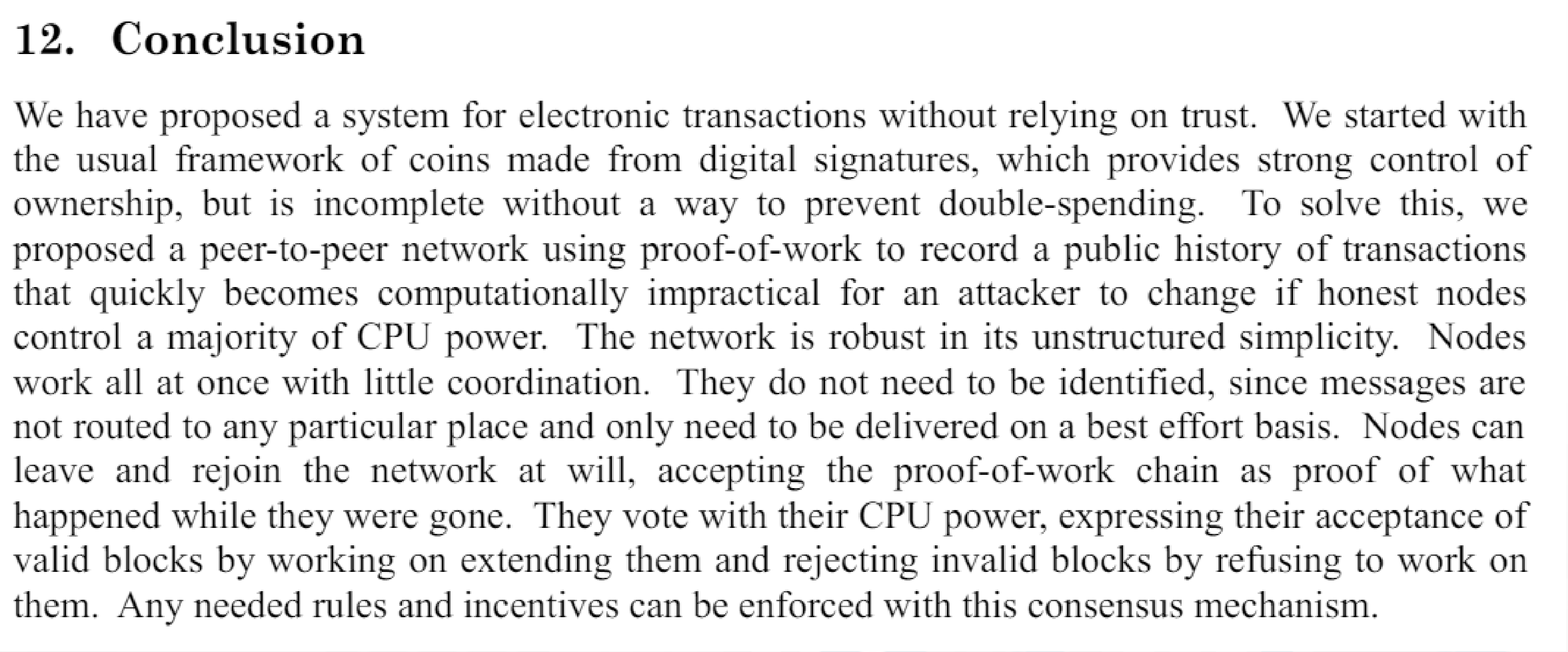
Race between the Honest Chain and the Attacker Chain

- The competition between the honest chain and the attacker's chain is described as a "Binomial Random Walk.“
- The probability for the honest chain to extend one block is "p“
- while that for the attacker's chain is "q".
- The probability of an attacker recovering a disadvantage of "z" blocks is analogous to the "Gambler's Ruin" problem.



In the classic Gambler's Ruin problem, a gambler starts with an initial fortune of i dollars and on each game, the gambler wins \$1 with probability p or loses \$1 with probability q = 1 – p, where 0 ≤ p ≤ 1. The gambler will stop playing if either N dollars are accumulated or all money has been lost. 11 mag 2020

Is it possible to determine the probability of success of a possible attack?



12. Conclusion

We have proposed a system for electronic transactions without relying on trust. We started with the usual framework of coins made from digital signatures, which provides strong control of ownership, but is incomplete without a way to prevent double-spending. To solve this, we proposed a peer-to-peer network using proof-of-work to record a public history of transactions that quickly becomes computationally impractical for an attacker to change if honest nodes control a majority of CPU power. The network is robust in its unstructured simplicity. Nodes work all at once with little coordination. They do not need to be identified, since messages are not routed to any particular place and only need to be delivered on a best effort basis. Nodes can leave and rejoin the network at will, accepting the proof-of-work chain as proof of what happened while they were gone. They vote with their CPU power, expressing their acceptance of valid blocks by working on extending them and rejecting invalid blocks by refusing to work on them. Any needed rules and incentives can be enforced with this consensus mechanism.

Hands On?

https://github.com/butikofer/coding_a_basic_blockchain

The concept of Web 3.0 has been discussed for some time, but it is only with the development of blockchain technology that it is becoming a reality. The centralization of Web 2.0 around large platforms and data centers has created numerous issues related to security, privacy, control, and concentration of power in the hands of large enterprises.

Blockchain technology provides the protocols and cryptography for a **globally distributed network** of computers to collaborate on maintaining a public, secure database. With a virtual machine like Ethereum, we can run code on this network, creating a new set of distributed applications.

In web 3 machines will come online and the internet will become something much more physical as billions of **devices and actuators connected to all sorts of things** from tractors to watches to factories and drones **enabling them to interact and coordinate** machine to machine the value of the Internet of Things.

these technologies will have to communicate securely, dynamically allocate resources, and operate within a distributed secure infrastructure, such as the blockchain.

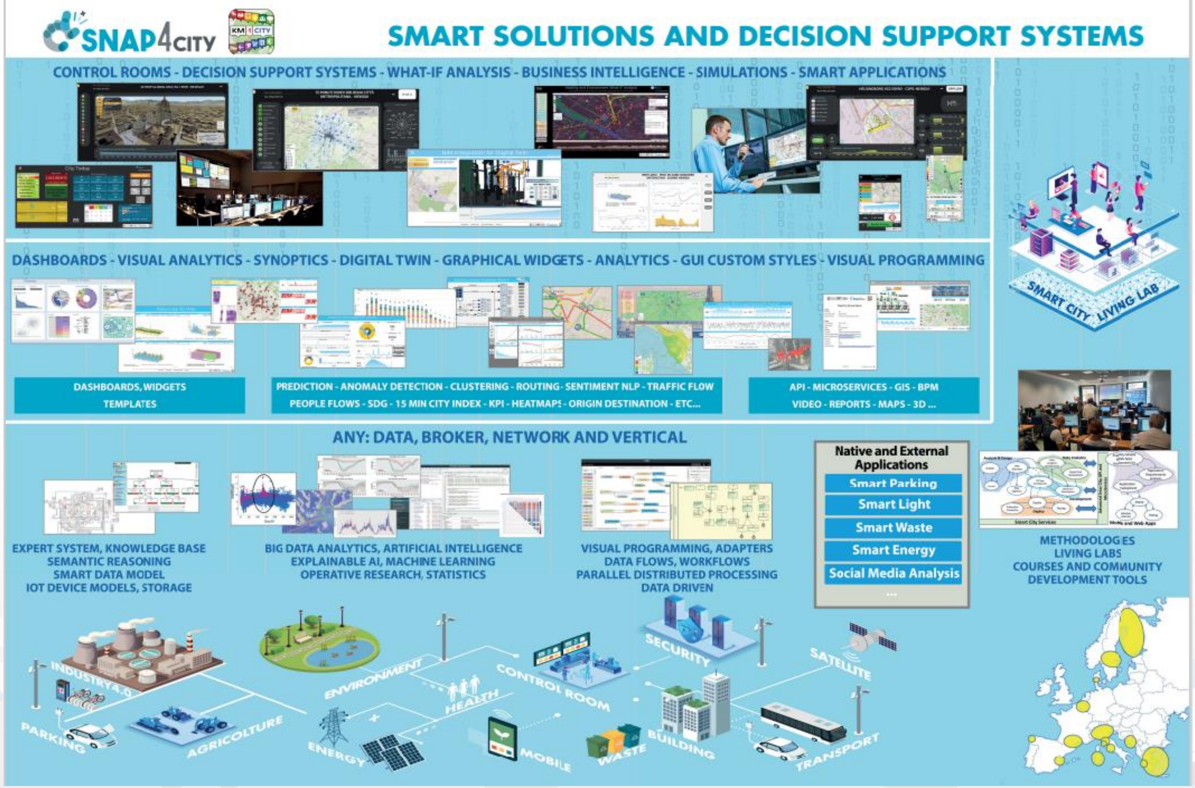
7 IoT Problems

- **Inadequate Authentication and Authorization**: Many IoT devices and systems lack proper authentication and authorization mechanisms, making them vulnerable to unauthorized access and control.
- **Lack of Standardization**: IoT protocols and technologies makes it difficult to ensure compatibility and interoperability between devices and systems.
- **Complexity**: The complexity of IoT systems makes them vulnerable to attacks that exploit vulnerabilities in the software and hardware components.
- **Privacy Concerns**: IoT devices collect and transmit large amounts of personal data, raising concerns about privacy and data protection.
- **Inadequate Security Controls**: Many IoT devices and systems lack adequate security controls, making them vulnerable to attacks that exploit software and hardware vulnerabilities.
- **Limited Update and Patching**: IoT devices and systems may not receive timely updates and patches, leaving them vulnerable to attacks that exploit known vulnerabilities.
- **Physical Security:** IoT devices may be physically vulnerable to attacks, such as tampering or theft.

IoT Platforms

Security and Privacy aspects regarding the accessibility of the data for specific owners and/or organizations

What about data integrity certification?


SNAP4CITY
SMART SOLUTIONS AND DECISION SUPPORT SYSTEMS
CONTROL ROOMS - DECISION SUPPORT SYSTEMS - WHAT-IF ANALYSIS - BUSINESS INTELLIGENCE - SIMULATIONS - SMART APPLICATIONS
DASHBOARDS - VISUAL ANALYTICS - SYNOPTICS - DIGITAL TWIN - GRAPHICAL WIDGETS - ANALYTICS - GUI CUSTOM STYLES - VISUAL PROGRAMMING
DASHBOARDS, WIDGETS TEMPLATES
PREDICTION - ANOMALY DETECTION - CLUSTERING - ROUTING - SENTIMENT NLP - TRAFFIC FLOW PEOPLE FLOWS - SDG - 15 MIN CITY INDEX - KPI - HEATMAPS - ORIGIN DESTINATION - ETC...
API - MICROSERVICES - GIS - BPM VIDEO - REPORTS - MAPS - 3D ...
SMART CITY LIVING LAB
ANY: DATA, BROKER, NETWORK AND VERTICAL
EXPERT SYSTEM, KNOWLEDGE BASE SEMANTIC REASONING SMART DATA MODEL IOT DEVICE MODELS, STORAGE
BIG DATA ANALYTICS, ARTIFICIAL INTELLIGENCE EXPLAINABLE AI, MACHINE LEARNING OPERATIVE RESEARCH, STATISTICS
VISUAL PROGRAMMING, ADAPTERS DATA FLOWS, WORKFLOWS PARALLEL DISTRIBUTED PROCESSING DATA DRIVEN
Native and External Applications
Smart Parking
Smart Light
Smart Waste
Smart Energy
Social Media Analysis
...
METHODOLOGIES LIVING LABS COURSES AND COMMUNITY DEVELOPMENT TOOLS
INDUSTRY4.0
PARKING
AGRICOLTURE
ENVIRONMENT
ENERGY
HEALTH
CONTROL ROOM
MOBILE
WASTE
SECURITY
BUILDING
SATELLITE
TRANSPORT

The rise of BIoT

The combination of Blockchain and Internet of Things (BIoT) is a promising solution for addressing the challenges of data security, privacy, and trust in IoT systems. BIoT can help provide a secure and *transparent* way to manage and transfer data, while also enabling greater interoperability and efficiency.

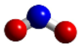
- One of the key applications of BIoT is **data certification,** which involves using Blockchain to certify the integrity and authenticity of data collected by IoT devices. This can be particularly important in applications such as environmental monitoring, where the accuracy and reliability of data can have significant implications for public health and safety.

Can you suggest an example of application in which the combi

Blockchain and IoT would be extremely beneficial?

Air Quality Example

The **European Commission** uses a network of sensors to monitor air quality standards across different cities. These sensors collect data on air pollutants such as nitrogen dioxide, ozone, and particulate matter. However, there are concerns about the accuracy and reliability of the data, as well as the potential for tampering or manipulation.



By using Blockchain to certify the data collected by these sensors, the European Commission can provide greater transparency and accountability in the air quality monitoring process. Each sensor can be assigned a unique digital identity on the Blockchain, and the data collected can be stored in an immutable and tamper-proof way.

This can help ensure that the data is accurate and reliable, and that it can be trusted by policymakers, researchers, and the public. It can also help identify any anomalies or discrepancies in the data, and enable more effective and targeted interventions to improve air quality standards.

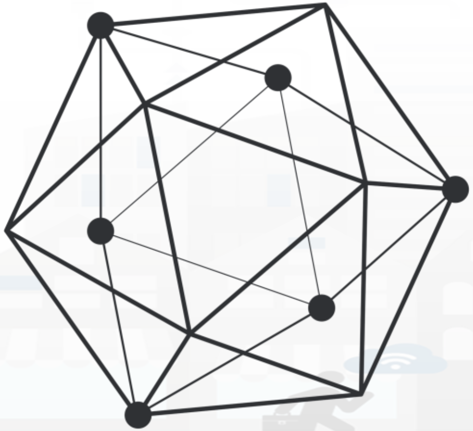
7 BIoT Benefits

shutterstock.com · 336997097

- **Enhanced Security**: Blockchain technology provides a secure and tamper-proof way to store and manage data, which can help address security concerns in IoT systems.
- **Increased Transparency**: The transparent and decentralized nature of Blockchain can help improve transparency and accountability in IoT systems.
- **Improved Data Management:** The ability of Blockchain to store and manage data in a secure and immutable way can help improve data management in IoT systems.
- **Increased Efficiency**: The use of Blockchain in IoT systems can help reduce the need for intermediaries and increase the efficiency of transactions and data transfers.
- **Improved Interoperability**: Blockchain can help address the lack of standardization in IoT protocols and technologies, enabling greater interoperability between devices and systems.
- **Improved Trust**: The use of Blockchain in IoT systems can help improve trust between parties by providing a secure and transparent way to manage transactions and data.
- **Data Certification:** The use of Blockchain in IoT systems data can be certified and validated upon eventual modification.

HYPERLEDGER FABRIC

Hyperledger Fabric



What is Hyperledger Fabric?

- Hyperledger Fabric is an open source enterprise-grade permissioned distributed ledger technology (DLT) platform, developed by The Linux Foundation.

Main differences between Fabric and other blockchain platforms:

- Modular and configurable.
- No DSL specific languages to write smart contracts (chaincodes).
- Permissioned platform.
- Pluggable consensus algorithm.
- Native cryptocurrency not required.

Modularity

To achieve a certain degree of personalization Fabric is composed by a plethora of modular components, specifically:

- The ordering service responsible for the consensus on the order of transactions and then the broadcasting to the peers
- MSP(Membership service provider) responsible for associating entities with cryptographic identities
- Chaincode (Fabric’s smart contracts) that run isolated on a docker container written in various general purpose programming languages.
- The ledger can use a variety of DBMSs
- Endorsing and validation policies can be configured per application specific requirements

Permissioned vs Permissionless

PERMISSIONLESS

- Anyone can participate as an anonymous user.
- There is no trust other than the state of the blockchain that, ro a certain depth is immutable.
- Typically employ a native cryptocurrency to offer incentive to offset the high costs to participate in a form of BFT (Byzantine Fault Tolerant) consensus based on "Proof of Work" like Bitcoin or Ethereum (pre EIP-3675)

PERMISSIONED

- Operate on a set of known and identified participant under a governance model that yield a certain degree of trust.
- Provides a way to secure interactions between a group of entities that not fully trust each other.
- Using identities "costly" mining is not required.
- If a bad actor introduce bad code, is recorded and with identities can be easily identified and handled in accordance with the terms of the governance model.

HYPERLEDGER FABRIC

Smart contracts / Chaincode

It’s the business logic of a blockchain application, gaining it’s security/trust from the underlying consensus among the peers.

There are 3 key point that apply to a smart contract:

- The run concurrently in a network
- The may be deployed dynamically usually by anyone on the network
- Application code should be treated as untrusted and potentially even malicious

Most smart contracts use a **order-execute** architecture, in which the consensus protocol:

➔ validate and orders transactions then popagates them to all the peers

➔ each peer then executes the transactions sequentially

This architecture can be found in virtually all existing blockchain systems

Smart contracts / Chaincode

Smart contract must be deterministic otherwise consensus might never be reached, to address this issue many platform require smart contract written in DSL (Domain specific languages) (ex. Solidity).

This approach limits widespread adoption because learning and programming in a new, specific language can lead to errors.

To address this issue fabric utilize a the so called **Execute-order-validate** architecture

- **execute** a transaction and check it's correctness
- **order** a transaction
- **validate** the transaction against an application-specific endorsement policy before committing it to the ledger.

The first step eliminate the non-determinism problem as inconsistent transactions can be filtered out before ordering.

This also enables use of standard programming languages as Java, Javascript and Go.

Private data

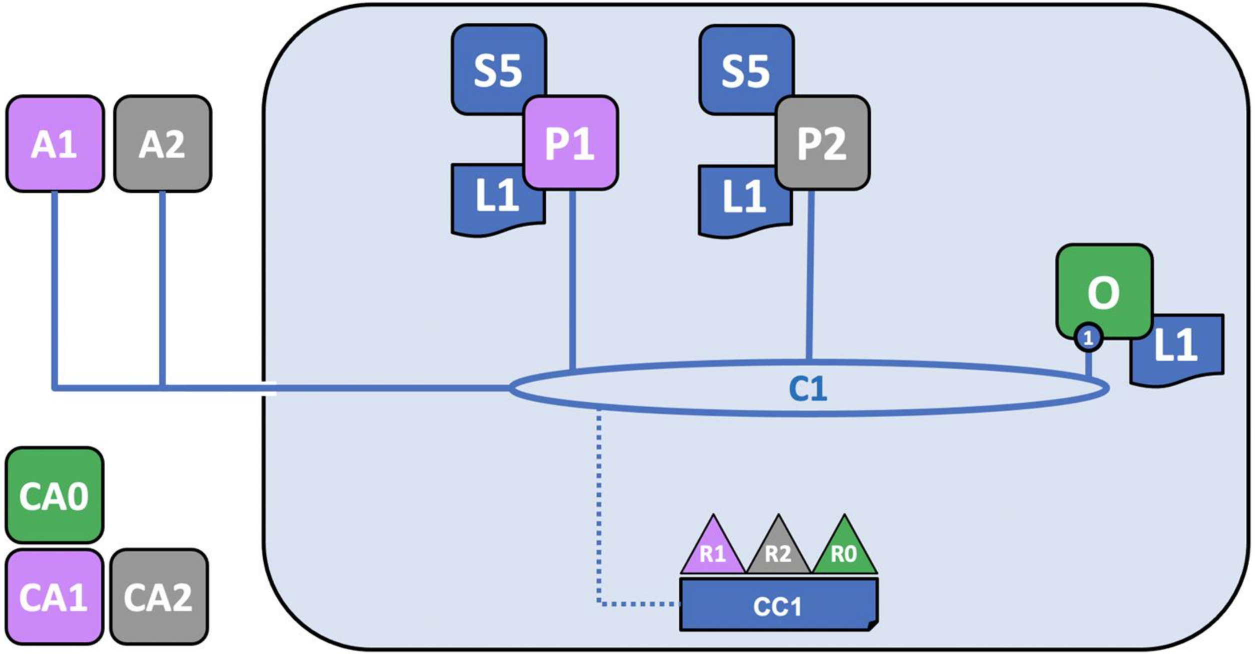
Hyperledger Fabric enables confidentiality through its channel architecture and private data feature.

In channels, participants on a Fabric network establish a sub-network where every member has visibility to a particular set of transactions.

Thus, only those nodes that participate in a channel have access to the smart contract (chaincode) and data transacted, preserving the privacy and confidentiality of both.

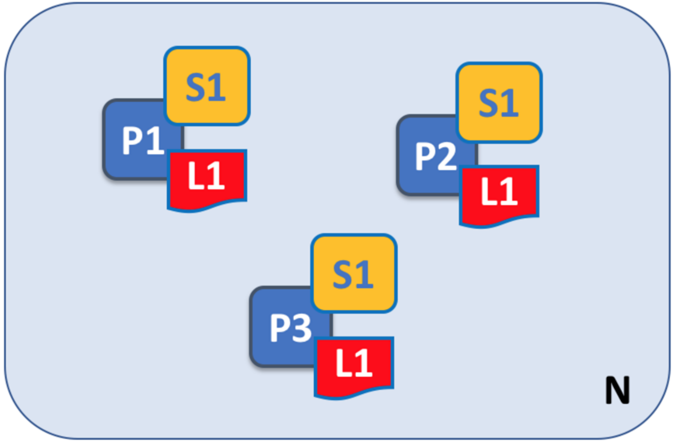
Fabric Sample network

Three organizations, R1, R2, and R0 have jointly decided that they will establish a network. This network has a configuration, CC1, which all of the organizations have agreed to and which lists the definition of the organizations as well as the policies which define the roles each organization will play on the channel.



Peers

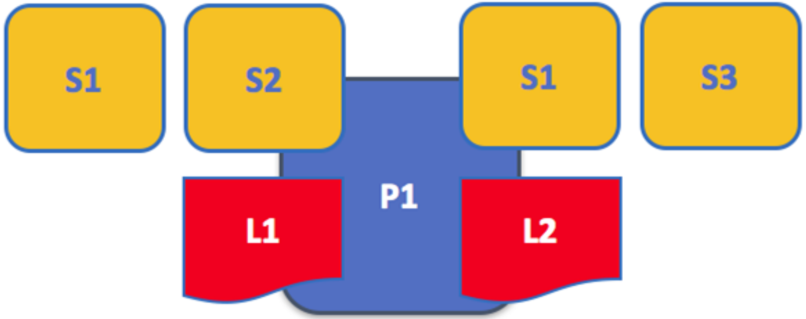
A fundamental element of a Hyperledger Fabric blockchain network is the set of **peers**. Peers are fundamental because they manage ledgers and smart contracts. Recall that a ledger immutably records all of the transactions generated by smart contracts and endorsed by the required organizations. Smart contracts and ledgers encapsulate the *processes* and *information*, respectively, that are shared by channel peers.



N	Blockchain network
P	Peer node
S	Smart contract (aka chaincode)
L	Ledger

Peers

- A peer is capable of hosting more than one ledger, which is useful because it allows for a flexible system design where a single peer can belong to multiple channels in a Fabric network.



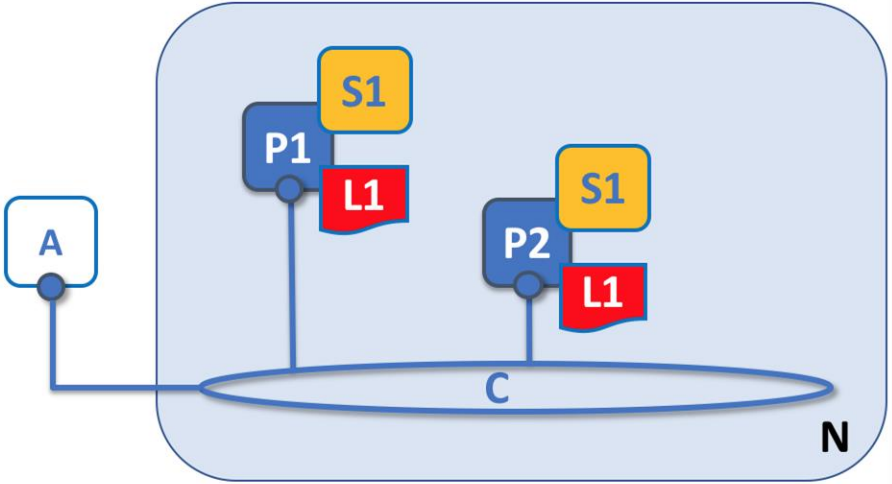
- A peer also is capable of hosting multiple chaincodes, this introduce us to the importance of **channels**

Channels

A **channel** is a mechanism by which components within a Fabric blockchain network **communicate** and transact **privately**.

Channel components include peer nodes, orderer nodes and applications, and by joining a channel, they agree to collaborate to collectively manage and share identical copies of the ledger.

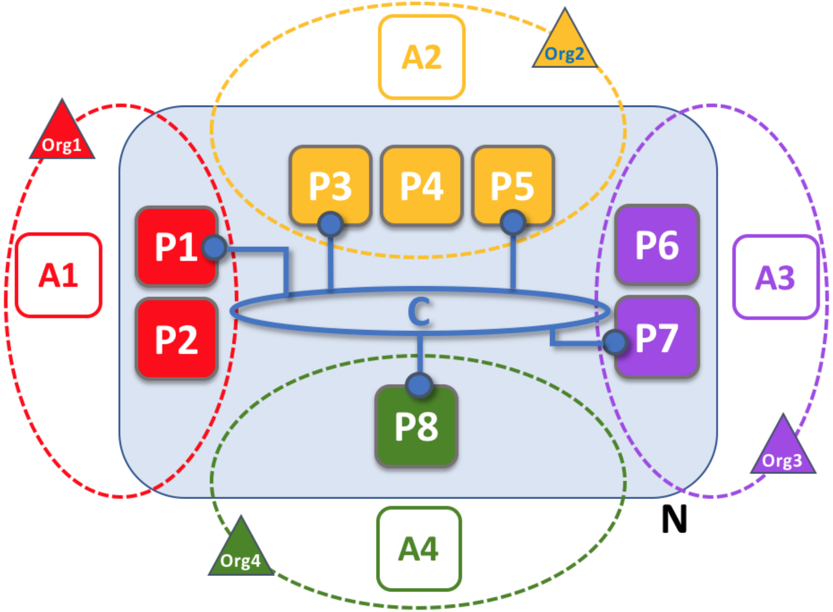
It's more accurate to think of a channel as a logical structure that is formed by a collection of physical peers. *It is vital to understand this point — peers provide the control point for access to, and management of, channels*.



Organizations

Fabric blockchain networks are administered by a collection of organizations rather than a single organization. Peers are central to how this kind of distributed network is built because they are owned by and are the network connection points for these organizations.

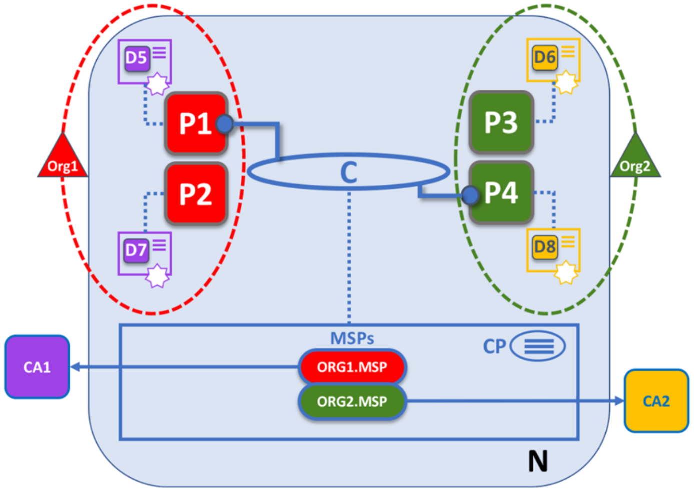
The network is both formed and managed by the multiple organizations that contribute resources to it. The network grows as resources are provided by collaborating organizations, increasing the resiliency and security of the network.



Identities

Peers have an identity assigned to them via a digital certificate (X.509) from a particular certificate authority.

Think of a digital certificate as like an ID card that provides verifiable information about a peer. *Each and every peer in the network is assigned a digital certificate by an administrator from its owning organization.*

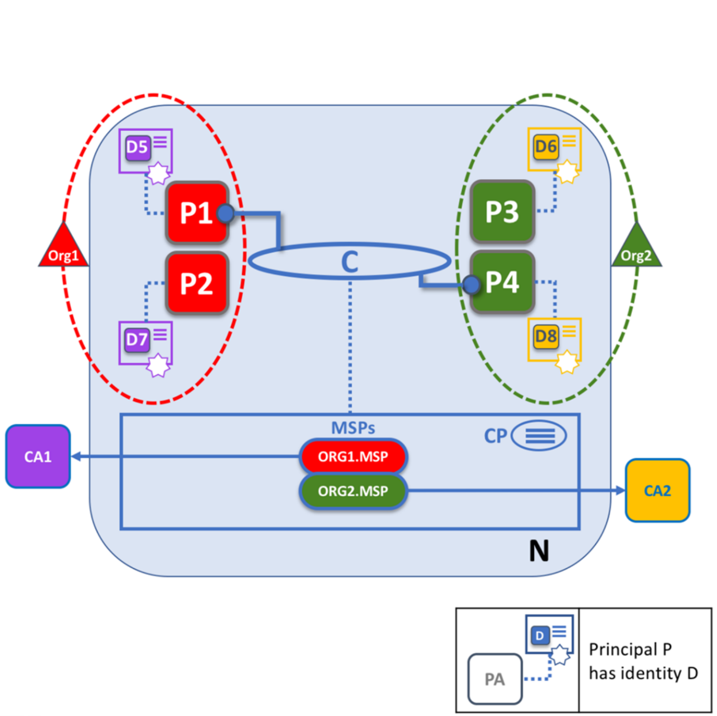


N	Blockchain Network	P	Peer
C	Channel	Org	Organization
I	Identity	PA C	Principal PA (e.g. P1,P4) communicates via channel C.
CP	Channel policy		
CA	Certificate Authority	MSP	Membership Service Provider
R A1 P1 P2	Organization R owns application A1 and peers P1, P2.		
C CP	Channel C subject to policy CP.	MSP1 MSP2 CP	Channel policy CP contains MSPs: MSP1 and MSP2.
MSP1 → CA1	MSP1 selects the Certificate Authority CA1 to provide certificates for it.		

PA D	Principal P has identity D

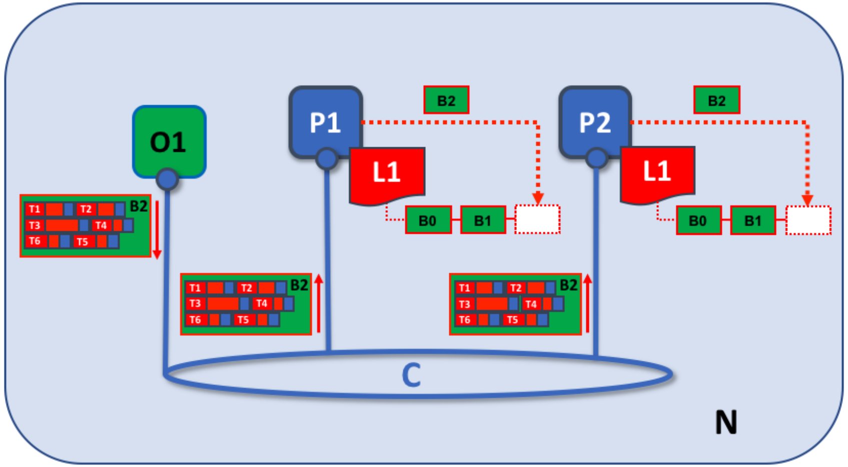
Identities (2)

Whenever a peer connects to a Fabric network channel, *a policy in the channel configuration uses the peer's identity to determine its rights.* The mapping of identity to organization is provided by a component called a *Membership Service Provider* (MSP) — which determines how a peer gets assigned to a specific role in a particular organization and accordingly, gains authorized access to resources. Moreover, a peer can be owned only by a single organization, and is therefore associated with a single MSP.



N	Blockchain Network	P	Peer
C	Channel	Org	Organization
I	Identity	PA C	Principal PA (e.g. P1,P4) communicates via channel C.
CP	Channel policy		
CA	Certificate Authority	MSP	Membership Service Provider
A1 R P1 P2		Organization R owns application A1 and peers P1, P2.	
C CP	Channel C subject to policy CP.	MSP1 MSP2 CP	Channel policy CP contains MSPs: MSP1 and MSP2.
MSP1 → CA1		MSP1 selects the Certificate Authority CA1 to provide certificates for it.	

Orderers and transactions



N	Blockchain Network	P	Peer
C	Channel	O	Orderer
L	Ledger	B	Block B
L1 B0 B1	Ledger L1 has blockchain with blocks B0, B1	T1 B1 T2 T3	Block B1 contains transactions T1, T2, T3…
B1 C	Block B1 flows on channel C	PA C	Principal PA (P1, P2) communicates via channel C.

The mechanism by which applications and peers interact with each other to keep the ledger current and consistent across a channel is mediated by special nodes called *orderers*.

A ledger update transaction is different from a query transaction because a single peer cannot, on its own, update the ledger — that requires the consent of other peers in the network, a process known as *consensus*.

Phase 1: Transaction proposal

In Phase 1, the client application initiates a request to the Fabric Gateway service to evaluate a transaction proposal.

The target peer, selected by the client application, executes the transaction by invoking chaincode — this step can be described as simulating the transaction, because it runs the transaction without any effect on the ledger. The peer then returns its transaction result to the client.

The gateway service also forwards the transaction proposal to the required *endorsing peers* (based on the endorsement policies), which also execute the transaction and return their results to the peer. The gateway service collects all responses, and if they collectively satisfy the endorsement policies, **forwards the transaction to the ordering service**.

Note that a peer endorses a proposal response by adding its digital signature, and signing the entire payload using its private key. This endorsement can be subsequently used to prove that this organization's peer generated a particular response.

Phase 2: Transaction ordering

The second phase of the transaction workflow is the ordering and packaging phase. The ordering service (running on orderer nodes) receives transactions containing signed and endorsed proposal responses, from one or more applications via the gateway service, and orders and packages the transactions into blocks. These are the blocks (which are also ordered) — consisting of endorsed and ordered transactions — that make up a Fabric blockchain ledger.

Phase 3: Transaction ordering

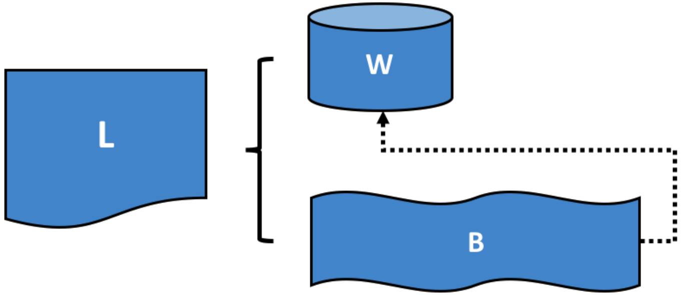
The third and final phase of the transaction workflow is the distribution of ordered transactions from orderers to peers. Each peer then validates each transaction, in the correct order, and ensures that each transaction has been consistently endorsed by all required organizations. Only then does the peer commit the block to its copy of the channel ledger.

It’s also worth noting that not every peer needs to be connected to an orderer — peers can cascade blocks to other peers using the **gossip** protocol, who also can process them independently.

Ledger and world state database

Firstly, there’s a **world state** – a database that holds **current values** of a set of ledger states. *The world state makes it easy for a program to directly access the current value of a state rather than having to calculate it by traversing the entire transaction log*. Ledger states are, by default, expressed as **key-value** pairs.

Secondly, there’s a **blockchain** – a transaction log that records all the changes that have resulted in the current world state. Transactions are collected inside blocks that are appended to the blockchain – enabling you to understand the history of changes that have resulted in the current world state. The blockchain data structure is very different to the world state because once written, it cannot be modified; it is **immutable**.

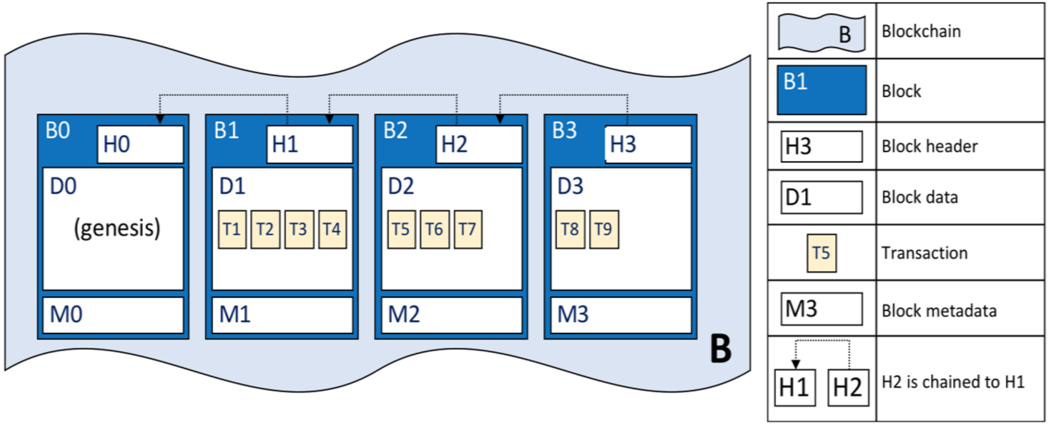


L	Ledger
W	World State
B	Blockchain
L { B W	L comprises B and W
W◄······ B······	B determines W

Fabric Blockchain

The blockchain is structured as sequential log of interlinked blocks, where each block contains a sequence of transactions, each transaction representing a query or update to world state.

What's important is that block sequencing, as well as transaction sequencing within blocks, is established when blocks are first created by a Hyperledger Fabric component called the **ordering service**.

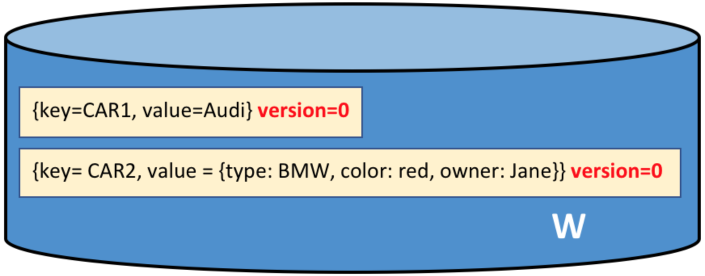


World State database

The world state holds the current value of the attributes of a business object as a unique ledger state. That's useful because programs usually require the current value of an object; it would be cumbersome to traverse the entire blockchain to calculate an object's current value.

An application program can invoke a smart contract which uses simple ledger APIs to **get**, **put** and **delete** states.

The world state given that is implemented as a database , provides a rich set of operators for the efficient storage and retrieval of states, so we can give the world state database **complex queries**.



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

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

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.

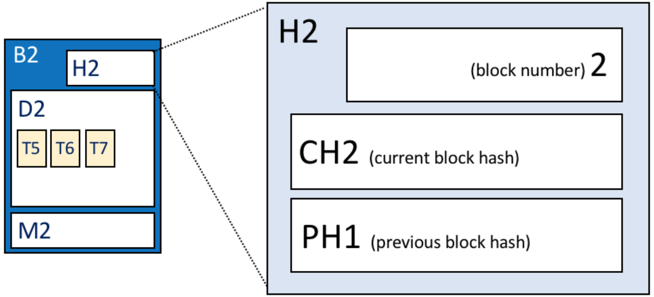
Blocks

A block has 3 sections:

- **Block Header**
  - **Block number**: An integer starting at 0 (the genesis block), and increased by 1 for every new block appended to the blockchain.
  - **Current Block Hash**: The hash of all the transactions contained in the current block.
  - **Previous Block Header Hash**: The hash from the previous block header.
- **Block Data**

  This section contains a list of transactions arranged in order. It is written when the block is created by the ordering service.
- **Block Metadata**

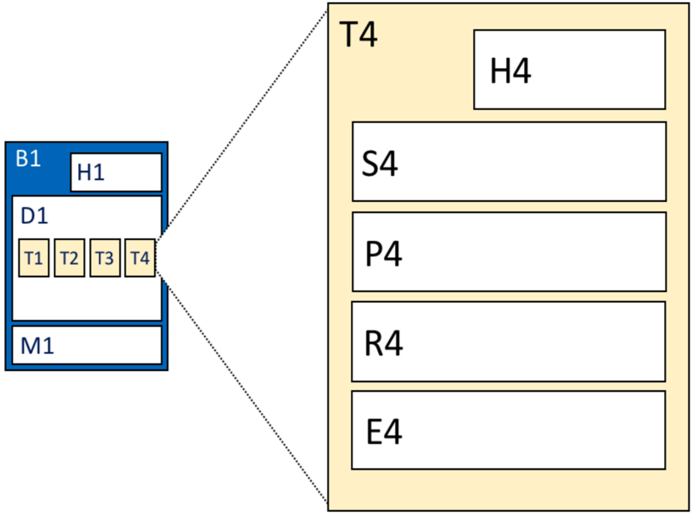
  This section contains the certificate and signature of the block creator which is used to verify the block by network nodes.



H2	Block header
2	Block number
CH2	Hash of current block transactions
PH1	Copy of hash from previous block
H2 ◁ V2	V2 is detailed view of H2

Transactions

- **Header**
  This section, illustrated by H4, captures some essential metadata about the transaction – for example, the name of the relevant chaincode, and its version.
- **Signature**
  This section, illustrated by S4, contains a cryptographic signature, created by the client application. This field is used to check that the transaction details have not been tampered with, as it requires the application's private key to generate it.
- **Proposal**
  This field, illustrated by P4, encodes the input parameters supplied by an application to the smart contract which creates the proposed ledger update. When the smart contract runs, this proposal provides a set of input parameters, which, in combination with the current world state, determines the new world state.
- **Response**
  This section, illustrated by R4, captures the before and after values of the world state, as a **Read Write set** (RW-set). It's the output of a smart contract, and if the transaction is successfully validated, it will be applied to the ledger to update the world state.
- **Endorsements**
  As shown in E4, this is a list of signed transaction responses from each required organization sufficient to satisfy the endorsement policy. You'll notice that, whereas only one transaction response is included in the transaction, there are multiple endorsements. That's because each endorsement effectively encodes its organization's particular transaction response – meaning that there's no need to include any transaction response that doesn't match sufficient endorsements as it will be rejected as invalid, and not update the world state.



T4	Transaction
H4	Header
S4	Signature
P4	Proposal
R4	Response
E4	Endorsements
T4 ◁ V4	V4 is detailed view of T4

World State Options

The world state is physically implemented as a database, to provide simple and efficient storage and retrieval of ledger states. As we've seen, ledger states can have simple or compound values, and to accommodate this, the world state database implementation can vary, allowing these values to be efficiently implemented. Options for the world state database currently include LevelDB and CouchDB.

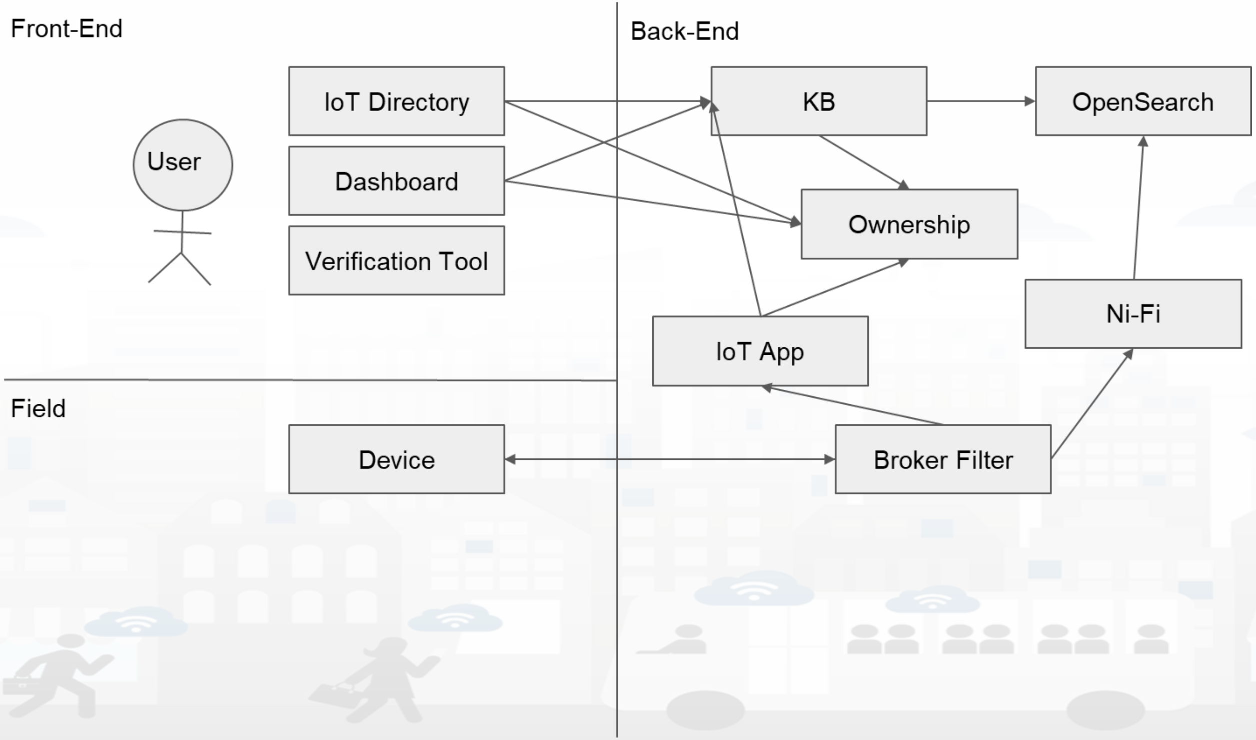
LevelDB is the default and is particularly appropriate when ledger states are simple key-value pairs. A LevelDB database is co-located with the peer node – it is embedded within the same operating system process.

CouchDB is a particularly appropriate choice when ledger states are structured as JSON documents because CouchDB supports the rich queries and update of richer data types often found in business transactions. Implementation-wise, CouchDB runs in a separate operating system process, but there is still a 1:1 relation between a peer node and a CouchDB instance.

All of this is invisible to a smart contract.

Hyperledger Fabric and Snap4city integration

Snap4City IoT Platform Architecture



Overview

The platform is interfaced to the real-world, in which a multitude of IoT Devices are located and supported (hosting several sensors and actuators). A simple IoT Device can be a microcontroller enriched with sensors/actuators capable of send/receive messages with some IoT Brokers. Since the IoT Platform needs to connect multiple IoT Brokers and Devices, an **IoT Directory** listing them and providing general abstraction services is needed. In The **IoT Directory** of Snap4City the figures managed are conceptualized as **IoT Device Models**, **IoT Devices**, **IoT Devices Data**, through an **IoT Broker Filter**.

The **IoT Broker Filter**, handles all the aspects for registration data injection and retrieval interfacing with the **Ownership Module** regarding the security aspects and delegating to **Ni-Fi** for the storage of the specific semantic information into the **Knowledge Base** (KB) and for the data storage in **Open Search**.

The IoT platform can not be totally agnostic about the data structure/model or the device broker used. To assess this Snap4City conceptualized the **IoT Device Model** as the set of information needed to recognize and manage device messages with specific attributes exchanged with any kinds of broker as well as the ownership aspect. The IoT Device Model provide a formal model format for the messages with formalized variables/attributes with data types, units, etc… assuring data flexibility.

Deepening

The IoT Device Models, once defined, through the IoT Broker Filter are mapped into the KB after the enrichment of the data through Ni-Fi to be interoperable with the rest of information. The structured semantically searchable new data models can then be linked with other entities and this improves performance of data ingestion, insertion and retrieval capabilities of the storage. The adoption of Km4City Ontology , creates a uniform layer abstracting from physical details and mechanisms needed to access through different Brokers and usage of several data models and their validation, as well as semantic interoperability and matching.

Once the IoT Device Model is defined and semantically conceptualized in the KB it can be used for the creation of the conceptualized **IoT Device**. This process can be repeated to instantiate multiple data entities with the same structure in terms of attributes and contextual information by the **IoT Entity Owner**. The IoT Entity Owner is the only one authorized to manipulate those entities, and thus entitled to provide delegation in access to those elements. The rights of the entities (both ownership and delegation) are managed by the **Ownership Module** who the IoT Broker Filter interfaces with for the security aspects.

The IoT Devices can send the **IoT Device Data** into the Snap4City platform. The data specifically are sent to the IoT Broker Filter that verifies the right to post a data message on the platform for the specific IoT Device interfacing with the Ownership Module. If the procedure is successful the data are passed by the IoT Broker Filter that pushes them on Ni-Fi, which in turn enriches them on the basis of KB info (performing a query or caching it), to finally post data on OpenSearch for real-time event-driven expositions on some front-end real-time Dashboard or data retrieval through the platform APIs, or managed into some IoT Applications to perform data elaboration and/or data analytics.

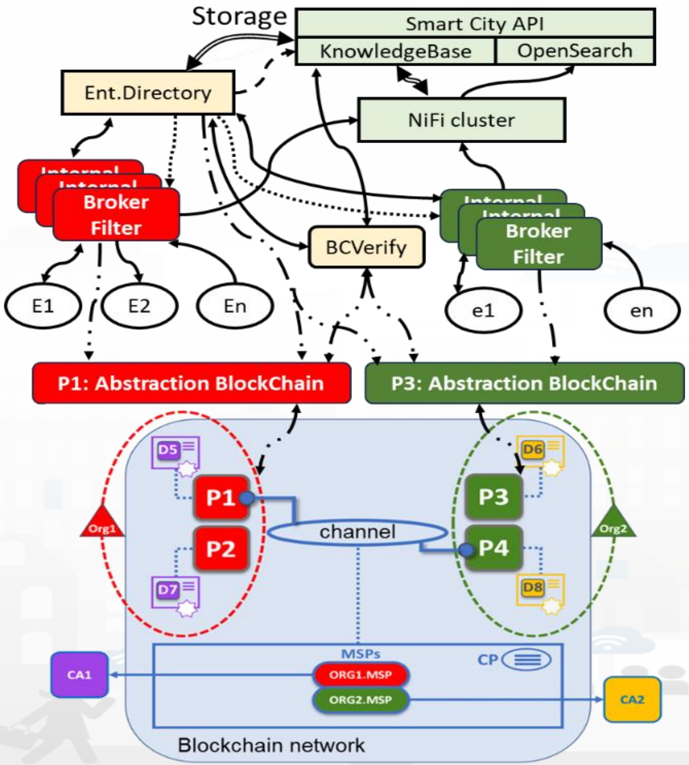
Functioning and Security

The platform is able to verify if the messages received from IoT Device are correct in terms of IoT Data Model including verification of attribute conformance before accepting them. Every IoT Device should be formally registered in the platform before accepting their IoT Device Data, and they have to pass a verification phase at run time.

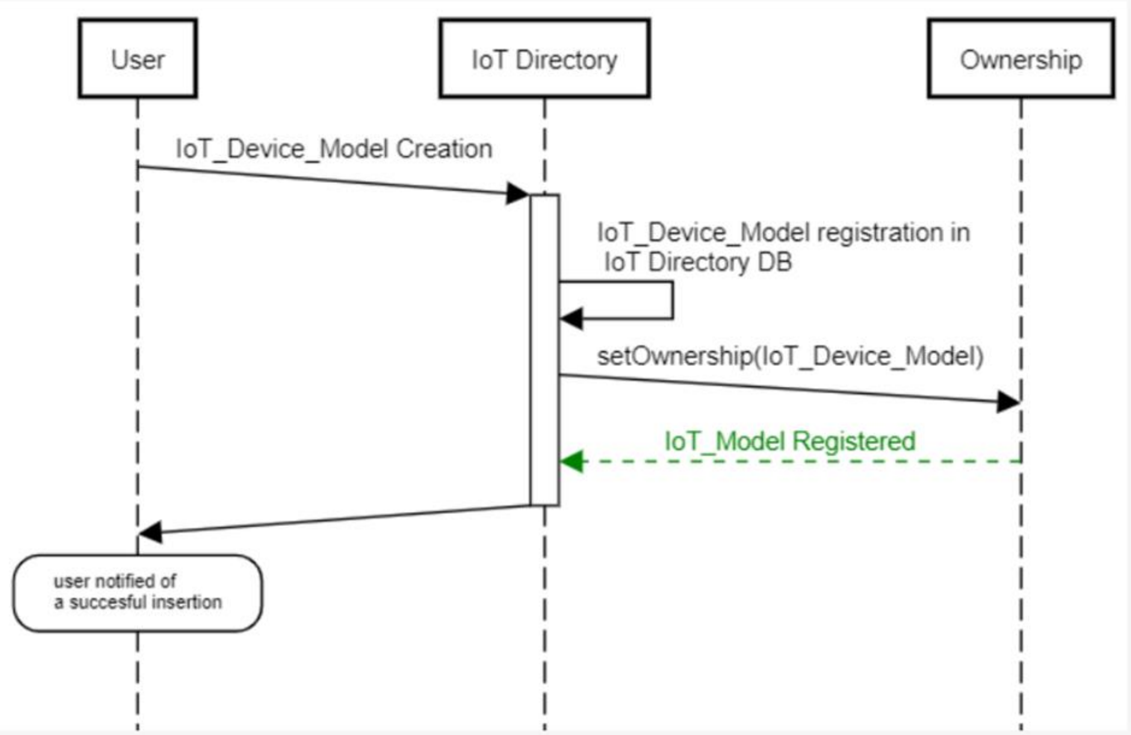
The Snap4City platform architecture can handle sensible data with respect to vulnerabilities and in respect of the GDPR (General Data Protection Regulation) managing the ownership and access grants of the devices. According to GDPR any entity has to start as private of the owner and the platform then is able to assign/change this and create access grants to entities (IoT Brokers, Devices, and Data Models). The delegation management allows the possibility to grant, list and revoke grants.

On the platform have been conducted multiple stress and penetrations test that demonstrate the robustness of the solution with respect to a large number of potential vulnerability aspects, The integration of the blockchain could provide the certification of data integrity in terms that even if a device owner or a member of the organization to which it has been delegated the access to the device if someone modify a data stored in the platform this can be verified and highlighted.

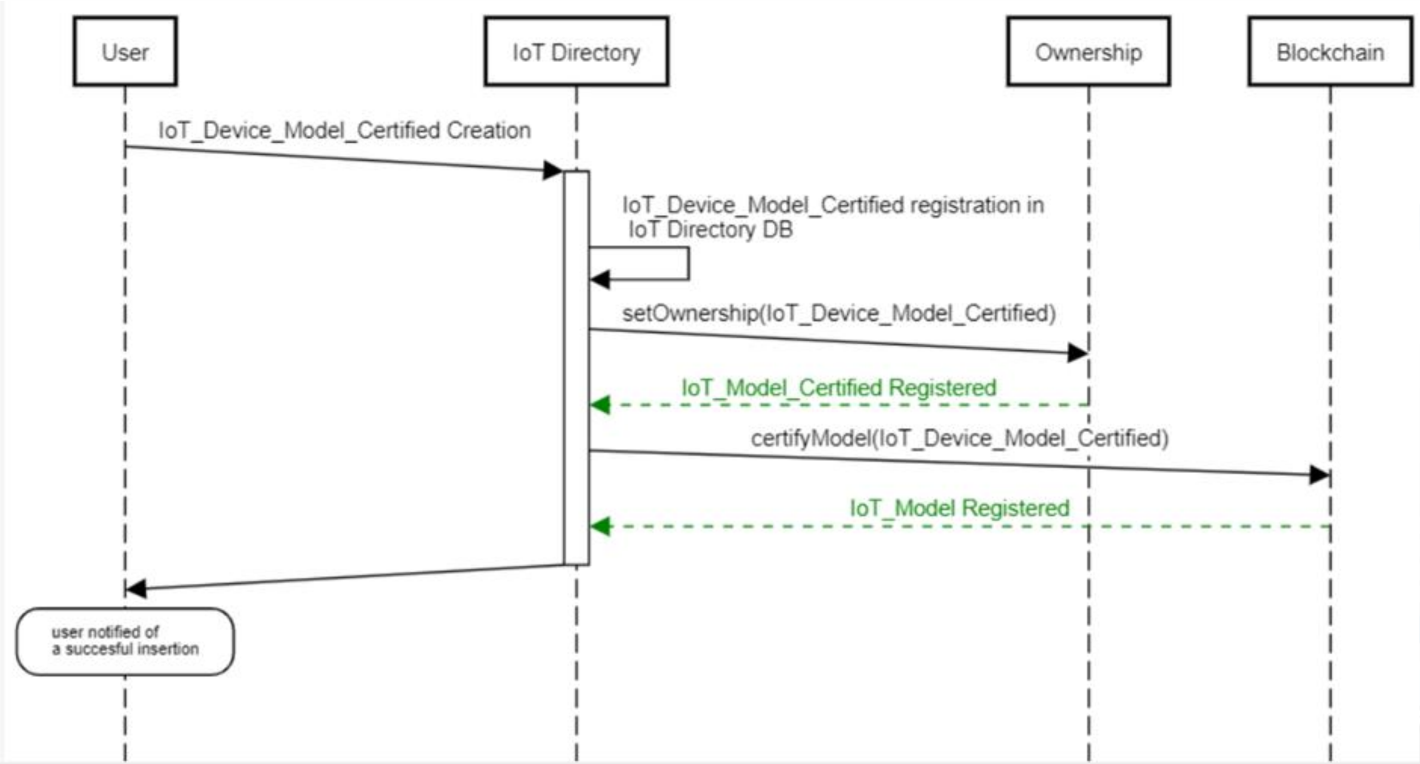
Blockchain Integration



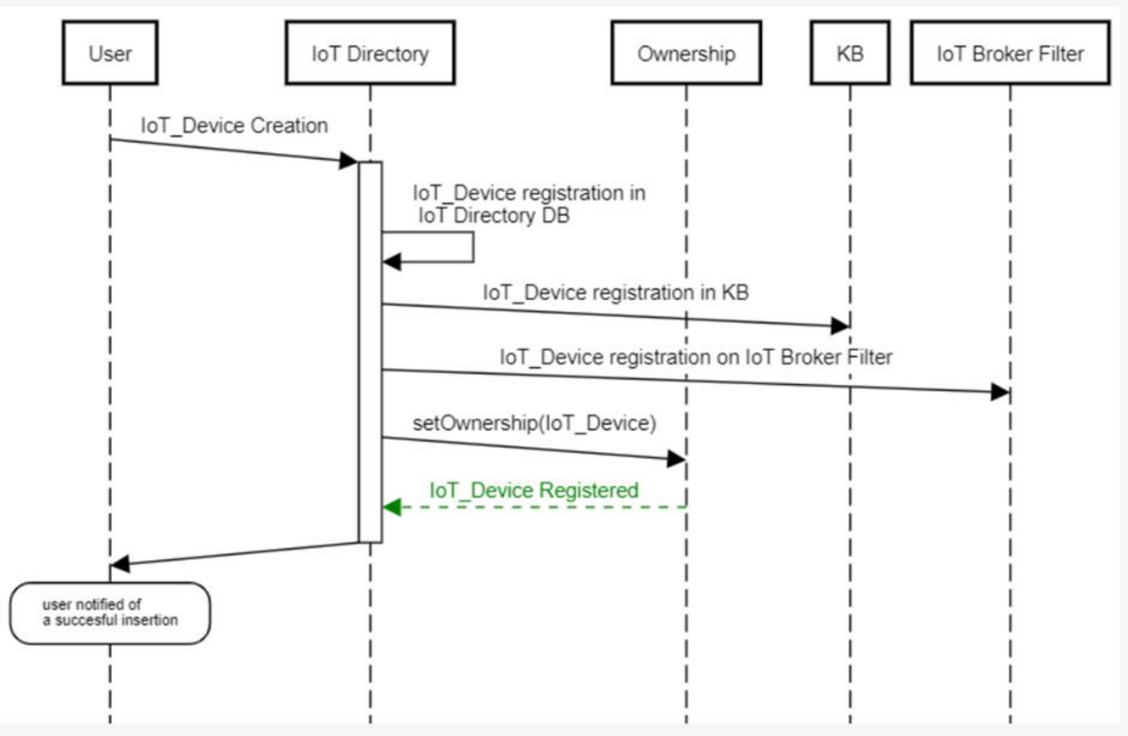
IoT Model Registration



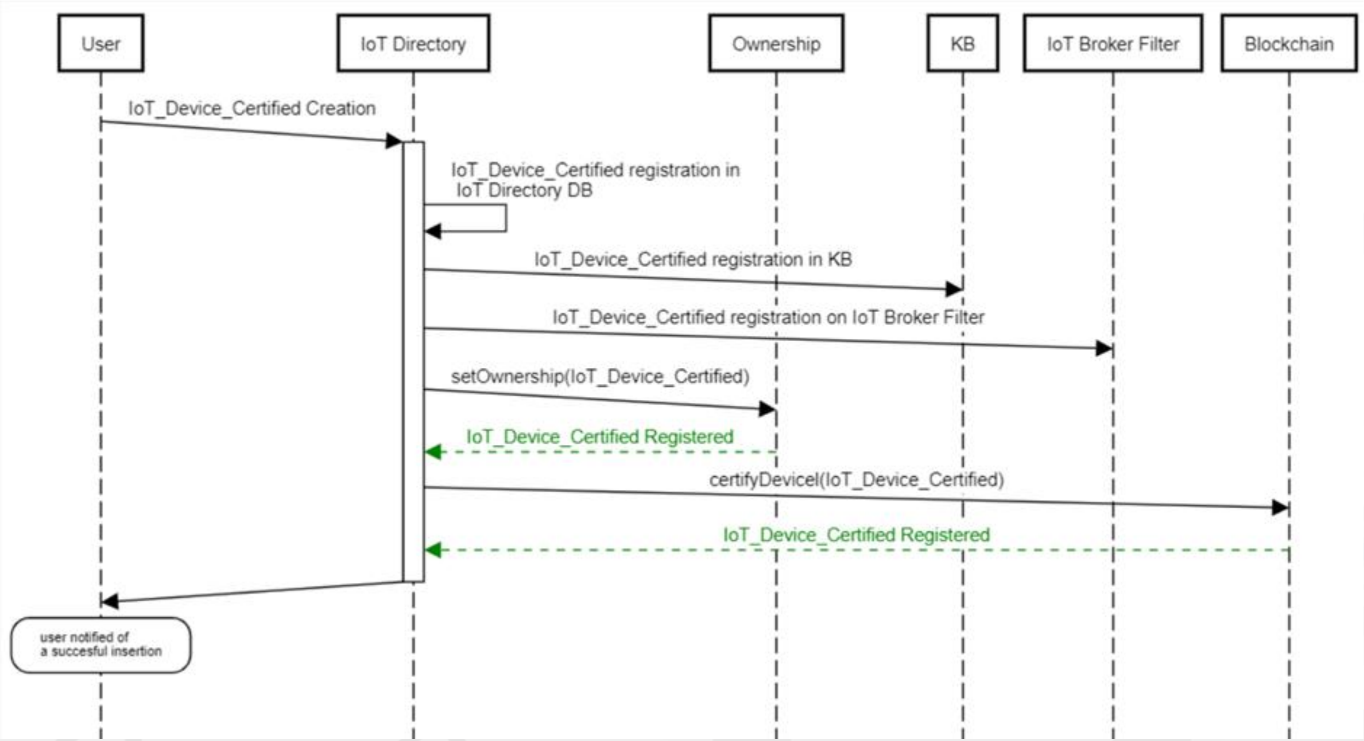
IoT Model Registration + BlockChain


User
IoT Directory
Ownership
Blockchain
IoT_Device_Model_Certified Creation
IoT_Device_Model_Certified registration in IoT Directory DB
setOwnership(IoT_Device_Model_Certified)
IoT_Model_Certified Registered
certifyModel(IoT_Device_Model_Certified)
IoT_Model Registered
user notified of a succesful insertion

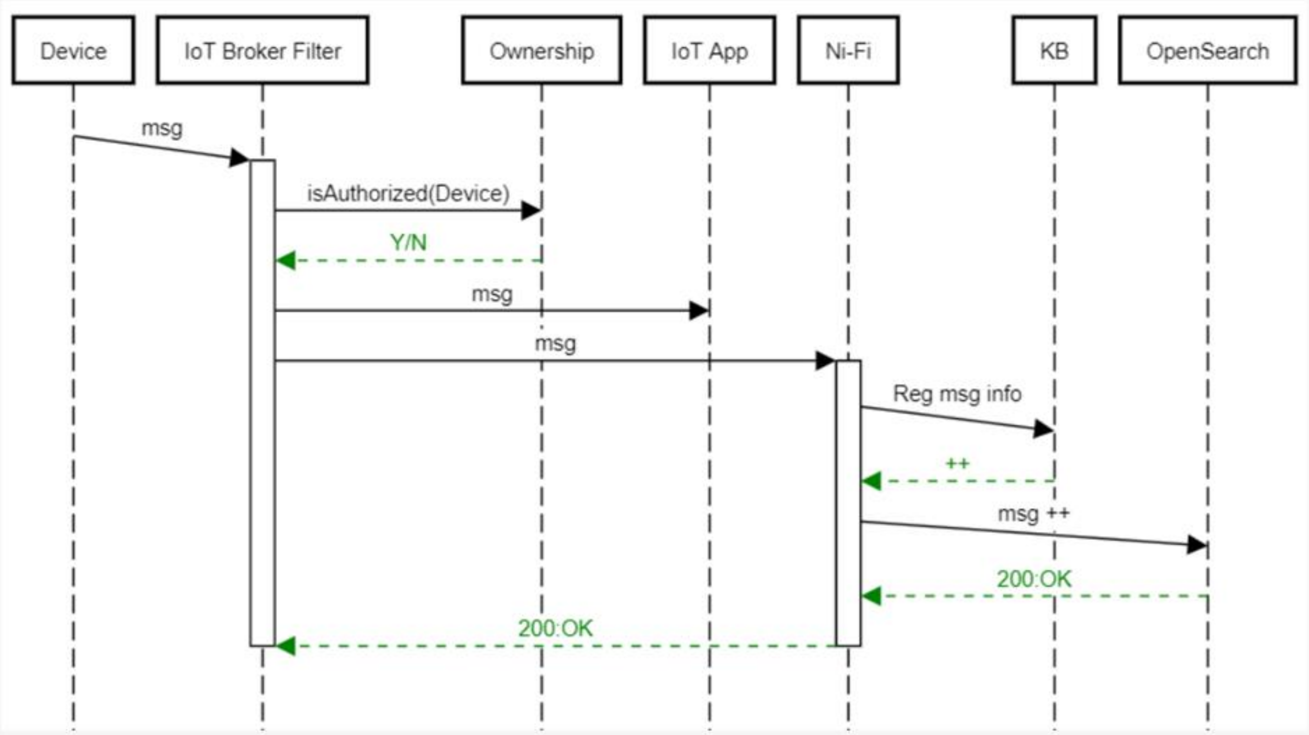
IoT Device Registration


User
IoT Directory
Ownership
KB
IoT Broker Filter
IoT_Device Creation
IoT_Device registration in IoT Directory DB
IoT_Device registration in KB
IoT_Device registration on IoT Broker Filter
setOwnership(IoT_Device)
IoT_Device Registered
user notified of a succesful insertion

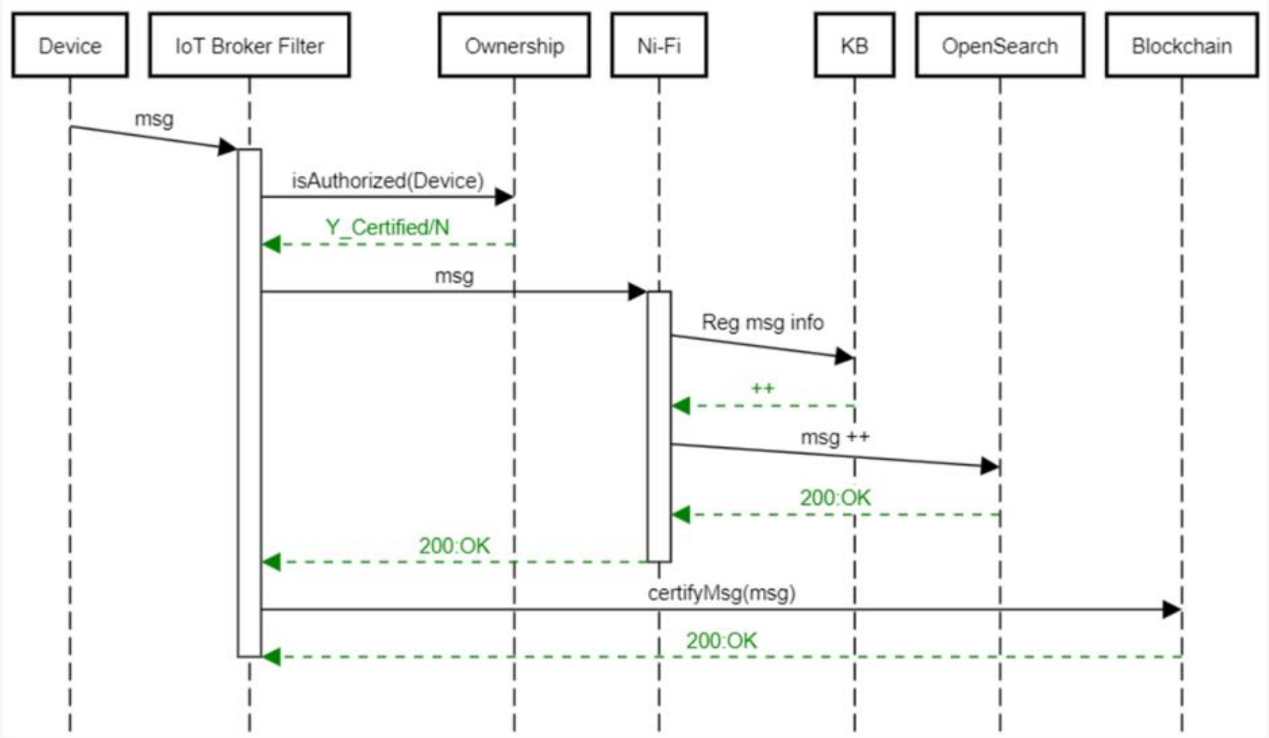
IoT Device Registration + Blockchain


User
IoT Directory
Ownership
KB
IoT Broker Filter
Blockchain
IoT_Device_Certified Creation
IoT_Device_Certified registration in IoT Directory DB
IoT_Device_Certified registration in KB
IoT_Device_Certified registration on IoT Broker Filter
setOwnership(IoT_Device_Certified)
IoT_Device_Certified Registered
certifyDeviceI(IoT_Device_Certified)
IoT_Device_Certified Registered
user notified of
a succesful insertion

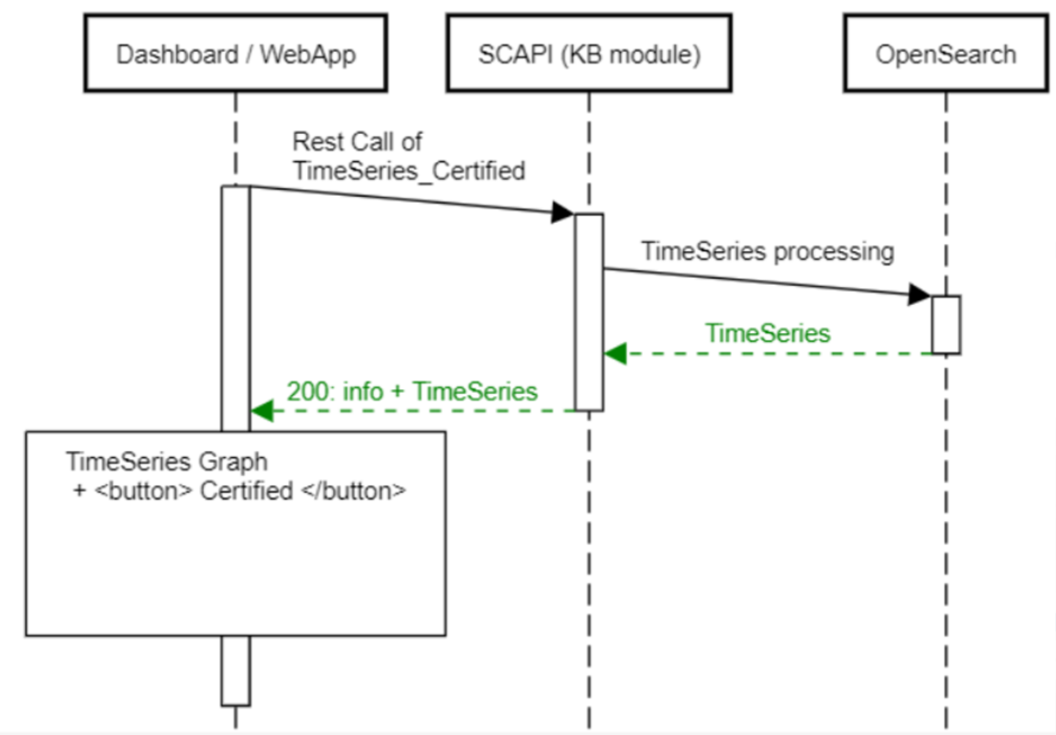
Data Ingestion



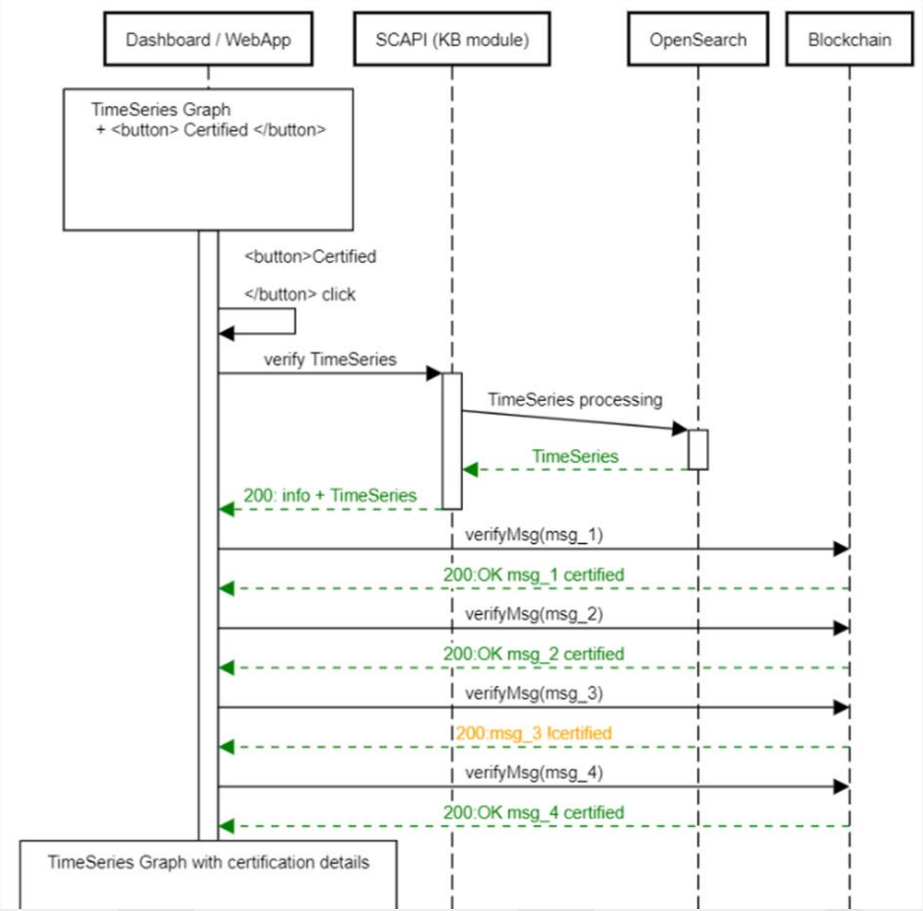
Data Ingestion + Blockchain



Data Retrieval



Data Integrity Verification



Dashboard / WebApp

SCAPI (KB module)

OpenSearch

Blockchain

TimeSeries Graph
+ <button> Certified </button>

<button>Certified

</button> click

verify TimeSeries

TimeSeries processing

TimeSeries

200: info + TimeSeries

verifyMsg(msg_1)

200:OK msg_1 certified

verifyMsg(msg_2)

200:OK msg_2 certified

verifyMsg(msg_3)

200:msg_3 Icertified

verifyMsg(msg_4)

200:OK msg_4 certified

TimeSeries Graph with certification details

Blockchain abstraction layer

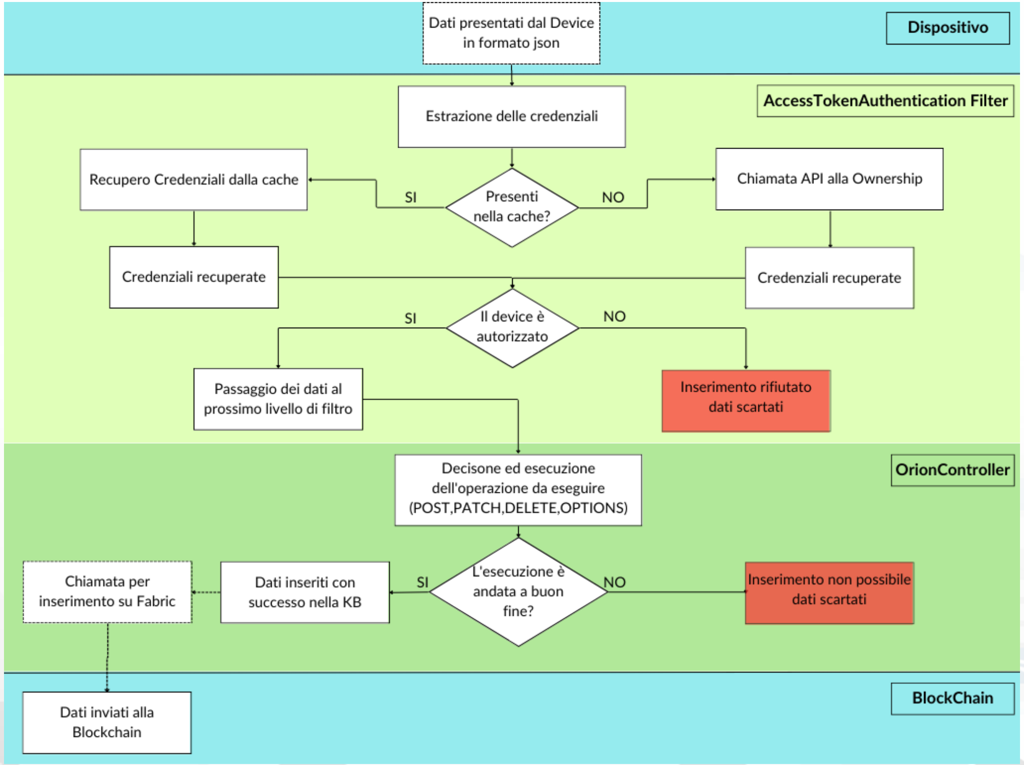
Blockchain abstraction layer (Device -> Orion Filter)

A device that wants to send data to the platform sends measured data and its access token to the Orion broker Filter (**Green**), as seen in the previous slides.

The filter uses a cache system to check if the device credentials are already saved. If they are present, the data is sent to the KB for insertion.

The blockchain comes into play when a device that is 'Certified' writes something to the KB. After the credentials check, the filter performs an API call to a made-to-measure endpoint(**Blue**).

In the case of a failed credential check, wrong permissions, or an error in the KB insertion, the data is discarded and the blockchain API endpoint is never called.



Dispositivo

Dati presentati dal Device in formato json

AccessTokenAuthentication Filter

Estrazione delle credenziali

Presenti nella cache?

SI

Recupero Credenziali dalla cache

Credenziali recuperate

NO

Chiamata API alla Ownership

Credenziali recuperate

Il device è autorizzato

SI

Passaggio dei dati al prossimo livello di filtro

NO

Inserimento rifiutato dati scartati

OrionController

Decisone ed esecuzione dell'operazione da eseguire (POST,PATCH,DELETE,OPTIONS)

L'esecuzione è andata a buon fine?

SI

Dati inseriti con successo nella KB

Chiamata per inserimento su Fabric

NO

Inserimento non possibile dati scartati

BlockChain

Dati inviati alla Blockchain

Blockchain abstraction layer (Orion Filter -> API endpoint)

When the Endpoint receives data from the Orion Controller checks, once again the device credentials calling the keycloak service.

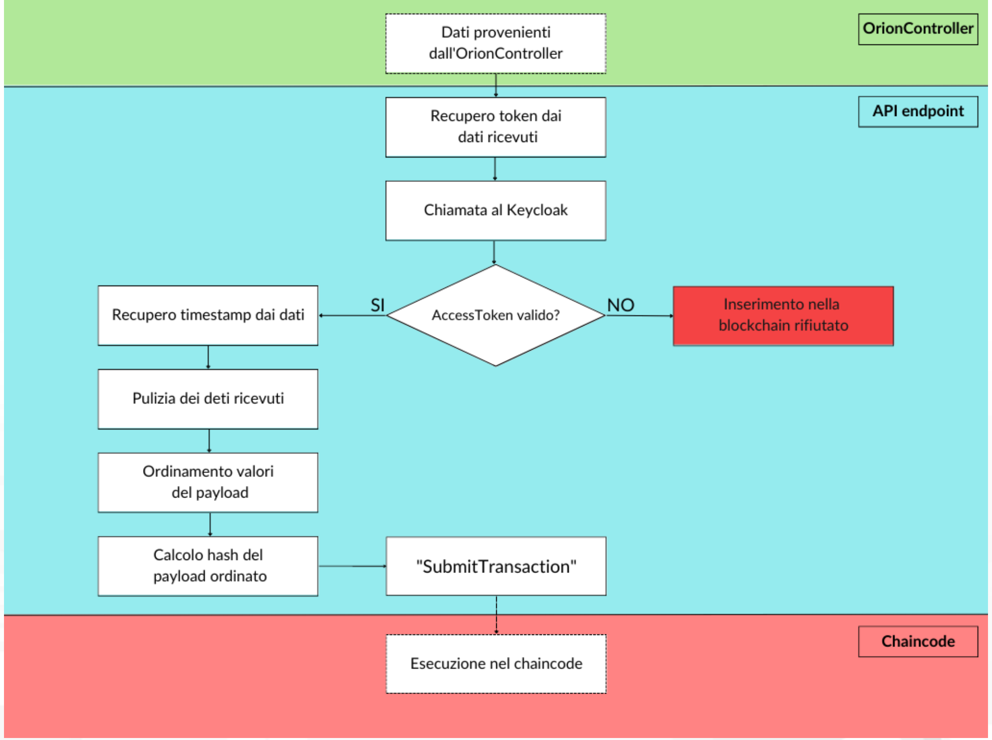
If the device credentials are valid the endpoint performs a sort of cleaning and ordering of the data that he received.

For example, a device send data in a json like this below:

```
"results":[
                 "temperature":{"type":"float","value":"20"},
        "altitude":{"type":"float","value":"30"},
        "dateObserved":{"type":"timestamp","value":"2023-03-
23T17:11:27.009Z"}}]
```

After cleaning and ordering the json will look like this, variables in alphabetical order and data types removed:

```
"results":[
        "altitude":{"value":"30"},
        "dateObserved":{,"value":"2023-03-23T17:11:27.009Z"},
        "temperature":{"value":"20"}]
```


OrionController
Dati provenienti dall'OrionController
API endpoint
Recupero token dai dati ricevuti
Chiamata al Keycloak
AccessToken valido?
SI
NO
Inserimento nella blockchain rifiutato
Recupero timestamp dai dati
Pulizia dei deti ricevuti
Ordinamento valori del payload
Calcolo hash del payload ordinato
"SubmitTransaction"
Chaincode
Esecuzione nel chaincode

Blockchain abstraction layer (Orion Filter -> API endpoint)

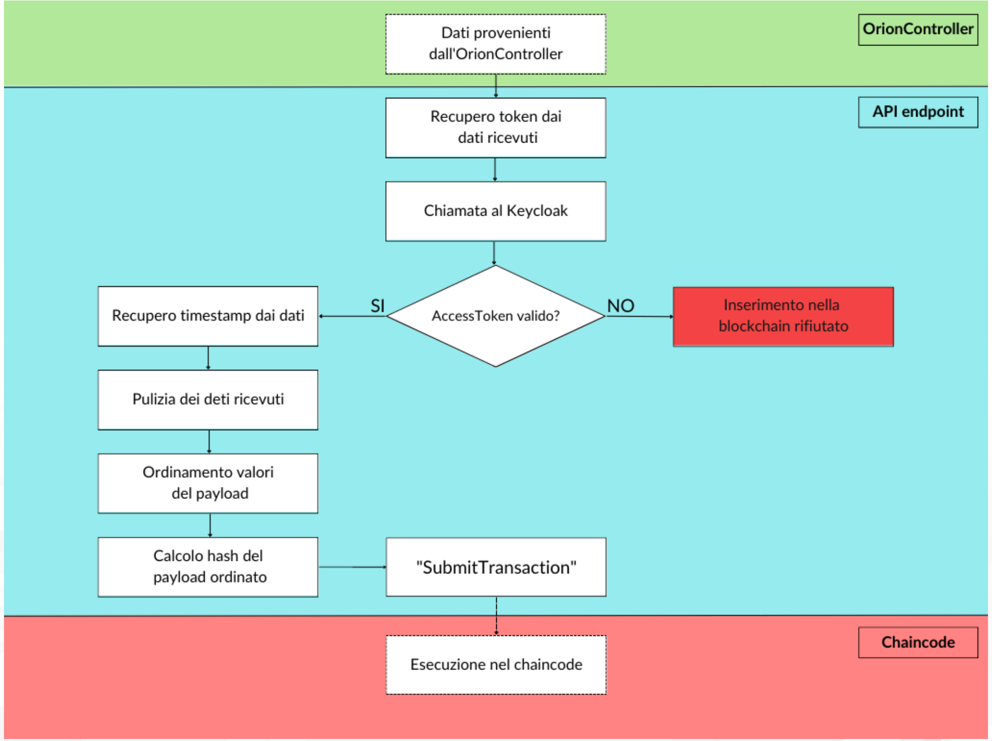
Why this operations are performed?

The removal of data types is to avoid the writing of useless data on the BC, when recovering data from the blockchain Snap4city doesn't provide data types so we don’t need them.

On the other hand the ordering is fundamental for the next step, the hashing.

Using a SHA256 function on the json we calculate an hash on the data, for this step is primary to have the data (temp,dateObserved,altitude) always in the same order to avoid different hashes in spite of being calculated using the same data.

Next, the api forwards some basic info about the device and the hash just calculated to the fabric chaincode (**red**) using the “Submit Transaction” function provided by the fabric SDK



OrionController

Dati provenienti dall'OrionController

API endpoint

Recupero token dai dati ricevuti

Chiamata al Keycloak

AccessToken valido?

SI

NO

Inserimento nella blockchain rifiutato

Recupero timestamp dai dati

Pulizia dei deti ricevuti

Ordinamento valori del payload

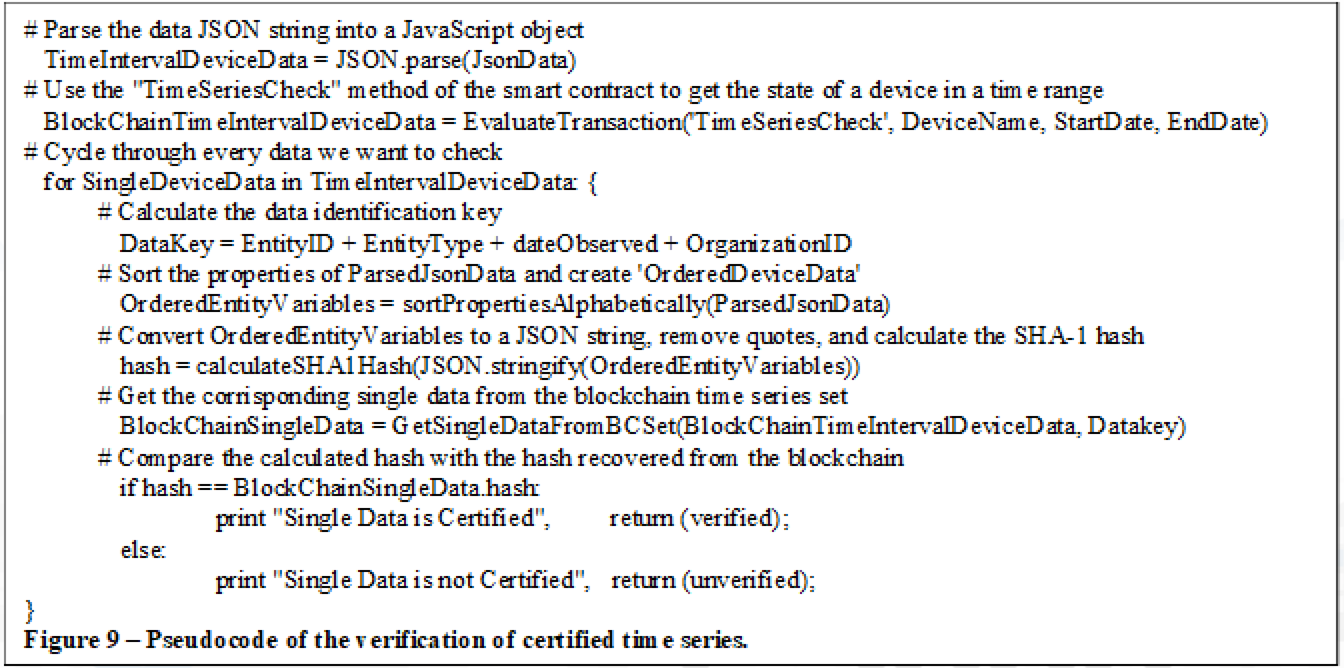
Calcolo hash del payload ordinato

"SubmitTransaction"

Chaincode

Esecuzione nel chaincode

Orion Filter -> API endpoint InDepth: Pseudocode



```
# Parse the data JSON string into a JavaScript object
  TimeIntervalDeviceData = JSON.parse(JsonData)
# Use the "TimeSeriesCheck" method of the smart contract to get the state of a device in a time range
  BlockChainTimeIntervalDeviceData = EvaluateTransaction('TimeSeriesCheck', DeviceName, StartDate, EndDate)
# Cycle through every data we want to check
  for SingleDeviceData in TimeIntervalDeviceData: {
        # Calculate the data identification key
          DataKey = EntityID + EntityType + dateObserved + OrganizationID
        # Sort the properties of ParsedJsonData and create 'OrderedDeviceData'
          OrderedEntityVariables = sortPropertiesAlphabetically(ParsedJsonData)
        # Convert OrderedEntityVariables to a JSON string, remove quotes, and calculate the SHA-1 hash
          hash = calculateSHA1Hash(JSON.stringify(OrderedEntityVariables))
        # Get the corrisponding single data from the blockchain time series set
          BlockChainSingleData = GetSingleDataFromBCSet(BlockChainTimeIntervalDeviceData, Datakey)
        # Compare the calculated hash with the hash recovered from the blockchain
          if hash == BlockChainSingleData.hash:
                    print "Single Data is Certified",      return (verified);
          else:
                    print "Single Data is not Certified",   return (unverified);
}
```

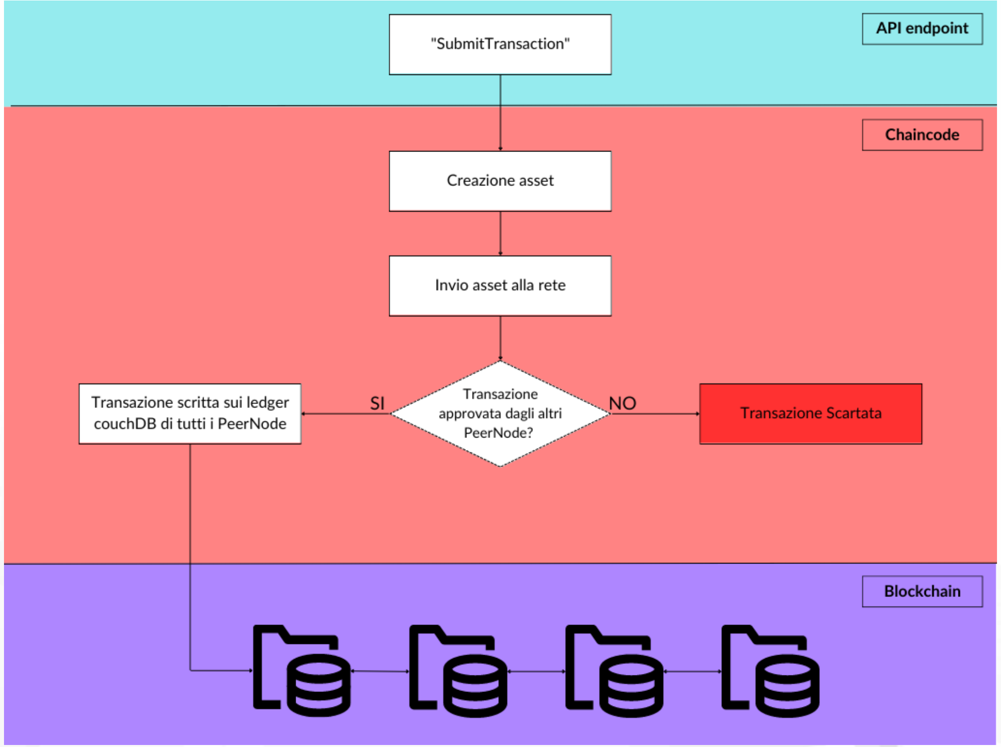
Figure 9 – Pseudocode of the verification of certified time series.

Blockchain abstraction layer (API endpoint -> Chaincode)

The chaincode after receiving data from the API endpoint, create an asset in a clever format to store information to CouchDB then using built-in functions sen the transaction proposal to the fabric network.

If the transaction is well built and then approved by the other peers the is written to the blockchain(**purple**).

Note that apart from basic information, no clear data is written on the blockchain. All the data related to measurements are encoded in the previously calculated hash function. Using this method, we cannot directly recover data. However, if we have certain data that we want to check for certification, we can calculate the hash of the data we possess and compare it to the hash stored on the blockchain. If they correspond, the data we have is certified.



Integration Benefits

Now that there has been presented in the previous sections how the Snap4City platform is structured, there has been studied a possible solution with the main goals of:

RQ1 Interoperability: Adding integration capabilities should be, however possible, to the current Snap4City structure respecting the existing bidirectional and modular architecture of Snap4City, leaving the user the possibility to use or not the blockchain.

RQ2 Ease of use: Since the blockchain is a recently developed technology, the user of the platform should have an interaction with it that is as simple as possible and fluid as possible without exposing it to implementation details.

RQ3 Performance: The certification on the blockchain should not worsen the current ones performance of Snap4City where they are essential for correct functioning of the platform since it has to handle a large amount of data and transactions real-time.

RQ4 Multi Blockchain Integration: the possibility of integrating multiple blockchains and from different architecture structures.

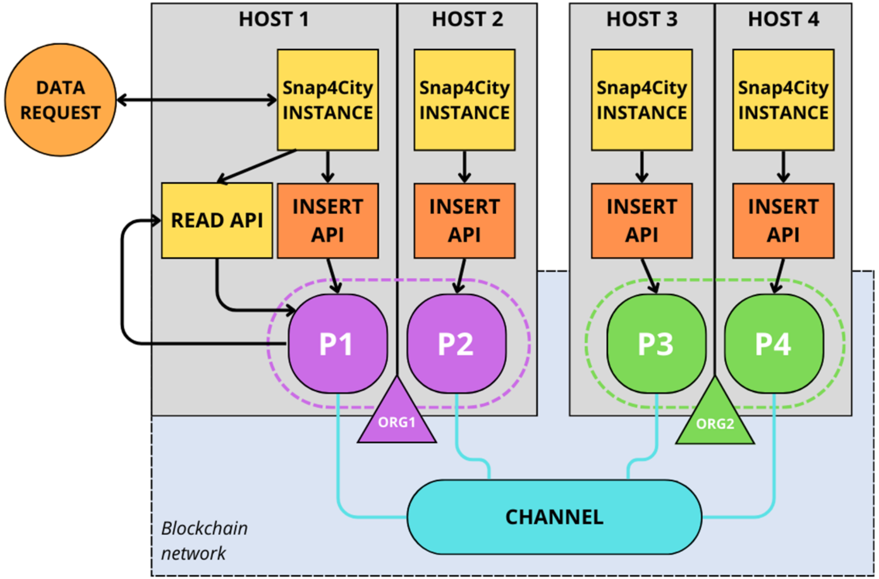
To assess this there has been designed an abstraction layer for the integration process of the normal procedures for IoT Models definition IoT device creation and IoT Device data storage and the corresponding storing actions on the blockchain. This has been realized using Node.js with Express [4] for the set of API endpoints that handle on the blockchain the operation of:

- IoT Device Model insertion and query.
- IoT Device insertion and query
- IoT Device Data entry and query.

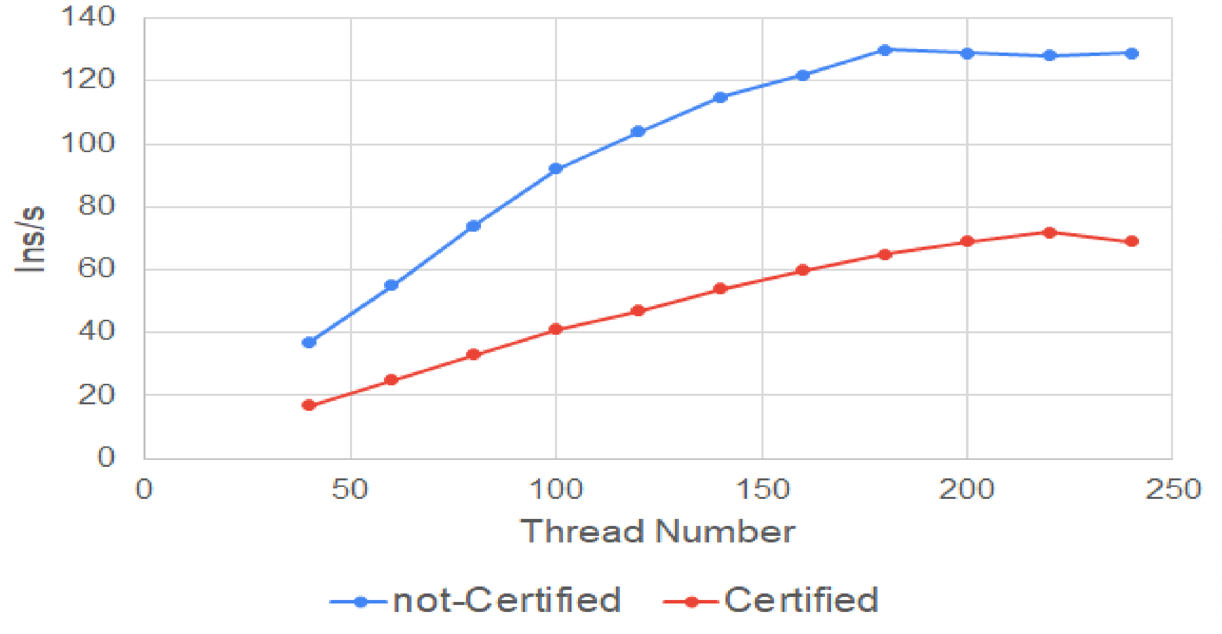
Performance Analysis

To assess the performance of the solution provided a deploy with 2 organizations distributed on 4 locations/hosts has been performed, as shown in Figure 10. The blockchain network has been based on 4 Snap4City platform instances distributed on 4 hosts respectively, with peers P1, P2 belonging to Org1, and P3, P4 belonging to Org2. Each Host had 24 cores at 2.1 GHz, 32 GB RAM.

Performance assessment has been performed for both tasks: certification at the insertion of Data Messages as Time Series (which is the most critical condition), and verification of Time Series.



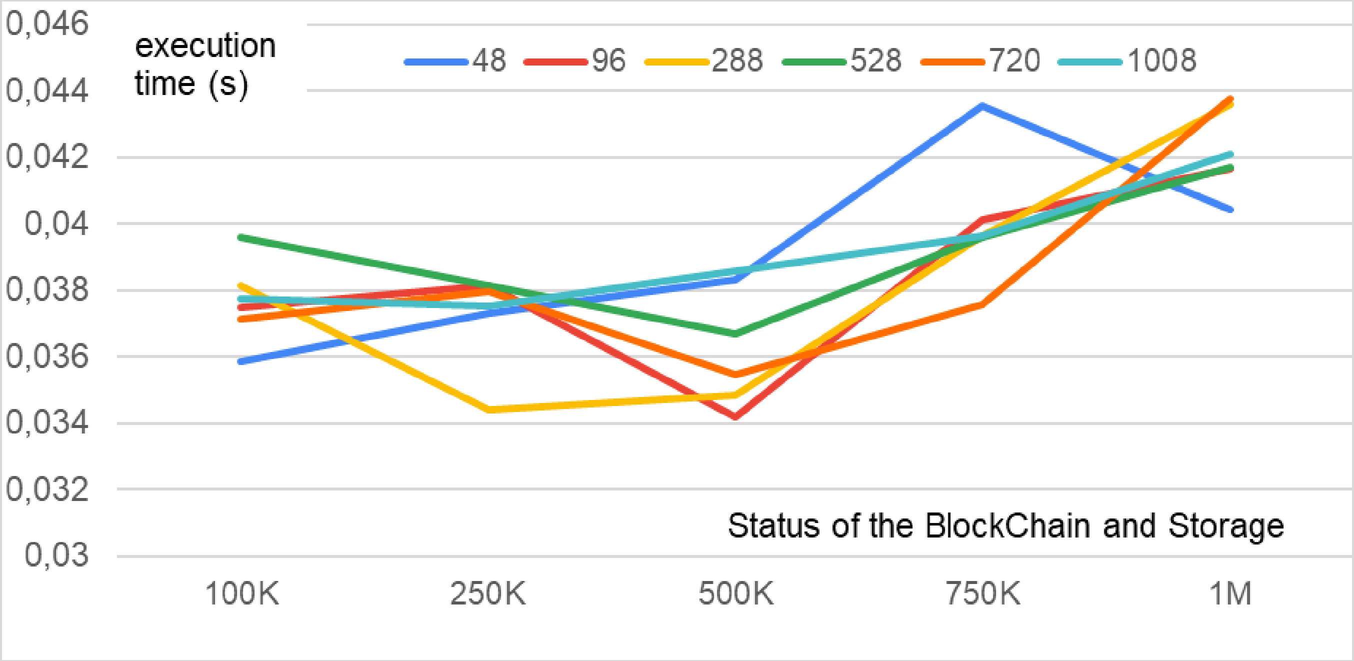
Certified/Non-Certified data Insertion



Certified/Non-Certified data Insertion

The platform reached its maximum acceptable workload when 240 Entities/threads provided messages at the same time, requesting the insertion of new data, obtaining the certification from the blockchain confirming that the has been saved in sufficient number of nodes/hosts; the CPU workload was at 65% and RAM memory at 11Gbyte, while the transfer rate on writing on HD turned out to be of 22Mbyte/s, which is far from the maximum allowed rate of the platform. Memory usage was relatively low, 20GB was used for about 50 active containers on Snap4City platform plus the Hyperledger Fabric. In the case of non-certified Entities/Data Messages the platform reached a max rate of 129ins/s (insertions per second) (which is equivalent to transactions per second presented in other papers), while with certification we observed a max rate of about 70 ins/s, which implies a reduction of performance of about the 46%.

Snap4city Blockchain verification performance



Snap4city Blockchain verification performance

All the experiments have been execution 5 times obtaining standard deviation in the different cases from 0,0012 to 0.0019. The registered degradation of performance was mainly due to the blockchain for about the 2% in passing from 100K to 1M. The incidence of recomputing hash on the Data Message stored in Snap4City storage and making comparison with respect to the recovered hash from blockchain cost about 0,64% of the execution time, while the mean execution time for retrieving Data Message (all variables) from Snap4City Storage from OpenSearch cost about 17,6%. All the rest, which is the 81,76% of the execution time has been due to the recovering of data from blockchain, which was based on CouchDB which has been used to store JSON in the chaincode namespace.

Appendix

Proof of Stake(PoS)

Proof of Stake (PoS) is a consensus algorithm used in blockchain networks to achieve consensus and validate transactions. Unlike Proof of Work (PoW), which relies on computational work, PoS selects validators based on the number of tokens or "stake" they hold in the network.

In a PoS system, validators are chosen to create new blocks and validate transactions based on their stake. The higher the stake a validator holds, the greater their chances of being chosen as the next validator.

CARDANO

Proof of stake step-by-step

How PoS works:

1. Validators and Stake: Participants in the network are validators who hold and "stake" a certain amount of the network's native cryptocurrency or tokens. The stake represents their ownership and interest in the network.
2. Block Creation and Validation: Validators are selected to create new blocks and validate transactions based on their stake. The probability of being chosen as a validator is proportional to the amount of stake they hold. Validators take turns creating blocks and including a set of valid transactions within them.
3. Consensus and Finality: Once a block is created and validated by the chosen validator, it is added to the blockchain. Consensus is achieved when the majority of validators agree on the validity of the block. This consensus process provides finality, meaning that once a block is added, it is unlikely to be reversed.
4. Rewards and Incentives: Validators are rewarded for their participation and contribution to the network. The rewards can come in the form of transaction fees or newly minted tokens. These incentives encourage validators to act honestly and in the best interest of the network. Validators also risk losing a portion of their stake if they behave maliciously or attempt to attack the network.

PoS pros and cons

PROS

PoS offers several advantages, including energy efficiency (as it doesn't require extensive computational work) and scalability. It also reduces the possibility of a 51% attack since an attacker would need to acquire a majority stake in the network to manipulate it.

CONS

Implementing PoS requires careful design and consideration of various factors, such as stake distribution and mechanisms to handle potential attacks.

Appendix

Smart Contracts

Smart contracts allow for the automation of the process of **verifying and enforcing the terms of an agreement**, and they can facilitate the ***exchange of assets*** without the need for intermediaries.

With smart contracts, parties can confidently enter into agreements with each other, knowing that the terms of the agreement will be enforced without the need for intermediaries such as banks or lawyers. This not only saves time and money but also increases the efficiency and transparency of transactions.

Smart contracts have many potential applications, from financial transactions and insurance policies to supply chain management and voting systems. They are a **key component of the second generation of blockchains,** enabling new types of decentralized applications and innovative business models.

Smart Contracts

Suppose a **fuel supply company wants to incentivize the use of more sustainable energy sources and reduce greenhouse gas emissions.** The company could create a smart contract that links <u>fuel prices</u> to the <u>carbon emissions produced by its production</u> and the degree to which the product is environmentally sustainable.

The contract could work like this: *each time a customer purchases a certain amount of fuel, the contract will check the amount of carbon emissions produced during production and determine the final price based on the product's level of eco-sustainability*.
For example, if the product was produced using renewable energy sources such as solar or wind power, the price might be lower than the price of a product produced using fossil energy sources.

Tokens

Tokens are often used as a means to incentivize certain behaviors within the network, such as contributing computing power or resources, or participating in decision-making processes. They can also be used to represent ownership of assets or access to services within the network.

- Cryptocurrencies are digital assets that are designed to work as a **medium of exchange**, just like traditional currencies.
- However, unlike traditional currencies that are issued and regulated by central authorities such as governments or central banks, cryptocurrencies are **decentralized and based on blockchain technology**.

Cryptocurrencies are created through a process called **mining**, which involves solving complex mathematical algorithms using powerful computers. This process verifies transactions on the blockchain and adds new blocks to the chain. Miners are rewarded with new units of the cryptocurrency for their work in maintaining the network.

Cryptocurrencies are stored in **digital wallets** and can be sent and received directly between users without the need for intermediaries such as banks or payment processors. Transactions are recorded on the blockchain, which is a decentralized ledger that is publicly accessible and transparent. This ensures the security and integrity of the network, as transactions cannot be altered or tampered with once they are recorded on the blockchain.

NFTs, or non-fungible tokens, are a type of digital asset that are unique. NFTs are often used to **represent ownership or proof of authenticity of digital content** such as artwork, music, videos, or even tweets. They are built on top of blockchain technology, and are stored in a decentralized manner, making them tamper-proof and transparent.

In a token economy, NFTs can be used as a means of exchange within a decentralized marketplace or ecosystem. For example, an artist could create a unique piece of digital art and sell it as an NFT. The NFT would then represent ownership of that specific piece of art, and the artist could choose to sell or trade the NFT as they wish. Similarly, a musician could release an album as an NFT, with each token representing ownership of a specific copy of the album.

이더리움

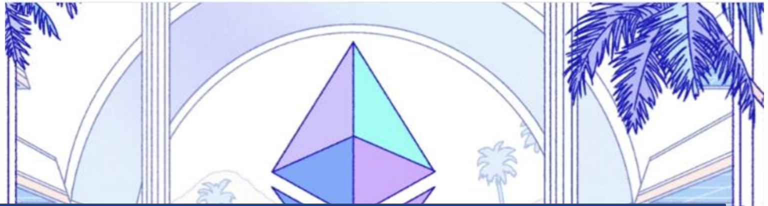
Ti diamo il benvenuto su Ethereum

Ethereum è la tecnologia gestita dalla community che alimenta la criptovaluta ether (ETH) e migliaia di applicazioni decentralizzate.

https://www.smh.com.au/business/entrepreneurship/they-re-digital-desirable-and-can-be-worth-millions-but-what-are-nfts-20220203-p59tfn.html

NFTs, or non-fungible tokens, are a type of digital asset that are unique. NFTs are often used to **represent ownership or proof of authenticity of digital content** such as artwork, music, videos, or even tweets. They are built on top of blockchain technology, and are stored in a decentralized manner, making them tamper-proof and transparent.

In a token economy, NFTs can be used as a means of exchange within a decentralized marketplace or ecosystem. For example, an artist could create a unique piece of digital art and sell it as an NFT. The NFT would then represent ownership of that specific piece of art, and the artist could choose to sell or trade the NFT as they wish. Similarly, a musician could release an album as an NFT, with each token representing ownership of a specific copy of the album.



Smart contracts, a set of rules that dictate how NFTs function, including providing their namesake non-fungibility and proof of ownership, and managing the transferability of the asset. The most commonly used contract that writes these rules is known as **ERC-721**, which includes a unique ID within the contract that cannot change and is written permanently into the blockchain.

Esplora Ethereum

https://www.smh.com.au/business/entrepreneurship/they-re-digital-desirable-and-can-be-worth-millions-but-what-are-nfts-20220203-p59tfn.html