Blockchain & Cryptoeconomics

DR. EVANGELOS POURNARAS

1. Introduction

Disclaimer

- **Do not use** the sample addresses & secret keys to perform real transactions. You may lose your money.
- **Do not use** any information provided in this module to make any investment decisions or transactions

DISC Lab

Dr. Evangelos Pournaras [Lecturer]: UKRI Future Leaders Fellow, Associate Professor, Alan Turing Fellow, Research Associate at UCL Centre of Blockchain Technologies, enterprise-ready blockchain industry experience

Email: e.pournaras@leeds.ac.uk

Dr. Rabiya Khalid [Assistant]: Research Fellow on Blockchain Systems

Email: R.Khalid@leeds.ac.uk

Communication: Teams direct message (preferred), or email

DISC Lab

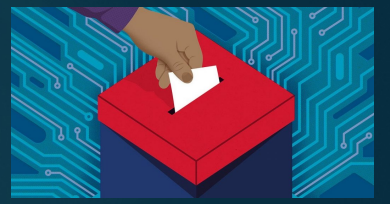
Lab: Distributed Intelligent Social Computing

Mission: *Provide new inter-disciplinary foundations of how to build & manage socio-technical systems supported by the Internet of Things & AI*

Applications: Smart Cities, Smart Grids, digital democracy

Research projects:

- Digitally-assisted Collective Governance of Smart City Commons-ARTIO, UKRI Future Leaders
- Trust & legitimation in the digital democracy, NRP77 SNSF
- Socially Responsible AI for Distributed Autonomous Systems, White Rose Collaboration Project
- Innovative Integrated Tools & Technologies to Protect & Treat Drinking Water from Disinfection By products, Horizon Europe




Module Objectives

To develop a practical understanding & skills for distributed ledgers & cryptoeconomic systems, to design & develop decentralized applications (smart contracts) running on the blockchain.

Expected learning outcomes:

- Understand the design of distributed ledgers & cryprtoeconomic systems
- Understand the context to which distributed ledgers are applied
- Analyze performance trade-offs between the different distributed ledgers
- Design decentralized applications & implement smart contracts

Syllabus

Introduction

Historical perspective

Finance & economics background

Blockchain systems

Cryptography

Bitcoin & Ethereum

Smart contracts

Consensus Algorithms

Blockchain with IoT and AI

Self-governance and Future Perspectives

Prerequisite Qualifications

- Computer networks
- Programming & software engineering
- Operating Systems, UNIX

Module Delivery

Learning material: Minerva, slides + labs, references

Grade: 60% for coursework [individual] & 40% for exam

Coursework: practicing smart contracts development

- **Release**: 27.3.2023
- **Introduction**: 31.3.2023
- **Submission**: 2.5.2023

Key for success

1. Stay proactive, ask questions, follow with the uploaded material
2. Participate & engage in the class & online
3. Be organized, plan & schedule your learning for all your modules
4. Be curious, follow the material, structure it, make notes, etc.

Your Voice: Why Blockchain?

Why did you choose this module?

What would you like to learn?

How would you like to use the knowledge & skills in the future?

Questions?

Blockchain & Cryptoeconomics

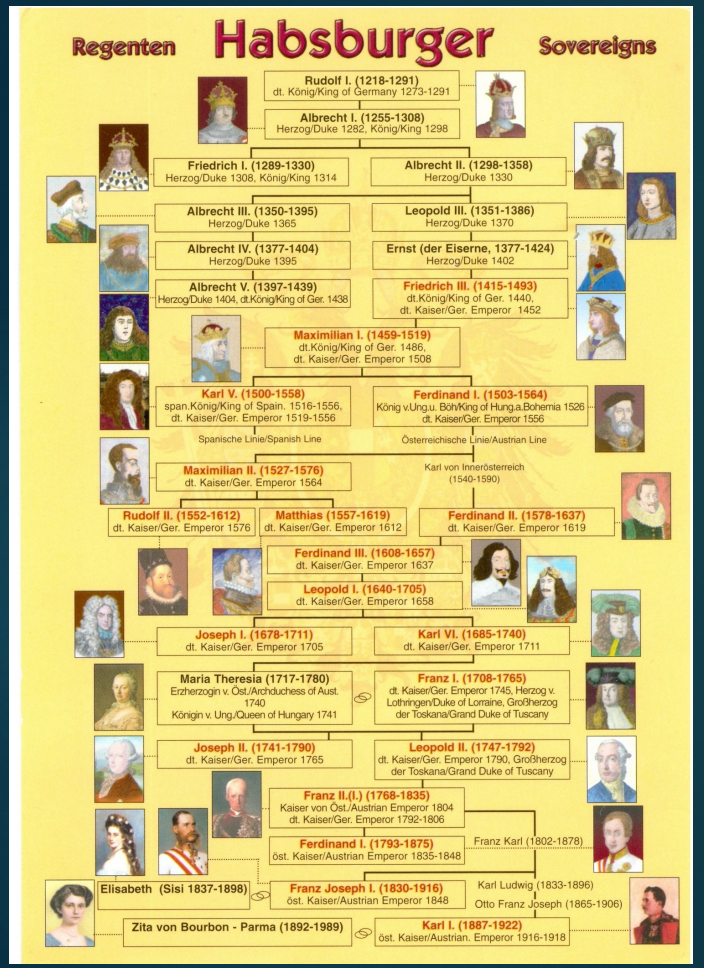
DR. EVANGELOS POURNARAS

2. Historical Perspective: Money & Ledgers

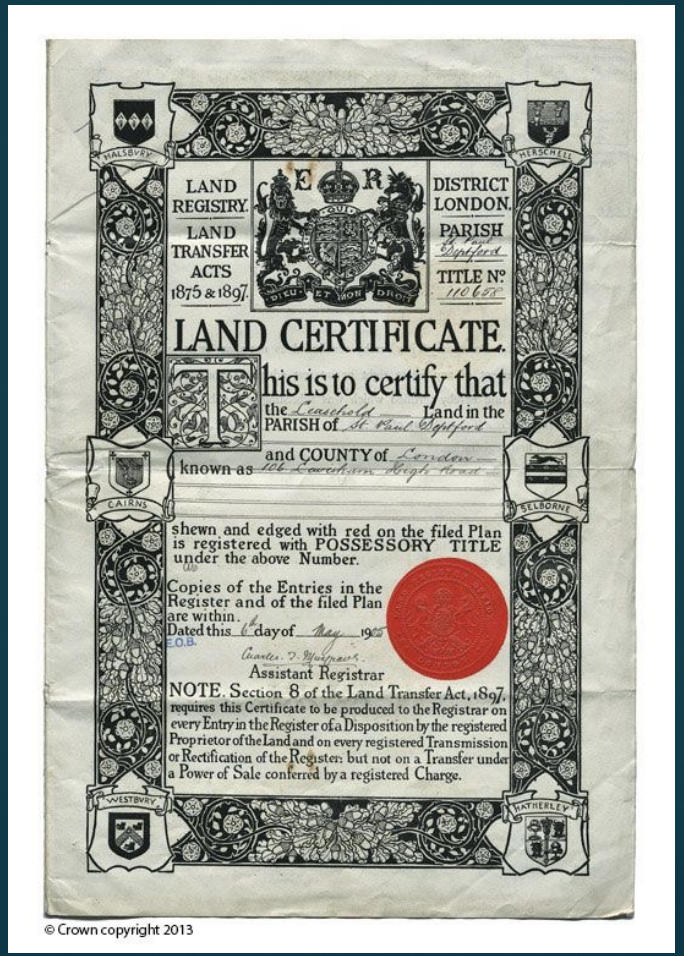
Earlier Ledgers & Miners

- **7th century** B.C.**:** Lydians & Greeks create standard coinage.
- **14th century:** Merchant banks (e.g. Medicis) expand involvement in multi- state finance, trade & manufacturing.
- **17th century:** By loaning out the value of deposited money, bankers increase economic productivity while creating new sources of risk that regularly result in local crashes & even wide-spread depressions. Central banks emerge, linking banking with taxation.
- **18th century:** The gold standard evolves from previous tactics in which circulating money was loosely controlled by a reserve of precious metals. This lowers risk.
- **20th century:** The gold standard is replaced by the Basel Accords, which say that holding easily sold assets is just as good as holding gold.

Earlier Ledgers & Miners



Genealogical Trees


LAND REGISTRY
LAND TRANSFER ACTS 1875 & 1897

DISTRICT LONDON
PARISH St Paul Deptford
TITLE No 110655

LAND CERTIFICATE

This is to certify that the Leasehold Land in the PARISH of St Paul Deptford and COUNTY of London known as 101 Lewisham High Road

shewn and edged with red on the filed Plan is registered with POSSESSORY TITLE under the above Number.

Copies of the Entries in the Register and of the filed Plan are within.

Dated this 6th day of May 1905

E.O.B.

Assistant Registrar

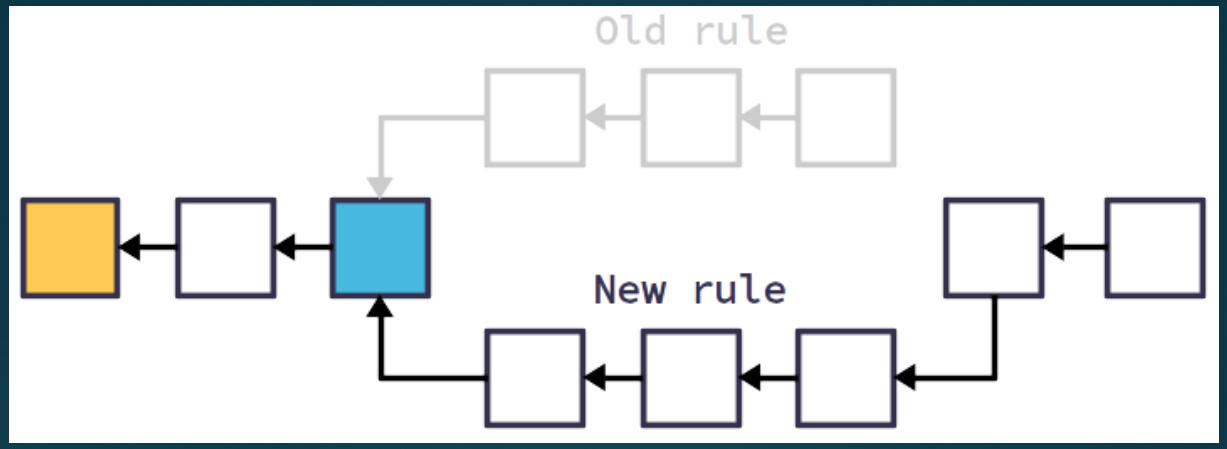
NOTE. Section 8 of the Land Transfer Act, 1897, requires this Certificate to be produced to the Registrar on every Entry in the Register of a Disposition by the registered Proprietor of the Land and on every registered Transmission or Rectification of the Register; but not on a Transfer under a Power of Sale conferred by a registered Charge.

© Crown copyright 2013

Land Registry Title Deeds

Verification by Notaries

Rules of the Game: Disagree?



Forks

Hundred Years War

War of Austrian Succession

Money as Physical Objects

Chinese cowry shell

Lydian electrum coin

Tally sticks

Athenean tetradrachm

Stone money

Clay tablet

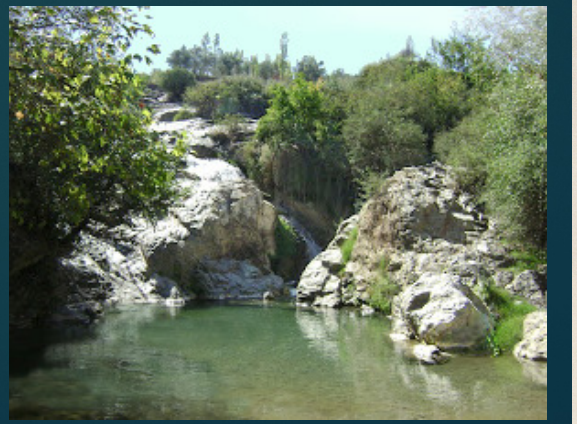
Myths: From Midas to Crypto

Pactolus river in Turkey is very rich in electrum, the basis of the economy in ancient state of Lydia. How?

Dionysys gave to King Midas the gift of turning everything he touches into gold.

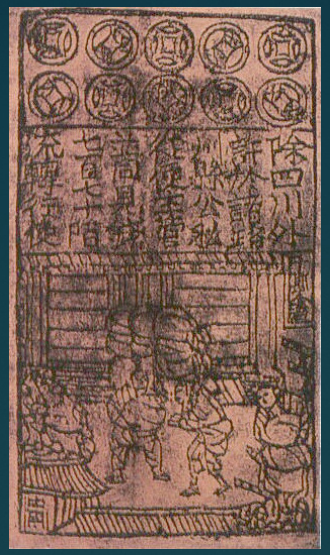
But this proved to be a curse that could only get rid of once he washed Dionysus' gift away in the river Pactolus.

How do the nowadays cryptocurrencies recall the story of Midas?

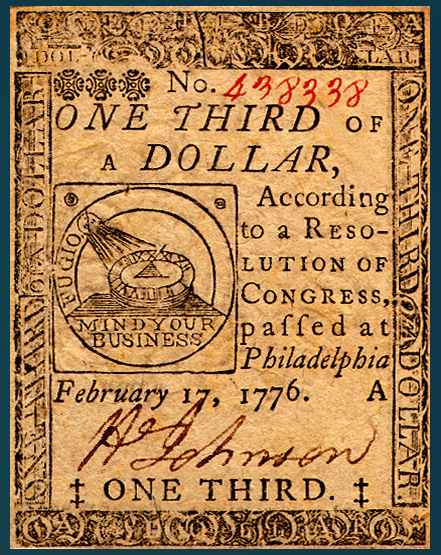


Paper Money

- No intrinsic value
- A less costly instrument of commerce
- Convenient
- Easier to circulate, less costly to produce & maintain
- Difficult to replicate by a third party
- Still subject of debate: when & how often to print?



11th century Chinese paper money



Continental dollar

Stockholm Banco in 1666

Soviet chevronets

1805 British pound

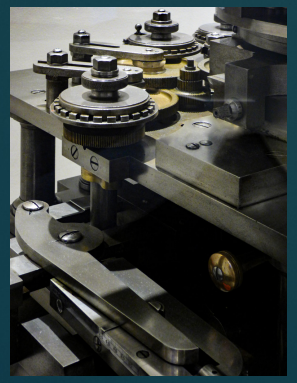
Earlier Anti-counterfeiting Measures

Counterfeiting is as old as money

Often punishable by death

Anti-counterfeiting measures:

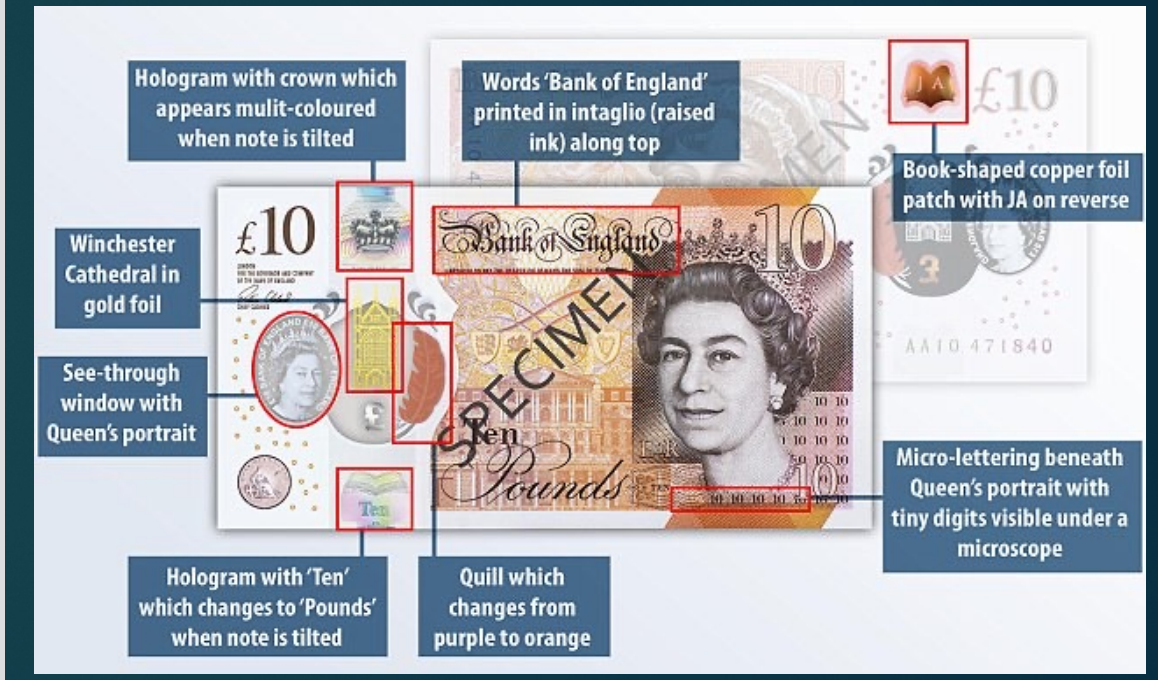
- Specially prepared paper (acoustic characteristics)
- Protection strip
- Hologram
- Flatbed printing plates
- Nature prints: patters of real leaves
- Geometric lathe
- Polymer banknotes



Coin debasement

- Clipping
- Sweating
- Plugging

Security Features of Banknotes



Protecting German Marks Notes

German banknote serial number:

- alphanumerical code, numbers & letters
- Unique per banknote
- Mathematical principle/algorithm for calculation is secret
- **Check digit**: redundancy check used for error detection on identification numbers (binary parity bit)

DA794310OZ7
Letters to numbers:

A	D	G	K	L	N	S	U	Y	Z
0	1	2	3	4	5	6	7	8	9

$a_1a_2a_3a_4a_5a_6a_7a_8a_9a_{10}a_{11}$: 10784310097

a_{11}: check digit, calculated such that the checksum:

$\sum_{n=1}^{11} \pi^{n \bmod 11}(a_n)=$

$\pi(a_1) \cdot \pi^2(a_2) \cdot \ldots \cdot \pi^{10}(a_{10}) \cdot a_{11} = 0$

where each permutation is defined as:

$$\pi = \begin{pmatrix} 0\,1\,2\,3\,4\,5\,6\,7\,8\,9 \\ 1\,5\,7\,6\,2\,8\,3\,0\,9\,4 \end{pmatrix}$$

& the associative operation " · " as on the table:

·	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	0	6	7	8	9	5
2	2	3	4	0	1	7	8	9	5	6
3	3	4	0	1	2	8	9	5	6	7
4	4	0	1	2	3	9	5	6	7	8
5	5	9	8	7	6	0	4	3	2	1
6	6	5	9	8	7	1	0	4	3	2
7	7	6	5	9	8	2	1	0	4	3
8	8	7	6	5	9	3	2	1	0	4
9	9	8	7	6	5	4	3	2	1	0

Earlier Anti-counterfeiting Measures

$\pi(1) \cdot \pi^2(0) \cdot \pi^3(7) \cdot \pi^4(8) \cdot \pi^5(4) \cdot \pi^6(3) \cdot \pi^7(1) \cdot \pi^8(0) \cdot \pi^9(0) \cdot \pi^{10}(9) \cdot 7 = 0$

$\pi(1)$: $1 \rightarrow 5$

$\pi^2(0)$: $0 \rightarrow 1 \rightarrow 5$

$\pi^3(7)$: $7 \rightarrow 0 \rightarrow 1 \rightarrow 5$

$\pi^4(8)$: $8 \rightarrow 9 \rightarrow 4 \rightarrow 2 \rightarrow 7$

$\pi^5(4)$: $4 \rightarrow 2 \rightarrow 7 \rightarrow 0 \rightarrow 1 \rightarrow 5$

$\pi^6(3)$: $3 \rightarrow 6 \rightarrow 3 \rightarrow 6 \rightarrow 3 \rightarrow 6 \rightarrow 3$

$\pi^7(1)$: $1 \rightarrow 5 \rightarrow 8 \rightarrow 9 \rightarrow 4 \rightarrow 2 \rightarrow 7 \rightarrow 0$

$\pi^8(0)$: $0 \rightarrow 1 \rightarrow 5 \rightarrow 8 \rightarrow 9 \rightarrow 4 \rightarrow 2 \rightarrow 7 \rightarrow 0$

$\pi^9(0)$: $0 \rightarrow 1 \rightarrow 5 \rightarrow 8 \rightarrow 9 \rightarrow 4 \rightarrow 2 \rightarrow 7 \rightarrow 0 \rightarrow 1$

$\pi^{10}(9)$: $9 \rightarrow 4 \rightarrow 2 \rightarrow 7 \rightarrow 0 \rightarrow 1 \rightarrow 5 \rightarrow 8 \rightarrow 9 \rightarrow 4 \rightarrow 2$

$5 \cdot 5 \cdot 5 \cdot 7 \cdot 5 \cdot 3 \cdot 0 \cdot 0 \cdot 1 \cdot 2 \cdot 7$

$= (5 \cdot 5) \cdot (5 \cdot 7) \cdot (5 \cdot 3) \cdot (0 \cdot 0) \cdot (1 \cdot 2) \cdot 7$

$== 0 \cdot 3 \cdot 7 \cdot 0 \cdot 3 \cdot 7$

$= (0 \cdot 3) \cdot (7 \cdot 0) \cdot (3 \cdot 7)$

$= 3 \cdot 7 \cdot 5 = (3 \cdot 7) \cdot 5$

$= 5 \cdot 5$

$= 0$

Exercises

Check whether the following serial numbers of German bank notes are fake or not:

- GG6414493L3
- DA7843100Z8
- AD5203416U6

Questions?

Blockchain & Cryptoeconomics

DR. EVANGELOS POURNARAS

2. Finance & Economics Background

The Role of Money

- Medium of exchange
- Mean of payments
- Store of value
- Unit of account
- Perpretual need for good & services

Historically, money usually becomes anything used to discharge taxes

Monetary vs. barter economy:

- Money has to be represented by a token
- Money has to be accepted as a final settlement of all transactions: terminating credit/debit relationships between parties
- Money should not grant priviledges of seigniorage to any agent making a payment: requiring the presence of a bank as a third party to any non-cash transaction

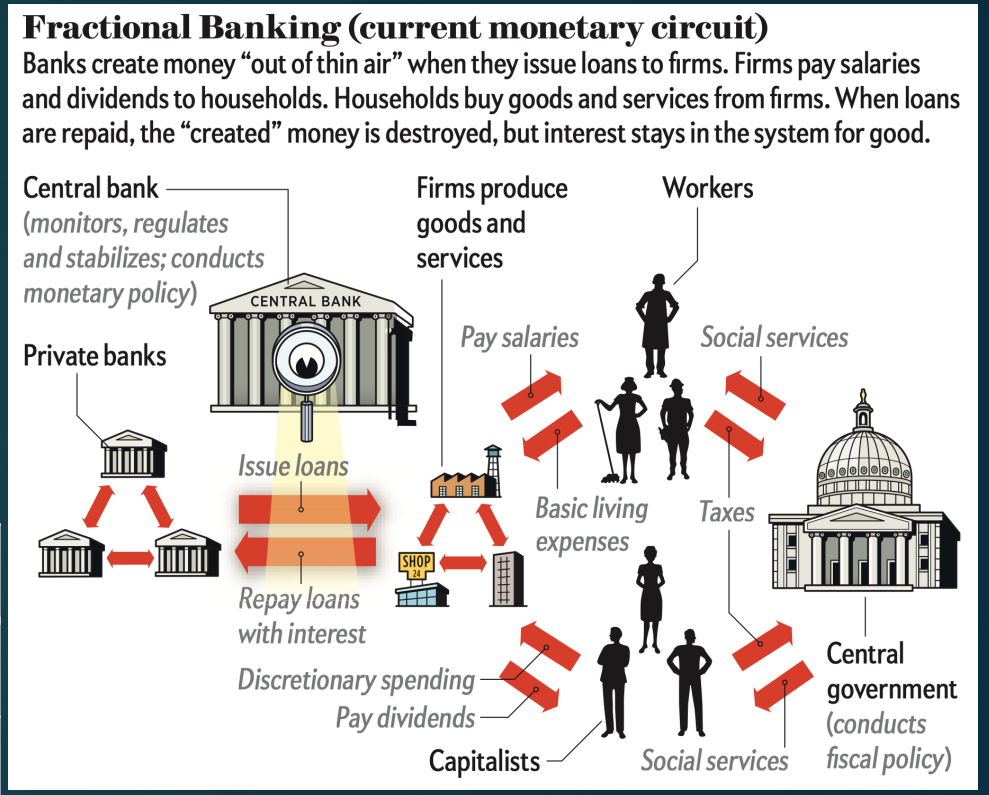
Requirements for Electronic Money

- **Online payment**: can be securely used online
- **Offline payment**: can be securely utilized offline
- **Non-reproducibility**: cannot be copied & reused
- **Anonymity/pseudonymity**: no reveal of identities in transactions
- **Transferabolity**: can be transferted among parties
- **Divisibility**: can be subdivided as necessary

Origin of Money: Theories

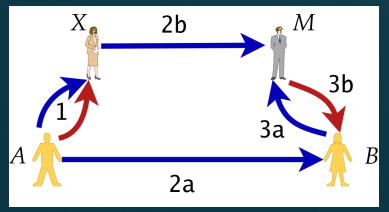
- Credit creation theory of money
- Fractional reserve theory of money
- Financial intermediation theory of money
- (Modern) Monetary circuit theory

	Central Bank	Government	Private Banks	Firms	Households
Central Bank		loans + interests			
Government			interests	consumables	social services
Private Banks		loans + taxes		loans + interests	Interests + dividents
Firms		taxes	interests		wages
Households		taxes		consumables	



Overview of the Financial System

- **Money**: obligation of private banks
- **Cash**: obligation of central bank
- Banks need sufficient capital cushions & liquidity to avoid defaults
- **Federal Deposit Insurance Corporation**: alleviates the banks default risk for small deposits
- **Loans**: follow dynamics of supply-demand equilibrium
- **Capacity of loans**:
  - Banking capital & liquidity
  - Collaterals

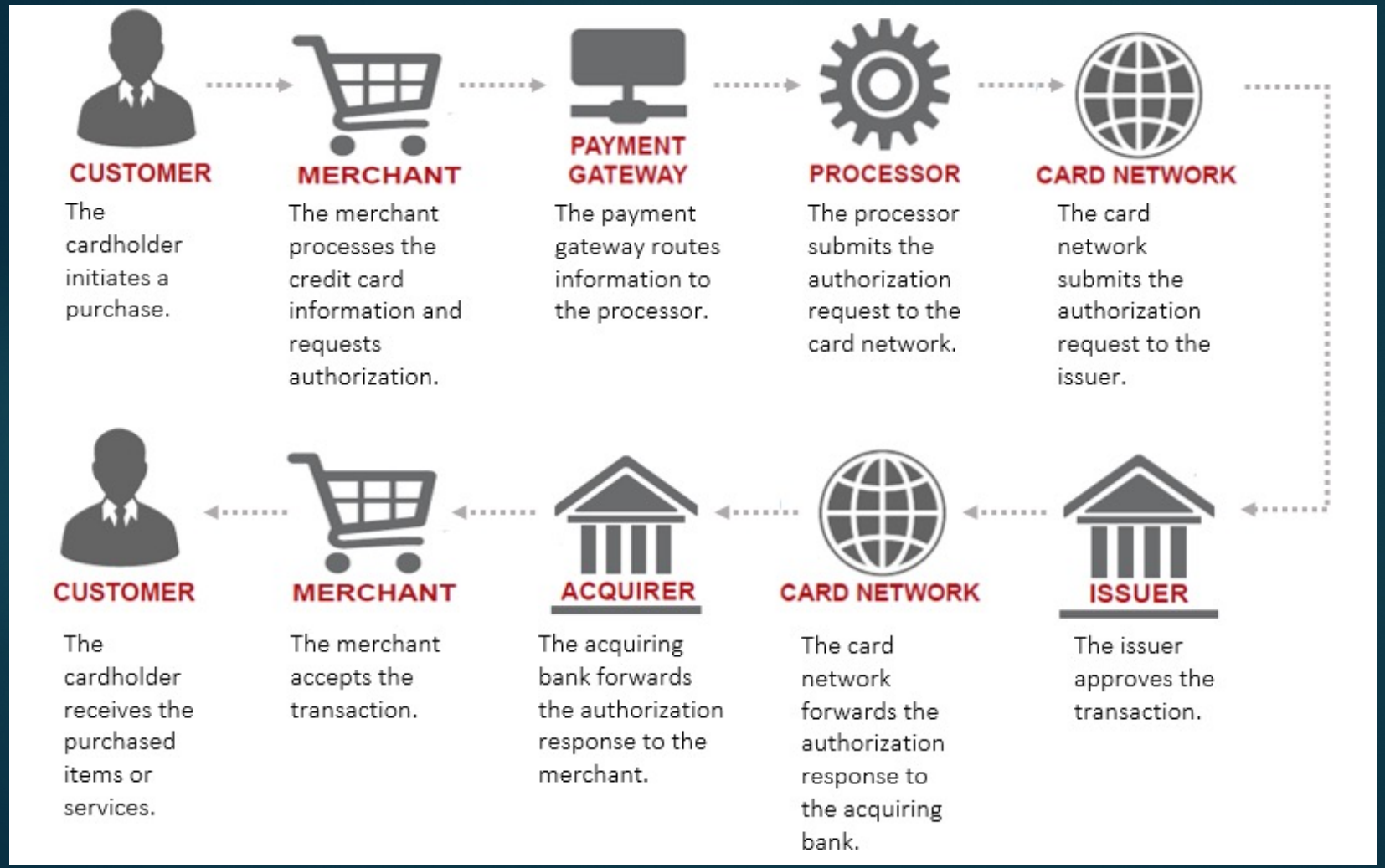


- Banks compete with & help each other to balance cash holdings
  - Interbank linkages with risk
- How banks support central bank:
  - Know your Customer (KYC) services
  - Anti-Money Laundering (AML) services
- Credit card payments: very complex, inefficient & costly
- SWIFT messaging network for banks communication
- Forex trading: exchanges to foreign currencies, moving money across borders
- Hawala: informal value transfer system

Monetary System Challenges

- Negative interest rates
- Currency increases but its velocity decreases: money is used less efficiently
- Bank reserves increase, lending opportunities decrease
- Too many intermediaries
- Information asymmetry creating heavy concetration of wealth

Credit Card Transaction Lifecycle



Ledgers

Non-banking firm:

- $A = L + K$

Banking:

- $A + A' + C = L + L' + K$

A: assets

A': interbank assets

C: central bank cash

L: liabilities

L': central bank liabilities

K: equity/capital

Money Creation

Scenario: 1 bank, 1 borrower, default & no default of the borrower

Assumption: bank can lend money as it does not operate at full capacity of capital & liquidity

	State 1	State 2: Borrowing 4M	State 3a: No Default	State 3b: Default
External Assets	50	54	50	50
Interbank Assets	20	20	20	20
Cash	10	10	10.5	10
External Liabilities	60	64	60	64
Interbank Liabilities	13	13	13	13
Capital	7	7	7.5	3
Assets - Liabilities	**0**	**0**	**0**	**0**

Creating money "out of thin air"
but borrower possesses collateral

Money Creation

Scenario: 2 banks & central bank, 2 borrowers, default & no default of the borrower

Assumption: liquidity is important, bank assets are reserves (cash, liability of central bank)

	State 1	State 2: Individual borrows 4M from Bank 1	State 3: Bank 1 borrows 4M from Bank 2	State 4a: individual borrower repays	State 5a: Bank 1 repays Bank 2	State 4b: individual borrower defaults	State 5b: Bank 1 repays Bank 2
	Bank 1	Bank 1	Bank 1	Bank 1	Bank 1	Bank 1	Bank 1
External Assets	50	54	54	50	50	50	50
Interbank Assets	20	20	20	20	20	20	20
Cash	10	6	10	14.5	10.25	10	5.75
External Liabilities	60	60	60	60	60	60	60
Interbank Liabilities	13	13	17	17	13	17	13
Capital	7	7	7	7.5	7.25	3	2.75
Assets - Liabilities	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	Bank 2	Bank 2	Bank 2	Bank 2	Bank 2	Bank 2	Bank 2
External Assets	80	80	80	80	80	80	80
Interbank Assets	15	15	19	19	15	19	15
Cash	15	19	15	11	15.25	11	15.25
External Liabilities	70	74	74	70	70	70	70
Interbank Liabilities	30	30	30	30	30	30	30
Capital	10	10	10	10	10.25	10	10.25
Assets - Liabilities	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Questions?

Blockchain & Cryptoeconomics

DR. EVANGELOS POURNARAS

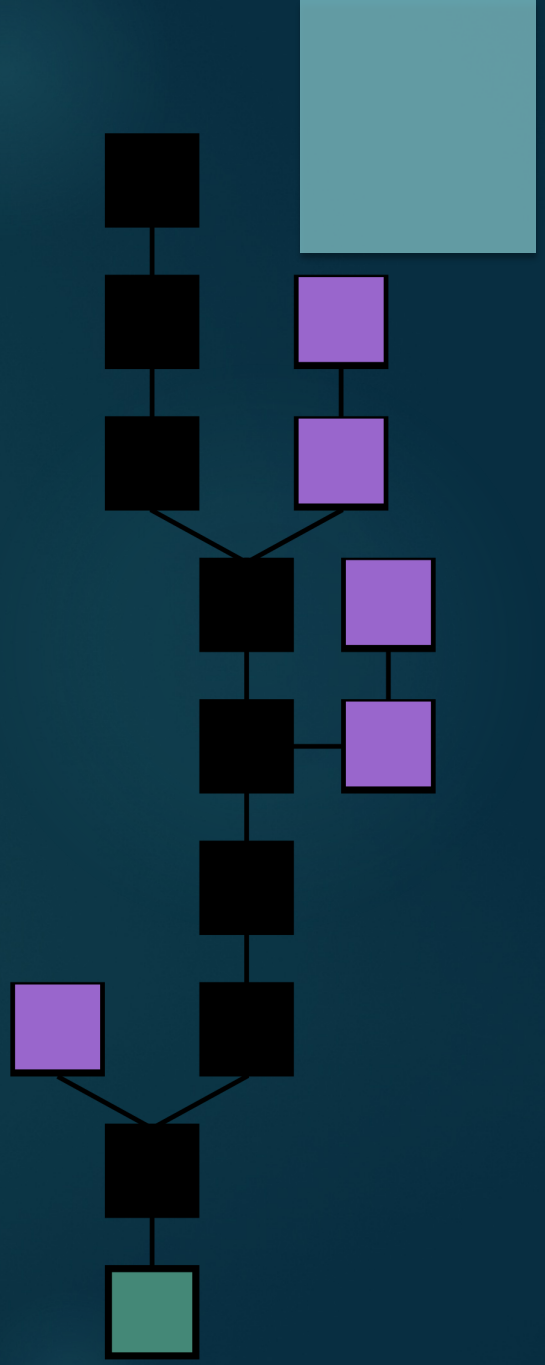
4. Blockchain Systems

Blockchain in a Nutshell

Blockchain is a shared distributed ledger designed to **record transactions & track changes in assets ownership**

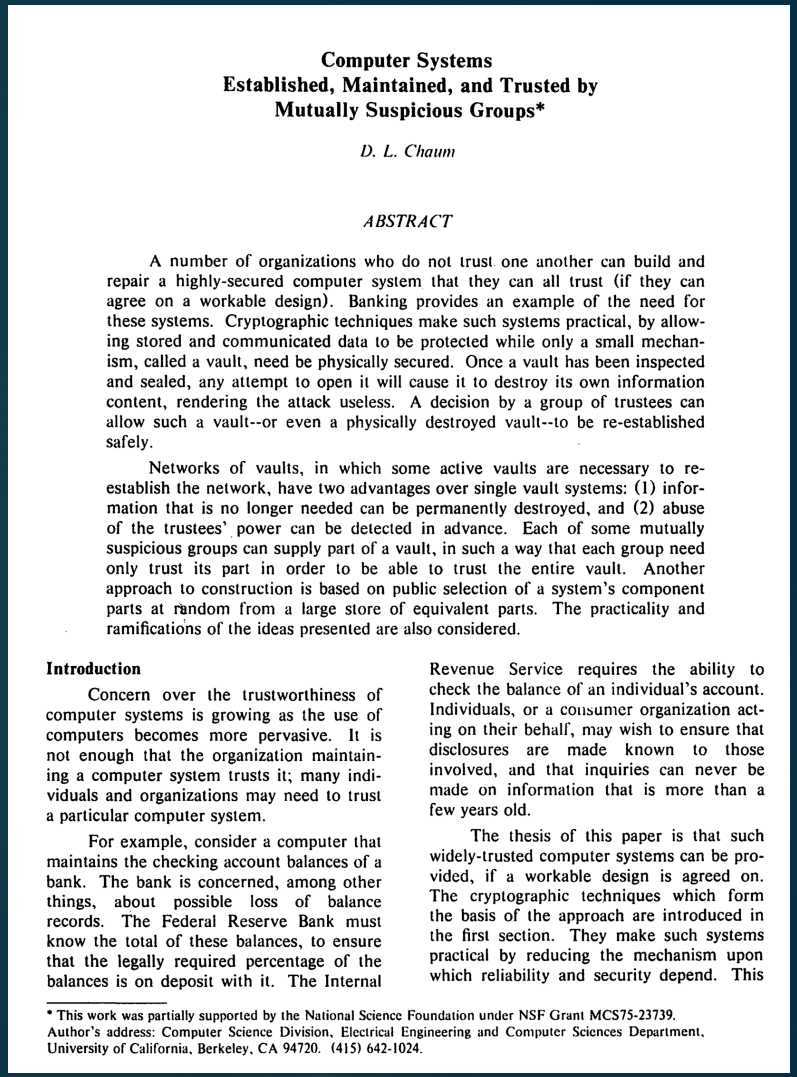
Assets:

- **Tangible**: e.g. money, shares, real estate
- **Intangible**: intellectual property, patents, trademarks, copyrights, goodwill, brand recognition



The “Ingredients” were there

- Digital signatures
- Merkle trees
- Chaining
- Proof-of-work based on cryptographic hash functions
- Proof-of-work based on hashcash



Computer Systems Established, Maintained, and Trusted by Mutually Suspicious Groups*

D. L. Chaum

ABSTRACT

A number of organizations who do not trust one another can build and repair a highly-secured computer system that they can all trust (if they can agree on a workable design). Banking provides an example of the need for these systems. Cryptographic techniques make such systems practical, by allowing stored and communicated data to be protected while only a small mechanism, called a vault, need be physically secured. Once a vault has been inspected and sealed, any attempt to open it will cause it to destroy its own information content, rendering the attack useless. A decision by a group of trustees can allow such a vault--or even a physically destroyed vault--to be re-established safely.

Networks of vaults, in which some active vaults are necessary to re-establish the network, have two advantages over single vault systems: (1) information that is no longer needed can be permanently destroyed, and (2) abuse of the trustees' power can be detected in advance. Each of some mutually suspicious groups can supply part of a vault, in such a way that each group need only trust its part in order to be able to trust the entire vault. Another approach to construction is based on public selection of a system's component parts at random from a large store of equivalent parts. The practicality and ramifications of the ideas presented are also considered.

Introduction

Concern over the trustworthiness of computer systems is growing as the use of computers becomes more pervasive. It is not enough that the organization maintaining a computer system trusts it; many individuals and organizations may need to trust a particular computer system.

For example, consider a computer that maintains the checking account balances of a bank. The bank is concerned, among other things, about possible loss of balance records. The Federal Reserve Bank must know the total of these balances, to ensure that the legally required percentage of the balances is on deposit with it. The Internal Revenue Service requires the ability to check the balance of an individual's account. Individuals, or a consumer organization acting on their behalf, may wish to ensure that disclosures are made known to those involved, and that inquiries can never be made on information that is more than a few years old.

The thesis of this paper is that such widely-trusted computer systems can be provided, if a workable design is agreed on. The cryptographic techniques which form the basis of the approach are introduced in the first section. They make such systems practical by reducing the mechanism upon which reliability and security depend. This

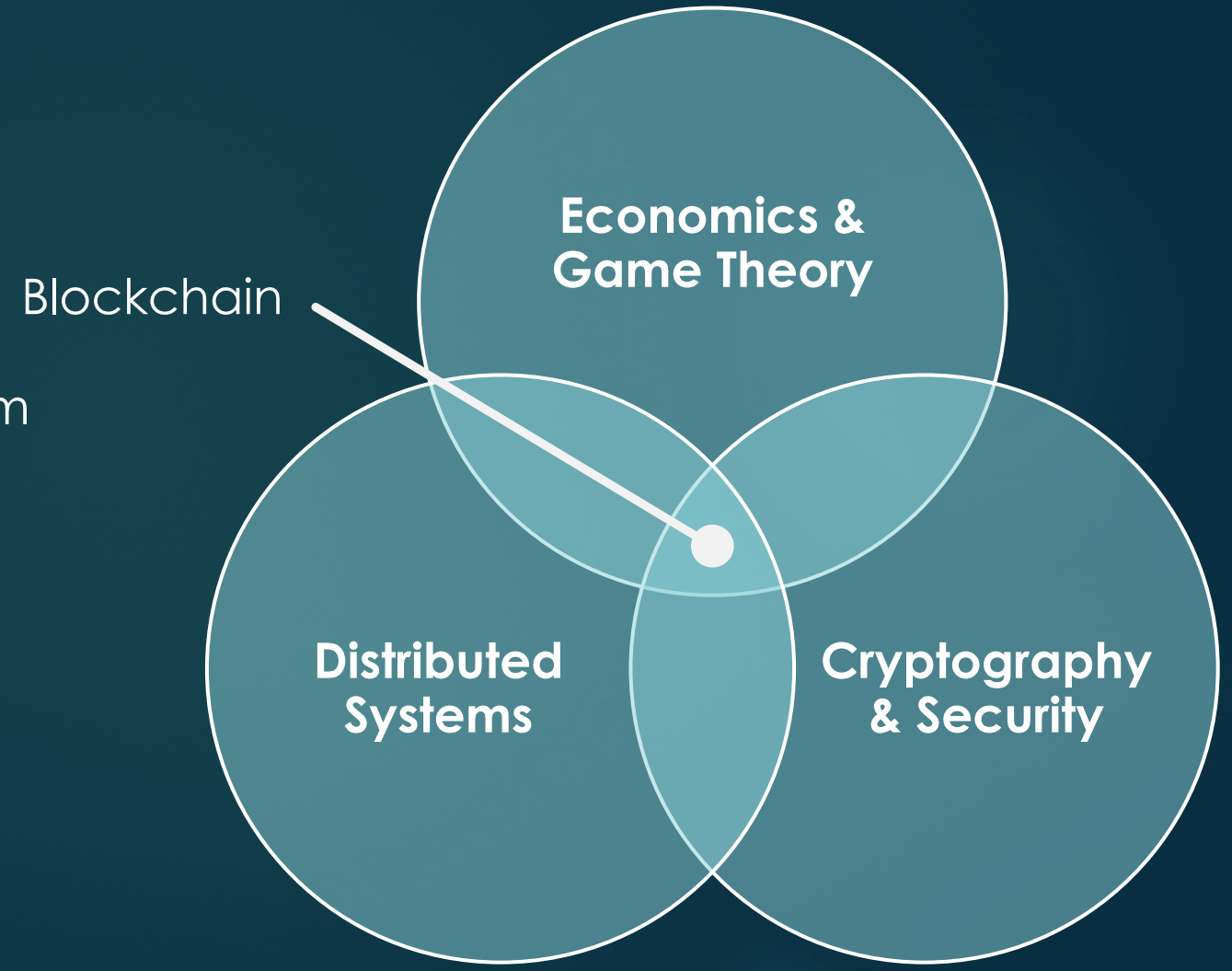
* This work was partially supported by the National Science Foundation under NSF Grant MCS75-23739.
Author's address: Computer Science Division, Electrical Engineering and Computer Sciences Department, University of California, Berkeley, CA 94720. (415) 642-1024.

What Means to have a Blockchain

- **Endogenous maintenance** of a ledger by its own users
- **No preliminary permission** requirement
- **A complete audit trail** of every transaction
- **Free accessible view** over the complete accounting
- **Highly resilient** to several attacks, can enforce protocol rules
- **No single point of trust**
- **Incentivized collaborative behavior**, punishing disruptive actors despite anonymity

Why Blockchain is so Complex?

- A techno-socio-economic system
- Large-scale & dynamic
- Inter-disciplinary & fragmented knowledge foundations

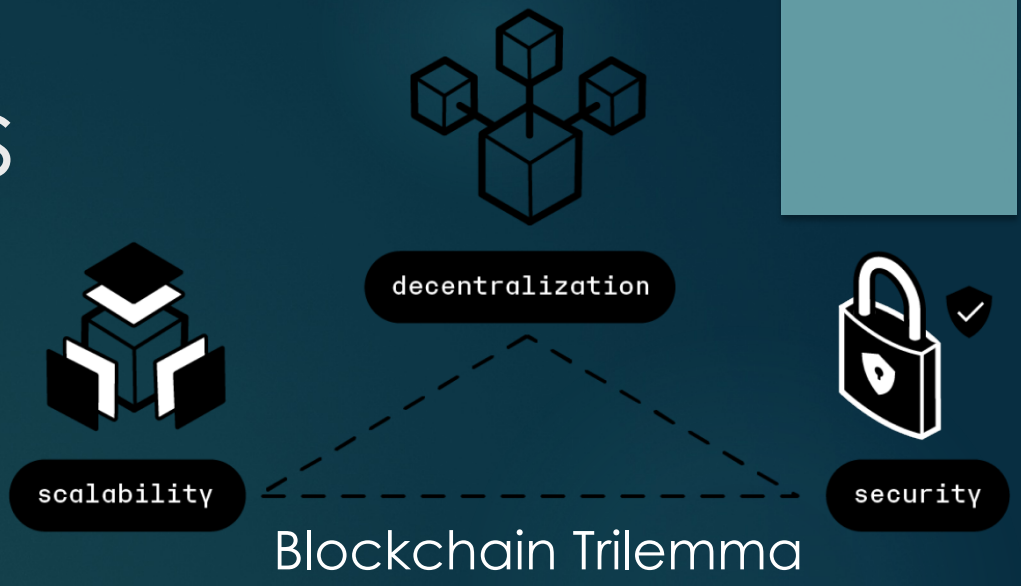


Benefits of Blockchain

- **Risks reduction** – higher business interoperability
- **Costs reduction** – no intermediaries
- **Reorganized business modus operandi**: digital identities & networking
- **Increased transactional trustworthiness**
- **Transparency**
- **Reliability**
- **Improvement of data quality & accuracy**
- **Reduction of fraud & cybercrime**

Addressed Challenges

- Trust on peers that **arbitrary join & leave anonymously**?
- Open & operate an account **without a bank?**
- **Demonstrate account ownership** under anonymity & lack of legal identities?
- **Where to store the ledger** without trusted data store provider?
- **Avoid data tampering** in presence of dishonest participants & without a trusted auditor?
- **How to regulate monetary policy** without a central bank?



Blockchain Trilemma

The Difficulty in Scaling Blockchains: A Simple Explanation

Maarten van Steen
University of Twente, The Netherlands
m.r.vansteen@utwente.nl

Andrew A. Chien
University of Chicago, USA
achien@cs.uchicago.edu

Patrick Eugster
Università della Svizzera italiana (USI), Switzerland
patrick.thomas.eugster@usi.ch

Abstract—Blockchains have become immensely popular and are high on the list of national and international research and innovation agenda's. This is partly caused by the numerous interesting applications, combined with the promise of full decentralization and high scalability (among others). Keeping that promise, however, is technically extremely demanding and has already required substantial scientific efforts, which so far have not been overwhelmingly successful. In this paper, we provide a laymen's description of what may turn out to be a fundamental hurdle in attaining blockchains that combine scalability, high transaction processing capabilities, and are indeed fully decentralized.

than cryptocurrencies have been proposed. Most, if not all of these applications exploit the fact that one can avoid relying on a trusted entity, be it a governmental body, a bank, or an educational institute, to name few.

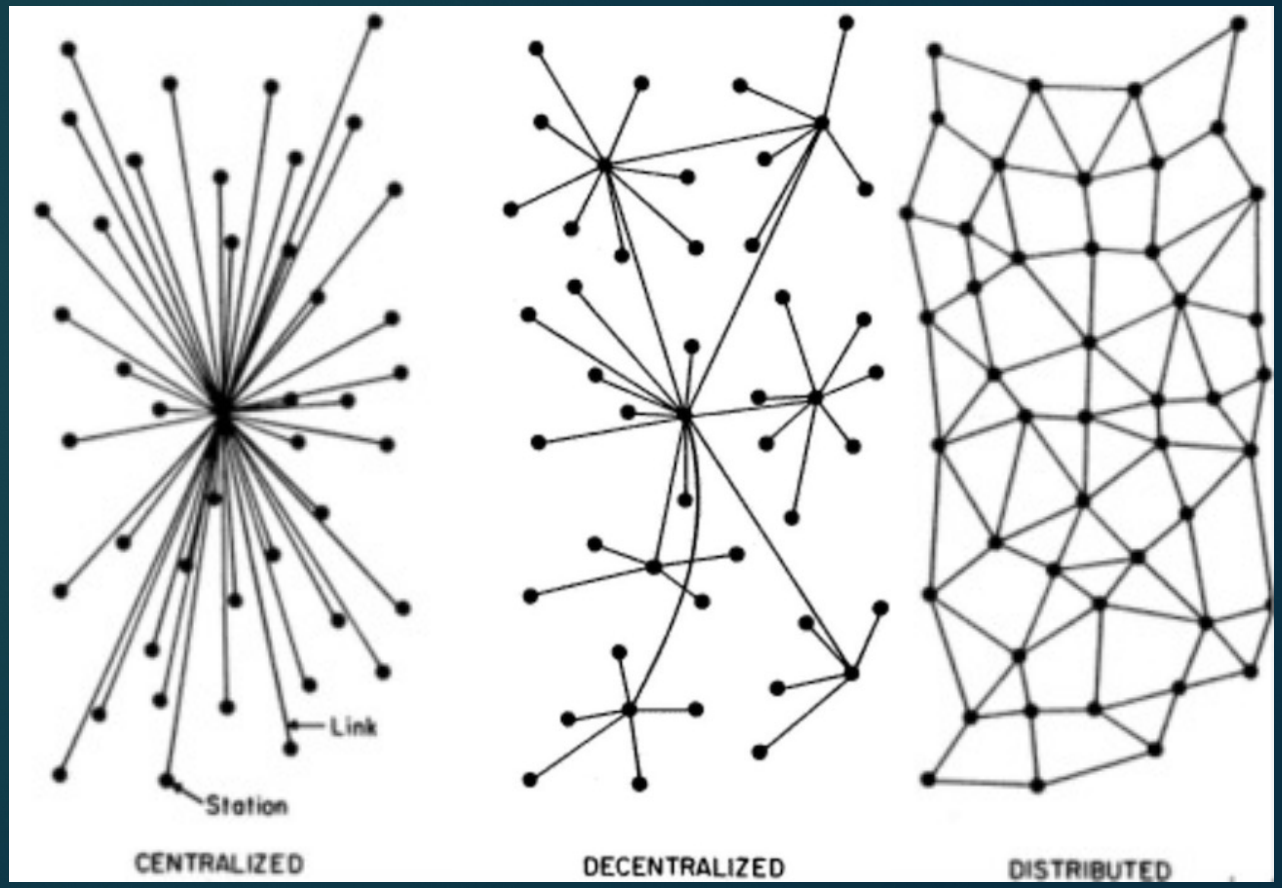
One of the problems with blockchains is that the underlying technology is not that simple to explain, let alone easy to understand. In fact, we argue that a significant level of technical expertise is needed to grasp the many underpinnings of blockchains and that it requires some serious studying even by experts to understand all the details. As a result of these intricacies, there is now a huge gap between understanding appli-

Organizing Information

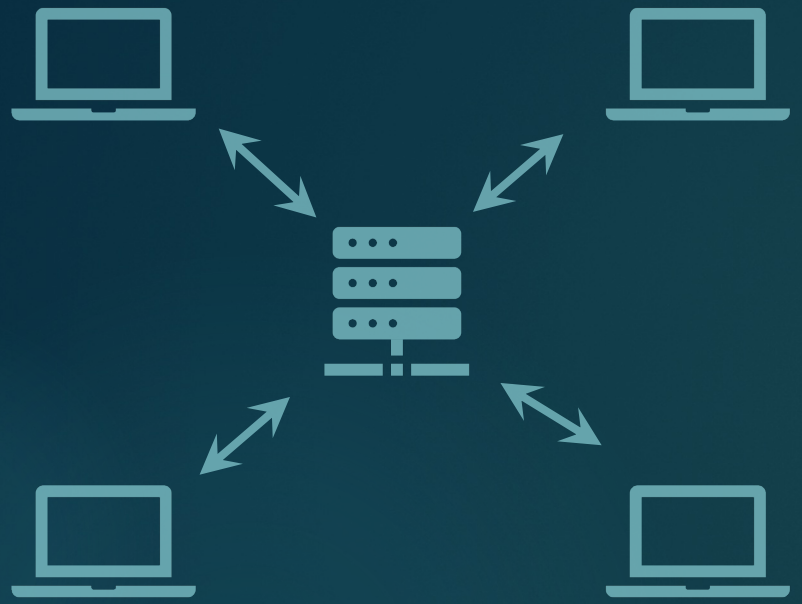
- **Centralized**: single hub
- **Decentralized**: multiple hubs
- **Distributed**: peer-to-peer

Examples:

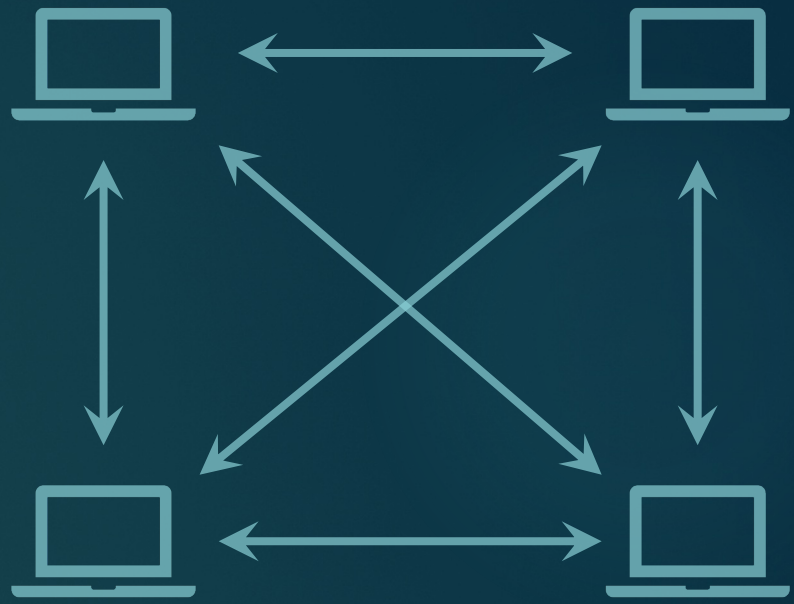
- Domestic banking?
- Cross-border-banking?
- Bitcoin?



Communication Models



Client-server



Peer-to-peer

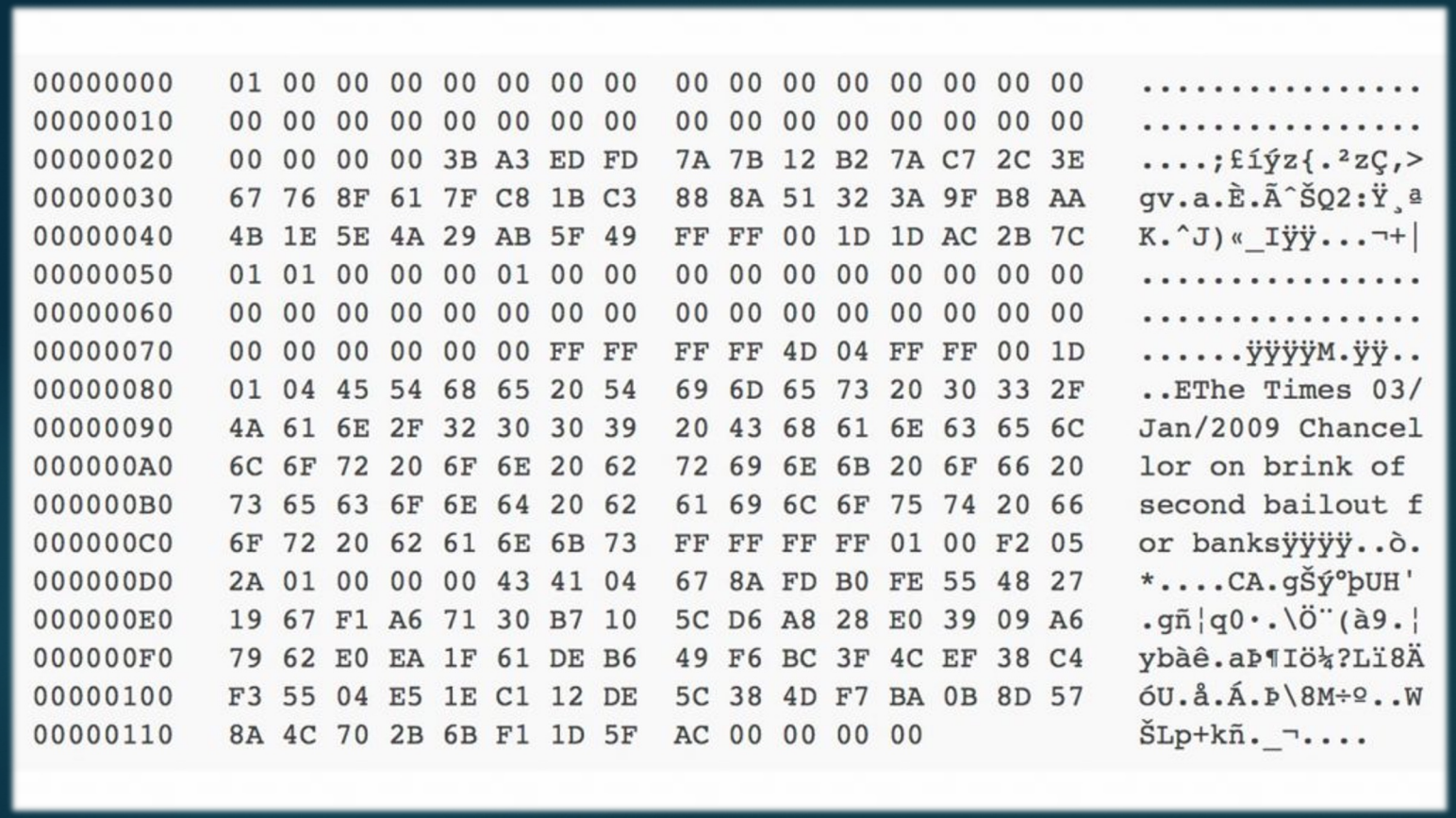
Pros: simple, no need of network discovery, efficient data discovery, synchronization

Cons: single point of failure/trust, unscalable

Pros: Redundacy, every peer is client/server at the same time, resilient, scalable, self-*

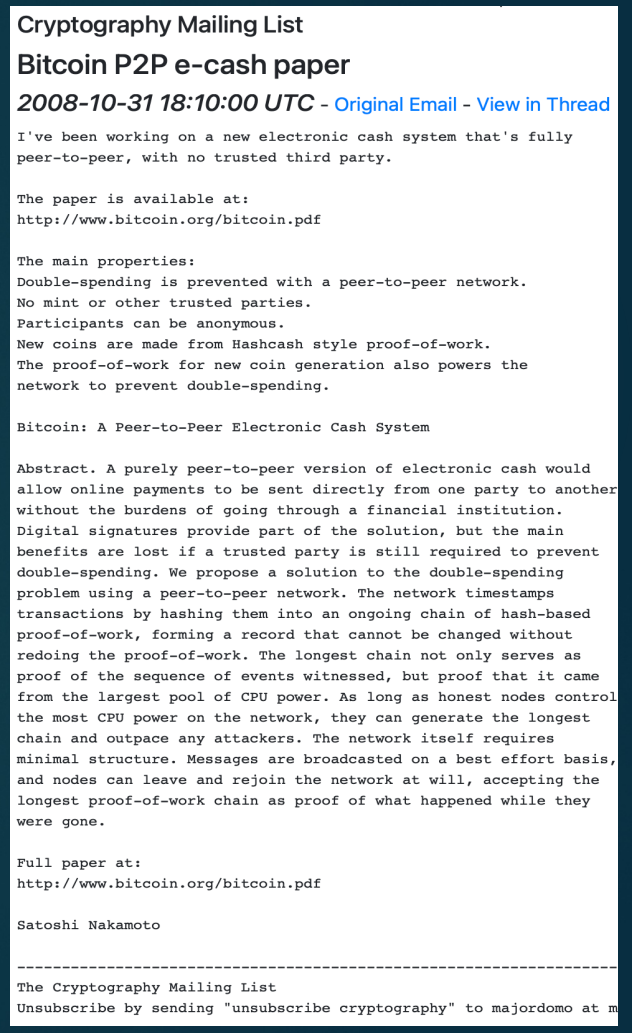
Cons: complex protocols for network discovery, synchronization, hard to debug

Genesis – How It All Started



```
00000000   01 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ................
00000010   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ................
00000020   00 00 00 00 3B A3 ED FD   7A 7B 12 B2 7A C7 2C 3E   ....;£íýz{.²zÇ,>
00000030   67 76 8F 61 7F C8 1B C3   88 8A 51 32 3A 9F B8 AA   gv.a.È.Ã^ŠQ2:Ÿ¸ª
00000040   4B 1E 5E 4A 29 AB 5F 49   FF FF 00 1D 1D AC 2B 7C   K.^J)«_Iÿÿ...¬+|
00000050   01 01 00 00 00 01 00 00   00 00 00 00 00 00 00 00   ................
00000060   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ................
00000070   00 00 00 00 00 00 FF FF   FF FF 4D 04 FF FF 00 1D   ......ÿÿÿÿM.ÿÿ..
00000080   01 04 45 54 68 65 20 54   69 6D 65 73 20 30 33 2F   ..EThe Times 03/
00000090   4A 61 6E 2F 32 30 30 39   20 43 68 61 6E 63 65 6C   Jan/2009 Chancel
000000A0   6C 6F 72 20 6F 6E 20 62   72 69 6E 6B 20 6F 66 20   lor on brink of
000000B0   73 65 63 6F 6E 64 20 62   61 69 6C 6F 75 74 20 66   second bailout f
000000C0   6F 72 20 62 61 6E 6B 73   FF FF FF FF 01 00 F2 05   or banksÿÿÿÿ..ò.
000000D0   2A 01 00 00 00 43 41 04   67 8A FD B0 FE 55 48 27   *....CA.gŠý°þUH'
000000E0   19 67 F1 A6 71 30 B7 10   5C D6 A8 28 E0 39 09 A6   .gñ¦q0·.\Ö¨(à9.¦
000000F0   79 62 E0 EA 1F 61 DE B6   49 F6 BC 3F 4C EF 38 C4   ybàê.aÞ¶Iö¼?Lï8Ä
00000100   F3 55 04 E5 1E C1 12 DE   5C 38 4D F7 BA 0B 8D 57   óU.å.Á.Þ\8M÷º..W
00000110   8A 4C 70 2B 6B F1 1D 5F   AC 00 00 00 00            ŠLp+kñ._¬....
```

"The Times 03/Jan/2009 Chancellor on brink of second bailout for banks"



Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-10-31 18:10:00 UTC - Original Email - View in Thread

I've been working on a new electronic cash system that's fully peer-to-peer, with no trusted third party.

The paper is available at:
http://www.bitcoin.org/bitcoin.pdf

The main properties:
Double-spending is prevented with a peer-to-peer network.
No mint or other trusted parties.
Participants can be anonymous.
New coins are made from Hashcash style proof-of-work.
The proof-of-work for new coin generation also powers the network to prevent double-spending.

Bitcoin: A Peer-to-Peer Electronic Cash System

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without the burdens of going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted 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 honest nodes control the most CPU power on the network, they can generate the longest chain and outpace any attackers. The network itself requires minimal structure. Messages are broadcasted 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.

Full paper at:
http://www.bitcoin.org/bitcoin.pdf

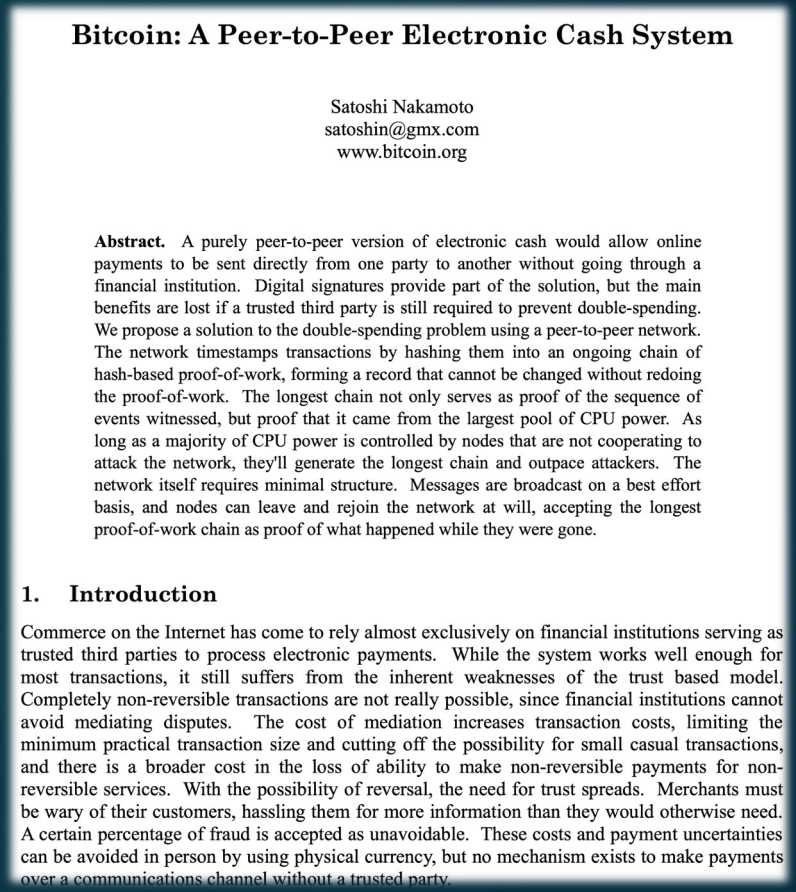
Satoshi Nakamoto

The Cryptography Mailing List
Unsubscribe by sending "unsubscribe cryptography" to majordomo at m

https://satoshi.nakamotoinstitute.org

A P2P Electronic Cash System

- Published 2008, 1st implementation 2009
- Satoshi Nakamoto (anonymous)
- A decentralized P2P technology to maintain a monetary ledger
- A new form of money exempt of central governance or intermediaries
- Innovations:
  - High availability
  - Location independence
  - Inherently censorship resistant
- Impact:
  - S$200 billion in 11 years
  - Inspired other new projects



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.

1. Introduction

Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the minimum practical transaction size and cutting off the possibility for small casual transactions, and there is a broader cost in the loss of ability to make non-reversible payments for non-reversible services. With the possibility of reversal, the need for trust spreads. Merchants must be wary of their customers, hassling them for more information than they would otherwise need. A certain percentage of fraud is accepted as unavoidable. These costs and payment uncertainties can be avoided in person by using physical currency, but no mechanism exists to make payments over a communications channel without a trusted party.

Who is Satoshi Nakamoto?

We do not know!

- **Last message**: April 2011 & disclaimer in March 2014: "I am not Dorian Nakamoto"
- Post times suggest residing: UK or East/West US Coast
- British spelling
- Excellent knowledge of economics, cryptography & P2P networking
- Deep understanding of C++ language
- A person or a team of people?
- Owns at least 1M BTC ~ $20B
- BTC in his wallet are still there
- 10 BTC to Hal Finney in 2009

Why is he anonymous?

- Enormous wealth
- Life risks by invividuals & goverments
- Illegal, FBI prosecutions, e.g. e-Gold 2007
- Retain Bitcoin openness & decentralization

Hal Finney

Nick Szabo

Dorian Nakamoto

Craig Wright

?

Distributed Identity: Identification

Digital signature:

- **Public key** (public identifier): 
- **Private/secret key** (confidential): 

- Computed using the private key
- Verified by the public key
- *Computationally infeasible to derive the signature of a message corresponding to the public key without knowing the secret key*
- *A message authenticated with a signature verifiable with the public key is assumed to originate from an entity with knowledge of the corresponding secret key*

Distributed Identity: Wallets

Wallet: a public/private keypair, or multiple new keypairs for every incoming/outgoing transaction

A form of decentralized bank account:

- Receive funds
- Execute transfers out
- Attached to a balance & transaction history

Public key: 

- Bank account number
- Shared to allow transfers

Private key: 

- Known only to wallet owner
- Authorizes transfers out via a digital signature authenticating the transfer

Network Nodes

Archive nodes: Store a complete copy of the blockchain

Seed nodes: Serve as entry points for new (archive) nodes

Number of Bitcoin nodes in 2020: 10,000, top-3 [GER, US, FR] ~ 40%

Nodes act as protocol enforcers, enforce *consensus rules*:

- Monitor traffic
- Verify validity of transactions & blockchain
- Application of protocol rules
- Cooperation to detect/ignore invalid transactions & incorrect accounting operations
- Open-source software is audited

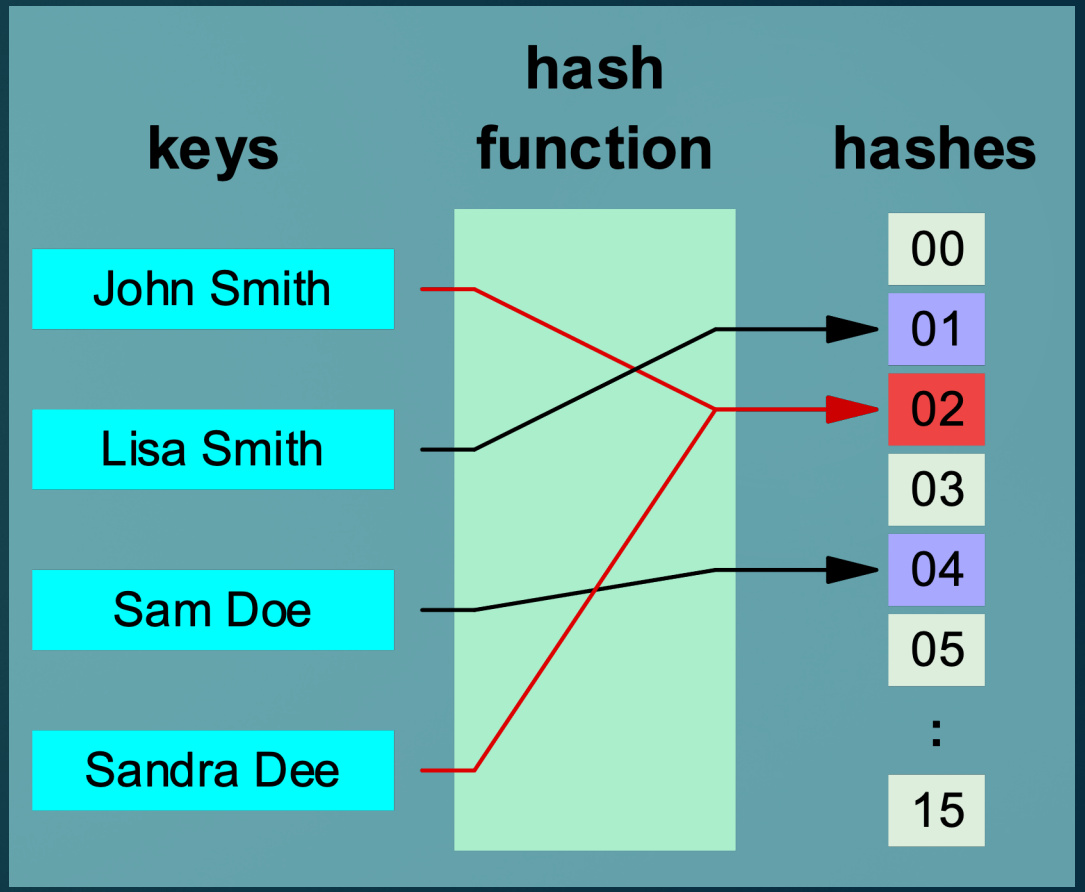
Node lifecycle:

- **Bootstrap phase**: connects to a connected seed node with a public known IP address
- **Peer discovery**: recursive request of IP addresses from connected nodes
- **State recovery**: full or partial cooperative download of the ledger
- **Network servicing**: relays copies of the ledger subject of consensus rules

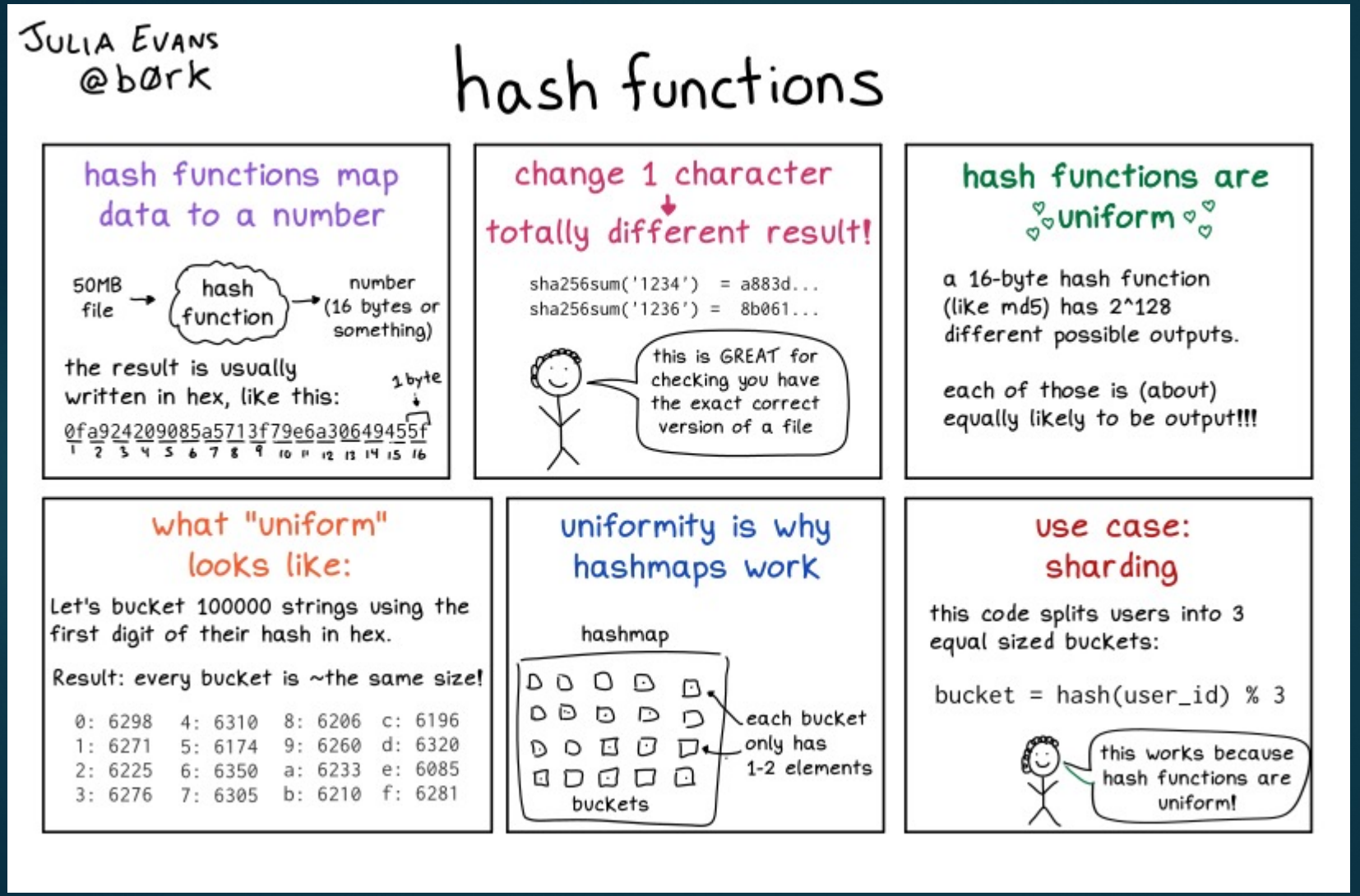
Hash Function

A function that maps data of arbitrary size to data of fixed size

- **Determinisitc**: a certain input always hashes to the same output
- **One-way process**: the output cannot give back the input
- **Colission resistant**: hard for two different inputs to hash the same output



Hash Function



Blockchain Data Structure

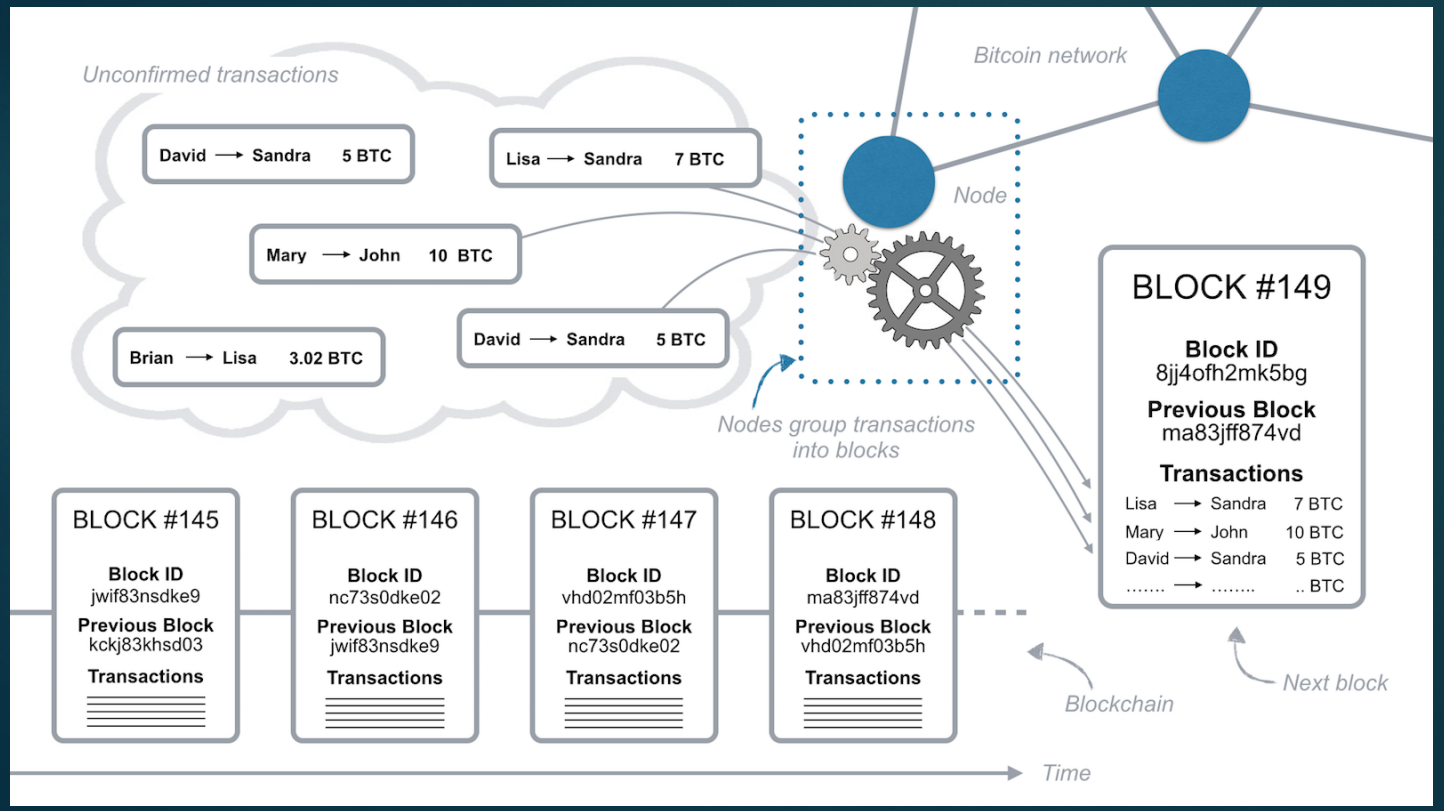
What is a blockchain data structure:

- *A cryptographic chain of data blocks containing an exchaustive list of validated transactions from genesis block to latest block*
- Block i contains a batch of transactions that chronologically:
  - succeed transactions of previous parent blocks
  - preceds transactions of future children blocks
  - exception is the genesis block
- Formally:
  - hash(message)
  - $block_0$: ($transactions_0$)
  - $block_i$: (hash($block_{i-1}$),$transactions_i$), for i>0 & $transactions_i$ the list of transactions in $block_i$
  - $(block_i)_{i=0}^{N}$, where N is the index of the latest block

How Blockchain Works

Tampering past blocks is not possible because:

- Changing the content of a block generates a new hash($block_i$)
- All subsequent blocks hash($block_{i+1}$),… are influenced, breaking the chain validity



A node adds transations to blocks as they wish: transaction fees/size, order of arrival, etc.

Mining & Miners

Mining: the process of adding blocks to the blockchain

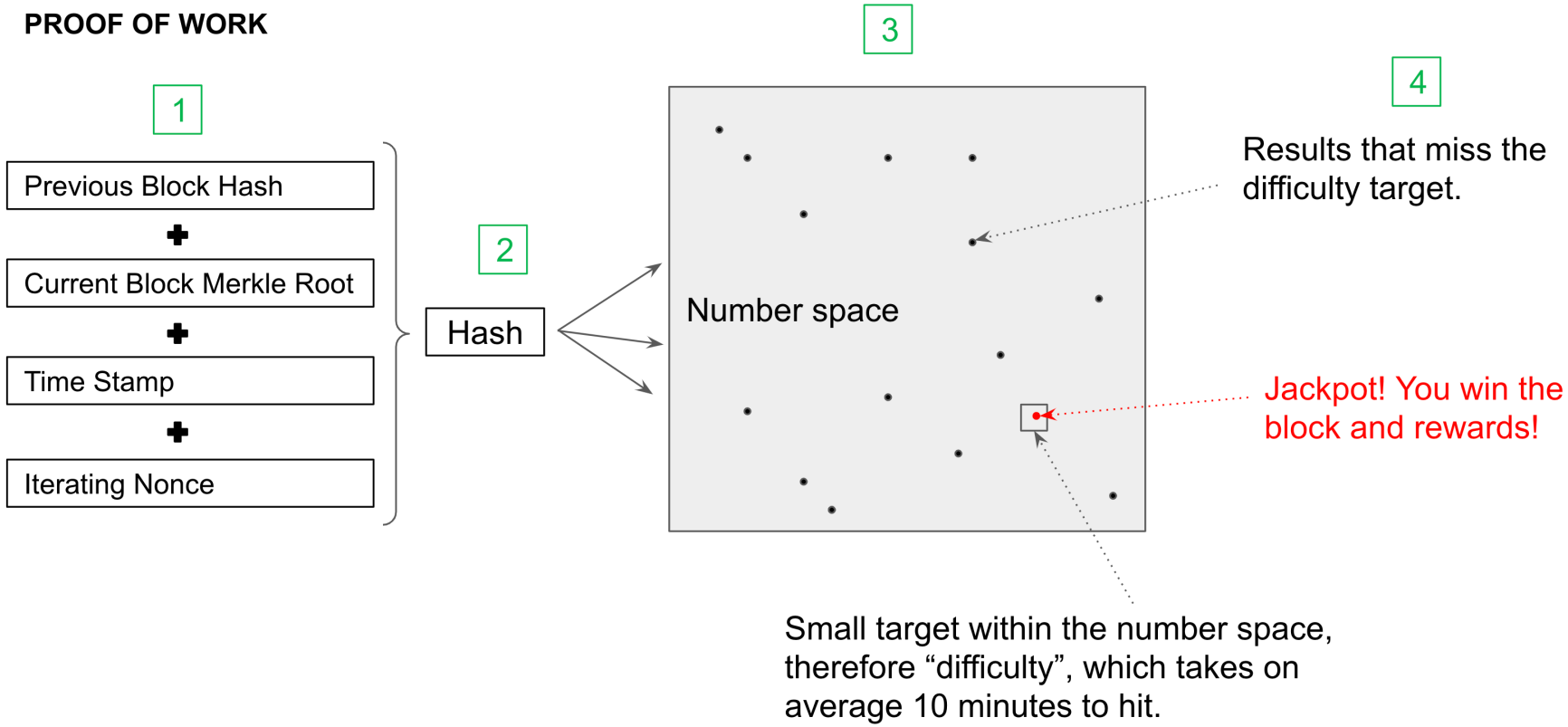
The rate of adding blocks is constant

- Bitcoin: ~ every 10 mins
- Ethereum: ~ 14 sec

How? Hashing must meet the network difficulty target:

- generate (& find) nonce: *number used once*
  - *Bitcoin*: 32-bit (4 byte) field, block hash will contain a #of leading `0's
  - Infeasible to predict which combination of input bits yields the right hash
  - *Brute force*: requires times & resources
  - *Proof of work*: the presentation of the block with the correct nonce value

Mining & Miners



Proof of Work

- Very difficult to produce the proof but easy to validate
- Requires significant computational resources & energy
- GPUs can execute more instructions per clock
  - CPUs are good for complex manipulations in small datasets
  - GPUs are good for simple manipulations of large datasets
- Miners' incentives: winning transaction & mining fees paying for the resources invested to participate

Limits of Proof of Work

- Consumes high energy, influenced by the total mining, unsustainable
- Slow transactions throughput, production of blocks is slow
- Power concetration is discouraged but stil possible: large miners
- No guarantee for the finality of transactions
- Competitve mining requires high-performance specialized hardware and access to low electricity prices

Scalability limits

Proof of Stake

- Miners vote for the next valid block
- Against Sybil attacks: voting influence proportional to the amount of staked coins
- Consumes neglible amount of energy
- Higher transaction throughput
- Scaling techniques such as sharding

Limits of Proof of Stake

- **Nothing at stake**: Staking is costless, miners can vote for blocks in forks without penalty
- **Resilience in extreme events**: long-term network splits, harder to merge forks
- **Accessibility**: cryptocurrency is needed
- Rich gets richer effects
- Lower rewards than proof of work

Blockchain Nodes

Full nodes

- Ensure the blockchain integrity
- Can be mining nodes, but not all full nodes are mining nodes
- E.g. desktop wallet
- Have a blockchain copy with every block and transaction
- They only receive rewards if they are mining nodes

Light nodes

- Maintain the headers of the blockchain
- Verify transactions via *Simplified Payment Verification* (SPV): connect to full nodes and transmit transations without downloading the blockchain
- Dowload only block headers & not the whole blockchain
- Operate usually as wallets, e.g. Coinbase: https://www.coinbase.com

Blockchain Nodes

REACHABLE BITCOIN NODES
Updated: Sun Jan 29 22:00:31 2023 GMT

13228 NODES **CHARTS**

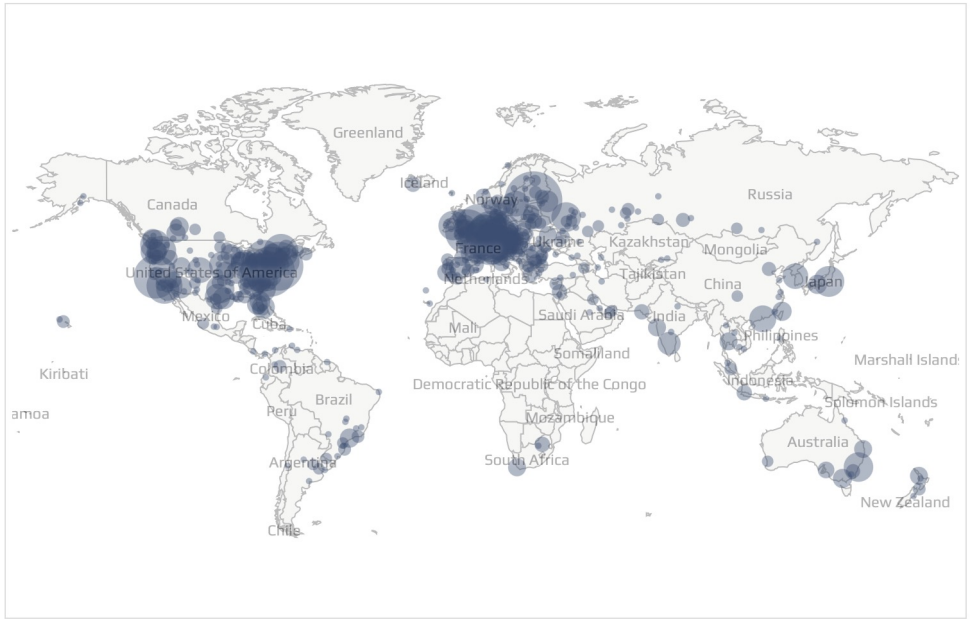
IPv4: -2.6% / IPv6: -1.5% / .onion: -14.9%

Top 10 countries with their respective number of reachable nodes are as follows.

RANK	COUNTRY	NODES
1	n/a	6855 (51.82%)
2	United States	1692 (12.79%)
3	Germany	1311 (9.91%)
4	France	425 (3.21%)
5	Netherlands	376 (2.84%)
6	Canada	285 (2.15%)
7	Finland	231 (1.75%)
8	United Kingdom	211 (1.60%)
9	Russian Federation	164 (1.24%)
10	Switzerland	130 (0.98%)

All (89) »

NOTE / The data above includes reachable Bitcoin nodes only. View combined estimation of reachable and unreachable Bitcoin nodes »



Map shows concentration of reachable Bitcoin nodes found in countries around the world. **LIVE MAP**

https://bitnodes.io

Permissioning

Operationalizing blockchain as a centralized or fully decentralized system

Who can participate in the network is determined by the protocol itself:

- **Permissionless ledger [public]**: any user can join, download blockchain, and maintain it without preliminary human authorization
  - Bitcoin & Ethereum
- **Permissioned ledger [private]**: who can read, update or access the blockchain comes with restrictions
  - R3 Corda, Hyperledger Fabric
- **Hybrid ledger**: flexible adaptive alternative between public & private ledger
  - Dragonchain
- **Sidechain**: a blockchain network tied on the main chain to test new features & other use cases, speed up transactions finality & lower transaction costs
  - Rootstock, Ardor

Permissioning

Choice is usually a trade-off between:

- Performance & available computational resources
- Security, privacy, trust & resilience
- Other organizational requirements & constraints

Permissionless vs. permissoned blockchain

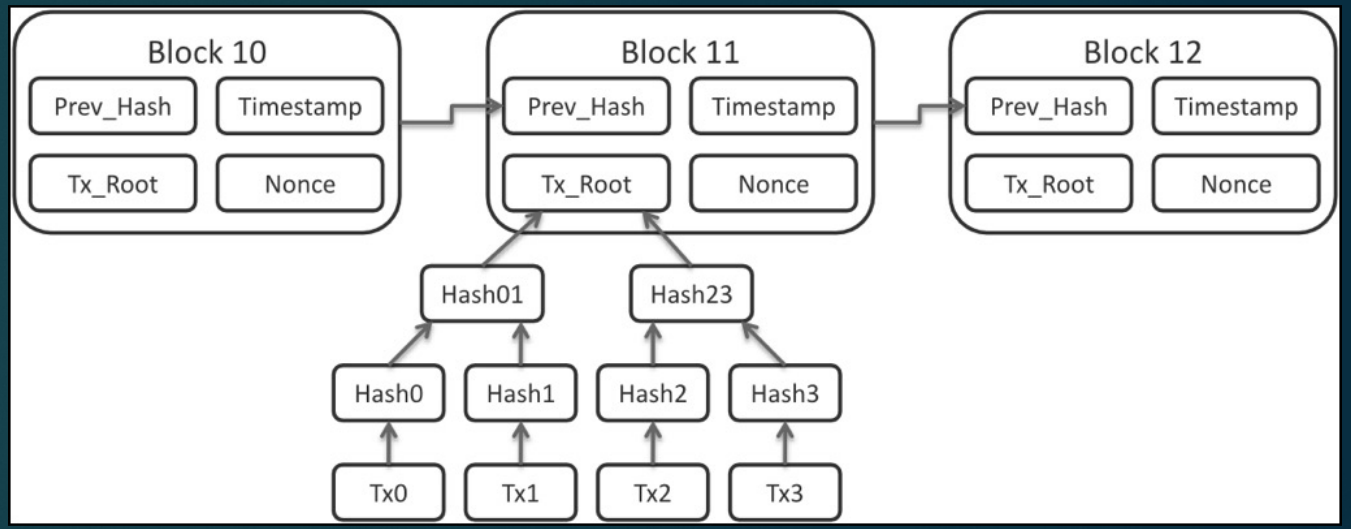
	Permissionless	Permissioned
OVERVIEW	Open network available for anyone to interact and participate in consensus validation. Fully decentralized across unknown parties.	Closed network. Designated parties interact and participate in consensus validation. Partially decentralized (i.e., distributed across known parties).
ALSO KNOWN AS	Public, trustless.	Private, permissioned sandbox.
KEY ATTRIBUTES	■ Full transparency of transactions, based on open source protocols ■ Development via open source ■ Mostly anonymous, with some exceptions ■ Privacy depends on technological limitations or innovations ■ No central authority ■ Often involves digital asset or token for incentives	■ Controlled transparency, based on organizations' goals ■ Development via private entities ■ Not anonymous ■ Privacy depends on governance decisions ■ No single authority, but a private group authorizes decisions ■ May or may not involve digital assets or tokens
BENEFITS	■ **Broader decentralization**, extending access across more network participants ■ **Highly transparent**, which is beneficial for speed and reconciliation across unknown parties ■ **Censorship resistant**, due to accessibility and participation across locations and nationalities ■ **Security resilience**, since attackers cannot target a single repository, and it is costly and difficult to corrupt 51% of the network	■ **Incremental decentralization**, but participation from multiple businesses helps mitigate risks of highly centralized models ■ **Stronger information privacy** because transaction information is only available based on permissions ■ **Highly customizable** to specific use cases through diverse configurations, modular components and hybrid integrations ■ **Faster and more scalable**, since fewer nodes manage transaction verification and consensus
PITFALLS	■ **Less energy efficient** because network-wide transaction verification is resource-intensive ■ **Slower and difficult to scale**, as high volume can strain network-wide transaction verifications ■ **Less user privacy** and information control	■ **Limited decentralization** because a network with fewer participants increases risk of corruption or collusion ■ **Risk of override**, since owners and operators can control or change the rules of consensus, immutability, or mining ■ **Less transparent to outside oversight**, since participants are limited and operators determine privacy requirements
MARKET TRACTION	■ Peer-to-peer ■ Business-to-consumer ■ Government-to-citizens	■ Business-to-business ■ Business-to-consumer ■ Governments-to-organizations

Block Structure

Block header

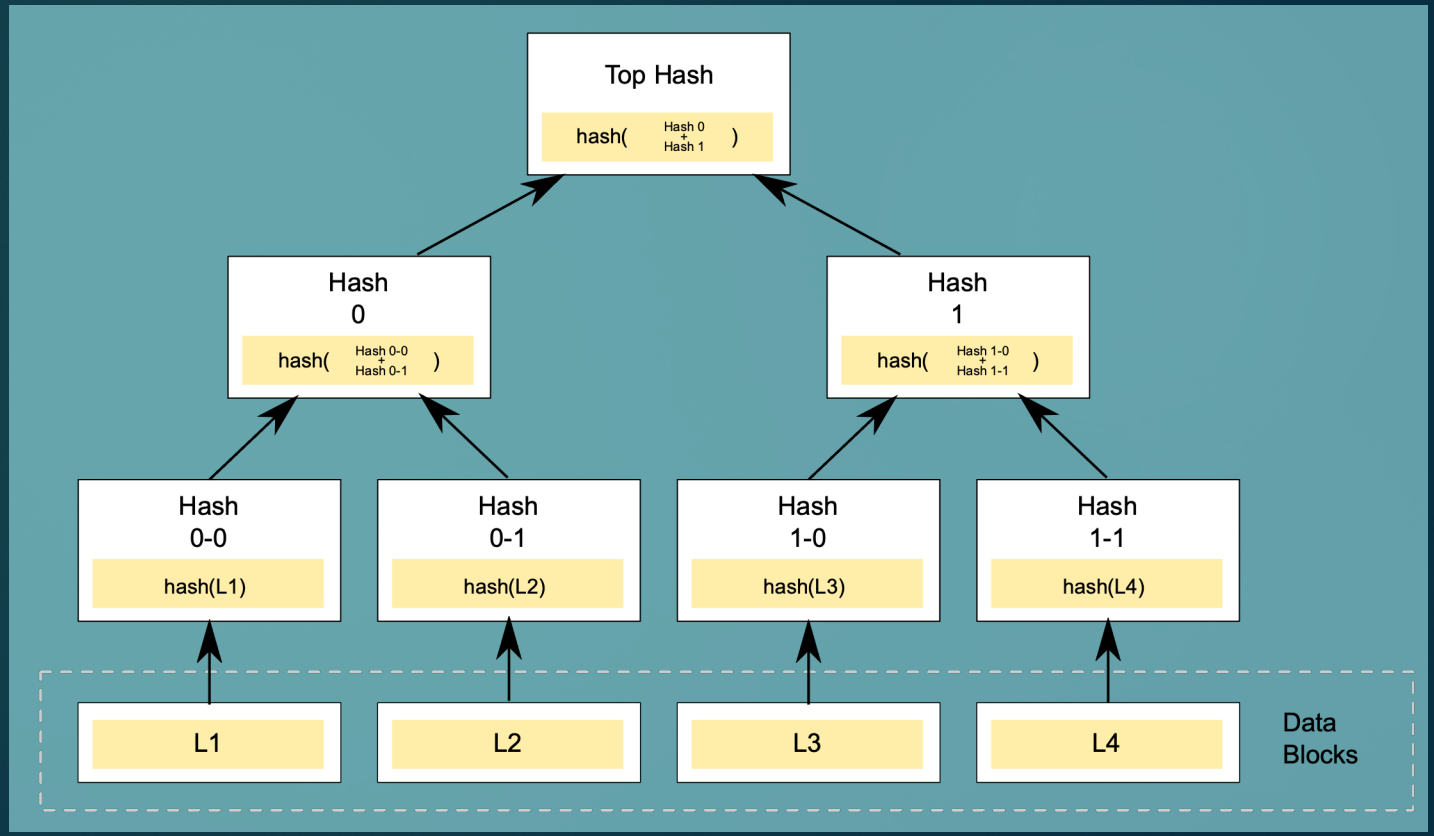
- Hash of previous block
- Timestamp
- Merkle tree
- Nonce, network difficulty target

List of transactions



Merkle Tree

- **Leaves**: transactions
- **Non-leaf nodes**: hashes
- **Merkle root**: stored in block header
- **Block**: contains transactions as a Merkle tree
- **SPV use**: match of Merkle roots calculated with $\log_2 n$ local & remote hashes to verify transactions of a light node, where n is the # of transactions



Incentives

Problem: *Why would users engage in producing blocks subject of obtaining a valid proof-of-work, which requires expensive resources?*

e.g. electricity, investment in infrastructure, maintenance depreciation costs

Block rewards:

- Given to miners who verify the transactions blocks
- A portion of newly minted digital tokens
- Block reward schedule (monetary policy) is part of the protocol
- Bitcoin rewards halves every 4 years (~210K blocks)
  - Will stop on maximum circulation of 21M

Transaction fees:

- Sum of all transaction fees of the transactions included in the block
- Payed by respective senders
- Optional: but incentivize inclusion of transactions in the block (higher priority)

Payed to miners

Incentives & Difficulty

Difficulty: endogenously defined & time-varying so that block production rate is stable

Duration:

- large enough such that two valid blocks are not submitted at the same time
- short enough for high processing throughput of transactions

Equilibria: expected mining gains balance running costs

- Price of electricity
- Computational efficiency of hardware
- Exchange rate

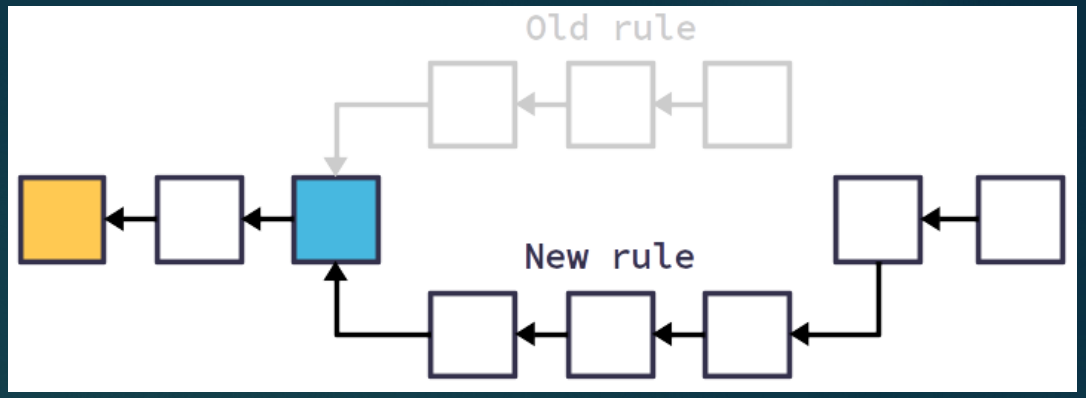
Higher exchange rate:

- Higher total value in the blockchain
- More gains for miners
- More miners joining
- Higher difficulty

Lower exchange rate:

- Lower total value in the blockchain
- Lower gains for miners/bankruptcy
- Less miners' activity or leaving
- Lower difficulty

Hard Forks



Nodes split in two groups that maintain two different blockchain versions with different rules, **not backward compatible**

Why forks happen:

- Solving the computational puzzle almost at the same time
- Security attack: revert/modify the ledger accounting

How to resolve:

longest chain rule, wait for longest cumulative difficulty (usually longest branch)

Examples:

- **Ethereum vs Ethereum Classic 2016**: hacking by code vulnerability
- **Nxt rejected hard fork proposal 2014**: 50M NXT theft from major cryptocurrency exchange
- **Bitcoin Cash hard fork 2017**: disagreement how to increase transactions per second for demand

Smart Contracts

A computer program or transaction protocol to automatically execute, control or document events & actions according to the terms of a contract or agreement.

Benefits:

- No need for trusted intermidiators
- Reduction of arbitration costs
- Reduction of fraud losses
- Lower risks for malicious & accidental exceptions
- Recuced friction between parties
- Reduction of moral hazards
- No legal agreement, but smart legal contracts emerge
- Turing-complete Solidity language for smart contracts on, e.g. Ethereum

How they work:

- Smart constract code is included in transactions
- Execution: once the transaction is verified & added to the blockchain
- Anti-Tampering/immutability: via Byzantine fault-tolerant consensus algorithms
- Clients interact with smart contract via transactions
- Transactions with a smart contract can invoke other ones
- Deterministic processes to ensure Byzantine fault-tolerance

Pizza Day

May 22, 2010, Laszlo Hanyecz

10,000 BTC ~ $100M

Altcoins

Bitcoin successors

- New mining algorithm for higher decentralization
- Hashrate from 10 to 2 mins
- New inflation schedule, while maintaining capped supply

litecoin

- Meme inspired cryptocurrencies
- No actual innovation
- $400M market capitalization

Initial Coin Offerings

A type of crowd-funding using cryptocurrencies by speculators/investors in exchange of legal tenders or other cryptocurrencies

- Capital for startup companies
- Based on white papers & business plans
- Functional units of currency if/when funding goal is met
- Loose regulations: no need to the scrutiny of other intermediaries (venture capitalists, banks, stock exchanges) offering future profits or joint ownership
- Often vehicle for scams & fraud
- Market capitalization in Jan 2018: $831B before collapsing by more than 85%

Blockchain & Cryptoeconomics (Labs)

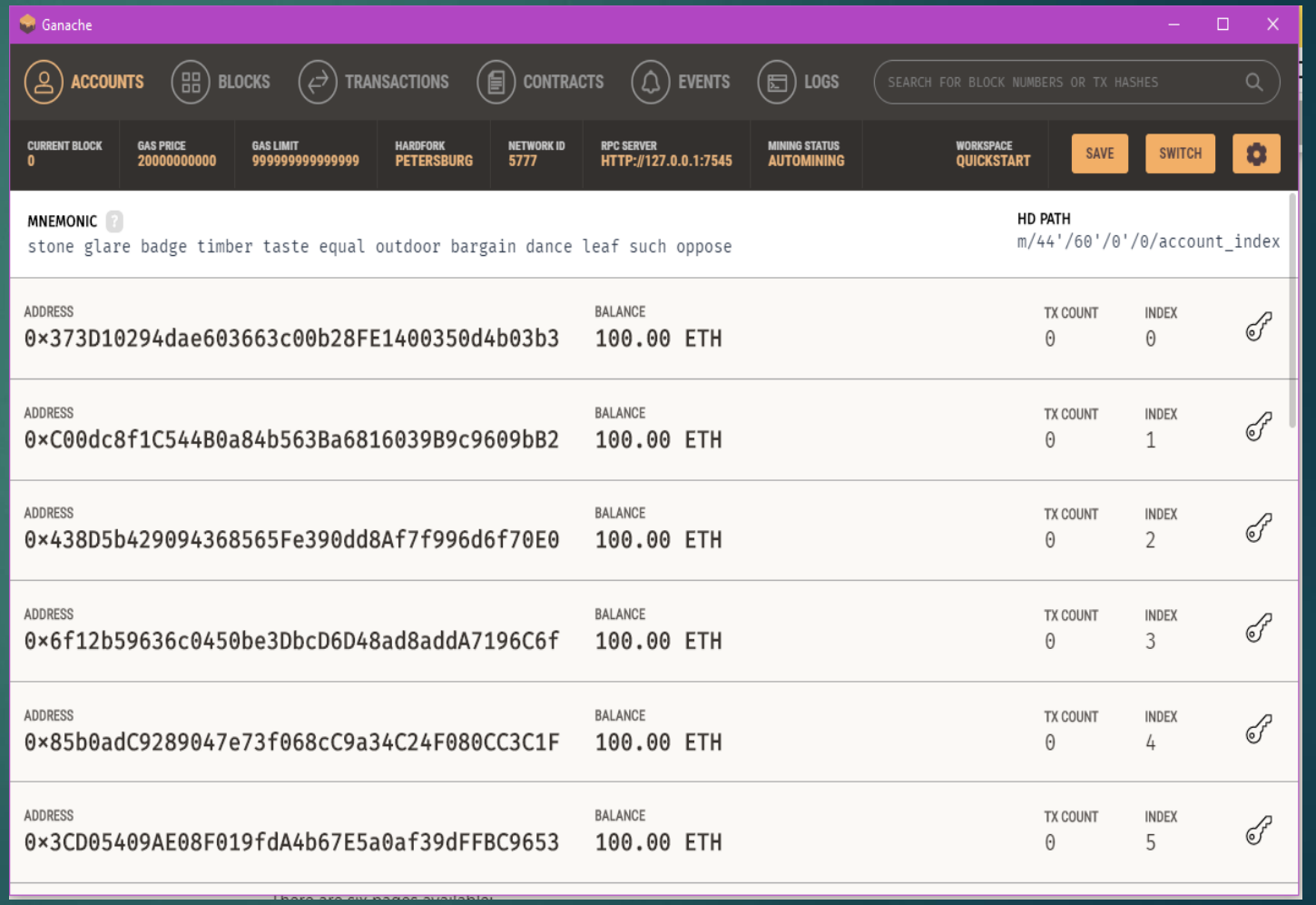
RABIYA KHALID

Ganache

- Developing dApps can be quite the hassle
- To develop a dApp, you need to simulate the blockchain
- Ganache is a development tool to run a local Ethereum blockchain
- It is helpful in all parts of the development process
- Provides deterministic and safe environment to develop, deploy and test smart contracts
- Geth and parity are also popular Ethereum clients
- We need to setup Miners and network nodes
- Ganache takes care of all these things

Ganache

- Accounts tab shows the accounts generated and their balances
- Blocks tab shows each mined block along with gas used and transactions
- Transactions tab lists all transactions run in blockchain
- Contracts tab lists contracts in truffle projects
- Events tab lists all triggered events within the workspace
- Logs tab contains logs for the server



MetaMask

- Ethereum blockchain wallet
- Allows to buy, send and receive Ethers
- Installed as a browser extension.
- Follow the instructions to create your wallet
- Save your secret recover phrase (mnemonics)
- You can recover you account using this phrase
- On loosing the phrase, it will be impossible to recover your account

Cryptographic Hash Methods

- Hash function converts data to a fixed length string
- The output is unique for each input
- Slightest change in input results in different output
- The output is nor reversible
- How hashing works in blockchain

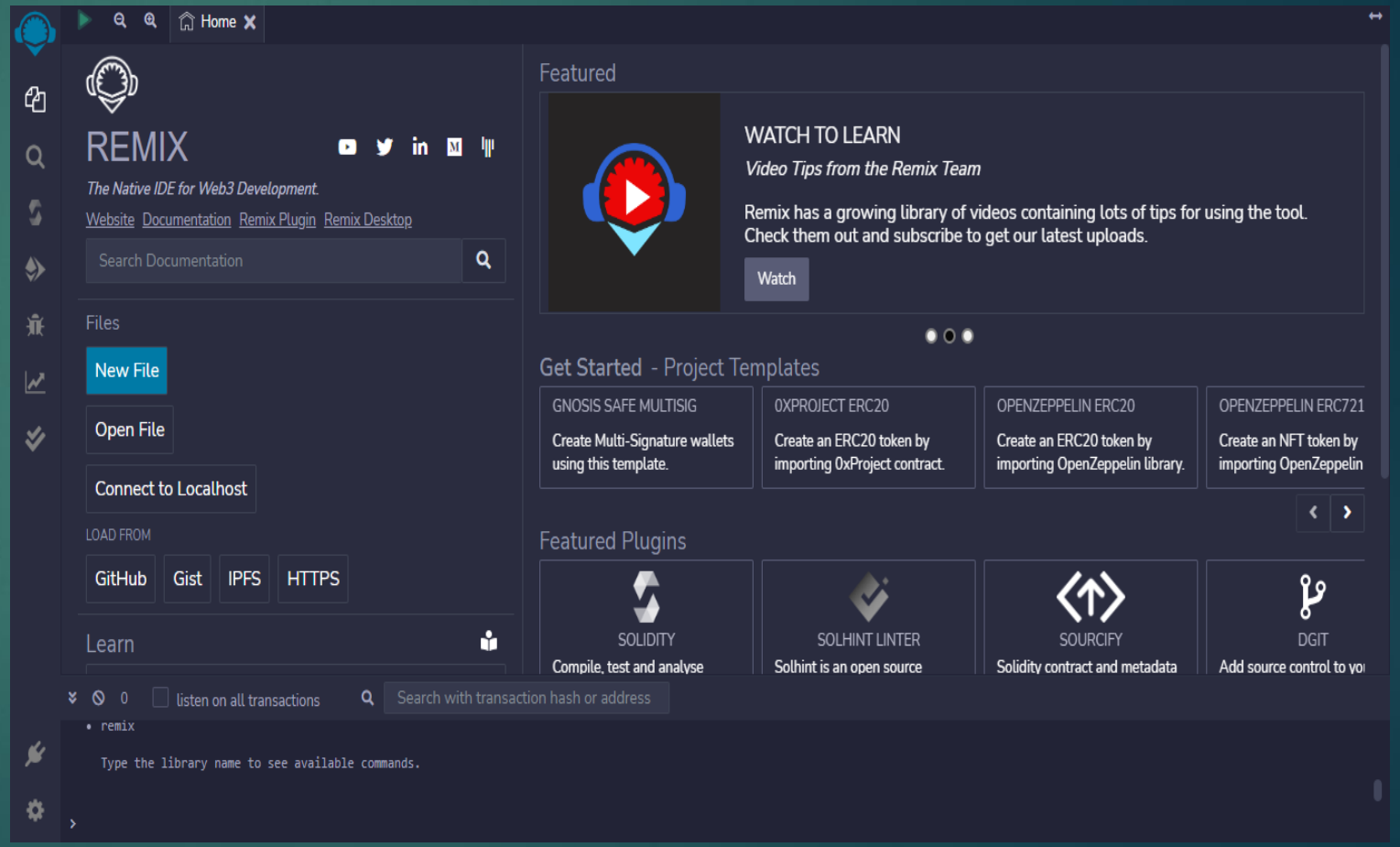
Smart contracts

- Simple programs stored on blockchain
- Execute when predetermined conditions are met
- Automate the execution of an agreement between two parties without third party
- Can be used
  - Releasing funds
  - Registering vehicles
  - Issuing a ticket, etc.

```
pragma solidity ^0.5.0;

contract PvsV{

    int number = 6;

    function testPure(int a) public pure returns(int){
        return(a+6);
    }

     function testview(int a) public view returns(int){
        return(a+number);
    }

}
```

Remix IDE

- A browser based IDE
- Allows to write, compile and debug solidity smart contracts
- Provides user friendly interface
- Has a built-in solidity compiler
- Accessible from any browser
- Has a large community support
- Integration with Ethereum and local network



Solidity

- Solidity was developed for Ethereum blockchain
- Became a standard for writing smart contracts
- Adopted by Ethereum community: developers and users
- Influenced by several programming languages
  - JavaScript
  - C++
  - Python
- Provides rich set of features
- Has a large community support
- Other programming languages such as Vyper, Binance smart chain and rust are also available
- Solidity remains widely used and supported language

Solidity

Variable naming rules

- Variable name should not start with numeric values
- Variable name should not be same as reserved words
- Variable names are case sensitive

Solidity

Variable Types

- State variables
  - Values are stored in smart contract`s memory
  - Accessible throughout the smart contract
  - Variables have 3 states: public, internal, private
  - Can be accessed from outside of the smart contract based on the state
- Local variables
  - Declared inside a function
  - Destroyed when scope of the function ends
  - Cannot be accessed outside of the contract
- Global variables
  - Special type of variables
  - Store blockchain properties such as block timestamp, block number and gas limit

Solidity

Functions

- A chunk of code, with a name
- Can be used over and over again
- Allows programmers to avoid repetition in code
- Divides a program into smaller chunks
- Access modifiers of functions
  - Public
  - External
  - Internal
  - Private

Solidity

Function types

- Payable
  - Used to transfer Ethers from an account to a smart contract
- View
  - Used to read data from blockchain
  - Reads local variables, inputs of the function and returns output
- Pure
  - Does not alter state of the contract
  - No data is saved or read from the blockchain

Solidity

Modifier

- Used to add a pre-requisite of a function
- A function executes only if the condition defined in the modifier is true
- An exception is thrown if condition is false
- Modifier is defined before using it with a function

Solidity

Arrays

- A data structure that stores a collection of elements of same type
- Each element is identified by an index or key
- Solidity support
  - Static array
  - Dynamic array
- Delete an element
  - Pop() function: Deletes an element from the last index
  - Delete keyword: Deletes an element based on the input index

Solidity

Decision making

- Solidity supports decision making statements
- It supports
  - If statement
  - If… else statement
  - If… else if statement
- **Loops**
- Solidity supports
  - While loop
  - Do… while loop
  - For loop
  - Loop control

Solidity

Struct

- Used to create a datatype with multiple properties
- Groups different types of related data and stores it in a single record
- **Mapping**
- It is like a dictionary or hash table
- Consists of a key value and a single or group of values stored against it
- Commonly used to keep track of objects of a structure

Solidity

Inheritance

- A smart contract can be inherited from another smart contract
- Solidity supports multiple inheritance
  - A parent smart contract can have multiple child smart contracts
  - A child smart contract can have multiple parent smart contracts
- Two independent smart contracts can also interact with each other

Truffle framework

- A suite of tools to write and deploy smart contracts
- Provides boxes which are packages of code templates
- Boxes can be unboxed in a project to get a quick start
- Boxes contain organized files and folders necessary for a project
- Developers prefer truffle over Remix for complex projects
- User can directly interact with smart contract
- Interactions with smart contracts are possible by writing direct script
- Some dependencies are required to be installed

Truffle framework

Dependencies

- Install Node package dependency (npm) by installing node.js
- Check version of the installed node version by node –v
- Install truffle framework by "*install truffle*" command on terminal
- Install Ganache
- Install a text editor

Truffle framework

Truffle project

- Create a project directory by "*mkdir project_1*" command
- Move to that directory
- Unbox the truffle petshop box package by "*truffle unbox pet-shop*"
- Open text editor
- Create a new file in the contracts folder and rename it
- Write a setter and getter functions in the contract
- Go to migration directory and create migration file for your smart contract
- Migration files are executed in the same sequence as they appear in the folder

Truffle framework

Truffle project

- Open terminal
- Type "*truffle console*"
- Deploy smart contract by typing "*migrate*"
- Get the address of a smart contract by typing "*contractname.deployed().then((ins)=>{con=ins})*"
- Pass value to the setter function by typing "*con.set(12)*"
- Get the value using the getter function "*con.get()*"

Questions?

Blockchain & Cryptoeconomics

DR. EVANGELOS POURNARAS

5. Decrypting Blockchain Design

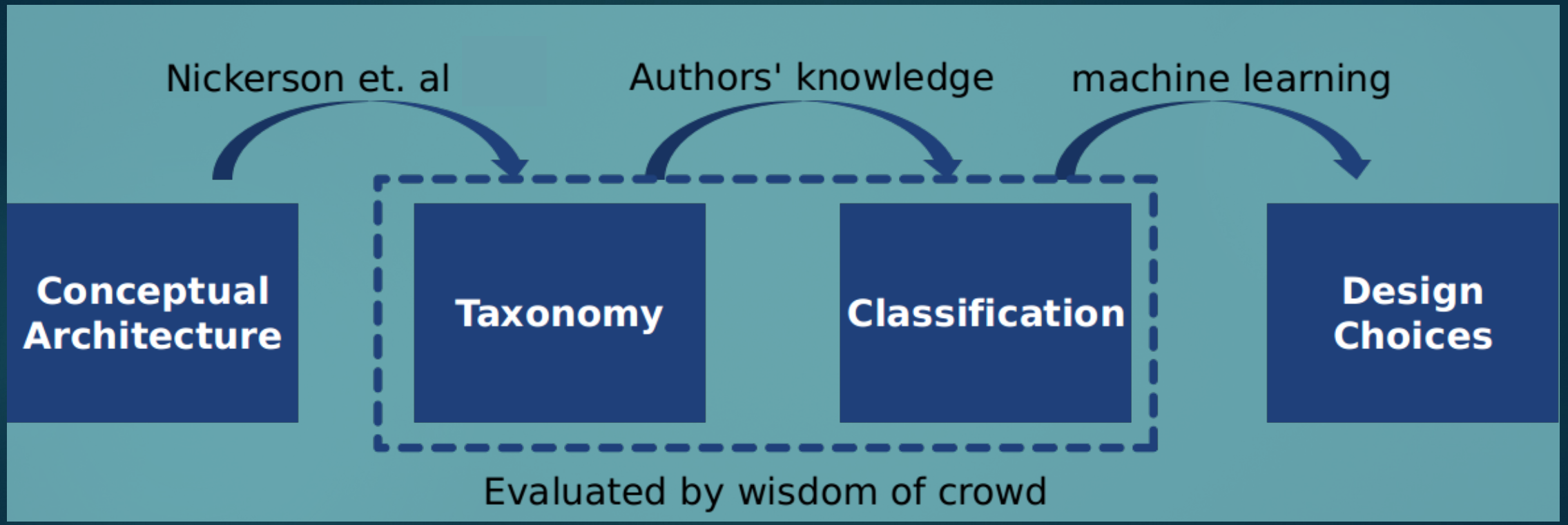
Designing Blockchain Systems

Blockchain & cryptoeconomic systems are very complex but >1000 systems emerged raising $600B in investments in 2016

How such systems are designed? What design decisions are involved? How these decisions influence the functionality, performance & supported applications?

- *A conceptual architecture of distributed ledgers*
- *A taxonomy of distributed ledgers*
- *A classification of distributed ledgers using real-world data & wisdom of the crowd*
- *A design guideline for distributed ledgers*

Analysis Approach



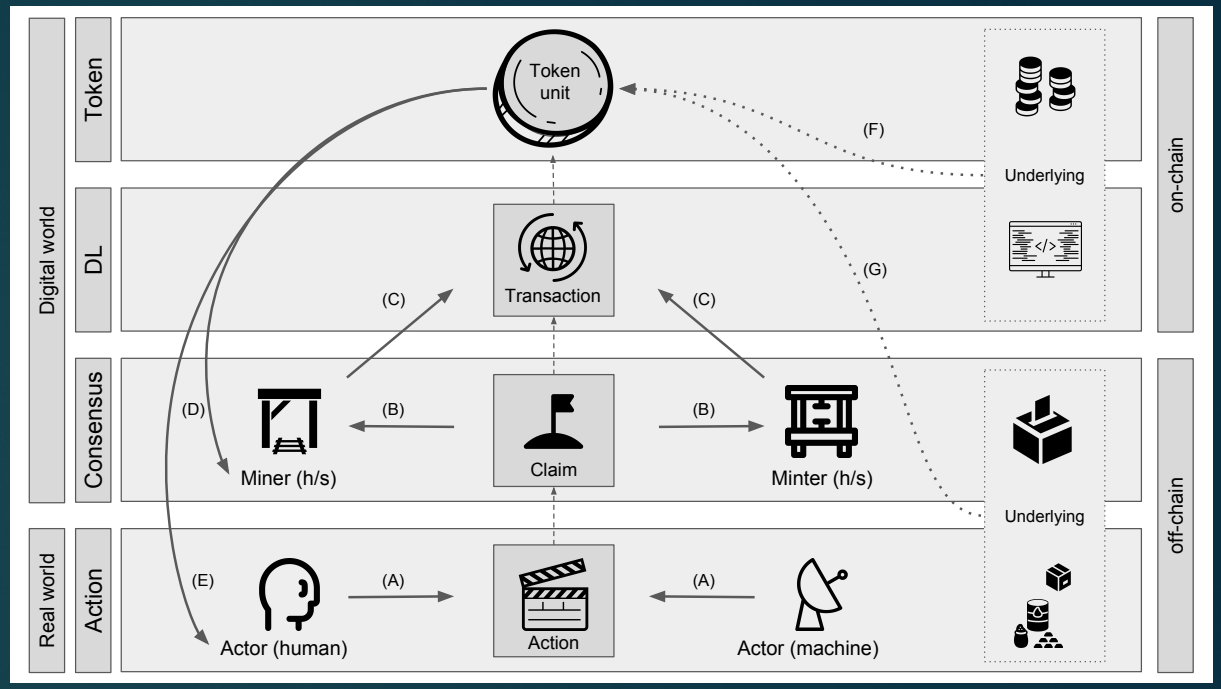
Conceptual Architecture

Action: performed by a human/machine in real-world resulting in a claim in the digital world

Consensus: the agreement of miners/minters on claims that result in writing on the distributed ledger

Distributed ledger: claims combined into blocks & written to the ledger (transactions)

Token: rewards given for an action or participating in the consensus, backed up by an on-chain or off-chain source of value



On-chain: existence on the distributed ledger - transactions & smart contracts

Off-chain: existence on the consensus network or real-world

Taxonomy

Limitation of blockchain taxonomies: no cryptoeconomic design, qualitative, no systematic validation

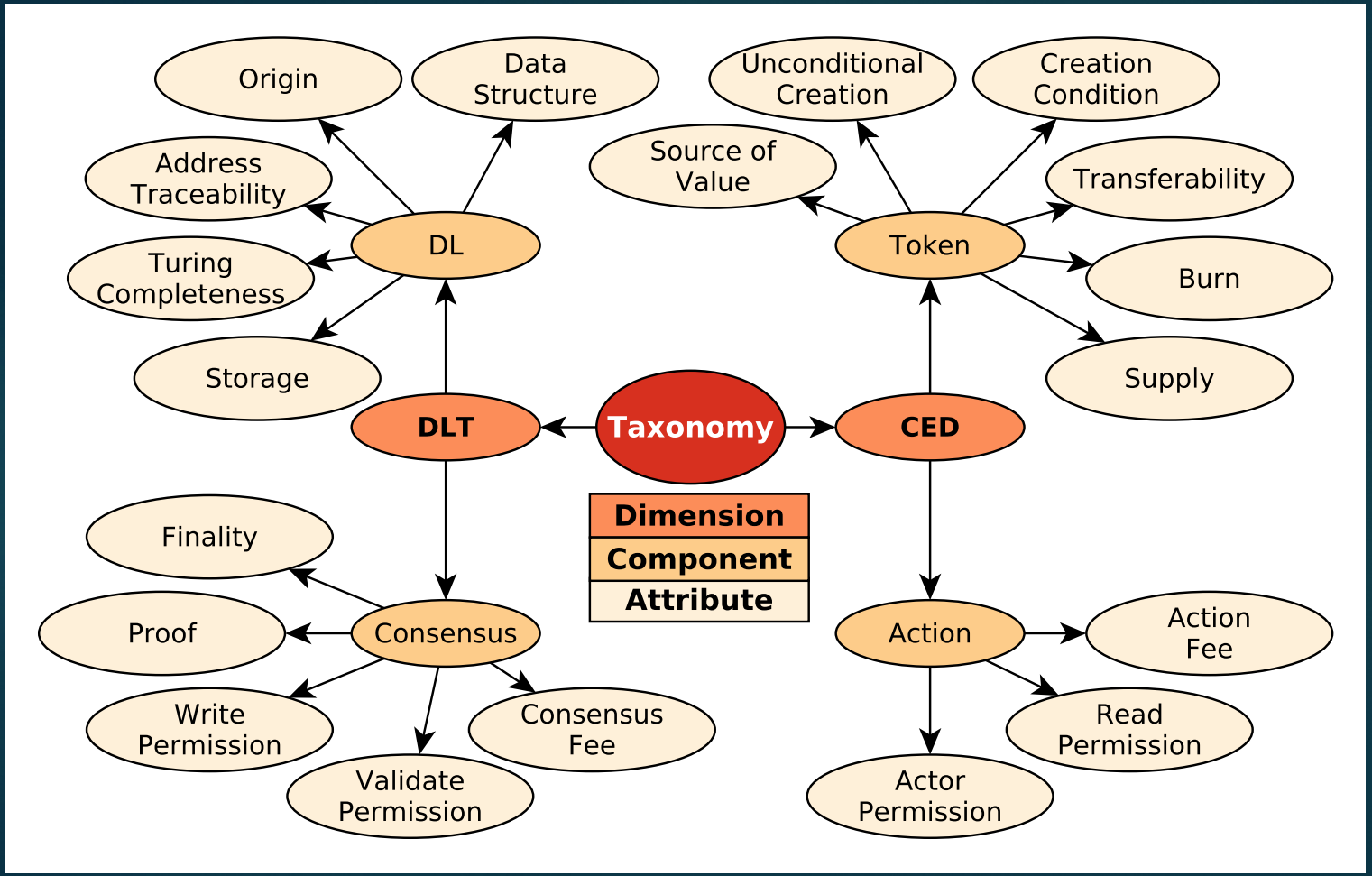
What is a useful taxonomy: one that used a minimal number of attributes to differentiate the objects of interest

- comprehensible
- robust

How to assess usefulness:

- Qualitative criteria based on theory
- Quantitative criteria based on machine learning
- Wisdom of the crowd & experts' feedback

Taxonomy



Method: Nickerson et al.

Goal: capture key design choices

Relies on the four components:

- Distributed ledger
- Consensus
- Action
- Cryptoeconomic design

Taxonomy: Distributed Ledger

Data structure: blockchain, directed acyclic graph, other

- Bitcoin, Ethereum, Litecoin, IOTA, Ripple

Origin: native, external, hybrid

- Bitcoin, NXT, Aragon, Augur, Counterparty, Factom

Address traceability: obfuscatable, linkable

- Zcash, Monero, Bitcoin, Ripple

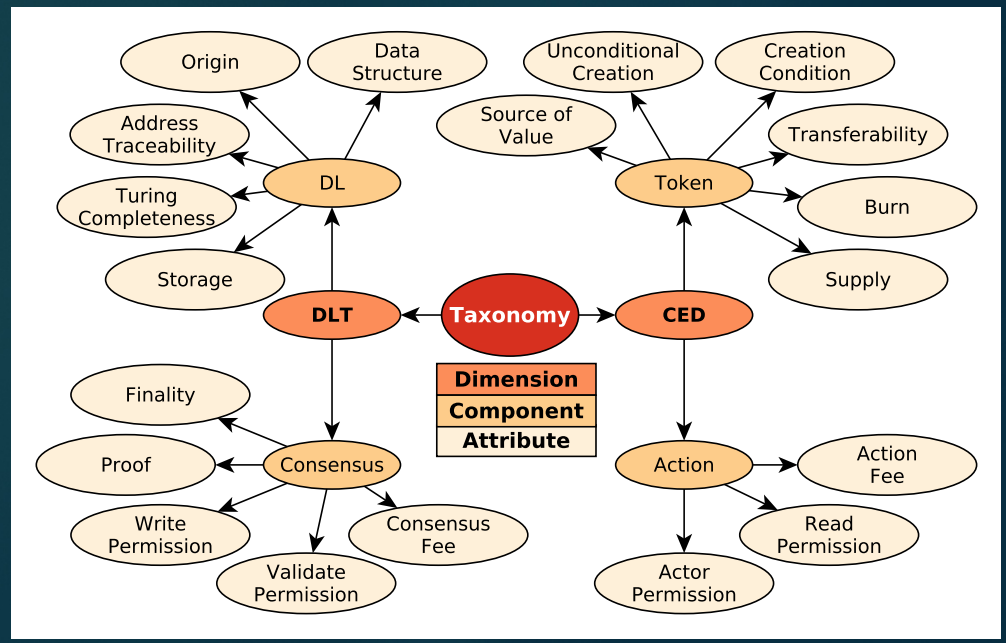
Turing completeness: yes, no

- Ethereum, Bitcoin,

Storage: yes, no

- Bitcoin, IOTA

Distributed ledger: *A distributed data structure, containing entries that serve as digital records of actions.*



Taxonomy: Consensus

Finality: deterministic, probabilistic

- Byzantine Fault tolerance, Ripple, Bitcoin, IOTA

Proof: proof-of-work, proof-of-stake, hybrid, other

- Bitcoin, Ardor

Write permission: restricted, public

- Practical Byzantine Fault Tolerance, Ripple, Bitcoin

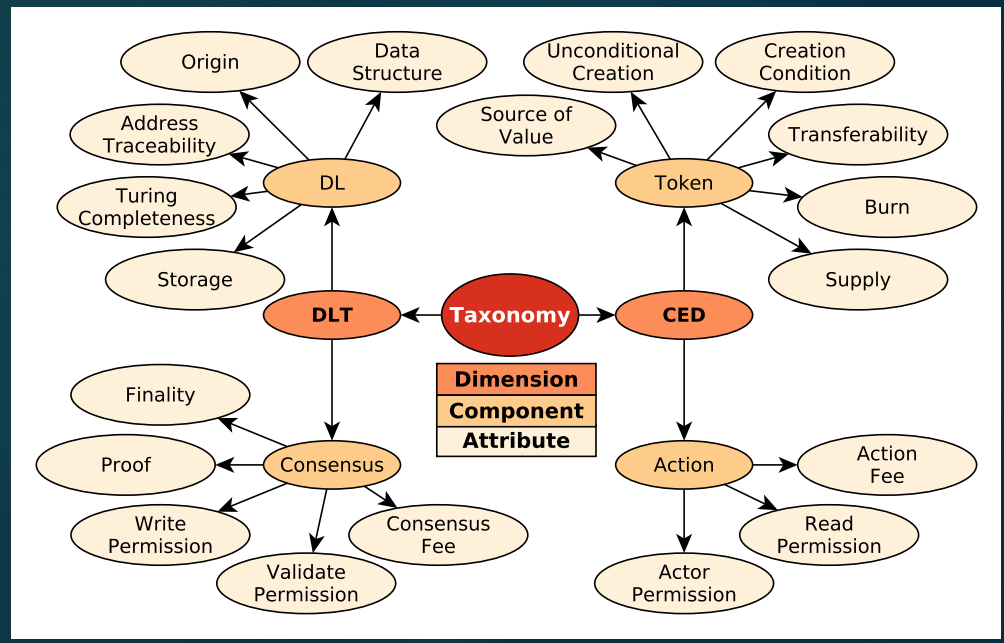
Validation permission: restricted, public

- IOTA, Bitcoin

Fee: yes, no

- Bitcoin, IOTA

Consensus: *The mechanism through which entries are written to the distributed ledger, while adhering to a set of rules that all participants enforce when an entry containing transactions is validated.*



Taxonomy: Action

Actor permission: restricted, public

- Ripple, Bitcoin

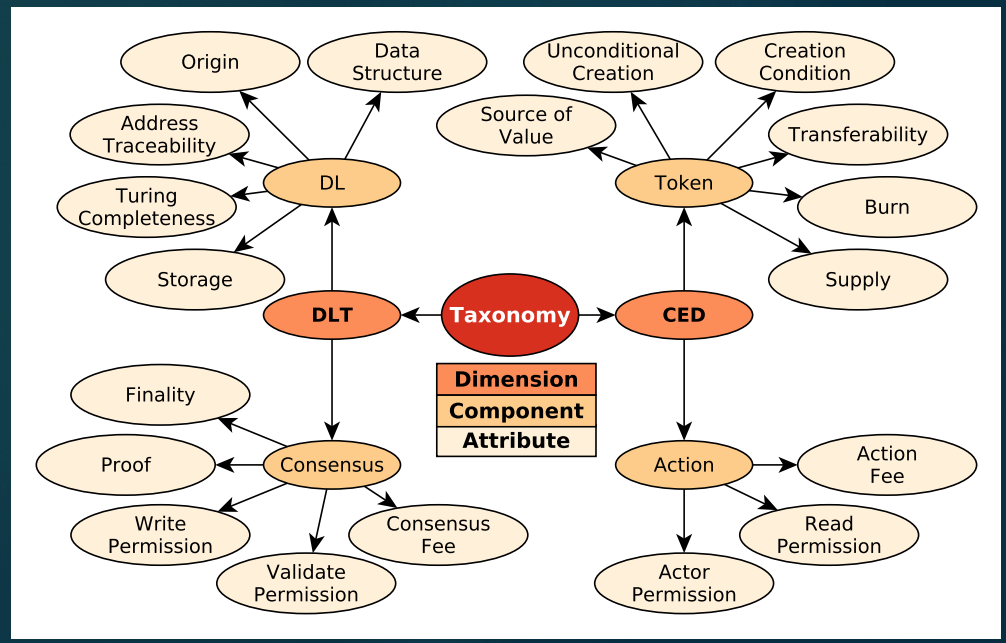
Read permission: restricted, public

- Zcash, Bitcoin

Action fee: yes, no

- Augur (service providers), Ripple (destroyed), Bitcoin

Action: *One or more real-life activities that can be digitally represented in a DLT system as a transaction.*



Taxonomy: Cryptoeconomic Design

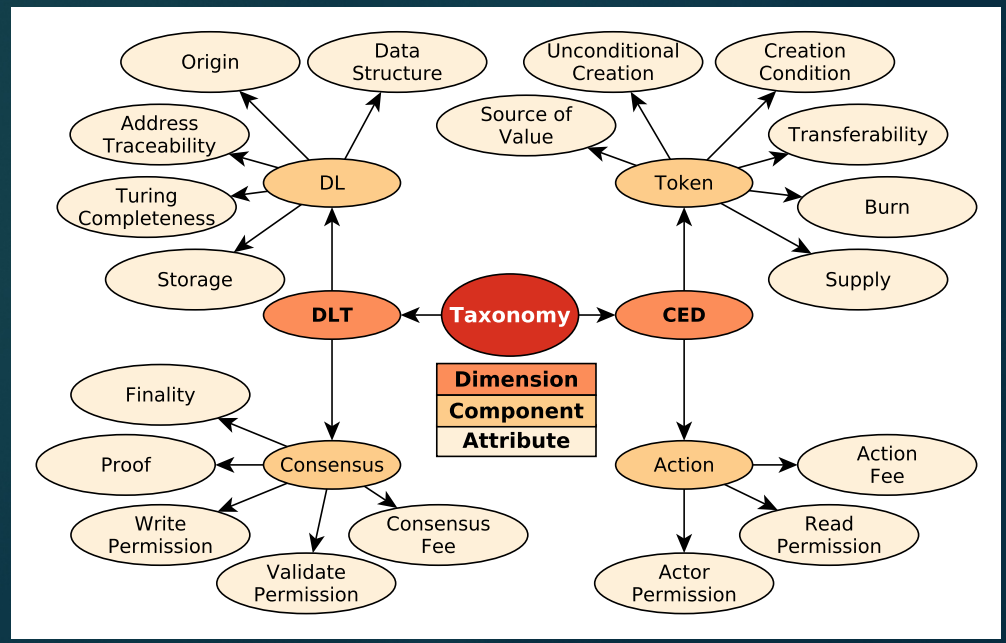
Supply: capped, uncapped

- Bitcoin (21M), Dogecoin
  - Capped supply:
    - Appreciation of perceived token value
    - Deflation in the nominated token prices
    - Appreciation of exchange rates
    - More stable system

Burn: yes, no

- Ripple, Bitcoin
- Reducing token supply by removing token units
  - For constant demand, token units appreciate & result in better exchange rates with other tokens

Token: *A unit of value issued within a DLT system & which can be used as a medium of exchange or unit of account.*



Taxonomy: Cryptoeconomic Design

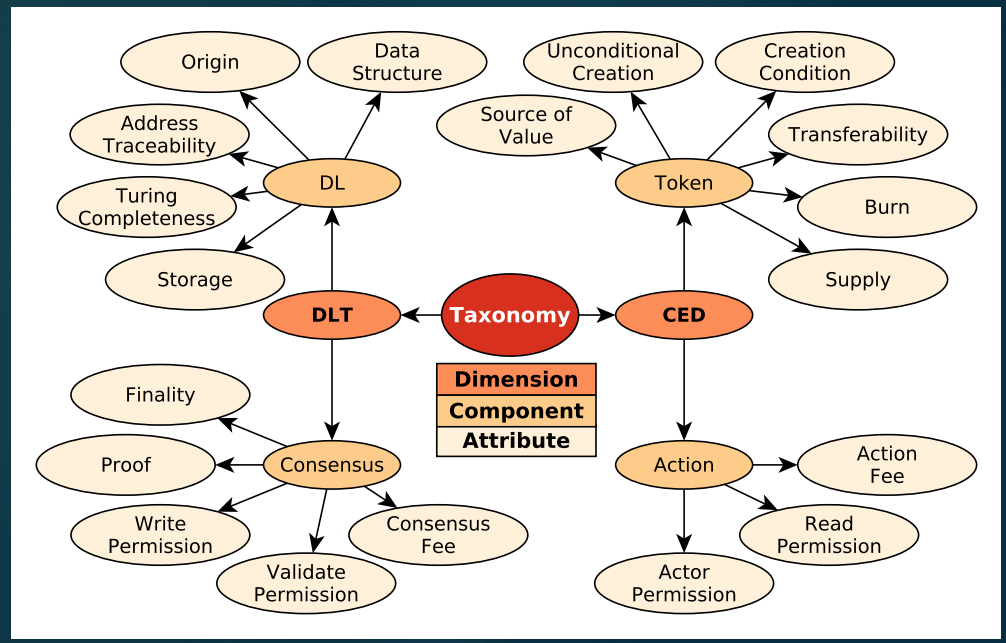
Transferability: transferable, non-transferable

- Bitcoin, Akasha (reputation: Mana, Essense)

Creation condition: consensus, action, both, none

- Bitcoin (block rewards), Steemit (new blog articles), DASH (consensus & obsfucation), Ripple
- The incentivization of consensus or an action is linked or not to the creation of tokens

Token: *A unit of value issued within a DLT system and which can be used as a medium of exchange or unit of account.*



Taxonomy: Cryptoeconomic Design

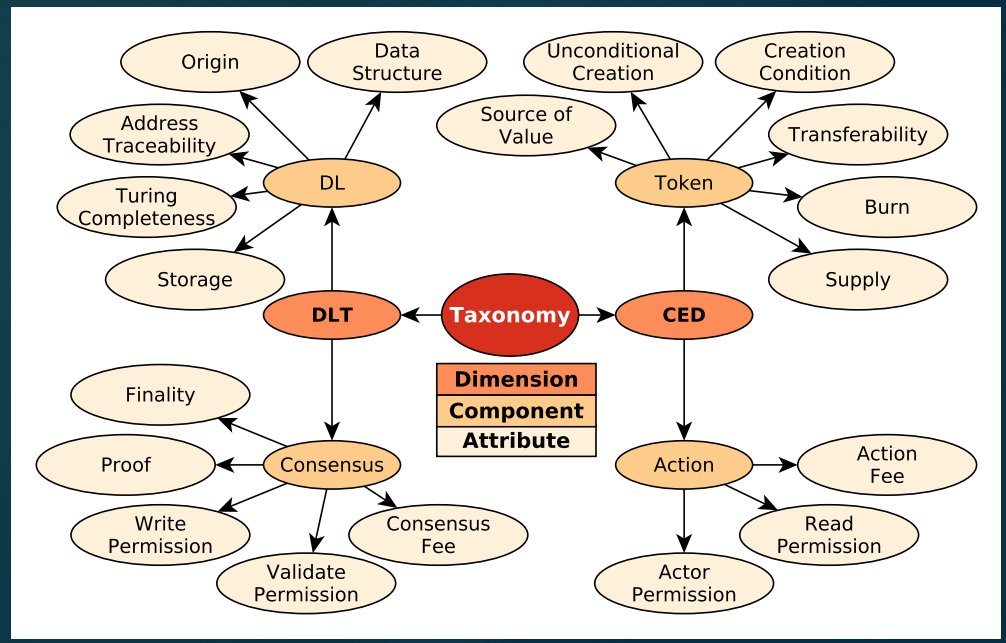
Unconditional creation: all, partial, none

- Ripple (genesis), Augur, Bitcoin
- New token units created that do not incentivize the consensus mechanism or action

Source of value: token, distributed ledger, consensus, action, none

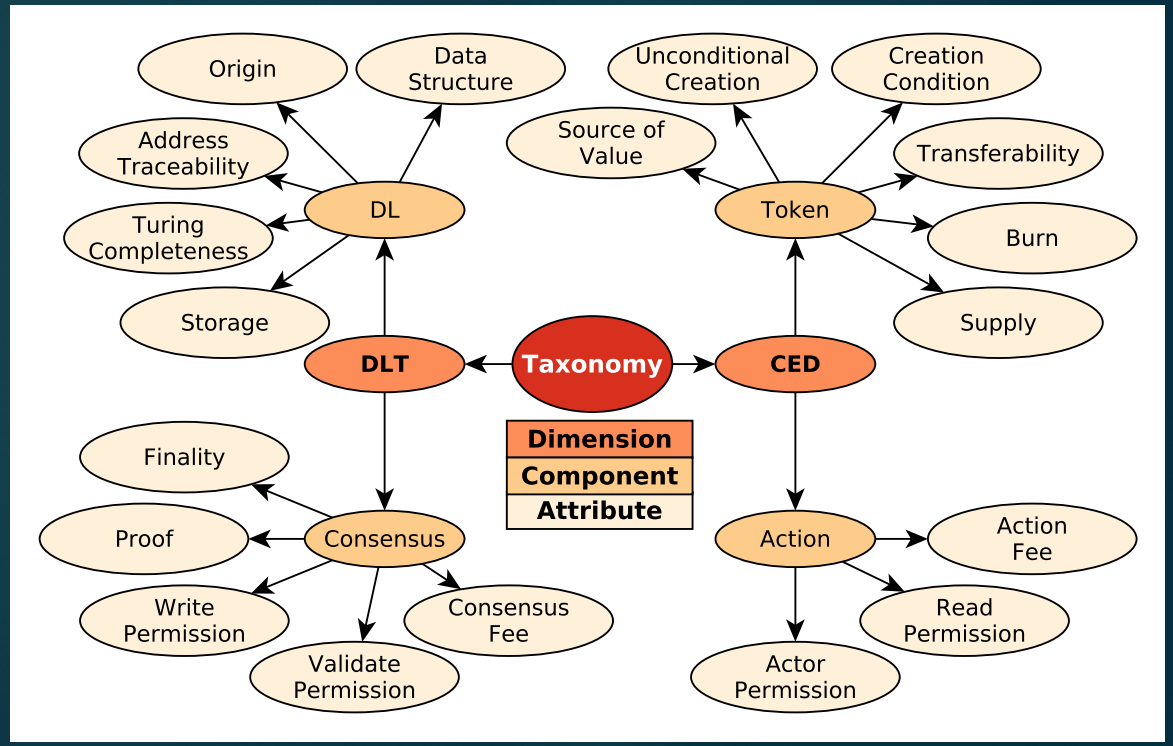
- Ethereum (smart contracts), Golem (computations), Siacoin (on/off-chain storage)
- The source of a token value & what it consists of

Token: *A unit of value issued within a DLT system and which can be used as a medium of exchange or unit of account.*



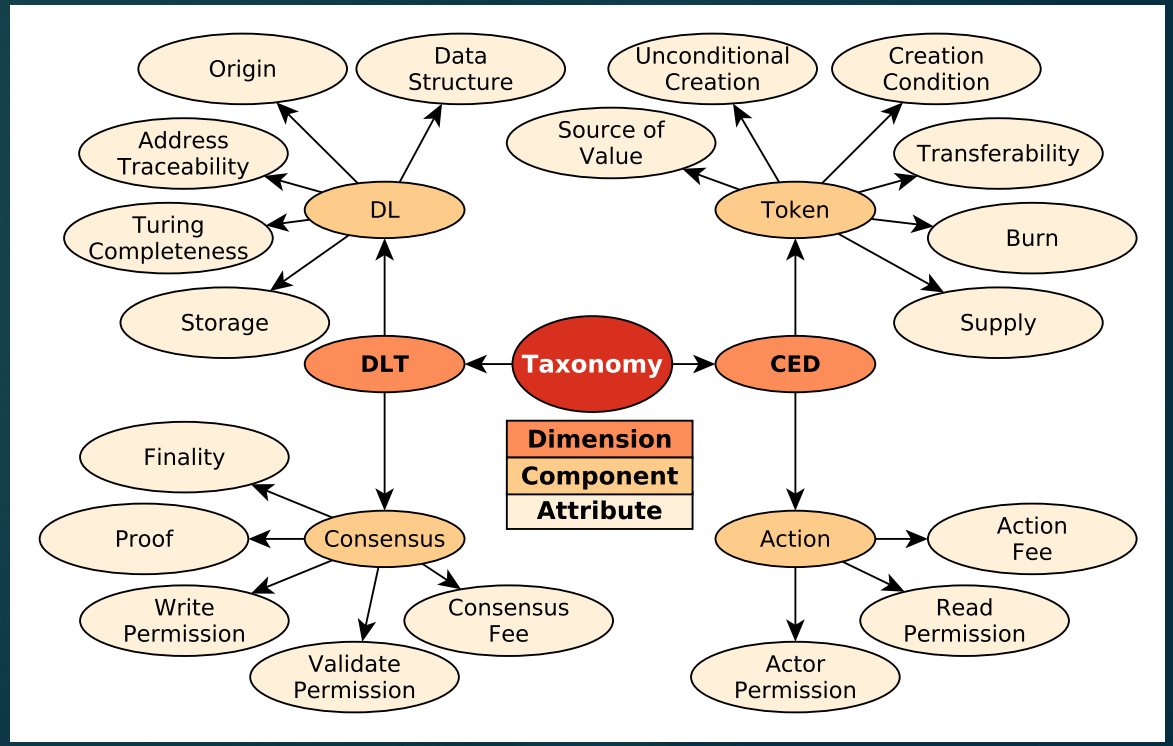
Exarcise: Design Space

System Term	Formal Definition
Blockchain	*blockchain* type
1st layer	*native* ownership
2nd layer	*external* ownership
Permissioned	*restricted* write permission OR *restricted* validator permission
Permissionless	*public* write permission AND *public* validator permission
Public	⟨*permissionless DLT system*⟩ AND *public* actor permission
Private	⟨*permissioned DLT system*⟩ AND *restricted* actor permission
Privacy	*obfuscatable* traceability AND ⟨*public DLT System*⟩
Infrastructure	*yes* turing completeness OR *yes* storage
Blockchain-as-a-Service	∄ Action component attribute AND ∄ Token component attribute
Cryptoeconomic Term	
Utility token	*distributed ledger* underlying OR *action* underlying
Asset token	*token* underlying OR *action* (physical good) underlying
Payment token	*yes* transferability



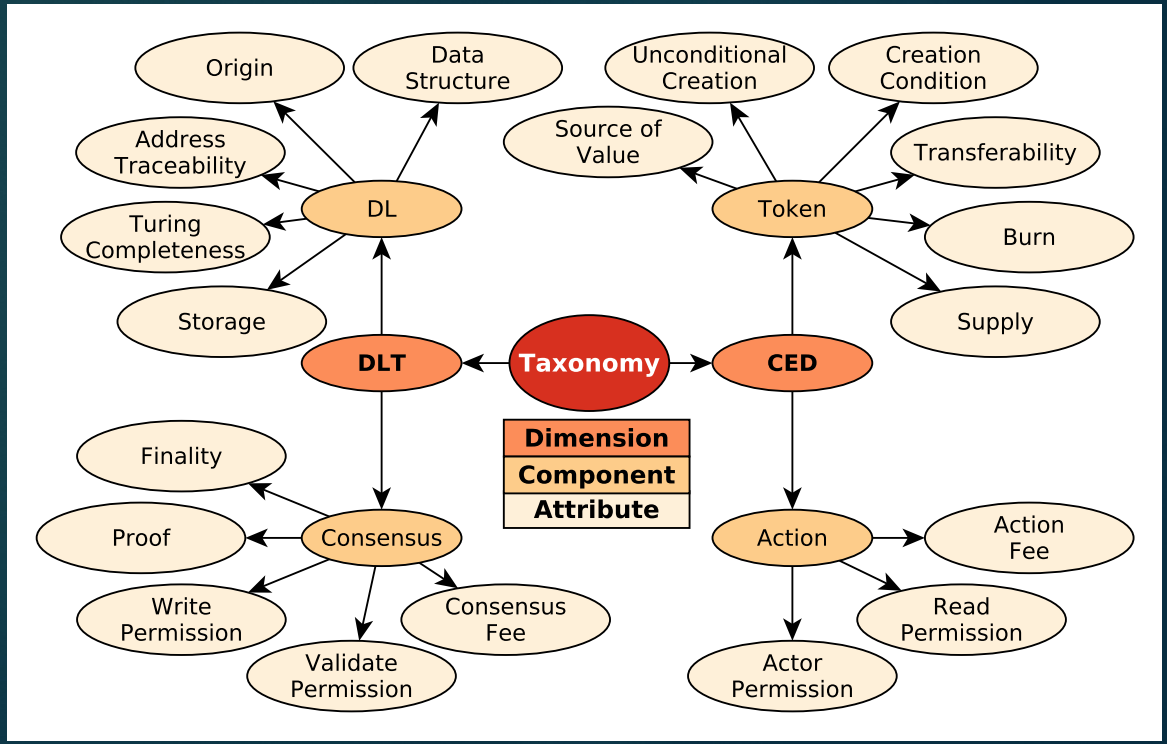
Exercise: Design Space

System Term	Formal Definition
Blockchain	
1st layer	
2nd layer	
Permissioned	
Permissionless	
Public	?
Private	
Privacy	
Infrastructure	
Blockchain-as-a-Service	
Cryptoeconomic Term	
Utility token	
Asset token	?
Payment token	



Exercise: Design Space

System Term	Formal Definition
Blockchain	*blockchain* type
1st layer	*native* ownership
2nd layer	*external* ownership
Permissioned	*restricted* write permission OR *restricted* validator permission
Permissionless	*public* write permission AND *public* validator permission
Public	⟨*permissionless DLT system*⟩ AND *public* actor permission
Private	⟨*permissioned DLT system*⟩ AND *restricted* actor permission
Privacy	*obfuscatable* traceability AND ⟨*public DLT System*⟩
Infrastructure	*yes* turing completeness OR *yes* storage
Blockchain-as-a-Service	∄ Action component attribute AND ∄ Token component attribute
Cryptoeconomic Term	
Utility token	*distributed ledger* underlying OR *action* underlying
Asset token	*token* underlying OR *action* (physical good) underlying
Payment token	*yes* transferability



Exercise: Classification Case Studies

		[1]	[2]	[3]	[4]	[5]	[6]
LEDGER	*Origin*	External: Ethereum (Emulated)	Hybrid: Ethereum (Emulated)	Native	External: Ethereum	Native	External: Ethereum
CONSENSUS	*Data structure*	–	Blockchain	Blockchain	–	Blockchain	–
	Address Trace.	–	Not specified	Traceable	–	Traceable	–
	Turing Compl.	–	Not specified	No	–	No	–
	Storage	–	Yes		–	Not specified	–
	Finality	–	Probabilisitc	Probabilistic	–	Deterministic	–
	Proof	–	Not specified	Not specified	–	PoS	–
	Write Perm.	–	Restricted	Restricted	–	Restricted	–
	Validate Perm.	–	Restricted	Restricted	–	Restricted	–
	Consensus Fee	–	Not specified	Not specified	–	No	–
ACTION	*Actor Permission*	Public	Not specified	Restricted	Public	Restricted	Restricted
	Read Permission	Public	Not specified	Restricted	Restricted	Not specified	Public
	Action Fee	No	Yes	Not specified	No	No	No
TOKEN	*Supply*	–	–	–	–	–	–
	Burn	–	–	–	–	–	–
	Transferability	–	–	–	–	–	–
	Creation Cond.	–	–	–	–	–	–
	Uncond. Creation	–	–	–	–	–	–
	Source of Value	–	–	–	–	–	–

- Focus on blockchain (rather than DAG)
- No cryptoeconomic design
- [3,5]: Native distributed ledgers developed
- [1,2,4,6]: Computationally rely on Ethereum
- [2]: Hybrid approach combining native blockchain & Ethereum
- No use of other DLT system as 1st layer

[1] Khalid, U., Asim, M., Baker, T., Hung, P.C., Tariq, M.A., Raf- ferty, L.: *A decentralized lightweight blockchain-based authentication mechanism for IoT systems*. Clust. Comput. 23, 2067–2087 (2020)

[2] Li, H., Pei, L., Liao, D., Wang, X., Xu, D., Sun, J.: *BDDT: use blockchain to facilitate IoT data transactions*. Clust. Comput. (2020).

[3] Rosa, M., Barraca, J.P., Rocha, N.P.: *Blockchain structures to guarantee logging integrity of a digital platform to support community-dwelling older adults*. Clust. Comput. 23, 1887–1898 (2020)

[4] Gonzlez, J.C., Garca-Daz, V., Nez-Valdez, E.R., Gmez, A.G., Crespo, R.G.: *Replacing email protocols with blockchain-based smart contracts*. Clust. Comput. (2020).

[5] Singh, N., Kumar, T., Vardhan, M.: *Blockchain-based e-cheque clearing framework with trust based consensus mechanism*. Clust. Comput. (2020).

[6] Latif, R.M.A., Farhan, M., Rizwan, O., Hussain, M., Jabbar, S., Khalid, S.: *Retail level blockchain transformation for product supply chain using truffle development platform*. Clust. Comput. (2020).

Studied Systems

1st layer systems (mainchains), no 2nd layer systems (sidechains)

Main token, no secondary tokens

No future system features

Systems than maintain a native cryptoeconomic token

- Blockchain as a service (e.g. Hyperledger Fabric) not considered

CoinMarketCap

- **Market capitalization**: good proxy for viability

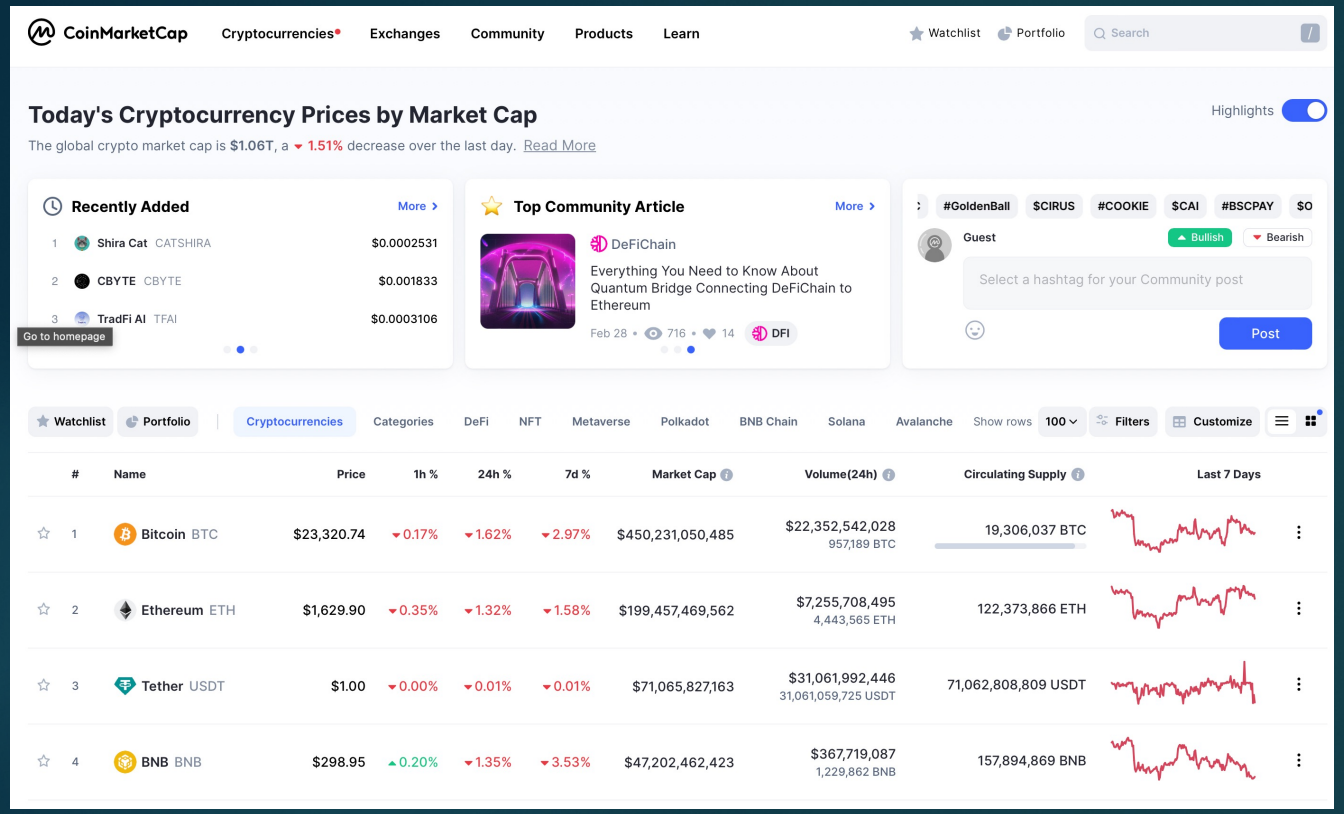
CoinCodeCap

- **Active system development**: # of Github code commitments, contributors, stars

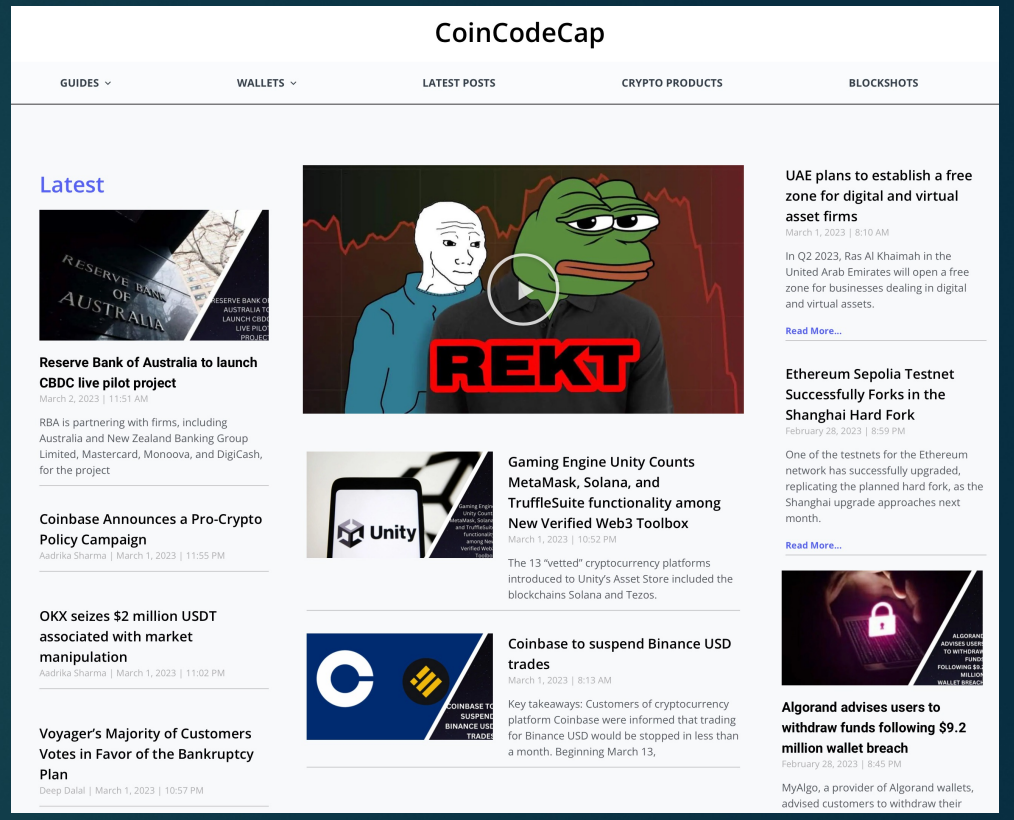
1	Aragon	11	Dash	21	KIN
2	Ark	12	Decred	22	Komodo
3	Augur	13	DigiByte	23	Lisk (Mainchain)
4	Bancor	14	Dogecoin	24	Litecoin
5	Bitcoin	15	EOS	25	Loopring
6	Bitcoin Cash	16	Ethereum	26	MOAC (MotherChain)
7	Bitcoin Gold	17	Factom	27	Monacoin
8	BitShares	18	Gnosis	28	Monero
9	Byteball	19	Golem	29	Nebulas
10	Cardano	20	IOTA	30	NEM

31	NEO	40	Skycoin	40	Skycoin
32	Nexus	41	Steem	41	Steem
33	PIVX	42	Stellar	42	Stellar
34	Qtum	43	Storj	43	Storj
35	ReddCoin	44	Stratis	44	Stratis
36	Ripple	45	Syscoin	45	Syscoin
37	SafeNetwork	46	TRON	46	TRON
38	Siacoin	47	Verge	47	Verge
39	SingularityNET	48	Waves	48	Waves
40	Skycoin	49	Zcash	49	Zcash
		50	Zcoin	50	Zcoin

Datasets

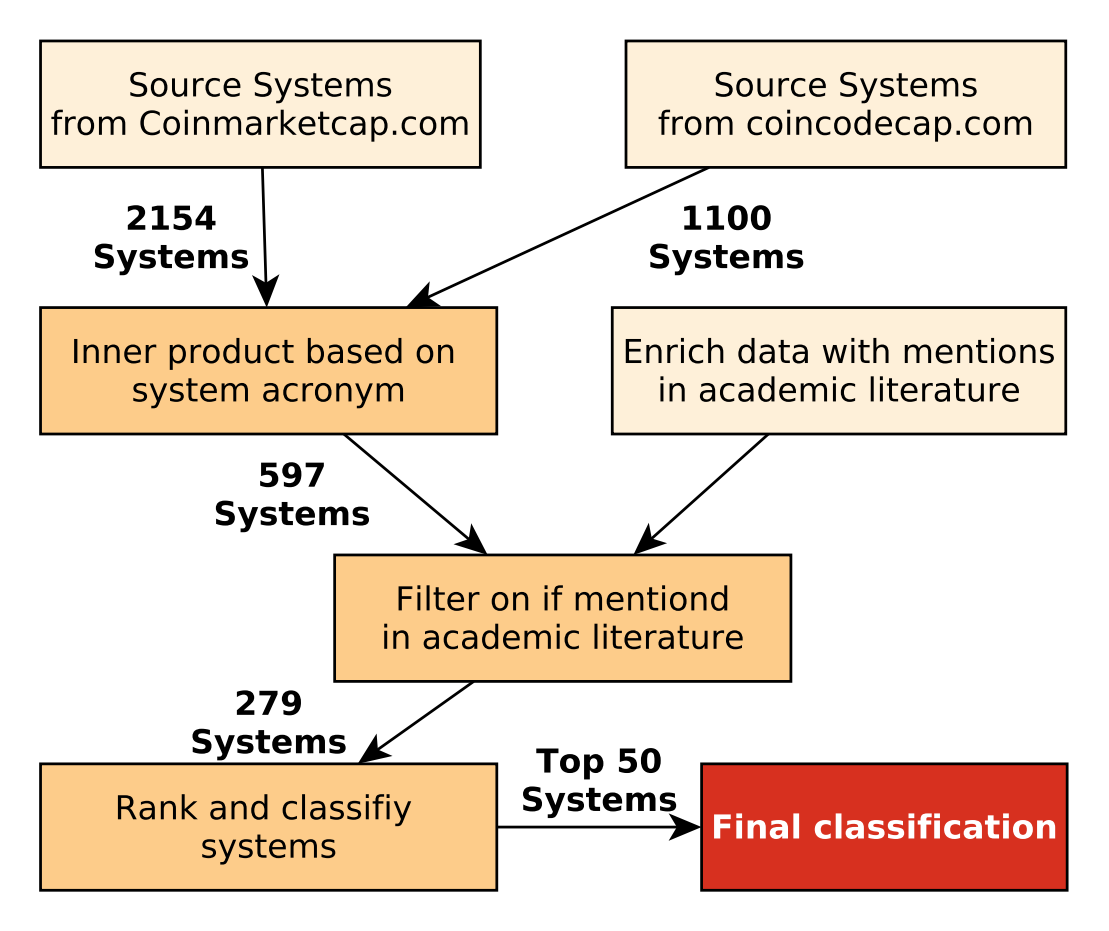


https://coinmarketcap.com



https://coincodecap.com

Data Collection



17th of April 2019

# of mentions in Elsevier ScienceDirect DB>0

- *"PROJECT NAME" AND (Blockchain OR Ledger)*

Classification sources:

- academic literature
- websites
- whitepapers

$r(i)=0.6 \times m_{cap}(i) + 0.3 \times c_{commit}(i) + 0.1 \times c_{contr}(i)$

weights selection: limitations of Github ranks

Blockchain Community Feedback

DLT system	Total
Aragon	2
Ark	1
Bitcoin	1
Bitcoin Cash	2
BitShares	2
Byteball	1
Cardano	3
Dash	6
Decred	1
DigiByte	1
Ethereum	1
Factom	1
Golem	1
IOTA	2
Komodo	1
MOAC-MotherChain	1
Monacoin	1
Monero	4
NEM	1
NEO	1
Nexus	2
PIVX	1
ReddCoin	1
Siacoin	1
Skycoin	1
Steem	1
Stellar	1
Storj	1
Stratis	1
TRON	2
Verge	1
Waves	1
Zcash	2
Total	**50**

22nd of March 2019, 24th of July 2019

Github contributors to DLT systems

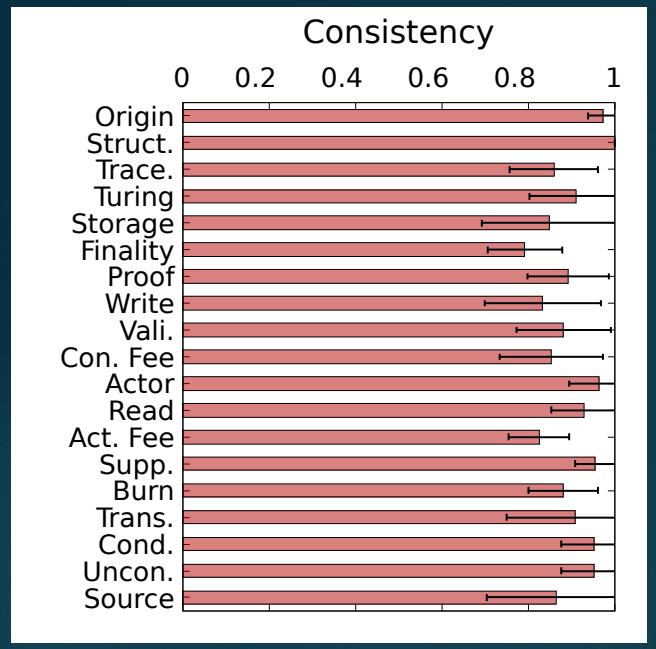
Feedback response: 85/326 ➤26.1%

Feedback completion: 50/85 ➤ 58.8%

Role in Project	Total
Project Lead	7
Core/Team Developer	21
Team Member	8
Advisor	1
Community Developer	4
Community Member	2
Other	7
Total	**50**

Experience	Total
$>$ 3 years	15
1-3 years	29
$<$ 1 year	6
Total	**50**

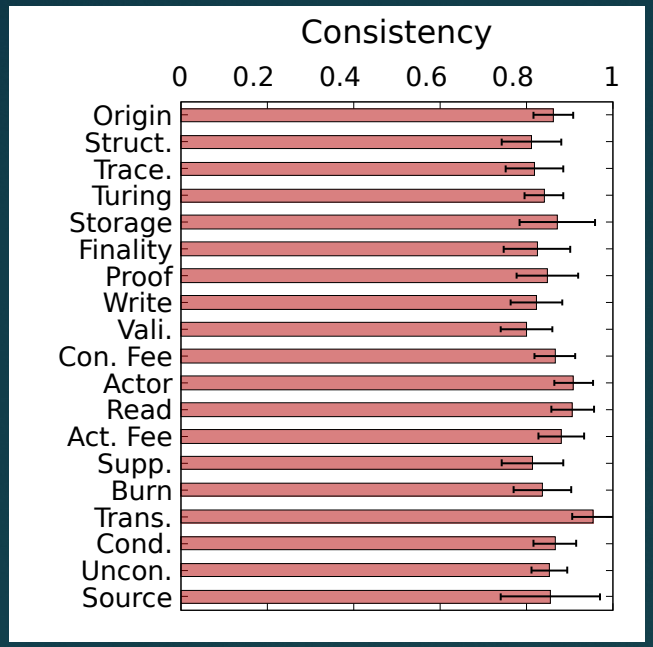
Validation: Classification-Taxonomy



Agreement to classification

Components: 83.7%

Attribute: 89.9%



Expressiveness

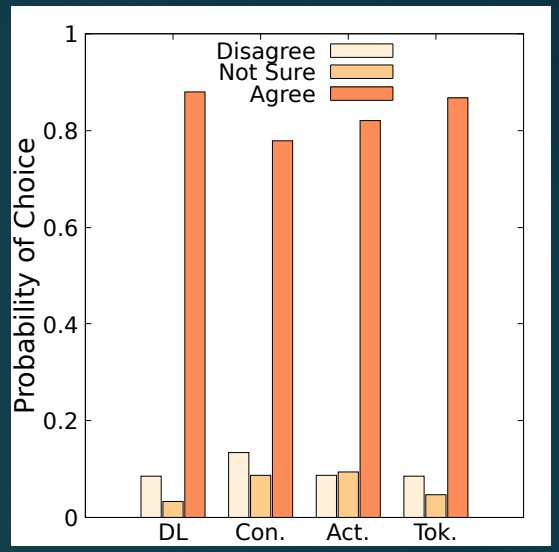
Components: 79%

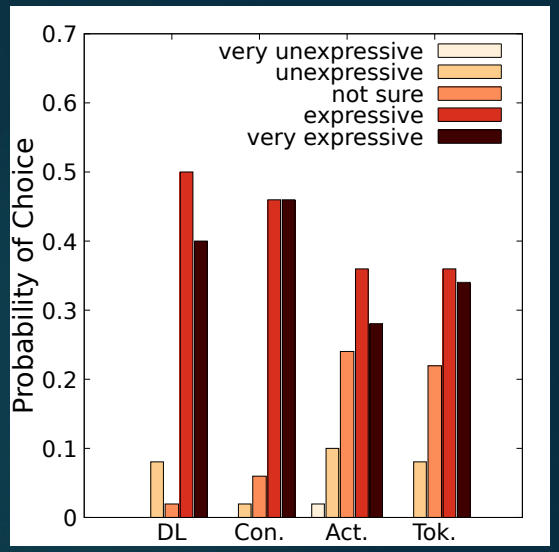
Attribute: 85.5%

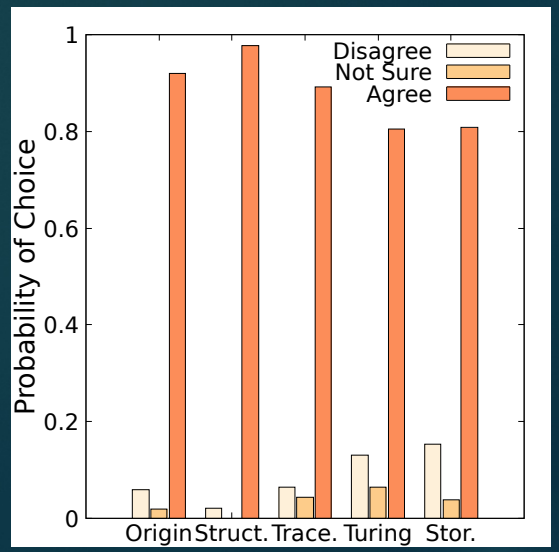
- **A taxonomy is expressive** when it is robust and comprehensive according to Nickerson et al
- *How expressive is [component/attribute] to differentiate between and classify DLT systems?* [Likert]

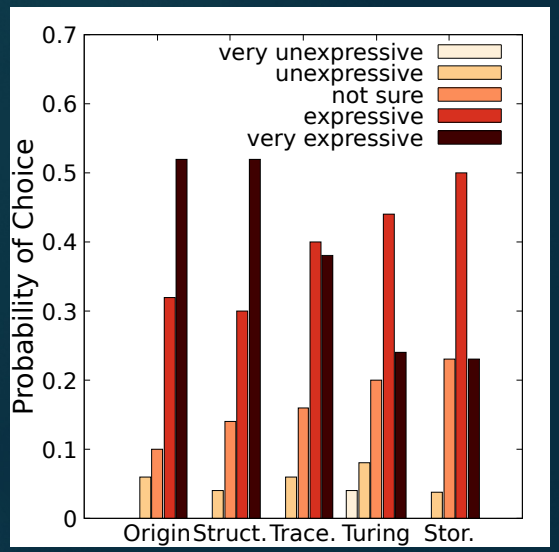
Classification: Distributed Ledger

ID	DLT system	Origin	Type	Address Traceability	Turing Completeness	Storage
1	Aragon	External (Ethereum)	-	-	-	-
2	Ark	Native	Blockchain	Linkable	No	Yes
3	Augur	External (Ethereum)	-	-	-	-
4	Bancor	External (Ethereum and EOS)	-	-	-	-
5	Bitcoin	Native	Blockchain	Linkable	No	Yes
6	Bitcoin Cash	Native	Blockchain	Linkable	No	Yes
7	Bitcoin Gold	Native	Blockchain	Linkable	No	Yes
8	BitShares	Native	Blockchain	Linkable	No	Yes
9	Byteball	Native	DAG	Obfuscatable	No	Yes
10	Cardano	Native	Blockchain	Linkable	No	Yes
11	Dash	Native	Blockchain	Obfuscatable	No	Yes
12	Decred	Native	Blockchain	Linkable	No	No
13	DigiByte	Native	Blockchain	Linkable	No	Yes
14	Dogecoin	Native	Blockchain	Linkable	No	Yes
15	EOS	Native	Blockchain	Linkable	Yes	Yes
16	Ethereum	Native	Blockchain	Linkable	Yes	Yes
17	Factom	Hybrid (Bitcoin)	Blockchain	Linkable	No	Yes
18	Gnosis	External (Ethereum)	-	-	-	-
19	Golem	External (Ethereum)	-	-	-	-
20	IOTA	Native	DAG	Linkable	No	No
21	KIN	Native	Other	Linkable	No	Yes
22	Komodo	Hybrid (Bitcoin)	Blockchain	Obfuscatable	No	Yes
23	Lisk (Mainchain)	Native	Blockchain	Linkable	No	Yes
24	Litecoin	Native	Blockchain	Linkable	No	Yes
25	Loopring	External (Ethereum, Qtum, NEO)	-	-	-	-
26	MOAC (MotherChain)	Native	Blockchain	Linkable	No	Yes
27	Monacoin	Native	Blockchain	Linkable	No	Yes
28	Monero	Native	Blockchain	Obfuscatable	No	Yes
29	Nebulas	Native	Blockchain	Linkable	Yes	Yes
30	NEM	Native	Blockchain	Linkable	No	Yes
31	NEO	Native	Blockchain	Linkable	Yes	Yes
32	Nexus	Native	Blockchain	Linkable	No	Yes
33	PIVX	Native	Blockchain	Obfuscatable	No	Yes
34	Qtum	Native	Blockchain	Linkable	Yes	Yes
35	ReddCoin	Native	Blockchain	Linkable	No	Yes
36	Ripple	Native	Other	Linkable	No	Yes
37	SafeNetwork	Native	Other	Linkable	Yes	Yes
38	Siacoin	Native	Blockchain	Linkable	No	Yes
39	SingularityNET	External (Ethereum)	-	-	-	-
40	Skycoin	Native	Blockchain	Obfuscatable	Yes	Yes
41	Steem	Native	Blockchain	Linkable	No	Yes
42	Stellar	Native	Other	Linkable	No	Yes
43	Storj	External (Ethereum)	-	-	-	-
44	Stratis	Native	Blockchain	Obfuscatable	No	Yes
45	Syscoin	Native	Blockchain	Obfuscatable	No	Yes
46	TRON	Native	Blockchain	Linkable	Yes	Yes
47	Verge	Native	Blockchain	Obfuscatable	No	Yes
48	Waves	Native	Blockchain	Linkable	Yes	Yes
49	Zcash	Native	Blockchain	Obfuscatable	No	Yes
50	Zcoin	Native	Blockchain	Obfuscatable	No	No



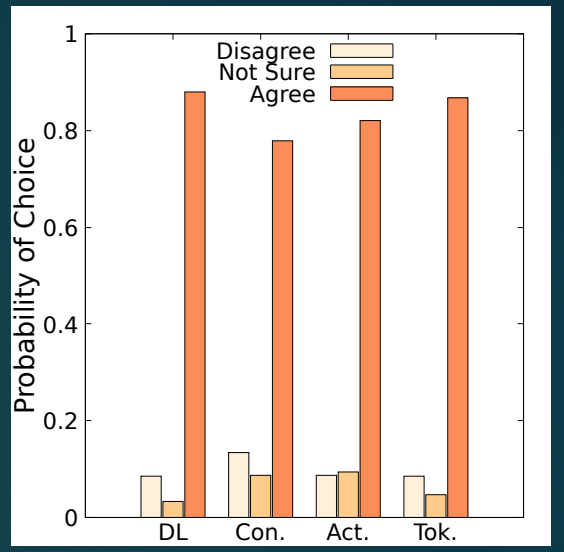


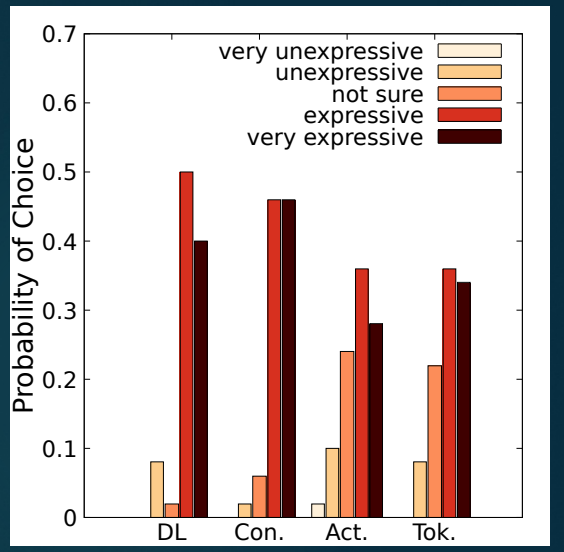


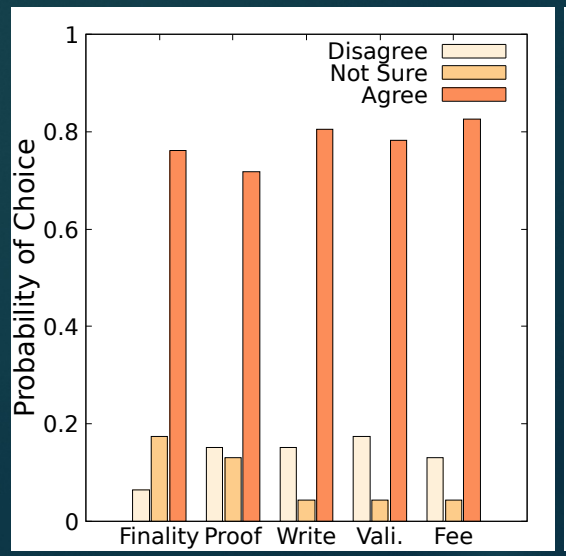


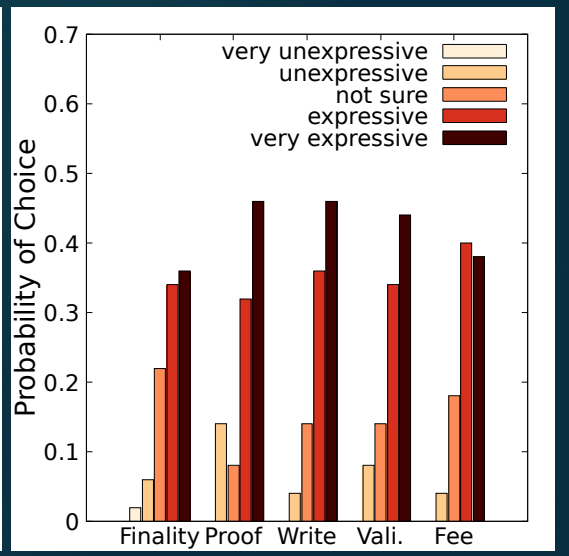
Classification: Consensus

ID	DLT system	Finality	Proof	Write Permission	Validate Permission	Fee
1	Aragon	-	-	-	-	-
2	Ark	Probabilistic	PoS	Restricted	Restricted	Yes
3	Augur	-	-	-	-	-
4	Bancor	-	-	-	-	-
5	Bitcoin	Probabilistic	PoW	Public	Public	Yes
6	Bitcoin Cash	Probabilistic	PoW	Public	Public	Yes
7	Bitcoin Gold	Probabilistic	PoW	Public	Public	Yes
8	BitShares	Deterministic	PoS	Restricted	Public	Yes
9	Byteball	Deterministic	Other	Public	Restricted	Yes
10	Cardano	Probabilistic	PoS	Restricted	Restricted	Yes
11	Dash	Probabilistic	PoW	Public	Public	Yes
12	Decred	Probabilistic	Hybrid	Public	Public	Yes
13	DigiByte	Probabilistic	PoW	Public	Public	Yes
14	Dogecoin	Probabilistic	PoW	Public	Public	Yes
15	EOS	Deterministic	PoS	Restricted	Restricted	No
16	Ethereum	Probabilistic	PoW	Public	Public	Yes
17	Factom	Probabilistic	Other	Restricted	Restricted	No
18	Gnosis	-	-	-	-	-
19	Golem	-	-	-	-	-
20	IOTA	Probabilistic	PoW	Public	Restricted	No
21	KIN	Deterministic	Other	Restricted	Restricted	No
22	Komodo	Probabilistic	PoW	Public	Public	Yes
23	Lisk (Mainchain)	Probabilistic	PoS	Restricted	Restricted	Yes
24	Litecoin	Probabilistic	PoW	Public	Public	Yes
25	Loopring	-	-	-	-	-
26	MOAC (MotherChain)	Probabilistic	PoW	Public	Public	Yes
27	Monacoin	Probabilistic	PoW	Public	Public	Yes
28	Monero	Probabilistic	PoW	Public	Public	Yes
29	Nebulas	Deterministic	Pos	Restricted	Restricted	Yes
30	NEM	Probabilistic	PoS	Restricted	Restricted	Yes
31	NEO	Deterministic	Other	Restricted	Restricted	Yes
32	Nexus	Deterministic	Hybrid	Public	Public	Yes
33	PIVX	Deterministic	PoS	Public	Public	Yes
34	Qtum	Probabilistic	PoS	Public	Public	Yes
35	ReddCoin	Probabilistic	PoS	Public	Public	Yes
36	Ripple	Deterministic	Other	Restricted	Restricted	No
37	SafeNetwork	Deterministic	Other	Restricted	Restricted	No
38	Siacoin	Probabilistic	PoW	Public	Public	Yes
39	SingularityNET	-	-	-	-	-
40	Skycoin	Deterministic	Other	Public	Public	No
41	Steem	Deterministic	PoS	Restricted	Restricted	No
42	Stellar	Deterministic	Other	Restricted	Restricted	No
43	Storj	-	-	-	-	-
44	Stratis	Probabilistic	PoS	Public	Public	Yes
45	Syscoin	Probabilistic	PoW	Public	Public	Yes
46	TRON	Deterministic	PoS	Restricted	Restricted	No
47	Verge	Probabilistic	PoW	Public	Public	Yes
48	Waves	Deterministic	PoS	Restricted	Public	Yes
49	Zcash	Probabilistic	PoW	Public	Public	Yes
50	Zcoin	Probabilistic	PoW	Public	Restricted	Yes



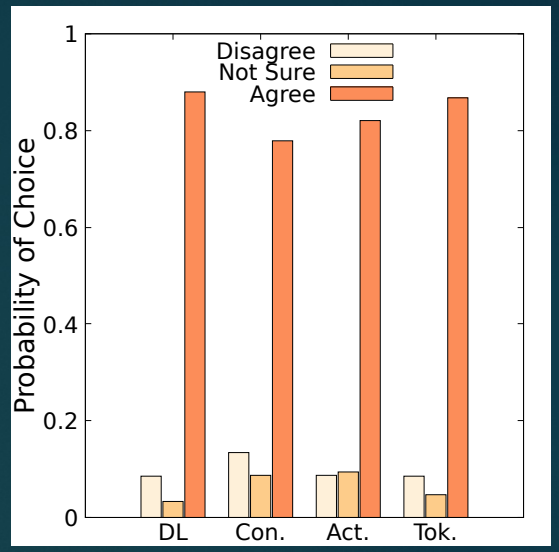


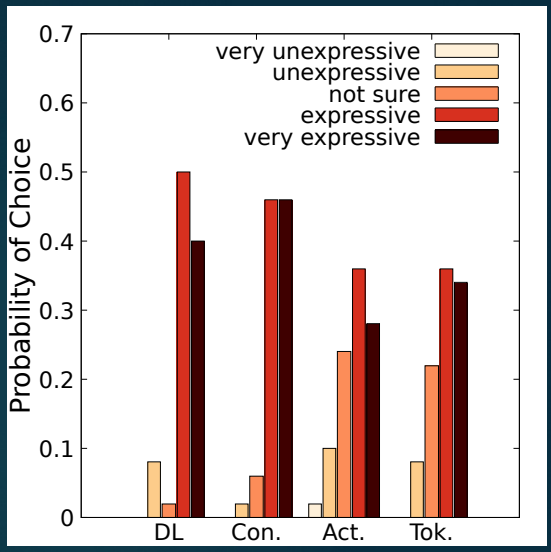


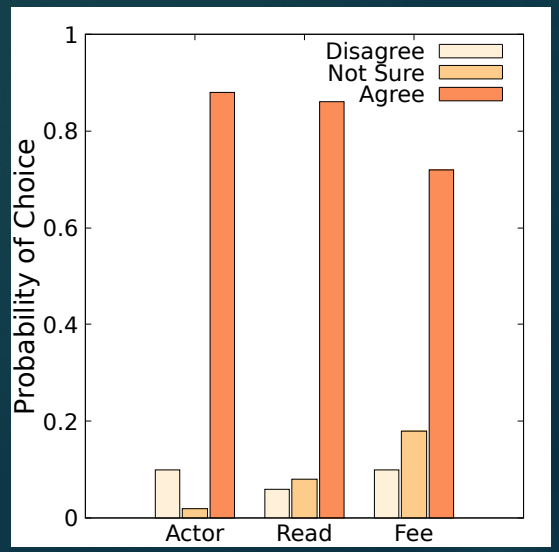


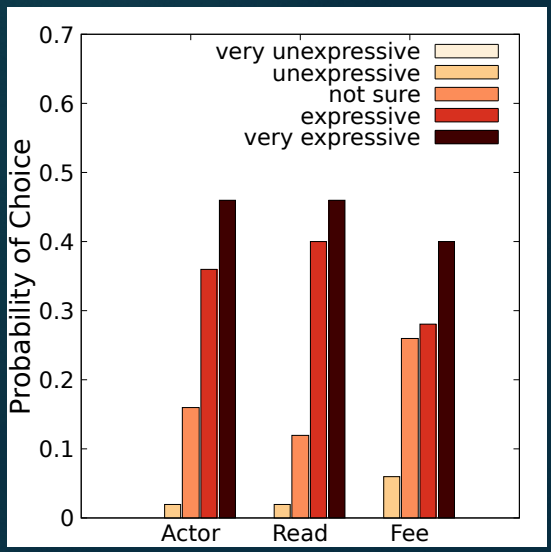
Classification: Action

ID	DLT system	Actor Permission	Read Permission	Fee
1	Aragon	Public	Public	Yes
2	Ark	Public	Public	No
3	Augur	Public	Public	Yes
4	Bancor	Public	Public	No
5	Bitcoin	Public	Public	No
6	Bitcoin Cash	Public	Public	No
7	Bitcoin Gold	Public	Public	No
8	BitShares	Public	Public	Yes
9	Byteball	Public	Restricted	No
10	Cardano	Public	Public	No
11	Dash	Public	Public	Yes
12	Decred	Public	Public	No
13	DigiByte	Public	Public	No
14	Dogecoin	Public	Public	No
15	EOS	Public	Restricted	No
16	Ethereum	Public	Public	No
17	Factom	Public	Public	Yes
18	Gnosis	Public	Public	Yes
19	Golem	Public	Public	Yes
20	IOTA	Public	Public	No
21	KIN	Restricted	Public	Yes
22	Komodo	Public	Public	No
23	Lisk (Mainchain)	Public	Public	Yes
24	Litecoin	Public	Public	No
25	Loopring	Public	Public	Yes
26	MOAC (MotherChain)	Public	Public	No
27	Monacoin	Public	Public	No
28	Monero	Public	Restricted	No
29	Nebulas	Public	Public	No
30	NEM	Public	Public	No
31	NEO	Public	Public	Yes
32	Nexus	Public	Public	No
33	PIVX	Public	Restricted	Yes
34	Qtum	Public	Public	No
35	ReddCoin	Public	Public	No
36	Ripple	Restricted	Restricted	Yes
37	SafeNetwork	Public	Public	No
38	Siacoin	Public	Public	Yes
39	SingularityNET	Restricted	Public	No
40	Skycoin	Public	Public	Yes
41	Steem	Public	Public	No
42	Stellar	Restricted	Public	Yes
43	Storj	Public	Public	Yes
44	Stratis	Public	Public	No
45	Syscoin	Public	Public	Yes
46	TRON	Public	Public	Yes
47	Verge	Public	Public	No
48	Waves	Restricted	Public	No
49	Zcash	Public	Restricted	No
50	Zcoin	Public	Public	No



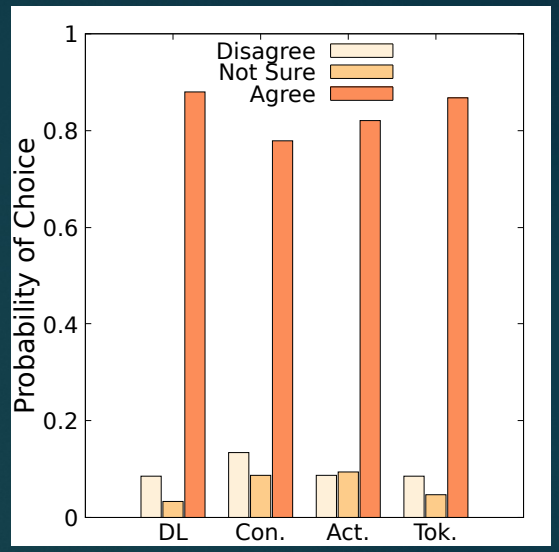


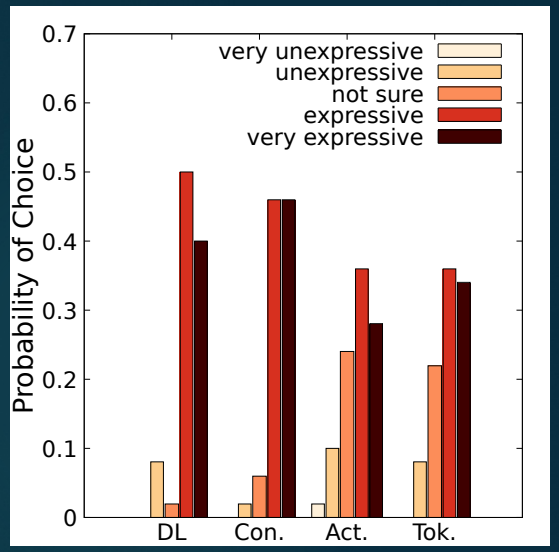


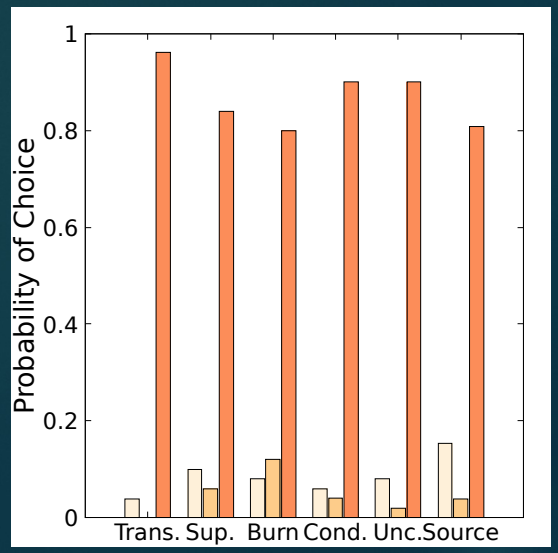


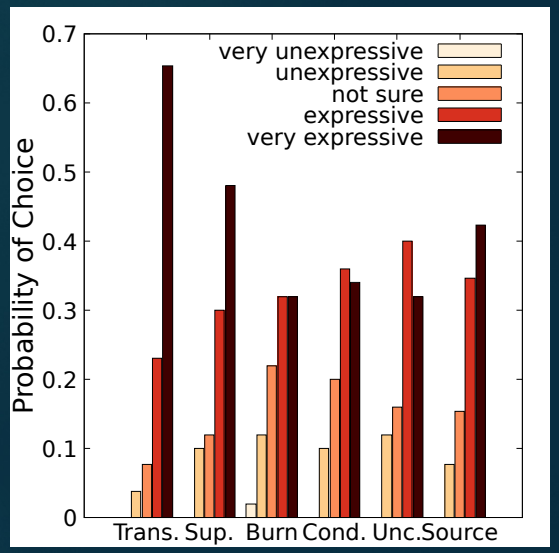
Classification: Token Design

ID	DLT system	Supply Property	Burn Property	Transferability	Creation Condition	Unconditional Creation	Underlying
1	Aragon	Capped	Yes	transferable	Action	Partially	Action
2	Ark	Uncapped	No	transferable	Consensus	Partially	DL, Consensus
3	Augur	Capped	No	transferable	Action	Partially	Action
4	Bancor	Uncapped	Yes	transferable	Action	Partially	Token
5	Bitcoin	Capped	No	transferable	Consensus	None	DL
6	Bitcoin Cash	Capped	No	transferable	Consensus	None	DL
7	Bitcoin Gold	Capped	No	transferable	Consensus	None	DL
8	BitShares	Capped	Yes	transferable	None	All	DL, Consensus, Action
9	Byteball	Capped	No	transferable	None	All	DL
10	Cardano	Capped	No	transferable	Consensus	Partially	DL, Consensus
11	Dash	Capped	No	transferable	Both	None	DL, Action
12	Decred	Capped	No	transferable	Both	Partially	Consensus, Action
13	DigiByte	Capped	No	transferable	Consensus	Partially	DL
14	Dogecoin	Uncapped	No	transferable	Consensus	None	DL
15	EOS	Uncapped	No	transferable	Consensus	Partially	DL
16	Ethereum	Uncapped	No	transferable	Consensus	Partially	DL
17	Factom	Uncapped	Yes	transferable	Consensus	Partially	DL, Action
18	Gnosis	Capped	Yes	transferable	None	All	Token
19	Golem	Capped	No	transferable	None	All	Action
20	IOTA	Capped	No	transferable	None	All	None
21	KIN	Capped	No	transferable	None	All	DL, Action
22	Komodo	Capped	No	transferable	Both	Partially	DL, Token
23	Lisk (Mainchain)	Uncapped	No	transferable	Consensus	Partially	DL, Consensus
24	Litecoin	Capped	No	transferable	Consensus	None	DL
25	Loopring	Capped	Yes	transferable	None	All	Action
26	MOAC (MotherChain)	Capped	No	transferable	Consensus	Partially	DL
27	Monacoin	Capped	No	transferable	Consensus	None	DL
28	Monero	Uncapped	No	transferable	Consensus	None	DL
29	Nebulas	Uncapped	No	transferable	Consensus	Partially	DL, Consensus
30	NEM	Capped	No	transferable	None	All	DL, Consensus
31	NEO	Capped	No	transferable	None	All	Consensus, Action, Token
32	Nexus	Uncapped	No	transferable	Both	Partially	DL, Consensus
33	PIVX	Capped	Yes	transferable	Both	Partially	DL, Consensus, Action, Token
34	Qtum	Capped	No	transferable	Consensus	Partially	DL, Consensus
35	ReddCoin	Uncapped	No	transferable	Consensus	Partially	DL, Consensus
36	Ripple	Capped	Yes	transferable	None	All	DL, Action
37	SafeNetwork	Capped	Yes	transferable	Consensus	Partially	Action
38	Siacoin	Uncapped	Yes	transferable	Consensus	Partially	DL, Action
39	SingularityNET	Capped	No	transferable	None	All	Action
40	Skycoin	Capped	No	transferable	None	All	Token
41	Steem	Uncapped	No	transferable	Both	None	Token
42	Stellar	Uncapped	No	transferable	None	All	DL, Action
43	Storj	Capped	No	transferable	None	All	Action
44	Stratis	Uncapped	No	transferable	Consensus	Partially	DL, Consensus, A
45	Syscoin	Capped	No	transferable	Both	Partially	DL, Action
46	TRON	Uncapped	Yes	transferable	Consensus	Partially	DL, Token
47	Verge	Capped	No	transferable	Consensus	None	DL
48	Waves	Capped	No	transferable	None	All	DL, Consensus
49	Zcash	Capped	No	transferable	Consensus	Partially	DL
50	Zcoin	Capped	Yes	transferable	Both	Partially	DL, Token









Blockchain Design Patterns

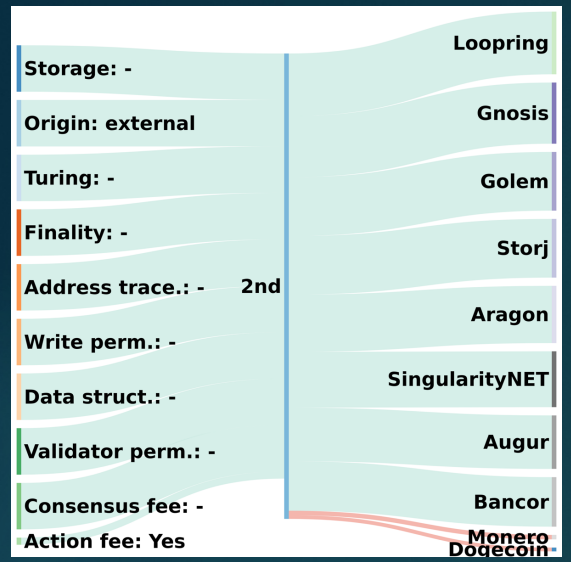
Multiple correspondence analysis

Key dimensions (92.6% of variation):

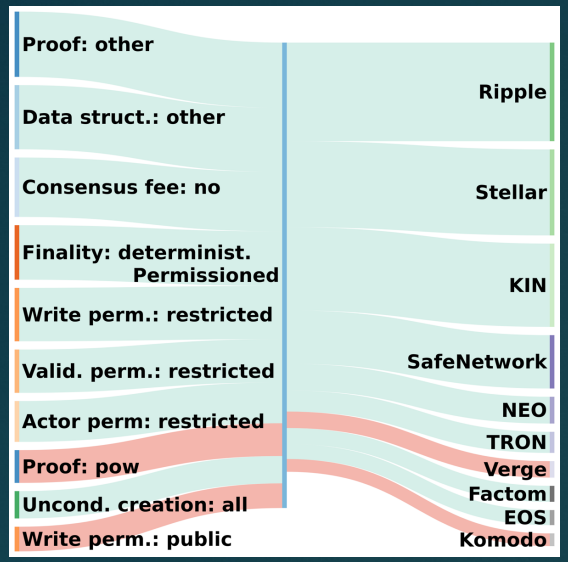
1. Layering: origin attribute (native vs. external)
2. Participation: openness (permissioned vs. permisionedless)
3. Staking: PoS for concensus
4. Cryptoeconomic complexity: complex (e.g. token interactions) to simple (no burn)

Dim	Description	Eigenvalue	Corrected variances	
			Benzceri	*Greenacre*
1	Layering	0.311	0.764	0.679
2	Participation	0.060	0.148	0.132
3	Staking capability	0.013	0.032	0.029
4	Cryptoec. complexity	0.007	0.018	0.016
5		0.006	0.014	0.012
6		0.003	0.008	0.007
7		0.002	0.006	0.005
8		0.002	0.004	0.004
9		0.001	0.003	0.002
10		0.001	0.002	0.001
11		0.001	0.001	0.001
12		0	0.001	0.000
13		0	0	0

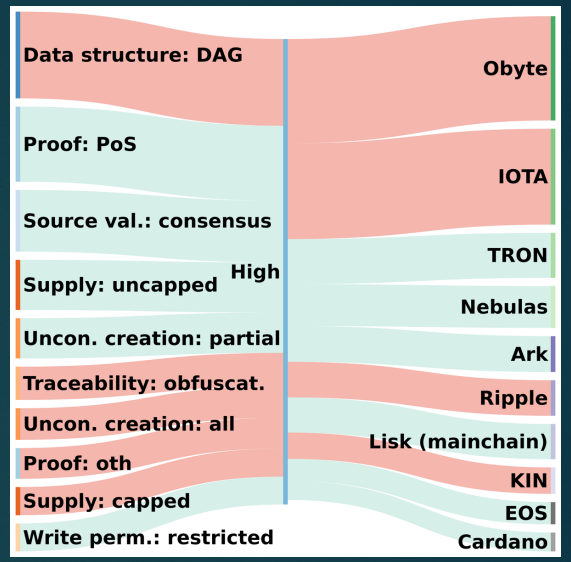
Blockchain Design Patterns



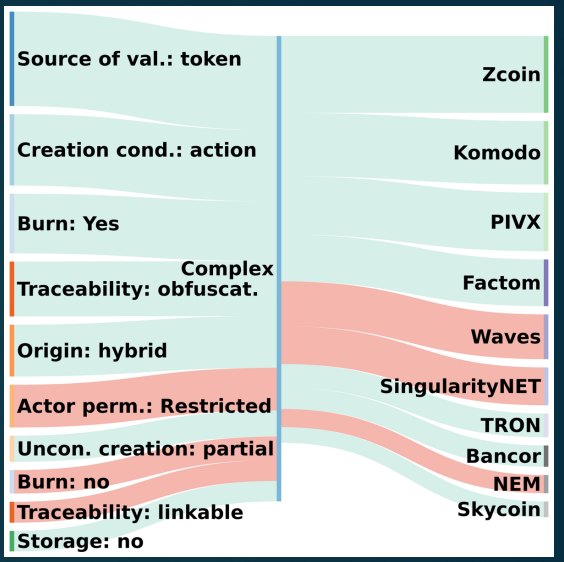
Layering



Participation



Staking



Token Complexity

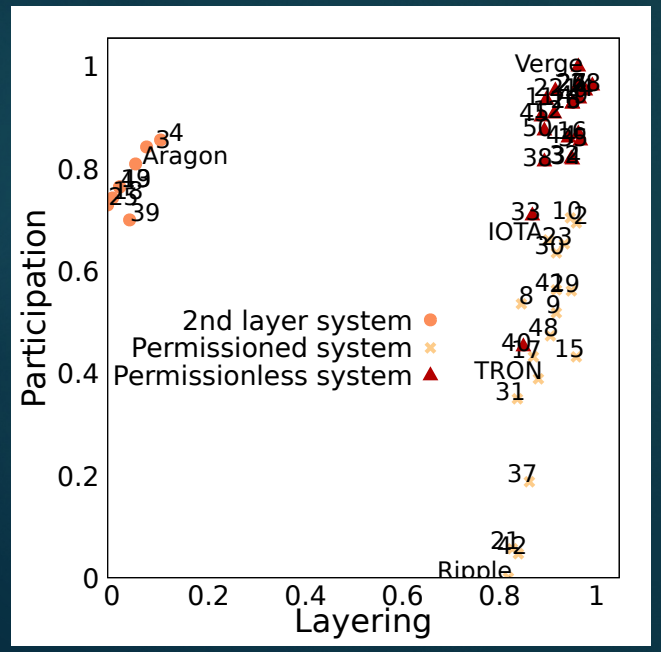
Blockchain Design Patterns

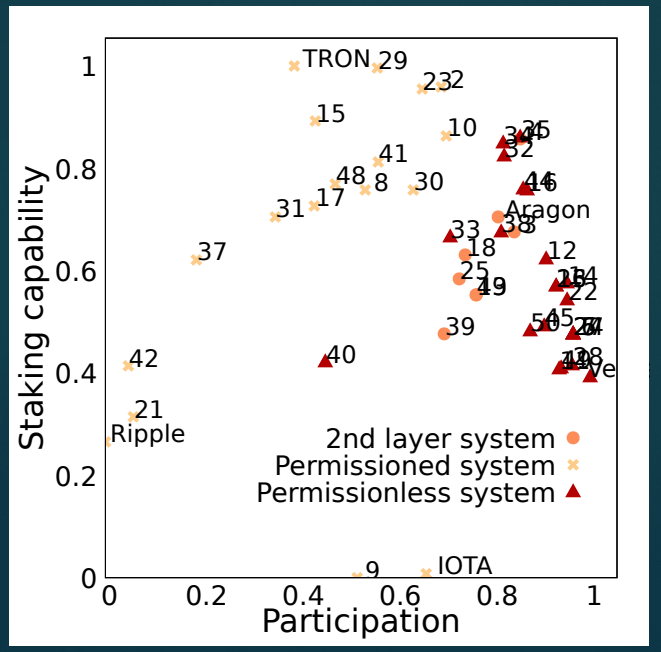
K-Means

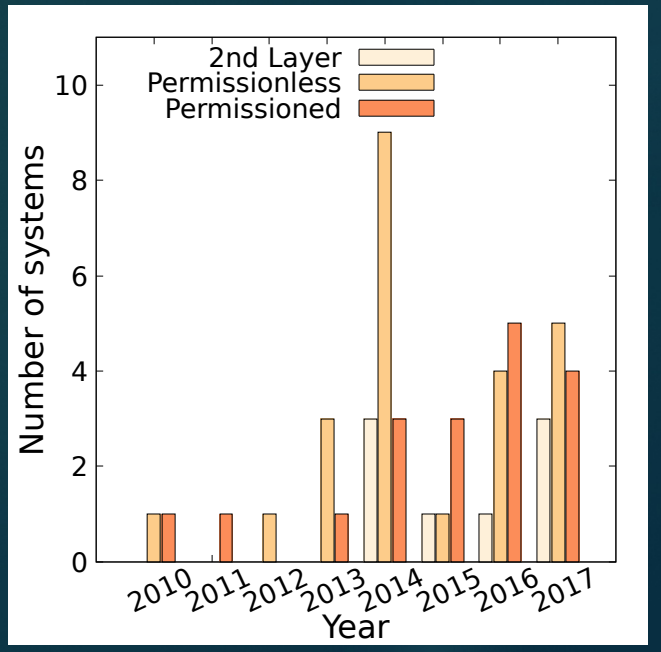
# of clusters (k=3):

- Bootstrap evaluation
- Silhouette & Calinski-Harabasz

k	Boot	Cluster					
		1	2	3	4	5	6
2	Mean	0.91	0.95	–	–	–	–
	brd	12	1	–	–	–	–
3	Mean	0.96	0.97	1	–	–	–
	brd	0	0	0	–	–	–
4	Mean	0.75	0.91	0.99	75	–	–
	brd	19	1	1	21	–	–
5	Mean	0.71	0.64	0.43	0.62	1	–
	brd	25	32	80	25	0	–
6	Mean	0.82	1	0.70	0.65	0.5	0.64
	brd	19	0	23	33	68	44







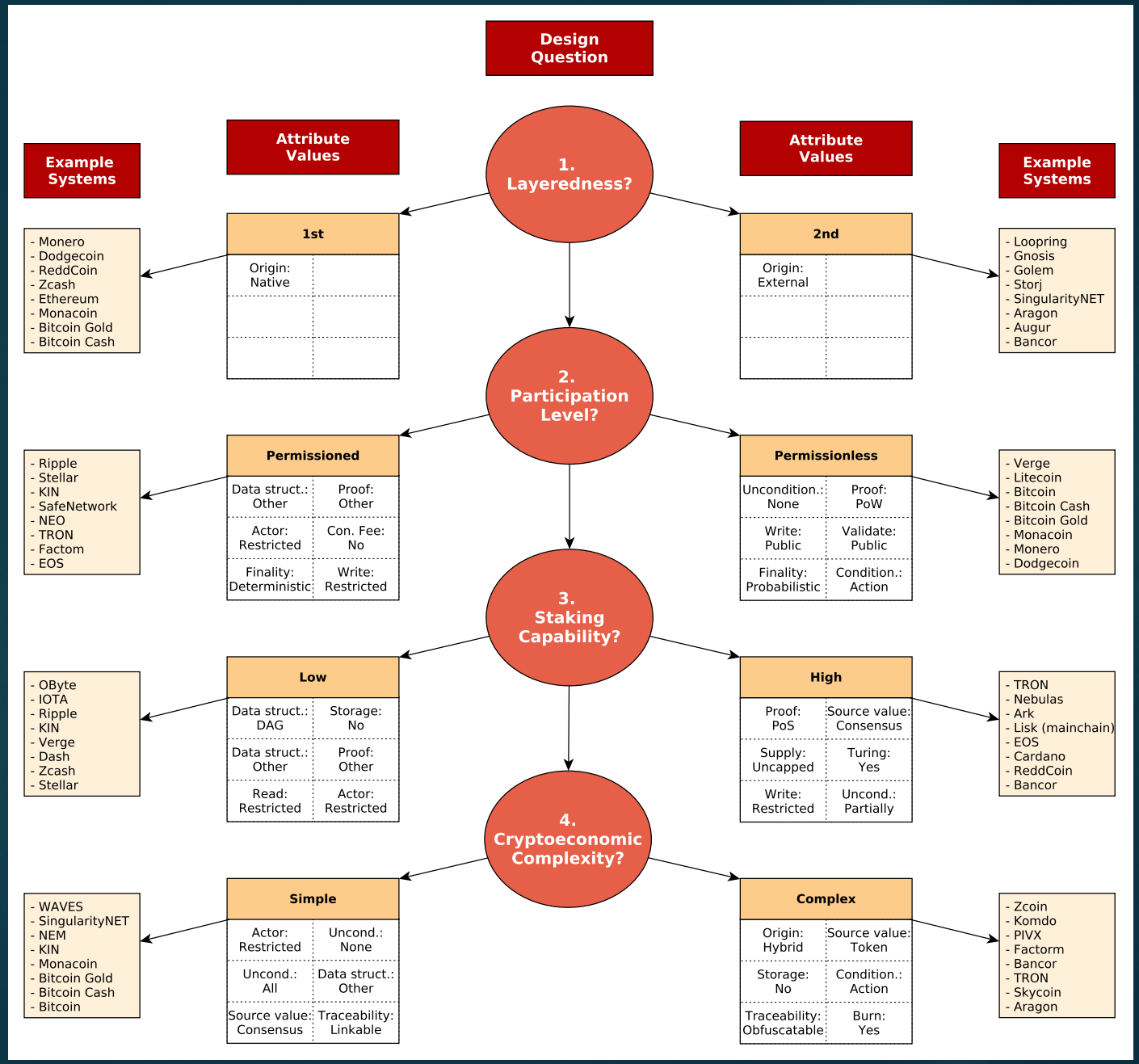
Blockchain Design Patterns

How ledgers are clustered based on the extracted design patterns:

id	DLT system	Cluster	Layer	Participation	Stakeability	Complexity
1	Aragon	1	0.06	0.81	0.71	0.64
2	Ark	0	0.97	0.69	0.96	0.31
3	Augur	1	0.08	0.84	0.68	0.49
4	Bancor	1	0.11	0.86	0.86	0.76
5	Bitcoin	2	0.97	0.96	0.48	0.22
6	Bitcoin Cash	2	0.97	0.96	0.48	0.22
7	Bitcoin Gold	2	0.97	0.96	0.48	0.22
8	BitShares	0	0.85	0.54	0.76	0.36
9	Byteball	0	0.93	0.52	0	0.6
10	Cardano	0	0.95	0.7	0.86	0.28
11	Dash	2	0.91	0.94	0.41	0.54
12	Decred	2	0.92	0.91	0.62	0.54
13	DigiByte	2	0.96	0.93	0.57	0.33
14	Dogecoin	2	0.99	0.95	0.58	0.24
15	EOS	0	0.97	0.43	0.89	0.5
16	Ethereum	2	0.97	0.87	0.76	0.35
17	Factom	0	0.88	0.43	0.73	0.89
18	Gnosis	1	0.01	0.74	0.63	0.57
19	Golem	1	0.03	0.76	0.55	0.24
20	IOTA	0	0.91	0.66	0.01	0.53
21	KIN	0	0.83	0.06	0.31	0.2
22	Komodo	2	0.92	0.95	0.54	0.95
23	Lisk-mainchain	0	0.94	0.65	0.96	0.36
24	Litecoin	2	0.97	0.96	0.48	0.22
25	Loopring	1	0	0.73	0.59	0.4
26	MOAC-MotherChain	2	0.96	0.93	0.57	0.33
27	Monacoin	2	0.97	0.96	0.48	0.22
28	Monero	2	1	0.96	0.42	0.47
29	Nebulas	0	0.96	0.56	1	0.34
30	NEM	0	0.93	0.63	0.76	0.16
31	NEO	0	0.84	0.35	0.71	0.46
32	Nexus	2	0.96	0.82	0.82	0.44
33	PIVX	2	0.88	0.71	0.67	0.91
34	Qtum	2	0.96	0.82	0.85	0.28
35	ReddCoin	2	0.97	0.86	0.86	0.31
36	Ripple	0	0.83	0	0.27	0.42
37	SafeNetwork	0	0.87	0.19	0.62	0.56
38	Siacoin	2	0.9	0.82	0.68	0.55
39	SingularityNET	1	0.04	0.7	0.48	0.01
40	Skycoin	2	0.86	0.45	0.42	0.72
41	Steem	0	0.93	0.56	0.81	0.6
42	Stellar	0	0.85	0.05	0.41	0.23
43	Storj	1	0.03	0.76	0.55	0.24
44	Stratis	2	0.95	0.86	0.76	0.45
45	Syscoin	2	0.89	0.9	0.49	0.64
46	TRON	0	0.89	0.39	1	0.76
47	Verge	2	0.97	1	0.39	0.37
48	Waves	0	0.91	0.47	0.77	0
49	Zcash	2	0.97	0.94	0.41	0.55
50	Zcoin	2	0.9	0.88	0.48	1

Design Guideline

- Linking design decisions
- Reduces modeling & accelerates the design phase
- Differentiate existing systems & their design
- Reason about current emerging practice
- Novelty, innovation



Questions?

Blockchain & Cryptoeconomics

DR. EVANGELOS POURNARAS

6. Applications: Drones

Unmanned Aerial Vehicles (UAVs)

Applications

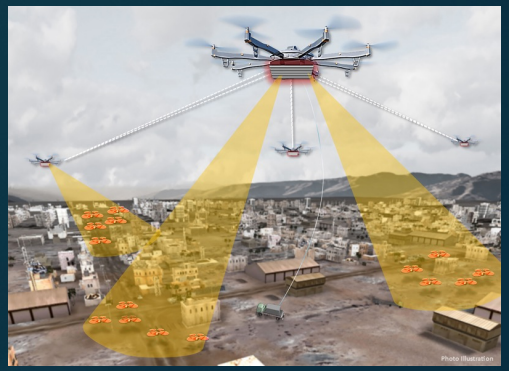
- Rescue operations
- Agriculture & farming
- Delivery goods & medical supplies
- Distributed monitoring & sensing
- Inventory management
- Relayed telecommunication

Elements of UAV missions:

- UAVs
- Base stations
- Collected sensor data
- Communication & interactions
- Flying & sensed environment





Why Blockchain for UAVs

- Secure transacting for data sharing/collection, charging, deliveries
- Fault-tolerant & resilient interactive swarm operations
- Traceability & verification of UAV operations & data

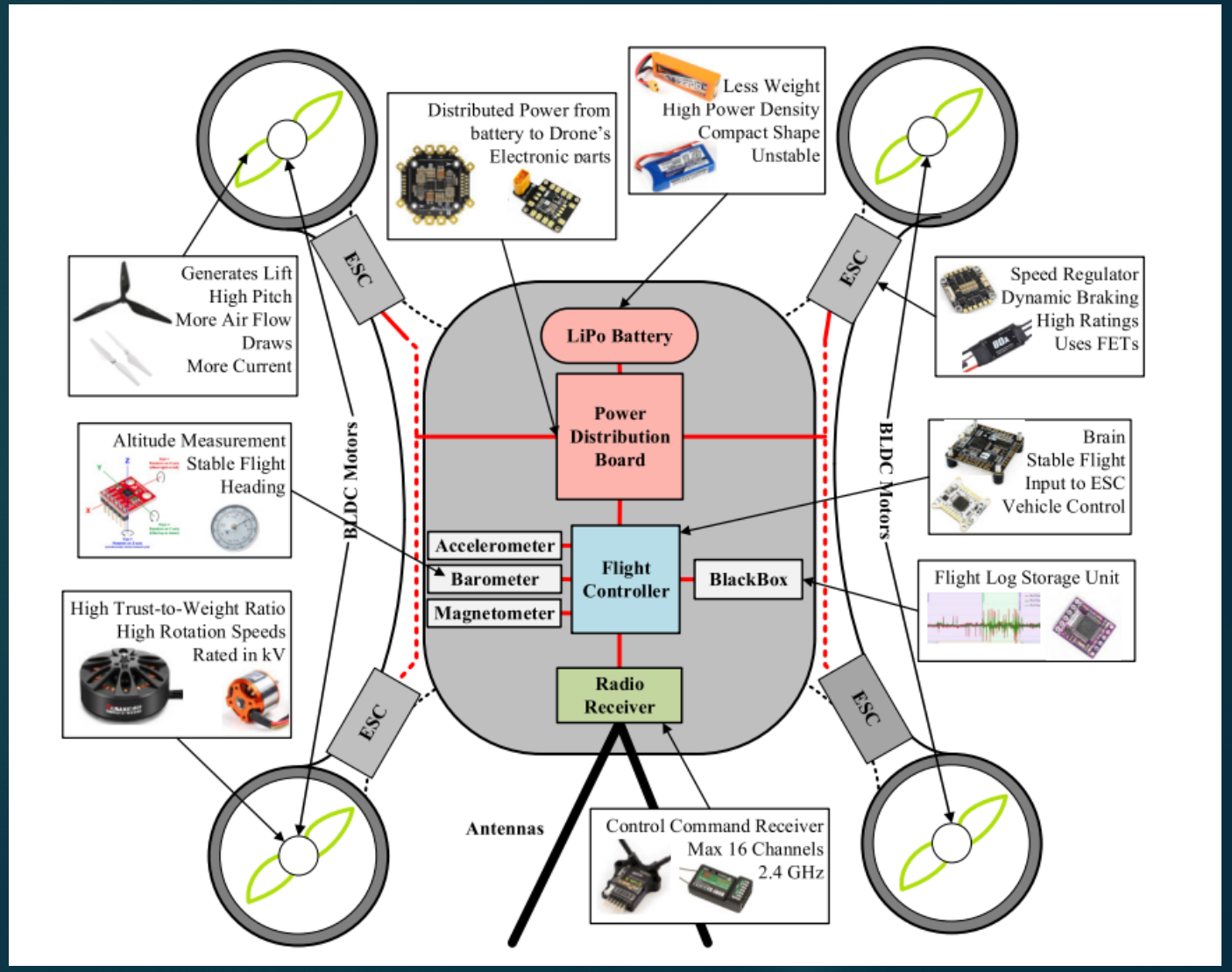
Classification of UAVs

- Multi-rotor
- Fixed wing
- Hybrid fixed/rotary wing

- Micro: $\leq 100\ g$
- Very small: $> 100\ g, \leq 2kg$
- Small: $> 2kg, \leq 25kg$
- Medium: $> 25kg, \leq 150kg$
- Large: $> 150kg,$

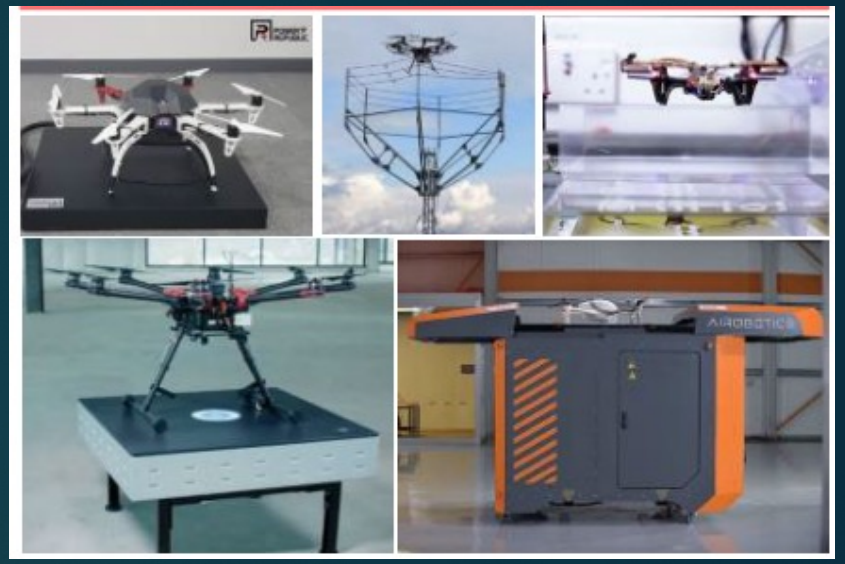
UAV Structure

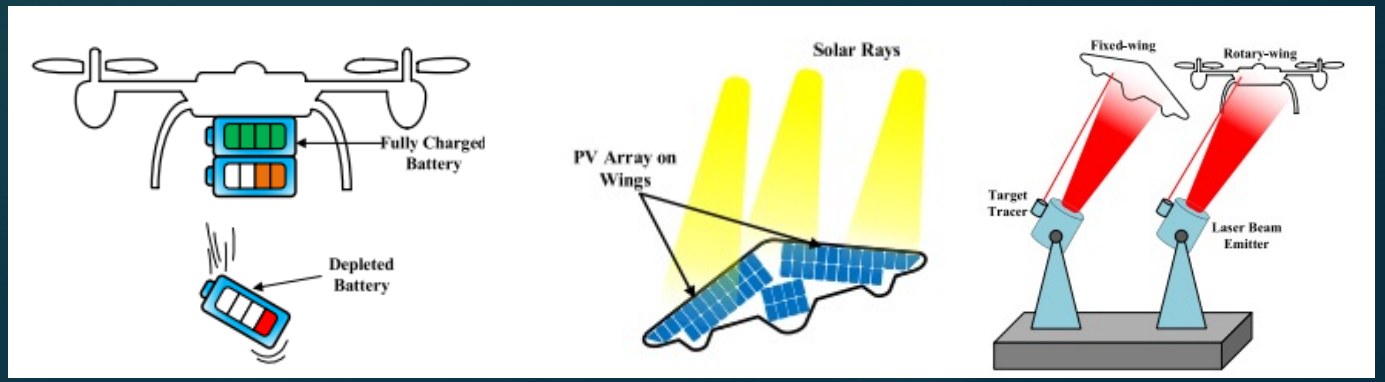
- Flight controller
- FPV camera
- LiPo battery
- Other sensors



UAV Battery Charging

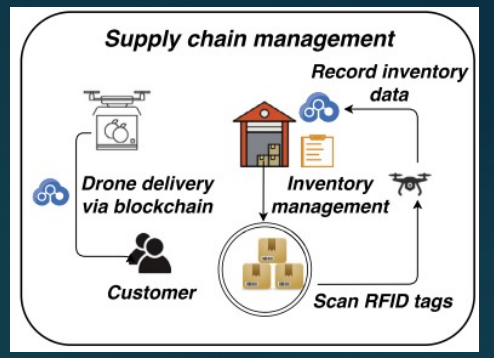
- **Flighing range**: limited due to battery capacity
- **Weight**: reduces max speed, range, altitude, load (e.g. equipment) that can be carried
- **Wireless battery charging**:
  - Capacitive, inductive & magnetic resonant charging
  - Laser & microwave-based charging
- **Non-wireless charging**:
  - Battery dumping
  - Solar rays
  - Gust soaring

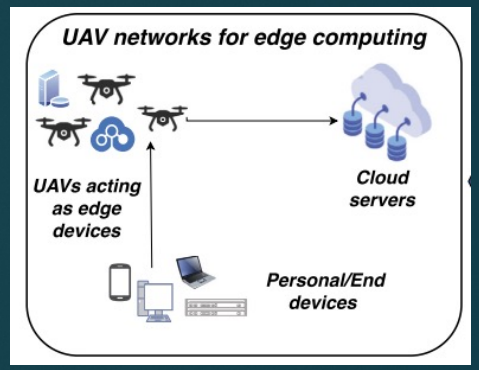


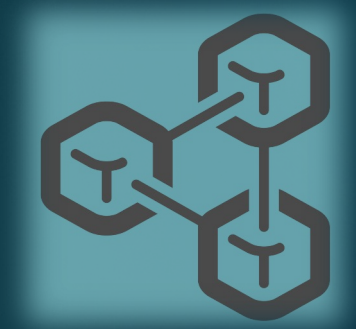


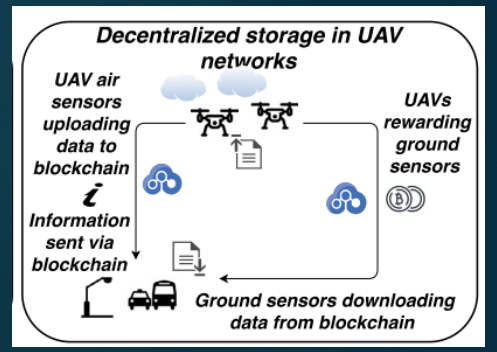
Blockchain UAV Applications

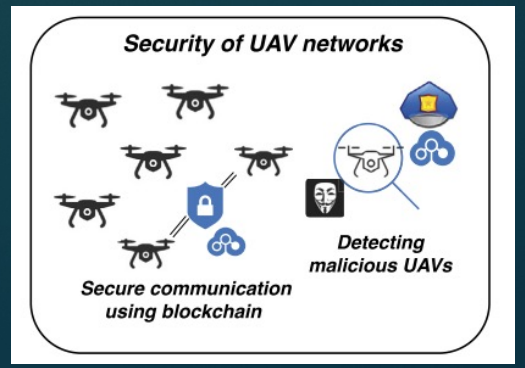
- Supply-chain management
- Decentralized storage
- Coordinated UAV services
- Security of UAV networks
- UAV networks for edge computing

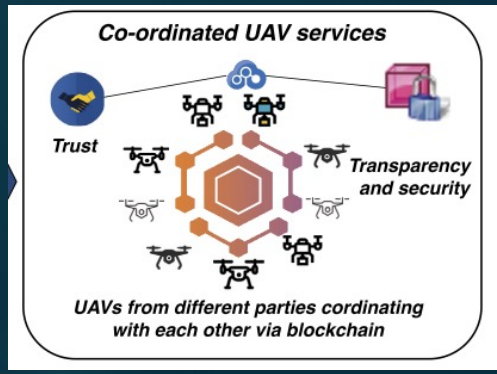












Supply-chain management

Challenge

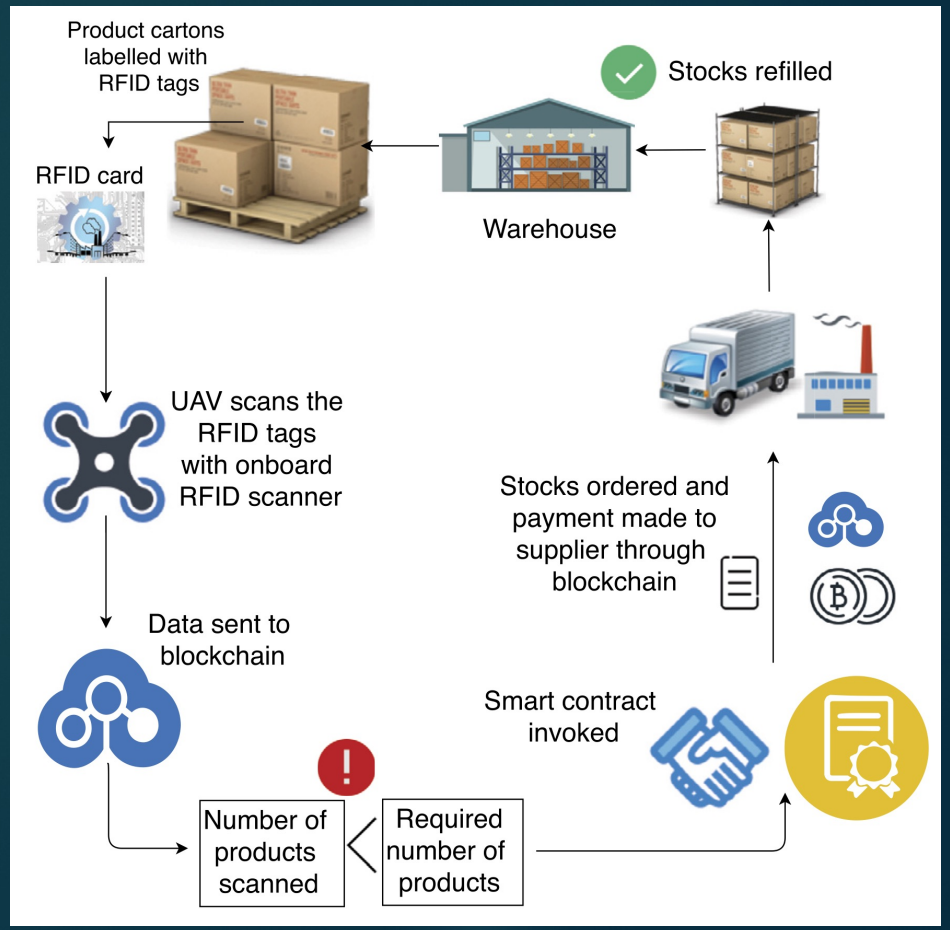
- Inventory management performed by multiple humans is costly, prone to delays & accounting errors

Solution:

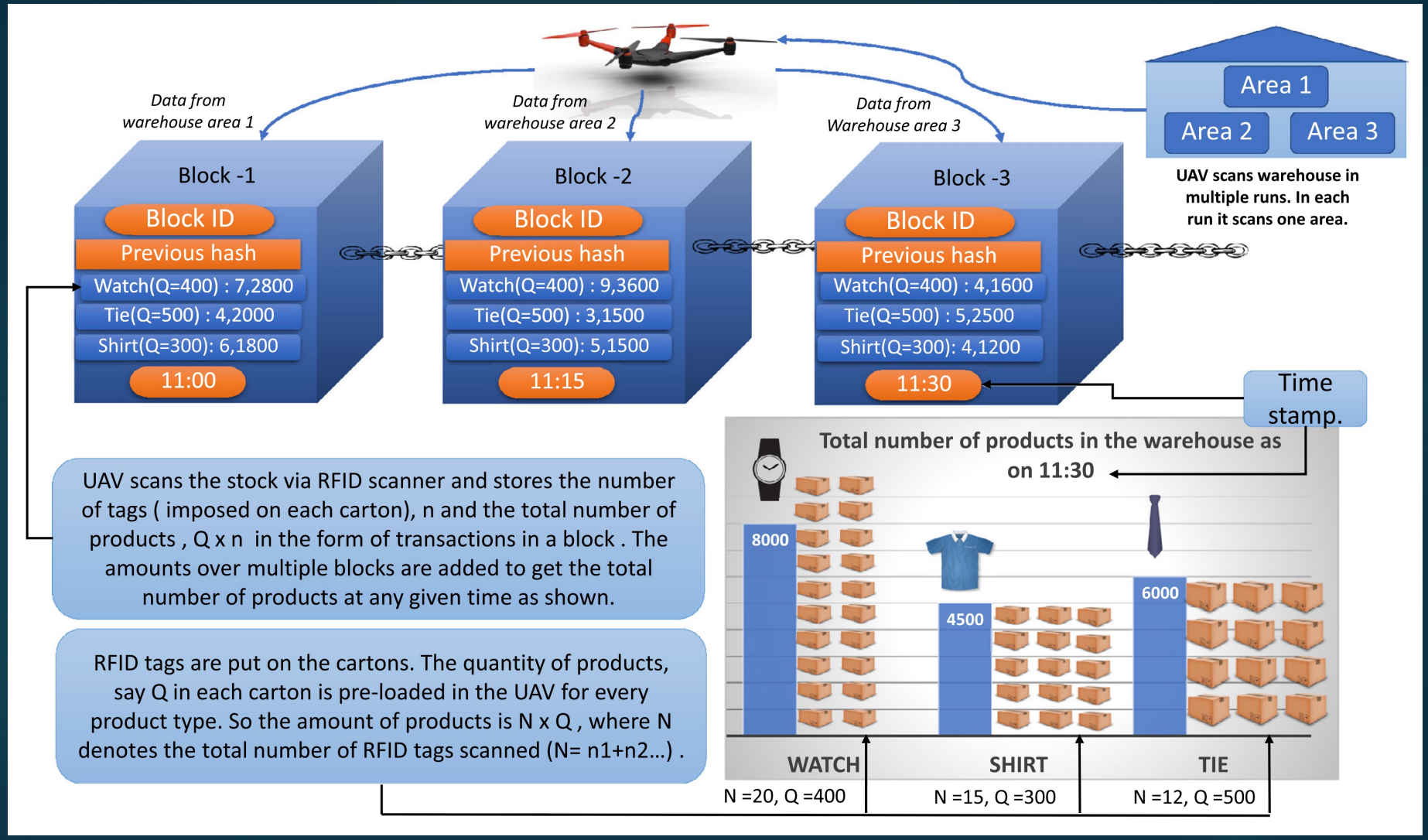
- Products are scanned (e.g. RFIDs) automatically by UAVs. They are validated & recorded in the blockchain for transparency

Blockchain role:

- Automated & secure data storage & verification, e.g. flow of stocks
- Automated & verified transactions, e.g. order of new supplies
- Automated & secure tasks execution, e.g. composite mission accomplishment



Supply-chain management



Coordinated UAVs

Challenge

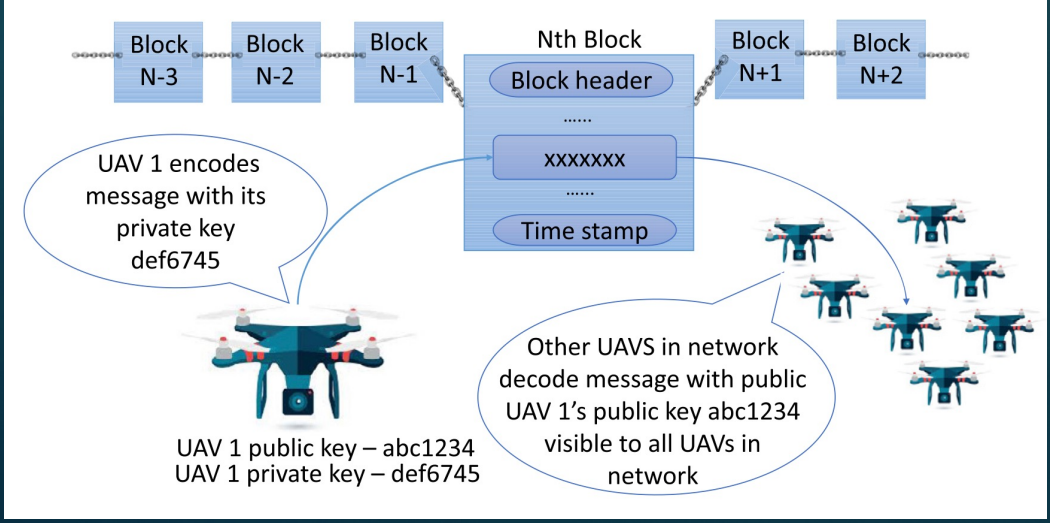
- Lack of reliable shared knowledge, secure interactions, secure resource sharing between different service providers

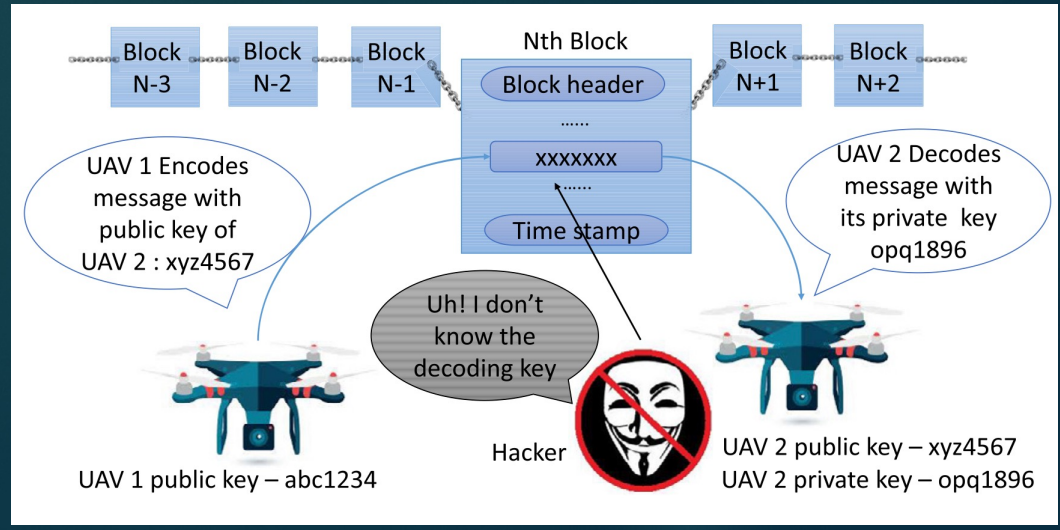
Solution:

- Sensing as a service
- Proof of location
- Secure load-distribution between vendors
- Synchronization

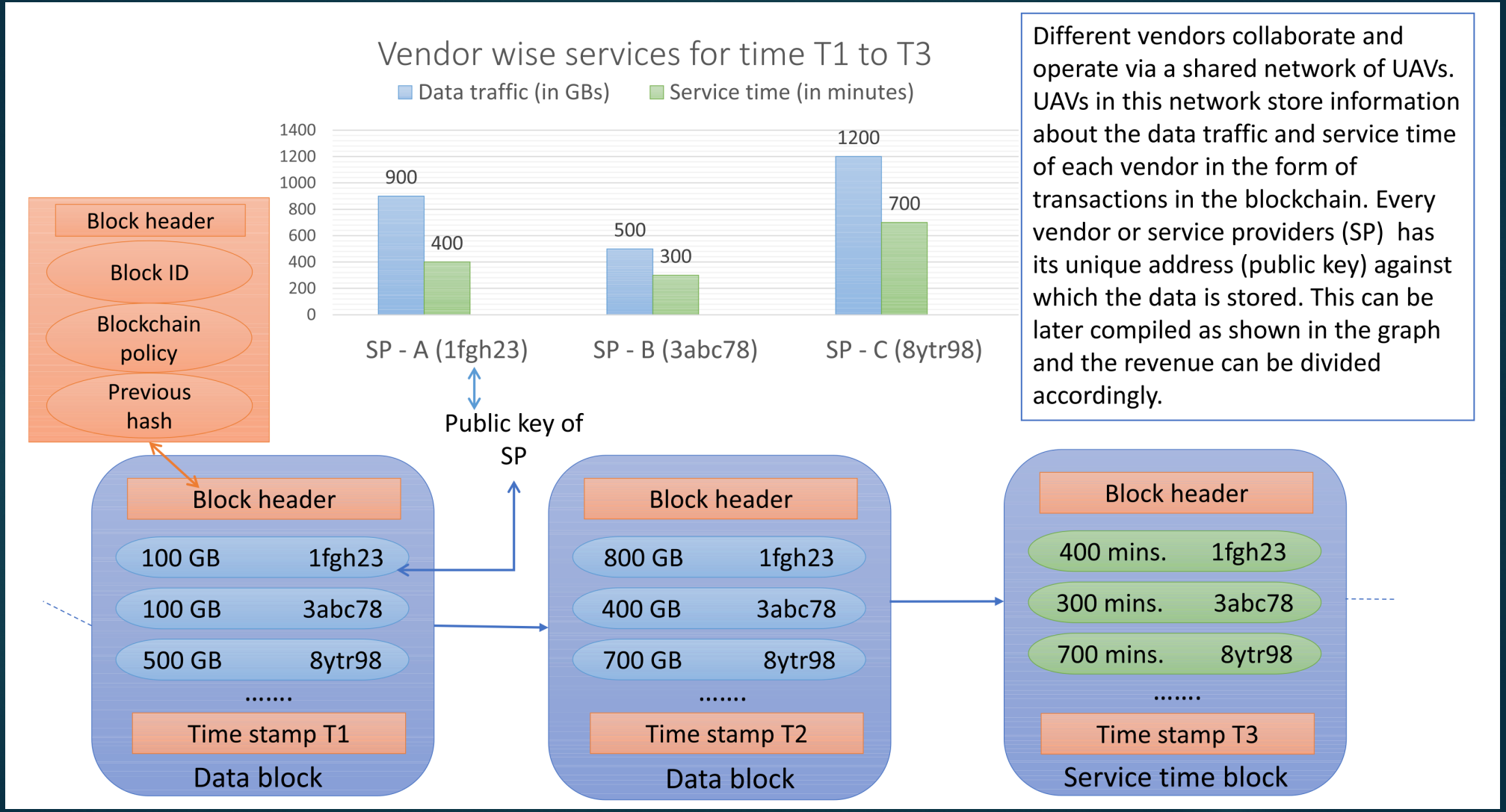
Blockchain role:

- Secure exchanges of data providers/consumers
- On-chain storage of coordinates
- Verification of load-balancing
- Reasoning based on secure past behavior





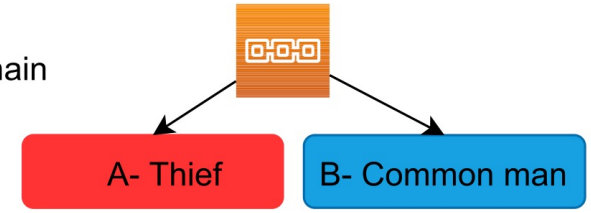
Coordinated UAVs



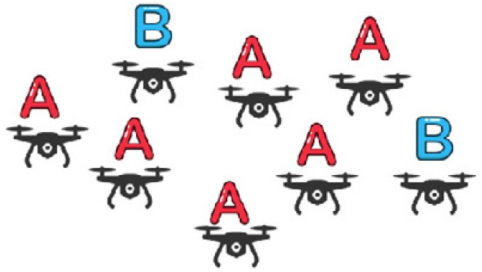
Different vendors collaborate and operate via a shared network of UAVs. UAVs in this network store information about the data traffic and service time of each vendor in the form of transactions in the blockchain. Every vendor or service providers (SP) has its unique address (public key) against which the data is stored. This can be later compiled as shown in the graph and the revenue can be divided accordingly.

Coordinated UAVs

1. UAV is confused whether the image is of a thief or common man

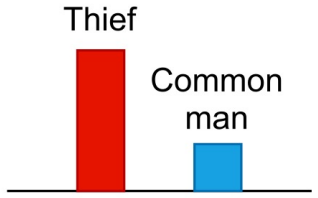


2. So it creates two addresses in the blockchain network for the two choices by making a transaction of amount zero to them:
   A- Thief
   B- Common man



3. Other UAVs in the network vote in one of these two options based on their past experience. Voting is done in form of transactions of amount 1 or 0 to addresses of A or B.

4. The option which receives the maximum votes is accepted as correct by the UAV.



Decentralized Storage

Challenge

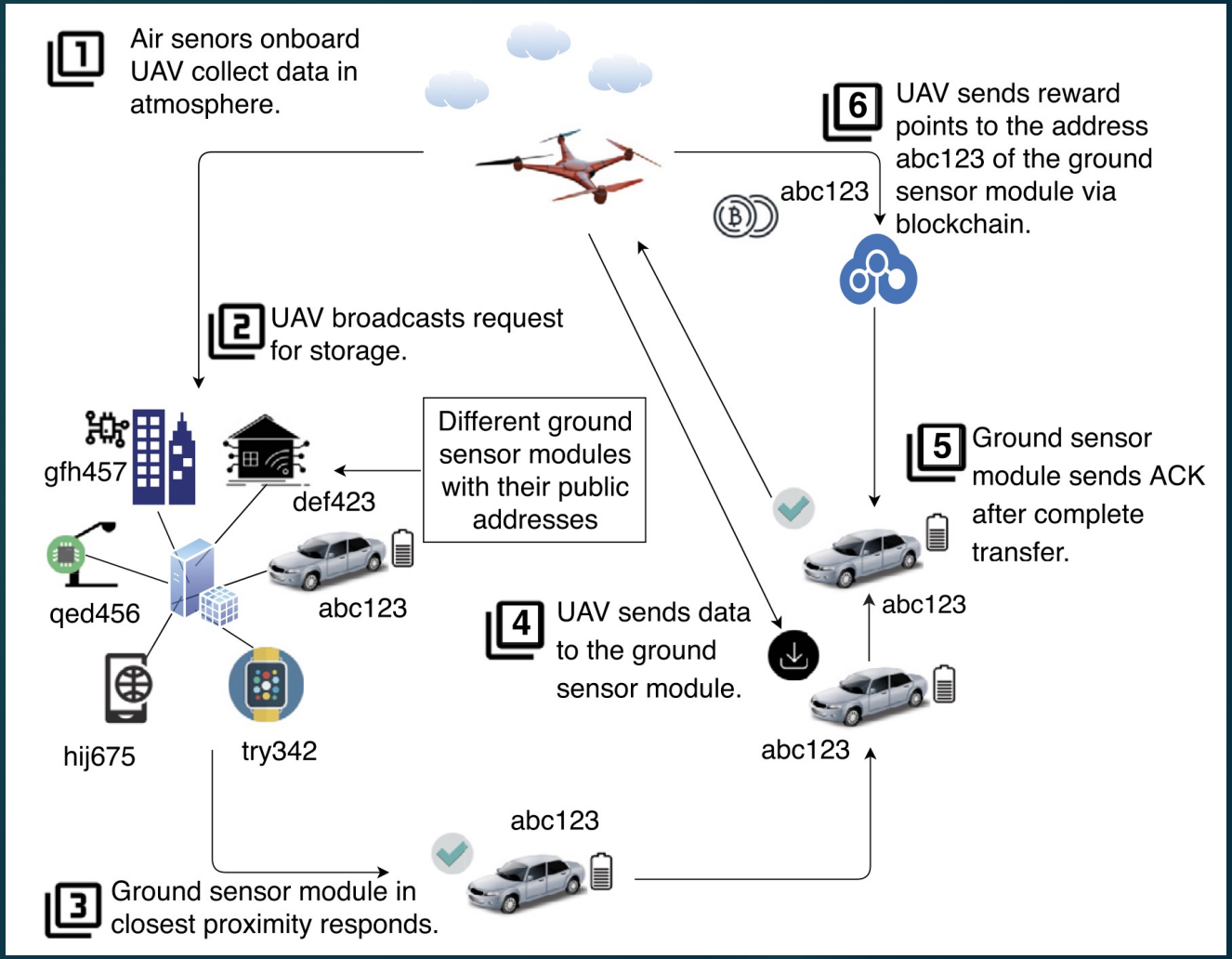
- UAVs are unreliable, inefficient & limited for (long-term) storage

Solution:

- Secure distributed storage at ground stations

Blockchain role:

- Higher drone security
- Incentive system for transactions between drones & ground stations



Security

Challenge

- How to preserve the autonomy of UAVs, while securing high-density operations & interactions between other UAVs & air traffic control

Solution:

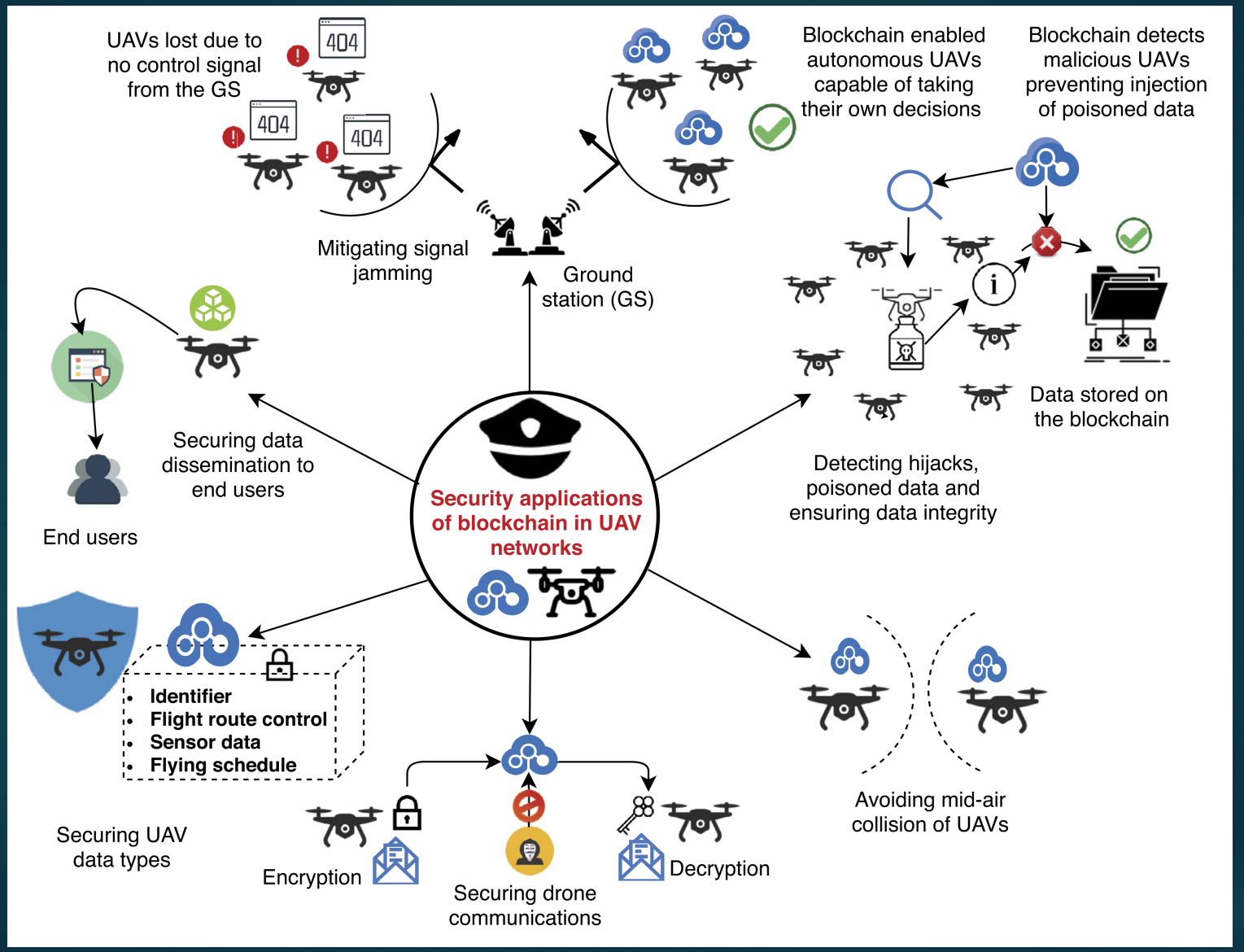
- Decentralized UAV operations

Blockchain role:

- No need for single trustworthy ground station
- Consensus for UAVs fault-tolerance
- Smart contracts for flying schedule & flight route control

- Jamming of UAV signals
- Detecting UAV hijacks, poisoned data & ensuring data integrity
- Avoiding mid-air collisions
- Securing UAVs communication
- Securing UAVs data types
- Securing data dissemination

Security



Edge Computing

Challenge

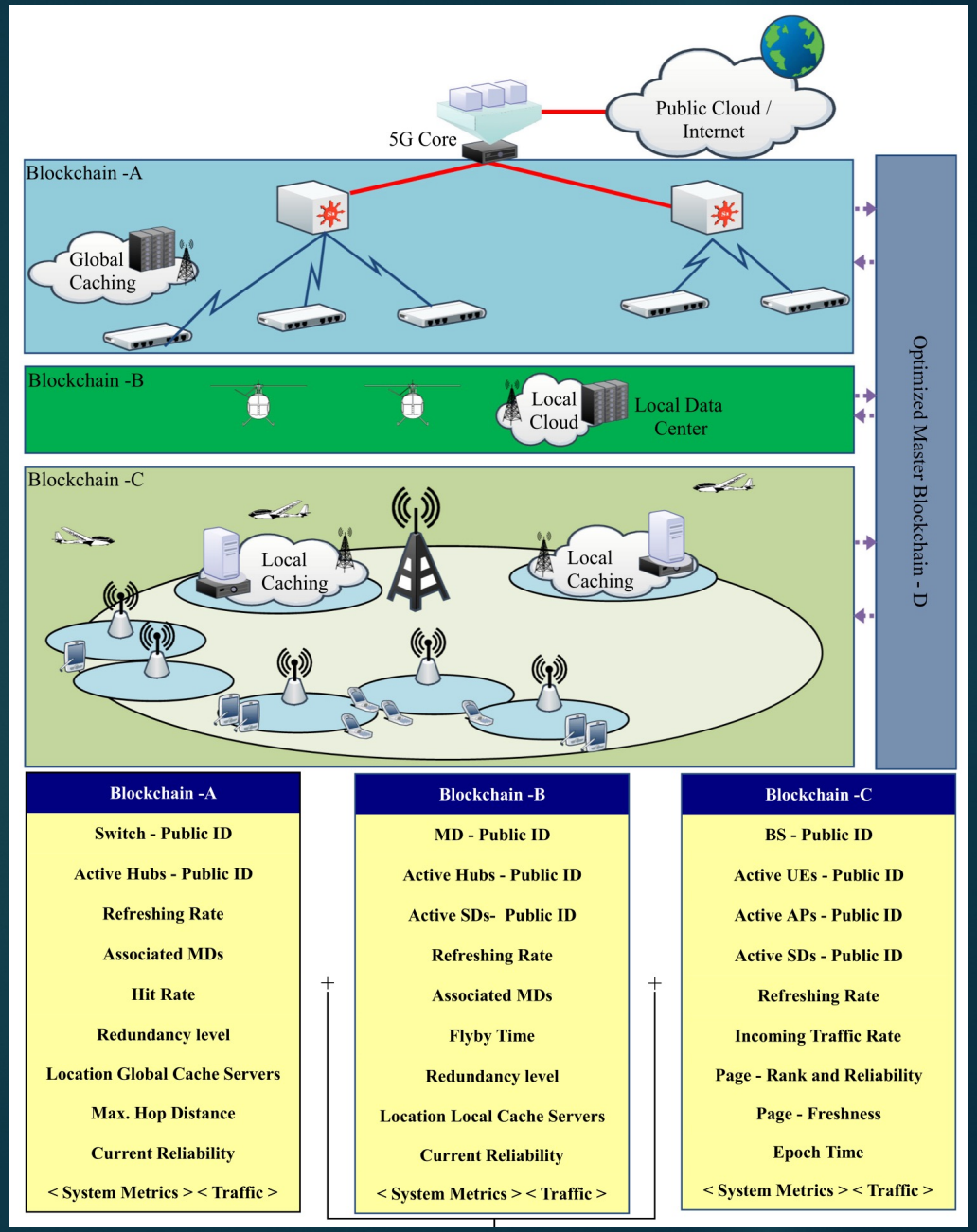
- Low latency, privacy-preserving data processing

Solution:

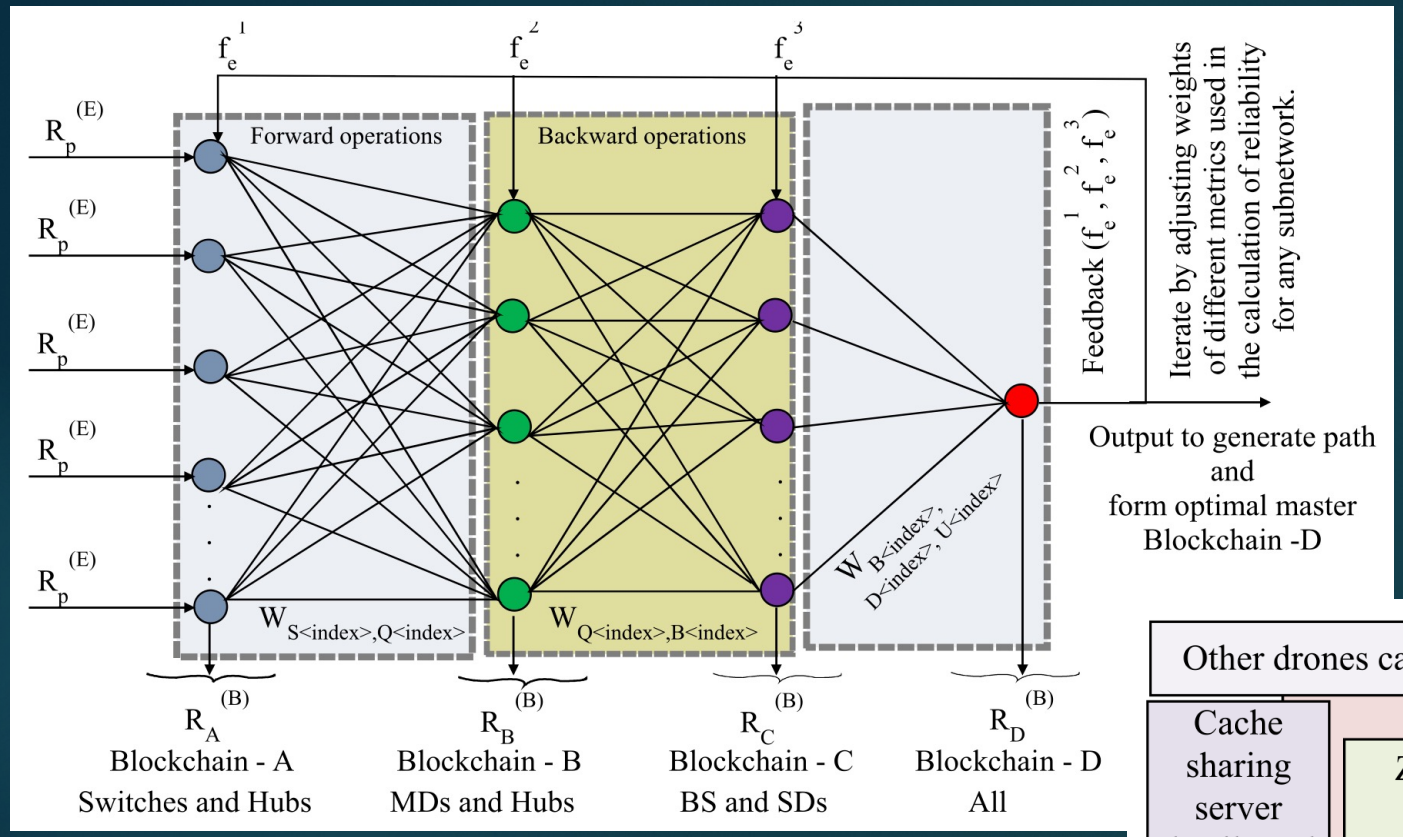
- Ultra-reliable caching system on the edge for optimal UAV path finding

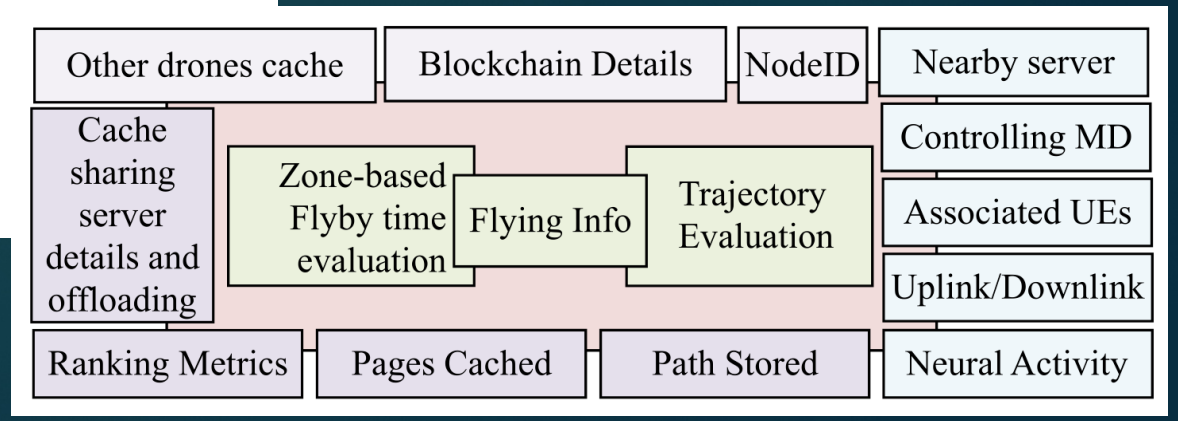
Blockchain role:

- Highly reliable communication



Edge Computing



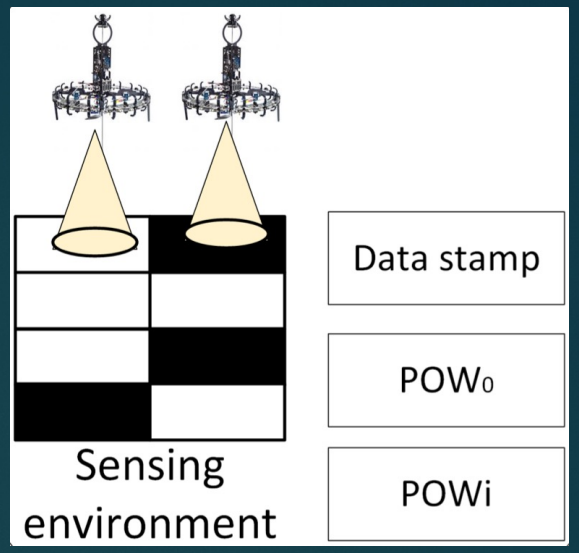


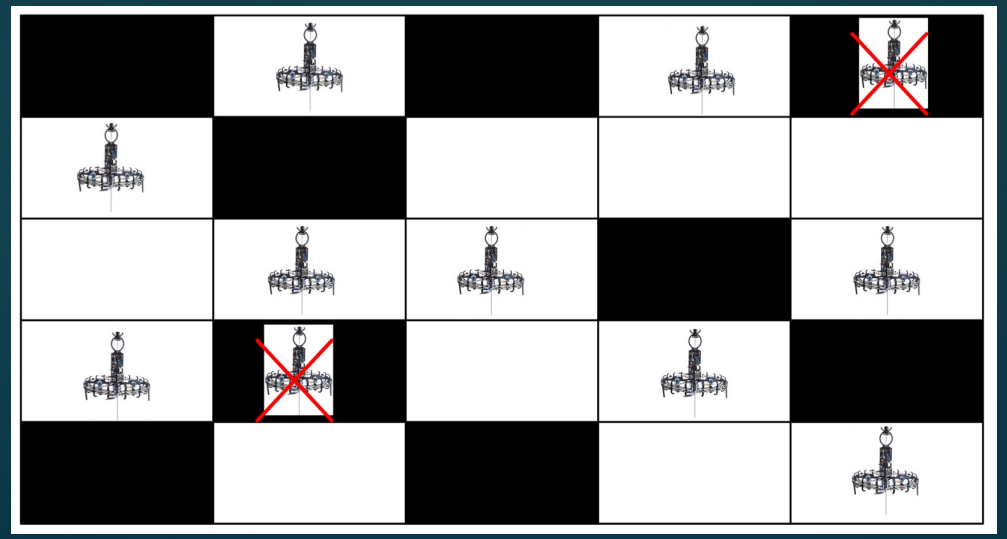
UAVs Consensus Model

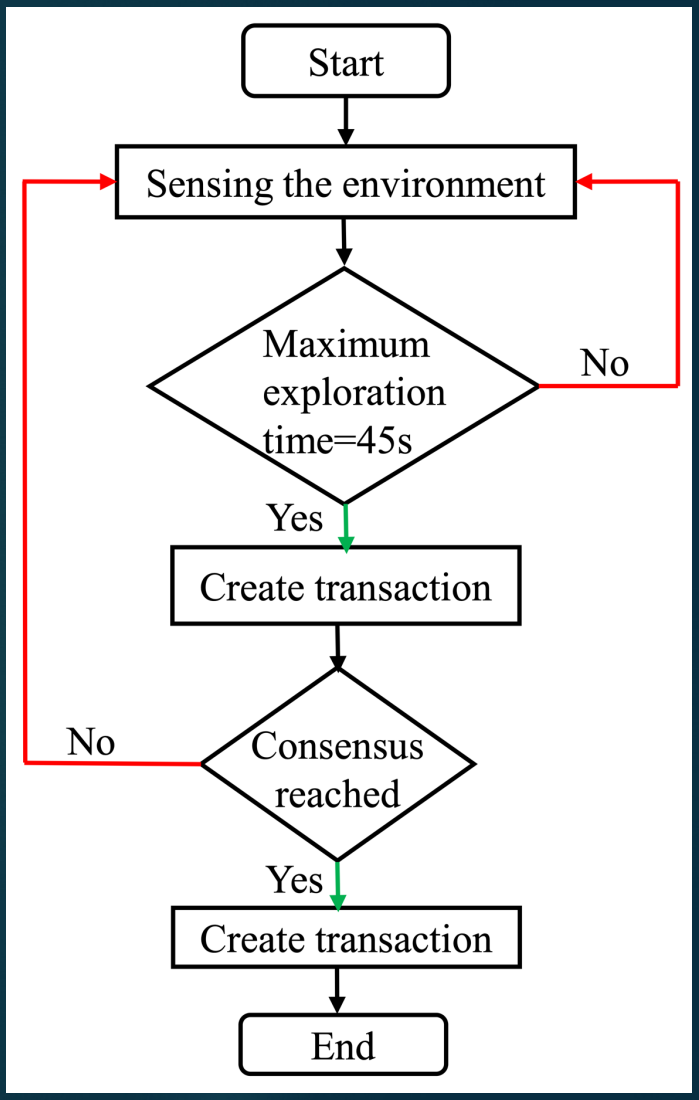
Sensing operations: agree on the frequencies of white/black cells

Proof of work

Rewards to drones that comply to the protocol







Energy-aware Swarm Consensus

Number of drones: n

Total execution time: T

Block generation rate: B

Avg. block generation time: $1/B$

Hashing power of each drone: H

Fraction of hashing power of drone i in the swarm: S_i, such that $\sum_{i=1}^{n} S_i = 1$

Distribution of hashing power over the drones: equal, $n \cdot S_i = 1$

Energy allocation for hashing: 20%

Battery capacity: C

Flying distance of a drone i: D_i

Power consumption of each drone: P

Ground speed of each drone: V

Energy-aware Swarm Consensus

Energy consumption of drone i for consensus: $E_i = {}^1/_B \cdot (S_i \cdot H)$

Total energy consumption of drone i for consensus: $E_T = {}^1/_B \cdot (S_i \cdot H) \cdot \mathrm{T} \leq 0.2 \cdot C$

$$S_i \leq \frac{0.2 \cdot (P + H)}{N \cdot H}$$

Avg. energy consumption of consensus: $E = {}^1/_B \cdot \left(\sum_{i=1}^{n} S_i \cdot H\right) = {}^1/_B \cdot H$

# of times consensus reached (throughput): $\frac{\sum_{i=1}^{n} D_i}{V} = \frac{\sum_{i=1}^{n} S_i}{B} \cdot T = \frac{T}{B}$

Total energy consumption of drone i: $\frac{D_i}{V} \cdot P + \frac{1}{B} \cdot (S_i \cdot H) \cdot T \leq C$

Total energy consumption of all drones: $\frac{\sum_{i=1}^{n} D_i}{V} \cdot P + \frac{T}{B} \cdot H \leq n \cdot C$

$$T \leq \frac{N \cdot C \cdot B}{P + H}$$

Questions?

Blockchain & Cryptoeconomics

DR. EVANGELOS POURNARAS

6. Cryptography

Cryptography

Until middle of 20th century all cryptography relied on **symmetric key paradigm**

- *Shared secret key to encrypt-decrypt a message*

1970s: a remarkable breakthrough: **asymmetric key cryptography**

- *A sender encrypts a message with receiver's public key*
- *A receiver decrypts a message with own secret key*

Cryptography & blockchain: assets ownership linked to particular public address

- *Irrefutable proof that signer owns the secret key associated to a public key*
- *Allows transfers of value from one legitimate address owner to another*

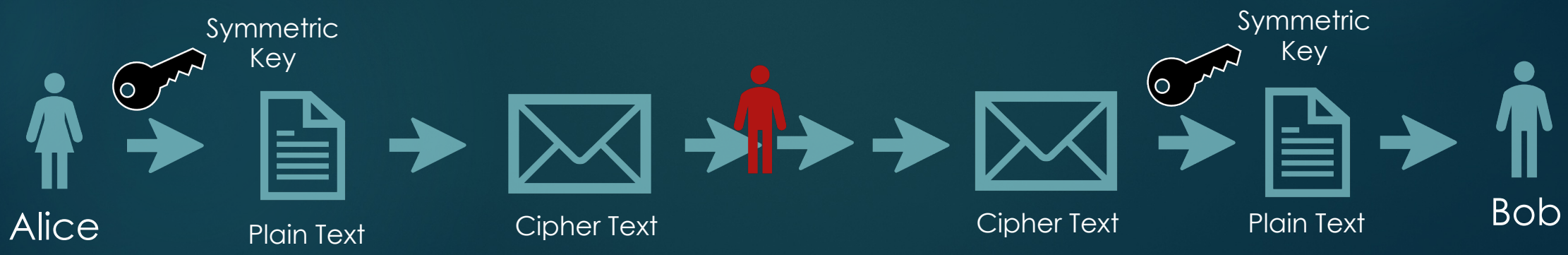
Cryptography is extraordinary **complex & simple**

- *Brute force attacks becomes computationally infeasible*
- **Probabilistic in nature**: *if p is huge prime number, x is secure, but if p is huge, checking if it is prime is hard, i.e. p is probabilistically prime*

Symetric Key Cryptography

A shared secret key used **for both encryption & decryption**

Kerckhoffs principe (Shannon's maxim): *a cryptographic scheme is secure when it cannot be broken even if everything about it is publicly known except the key*



Caesar's Cipher

Plaintext is replaced by the letter located k positions further to the right in the alphabet (cyclically closed)

Letters-to-numbers: $A \to 0, B \to 1, \dots, Z \to 25$

Key: $A \to F, k = 5$, *25 keys*

i-th letter of the message: $m_i, 0 \leq m_i \leq 25$

i-th letter of the cryptogram: $c_i = (m_i + k)\ mod\ 26$

i-th letter in the decryption: $m_i = (c_i - k + 26)\ mod\ 26$

ABCDEFGHIJKLMNOPQRSTUVWXYZ

FGHIJKLMNOPQRSTUVWXYZABCDE

Example:

- **Encryption**: $CAT = 02\ 00\ 19 \to 07\ 05\ 24 = HFY$
- **Decryption**: $\mathrm{HFY} = 07\ 05\ 24 \to CAT = 02\ 00\ 19$

Monoalphabetic: one-to-one letter mapping

Breaking the cipher: frequency analysis, guessing content

Affine Cipher

Pair of keys: $(l, k), 0 < l \leq 12, 0 \leq k < 26$

Expands the number of keys to $l \cdot k = 12 \cdot 26 = 312$

i-th letter of the cryptogram: $c_i = (l \cdot m_i + k)\ mod\ 26$

Vigenère Cipher

Stream of key: $A \to K, A \to E, A \to Y, KEY \to 10, 04, 24$

Caesar's encryption for one after another letter & key

Key is repeated if shorter than the message

i-th letter of the cryptogram: $c_i = (m_i + k_i)\ mod\ 26$

Example:

- **Encryption**: $CAT = 02\ 00\ 19 \to ?$

Polyalphabetic substitution

k_1:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

KLMNOPQRSTUVWXYZABCDEFGHIJ

k_2:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

EFGHIJKLMNOPQRSTUVWXYZABCD

k_3:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

YZABCDEFGHIJKLMNOPQRSTUVWX

Vigenère Cipher

Stream of key: $A \to K, A \to E, A \to Y, KEY \to 10, 04, 24$

Caesar's encryption for one after another letter & key

Key is repeated if shorter than the message

i-th letter of the cryptogram: $c_i = (m_i + k_i)\ mod\ 26$

Example:

- **Encryption**: $CAT = 02\ 00\ 19 \to 12\ 04\ 17 = MER$

Polyalphabetic substitution

k_1:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

KLMNOPQRSTUVWXYZABCDEFGHIJ

k_2:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

EFGHIJKLMNOPQRSTUVWXYZABCD

k_3:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

YZABCDEFGHIJKLMNOPQRSTUVWX

Other Schemes

Enigma machine

- Same principle as the Vigenère cipher with a long substitution sequence

One-time pad

- XOR bitwise arithmetic, key same size with message

Stream ciphers

- Generation of pseudo-random keystreams from a seed

Data Encryption Standard (DES)

- XOR operation & data scrambling
- Works along the Enigma machine

Advacned Encryption Standard (AES)

- Replaces DES

Asymetric Key Cryptography

Main building block: one-way function – easy to compute, hard to invert

- $y=f(x) \rightarrow$ easy
- $x=g(y) \rightarrow$ hard

Trapdoor one-way functions: hard to invert in general but easy to invert given some additional information

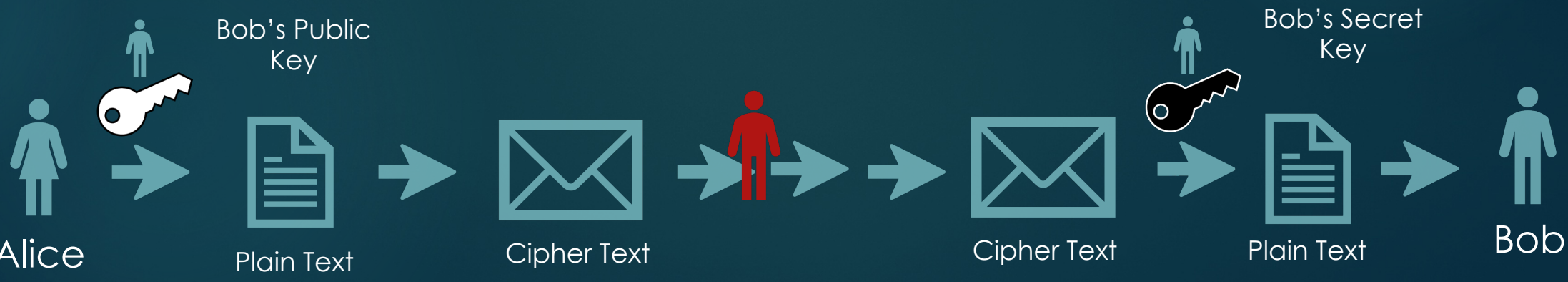
Asymetric Key Cryptography

1. Public key to encypt text

(sk,ps):=generatekeys();

cipher:=encrypt(pk,message)

message:=decrypt(sk,message)



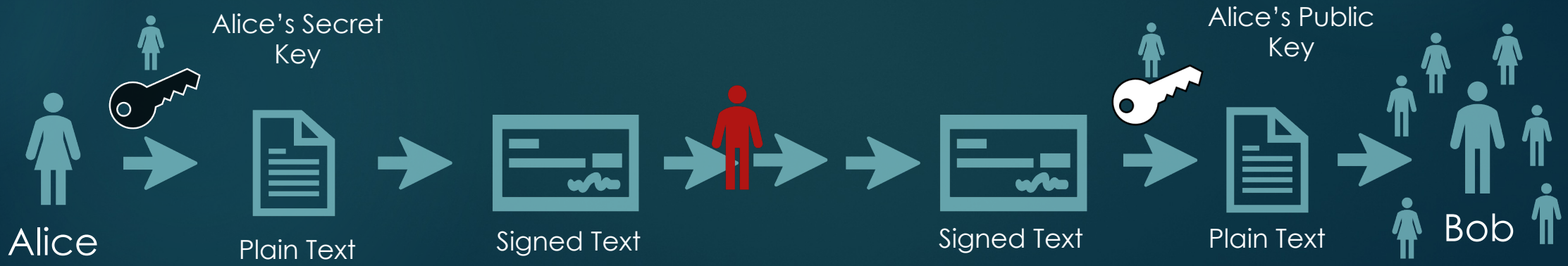
Asymetric Key Cryptography

2. Digital signature of message with secret key

```
(sk,ps):=generatekeys();
sig:=sign(sk,hash(message));
isValid:=verify(pk,hash(message),sig);
```

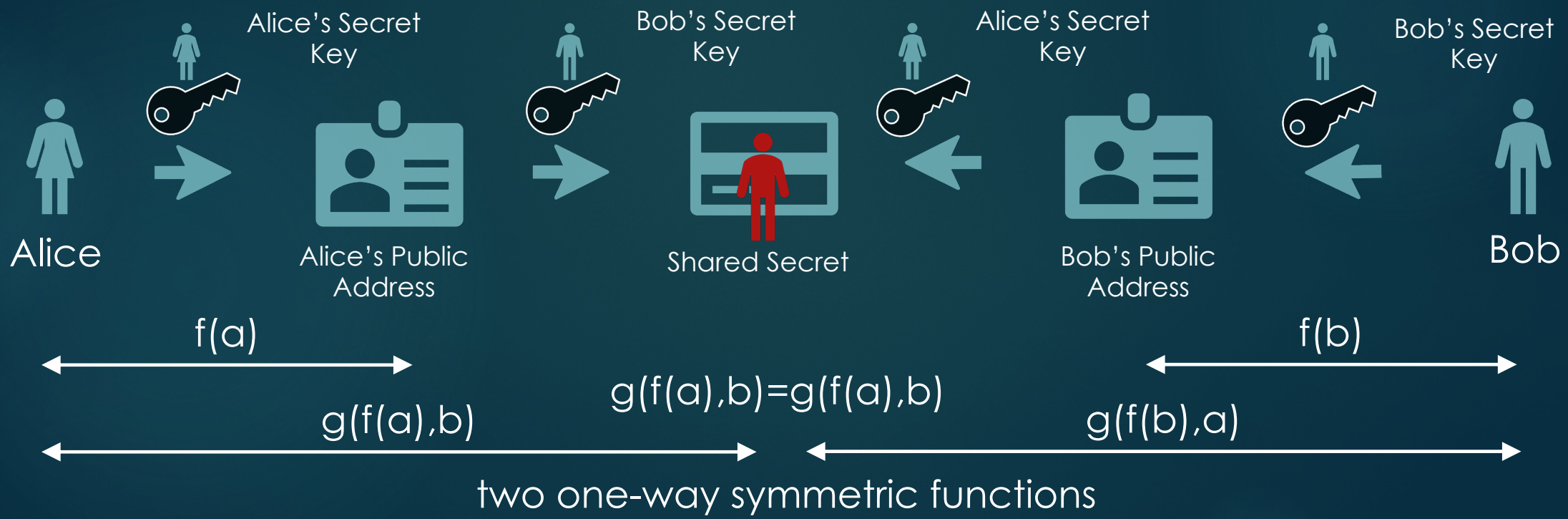
Need for hashing:

- Signature algorithms are slow (RSA, DSA, FFDSA, ECDSA)
- Input/output have the same size

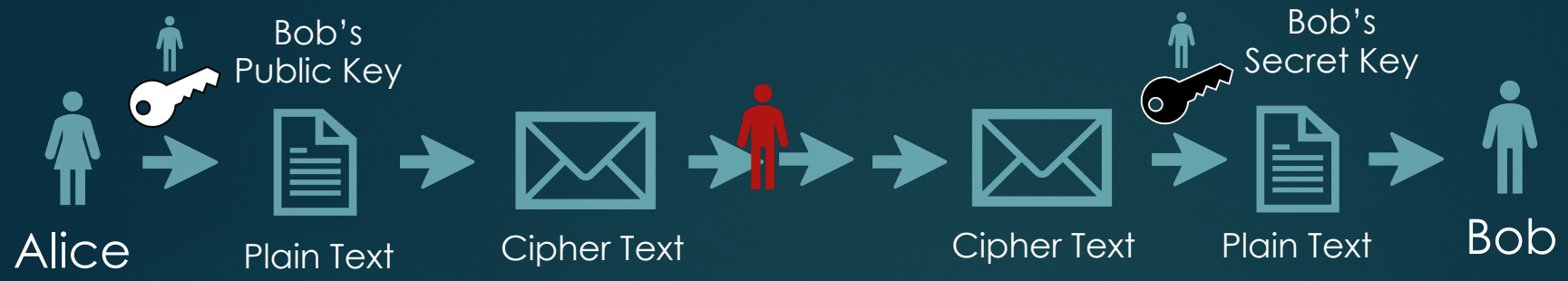


Asymetric Key Cryptography

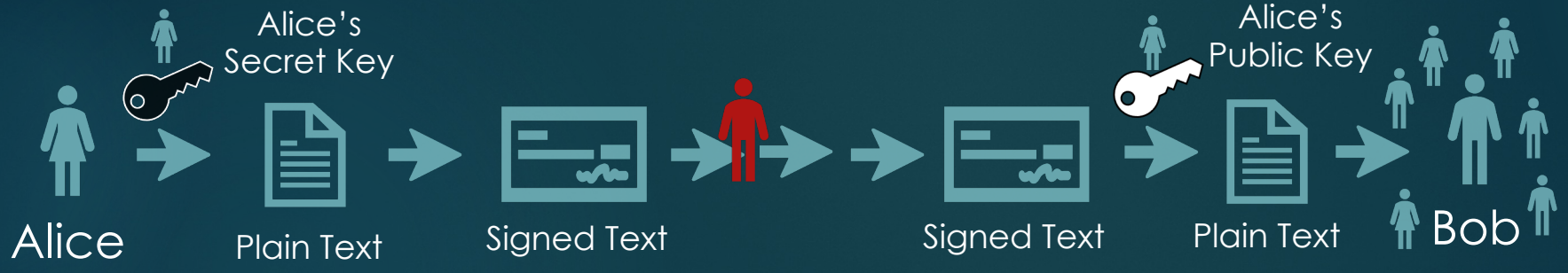
3. Common secret using both public & secret key



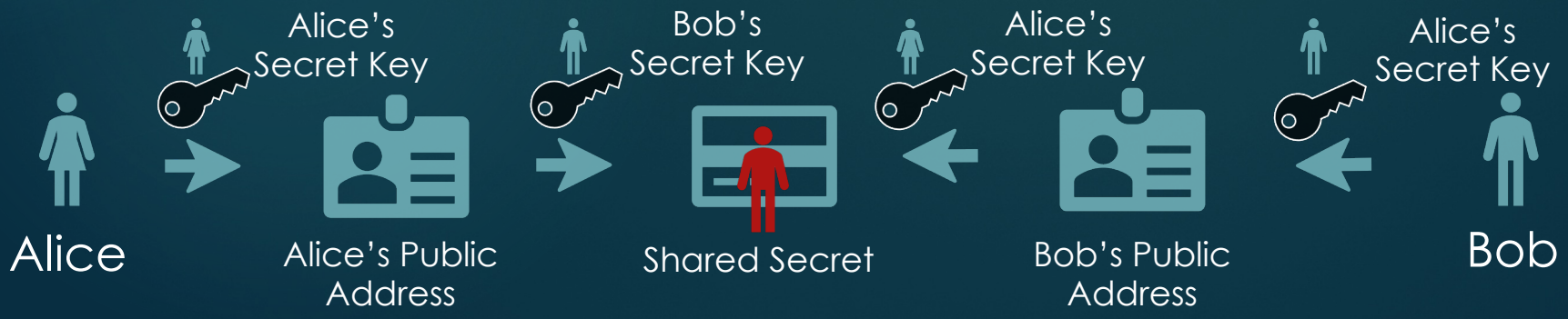
Asymetric Key Cryptography



1. Public key to encrypt text



2. Digital signature of message with secret key



3. Common secret using both public & secret key

RSA Algorithm

RSA: Ron Rivest, Adi Shamir & Leonard Adleman



First published in 1977

Relatively slow algorithm: often (& paradoxically) used to transmit shared keys for symmetric key cryptograhy for bulk encryption-decryption

RSA Algorithm

Encryption-decryption:

1. Bob chooses secret primes p & q, calculates n=p*q
2. Bob choose e with gcd(e,φ(n))=1
3. Bob calculates d so that d*e=1 mod φ(n)
4. Bob makes: public [n,e],  secret [p,q,d] 
5. Alice encrypts m as $c=m^e \bmod n$
6. Alice sends Bob c
7. Bob calculates $m=c^d \bmod n$ to recover Alice's message m

How to test that n is a prime number?

- **Fermat's little theorem**: n is composite if:
  - $A \neq 1 \ mod \ n$, otherwise likely to be prime
- Repeat for several As to increase the chances of p,q being primes

Euler totient function: φ(n)=(p-1)(q-1) or

Carmichael's totient function: λ(n)=lcm(p-1,q-1)

Calculation of d as the modular inverse of e mod φ(n). Why a d exists for d*e=1 mod φ(n)?

- Step 2: gcd(e,φ(n))=1, where e, φ(n) are integers & e is chosen to be relatively prime to φ(n)
- Existence of multiplicative inverses in modular arithmetic: Euclidean division algorithm & Bézout's identity
- Efficient calculation of d using the extended Euclidean algorithm

Convertion of message into a number – ASCII alphabet:

A	B	C	D	E	F	G	H	I	J	K	L	M
65	66	67	68	69	70	71	72	73	74	75	76	77
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
78	79	80	81	82	83	84	85	86	87	88	89	90

RSA in Action

p=601

q=7

n=?

φ(n)=?

e=1463

d=?

m=58

c=?

RSA in Action

p=601

q=7

n=601x7=4207

φ(n)=?

e=1463

d=?

m=58

c=?

RSA in Action

p=601

q=7

n=601x7=4207

φ(n)=600x6=3600

e=1463

d=?

m=58

c=?

Calculation of Secret Key

We know p & q, we need to calculate d such that d*e=1 mod φ(n)

For this, we apply the inverse of the Euclidean algorithm based on which $\gcd(e, \varphi(n)) = \gcd(1463,3600) = 1$

Euclidean algorithm:

$3600 = 2 \cdot 1463 + 674$
$1463 = 2 \cdot 674 + 115$
$674 = 5 \cdot 115 + 99$
$115 = 1 \cdot 99 + 16$
$99 = 6 \cdot 16 + 3$
$16 = 5 \cdot 3 + 1$

Calculation of Secret Key

We know p & q, we need to calculate d such that d*e=1 mod φ(n)

For this, we apply the inverse of the Euclidean algorithm based on which $\gcd(e, \varphi(n)) = \gcd(1463,3600) = 1$

Euclidean algorithm:

$3600 = 2 \cdot 1463 + 674$
$1463 = 2 \cdot 674 + 115$
$674 = 5 \cdot 115 + 99$
$115 = 1 \cdot 99 + 16$
$99 = 6 \cdot 16 + 3$
$16 = 5 \cdot 3 + 1$

Inverse steps:

$1 = 16 - 5 \cdot 3$

$1 = 16 - 5 \cdot (99 - 6 \cdot 16) = 31 \cdot 16 - 5 \cdot 99$
$1 = 31 \cdot (115 - 1 \cdot 99) - 5 \cdot 99 = 31 \cdot 115 - 36 \cdot 99$
$1 = 31 \cdot 115 - 36 \cdot (674 - 5 \cdot 115) = 211 \cdot 115 - 36 \cdot 674$
$1 = 211 \cdot (1463 - 2 \cdot 674) - 36 \cdot 674 = 211 \cdot 1463 - 458 \cdot 674$
$1 = 211 \cdot 1463 - 458 \cdot (3600 - 2 \cdot 1463) = 1127 \cdot 1463 - 458 \cdot 3600$

d=1127

$d \cdot e = 1 \, mod \, \varphi(n) = 1127 \cdot 1463 = 1 \, mod \, 3600$

RSA in Action

p=601

q=7

n=601x7=4207

φ(n)=600x6=3600

e=1463

d=1127

m=58

c=?

Encryption Step

Alice encrypts m as $c=m^e \bmod n$

m=58, e=1463, n=4207

Calculation of $m^e=58^{1463}$ is hard

Approach: modular binary exponentiation

$58^1 = 58 \rightarrow 58 \; mod \; 4207 = 58$
$58^2 = 3364 \rightarrow 3364 \; mod \; 4207 = 3364$
$58^4 = (58^2)^2 \rightarrow (3364)^2 \; mod \; 4207 = 3873$
$58^8 = (58^4)^2 \rightarrow (3873)^2 \; mod \; 4207 = 2174$
$58^{16} = (58^8)^2 \rightarrow (2174)^2 \; mod \; 4207 = 1815$
$58^{32} = (58^{16})^2 \rightarrow (1815)^2 \; mod \; 4207 = 144$
$58^{64} = (58^{32})^2 \rightarrow (144)^2 \; mod \; 4207 = 3908$
$58^{128} = (58^{64})^2 \rightarrow (3908)^2 \; mod \; 4207 = 1054$
$58^{256} = (58^{128})^2 \rightarrow (1054)^2 \; mod \; 4207 = 268$
$58^{512} = (58^{256})^2 \rightarrow (268)^2 \; mod \; 4207 = 305$
$58^{1024} = (58^{512})^2 \rightarrow (305)^2 \; mod \; 4207 = 471$
$58^{2048} = (58^{1024})^2 \rightarrow (471)^2 \; mod \; 4207 = 3077$

But we need 58^{1463}

Modular Binary Exponentation

e=1463=10110110111 [integer to binary]

Representing the exponent in a binary form 2^{k-1}:

$e=1463=1*2^{10}+0*2^{9}+1*2^{8}+1*2^{7}+0*2^{6}+1*2^{5}+1*2^{4}+0*2^{3}+1*2^{2}+1*2^{1}+1*2^{0}$

$e=1463=1*1024+0*512+1*256+1*128+0*64+1*32+1*16+0*8+1*4+1*2+1*1$

$m^e=58^{1463} = 58^{1024+256+128+32+16+4+2+1}$

$m^e=58^{1463} = 58^{1024} \cdot 58^{256} \cdot 58^{128} \cdot 58^{32} \cdot 58^{16} \cdot 58^{4} \cdot 58^{2} \cdot 58^{1}$

$c = m^e \bmod n = 58^{1463} \ mod\ 4207 = 471 \cdot 268 \cdot 1054 \cdot 144 \cdot 1815 \cdot 3873 \cdot 3364 \cdot 58 \ mod\ 4207$

$c = m^e \bmod n = 58^{1463} \ mod\ 4207 = 2937$

$58^{1} = 58 \rightarrow 58 \ mod\ 4207 = 58$
$58^{2} = 3364 \rightarrow 3364 \ mod\ 4207 = 3364$
$58^{4} = (58^{2})^{2} \rightarrow (3364)^{2} \ mod\ 4207 = 3873$
$58^{8} = (58^{4})^{2} \rightarrow (3873)^{2} \ mod\ 4207 = 2174$
$58^{16} = (58^{8})^{2} \rightarrow (2174)^{2} \ mod\ 4207 = 1815$
$58^{32} = (58^{16})^{2} \rightarrow (1815)^{2} \ mod\ 4207 = 144$
$58^{64} = (58^{32})^{2} \rightarrow (144)^{2} \ mod\ 4207 = 3908$
$58^{128} = (58^{64})^{2} \rightarrow (3908)^{2} \ mod\ 4207 = 1054$
$58^{256} = (58^{128})^{2} \rightarrow (1054)^{2} \ mod\ 4207 = 268$
$58^{512} = (58^{256})^{2} \rightarrow (268)^{2} \ mod\ 4207 = 305$
$58^{1024} = (58^{512})^{2} \rightarrow (305)^{2} \ mod\ 4207 = 471$
$58^{2048} = (58^{1024})^{2} \rightarrow (471)^{2} \ mod\ 4207 = 3077$

RSA in Action

p=601

q=7

n=601x7=4207

φ(n)=600x6=3600

e=1463

d=1127

m=58

c=2937

RSA in Action

p=601

q=7

n=601x7=4207

φ(n)=600x6=3600

e=1463

d=1127

m=?

c=2937

Decryption Step

Bob decrypts c as $m=c^d \bmod n$

c=2937, e=1463, n=4207, d=1127

Calculation of $c^d=2937^{1127}$ is hard

Approach: modular binary exponentiation

$2937^1 = 2937 \rightarrow 2937 \; mod \; 4207 = 2937$

$2937^2 = 8{,}625{,}969 \rightarrow 8{,}625{,}969 \; mod \; 4207 = 1619$

$2937^4 = (2937^2)^2 \rightarrow (1619)^2 \; mod \; 4207 = 200$

$2937^8 = (2937^4)^2 \rightarrow (200)^2 \; mod \; 4207 = 2137$

$2937^{16} = (2937^8)^2 \rightarrow (2137)^2 \; mod \; 4207 = 2174$

$2937^{32} = (2937^{16})^2 \rightarrow (2174)^2 \; mod \; 4207 = 1815$

$2937^{64} = (2937^{32})^2 \rightarrow (1815)^2 \; mod \; 4207 = 144$

$2937^{128} = (2937^{64})^2 \rightarrow (144)^2 \; mod \; 4207 = 3908$

$2937^{256} = (2937^{128})^2 \rightarrow (3908)^2 \; mod \; 4207 = 1054$

$2937^{512} = (2937^{256})^2 \rightarrow (1054)^2 \; mod \; 4207 = 268$

$2937^{1024} = (2937^{512})^2 \rightarrow (268)^2 \; mod \; 4207 = 305$

$2937^{2048} = (2937^{1024})^2 \rightarrow (305)^2 \; mod \; 4207 = 471$

Modular Binary Exponentation

d=1127=10001100111 [integer to binary]

Representing the exponent in a binary form 2^{k-1}:

$d=1127=1*2^{10}+0*2^{9}+0*2^{8}+0*2^{7}+1*2^{6}+1*2^{5}+0*2^{4}+0*2^{3}+1*2^{2}+1*2^{1}+1*2^{0}$

$d=1127=1*1024+0*512+0*256+0*128+1*64+1*32+0*16+0*8+1*4+1*2+1*1$

$c^{d}=2937^{1127}=2937^{1024+64+32+4+2+1}$

$c^{d}=2937^{1127}=2937^{1024}\cdot 2937^{64}\cdot 2937^{32}\cdot 2937^{4}\cdot 2937^{2}\cdot 2937^{1}$

$c^{d}\bmod n=2937^{1127}\ mod\ 4207=305\cdot 144\cdot 1815\cdot 200\cdot 1619\cdot 2937\cdot mod\ 4207$

$c^{d}\bmod n=2937^{1127}\ mod\ 4207=58$

$2937^{1}=2937\rightarrow 2937\ mod\ 4207=2937$
$2937^{2}=8{,}625{,}969\rightarrow 8{,}625{,}969\ mod\ 4207=1619$
$2937^{4}=(2937^{2})^{2}\rightarrow(1619)^{2}\ mod\ 4207=200$
$2937^{8}=(2937^{4})^{2}\rightarrow(200)^{2}\ mod\ 4207=2137$
$2937^{16}=(2937^{8})^{2}\rightarrow(2137)^{2}\ mod\ 4207=2174$
$2937^{32}=(2937^{16})^{2}\rightarrow(2174)^{2}\ mod\ 4207=1815$

$2937^{64}=(2937^{32})^{2}\rightarrow(1815)^{2}\ mod\ 4207=144$
$2937^{128}=(2937^{64})^{2}\rightarrow(144)^{2}\ mod\ 4207=3908$
$2937^{256}=(2937^{128})^{2}\rightarrow(3908)^{2}\ mod\ 4207=1054$
$2937^{512}=(2937^{256})^{2}\rightarrow(1054)^{2}\ mod\ 4207=268$
$2937^{1024}=(2937^{512})^{2}\rightarrow(268)^{2}\ mod\ 4207=305$
$2937^{2048}=(2937^{1024})^{2}\rightarrow(305)^{2}\ mod\ 4207=471$

RSA Algorithm

Digital Signature:

1. Alice chooses secret primes p & q, calculates n=p*q
2. Alice choose e with gcd(e,φ(n))=1
3. Alice calculates d so that d*e=1 mod φ(n)
4. Alice makes: public [n,e], secret [p,q,d]
5. Alice signs m as $s=m^d \bmod n$
6. Alice sends Bob s & m mod n
7. Bob calculates $m'=s^e$ & compares it with m: if m'=m, Alice has properly signed m

How to test that n is a prime number?

- **Fermat's little theorem**: n is composite if:
  - $A \neq 1 \ mod \ n$, otherwise likely to be prime
- Repeat for several As to increase the chances of p,q being primes

Euler totient function: φ(n)=(p-1)(q-1) or

Carmichael's totient function: λ(n)=lcm(p-1,q-1)

Calculation of d as the modular inverse of e mod φ(n). Why a d exists for d*e=1 mod φ(n)?

- Step 2: gcd(e,φ(n))=1, where e, φ(n) are integers & e is chosen to be relatively prime to φ(n)
- Existence of multiplicative inverses in modular arithmetic: Euclidean division algorithm & Bézout's identity
- Efficient calculation of d using the extended Euclidean algorithm

Convertion of message into a number – ASCII alphabet:

A	B	C	D	E	F	G	H	I	J	K	L	M
65	66	67	68	69	70	71	72	73	74	75	76	77
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
78	79	80	81	82	83	84	85	86	87	88	89	90

RSA in Action

p=601

q=7

n=601x7=4207

φ(n)=600x6=3600

e=1463

d=1127

m=58

s=?

m'=?

m=m' ⟶ ?

Blind RSA Digital Signature

1. Alice needs to prove that she created m
2. Alice needs to keep m a secret
3. Ben: helping intermediary, e.g. notary, election official
4. Ben makes: public $[n,e]$, secret $[p,q,d]$
5. Alice chooses a random integer k
6. Alice sends to Ben the message $\mu=k^e m \bmod n$ to sign
7. Ben signs & returns the following message to Alice: $\mu'=\mu^d=k^{ed}m^d=km^d \bmod n$
8. Alice divides by k (multiplication by modular inverse) & retrieves the message $m^d \bmod n$ signed by Ben

Chaum's Anonymous Digicash

Motivation: *Alice wants to send some money to Bob without Ben (bank) knows that is her who sends money to Bob*

1. Alice sends Ben L blinded messages: $\mu_l = k_l m_l \bmod n$, $\forall\, l \in \{1, \dots, L\}$, where μ_l contains:
   - A: the amount to be paid
   - v_l: a very large random number
2. Ben asks Alice to unblind randomly L-1 messages, while the message μ_λ remains blinded
3. Ben signs & sends to alice the message: $\mu_\lambda' = \mu_\lambda^d = k_\lambda m_\lambda^d \bmod n$
4. Alice extracts & sends to Bob the signed message $m_\lambda^d \bmod n$ & unsigned message m_λ
5. Bob checks the signature & sends the message & signature to Ben
6. Ben checks the signature & checks if v_λ is in the DB of spent coins:
   - Yes?: Double-spending, no transaction
   - No?: Ben transfers funds to Bob

ElGamal Algorithm

Introduced in 1985

Taher Elgamal: Egyptian cryptographer (father of SSL)

Based on **discrete logarithms**

More secure than vanilla RSA

- Uses a random (non-guessable, used once) r, helps with short messages
- Makes use of primitive roots

ElGamal cryptosystem turns Diffie-Hellman key exchange method into an encryption algorithm

ElGamal Algorithm

Encryption-decryption:

1. Alice chooses public prime p & g as a primitive root mod p

   Provided any h, $0<h<p$, there is an integer x such that:

   $h = g^x \bmod p$

2. Alice chooses a secret random integer x & calculates $h=g^x \bmod p$
3. Alice makes: public $[p,g,h]$, secret $[x]$
4. Bob chooses a random integer r, $0 \leq r < p-1$, encrypts the message m as a pair (c_1,c_2):
   - $c_1=g^r \bmod p$,
   - $c_2=(h^r m) \bmod p$

   & sends it to Alice

   Convertion of message into a number – ASCII alphabet:

A	B	C	D	E	F	G	H	I	J	K	L	M
65	66	67	68	69	70	71	72	73	74	75	76	77
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
78	79	80	81	82	83	84	85	86	87	88	89	90

5. Alice decrypts (c_1,c_2) as follows:
   - $s=c_1^x \bmod p$
   - $m= c_2\, s^{-1} \bmod p = c_2\, s^{p-2} \bmod p$

   s is an element in G: *a multiplicative cyclic group defined by a prime*

   - $s^{-1}=s^{p-2} \bmod p$, inverse of s equals $s^{\varphi(p)-1} \bmod p$, for a prime p then: $s^{p-2} \bmod p$
   - **Proof**: Modular multiplicative inverse & Euler's theorem

ElGamal in Action

p=997

g=7

x=146

h=?

r=59

m=52

c_1=?

c_2=?

s=?

m=?

ElGamal in Action

p=997

g=7

x=146

h=7^{146} mod 997 = 634

r=59

m=52

c_1=?

c_2=?

s=?

m=?

x=146=10010010 [integer to binary]

Representing the exponent in a binary form 2^{k-1}:

x=146=1*2^7+0*2^6+0*2^5+1*2^4+0*2^3+0*2^2+1*2^1+0*2^0

x=146=1*128+0*64+0*32+1*16+0*8+0*4+1*2+0*1

$g^x=7^{146} = 7^{128+16+2}$

$g^x=7^{146} = 7^{128} \cdot 7^{16} \cdot 7^2$

$g^x \bmod p = 7^{146} \ mod\ 997 = 762 \cdot 672 \cdot 49 \ mod\ 997 = 634$

Approach: modular binary exponentiation

$7^1 = 7 \rightarrow 7 \ mod\ 997 = 7$

$7^2 = 49 \rightarrow 49 \ mod\ 997 = 49$

$7^4 = (7^2)^2 \rightarrow (49)^2 mod\ 997 = 407$

$7^8 = (7^4)^2 \rightarrow (407)^2 mod\ 997 = 147$

$7^{16} = (7^8)^2 \rightarrow (147)^2 mod\ 997 = 672$

$7^{32} = (7^{16})^2 \rightarrow (672)^2 mod\ 997 = 940$

$7^{64} = (7^{32})^2 \rightarrow (940)^2 mod\ 997 = 258$

$7^{128} = (7^{64})^2 \rightarrow (258)^2 mod\ 997 = 762$

$7^{256} = (7^{128})^2 \rightarrow (762)^2 mod\ 997 = 390$

Encryption Step: c_1

Bob encrypts m as $c_1 = g^r m \bmod p$

m=52, g=7, r=59, p=997

Calculation of $g^r = 7^{59}$ is hard

Approach: modular binary exponentiation

$7^1 = 7 \rightarrow 7 \; mod \; 997 = 7$
$7^2 = 49 \rightarrow 49 \; mod \; 997 = 49$
$7^4 = (7^2)^2 \rightarrow (49)^2 mod \; 997 = 407$
$7^8 = (7^4)^2 \rightarrow (407)^2 mod \; 997 = 147$
$7^{16} = (7^8)^2 \rightarrow (147)^2 mod \; 997 = 672$
$7^{32} = (7^{16})^2 \rightarrow (672)^2 mod \; 997 = 940$
$7^{64} = (7^{32})^2 \rightarrow (940)^2 mod \; 997 = 258$

But we need 7^{59}

Modular Binary Exponentation

r=59=111011 [integer to binary]

Representing the exponent in a binary form 2^{k-1}:

$r=59=1*2^5+1*2^4+1*2^3+0*2^2+1*2^1+1*2^0$

$r=59=1*32+1*16+1*8+0*4+1*2+1*1$

$g^r=7^{59} = 7^{32+16+8+2+1}$

$g^r=7^{59} = 7^{32} \cdot 7^{16} \cdot 7^8 \cdot 7^2 \cdot 7^1$

$c_1=g^r \bmod p = 7^{59} \ mod \ 997 = 940 \cdot 672 \cdot 147 \cdot 49 \cdot 7 \ mod \ 997 = 602$

$c_1=gr \bmod p = 7^{59} \ mod \ 997 = 602$

$7^1 = 7 \rightarrow 7 \ mod \ 997 = 7$
$7^2 = 49 \rightarrow 49 \ mod \ 997 = 49$
$7^4 = (7^2)^2 \rightarrow (49)^2 mod \ 997 = 407$
$7^8 = (7^4)^2 \rightarrow (407)^2 mod \ 997 = 147$
$7^{16} = (7^8)^2 \rightarrow (147)^2 mod \ 997 = 672$
$7^{32} = (7^{16})^2 \rightarrow (672)^2 mod \ 997 = 940$
$7^{64} = (7^{32})^2 \rightarrow (940)^2 mod \ 997 = 258$

ElGamal in Action

p=997

g=7

x=146

$h=7^{146} \bmod 997 = 634$

r=59

m=52

$c_1=602$

$c_2=?$

s=?

m=?

Encryption Step: c_2

Bob encrypts m as $c_2=(h^r m) \bmod p$

m=52, h=634, r=59, p=997

Calculation of $h^r=634^{59}$ is computationally infeasible

Approach: modular binary exponentiation

$$634^1 = 634 \rightarrow 634 \; mod \; 997 = 634$$
$$634^2 = 401956 \rightarrow 401956 \; mod \; 997 = 165$$
$$634^4 = (634^2)^2 \rightarrow (165)^2 mod \; 997 = 306$$
$$634^8 = (634^4)^2 \rightarrow (306)^2 mod \; 997 = 915$$
$$634^{16} = (634^8)^2 \rightarrow (915)^2 mod \; 997 = 742$$
$$634^{32} = (634^{16})^2 \rightarrow (742)^2 mod \; 997 = 220$$
$$634^{64} = (634^{32})^2 \rightarrow (220)^2 mod \; 997 = 544$$

But we need 634^{59}

Modular Binary Exponentation

r=59=111011 [integer to binary]

Representing the exponent in a binary form 2^{k-1}:

$r=59=1*2^5+1*2^4+1*2^3+0*2^2+1*2^1+1*2^0$

$r=59=1*32+1*16+1*8+0*4+1*2+1*1$

$h^r=634^{59} = 634^{32+16+8+2+1}$

$h^r=634^{59} = 634^{32} \cdot 634^{16} \cdot 634^{8} \cdot 634^{2} \cdot 634^{1}$

$c_2=(h^r m) \bmod p = 634^{59} 52 \; mod \; 997 = 220 \cdot 742 \cdot 915 \cdot 165 \cdot 634 \cdot 52 \; mod \; 997 = 705$

$c_2=(h^r m) \bmod p = 634^{59} 52 \; mod \; 997 = 705$

$634^1 = 634 \rightarrow 634 \; mod \; 997 = 634$
$634^2 = 401956 \rightarrow 401956 \; mod \; 997 = 165$
$634^4 = (634^2)^2 \rightarrow (165)^2 mod \; 997 = 306$
$634^8 = (634^4)^2 \rightarrow (306)^2 mod \; 997 = 915$
$634^{16} = (634^8)^2 \rightarrow (915)^2 mod \; 997 = 742$
$634^{32} = (634^{16})^2 \rightarrow (742)^2 mod \; 997 = 220$
$634^{64} = (634^{32})^2 \rightarrow (220)^2 mod \; 997 = 544$

ElGamal in Action

p=997

g=7

x=146

$h=7^{146} \bmod 997 = 634$

r=59

m=52

$c_1=602$

$c_2=705$

s=?

m=?

Decryption Step: s

Alice decrypts c_1 as $s = c_1^x \bmod p$

c_1=602, x=146, p=997

Calculation of $c_1^x = 602^{146}$ is hard

Approach: modular binary exponentiation

$602^1 = 602 \rightarrow 602\ mod\ 997 = 602$

$602^2 = 362404 \rightarrow 362404\ mod\ 997 = 493$
$602^4 = (602^2)^2 \rightarrow (493)^2 mod\ 997 = 778$
$602^8 = (602^4)^2 \rightarrow (778)^2 mod\ 997 = 105$
$602^{16} = (602^8)^2 \rightarrow (105)^2 mod\ 997 = 58$
$602^{32} = (602^{16})^2 \rightarrow (58)^2 mod\ 997 = 373$
$602^{64} = (602^{32})^2 \rightarrow (373)^2 mod\ 997 = 546$
$602^{128} = (602^{64})^2 \rightarrow (546)^2 mod\ 997 = 13$
$602^{256} = (602^{128})^2 \rightarrow (13)^2 mod\ 997 = 169$

But we need 602^{146}

Modular Binary Exponentation

x=146=10010010 [integer to binary]

Representing the exponent in a binary form 2^{k-1}:

$x=146=1*2^7+0*2^6+0*2^5+1*2^4+0*2^3+0*2^2+1*2^1+0*2^0$

$x=146=1*128+0*64+0*32+1*16+0*8+0*4+1*2+0*1$

$c_1{}^x=602^{146} = 602^{128+16+2}$

$c_1{}^x = 602^{146} = 602^{128} \cdot 602^{16} \cdot 602^2$

$c_1{}^x \bmod p = 602^{146} \; mod \; 997 = 13 \cdot 58 \cdot 493 \; mod \; 997 = 838$

$602^1 = 602 \rightarrow 602 \; mod \; 997 = 602$

$602^2 = 362404 \rightarrow 362404 \; mod \; 997 = 493$
$602^4 = (602^2)^2 \rightarrow (493)^2 mod \; 997 = 778$
$602^8 = (602^4)^2 \rightarrow (778)^2 mod \; 997 = 105$
$602^{16} = (602^8)^2 \rightarrow (105)^2 mod \; 997 = 58$
$602^{32} = (602^{16})^2 \rightarrow (58)^2 mod \; 997 = 373$
$602^{64} = (602^{32})^2 \rightarrow (373)^2 mod \; 997 = 546$
$602^{128} = (602^{64})^2 \rightarrow (546)^2 mod \; 997 = 13$
$602^{256} = (602^{128})^2 \rightarrow (13)^2 mod \; 997 = 169$

ElGamal in Action

p=997

g=7

x=146

$h=7^{146} \bmod 997 = 634$

r=59

m=52

$c_1=602$

$c_2=705$

s=838

m=?

Decryption Step: m

Alice decrypts m as $m = c_2 \; s^{p-2} \bmod p$

c_2=705, s=838, p=997, calculation of $s^{p-2} = 838^{995}$ is hard

Approach: modular binary exponentiation

$838^1 = 838 \rightarrow 838 \; mod \; 997 = 838$

$838^2 = 702244 \rightarrow 702244 \; mod \; 997 = 356$

$838^4 = (838^2)^2 \rightarrow (356)^2 mod \; 997 = 117$

$838^8 = (838^4)^2 \rightarrow (117)^2 mod \; 997 = 728$

$838^{16} = (838^8)^2 \rightarrow (728)^2 mod \; 997 = 577$

$838^{32} = (838^{16})^2 \rightarrow (577)^2 mod \; 997 = 928$

$838^{64} = (838^{32})^2 \rightarrow (928)^2 mod \; 997 = 773$

$838^{128} = (838^{64})^2 \rightarrow (773)^2 mod \; 997 = 326$

$838^{256} = (838^{128})^2 \rightarrow (326)^2 mod \; 997 = 594$

$838^{512} = (838^{256})^2 \rightarrow (594)^2 mod \; 997 = 895$

$838^{1024} = (838^{512})^2 \rightarrow (895)^2 mod \; 997 = 434$

But we need 838^{995}

Modular Binary Exponentation

p-2=995=1111100011 [integer to binary]

Representing the exponent in a binary form 2^{k-1}:

p-2=995=$1*2^9+1*2^8+1*2^7+1*2^6+1*2^5+0*2^4+0*2^3+0*2^2+1*2^1+1*2^0$

p-2=995=1*512+1*256+1*128+1*64+1*32+0*16+0*8+0*4+1*2+1*1

$s^{p-2}=838^{995} = 838^{512+256+128+64+32+2+1}$

$s^{p-2}= 838^{995} = 838^{512} \cdot 838^{256} \cdot 838^{128} \cdot 838^{64} \cdot 838^{32} \cdot 838^{2} \cdot 838^{1} \cdot$

$c_2 \cdot s^{p-2}$ mod p $=705 \cdot 838^{995} \; mod \; 997 = 705 \cdot 895 \cdot 594 \cdot 326 \cdot 773 \cdot 928 \cdot 356 \cdot 838 \; mod \; 997 = 52$

$838^1 = 838 \rightarrow 838 \; mod \; 997 = 838$

$838^2 = 702244 \rightarrow 702244 \; mod \; 997 = 356$

$838^4 = (838^2)^2 \rightarrow (356)^2 mod \; 997 = 117$

$838^8 = (838^4)^2 \rightarrow (117)^2 mod \; 997 = 728$

$838^{16} = (838^8)^2 \rightarrow (728)^2 mod \; 997 = 577$

$838^{32} = (838^{16})^2 \rightarrow (577)^2 mod \; 997 = 928$

$838^{64} = (838^{32})^2 \rightarrow (928)^2 mod \; 997 = 773$

$838^{128} = (838^{64})^2 \rightarrow (773)^2 mod \; 997 = 326$

$838^{256} = (838^{128})^2 \rightarrow (326)^2 mod \; 997 = 594$

$838^{512} = (838^{256})^2 \rightarrow (594)^2 mod \; 997 = 895$

$838^{1024} = (838^{512})^2 \rightarrow (895)^2 mod \; 997 = 434$

ElGamal Algorithm

Digital Signature:

1. Alice chooses public prime p & g as a primitive root mod p

   Provided any h, 0<h<p, there is an integer r such that:

   $h = g^x mod\ p$

2. Alice chooses a secret random integer x & calculates $h=g^x \bmod p$
3. Alice makes: public [p,g,h], secret [x]
4. Alice chooses a random integer r, $0 \le r < p-1$, $\gcd(r,p-1)=1$ & calculates $y=g^r \bmod p$
5. Alice creates a signature as (y,s), where $s=(m+x*y)r^{-1} \bmod (p-1)$

   1 vs. 3 exponentiations

6. Bob verifies if the following are equal:
   - $h^{-y}\, y^s \bmod p$
   - $g^m \bmod p$

Why?

$h^{-y}\, y^s = (g^x)^{-y}\, (g^r)^{(m+x*y)r\wedge -1} \bmod p$
$= g^{-xy}\, g^{m+xy} \bmod p = g^m \bmod p$

ElGamal in Action

p=997

g=7

x=146

h=?

r=59

y=?

m=52

s=?

$h^{-y} \bmod p = ?$

$y^{s} \bmod p = ?$

$h^{-y}\, y^{s} \bmod p = ?$

$g^{m} \bmod p = ?$

ElGamal in Action

p=997

g=7

x=146

$h=7^{146} \bmod 997 = 634$

r=59

y=?

m=52

s=?

$h^{-y} \bmod p = ?$

$y^{s} \bmod p = ?$

$h^{-y}\, y^{s} \bmod p = ?$

$g^{m} \bmod p = ?$

ElGamal in Action

p=997

g=7

x=146

$h=7^{146} \bmod 997 = 634$

r=59

$y=7^{59} \bmod 997 = 602$

m=52

s=?

$h^{-y} \bmod p = ?$

$y^{s} \bmod p = ?$

$h^{-y}\, y^{s} \bmod p = ?$

$g^{m} \bmod p = ?$

ElGamal in Action

p=997

g=7

x=146

$h=7^{146} \bmod 997 = 634$

r=59

$y=7^{59} \bmod 997 = 602$

m=52

$s=(52+146*602)*59^{-1} \bmod 996 = (52+146*602)*287 \bmod 996 = 292$

$h^{-y} \bmod p = ?$

$y^{s} \bmod p = ?$

$h^{-y}\, y^{s} \bmod p = ?$

$g^{m} \bmod p = ?$

ElGamal in Action

p=997

g=7

x=146

$h=7^{146} \bmod 997 = 634$

r=59

$y=7^{59} \bmod 997 = 602$

m=52

$s=(52+146*602)*59^{-1} \bmod 996 = (52+146*602)*287 \bmod 996 = 292$

$h^{-y} \bmod p = (634^{602})^{-1} \bmod 997 = 708^{-1} \bmod 997 = 928$

$y^{s} \bmod p = 602^{292}$ mop 997 = 356

$h^{-y}\, y^{s} \bmod p = 928 * 356 \bmod 997 = 361$

$g^{m} \bmod p = ?$

ElGamal in Action

p=997

g=7

x=146

$h=7^{146} \bmod 997 = 634$

r=59

$y=7^{59} \bmod 997 = 602$

m=52

$s=(52+146*602)*59^{-1} \bmod 996 = (52+146*602)*287 \bmod 996 = 292$

$h^{-y} \bmod p = (634^{602})^{-1} \bmod 997 = 708^{-1} \bmod 997 = 928$

$y^{s} \bmod p = 602^{292}$ mop 997 = 356

$h^{-y}\, y^{s} \bmod p = 928 * 356 \bmod 997 = 361$

$g^{m} \bmod p = 7^{52} \bmod 997 = 361$

Schnorr Algorithm

Introduced in 1989

Claus Schnorr: German mathematician & cryptographer (father of SSL)

Patented until 2008

Extension of the ElGamal digital signature

- More efficient

Schnorr Algorithm

Digital Signature:

1. Alice chooses a secret random integer x, $0<x<q$ & calculates $h=g^x \bmod p$
2. Alice makes: public [p,q,g,h], secret [x]
3. Alice chooses a random integer r, $0<r<q$ & calculates $y=g^r \bmod p$
4. Alice calculates the hash $y'=f(m||y) \bmod q$ & creates a signature as (y',s), where $s=(r+x*y') \bmod q$
5. Bob verifies if the following are equal:
   - $(m||y) \bmod q$
   - $(m||g^s h^{-y'} \bmod p) \bmod q$

A common (p,q,g) for all users (precomputed):

- p: 1024-bit prime
- q: 160-bit prime
- g: $g^q = 1 \bmod p$

Generation algorithm:

- *Generate q*
- *Create a sequence $p=qp+1$ with random p until to hit a prime*
- *$g=\sigma^p \bmod p$ for g not 1*

Efficiency: signatures are generated mod q (instead of mod p)

2 exponentiations

Why?

$g^s h^{-y'} \bmod p = g^{(r+x*y')} g^{-x*y'} \bmod p = g^r g^{x*y'} g^{-x*y'} \bmod p = y$

Schnorr in Action

q=97

ρ=40

p=?

σ=61

g=?

x=76

h=?

r=22

y=?

y'=?

(φ, ψ primitive roots of q)

(y',s)=?

f(m||y) mod q = (m||g^s $h^{-y'}$ mod p) mod q ?

Schnorr in Action

$q=97$

$\rho=40$

$p= 97*40+1=3881$

$\sigma=61$

$g=?$

$x=76$

$h=?$

$r=22$

$y=?$

$y'=?$

(φ, ψ primitive roots of q)

$(y', s)=?$

$f(m||y) \bmod q = (m||g^{s} h^{-y'} \bmod p) \bmod q$?

Schnorr in Action

q=97

ρ=40

p= 97*40+1=3881

σ=61

g= 3440: 61^{40} =3440 mod 3881, 3440^{97} =1 mod 3881

x=76

h=?

r=22

y=?

y'=?

(φ, ψ primitive roots of q)

(y',s)=?

f(m||y) mod q = (m||g^s $h^{-y'}$ mod p) mod q ?

Schnorr in Action

$q=97$

$\rho=40$

$p= 97*40+1=3881$

$\sigma=61$

$g= 3440: 61^{40} =3440 \bmod 3881, 3440^{97} =1 \bmod 3881$

$x=76$

$h= 3440^{76} \bmod 3881=1568$

$r=22$

$y=?$

$y'=?$

(φ, ψ primitive roots of q)

$(y',s)=?$

$f(m||y) \bmod q = (m||g^s h^{-y'} \bmod p) \bmod q$?

Schnorr in Action

$q=97$

$\rho=40$

$p= 97*40+1=3881$

$\sigma=61$

$g= 3440: 61^{40} =3440 \bmod 3881, 3440^{97} =1 \bmod 3881$

$x=76$

$h= 3440^{76} \bmod 3881=1568$

$r=22$

$y=3440^{22} \bmod 3881 = 121$

$y'=?$

(φ, ψ primitive roots of q)

$(y', s)=?$

$f(m||y) \bmod q = (m||g^s h^{-y'} \bmod p) \bmod q$?

Schnorr in Action

$q=97$

$\rho=40$

$p= 97*40+1=3881$

$\sigma=61$

$g= 3440: 61^{40} =3440 \bmod 3881, 3440^{97} =1 \bmod 3881$

$x=76$

$h= 3440^{76} \bmod 3881=1568$

$r=22$

$y=3440^{22} \bmod 3881 = 121$

$y'= f(m||y)=\varphi^m \psi^y \bmod q = 10^{52} * 21^{121} \bmod 97 = 88 * 74 \bmod 97 = 13$

(φ, ψ primitive roots of q)

$(y', s)=?$

$f(m||y) \bmod q = (m||g^s h^{-y'} \bmod p) \bmod q$?

Schnorr in Action

$q=97$

$\rho=40$

$p= 97*40+1=3881$

$\sigma=61$

$g= 3440: 61^{40} =3440 \bmod 3881, 3440^{97} =1 \bmod 3881$

$x=76$

$h= 3440^{76} \bmod 3881=1568$

$r=22$

$y=3440^{22} \bmod 3881 = 121$

$y'= f(m||y)=\varphi^m \psi^y \bmod q = 10^{52} * 21^{121} \bmod 97 = 88 * 74 \bmod 97 = 13$

(φ, ψ primitive roots of q)

$(y',s)=(13, (22+76*13) \bmod 97) = (13,40)$

$f(m||y) \bmod q = (m||g^s h^{-y'} \bmod p) \bmod q$?

Schnorr in Action

$q=97$

$\rho=40$

$p= 97*40+1=3881$

$\sigma=61$

$g= 3440: 61^{40} =3440 \bmod 3881, 3440^{97} =1 \bmod 3881$

$x=76$

$h= 3440^{76} \bmod 3881=1568$

$r=22$

$y=3440^{22} \bmod 3881 = 121$

$y'= f(m||y)=\varphi^m \psi^y \bmod q = 10^{52} * 21^{121} \bmod 97 = 88 * 74 \bmod 97 = 13$

(φ, ψ primitive roots of q)

$(y',s)=(13, (22+76*13) \bmod 97) = (13,40)$

$f(m||y) \bmod q = (m||g^s h^{-y'} \bmod p) \bmod q$?: $554 * 2305 \bmod 3881 = 121 = y$

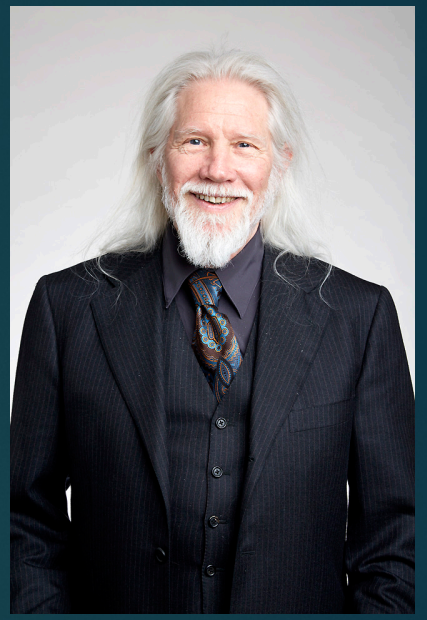
Diffie-Hellman Algorithm

Introduced in 1976

Invented by Whitfield Diffie & Martin Hellman

Security: calculation of discrete logarithms

Nowdays requires at least a p of 2048 bits or the use of alternatives, e.g. *elliptic curve cryptography*



Diffie-Hellman Algorithm

Key exchange:

1. Alice or Bob chooses parameters p & g, & makes them public
2. Alice chooses a secret random integer x, & calculates $h=g^x \bmod p$ that makes it public
3. Bob chooses a secret random integer r & calculates $y=g^r \bmod p$ that makes it public
4. Alice & Bob make: public [p,g,h], secret [x,r]
5. Alice & Bob calculate their shared secret key: $s = (g^x)^r = (g^r)^x = g^{x*r} \bmod p$

$G(a,b)=a^b \bmod p$

$F(a) = G(g,a)=g^a \bmod p$, where $G(g,a)$ is a primitive root mod p

Satisfied symmetry: $=G(F(x),r)=(g^x)^r = (g^r)^x =G(F(r),x) \bmod p$

Questions?

Blockchain & Cryptoeconomics

DR. EVANGELOS POURNARAS

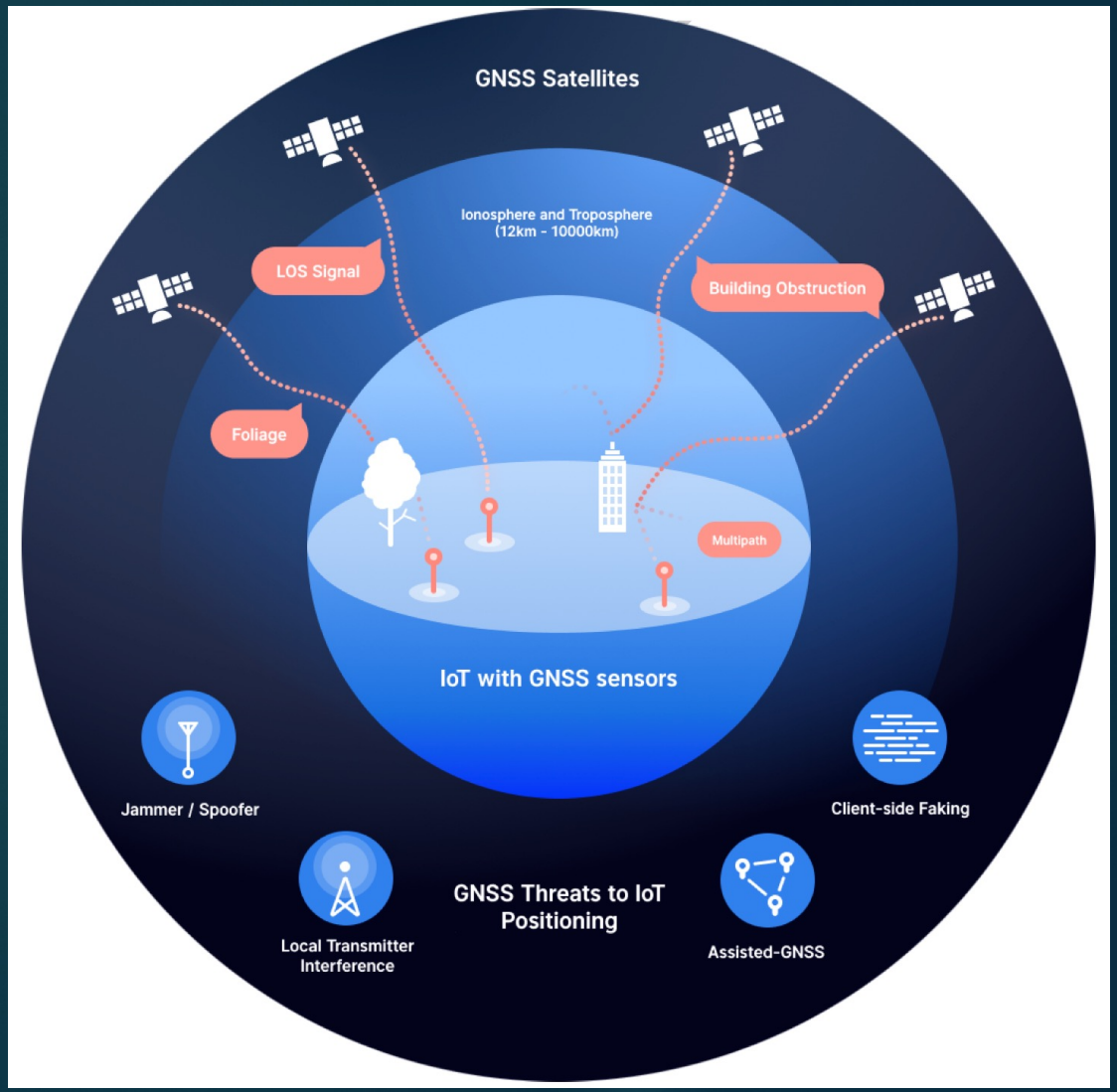
8. Proof of Witnessed Presence:

Digital Consensus in Physical World

Consensus on geolocation becomes critical

GPS is unreliable

- Single point of failure
- Does not penetrate indoors & underground
- Signal multipath increases with urban density
- Energy intensive for long-term maintenance devices
- Spoofing



Secure Localization & Verification

Spatially distributed signal information for localization:

- Transmission distance
- Anglular incidence parameters

Localization systems

- Node-centric vs. infrastructure-centric
- Rang-free vs. range-based: *signal propagation times, signal strength, angle of arrival*

Distance bounding protocols against adversaries:

- Signal interception
- Jamming
- Replay/modify packets

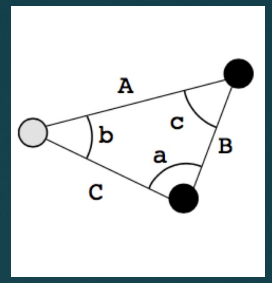
Challenge-response protocol solution:

- Measuring return time of flight

Positioning Systems

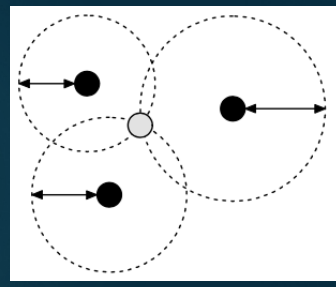
Triangulation

- **Lateration** (distance) vs. **angulation** (angle)
  - **Time of Flight** (ToF)
    - **Time of Arrival** (ToA):
      - Signal propagation time
      - Assumes synchronized clocks, known propagation velocity
    - **Time Difference of Arrival** (TDoA)
      - Two signal velocities: radio & acoustic
      - Special additional hardware
      - No synchronized clocks
    - **Round Trip Time of Flight** (RToF)
  - **Received Signal Strength Indicator** (RSSI)



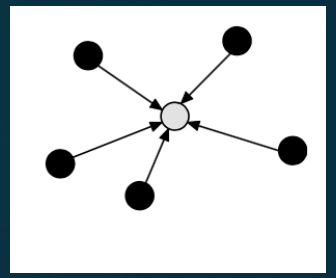
Trilateration

- Distance between transmitter-receiver (ToA, ToF, RSSI)
- 3 reference nodes with known positions
- Location: Intersection of three circles formed by reference nodes

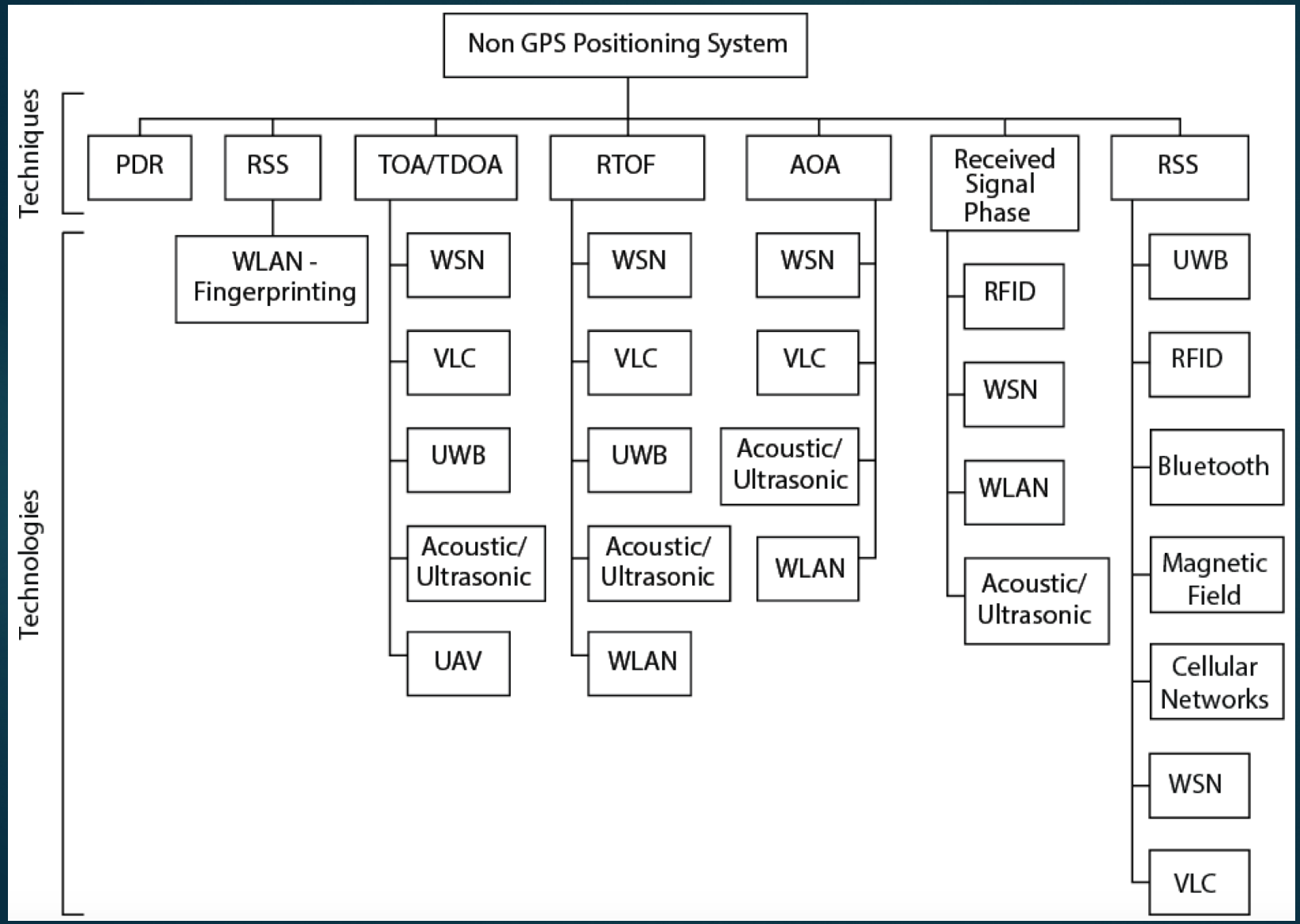


Multilateration

- TDoA
- Synchronized transmitters
- Location: intersection of at least two hyperbolas (3 antennas required)



Positioning Systems



Clock Synchronization

Local oscilators experience drift: temperature, aging

Clock drifts cause **gradual degradation of synchronicity**

Clock must be periodically **resynchronized**

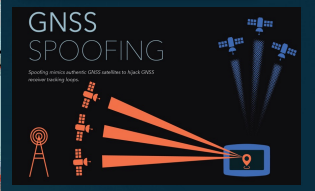
Byzantine fault-tolerant solution for clock synchronization: Malekpour's work, **F=N/3 adversaries**

- **System liveness**: system live for longer than convergence bounds
- **Convergence/closure**: synchronization in maintained
- **Congruence**: synchronization is detected locally

Adversary behaviors:

- **Benign**: detectably does not follow the protocol
- **Symmetric**: uncertainty on the same detected behavior
- **Bounded-arbitrary** (Byzantine): different behaviors detected

Spatio-temporal Evidence



Proving location & time: *localization mechanisms, sensor fussion, anomaly detection, social witnessing, etc.*

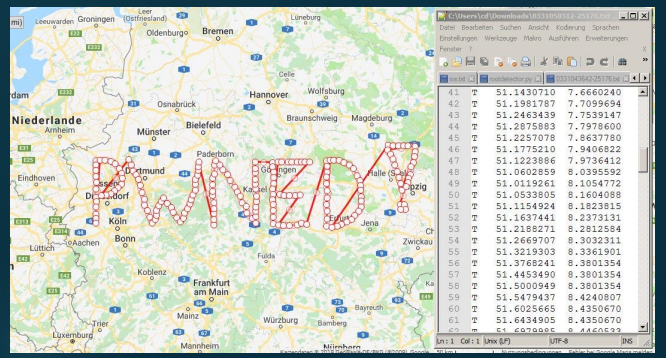
- GPS-based proofs of location are vulnerable
- Mobile cellular networks as oracles: require roaming
- Alternatives: LPWAN & P2P ad hoc opportunistic networks

Proving situational awareness: QR codes, puzzle questions, CAPTCHA-like tests, collaborative social challenges

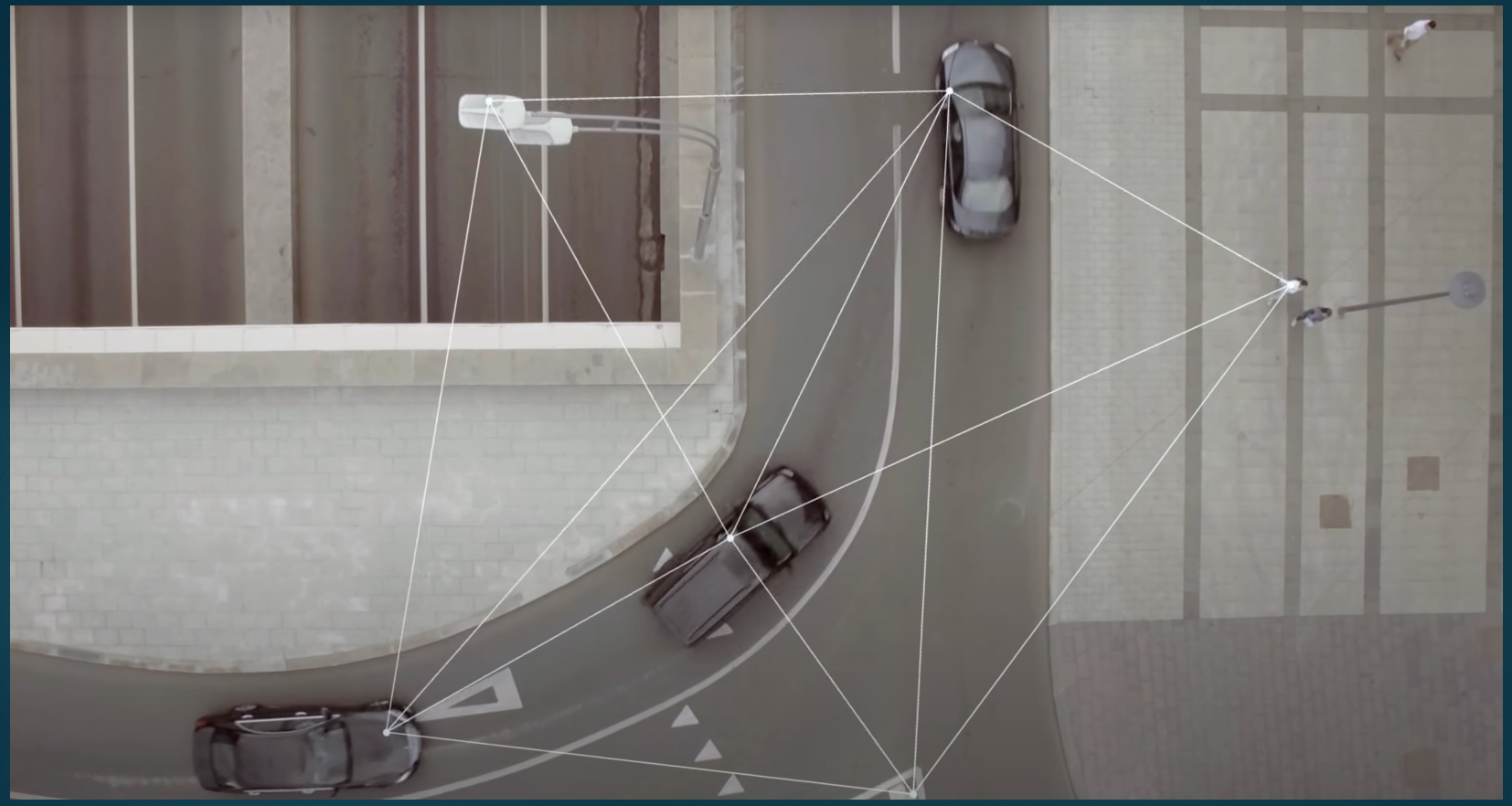
Approaches	GPS [76]	Mobile Cellular Network [103]	LPWAN [27]	P2P Ad Hoc Networks [35]
Infrastructure-independent	No	No	No	Yes
Decentralization	Low	Low	Medium	High
Access	Open	Closed	Open	Open
Management	Governmental-level	Enterprise-level	Community-level	Self-organized
Disaster Resilience	Medium	Medium	Medium	High
Coverage Range	Global	National	Urban	Localized
Indoor Coverage	No	Yes	Yes	Yes

Why blockchain?

- **Distributed trust & self-governance**: communities institutionalizing their own rules of consensus
- **Security/privacy mechanisms**: zero-knowledge proofs, homomorphic encryption, differential privacy, etc.
- **Incentive mechanisms**: crypto-economic models, token curated registries, multiple currencies for rewarding different community values



A Consensus-driven Map



FOAM

Source: https://www.youtube.com/watch?v=_Vr_cysyfOc

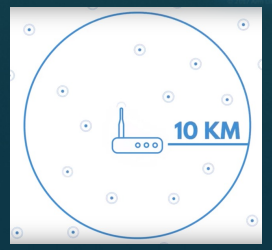
1. Localization Infrastructure

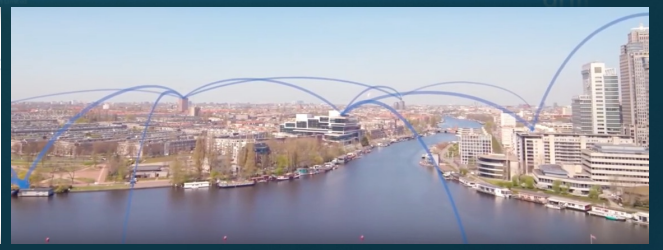
Open & decentralized solution

Crowd-sourced location service providers

LoRa WAN radio beacons with long-range metropolitan coverage

Token curated registries: stakes a safety deposit – FOAM token (cryptocurrency)





2. Zone Formation

Searching for other zone anchors (radio beacons)

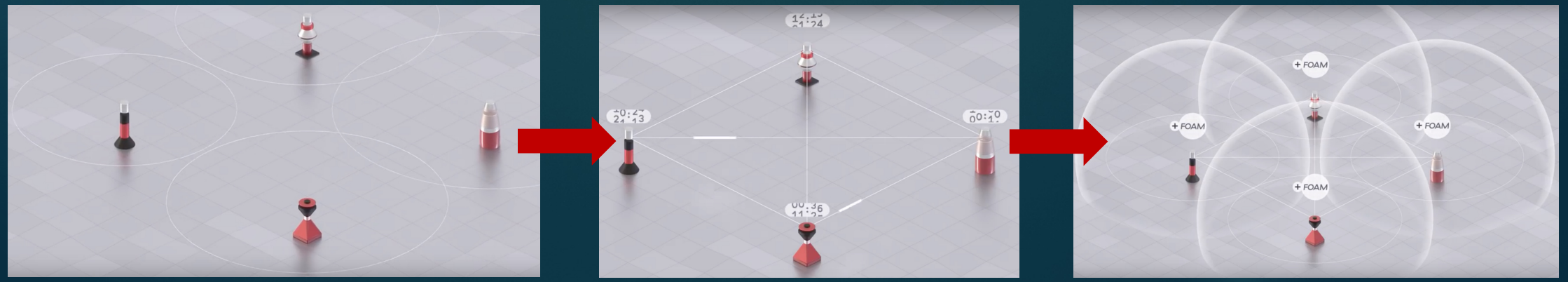
Discover & get connected

Exchange messages to synchronize their clocks

- Signal attenuation & propagation times
- Byzantine fault-tolerant clock synchronization

A time consensus results in a decentralized zone for location services

Rewarding zone anchors with FOAM tokens



3. Verifying Presence Claims

Location customers: make presence claims

Zone anchors: Mine triangulations & verify presence claims

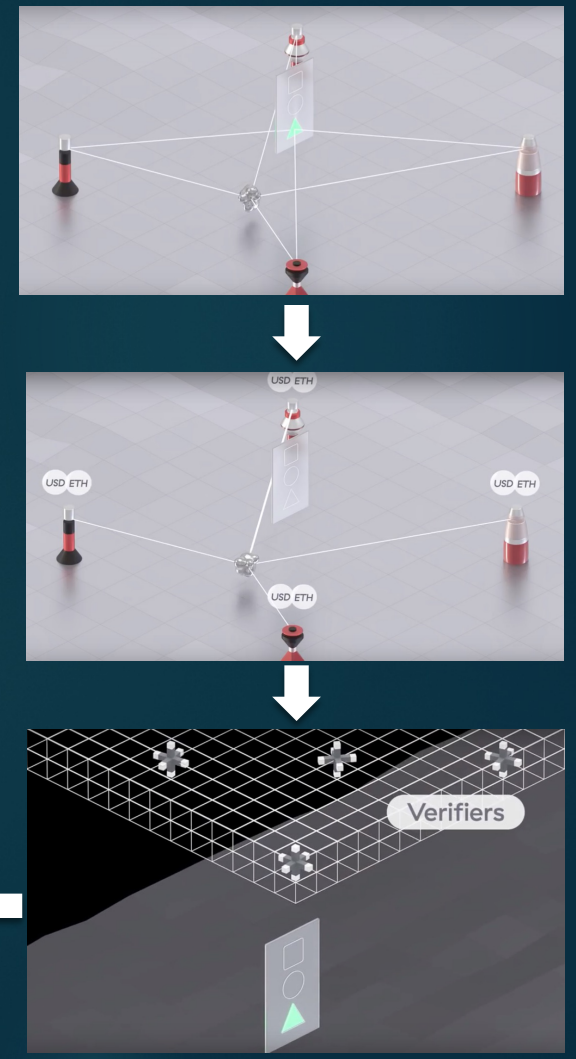
- Distance detection by message travel times

Location customers reward zone anchors with a fee

Local blockchain storage:

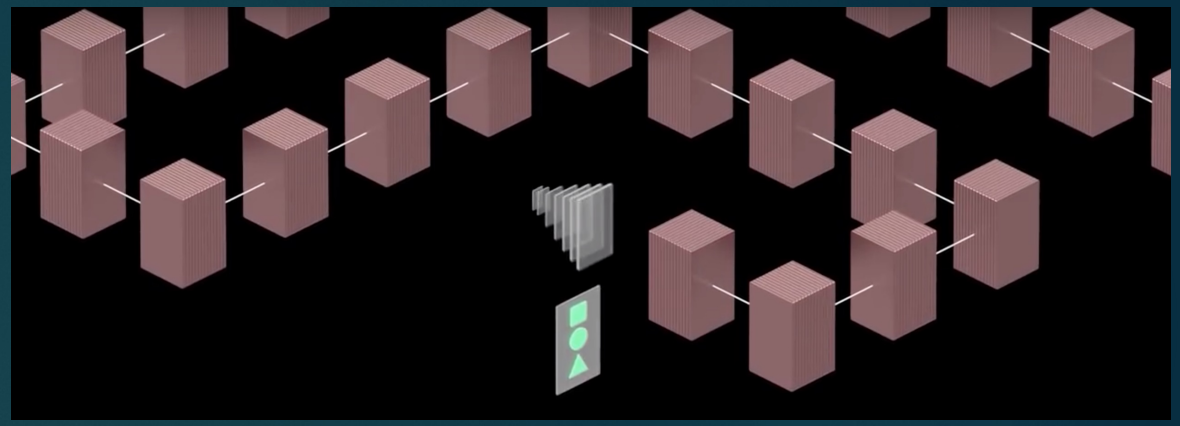
- Reaching consensus of customers' location

Presence claims are further verified among zones to make sure that **zone anchors remain in sync**

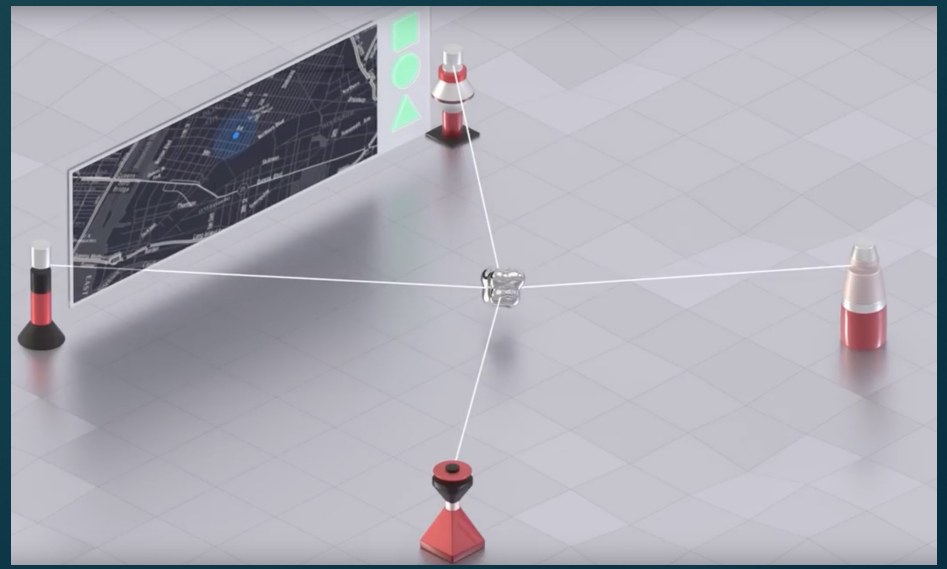


4. Publishing Proofs of Location

The verified presence claim is written into the Ethereum blockchain & made public



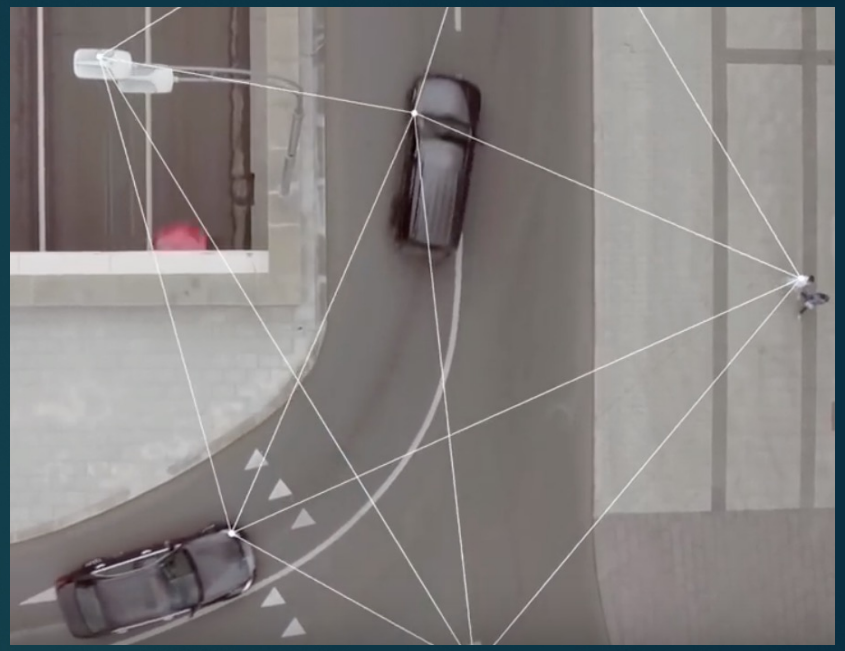
A consensus-driven map: Verified presence claims are made available to location customers via decentralized applications

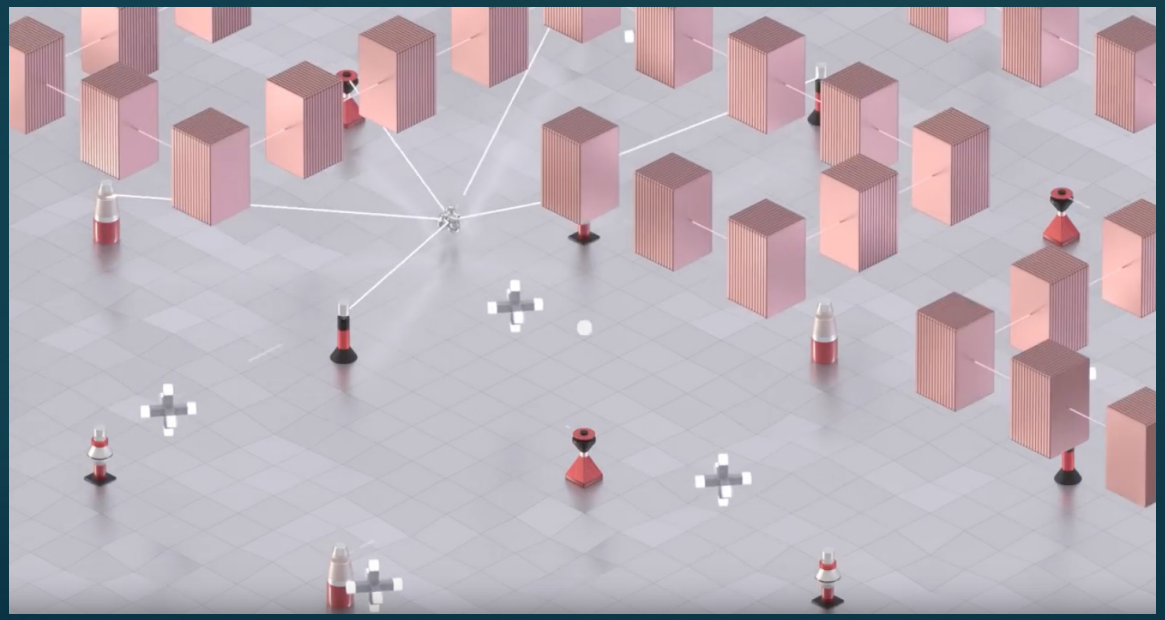


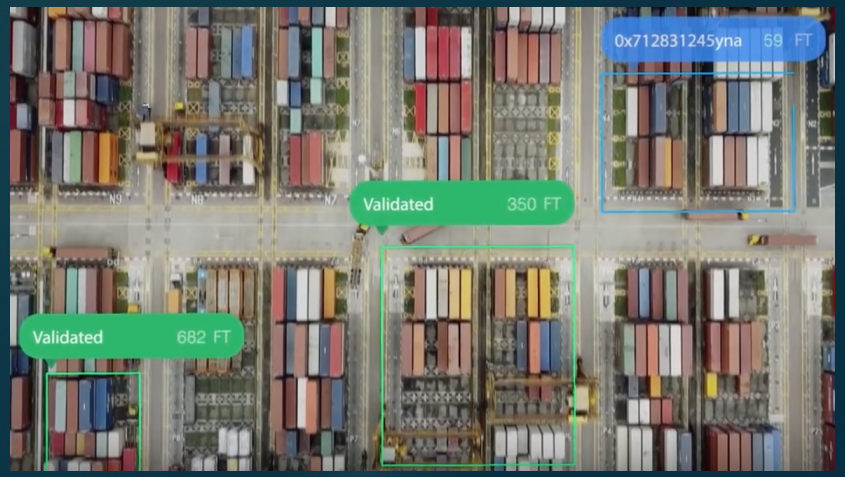
A Consensus-driven Map

Blockchain proofs of location with FOAM

Source: https://www.youtube.com/watch?v=_Vr_cysyfOc







Proof of Witnessed Presence

Validator set

- Physical distance (LPWAN, P2P Ad Hoc networks)
- Nodes for proof of stake

Validator weight

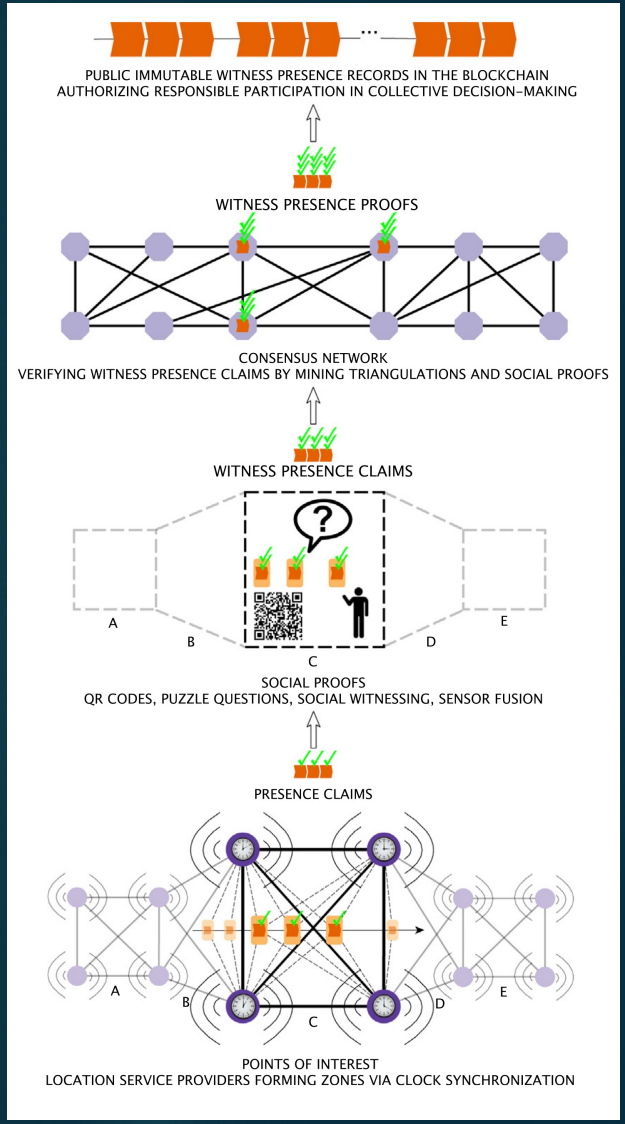
- Num. of staked tokens
- Reputation (participation level & legitimacy of witnessed presence)

Validator criteria

- Matching the signature to validator set
- Minimum staking
- No slashing: BFT clock synchronization

Validator verifiability

- Signed receipts of all clock sync messages



Incentivizing Witnessed Presence

Utility token rewarding:

- Citizens & community localization infrastructure for location proofs
- Social proofs in points of interest for proving social claims
- Use of computational resources for validation of witnessed presence claims in the consensus network

Witnessed presence claims:

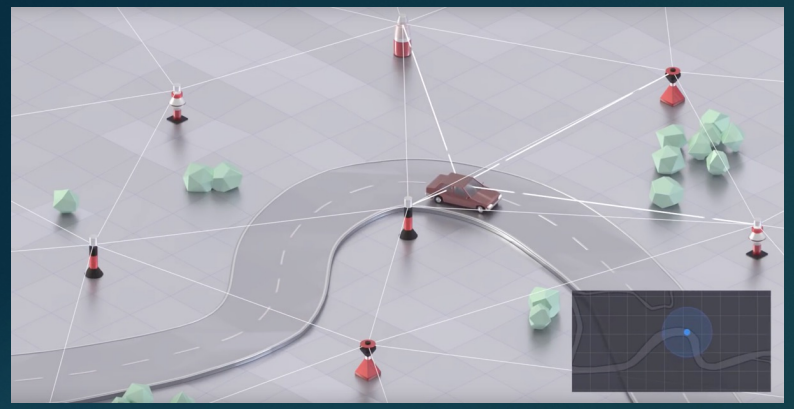
- Come with a fee (token/fiat) payed to witnessed presence service providers
- Further development/maintenance of infrastructure, improving localization accuracy, incentivizing new points of interest & augmenting them with social proofs

Penalties for slashing conditions >> short term rewards

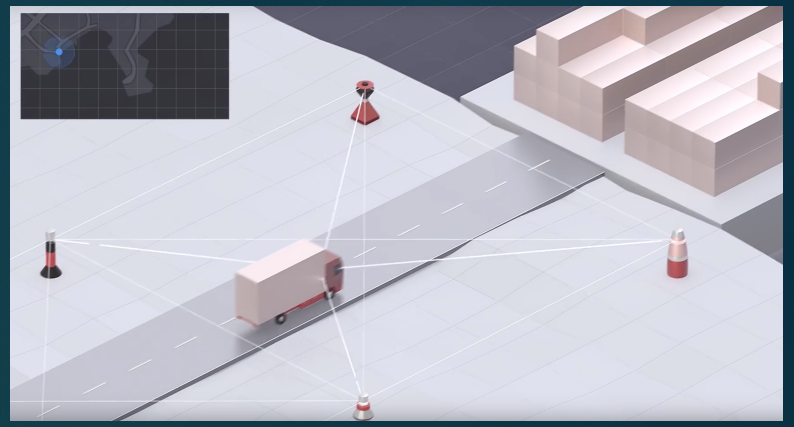
Resistance to Sybil attacks: entry, existence & exit costs

Incentives: economic, social, environmental & political

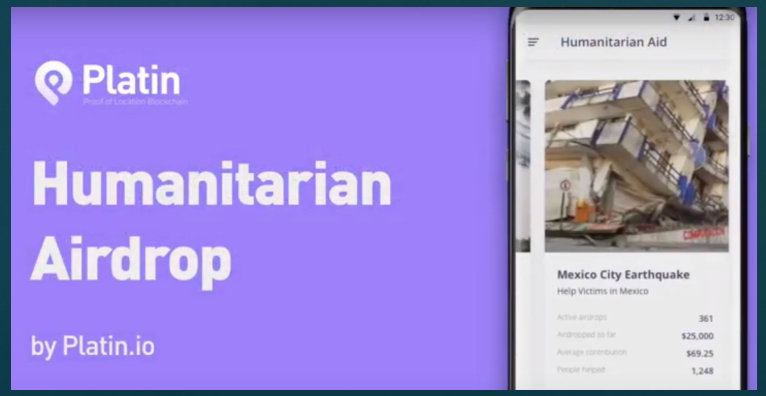
Decentralized Applications



Mobility & transport



Logistics & supply chain



A map of donations

Decentralized Applications

Augmented democracy: making citizens' decisions subject of proving witnessed presence. Verifying:

- Location
- Time
- Situation awareness

Reclaiming the public sphere of urban environments:

- Turning every urban spot into a secure digital voting center
- Beyond identity verification in voting
- Voting as a responsible informed testimony
- Voting as an intervention for an evidence-based solution
- Bring citizens' solutions to problems (no problems to citizens)

A digital revive of a cyber-physical Agora

Witnessed Presence of Accident Risk

Baseline: empirical accident data

Continuous risk model estimation

Real-world accident data reported by Federal Roads Office at Swiss Geoadmin

4 selected spots – extreme risk gradient

Treatment: witnessed presence of risk

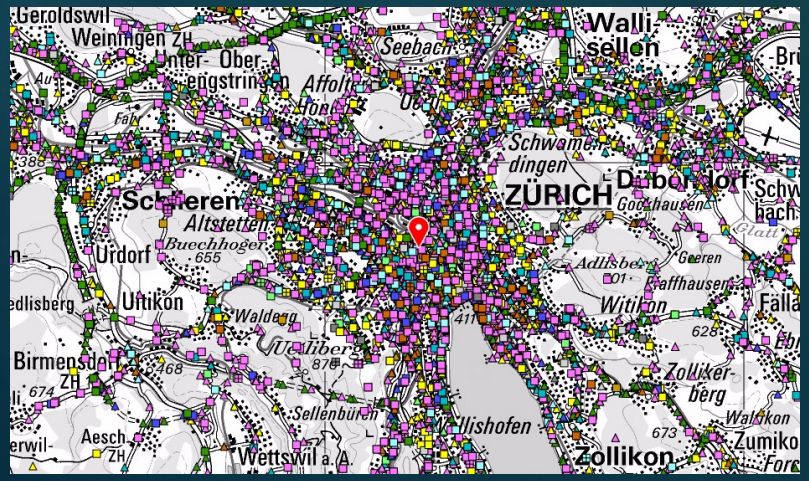
11 cyclers

Rating cycling risk at each spot

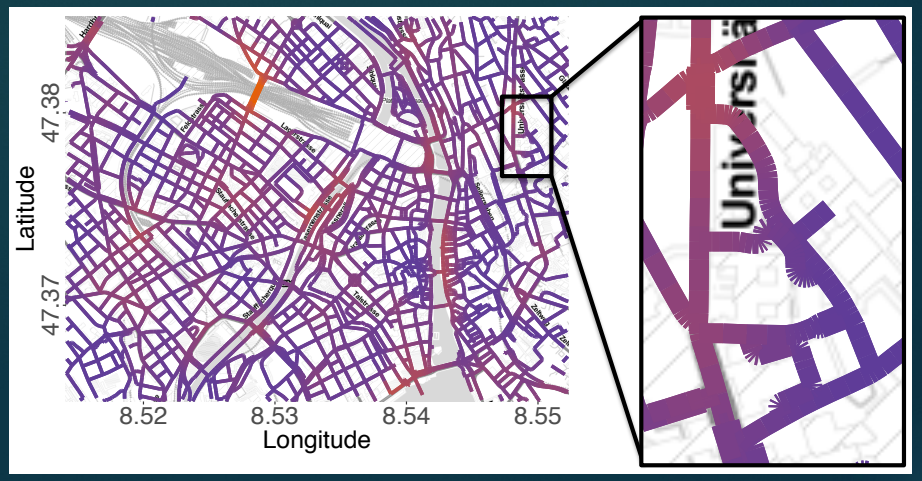
1. very safe to 5. very dangerous

Same bike, same time, same sequence

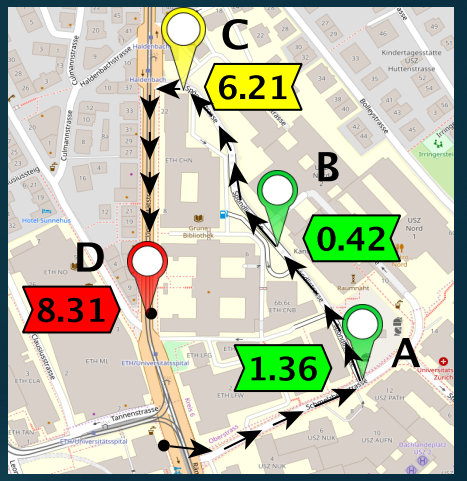
Accident data [2011-2017]



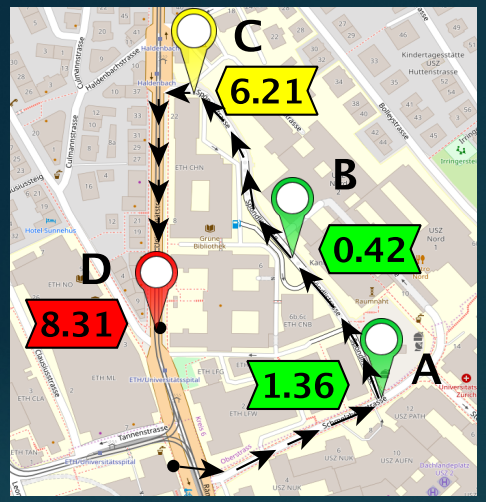
Continuous Risk Estimation Mode



Selected Spots



Witnessed Presence of Accident Risk



Spot A

Risk=1.36

Spot B

Risk=0.42

Spot C

Risk=6.21

Spot D

Risk=8.31

Locations	Test users:	1	2	3	4	5	6	7	8	9	10	11	Mean	Median	Actual cycling risk [42]
Spot A		2	2	2	1	1	1	1	2	2	1	2	1.55	2	1.36
Spot B		1	1	1	1	1	1	1	2	1	1	1	1.09	1	0.42
Spot C		2	1	1	1	2	3	1	3	4	2	2	2.0	2	6.21
Spot D		3	3	3	2	4	4	2	2	3	4	4	3.09	3	8.31

1. very safe to 5. ery dangerous

Pearson correlation:	0.94	0.85
Spearman correlation:	1.0	1.0

High matching between empirical risk map & perceived risk by witnessed presence

Questions?

Blockchain & Cryptoeconomics

DR. EVANGELOS POURNARAS

9. Foundations of Consensus

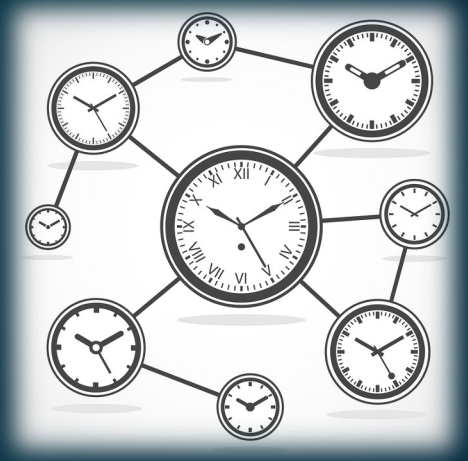
The Problem of Consensus

A fundamental computer science problem:
reliability in the presence of faulty processes

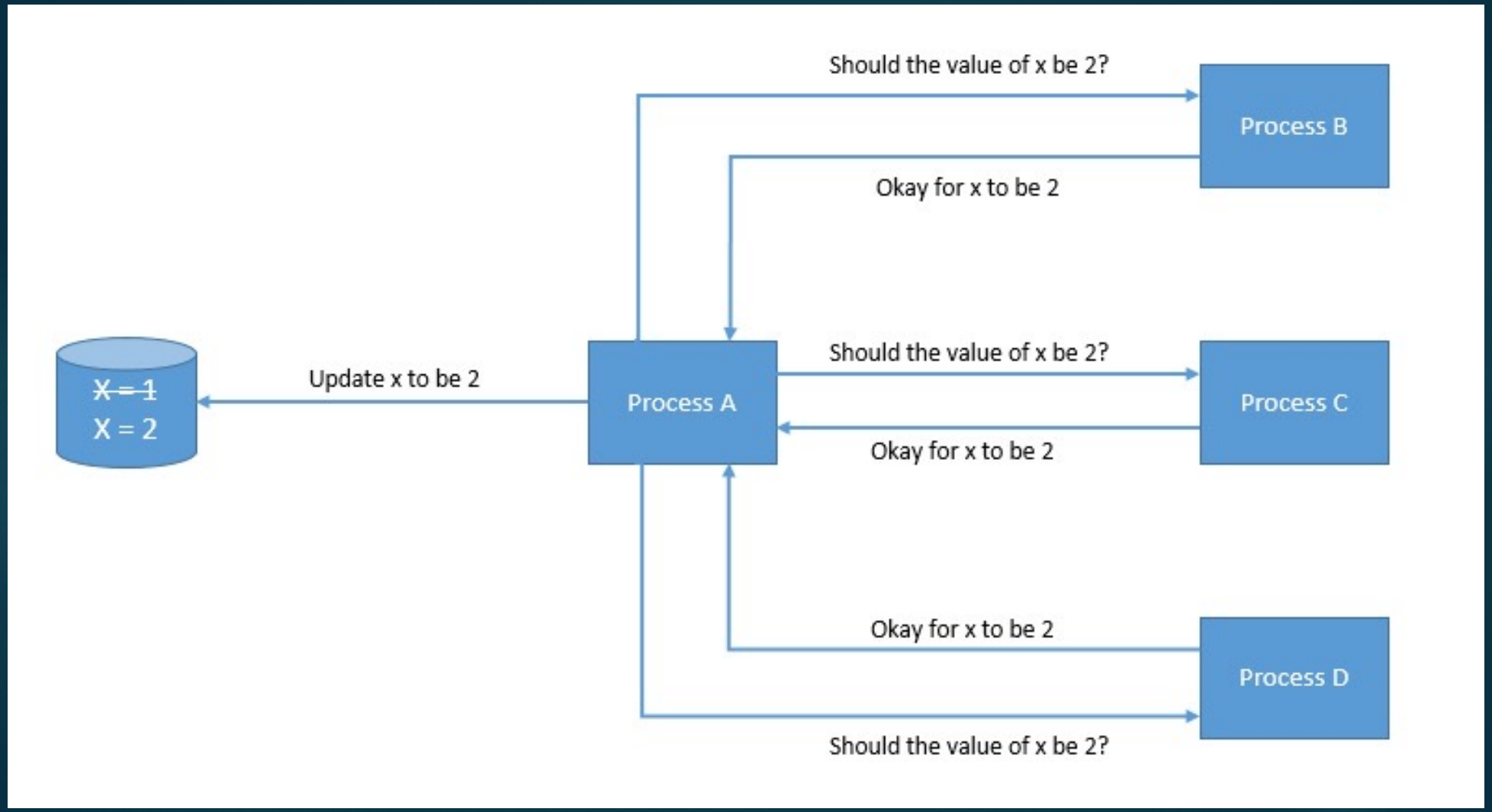
Coordination of processes to reach an agreement on a value

A problem with applicability beyond blockchain:

- *Cloud computing*
- *Clock synchronization*
- *UAVs & robotics*
- *Smart Grid*



The Problem of Consensus



Crash vs. Byzantine Faults

Crash faults: a process abruptly stops & does not resume

Byzantine fault: *malicious activity, conflicting views of reality, more arbitrary & disruptive*

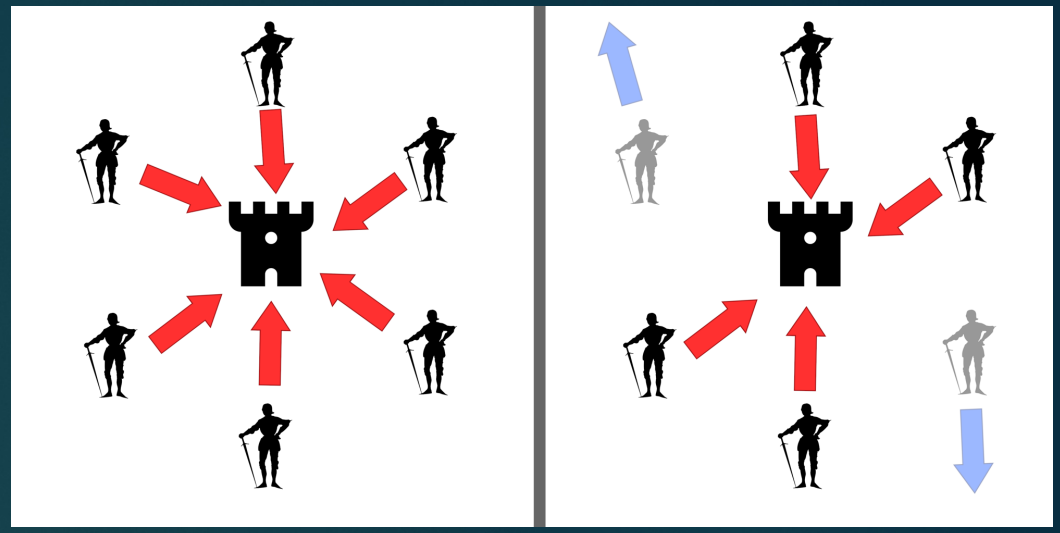
Consensus approach: *agree on a majority value*

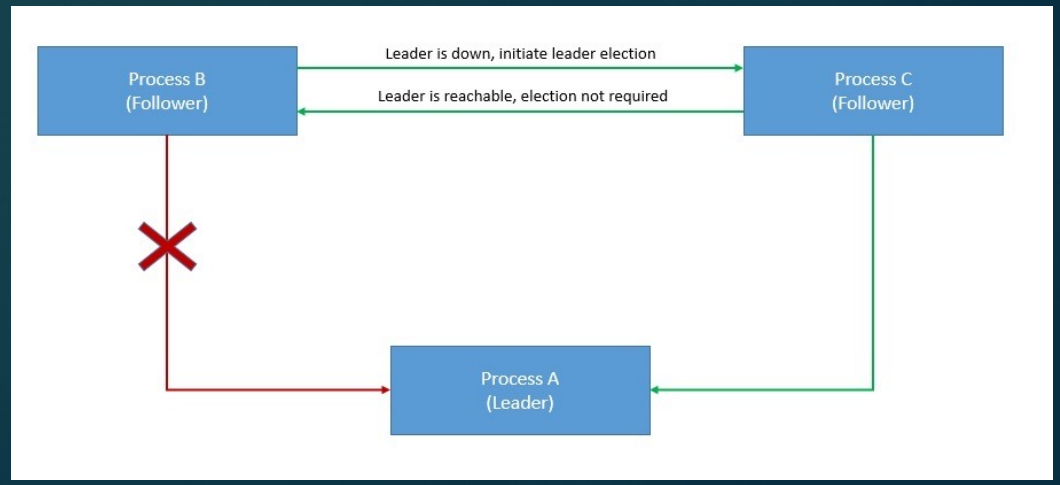
Faulty processes may skew the results.

Effect:

- No consensus
- Incorrect consensus

Fault tolerance: *resistant to limited # of faulty processes*





Attributes of Consensus

Properties:

- **Termination**: Eventually, every correct process decides a value
- **Integrity**: If all correct processes decide a value, any correct process must decide this value
  - **Stronger Byzantine version**: if a correct process decides a value, this this must have been proposed by a correct process
- **Agreement**: Every correct process must agree on the same value

t-resilient protocol: one that can correctly guarantee consensus among n processes of which at most t fail

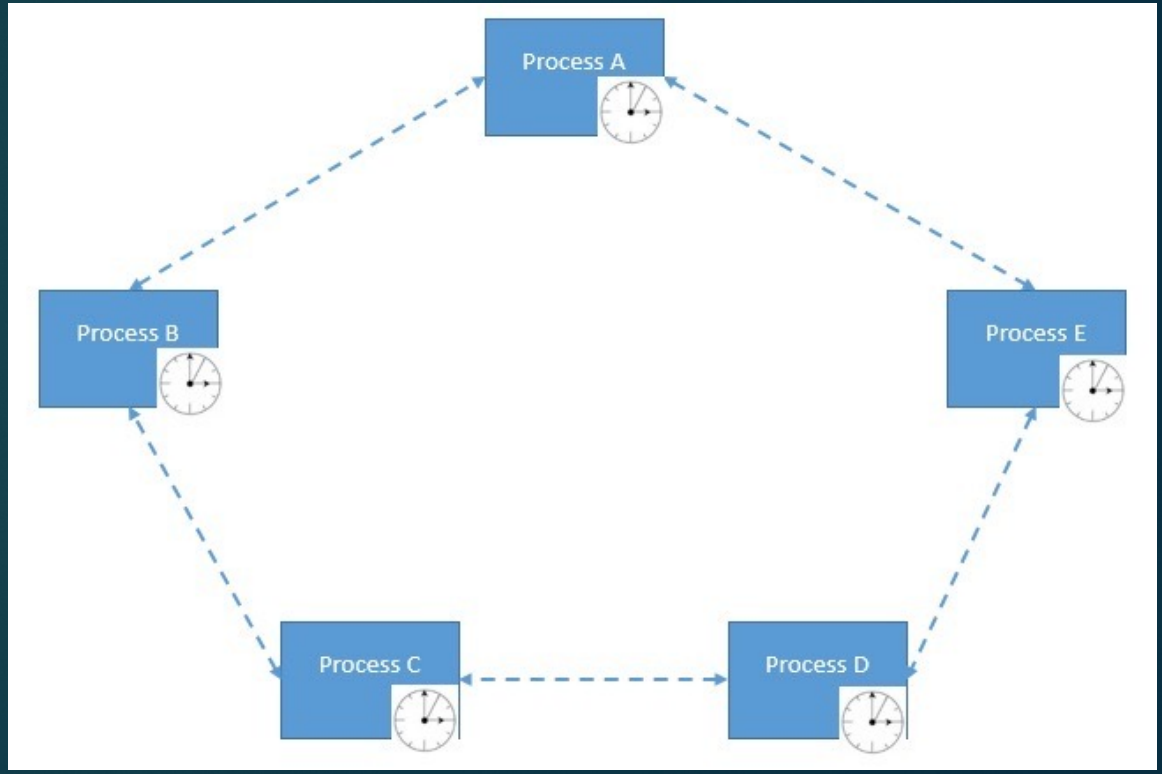
Performance assessment:

- **Time complexity [O(n)]**: number of rounds of message exchanges to reach consensus
- **Message complexity**: # of messages (message traffic)
- **Memory usage, message size**

Communication Models

Communication:

- **Asynchronous:** independent processes with their own clocks, reality
- **Synchronous:** communication in rounds, all messages sent & may received within a round, no influence of a message within the same round



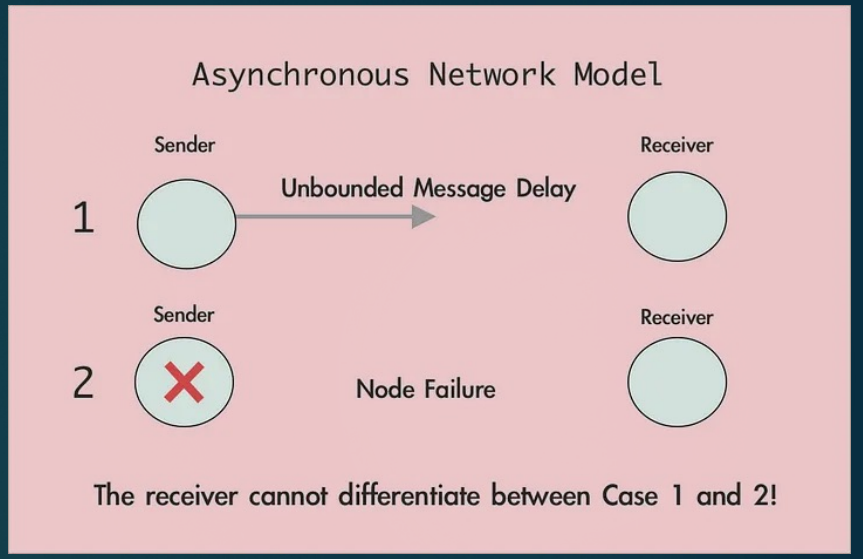
FLP Impossibility

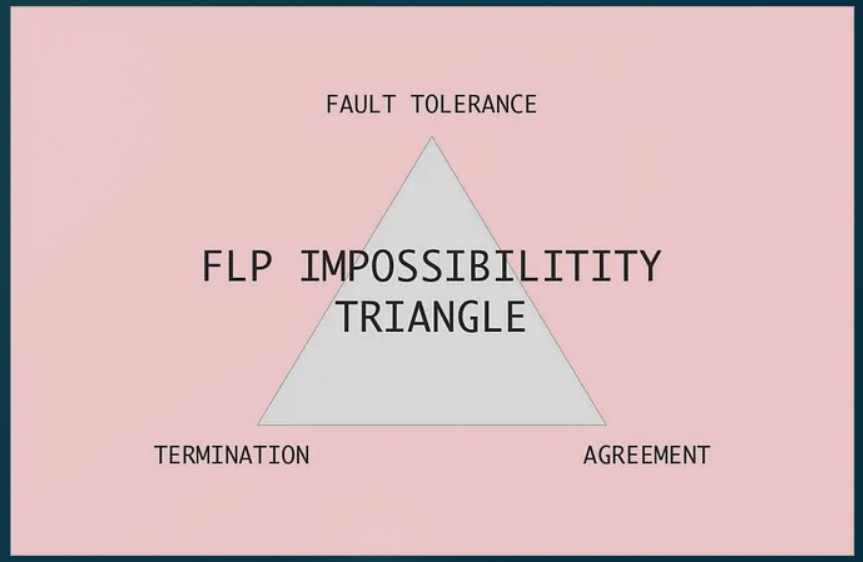


Michael J. Fischer Nancy A. Lynch Michael S. Paterson

Even when one process crashes, there is no deterministc algorithm that can solve the consensus problem

- Communication channels are reliable
- Non-Byzantine failures
- Cannot distinguish a faulty from a slow process
- **Proof insight**: decisions depend on the order of message arrival rather than on the values of the processes





FLP Impossibility in Real World

How to practically overcome impossibility:

- **Relax determinism**, introduce randomness, e.g. Las Vegas algorithm
  - Consensus is always reachable within an (unbounded) time
  - Probability to reach consensus increases exponentially with time
  - Broad applicability, e.g. Bitcoin consensus
- **Relax asynchronicity**, introduce partial synchrony, e.g. Casanova algorithm
  - Messages delivery within a bounded time Δ
  - Optimization of Δ to speed up consensus
- **Relax termination**, leaders' election e.g. Raft
  - Randomized election timeouts

Asynchronous Model	Real World System
Reliable Message Channels; Unbounded delays.	Resend message until receive ACK. Message timeout.
No clocks.	Wall clock time with limited synchronisation between nodes.
Crash failures cannot be detected reliably.	Usually detect failures with pre-determined timeout.

Types of Consensus

Protocols based on partially synchronous models

- 1/3 failures tolerance
- Bounded network latency
- Contains Byzantine faults

Deterministic protocols based on asynchronous models

- Unbounded network latency
- Impossibility, no tolerance of single faults

Protocols based on a synchronous model

- Bounded network latency
- 100% fault tolerance, but: limited nodes behavior for > ½ network failures

Agreement Problems

1. Termination of reliable broadcast

The General's Problem:

0,…,n-1 processes, process 0 transmits a value v to all processes such that:

1. *If process 0 is correct, every correct process receives v*
2. *For any two correct processes, each process receives the same value*

2. Consensus

- **Agreement**: All correct processes must agree on the same value
- **Weak validity**: For each correct process, its output must be input from a correct process

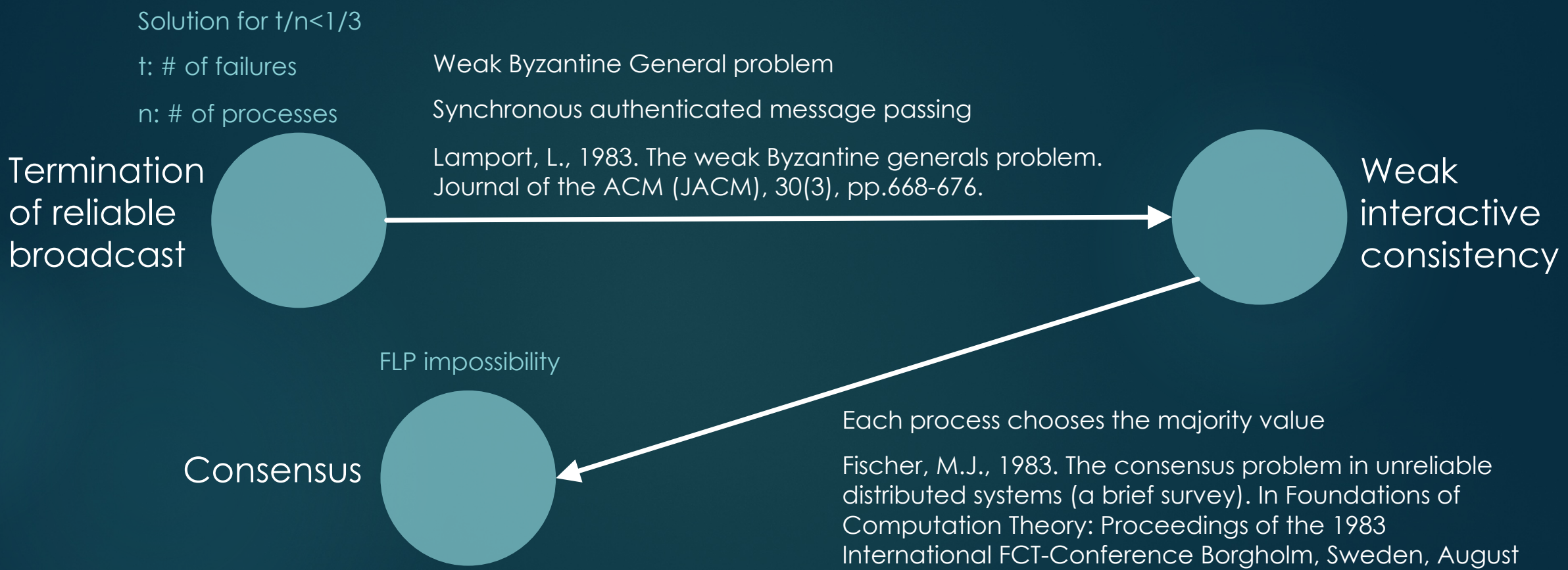
3. Weak interactive consistency

n processes with a private value in a partially synchronous system, communication in rounds to determine the public consensus value

1. *If a correct process sends v, all current processes receive v or nothing (integrity)*
2. *All sent messages by a correct processes are received in the same round by all correct processes (consistency)*

- **Strong validity**: If all correct processes receive the same input value, they must also output this value
- **Termination**: All processes must eventually decide on an output value

Solvability & Equivallency



The Paxos Algorithm

Used widely: Apache ZooKeeper, Google Chubby

Introduced by Leslie Lamport

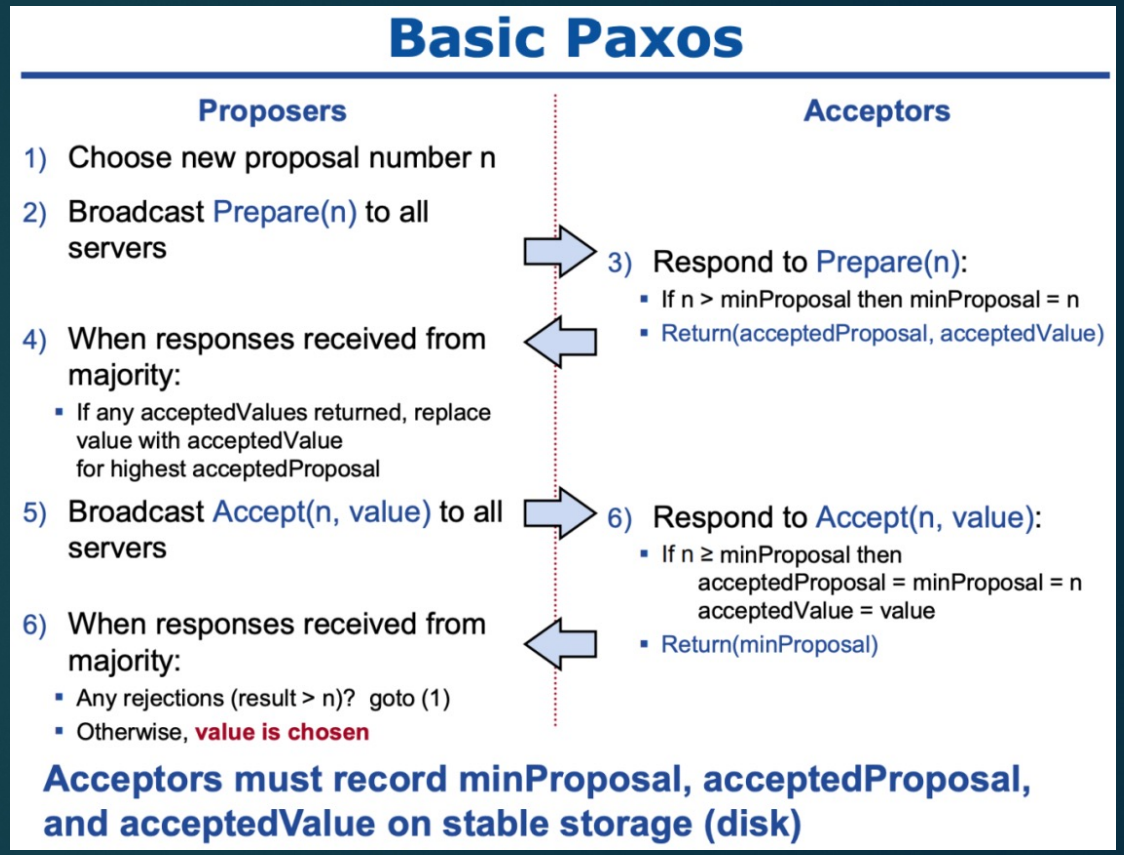
Notoriously complex

A non-Byzantine fault-tolerance algorithm

FLP relaxation: sacrifices liveness

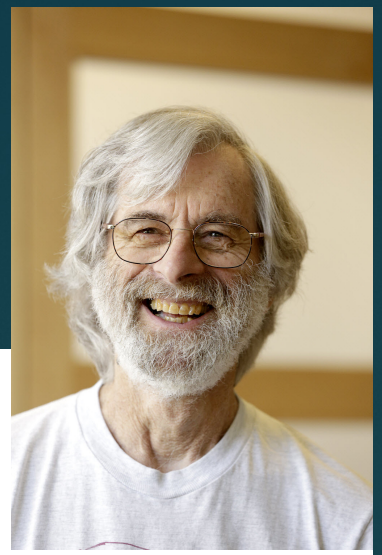
CAP relaxation: ensures consistency & partition tolerance, lowers availability (depends on # of "Learners")

Why it is powerful: conditions that affect liveness & availability are not easily triggered, & if triggered impact is not unacceptable



https://isaacwr.github.io/thesecretlivesofdata/paxos/#overview

The Paxos Algorithm



The Part-Time Parliament

LESLIE LAMPORT
Digital Equipment Corporation

Recent archaeological discoveries on the island of Paxos reveal that the parliament functioned despite the peripatetic propensity of its part-time legislators. The legislators maintained consistent copies of the parliamentary record, despite their frequent forays from the chamber and the forgetfulness of their messengers. The Paxon parliament's protocol provides a new way of implementing the state-machine approach to the design of distributed systems.

Categories and Subject Descriptors: C2.4 [**Computer-Communications Networks**]: Distributed Systems—*Network operating systems*; D4.5 [**Operating Systems**]: Reliability—*Fault-tolerance*; J.1 [**Administrative Data Processing**]: Government

General Terms: Design, Reliability

Additional Key Words and Phrases: State machines, three-phase commit, voting

"Inspired by my success at popularizing the consensus problem by describing it with Byzantine generals, ***I decided to cast the algorithm in terms of a parliament on an ancient Greek island****. Leo Guibas suggested the name Paxos for the island. I gave the Greek legislators the names of computer scientists working in the field, transliterated with Guibas's help into a bogus Greek dialect. (Peter Ladkin suggested the title.)* ***Writing about a lost civilization allowed me to eliminate uninteresting details and indicate generalizations by saying that some details of the parliamentary protocol had been lost****. To carry the image further, I gave a few lectures in the persona of an Indiana-Jones-style archaeologist, replete with Stetson hat and hip flask.* ***My attempt at inserting some humor into the subject was a dismal failure****. People who attended my lecture remembered Indiana Jones, but not the algorithm.* ***People reading the paper apparently got so distracted by the Greek parable that they didn't understand the algorithm****."*

PBFT Algorithm

Practical Byzantine Fault Tolerance

Improves earlier inefficient, slow & complex solutions

Complexity: from exponential to polynomial

Byzantine fault tolerance **if adversaries < 1/3**

Consensus based on **majority principle**

Leader node & backup nodes changing each round (view)

Safety on **asynchronous communication**

- But depends on message timeouts for **periodic synchronization**

Leader re-election is hard

- **Malicious leaders**: Sending messages very close to timeout – system slow down
- Repeated leaders election

FLP relaxation: sacrifices liveness, **CAP relaxation**: lowers availability

Many extensions

PBFT Protocol

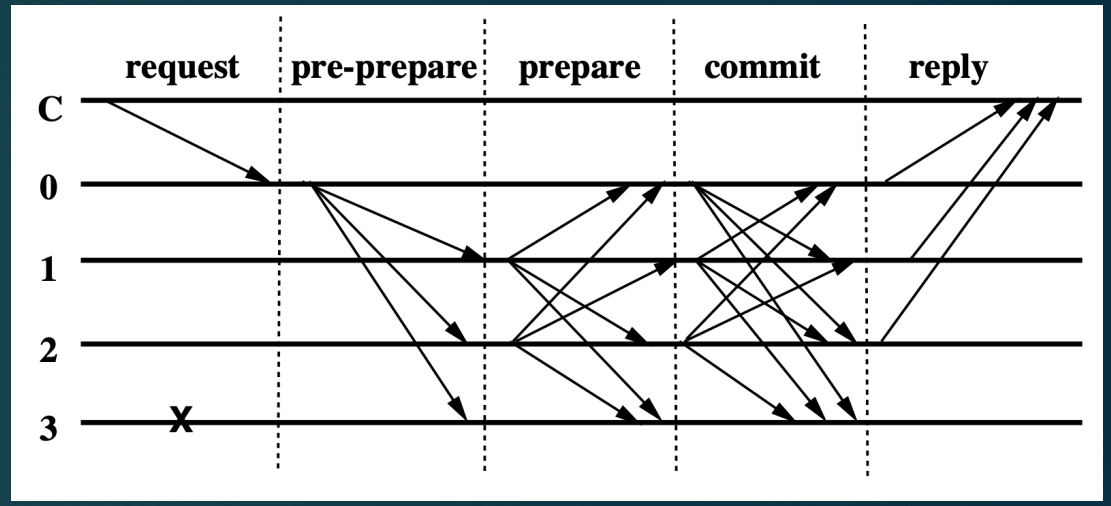
Launch: client c initiates service request to leader 0

Pre-prepare: Leader verifies request validity, assigns sequence num & broadcasts message

Prepare: Backup nodes verify request validity & accept sequence num. A prepare message is broadcasted

Commit: When prepare message received, commit messages are broadcasted

Reply: When received message received, a reply message is sent to client. It awaits f+1messages for success



Questions?