Blockchains & Distributed Ledgers

Lecture 01

Aggelos Kiayias

Introduction

- Why study blockchains?
- Introduction to Blockchain protocols
  - The never-ending book parable
- A new paradigm for information technology
  - What general IT problem does Bitcoin solve?
  - Resource-based systems
- Basic cryptographic primitives used by Bitcoin blockchain
  - Hash functions
  - Digital Signatures
  - Proof of work

Why study Blockchains?

Why study Blockchains?

- Provide good foundations for exploring the security of information systems in general.
- Highlight the importance of decentralisation, a property of increasing importance in the design of modern information systems.
- Facilitate a solid understanding of many security critical components, incl.
  - Key management.
  - Software security.
  - Privacy preserving technologies.
  - Public Key Infrastructure.
- They have an increasing impact on various aspects of societal organisation.
- It’s fun!

What is a blockchain ?

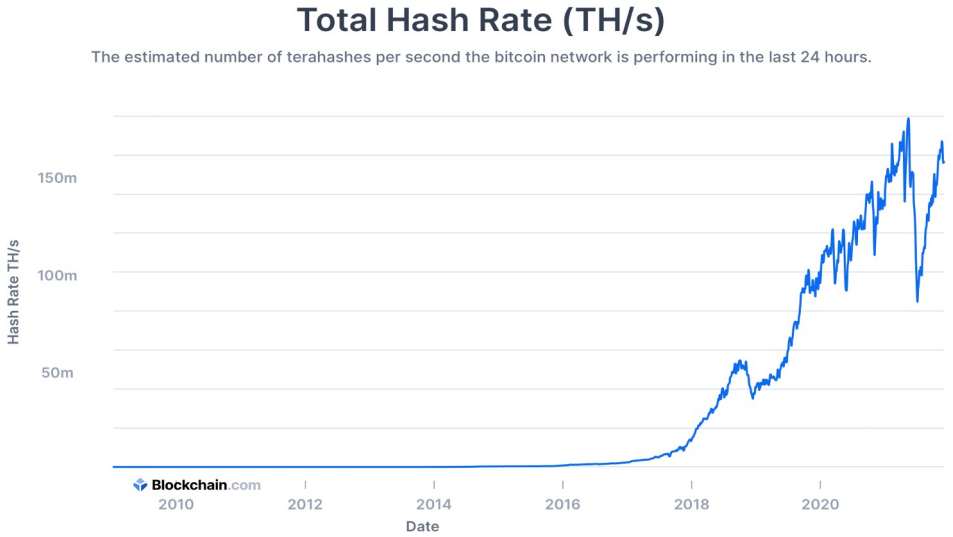
What is a blockchain ?

- A blockchain is a distributed database that satisfies a unique set of safety and liveness properties.
- Distributed ledgers use blockchain protocols as one means of implementation.
- Its concept was proposed as part of the Bitcoin system

The Bitcoin System

Is there something to learn from this successful deployment?

- A global IT system
- Offers a consistent database of transactions
- Maintainers are incentivized to service new transactions and keep the database consistent.
- Exhibited stability for over a decade of operation without a centralized setup or a single supporting organization.
- Participation grew exponentially (at times) since launch



How Bitcoin Blockchain Protocol Works: The never-ending book parable

A book of transactions

- Anyone can be a scribe and produce a page.
- New pages are produced indefinitely as long as scribes are interested in doing so.
- Each new page requires some effort to produce.

Importance of consensus

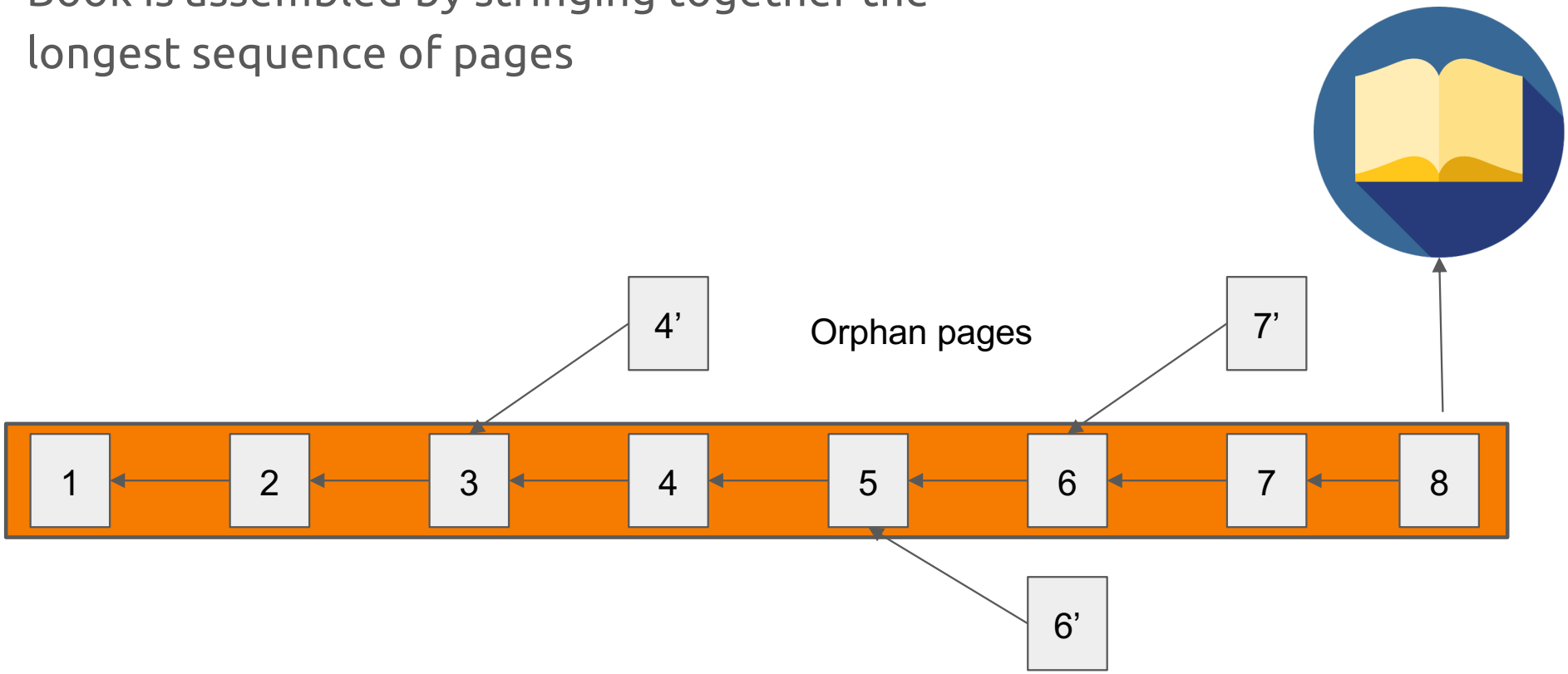
- If multiple conflicting books exist, which is the “right one” ?

Choosing the correct book ?

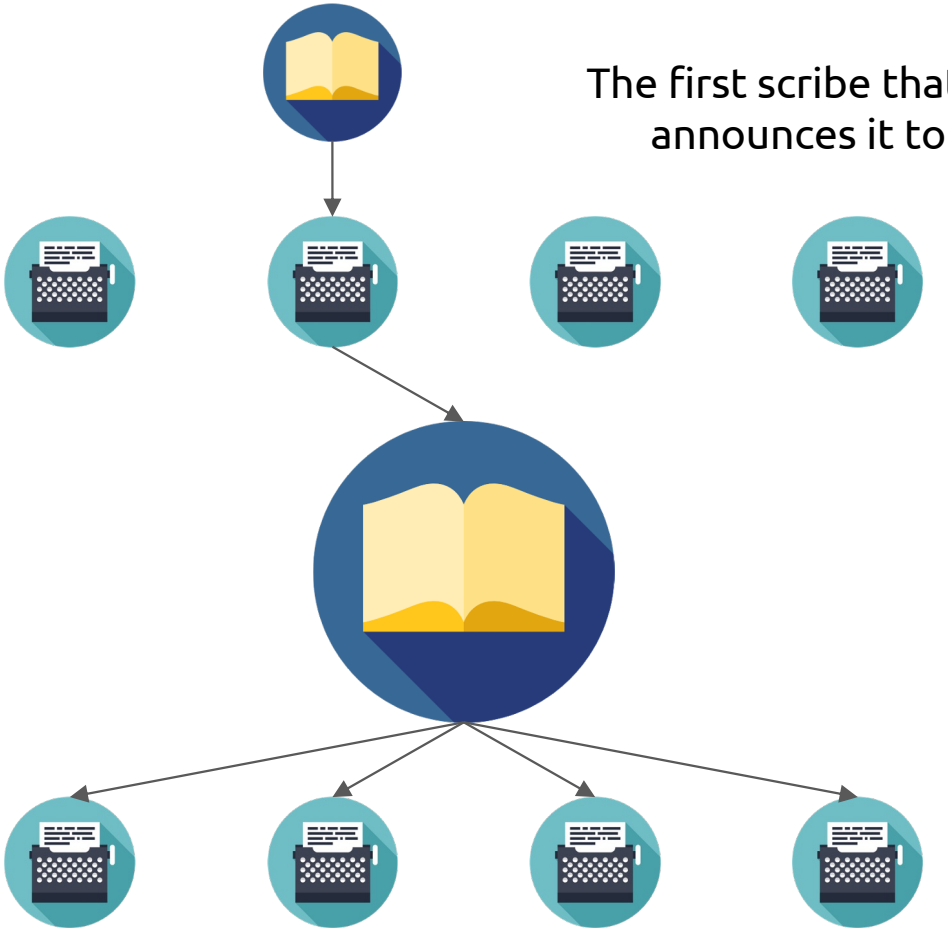
The correct book to work on & refer to is the book with the most pages. If multiple exist, just pick the first you heard.

Assembling the current book

- Each page refers only to the previous one
- Book is assembled by stringing together the longest sequence of pages



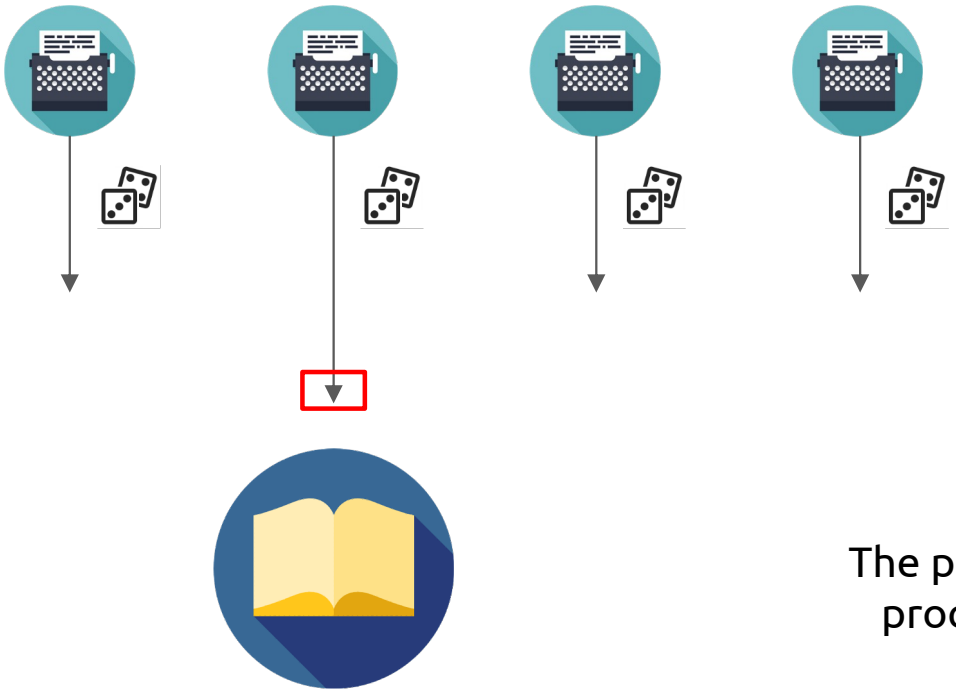
Rules of extending the book



The first scribe that discovers a page announces it to everyone else

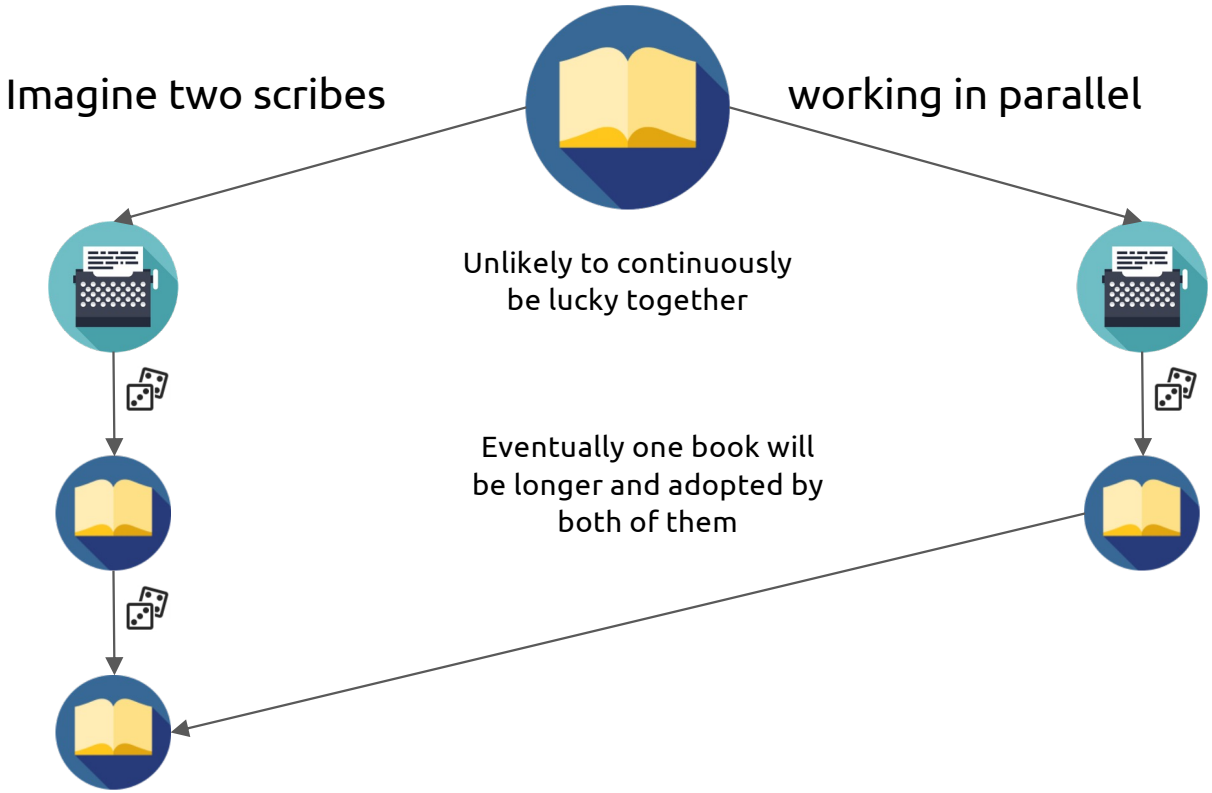
Effort is needed to produce a page

Equivalent to: each page needs a special combination from a set of dice to be rolled.



The probabilistic nature of the process is paramount to its security

The benefits of randomness

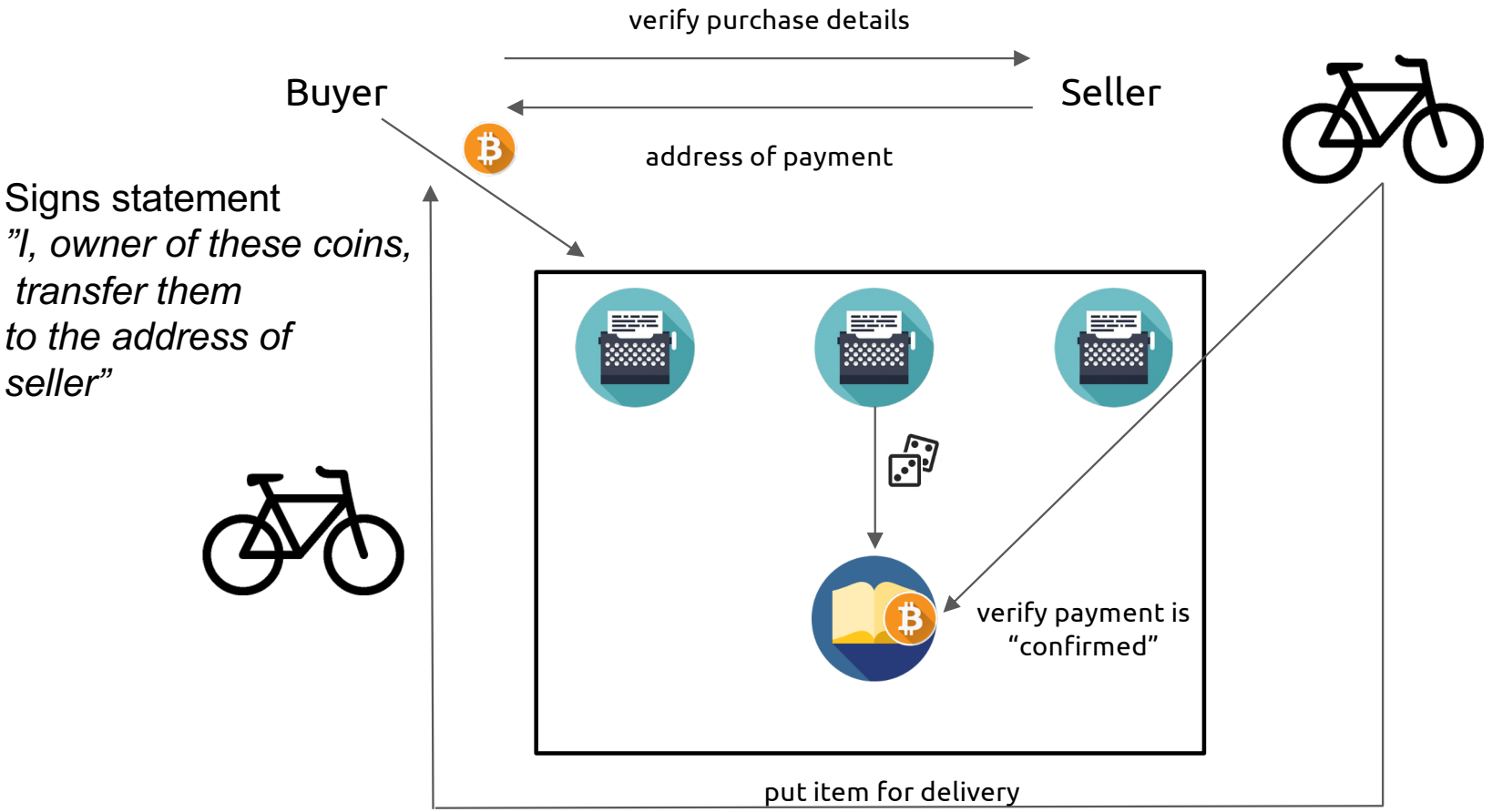


Symmetry Breaking

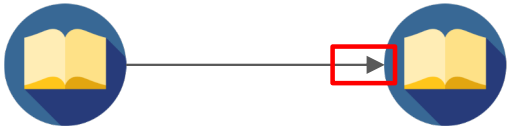
Being a scribe

- Anyone can be a scribe for the book.
  - Being a scribe is rewarded: they are allowed to make a special entry that carries their reward (this is how new Bitcoin is minted) 
- As long as one has a set of dice.
  - (these relate to the resources possessed by a party)
- The more dice one has, the higher the likelihood to produce the winning combination to make a page.

Using the book for monetary transactions



Parable & Reality

	The “blockchain”
	“Miners” / Computer systems that organize transactions in blocks
	Solving a cryptographic puzzle that is moderate hard to solve
	Using a computer to test for a solution from a large space of candidate solutions

Questions to Consider

- How are pages created? Since the book is empty at the beginning, where do the money come from?
- How to prove ownership of a digital good?
- How is it possible to sign something digitally?
- How does a page can refer to the previous page without ambiguity?

Questions to Consider

- How are pages created? Since the book is empty at the beginning, where do the money come from? - Proof-of-Work / Proof-of-Resource
- How to prove ownership of a digital good? – Public-key cryptography
- How is it possible to sign something digitally? - Digital signatures
- How does a page can refer to the previous page without ambiguity? - Hash functions

Hash Functions

- An algorithm that produces a fingerprint of a file.
- what are the required properties (traditionally):
  a. Efficiency
  b. A good spread for various input distributions.
- What are Security/Cryptographic considerations

$$\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^\lambda$$

Collision resistance

Collision attack

Find $x, y : \mathcal{H}(x) = \mathcal{H}(y)$

Second pre-image attack

Find $y : \mathcal{H}(x) = \mathcal{H}(y)$

For given x

Birthday paradox

- How many people should be in a room so that the probability that two of them share a birthday becomes larger than 50% ?

Paradox explained

n possible dates $\qquad$ k people

$$\Pr[\neg Col] =$$

$$\frac{n}{n}\frac{n-1}{n}\frac{n-2}{n}\cdots\frac{n-k+1}{n} = \prod_{\ell=1}^{k}(1-\frac{\ell}{n})$$

$$\leq \exp(-\frac{1}{n}\sum_{\ell=1}^{k}\ell) = \exp(-k(k+1)/2n)$$

$$\Pr[Col] = \frac{1}{2} \Rightarrow k \approx 1.177\sqrt{n}$$

What do we learn about collision finding?

Describe an algorithm that finds collisions taking advantage of the Birthday paradox.

Pre-image attack

Given $\mathcal{H}(m)$ $m \in \{0,1\}^t$

Find an element of $\mathcal{H}^{-1}(\mathcal{H}(m))$

Generic algorithm tries all possible candidates
Complexity:

Pre-image attack

Given $\mathcal{H}(m)$ $m \in \{0,1\}^t$

Find an element of $\mathcal{H}^{-1}(\mathcal{H}(m))$

Generic algorithm tries all possible candidates
Complexity: ? 2^t worst case

One-way functions

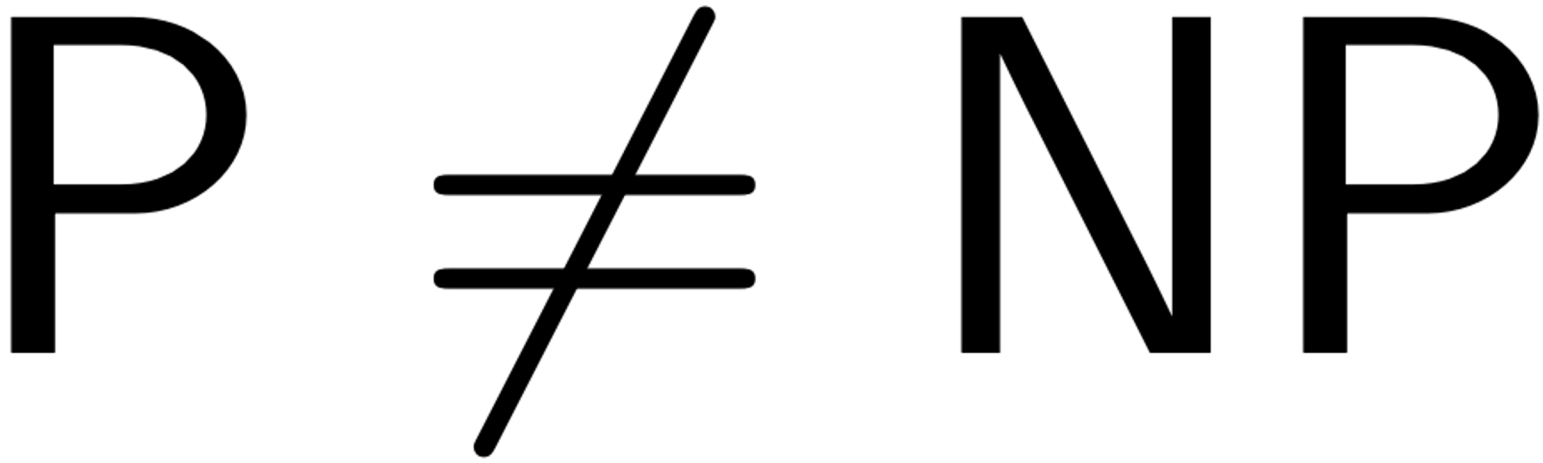
$$f : X \to Y$$

easy : given x find $f(x)$

hard : given $f(x)$ sample $f^{-1}(f(x))$

Do one-way functions exist?

Relates to most important open question in computer science right now:



Hash function instantiations

- **Retired.** MD5, SHA1.
- **Current.** SHA2, SHA3, available for 224,256,384,512 bits fingerprints.
- **Bitcoin.** Uses SHA2 with 256 bits output, SHA-256.

Digital Signatures

- Can be produced by one specified entity.
- Can be verified by anyone (that is suitably “equipped" and “initialised”).
- Cannot be forged on a new message even if multiple signatures have been transmitted.

Digital Signatures

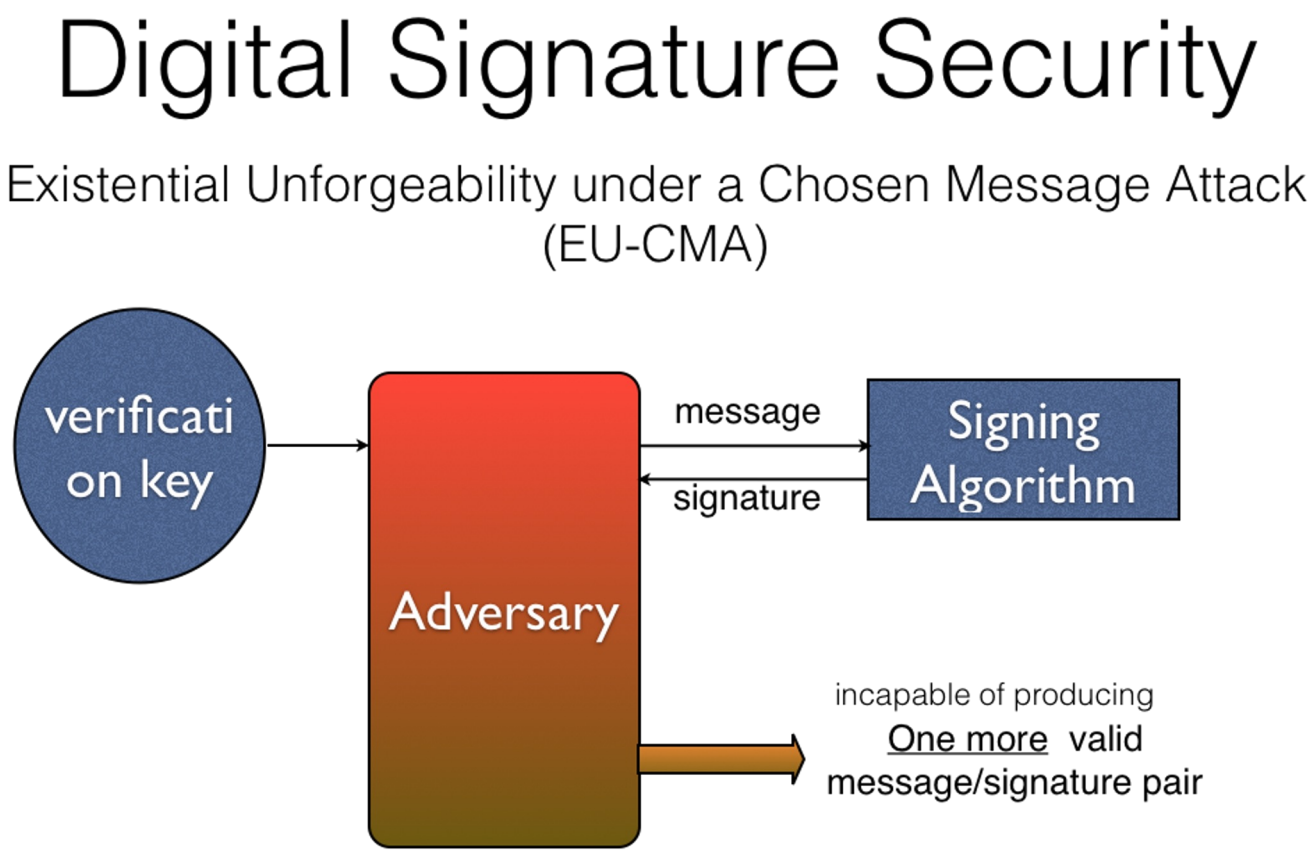
Three algorithms (**KeyGen**, **Sign**, **Verify**)

KeyGen : takes as input the *security parameter.*
returns the signing-key and verification-key.

Sign : takes as input the *signing-key* and
the *message* to be signed and
returns a signature.

Verify : takes as input the *verification-key,*
a *message* and a *signature* on the message and
returns either True or False.

Digital Signature Security



Constructing Digital Signatures

- Major challenge:
  - what prevents the adversary from learning how to *sign* messages by analyzing the *verification-key?*
- Exercise: construct a digital signature based on a hash-function that is one-time secure (i.e., it is secure for signing only a single message)

Digital Signature Implementations

- Based on the RSA (Rivest Shamir Adleman), one way trapdoor function (with hardness that relates to the factoring problem).
  - The RSA algorithm
- Based on the discrete-logarithm problem.
  - the DSA algorithm
- Bitcoin. Uses ECDSA, a DSA variant over elliptic curve groups. Since 2021, Schnorr signatures were introduced (also over elliptic curve groups)

Proof of Work

Objective: given some *data* ensure that some amount of work has been invested for them.

```
int counter;
counter = 0
while Hash(data, counter) > Target
        increment counter
return counter
```

In this case: proof-of-work of *data* equals to a value *w* with the property Hash(*data*, *w*) <= Target

(Informal) Properties: efficient verification, no computational shortcuts (i.e., independent of algorithm that computes it complexity is proportional to Target), independence for symmetry-breaking.

Proof-of-Work Algorithms

Hashcash (as in previous slide)

Memory hardness

- ASIC resistance (ASIC = application specific integrated circuit).
- A number of algorithms proposed: scrypt, argon, progpow

Recent years: much more attention to *proof of stake* algorithms

What general IT problem did Bitcoin solve?

How do we scale an IT service globally?

Scaling IT Services - Two Classical Models

Federation

Centralization

Is there another way?

Decentralization

Image from Wikipedia: Bust of Satoshi Nakamoto, creator of bitcoin, erected in the Graphisoft Park in Budapest

A motivating use-case: The Software Only Launch (SOL) problem for bottom up system launch

How to launch a system “bottom up”

Distribute Software via public channels
+
Launch date

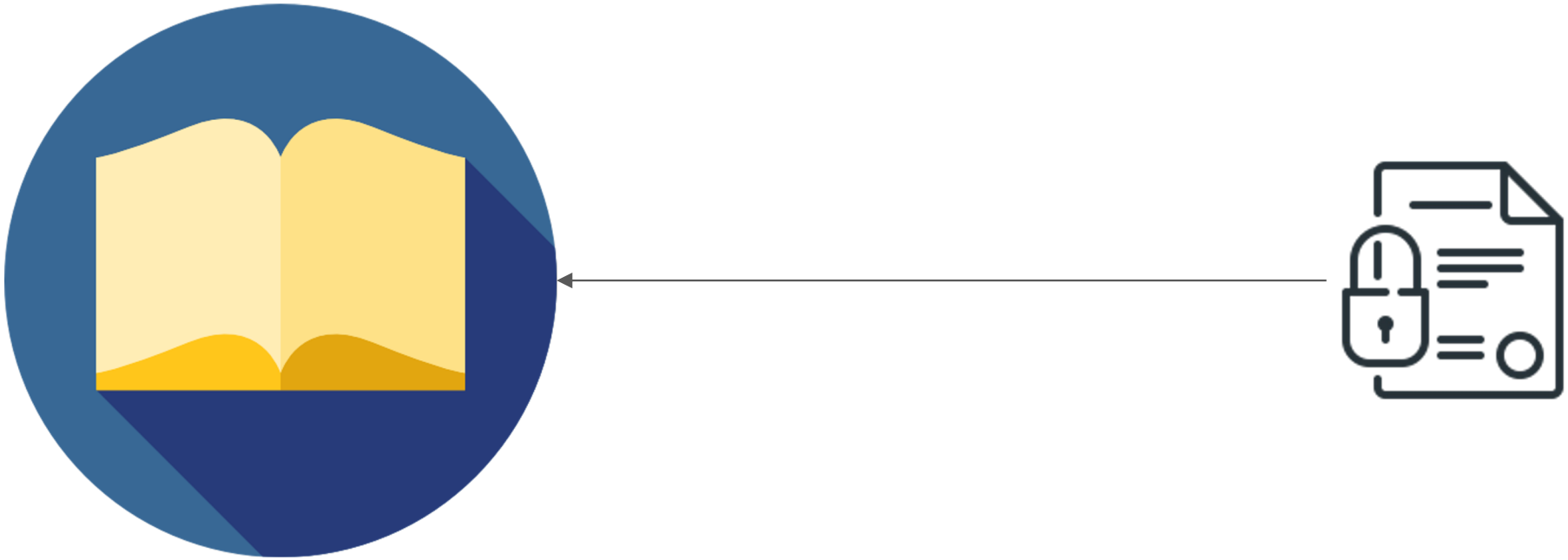
On launch date, some people run the software and bootstrap the system in a “bottom up” fashion

Image credit Bing Dall-E

Can we generalize Bitcoin’s approach?

- in order to solve similar problems related to scaling IT systems

Smart contract

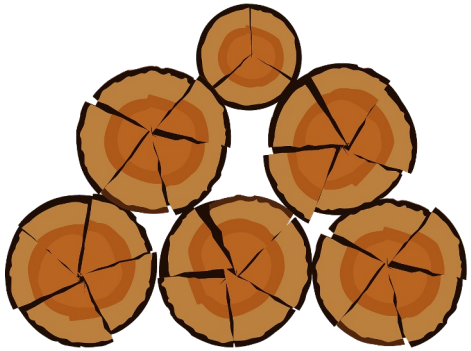
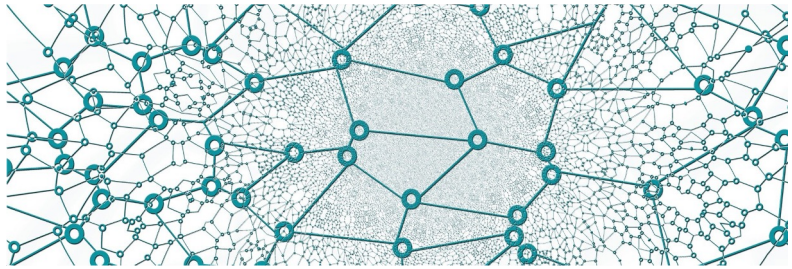


From Money to Smart Contracts

- Since we have created the book, why stop at recording monetary transactions, as bitcoin does?
- We can encode in the book's pages arbitrary relations between persons.
- Furthermore, scribes, can perform tasks such as verifying that stakeholders comply to contractual obligations … and take action if they do not.

Resource Based Systems

Aggelos Kiayias: Decentralizing Information Technology: The Advent of Resource Based Systems, **https://tinyurl.com/decentralizingit**

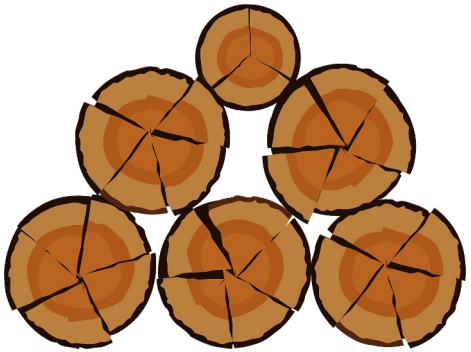
Cryptography	Resource based participation 	Tokenomics 
Distributed / Crypto Protocol Design	Decentralized Service Provision 	Rewards Sharing 

Main Technology

Economics

Game Theory

Resource Based Participation



- Relies on a “Proof of Resource” / “PoX”
  - X=Work (computational power)
    - Cost proportional to resource
    - Issues: hardware optimizations distort resource space; e-waste; energy waste.
  - X=Stake
    - Cost independent of resource
    - Issues: “nothing at stake”
- Other examples:
  - X=space, X=elapsed time [Intel SGX]
- System is not run by *n* parties, but by the entities capable of issuing PoX
  - The set of such parties may fluctuate and change significantly over time.

Resource examples

System	Resource
Bitcoin	Computational Power
Ethereum 1.0	Computational Power
Chia	Storage
Ethereum 2.0	Stake
Cardano	Stake
Tezos	Stake
Polkadot	Stake

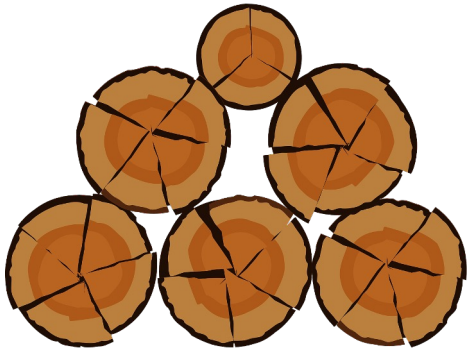
Resource Based Systems

Aggelos Kiayias: Decentralizing Information Technology: The Advent of Resource Based Systems, **https://tinyurl.com/decentralizingit**

Cryptography + Domain Technology

Economics

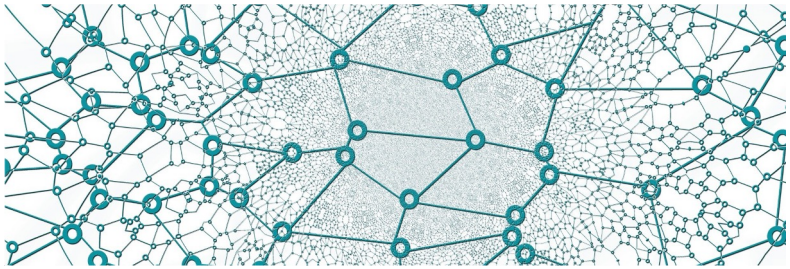
Resource based participation



Tokenomics

Distributed / Crypto Protocol Design

Decentralized Service Provision

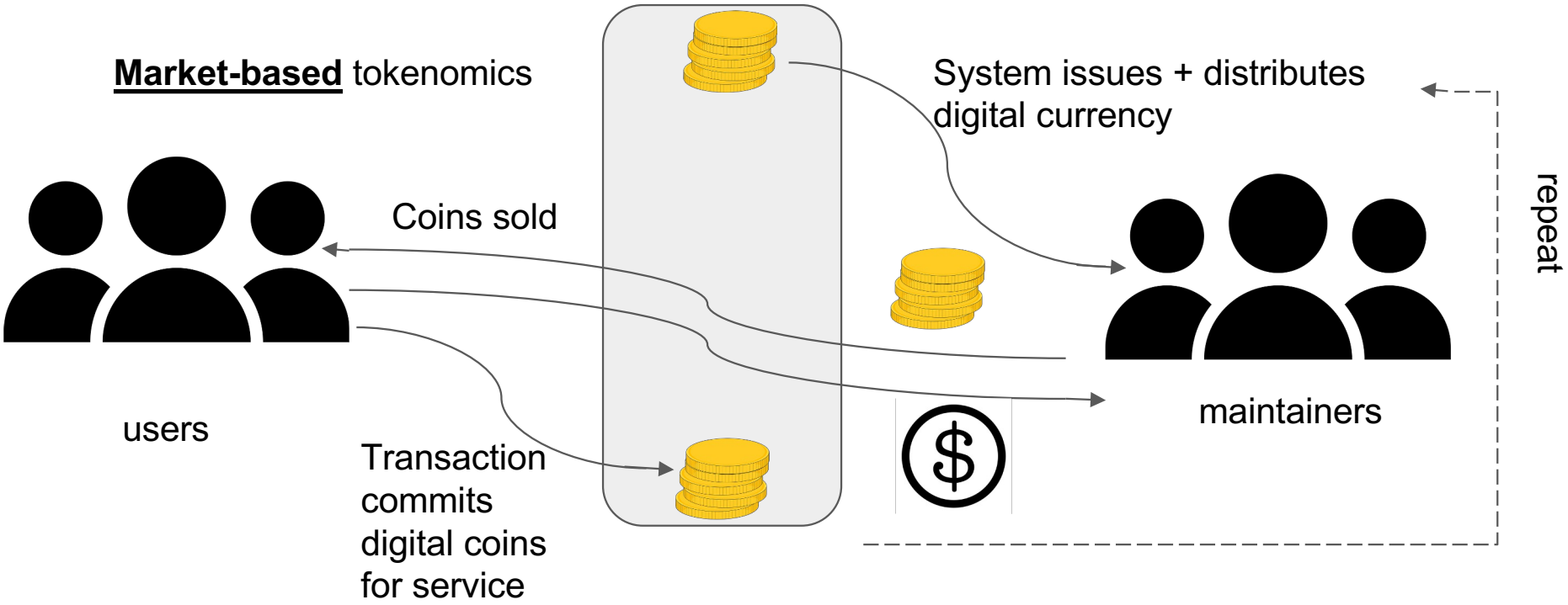


Game Theory

Rewards Sharing

Tokenomics

- Digital coins issued by system.
- Objective : influence utility so that engaging as a maintainer is attractive.



Tokenomics

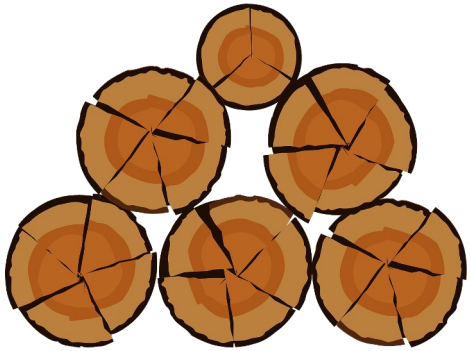
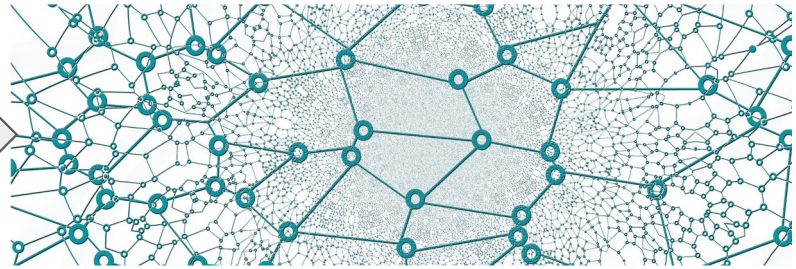
- The demand for service and the way this impacts the cost of system maintenance is countered by the utility increase due to the utilization of the system's digital asset.
  - E.g., in market-based tokenomics, the maintainers (collectively as one) make enough $$ by selling the system's digital currency so that engaging in maintenance at a suitable level of quality of service remains attractive.
  - Other types of tokenomics may be possible (e.g., reputation).

Resource Based Systems

Aggelos Kiayias: Decentralizing Information Technology: The Advent of Resource Based Systems, **https://tinyurl.com/decentralizingit**

Cryptography + Domain Technology

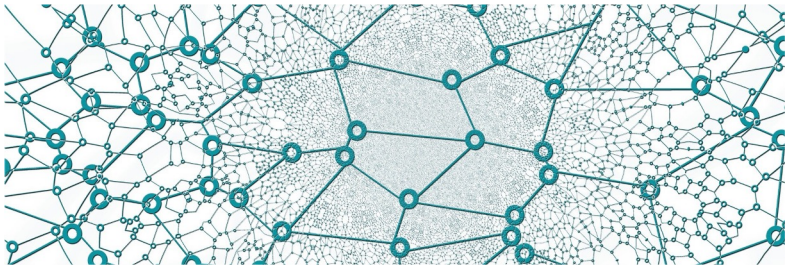
Distribu… Proto…

Resource based participation 	Tokenomics 
Decentralized Service Provision 	Rewards Sharing 

Economics

Game Theory

Decentralized Service Provision



- Implement service over an open network in a decentralized manner.
  Challenges:
  - Denial of Service - DoS Attack Mitigation [market based tokenomics : use transaction fees]
  - Consistency. System transactions are processed consistently network-wide.
  - Liveness. System transactions are processed in a timely manner without censorship.
- Facilitate fair recording of system maintainer efforts => performance metrics

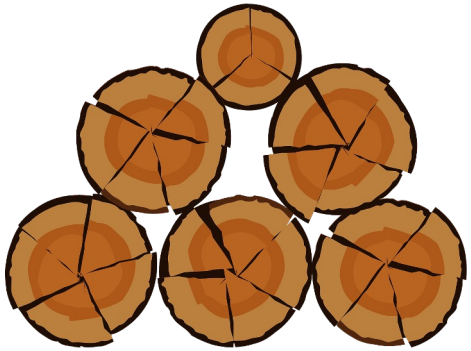
Resource Based Systems

Aggelos Kiayias: Decentralizing Information Technology: The Advent of Resource Based Systems, **https://tinyurl.com/decentralizingit**

Cryptography + Domain Technology

Economics

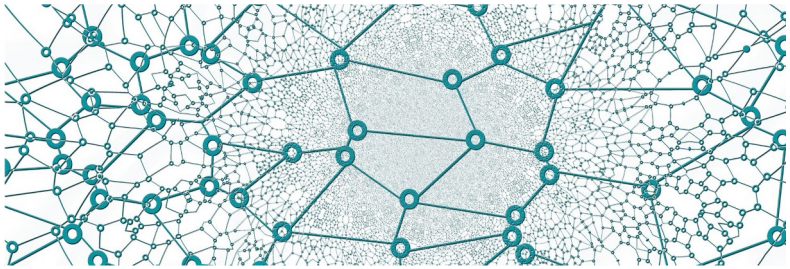
Resource based participation



Tokenomics

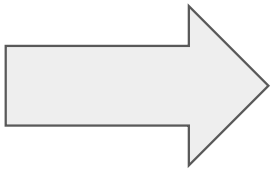
Distributed / Crypto Protocol Design

Decentralized Service Provision



Game Theory

Rewards Sharing



Rewards Sharing

- How do you turn a set of self-organizing resource holders to a set of well functioning system maintainers?
  - … without anyone centrally taking decisions…

- The goal is to distribute rewards fairly at regular intervals
  - Action-based or epoch-based
- Rewards sharing should support the desired properties of the system at an equilibrium
- We want the system to converge to “good” equilibria