Principles of Blockchains

https://web3.princeton.edu/principles-of-blockchains/

Professor Pramod Viswanath

Princeton University

Lecture 1. What are Blockchains?

Blockchains are decentralized digital trust platforms

This is a mouthful, so let us unpack it.

Trust

Human success is based on flexible cooperation in large numbers. This requires trust

Evolution of Trust over human history

Platform Economy

2023 September
Top US companies by market cap

1.	Apple	$2.9 T
2.	Microsoft	$2.5 T
3.	Alphabet	$1.7 T
4.	Amazon	$1.4 T
5.	Nvidia	$1.2T
6.	Tesla	$0.8 T

2011
Top US companies by market cap

1.	Exxon	$417 B
2.	Apple	$321 B
3.	Chevron	$215 B
4.	Microsoft	$213 B
5.	IBM	$207 B
6.	Walmart	$204 B

A Decentralized Platform?

- A decentralized Dropbox, eBay, Instagram?
- Incentives aligned with consumers and resource providers?
- No need for a trusted middle party?

Such is the siren song of blockchains.

Bitcoin: the original blockchain

Cryptocurrency

medium of exchange and store of value

Born during the **2008 Financial Crisis**

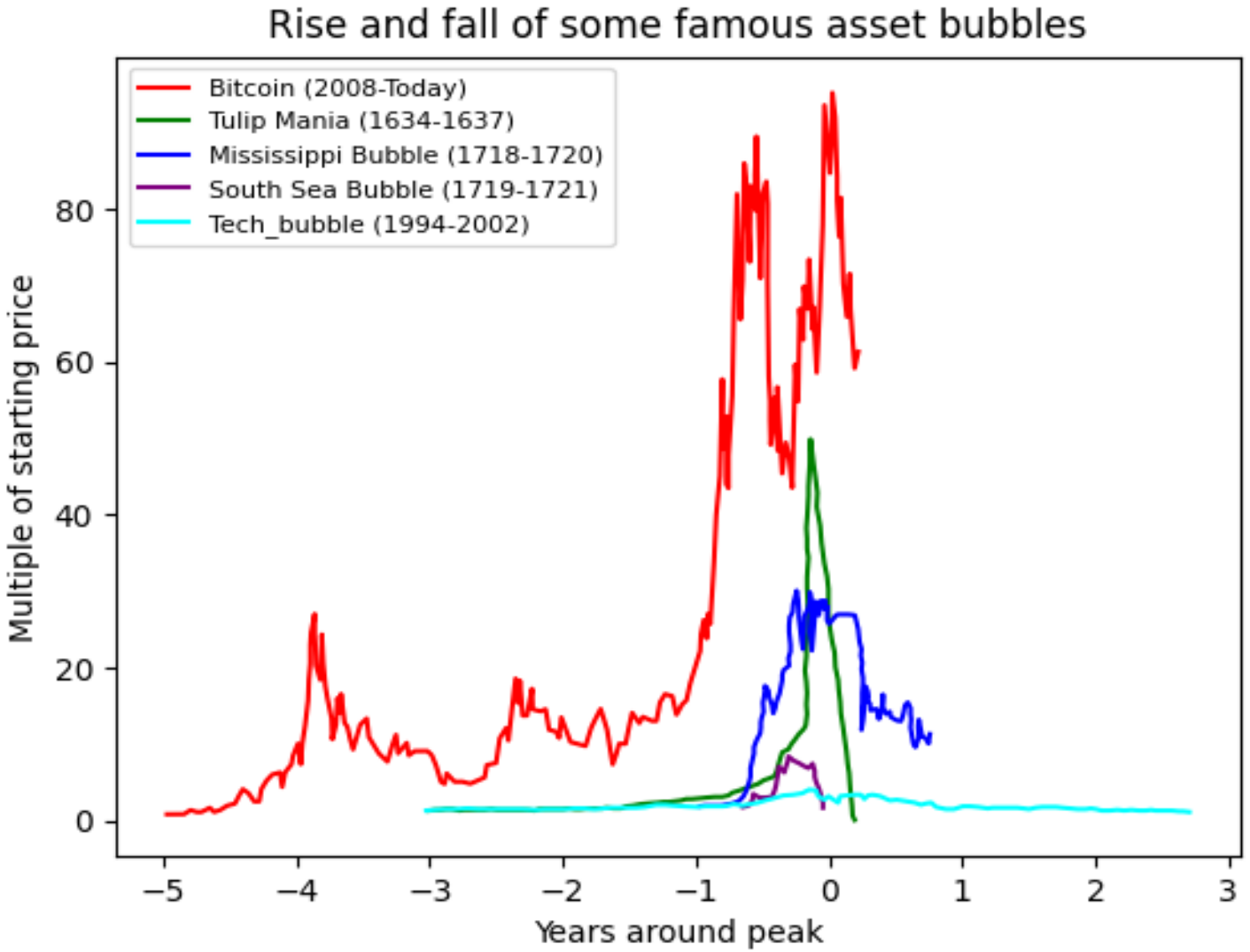
Anonymous inventor

pseudonym: Satoshi Nakamoto

Very secure

no attacks, has been live continuously

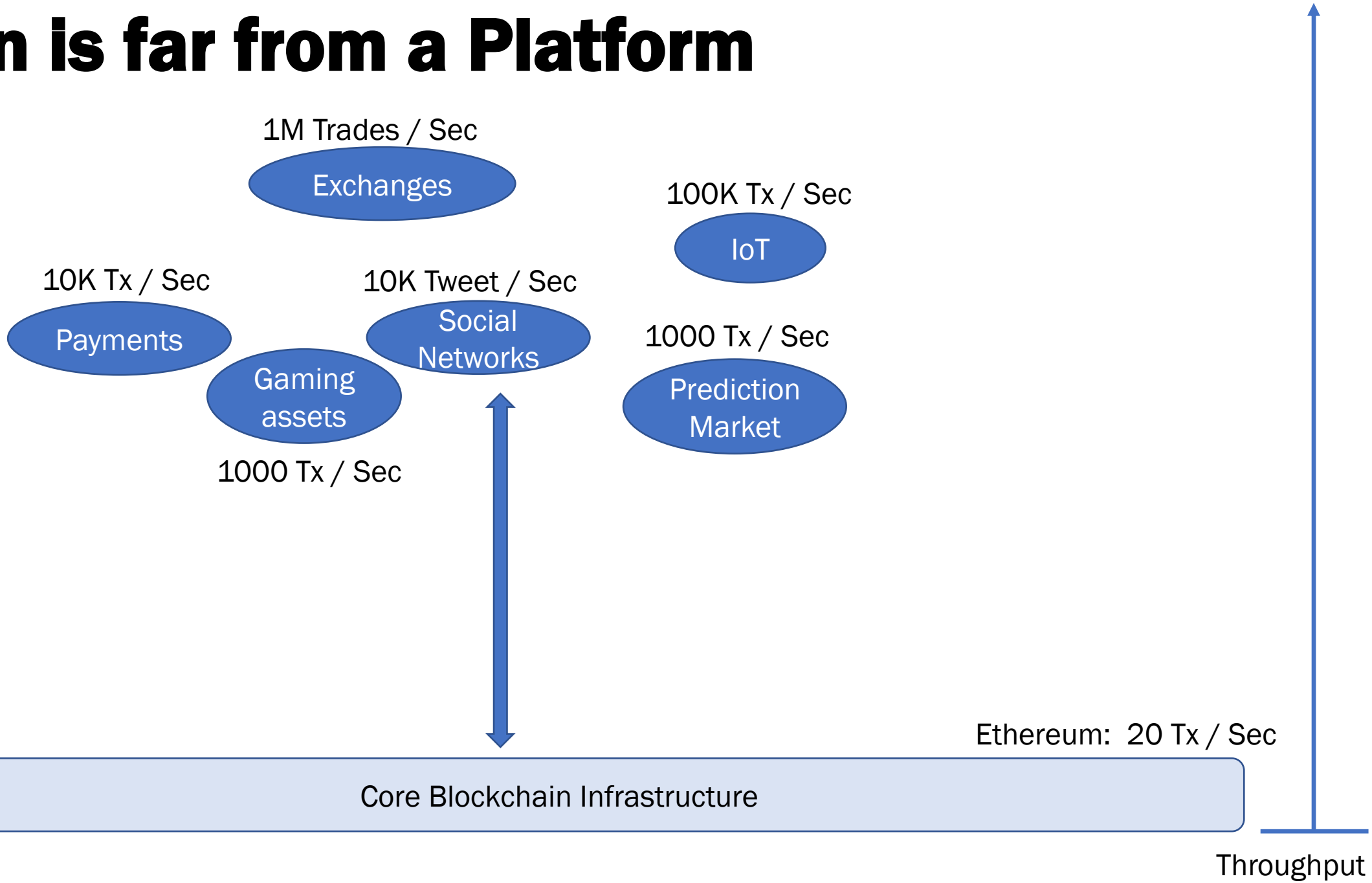
Bitcoin and Bubbles



Bitcoin performance

1. Security – 50% adversary
2. Transaction throughput – 7 tx/s
3. Confirmation Latency – hours
4. Energy consumption – medium size country
5. Compute – specialized mining hardware
6. Storage – everyone stores everything
7. Communication – everyone tx/rx everything

Bitcoin is far from a Platform



Building Block of Blockchains

- Platforms are applications built on networked computers

- Basic building block: **Decentralized Computer**

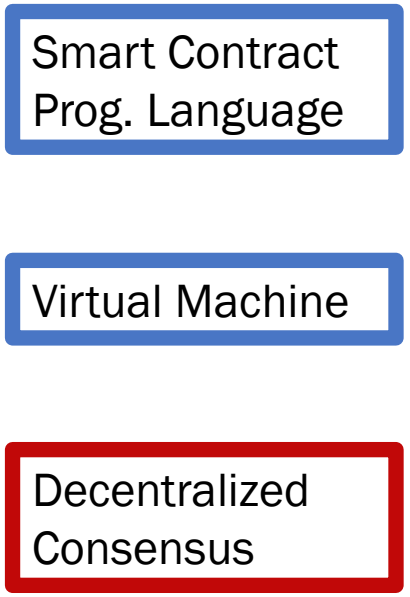
1. Multiple untrusted computers interacting with one another, forming **consensus** on an ordered list of instructions

2. A **virtual machine** interprets the instruction set

3. A programming language and a corresponding compiler provide a forum for **decentralized applications (dApps)**

Technical Components

- Decentralized Computer
  - Cryptographic data structures
  - Disk I/O and Database management
  - Memory management
  - Operating systems
  - Peer to peer networking
  - Consensus and distributed algorithms
- Virtual Machine
  - Reduced instruction set, incentives
  - General purpose programming language



Nearly all aspects of Computer Science

Technical Challenges

- **Permissionless**
  - Anyone can meaningfully participate
  - **Challenge**: need to prevent spam, bad actors
- **Dynamic availability and safety**
  - Consensus holds with dynamic participation of enough participants
  - **Challenge**: need to work even with bad actors
- **Cryptography**
  - Provides basic tools to address both design goals
  - **Challenge**: “The trouble is, the other side can do magic too, Prime Minister.”

Principles of Blockchains

- This course presents the **design space of blockchains**
  - **Principles** of good blockchain design choices
  - **Full-stack view**

- Pre-requisite: maturity with nearly all aspects of computer science

- Concretely: basic background in algorithms, probability, systems programming

Two types of Principles

- **Conceptual** Principles
  - Algorithmic designs
  - Byzantine resistance
  - Security analysis
  - Mechanism design and incentive compatibility

- **Engineering** Principles
  - Modular software stack
  - Software engineering
  - Integration and composition of different modules – secure, yet efficient

Course begins with Bitcoin

- We start with an in-depth view of the **design of Bitcoin**
  - The focus allows us to see the interacting components of the blockchain
  - Highlight the design constraints across the layers

- Bitcoin design is very simple and yet
  - remarkably secure, elegant in an engineering sense
  - security guarantees backed by sophisticated mathematics
  - Outstanding case study for a deep understanding of blockchains
  - Highlights **both conceptual and engineering principles**

Module 1. Bitcoin Blockchain

- Next four lectures: Cryptographic data structures, Consensus, Peer to Peer Networking, Transaction structure, Ledger state management
- Two lectures: mathematical security guarantees of Bitcoin
- Implementation-intensive: students implement a full-stack Bitcoin client

Module 2. Scaling Blockchain

- Adapting the Bitcoin design to scale its performance

- Scaling:
  - Throughput
  - Latency
  - Computation, Storage
  - Energy
  - Layer 2 scaling

The resulting blockchain designs are at the heart of many popular cryptocurrency platforms: Avalanche, Cardano, Solana, Polygon

Module 3. Beyond Bitcoin

- Incorporating features absent in the Bitcoin design

1. Finality
2. Privacy
3. Connecting blockchains: Bridges
4. Importing data into blockchains: Oracles

The resulting blockchain designs are at the heart of many popular cryptocurrency platforms: Zcash, ChainLink

Logistics: course content

Canvas: Access for registered students

Website for the course:
https://web3.princeton.edu/principles-of-blockchains/

Assignments and Projects on Github:
https://github.com/Blockchains-Princeton/COS-ECE470-fa2023

Communication: Instruction from staff via email
Students can post on Ed, Discord and Github

Office hours: TW 5-6pm - EQuad J323
W 8-9pm - Zoom

Logistics: Grading Rubric

- Programming assignments
  Bitcoin client project - 64%
- Homework assignments - 20%
- Final project/paper - 10%
- Groups of three
- In-class participation - 6%

Logistics: programming language

Rust

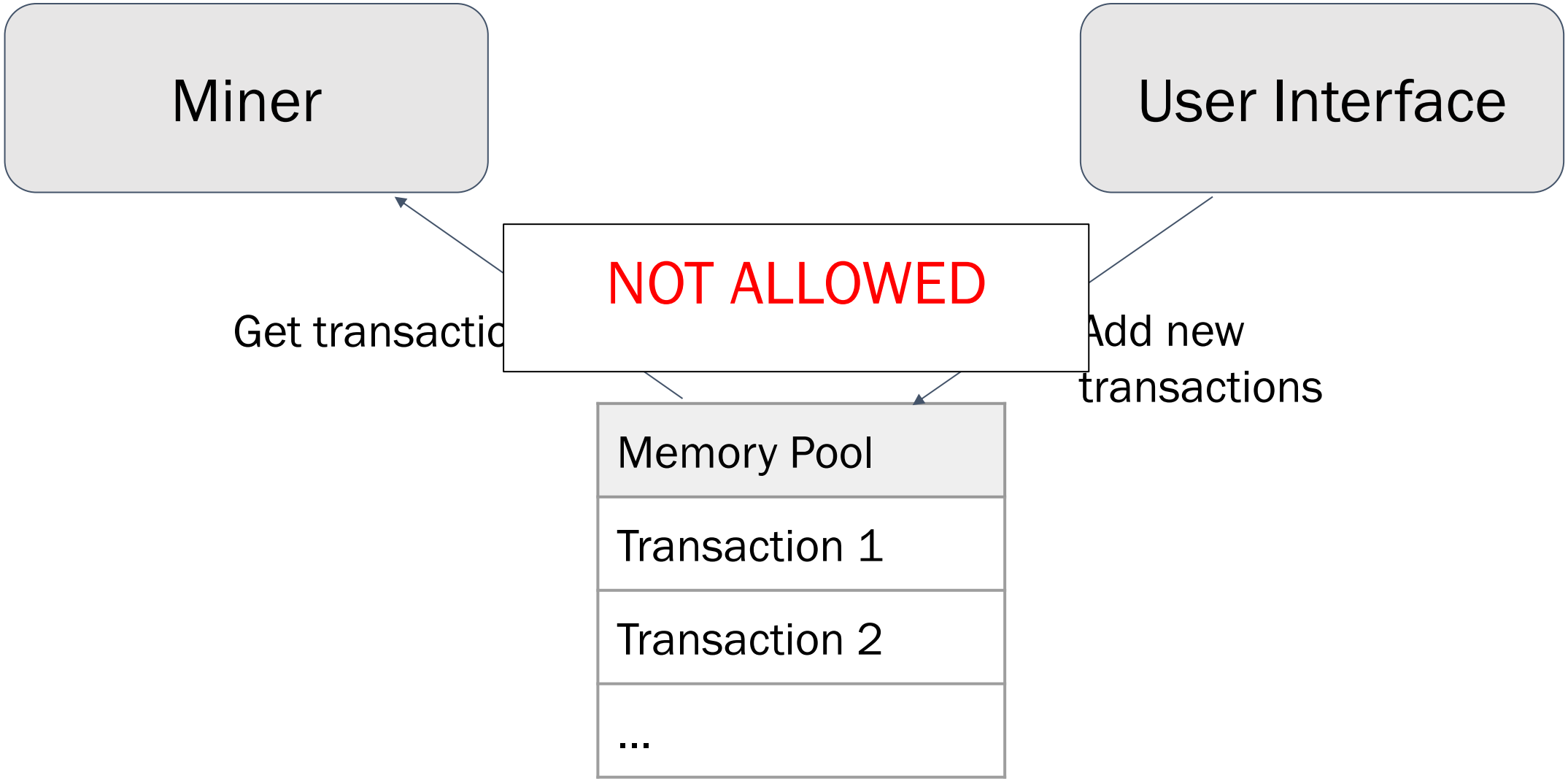
What is Rust?

Rust is a compiled language just like C/C++. Rust code is compiled to executable binary.

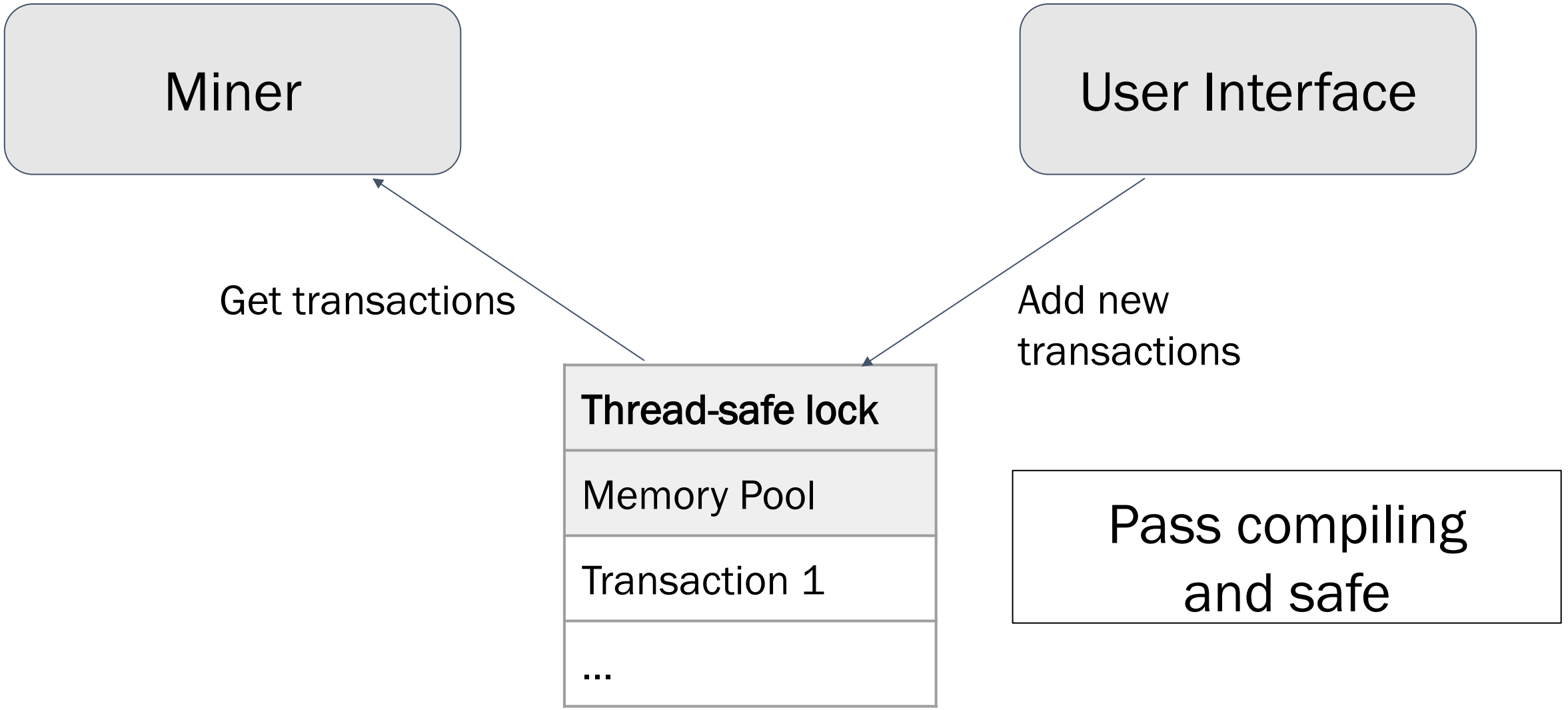
Reliability: Rust’s type system and ownership model guarantee memory-safety and thread-safety.

Programmers eliminate many classes of bugs at compile-time, where the compiler messages are used to eliminate bugs and make the program more reliable.

www.rust-lang.org



Rust compiler doesn’t allow more than one owner (Miner & UI) of an object (Memory Pool).



If an object is wrapped by a thread-safe lock, Rust compiler allows more than one owner (Miner & UI).

Besides its reliability, Rust also provides good ...

efficiency as C/C++,

network support as Go and Python,

functional-style programming as Scala,

community support as many other popular lanuages.

Why is Rust chosen for blockchains?

High reliability makes blockchains reliable and secure.

Good efficiency makes blockchains scalable.

Network support makes communication easy-to-code.

Influential blockchain projects are using Rust:

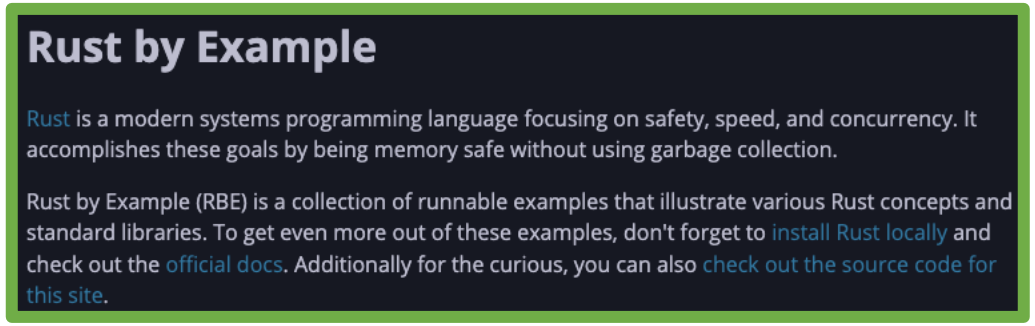
Programming assignment

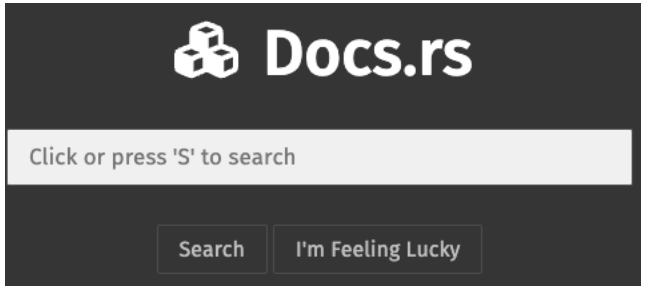
- Bitcoin client in Rust
- Due Thursdays 10pm
- Groups of 2 or 3
- Part 1 due September 14

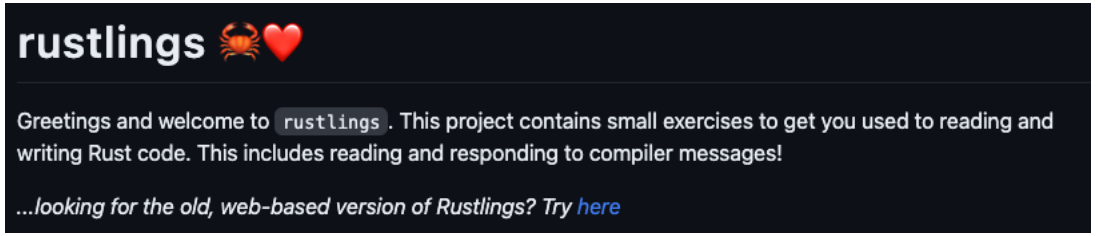
  https://github.com/Blockchains-Princeton/COS-ECE470-fa2023

  - Familiarity with Git
  - Installing Rust and Cargo
  - Learning Rust

Resources







Part 1 and 2 :

- Short exercises from rustlings
- Implementing crypto-primitives