CSE446: Blockchain & Cryptocurrencies

Lecture – 12: Ethereum - 1

Agenda

- Ethereum
- Motivations behind Ethereum
- Ethereum History
- Ethereum Components

Bitcoin review

- Each block is a list of transactions
  - Each transaction consumes one or more inputs;
  - Each transaction produces a set of outputs (amount + destination)
  - Input consumption has conditions (e.g., valid script, typically enforcing valid signature)

Bitcoin review

- Each block is a list of transactions
  - Each transaction consumes one or more inputs;
  - Each transaction produces a set of outputs (amount + destination)
  - Input consumption has conditions (e.g., valid script, typically enforcing valid signature)

- In practice, each input/output has script:
  - ScriptPubKey (outputs), ScriptSig (inputs)

Bitcoin review

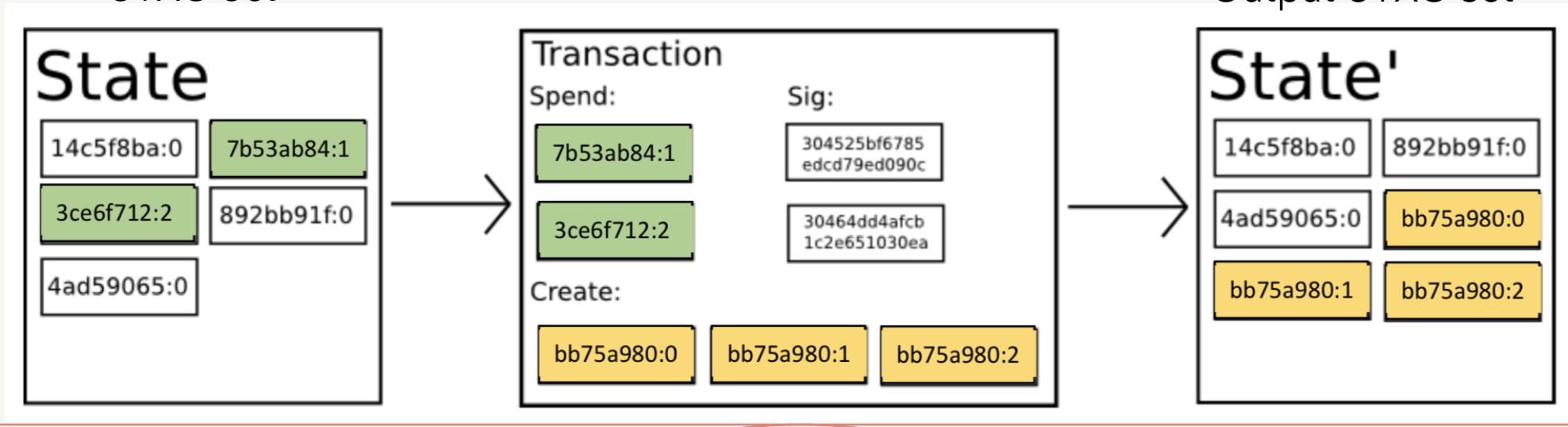
- Bitcoin script allows us to attach conditions to payments
  - However, script is deliberately limited
  - Highly limited access to global state data (chain state)
- Let's consider each Bitcoin transaction as a state transition function
- What is a state transition function?
  - A machine remains in a state (input state)
  - An instruction (input) causes the machine to do something (some operations)
  - The operations generates an output (output state)

Bitcoin review

- For bitcoin
  - What is the input state?
  - What is the output state?
  - What does a Transaction (an input) do to the state?

Bitcoin review

- Input state: list of coins available for spending (UTXO set)
- Transaction: set of instructions for updating the UTXO set
- Output state: Transfer of Bitcoin to one or more accounts updating the UTXO set



Smart-contracts

- Smart contracts were first proposed in the early 1990s by Nick Szabo, who coined the term
- A smart-contract is a computer program or a transaction protocol that is intended to automatically execute, control or document legally-relevant events and actions according to the terms of a contract or an agreement
- The objectives of smart contracts are the reduction of
  - need for trusted intermediaries
  - arbitration costs
  - fraud losses
  - malicious and accidental exceptions

Smart-contracts

- Can Bitcoin be used as a smart-contract platform?
- Yes, Bitcoin script is a ‘contract’ in the sense that it provides programmable conditions for redeeming a coin
- However, conditions are highly limited
- Can we use Bitcoin to
  - pay out a coin iff (if and only if) a user has a signing key?
  - implement a second currency/asset?
  - pay out a coin iff a candidate wins the US election?
  - pay out a coin iff a majority of users vote to invest in a service?

Smart-contracts

- Can Bitcoin be used as a smart-contract platform?
- Yes, Bitcoin script is a ‘contract’ in the sense that it provides programmable conditions for redeeming a coin
- However, conditions are highly limited
- Can we use Bitcoin to
  - pay out a coin iff (if and only if) a user has a signing key? ✅
  - implement a second currency/asset?
  - pay out a coin iff a candidate wins the US election?
  - pay out a coin iff a majority of users vote to invest in a service?

Smart-contracts

- Can Bitcoin be used as a smart-contract platform?
- Yes, Bitcoin script is a ‘contract’ in the sense that it provides programmable conditions for redeeming a coin
- However, conditions are highly limited
- Can we use Bitcoin to
  - pay out a coin iff (if and only if) a user has a signing key? ✓
  - implement a second currency/asset? ✓
  - pay out a coin iff a candidate wins the US election?
  - pay out a coin iff a majority of users vote to invest in a service?

Smart-contracts

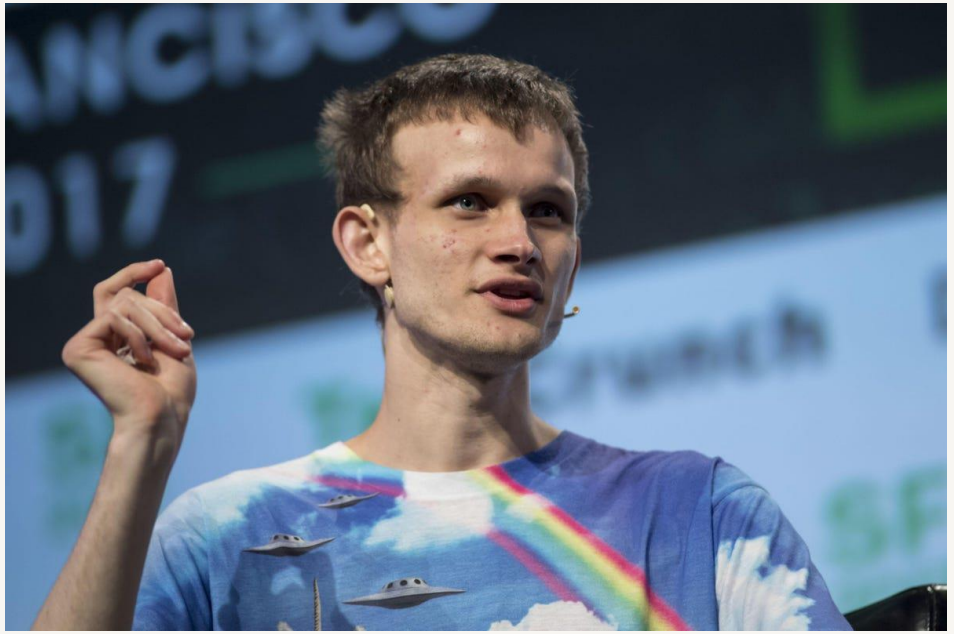
- Can Bitcoin be used as a smart-contract platform?
- Yes, Bitcoin script is a ‘contract’ in the sense that it provides programmable conditions for redeeming a coin
- However, conditions are highly limited
- Can we use Bitcoin to
  - pay out a coin iff (if and only if) a user has a signing key? ☑
  - implement a second currency/asset? ☑
  - pay out a coin iff a candidate wins the US election? ✗
  - pay out a coin iff a majority of users vote to invest in a service?

Smart-contracts

- Can Bitcoin be used as a smart-contract platform?
- Yes, Bitcoin script is a ‘contract’ in the sense that it provides programmable conditions for redeeming a coin
- However, conditions are highly limited
- Can we use Bitcoin to
  - pay out a coin iff (if and only if) a user has a signing key? ✅
  - implement a second currency/asset? ✅
  - pay out a coin iff a candidate wins the US election? ❌
  - pay out a coin iff a majority of users vote to invest in a service? ❌

Ethereum

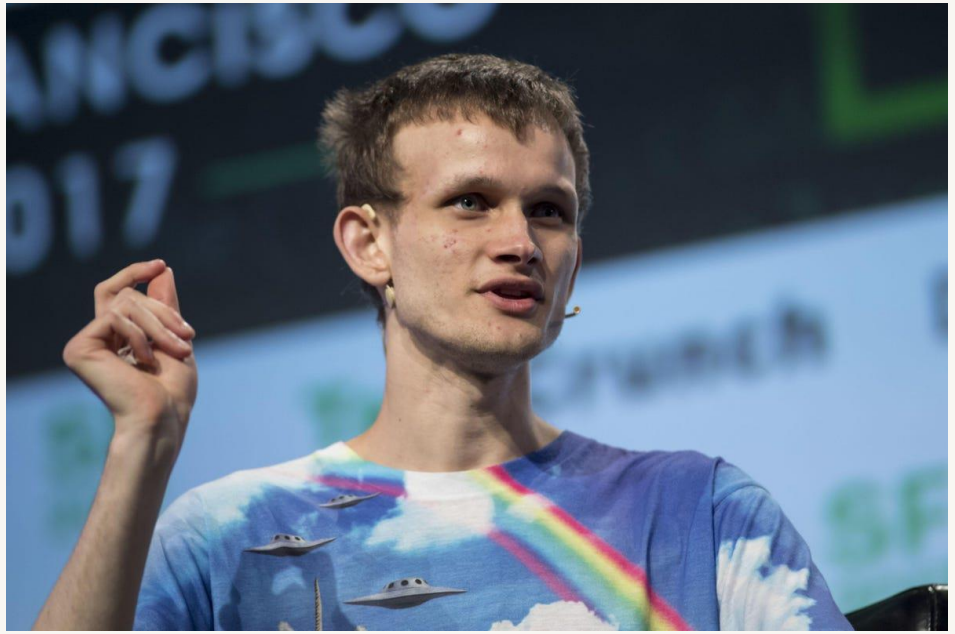
- Ethereum concept was proposed by Vitalik Buterin in 2013
- Basic idea: extend Bitcoin by adding Turing-complete scripting languages, with full access to chain state
- Ethereum is a computing platform on top of a blockchain
  - Equipped with a virtual machine called EVM (Ethereum Virtual Machine)
  - Based on a stack-based architecture with RAM, ROM and arbitrary storage



https://imageio.forbes.com/specials-images/imageserve/609034cec99cb743ece612fc/0x0.jpg?format=jpg&width=1200

Ethereum

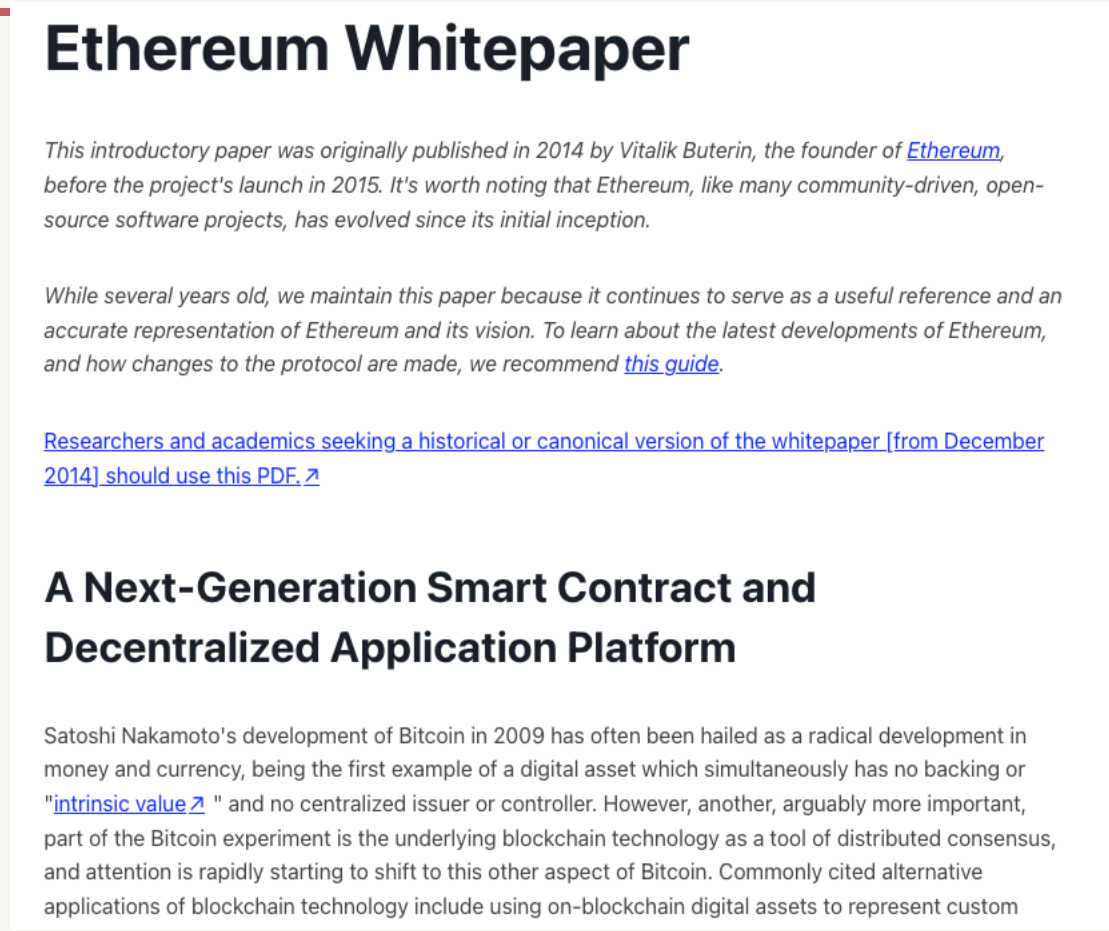
- Ethereum supports several new Turing-complete programming scripts/languages:
  - Solidity, Vyper and LLL
- Scripts run inside of the EVM, can call other scripts & each other
- Includes a native token (Ether/ETH) to pay for transactions
- Using these languages
  - Programs (smart-contracts) can be written to be executed in the blockchain
  - Data can be stored in the blockchain



https://imageio.forbes.com/specials-images/imageserve/609034cec99cb743ece612fc/0x0.jpg?format=jpg&width=1200

Ethereum whitepapr

- First draft was written by Vitalik Buterin himself (2013)
- Contains high level descriptions of Ethereum's core functionalities
- Living document and regularly updated by Ethereum core developers (not only Buterin!)
- Extensive summary of the Ethereum platform and technology



Ethereum Whitepaper

This introductory paper was originally published in 2014 by Vitalik Buterin, the founder of Ethereum, before the project's launch in 2015. It's worth noting that Ethereum, like many community-driven, open-source software projects, has evolved since its initial inception.

While several years old, we maintain this paper because it continues to serve as a useful reference and an accurate representation of Ethereum and its vision. To learn about the latest developments of Ethereum, and how changes to the protocol are made, we recommend this guide.

Researchers and academics seeking a historical or canonical version of the whitepaper [from December 2014] should use this PDF. ↗

A Next-Generation Smart Contract and Decentralized Application Platform

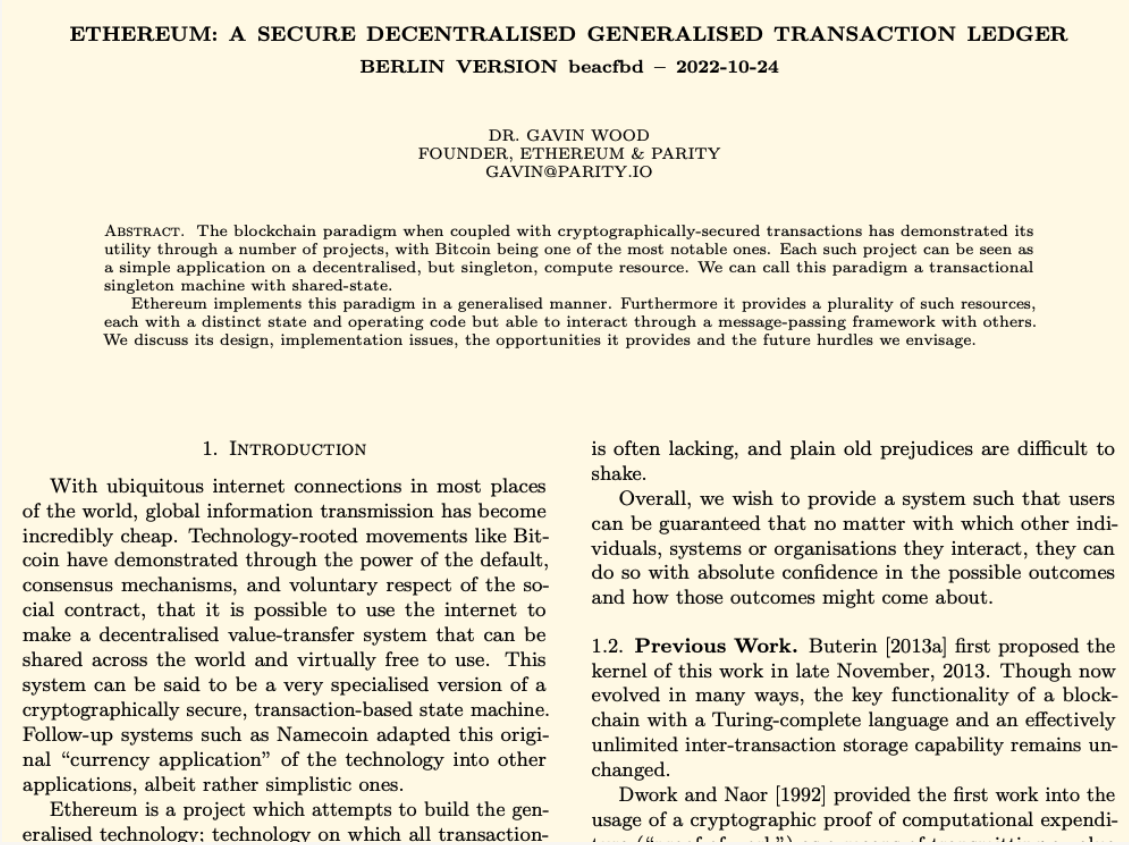
Satoshi Nakamoto's development of Bitcoin in 2009 has often been hailed as a radical development in money and currency, being the first example of a digital asset which simultaneously has no backing or "intrinsic value ↗ " and no centralized issuer or controller. However, another, arguably more important, part of the Bitcoin experiment is the underlying blockchain technology as a tool of distributed consensus, and attention is rapidly starting to shift to this other aspect of Bitcoin. Commonly cited alternative applications of blockchain technology include using on-blockchain digital assets to represent custom

07 Ethereum Basics - Gallersdörfer, U., Holl, P., & Matthes, F. (2020). "Blockchain-based Systems Engineering". Lecture Slides. TU Munich.

https://ethereum.org/en/whitepaper/

Ethereum yellowpaper

- Published in April 2014 by Dr. Gavin Wood
- Dr. Gavin Wood is still listed as the only author
- Defines the technical specification of Ethereum
- Very detailed, contains mathematical function definitions and byte code mappings
- Required to implement a full node
- Only updated when errors are found or the specification changes



ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER

BERLIN VERSION beacfbd – 2022-10-24

DR. GAVIN WOOD
FOUNDER, ETHEREUM & PARITY
GAVIN@PARITY.IO

ABSTRACT. The blockchain paradigm when coupled with cryptographically-secured transactions has demonstrated its utility through a number of projects, with Bitcoin being one of the most notable ones. Each such project can be seen as a simple application on a decentralised, but singleton, compute resource. We can call this paradigm a transactional singleton machine with shared-state.

Ethereum implements this paradigm in a generalised manner. Furthermore it provides a plurality of such resources, each with a distinct state and operating code but able to interact through a message-passing framework with others. We discuss its design, implementation issues, the opportunities it provides and the future hurdles we envisage.

1. INTRODUCTION

With ubiquitous internet connections in most places of the world, global information transmission has become incredibly cheap. Technology-rooted movements like Bitcoin have demonstrated through the power of the default, consensus mechanisms, and voluntary respect of the social contract, that it is possible to use the internet to make a decentralised value-transfer system that can be shared across the world and virtually free to use. This system can be said to be a very specialised version of a cryptographically secure, transaction-based state machine. Follow-up systems such as Namecoin adapted this original "currency application" of the technology into other applications, albeit rather simplistic ones.

Ethereum is a project which attempts to build the generalised technology; technology on which all transaction-

is often lacking, and plain old prejudices are difficult to shake.

Overall, we wish to provide a system such that users can be guaranteed that no matter with which other individuals, systems or organisations they interact, they can do so with absolute confidence in the possible outcomes and how those outcomes might come about.

1.2. **Previous Work.** Buterin [2013a] first proposed the kernel of this work in late November, 2013. Though now evolved in many ways, the key functionality of a blockchain with a Turing-complete language and an effectively unlimited inter-transaction storage capability remains unchanged.

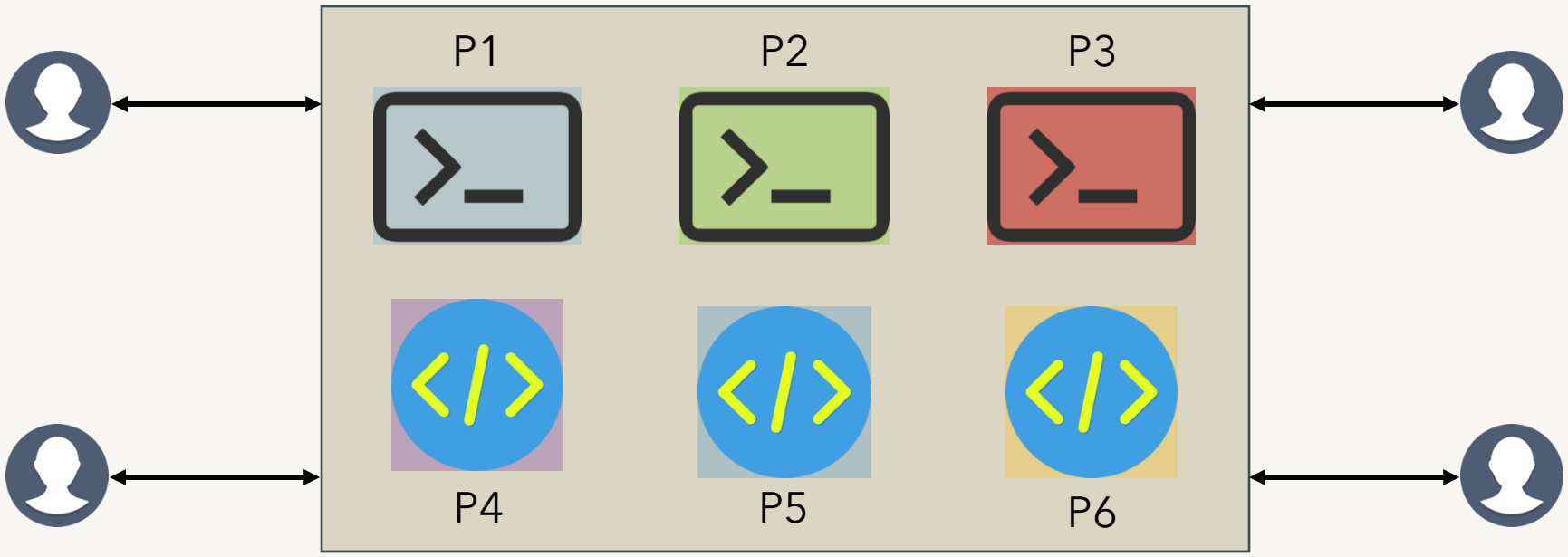
Dwork and Naor [1992] provided the first work into the usage of a cryptographic proof of computational expendi-

https://ethereum.github.io/yellowpaper/paper.pdf

Ethereum world computer

- What Ethereum is trying to do is to develop a world computer
- A world computer is something that anyone can
  - access
  - install programs (smart-contracts)
  - run (execute) those smart-contracts
  - store data into the world computer
  - retrieve data from the world computer

World computer



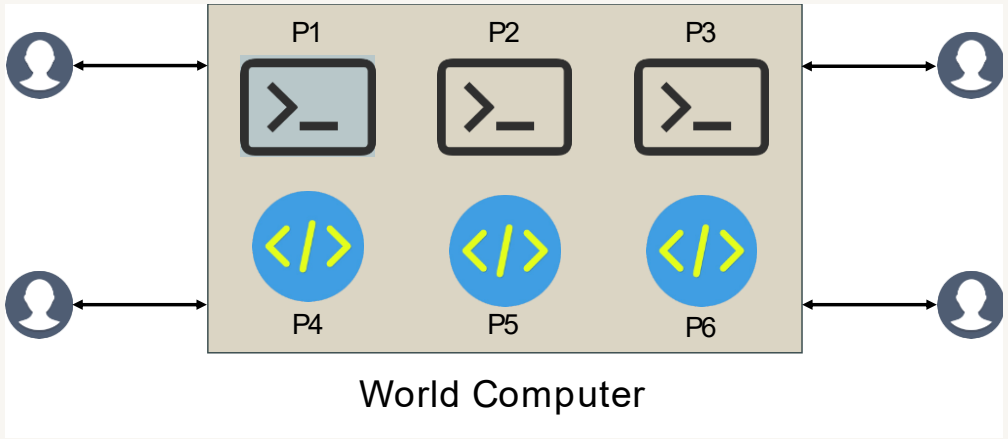
World Computer

Ethereum

- Drawing parallel to the existing centralised system:
  - A client usually connects to a central server (it can be a web server, database server or even a storage server)
  - This creates a bottleneck: a single point of failure in case when the server crashes

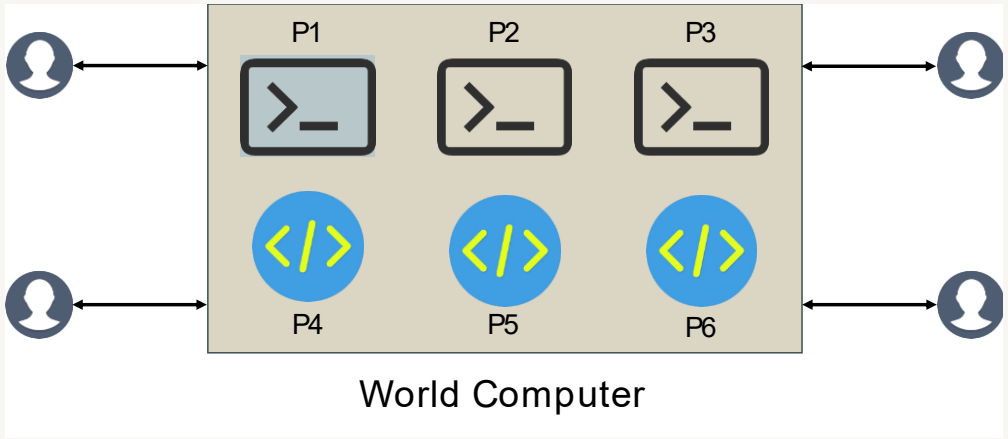
World computer

- Is a single world computer possible?
- Where will it be hosted?
- Who will control it?
- Who will provide the huge computation and storage capacity?

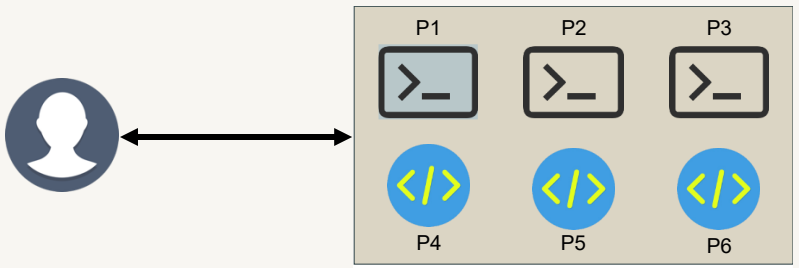


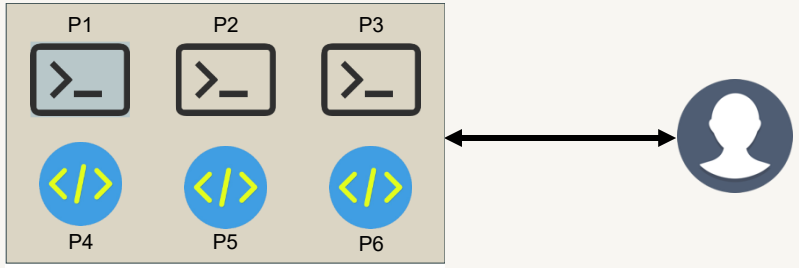
World computer

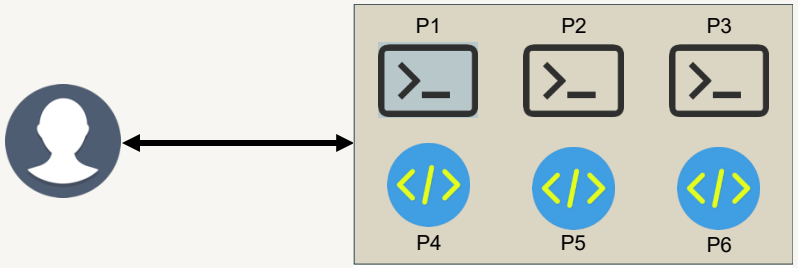
- How to avoid a single point of failure?
- What to do when attacks happen?
- How to mitigate attacks?
- Solution: Decentralise the world computer

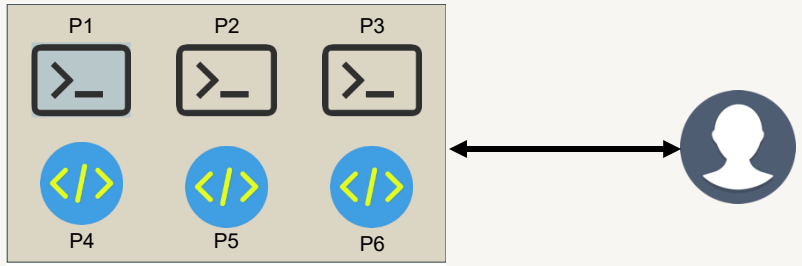


World computer



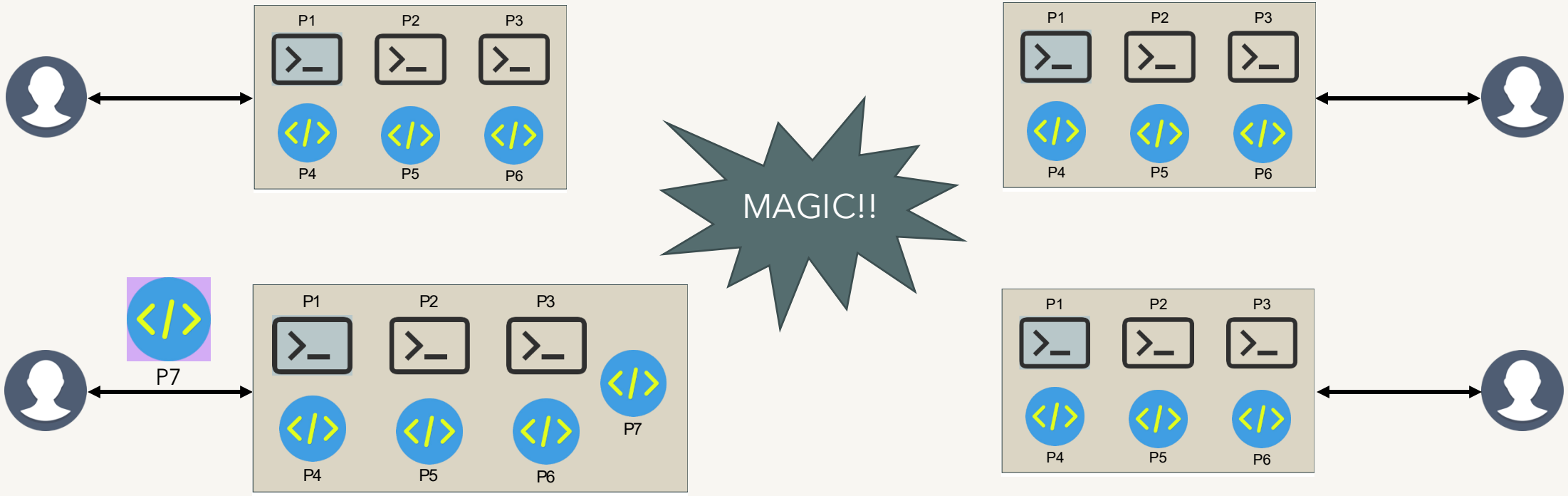




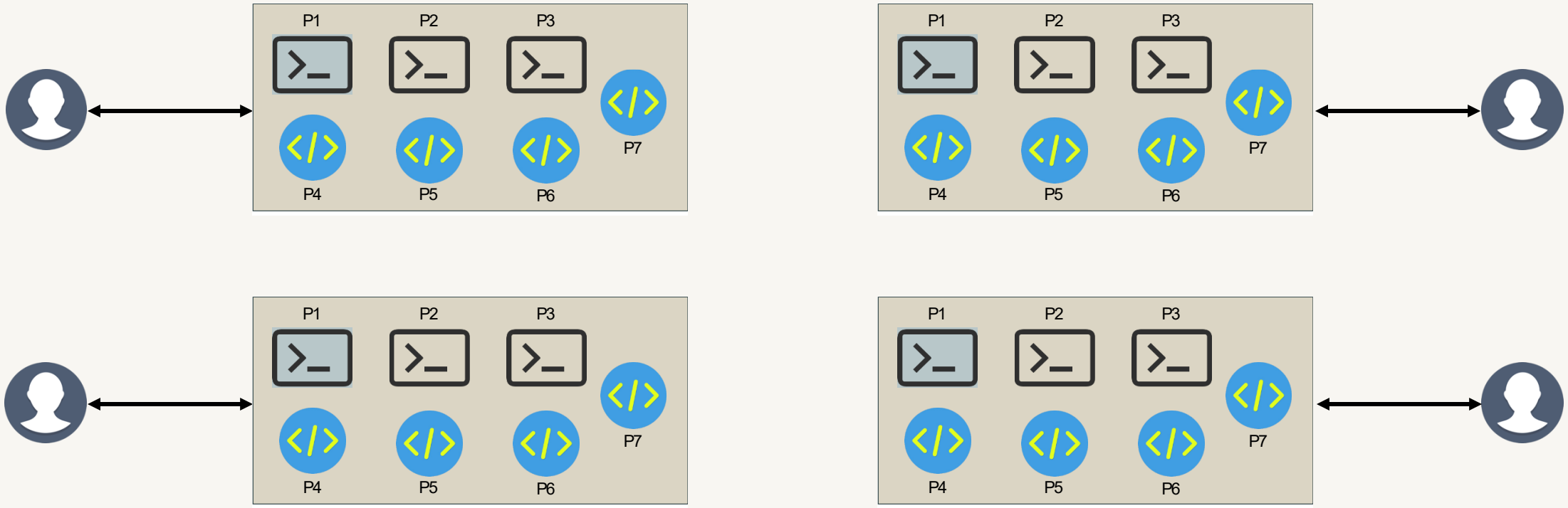


World computer

World computer

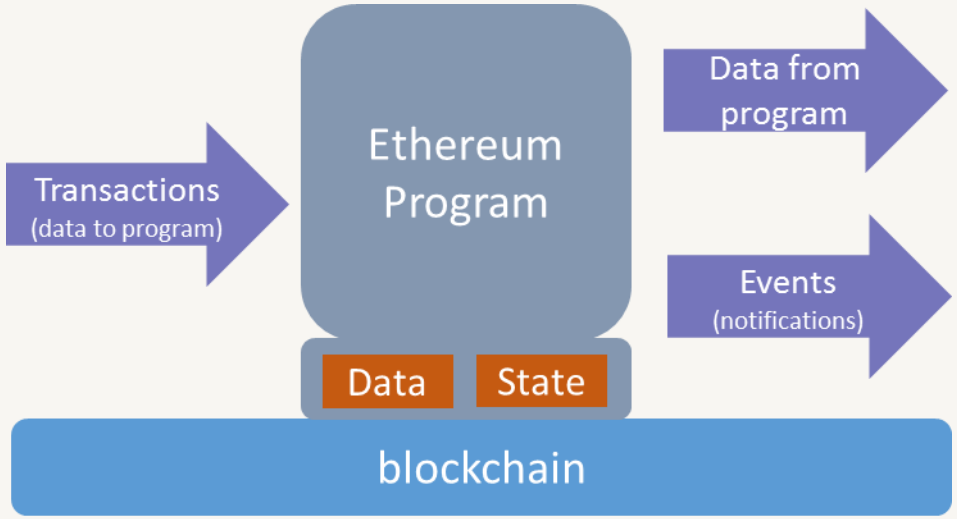


World computer



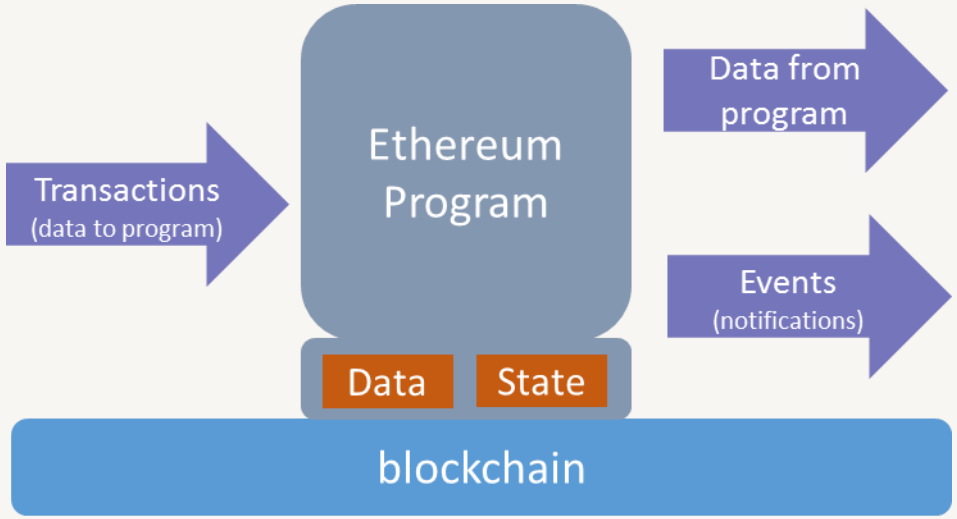
Ethereum

- Ethereum consists of EVM and its blockchain
- An Ethereum program is like any computer program
  - You can install it within the VM
  - Installing a program is known as deploying the code
- Upon receiving data/command via transactions
  - It executes the code based on its logic
  - This changes its state(s)



Ethereum

- States of the EVM and the data it contains are stored in the blockchain
- A consensus protocol ensures a universal synchronisation of state and data for each program in the blockchain
- Data can be retrieved from the program



Ethereum

- This blockchain based EVM distribution allows Ethereum to act as a decentralised World Computer!
- The decentralised nature of the platform guarantees that the application will always run without any downtime, modification, censorship, fraud or other interference!
- Being based on blockchain means once a code is deployed in the EVM, it is replicated to all nodes running the EVM
- This is an extremely powerful capability, sought after in many application domains!
- Apart from running applications, Ethereum blockchain can also transfer money between 2 parties without a central authority, just like Bitcoin

Ethereum

- Unlike any traditional server-based setting, here, you interact with the EVM inside your own computer!
- You can deploy (install) your code within a single EVM and then
  - it gets installed in every single EVM in each node within the network by means of consensus!
- Then, you interact with your code to
  - store data in the blockchain
  - call a function to perform some computations
- These change states of the EVM which are also get propagated using its consensus mechanism

Ethereum

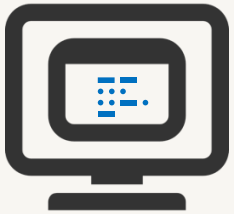
- An Ethereum program is called a smart-contract
  - Deployed in the blockchain via transactions
  - Costs a crypto-currency called *Ether*
- Almost works like a program written in a programming language in traditional computers
  - Can be invoked with data to perform specified operations on the supplied data which can be retrieved

Ethereum

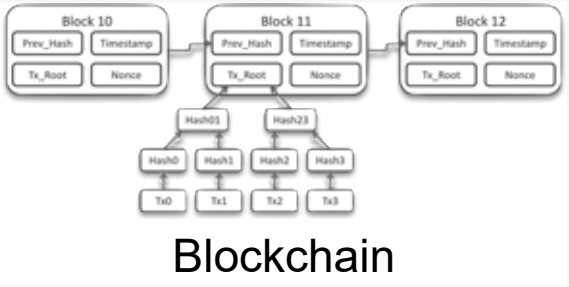
- Since smart-contracts are stored in a blockchain
  - Irreversible once part of a blockchain
  - Provides strong security guarantee against the compromisation of the infrastructure
- A DApp (Decentralised application) is a web-service
  - Provides an interface for interaction between the Ethereum blockchain and other web-services
  - Utilises JavaScript

Ethereum components

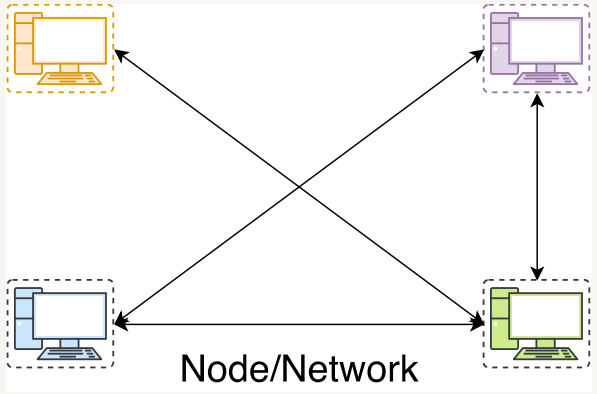
Users & code representation

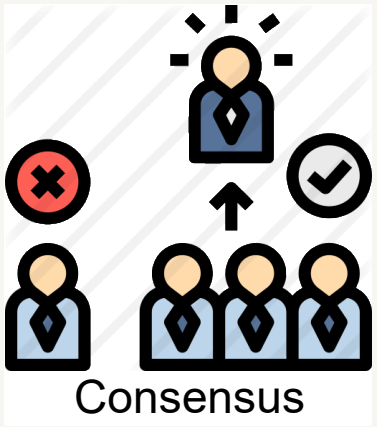


EVM



Blockchain



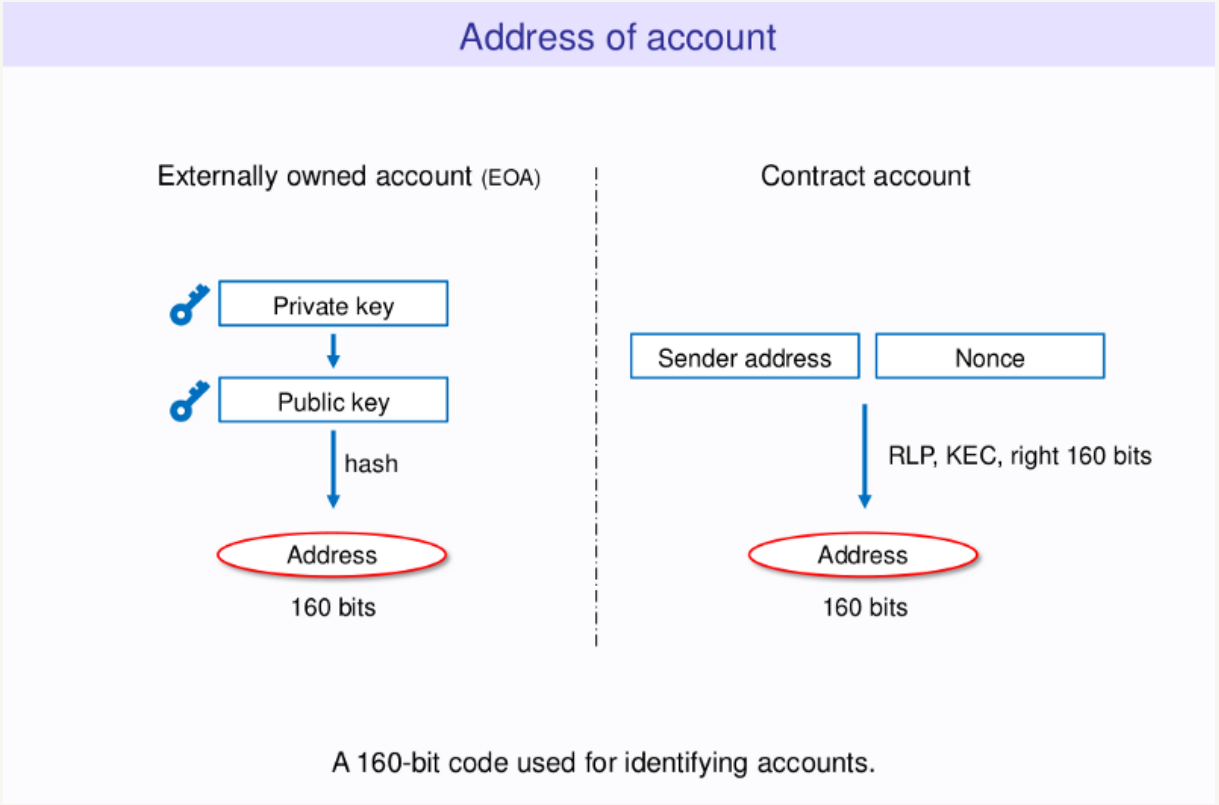


Consensus

Ethereum address

- Ethereum address is different to Bitcoin's address
  - It is generated in a similar way but using different mechanisms
- It utilises the similar concept of public and private key where the address is generated from the public key
  - Private key -> Public key -> keccak256(public key) -> last twenty bytes of the hash = Ethereum address
- Unlike Bitcoin, Ethereum has two types of accounts:
  - Externally owned accounts (EOA): represented by an address generated by a public/private key
  - Contract accounts: generated when a smart-contract is deployed in the EVM

Ethereum address



https://cdn-images-1.medium.com/max/800/1*zEEt3J7gnUiyOhzXVl1jHw.png

RLP - > Recursive Length Prefix (a data serialization process)
KEC -> Keccak 256 Hashing

Question?