# PROOF OF AUTHORITY

...

Gokul G
CB.EN.P2CYS20017

# Agenda

1. What is PoA
2. Consensus Comparison(why PoA)
3. History of PoA
4. AuRa
5. Clique
6. Examples

# What is PoA

- Consensus mechanism where network participants stake their identity and reputation
- Suitable for permissioned blockchains

# Consensus Comparison

**PoW:**

- Computers compete against each other to find solution of digital puzzle
- Incentive: Upkeep >> profit of forged reward
- Disadvantage:
  - Huge upkeep cost
    - GPU,ASICS needed
    - Environmental pollution
    - Electricity, water costs
  - Probabilistic finality
    - Depends on difficulty level
    - Whether current transaction maybe in a fork/bad block

PoS:

- Stake some funds, larger the fund higher the chance of becoming validator
- If you lie then you lose stake
- Incentive: stake >> profit of forged reward
- Comparing to PoW:
  - Not much computation required
  - Less pollution
  - More democratic
  - More fast consensus
- Disadvantage:
  - Assumes that bigger the numerical value, less chance staker will misbehave
  - Ignores what percentage of his money has the staker put

# Consensus Comparison

PoA:

- Better than PoW as
  - Not much computation costs
  - No need of mining blocks
  - Not much environmental problems
  - Deterministic calculation of when next block is created
- Better than PoS as
  - Removes monetary incentives for block creation
  - Removes concept of gas fees

PoA is:

- Used in permissioned blockchains
  - Offers consensus with few stakeholder trusted validators
    - Stringent KYC
    - High speed of consensus
  - Limits possibilities of bad actors
  - Mutable if required
- More scalable than PoW, PoS
  - Validators independent of other participants

- Centralized
- More geared towards enterprise
- By design exposing identity of validators
- By design capable of censorship/blacklisting
- Vulnerable to validator compromise leading to loss of integrity of Blockchain.

# History of PoA

- Ropsten DoS attack (2017)
  - Is PoW network, with low hash rate
  - Attacker
    - Amassed huge number of test ethers
    - Send flood of transactions with very high gas price
    - Caused DoS on Ropsten
- Reaction
  - Creation of Kovan
    - Developed by parity
    - Uses Authority Round (AuRa)
  - Creation of Rinkeby
    - Developed by Geth devs
    - Uses  clique

## Ethereum Testnet Has Suffered A Spam-Attack

**Ethereum network has suffered a spam attack. As noted, the failures in the Ropsten network began when testing new software. The Ethereum developers plan to activate it during the upcoming hardfork called Byzantium. At the same time, they argue that attackers are unlikely to be able to influence the timing of the activation of hardfork and there are no serious reasons for concern.**

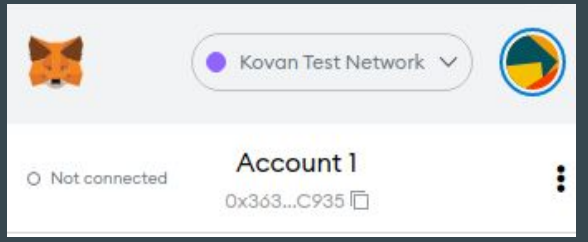
The spam-attack has pushed developers to move into a more secure network Rinkeby. But according to the developer of Raiden, Lefteris Karapetsas, network Ropsten is the closest analogue of the original Ethereum blockchain.

**Spam-attack the Ropsten**

The attackers have filled the Ropsten network with thousands of automatically generated spam transactions. The last such attack happened in March, and then the Ethereum team has forced to re-write a test version of the blockchain.

The team countered the attacks due to the increase in the cost of transactions, but this step adversely affected the possibilities of further testing the software. Subsequently, the developers stopped for a time to operate testnet Ropsten.

# Aura




- Block proposal
  - Validator chosen by round robin algorithm
  - Real time(hence needs to be synced by PTP or similar) is divided into slots
  - Each validator is given slot to propose block.
  - If current validator not responding move to next one

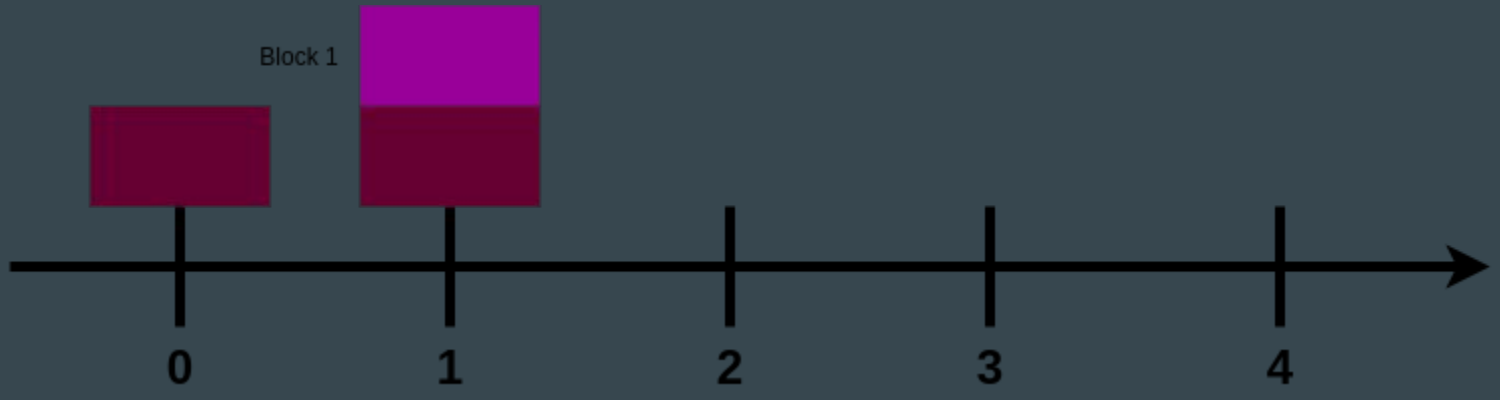
- Time is divided into discrete steps, where:
  - Step = Unix Time / Length of Step
    - Step 1 = 5 / 5
    - Step 20 = 100 / 5
- Each step has an assigned validator, chosen through:
  - Index = s *mod* n
    - Validators [s % n]
    - Validators [1 % 5] = Validators [1]

# Aura

- Finality in AuRa
  - Block is finalized when more than 50% of validators have confirmed blocks on top of it

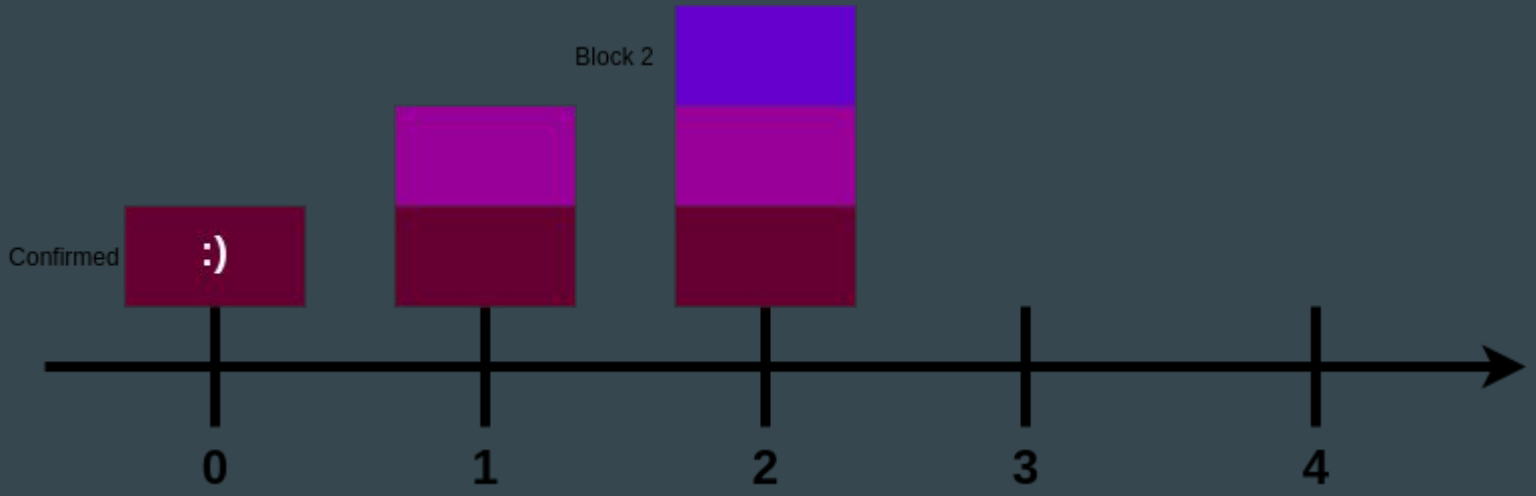
# Aura

- Finality in AuRa
  - Block is finalized when more than 50% of validators have confirmed blocks on top of it



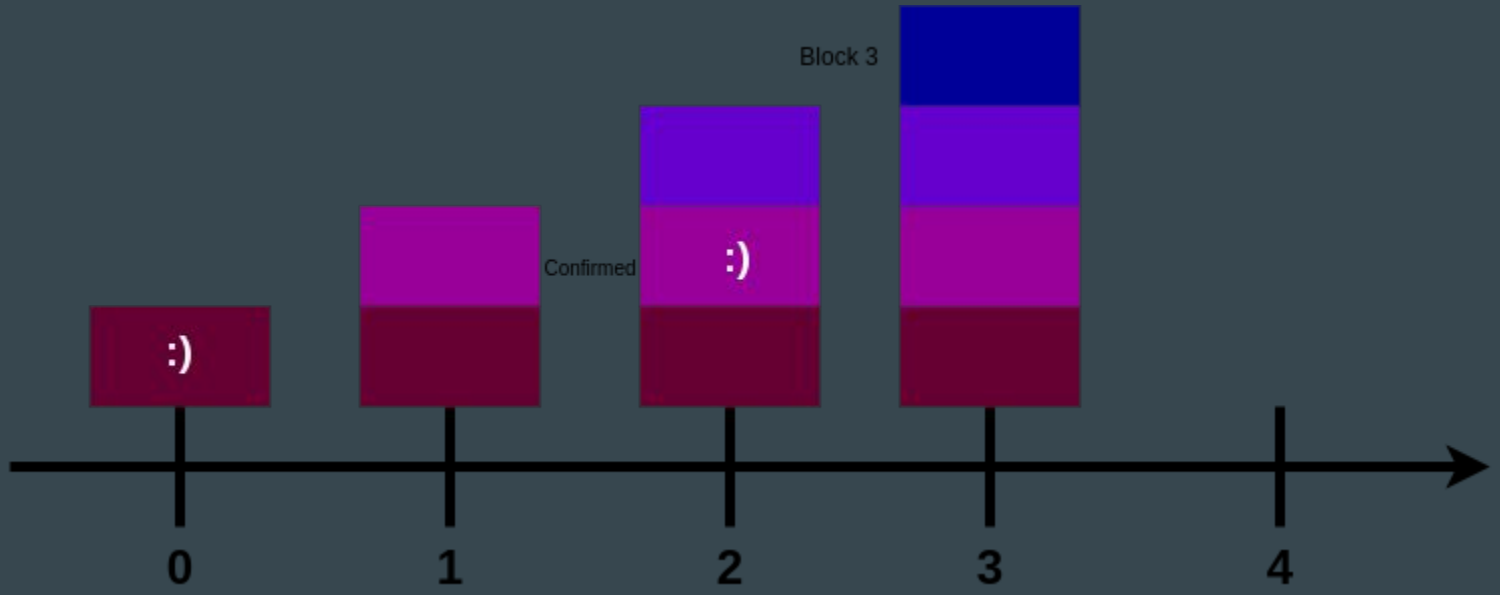
# Aura

- Finality in AuRa
  - Block is finalized when more than 50% of validators have confirmed blocks on top of it



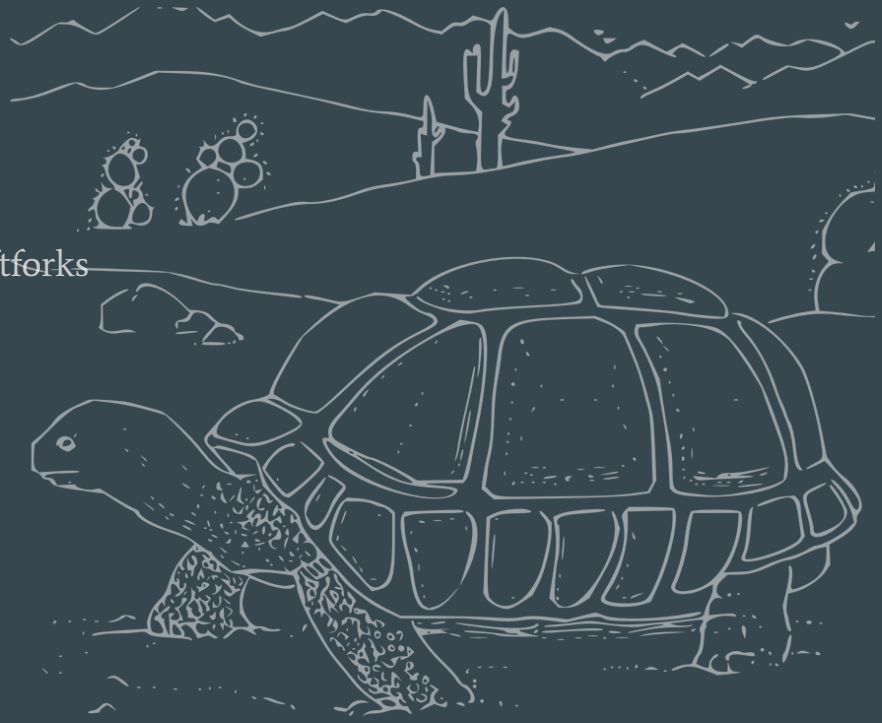
# Aura

- Finality in AuRa
  - Block is finalized when more than 50% of validators have confirmed blocks on top of it



# Aura

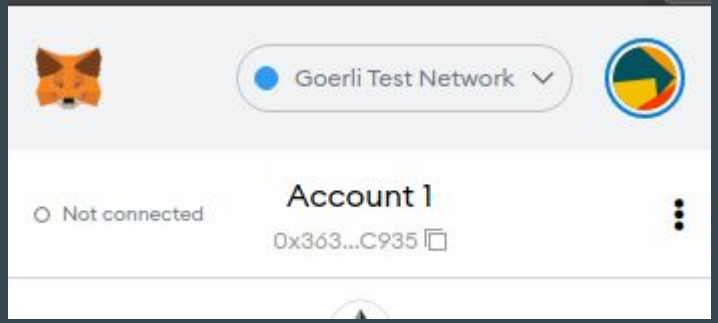
- Finality in Bitcoin blockchain
  - to avoid dropped transaction due to bitcoin softforks
  - wait for 6 blocks on average is  60 mins
- Finality in ETH
  - Wait is 1 min on average for one confirmation
  - Need 50 confirmations for larger transactions
- Where as finality in PoA
  - **5 secs per block** on average



# Aura

- Validator set changes
  - Hard code at chain level
    - In chain.json
    - If need to change validator set, do hard fork
  - Automate changes using smart contract
    - Decide easily who gets to be next validator without forking
    - Block 0-10, one set of static validator, 10-20 another set of validator, from 20 let the SC chose.

# Clique




- Defined in EIP-225
- Used in Rinkeby and Goerli testnets
- Block proposals
  - Validators are either IN_TURN or  NO_TURN
    - IN_TURN -> more weight or priority when it comes to proposing a block
    - NO_TURN -> less weight but still can propose block
  - After proposal, validator have to a predetermined time until it allowed to propose again to prevent dominance of one validator
  - Resolving softforks
    - Prefer chain with more net difficulty score
    - gives preference to chains built with IN_TURN validators
    - Difficulty of IN_TURN = 2
    - Difficulty of NO_TURN = 1

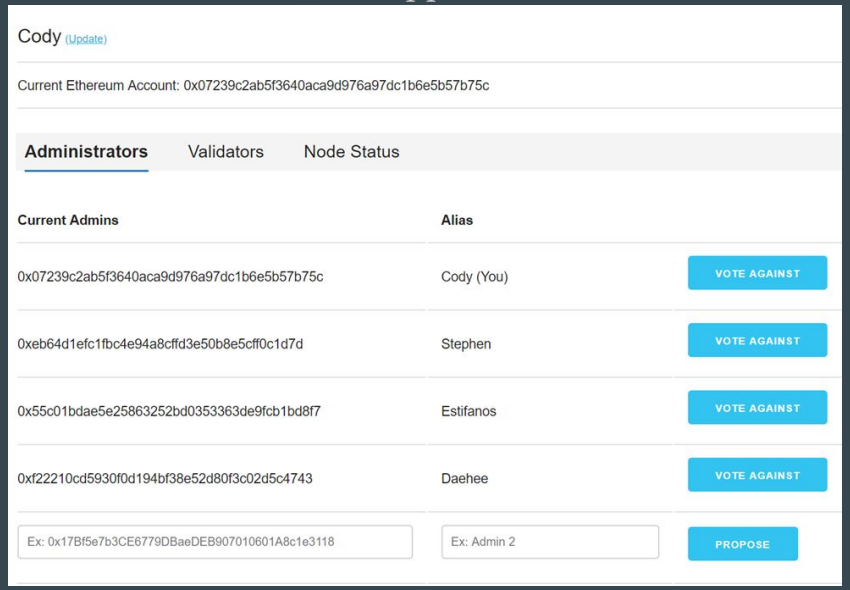
# Clique

- Validator set changes
  - Smart contract
  - Let the other validators decide
    - Per block, validator can vote who can approve the block
      - Only one vote per validator

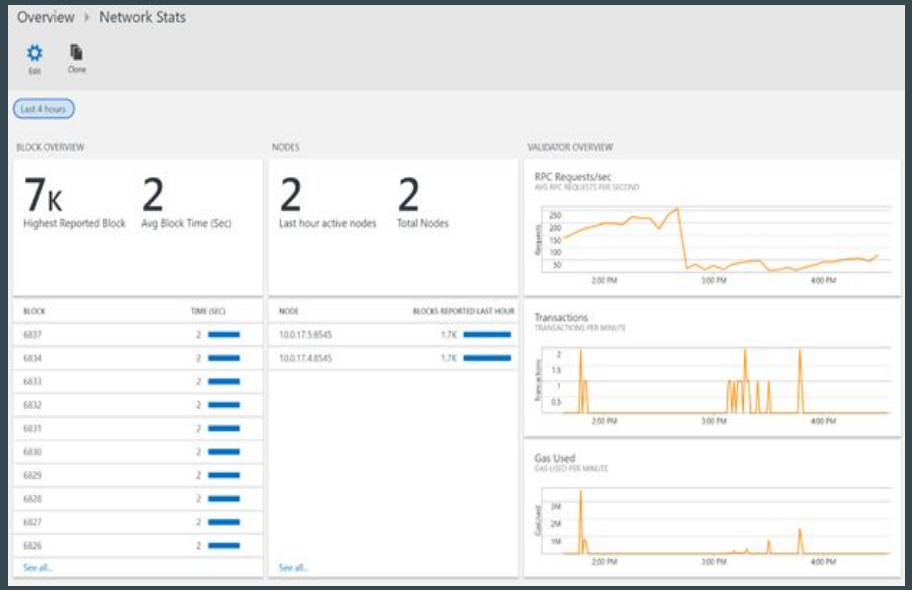
# Examples

- Used in testnets to deter DoS attacks
- Used in Azure platform

Governance Dapp




Azure monitor



# Examples

- In hyperledger Besu