Algorand FOUNDATION

Solving the Blockchain Trilemma

Tal Rabin
Fabrice Benhamouda

Expolab & UC Davis
Class ECS 189F

Why Blockchain & Crypto?

Focus of the Talk

Blockchain: Decentralized Technology

- Distributed, secure, efficient ledger
- Efficient means of exchange
- Infrastructure of the future
- Enables value to flow freely
- Opens doors for inclusive participation

Need each other

Crypto: An Asset Class

- Lending / Bonds
- Staking / Yield
- Futures / Derivatives / Swaps
- Stablecoins
- Tokenized Assets

Particularly, in Algorand participation is based on stake

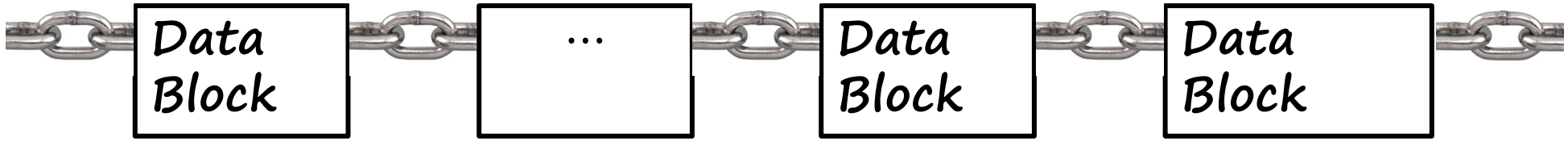
Databases Fails to Work for Many Applications

Centralized management:

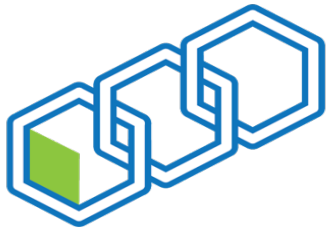
- Who has access
- What types of data they can have
- What is stored in it
- What is deleted
- What is archived

- Single point of failure (insider/intruder)
- Deny or fail to provide access
- Hard to access globally
- Hard to maintain and manage data replication across multiple databases
- Expensive, requires special skills

Blockchain (sequence of data organized by blocks)



(1) Writable by All (2) Readable by All (3) Tamperproof for All



Tamperproof

Transparency

Trust

Blockchain Properties

- ✓ Global instant access
- ✓ Trust and accountability
- ✓ Tamperproof append only log
- ✓ Cheap to transact and share information
- ✓ Virtually impossible to break the system!

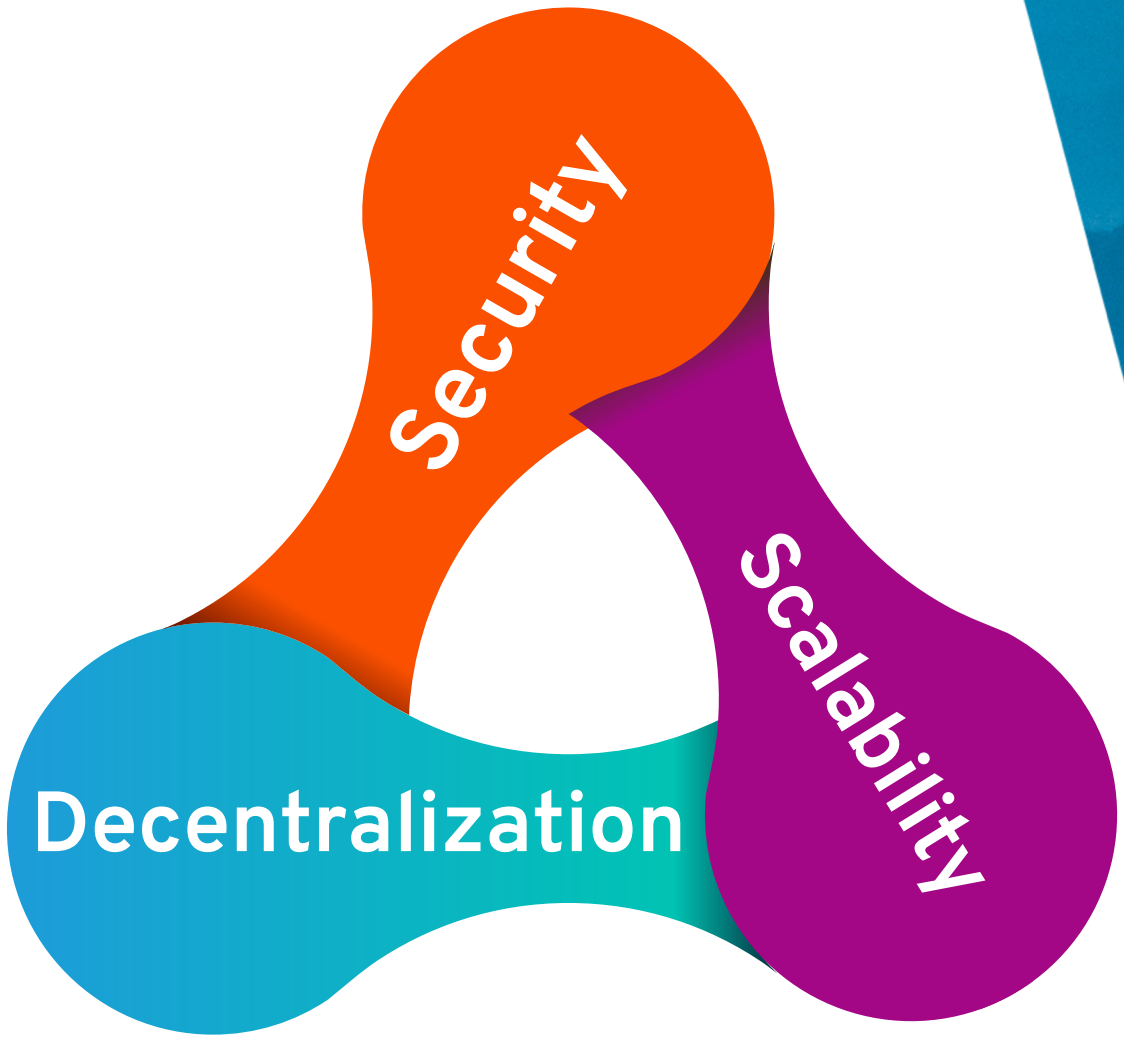
Blockchain is Good for

- ✓ Notarization and Storage
- ✓ Ordering of information
- ✓ Payments and cryptocurrencies
- ✓ Supply chains

And a Lot More

The Blockchain Trilemma

- Security
  - Transactions cannot be tampered nor removed once committed
- Scalability
  - Support high volume of transactions for real-world use
- Decentralization
  - Allows anybody to participate in the consensus



“At most two of…”

Algorand Blockchain Pure Proof-of-Stake

True Security

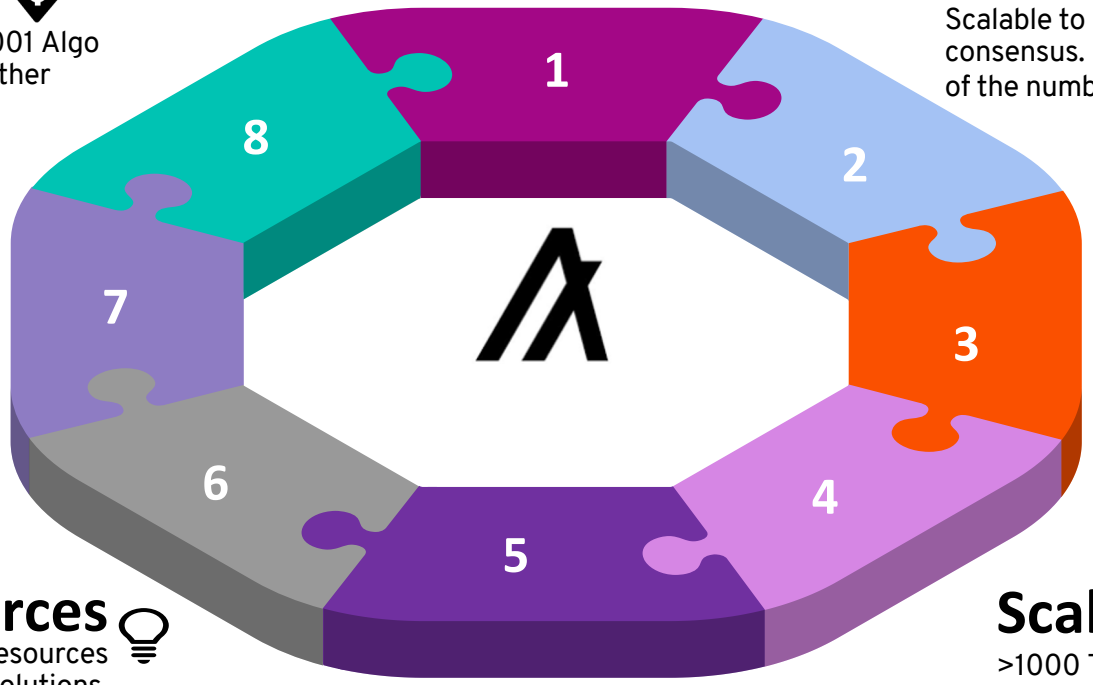
Inherent protection against protocol and network attacks, and highly resilient with global node distribution

Cost Efficient

Cost effective at scale with .001 Algo cost per transaction like all other transactions on the network

Participation Scalability

Scalable to billions of users participating in the consensus. Performance is (mostly) independent of the number of participation nodes.



Minimal Computation

Extremely energy efficient with minimal computations needed making it environmentally friends to run the network

True Decentralization

Influence on block generation is directly proportional to the total stake owned in the system, regardless of how many wallets the tokens are held in

Rich Developer Resources

Easy to use with templates and developer resources that include documentation, tutorials, solutions, forums and more

Scalability

>1000 Transactions Per Second: Scales to billions of users. Blocks propagate every 5 Seconds with immediate transaction finality

No Forking

Each block remains on the chain and is mathematically guaranteed not to fork so no small subset of the tokens can endanger the system or create a fork

General Approach for Generating New Blocks

- "Win" the right to add the new block
- Winner provides:
  - Certificate that it is the winner
  - The next block
- Block is added to the chain

Other Blockchain vs. Algorand's Blockchain

First Generation Blockchains
Proof-of-Work (PoW)

- Not simultaneously decentralized, scalable & secure
- High cost per transaction
- Lack speed, finality & throughput
- Consume an enormous amount power

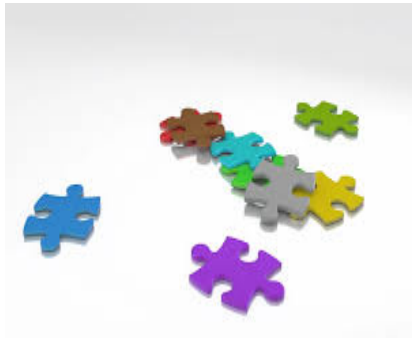
Delegated & Bonded Proof-of-Stake (PoS)

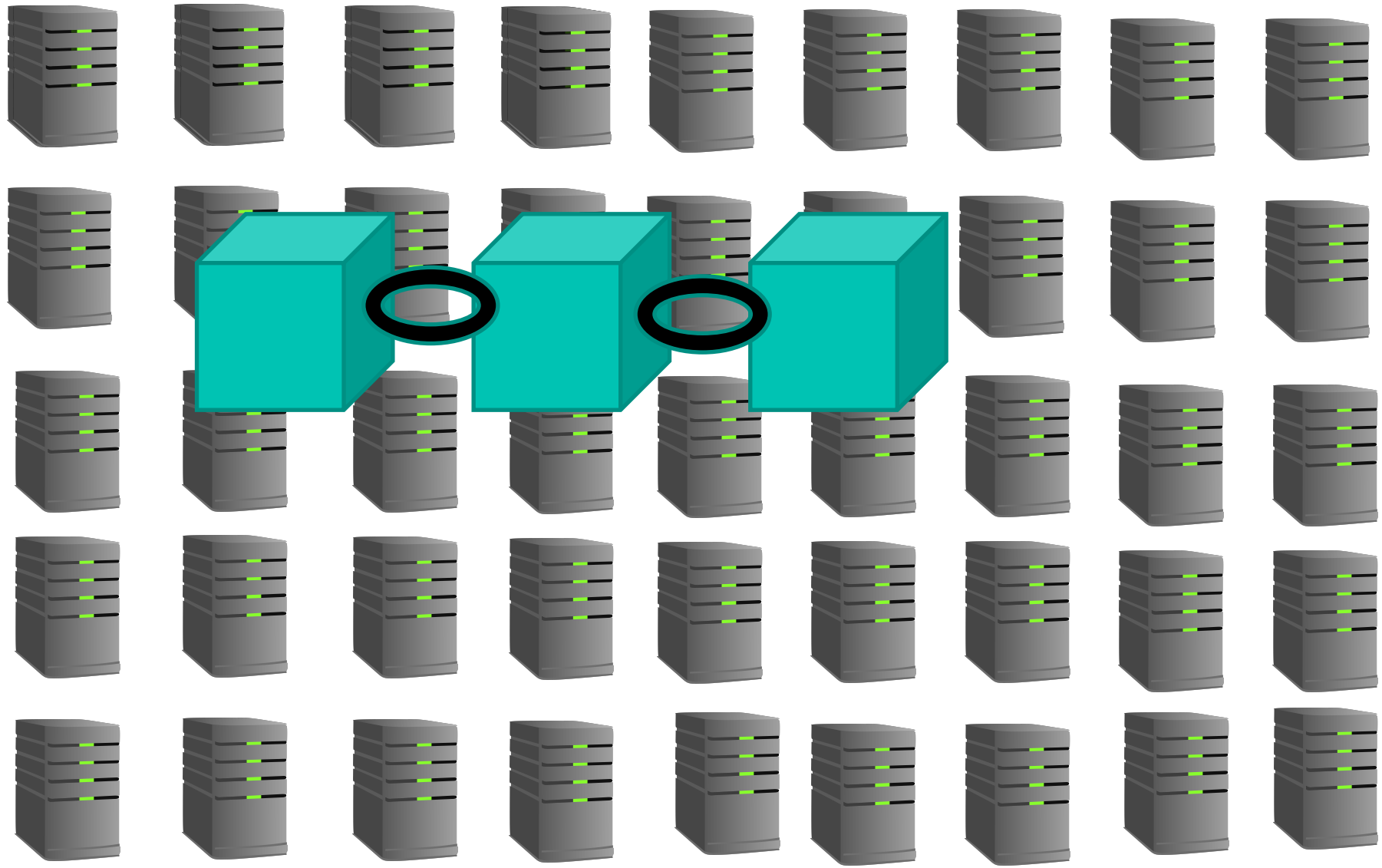
- Not simultaneously decentralized, scalable & secure
- Lack security:
  - Trust is centralized in Delegated systems
  - Bonded systems have a high barrier to entry

Algorand Pure Proof-of-Stake (PPoS)

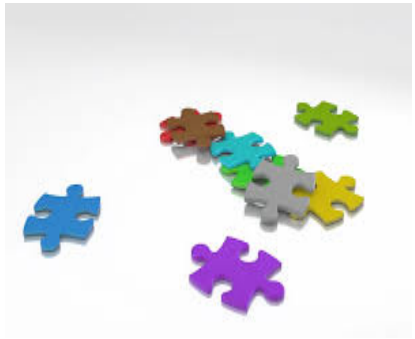
- Simultaneously decentralized, scalable & secure
- Low cost per transaction
- Speed & throughput at 1,000 TPS with Finality in < 5 Sec.
- Consume little power with minimum computation

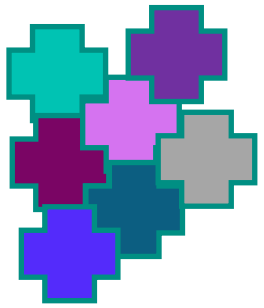
Bitcoin – Nakamoto's Consensus – Proof of Work

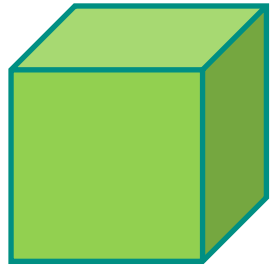




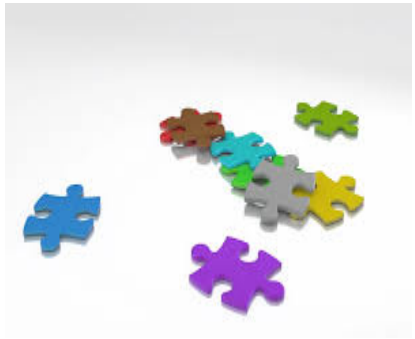
Bitcoin - Nakamoto's Consensus – Proof of Work

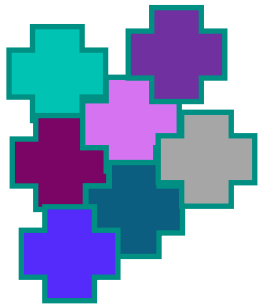


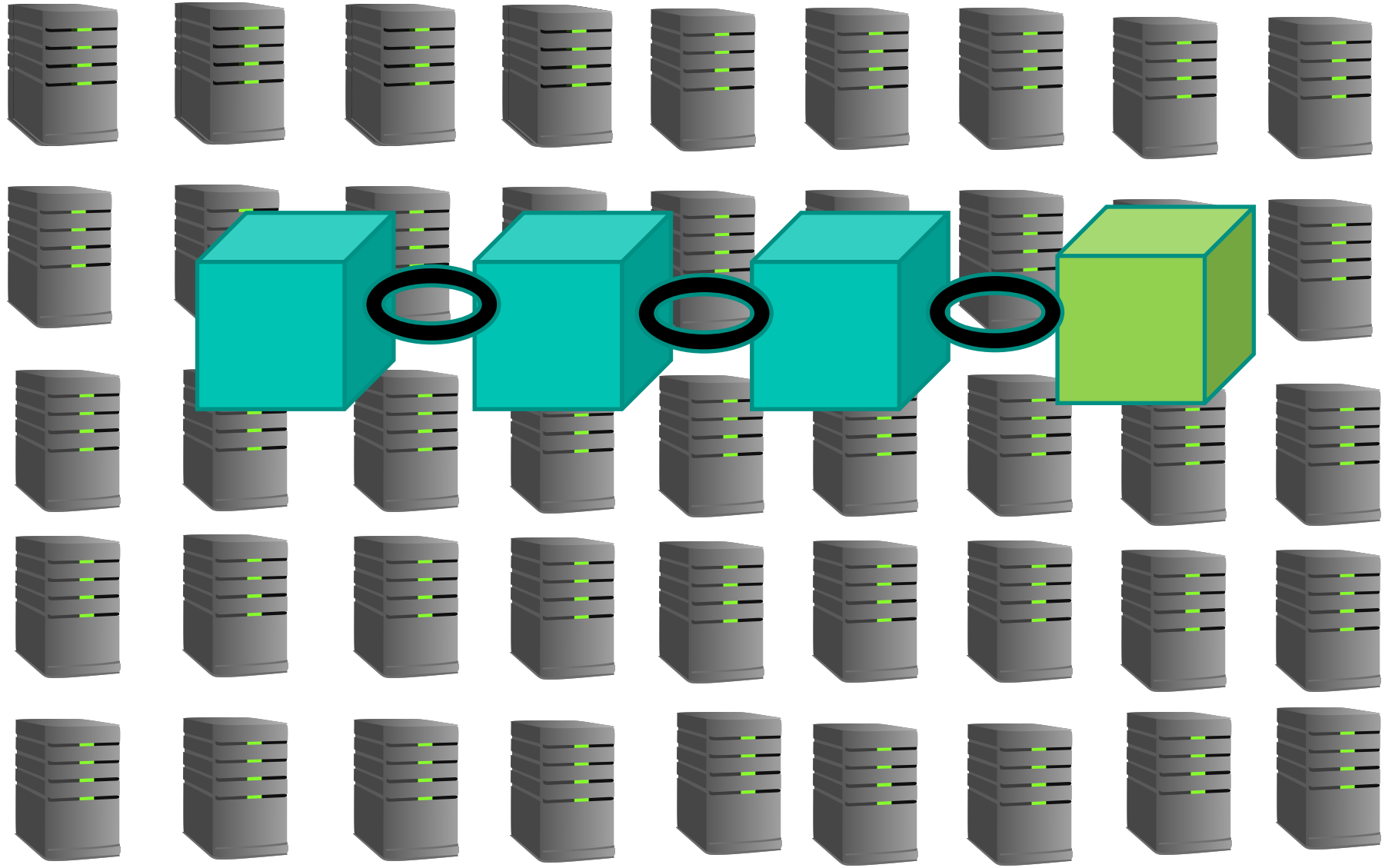




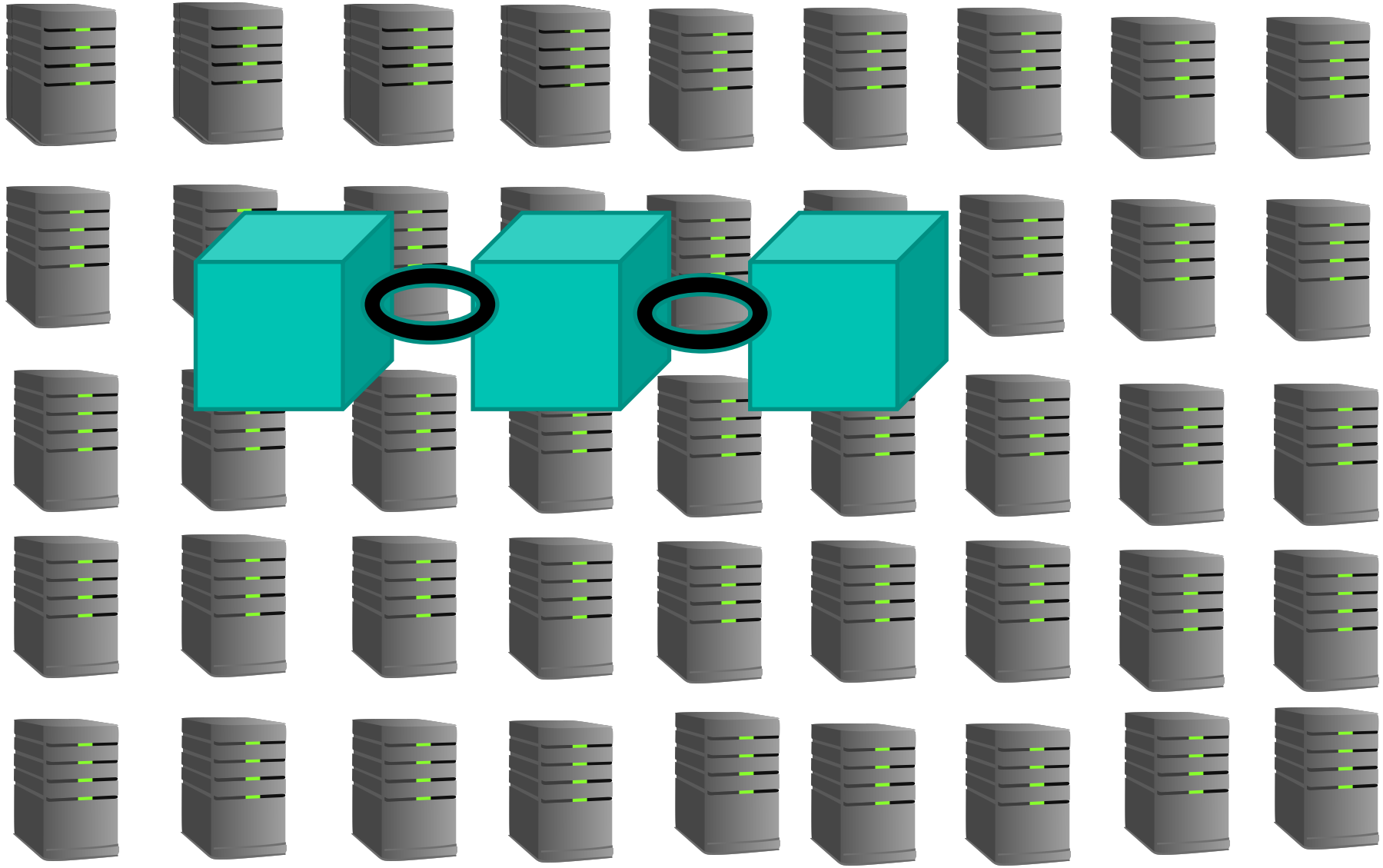
Bitcoin - Nakamoto's Consensus – Proof of Work



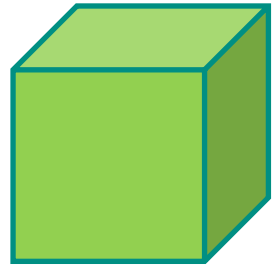




Algorand's Consensus – Proof of Stake

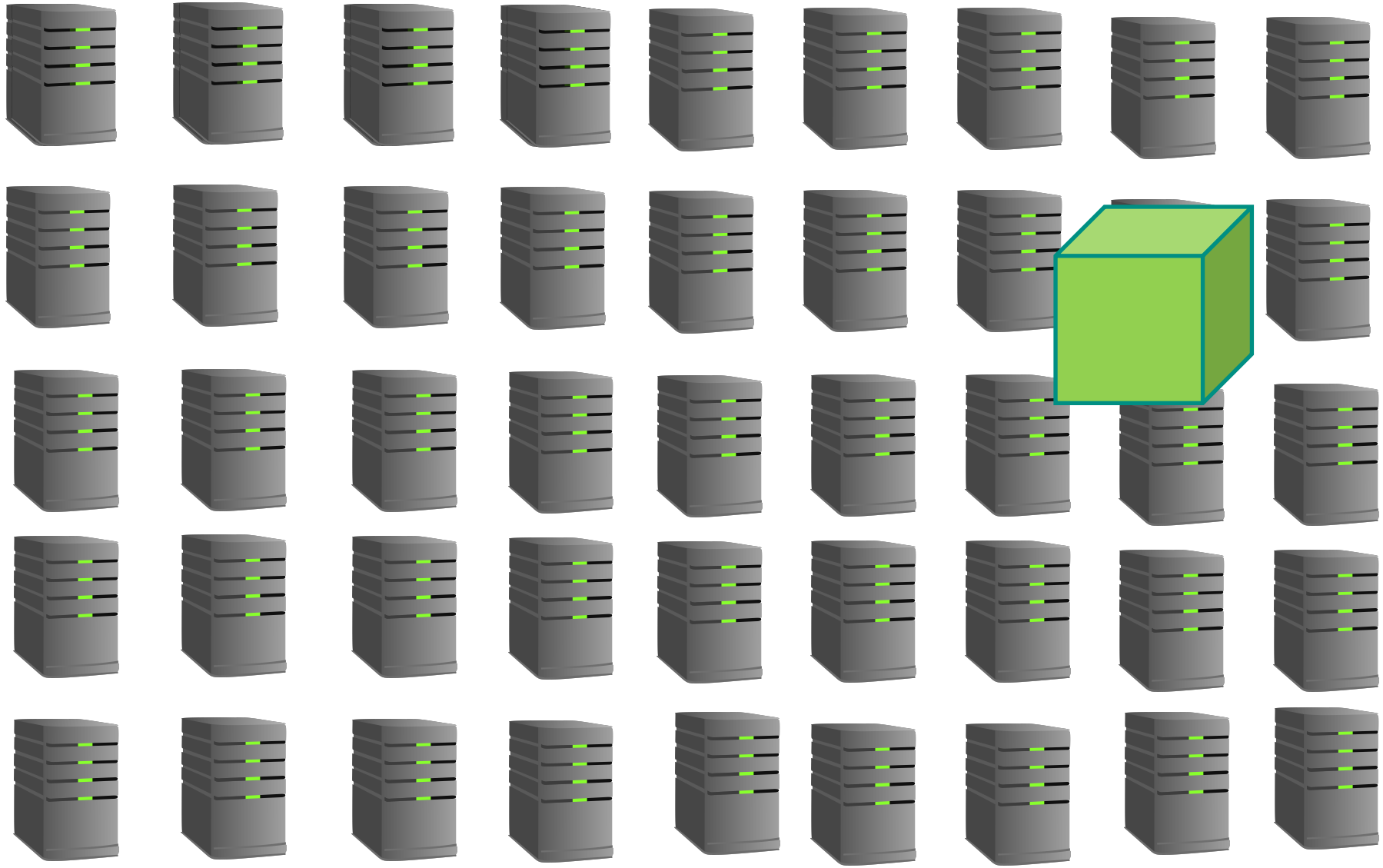


Algorand's Consensus – Proof of Stake

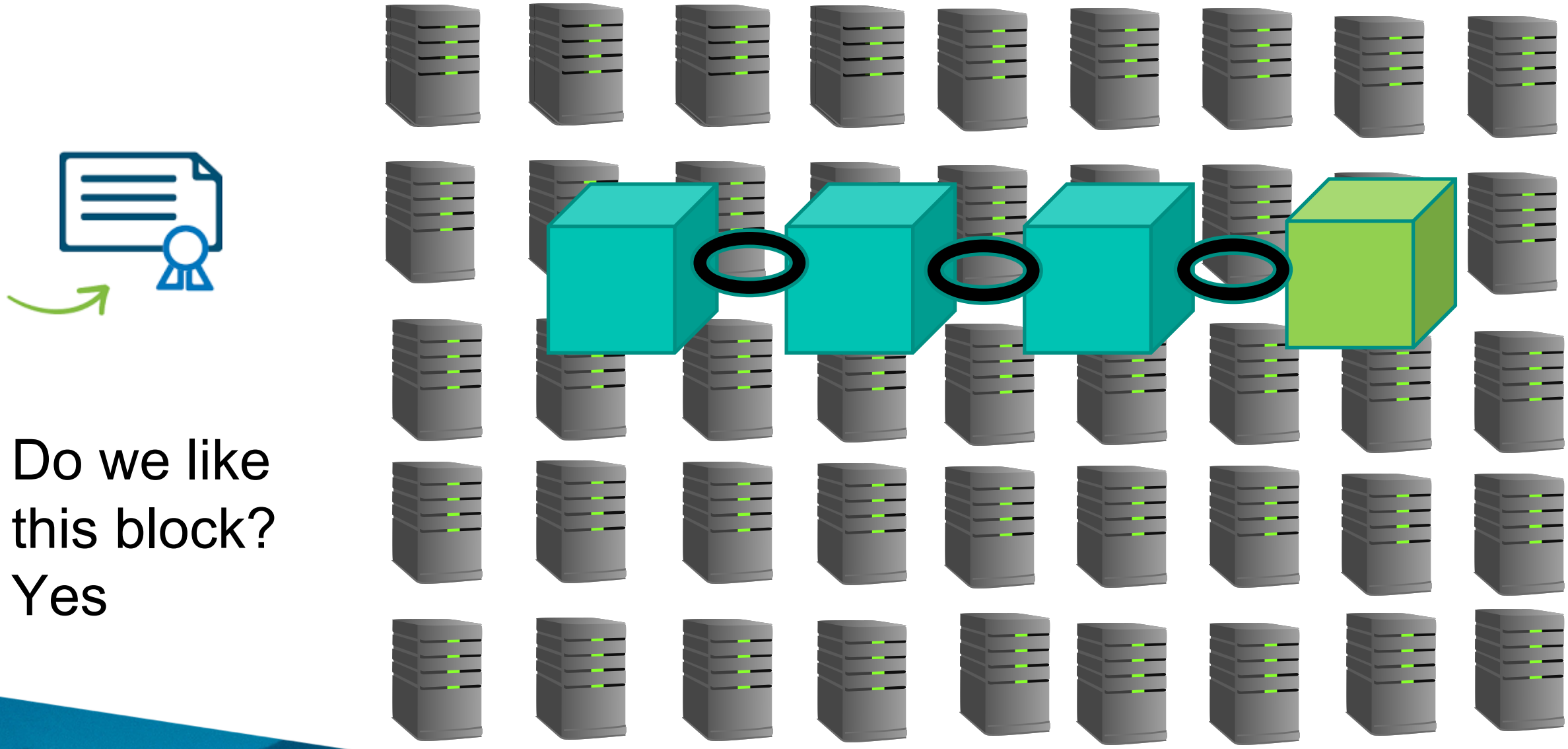


Algorand's Consensus – Proof of Stake

Do we like this block?

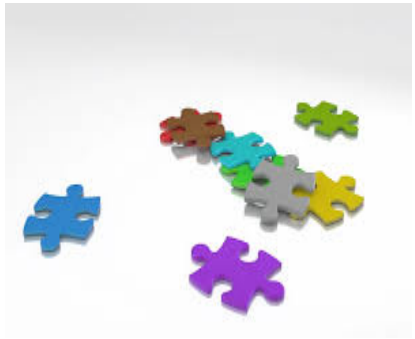


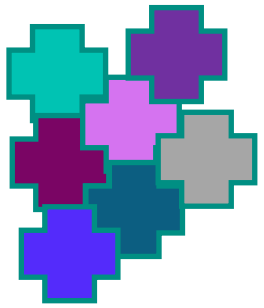
Algorand's Consensus – Proof of Stake

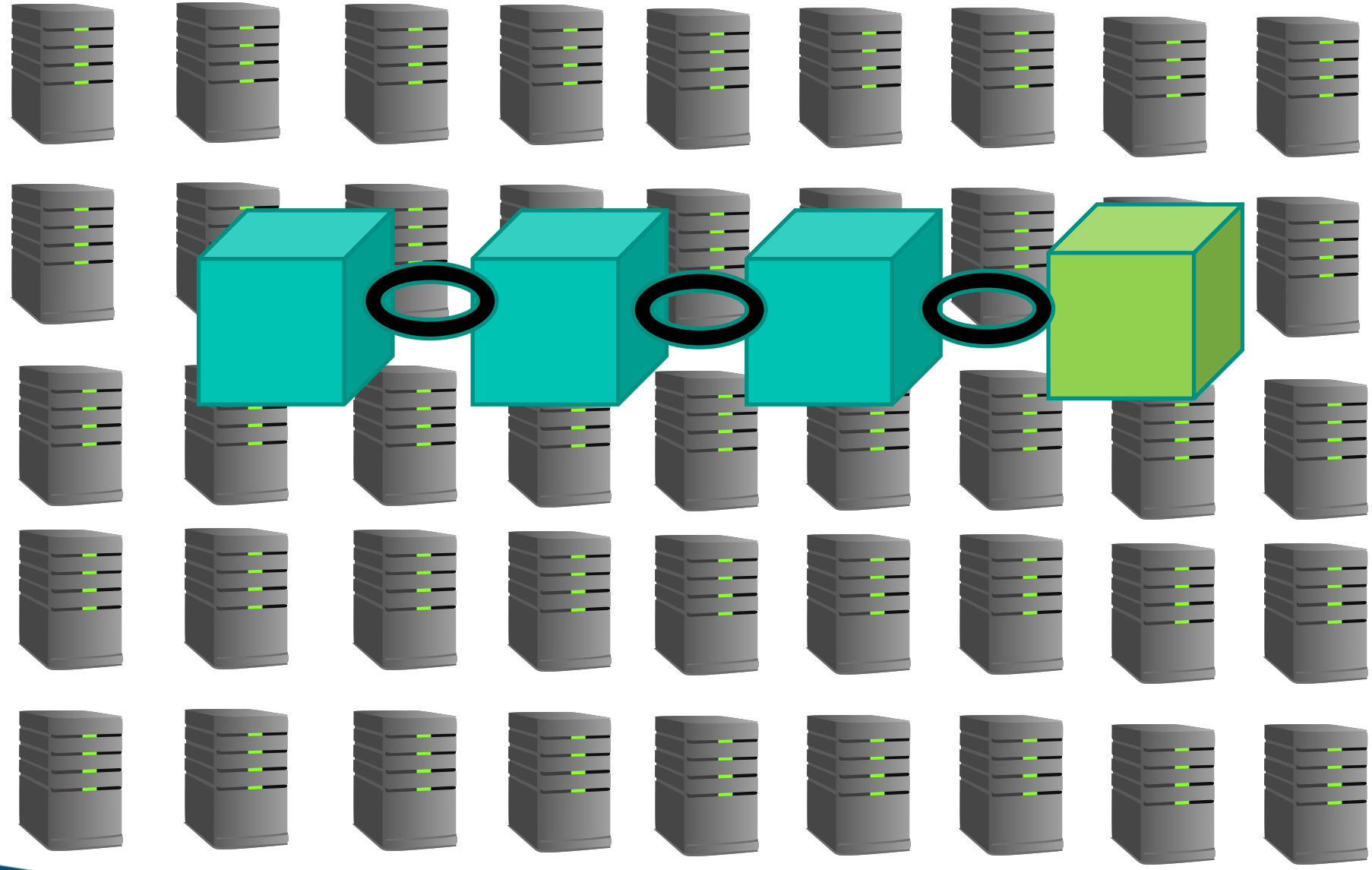


Do we like this block?
Yes

Bitcoin – Nakamoto's Consensus Proof of Work

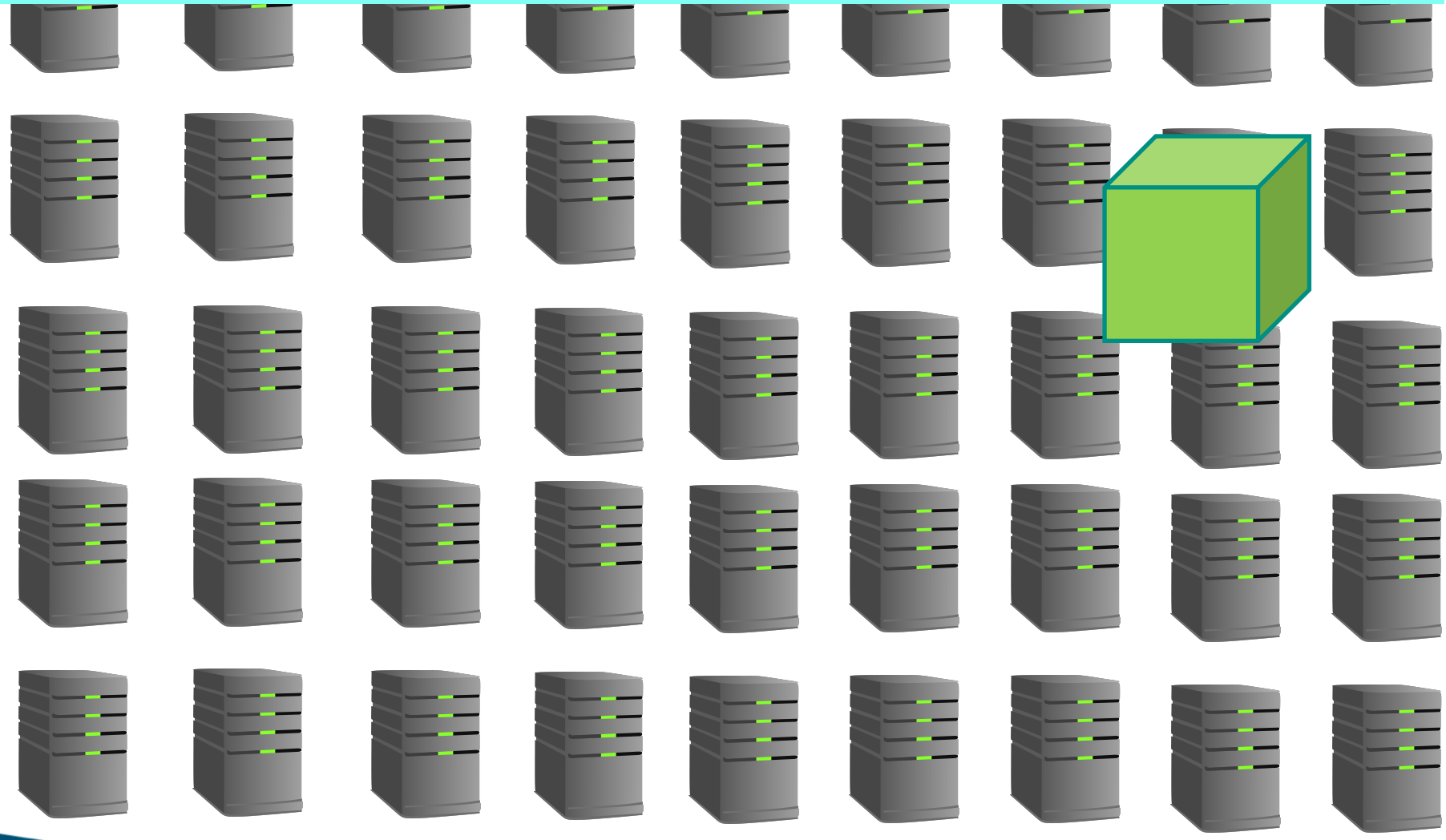






Algorand's Consensus – Proof of Stake

Certificate



Algorand's Consensus – Proof of Stake

Certificate created via a Verifiable Random Function (cryptographic sortition)

Do we like this block?? Resolved via Byzantine Agreement (BA). Yes

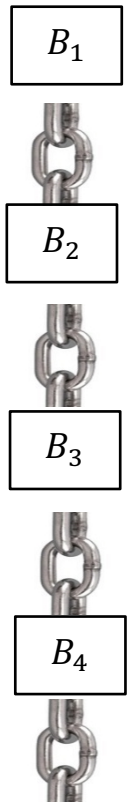
Each step of BA is performed by a different unknown set of parties.

Algorand's Consensus – Proof of Stake

Certificate created via a Verifiable Random Function (cryptographic sortition)

- Introduced by Micali, Rabin, Vadhan
- Very efficient to compute
- Very efficient to verify

Key Idea



1) Sample a small committee <u>at random from the set of all users</u>

How does is this committee chosen?

Same way as before. They win the right to be in the committee and present a certificate indicating that they have won.

3) The block is added to the chain

Technical Advancements

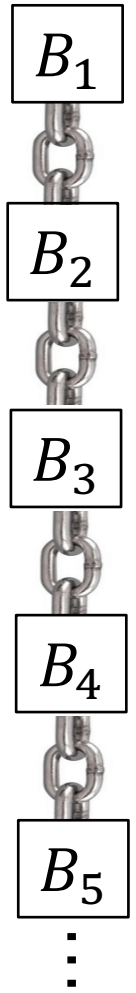
- A **new** an[Follow-up work: computing any function in this model, which we call YOSO You Only Speak Once]ew block while that

- **VRFs** (Verifiable Random Function) is open sourced and Cryptographic Self-Selection to Blockchains. Allow users to secretly, fairly and provable select themselves

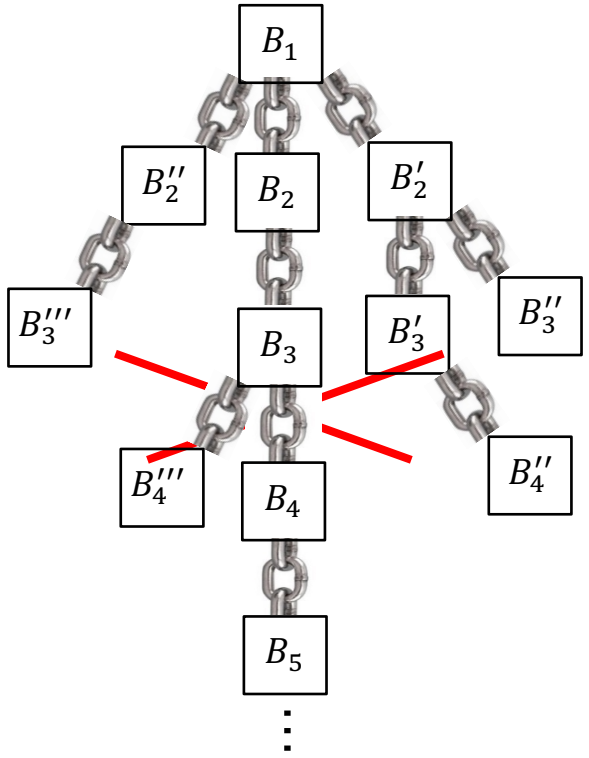
- **Player Replaceability** – Withstands the corruption of all users in the middle of a protocol.

Efficient one-by-one block generation

Algorand



vs.



Proof of Work

Never a fork

Finality!!

(transactions confirmed in seconds not an hour)

+ Efficiency

Main Assumption: 80% of honest money

Main Technical Advantages

- Trivial Computation
- True Decentralization
- Finality of Payments
- Scalability
- Security

Single Class of Users (no exogenous powers)

$Prob[fork] \leq 10^{-18}$

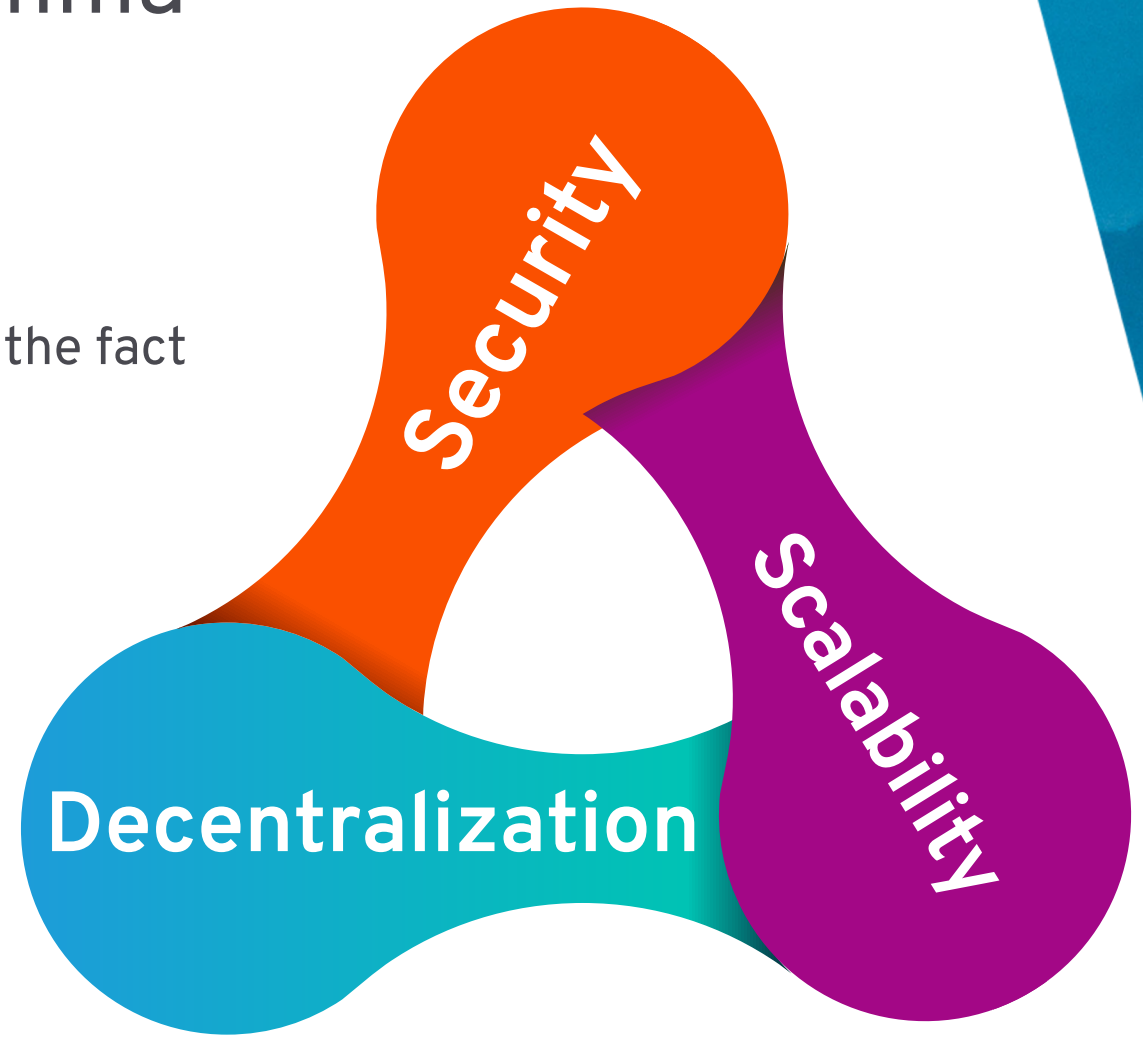
Blocks generated as fast as can be propagated

Against a **dynamic** Adversary

1 block/sec ⇒ 1 fork in the age of the Universe

Resolving the Blockchain Trilemma

- Security
  - Committee members are not known until after the fact
  - Everything is cryptographically signed
- Scalability
  - Minimal messages
  - Lottery execution extremely fast
  - Committees are small
- Decentralization
  - Low barrier to entry
  - Anyone can participate in consensus



“At most two of…”

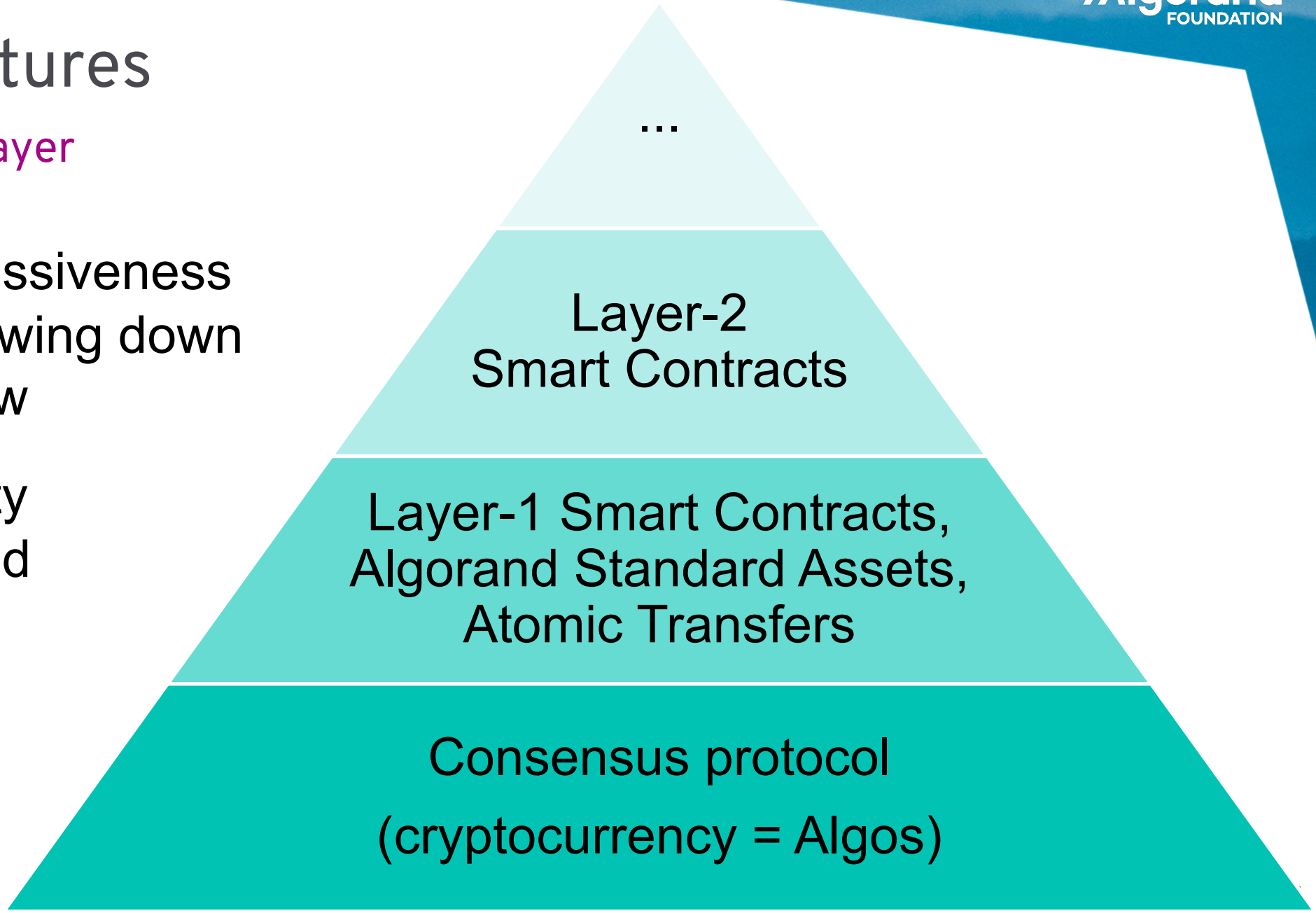
Algorand Features

Building layer by layer

1. More expressiveness
2. Without slowing down layers below

1. Same safety
2. Same speed
3. Same cost

Solving the Blockchain Trilemma



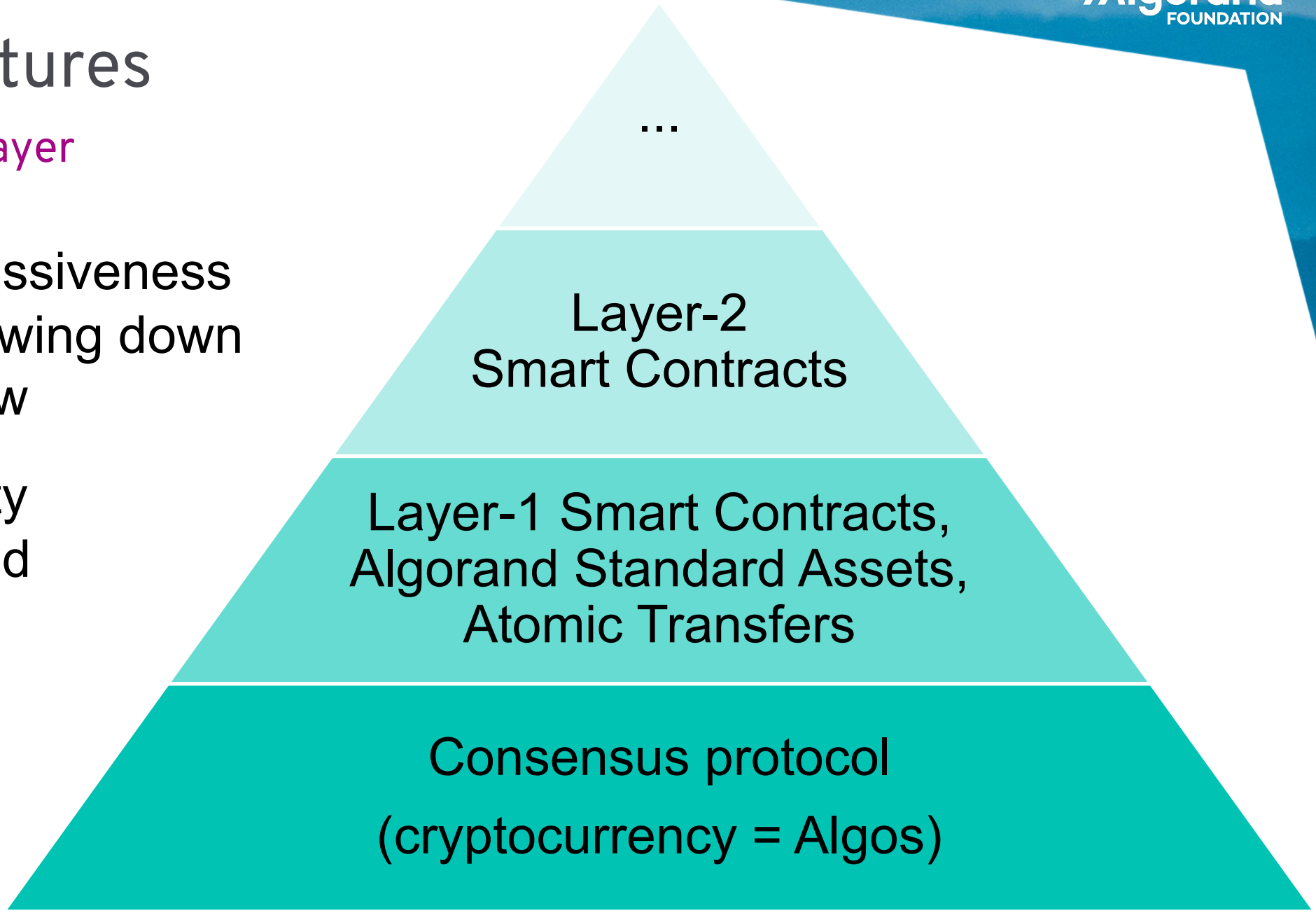
Algorand Features

Building layer by layer

1. More expressiveness
2. Without slowing down layers below

1. Same safety
2. Same speed
3. Same cost

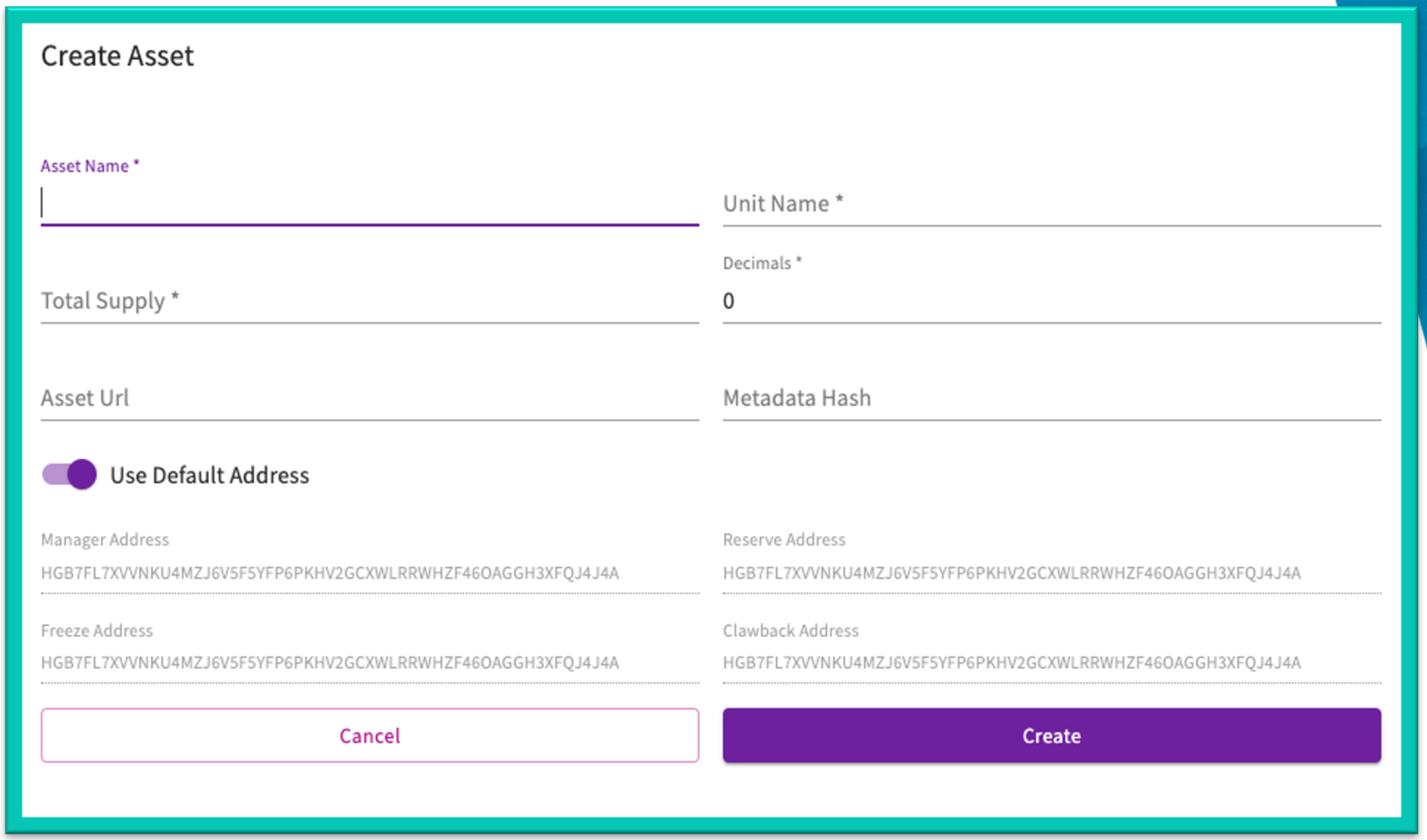
Solving the Blockchain Trilemma



Algorand Standard Assets (ASA)

Introduction

- Native token
  - Same transaction fee as the Algo
  - Same throughput/latency
- Optional administrator:
  - Mint and burn units
  - Freeze accounts
  - Revoke an asset
- Comparison with Ethereum
  - Similar to ERC-20/ERC-721
  - No smart contract
  - Lower transaction fee



Create your token in one-click on asa.algodesk.io !

Algorand Standard Assets (ASA)

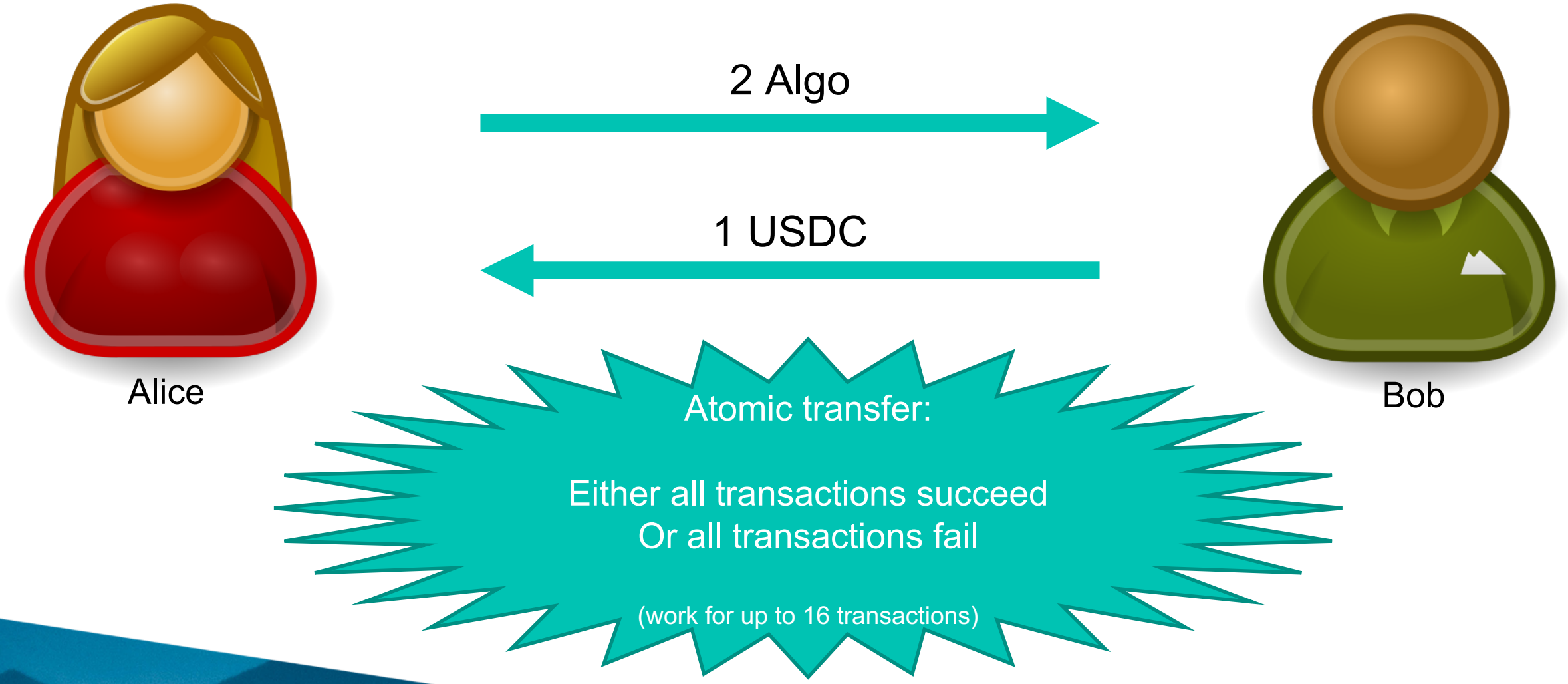
Real-World Examples

- Stable coins: USDC, USDT, …
  - https://www.circle.com/en/usdc, https://tether.to
- Rewards:
  - https://propsproject.com, https://planetwatch.io/
- Stocks, real-estate shares, …:
  - https://about.mese.io, https://assetblock.com

More than 2.5M transactions / week just for Props

- Notarization
  - https://dedit.io
- And many more…

Atomic Transfers / Group Transactions



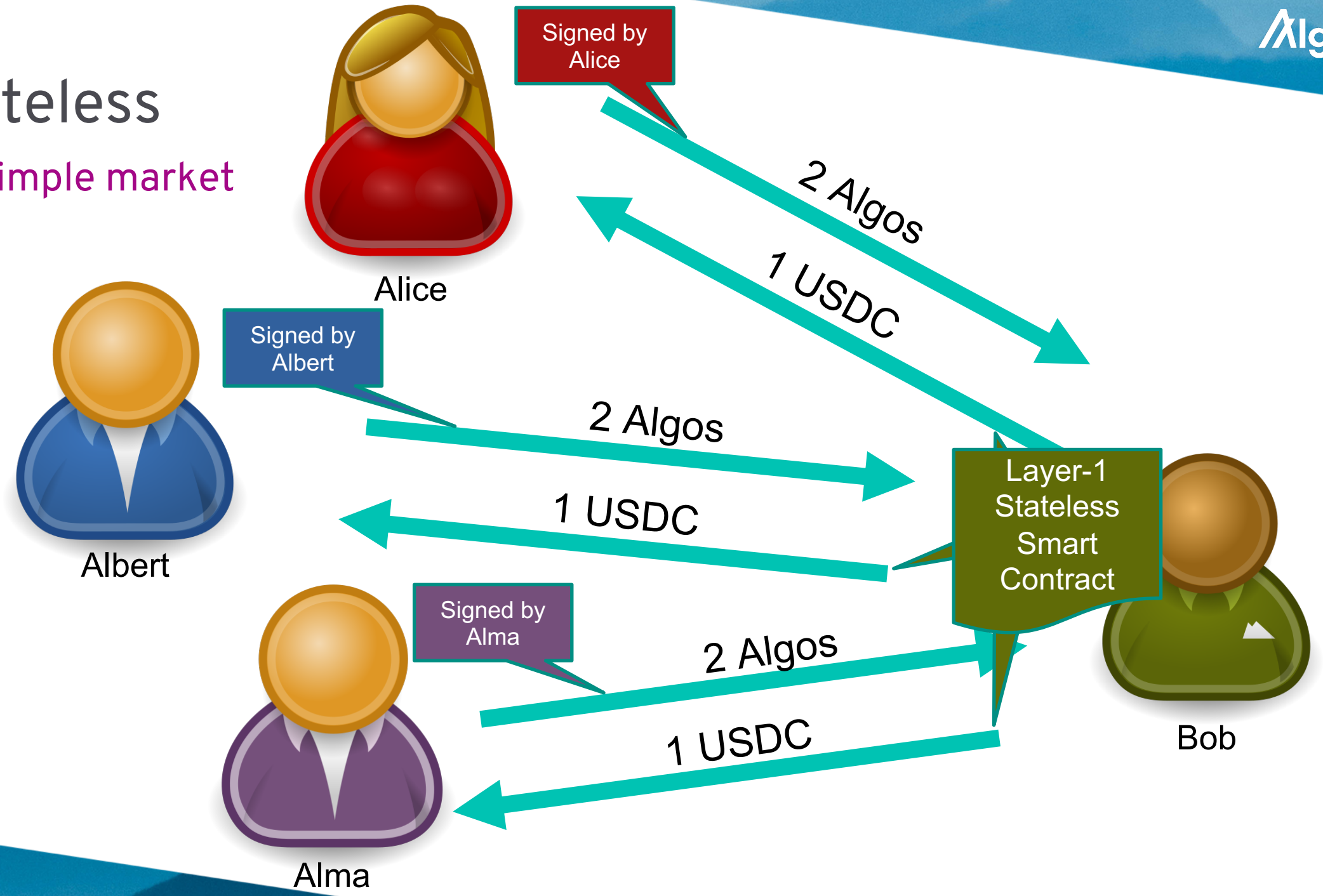
Layer-1 Stateless Smart Contracts

Introduction

- Approve / reject transactions from account
- Written in a simple stack-based language: TEAL
  - No loop, restriction on size and number of cryptographic operations
  - Advantages: easier to formally analyze & less error-prone than Solidity
  - PyTEAL: write scripts in Python
- Same transaction fee as normal transaction!
  - Same latency (1 block every 4.5s), same throughput (1MB block)
- Combinable with all the other features (atomic transfers, ASA, …)
  - Can check all the transaction fields

Stateless

A simple market



TEAL Stack Architecture

Program

```
txn CloseRemainderTo
addr SOEI...
==
txn Receiver
addr SOEI...
==
&&
arg 0
len
int 32
==
&&
arg 0
sha256
byte base64 VeU...
==
&&
txn CloseRemainderTo
addr RFGE...
==
...
```

Stack

uint64/[]byte
uint64/[]byte
uint64/[]byte

…(up to 1000)

Scratch Space

0: uint64/[]byte
1: uint64/[]byte
2: uint64/[]byte
...
255: uint64/[]byte

Args

(This txn only)

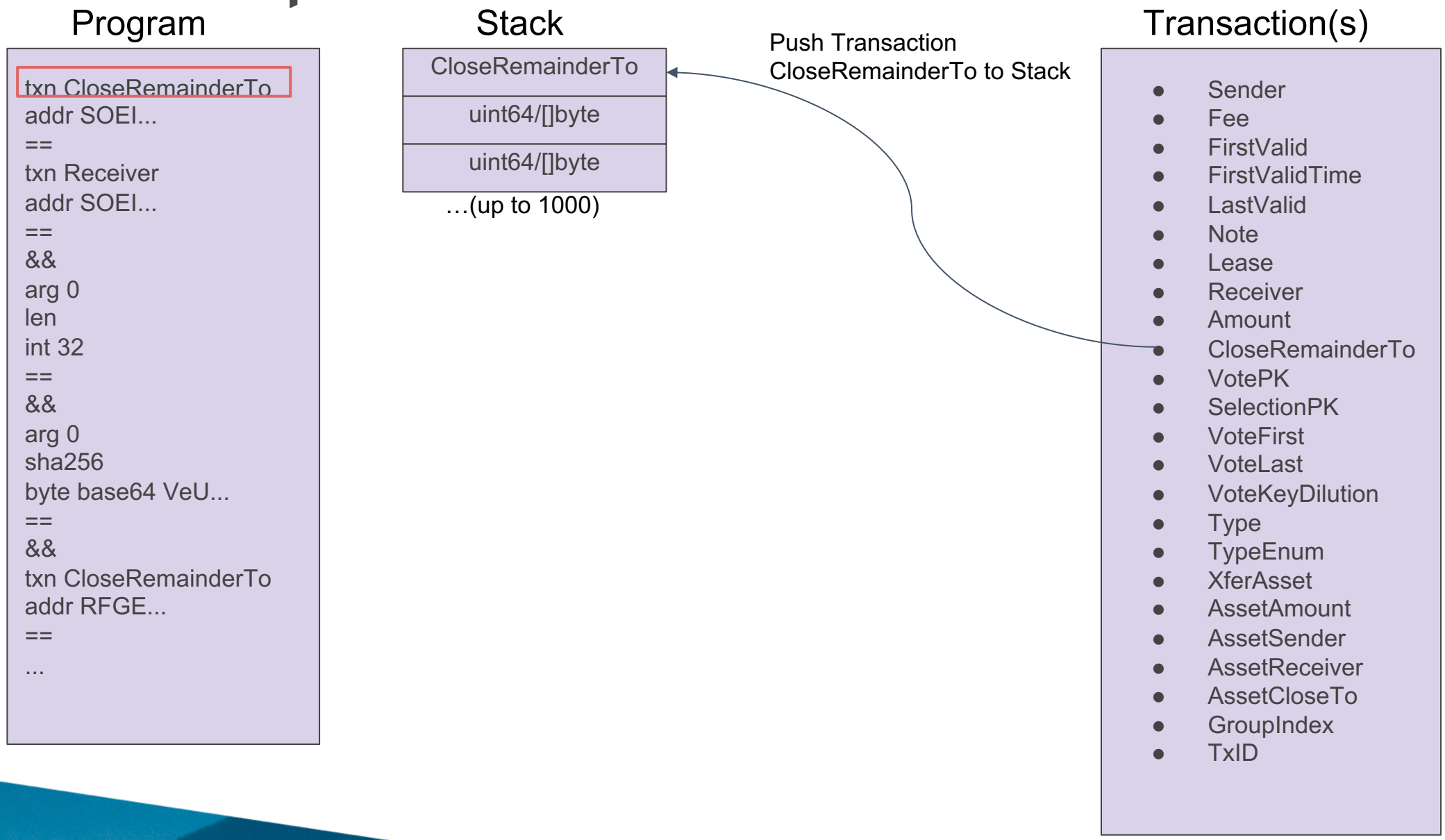
0: []byte
1: []byte
2: []byte

…(up to 255)

Transaction(s)

- Sender
- Fee
- FirstValid
- FirstValidTime
- LastValid
- Note
- Lease
- Receiver
- Amount
- CloseRemainderTo
- VotePK
- SelectionPK
- VoteFirst
- VoteLast
- VoteKeyDilution
- Type
- TypeEnum
- XferAsset
- AssetAmount
- AssetSender
- AssetReceiver
- AssetCloseTo
- GroupIndex
- TxID

TEAL Example

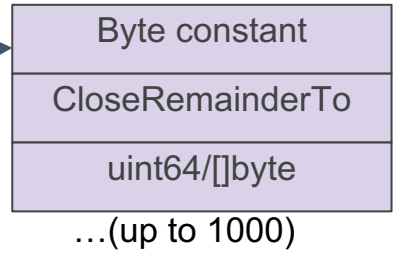


TEAL Example

Program

```
txn CloseRemainderTo
addr SOEI...
==
txn Receiver
addr SOEI...
==
&&
arg 0
len
int 32
==
&&
arg 0
sha256
byte base64 VeU...
==
&&
txn CloseRemainderTo
addr RFGE...
==
...
```

Stack



Convert address to byte constant and push to the stack

Transaction(s)

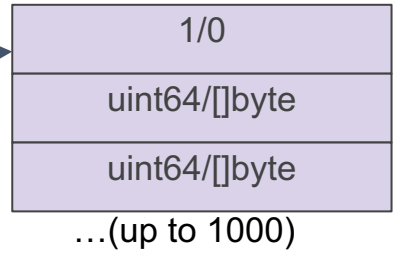
- Sender
- Fee
- FirstValid
- FirstValidTime
- LastValid
- Note
- Lease
- Receiver
- Amount
- CloseRemainderTo
- VotePK
- SelectionPK
- VoteFirst
- VoteLast
- VoteKeyDilution
- Type
- TypeEnum
- XferAsset
- AssetAmount
- AssetSender
- AssetReceiver
- AssetCloseTo
- GroupIndex
- TxID

TEAL Example

Program

```
txn CloseRemainderTo
addr SOEI...
==
txn Receiver
addr SOEI...
==
&&
arg 0
len
int 32
==
&&
arg 0
sha256
byte base64 VeU...
==
&&
txn CloseRemainderTo
addr RFGE...
==
...
```

Stack



Pops the top two values off the stack and replaces with 1 or 0 depending on if they were equal

Transaction(s)

- Sender
- Fee
- FirstValid
- FirstValidTime
- LastValid
- Note
- Lease
- Receiver
- Amount
- CloseRemainderTo
- VotePK
- SelectionPK
- VoteFirst
- VoteLast
- VoteKeyDilution
- Type
- TypeEnum
- XferAsset
- AssetAmount
- AssetSender
- AssetReceiver
- AssetCloseTo
- GroupIndex
- TxID

Layer-1 Stateless Smart Contract

A Puzzle

- Only approve/reject transactions
- Cannot directly store state
- But can use Algorand Standard Assets as state

   much more expressive than they look

- Example: Dutch auction
  - Puzzle: Find how to do it!

Layer-1 Stateful Smart Contracts

Introduction

- Applications on the blockchain
- Read & save state
- Written in the same stack-based language: TEAL
  - No loop, restriction on size and number of cryptographic operations
  - Restriction on storage: constant-size for global state & constant-size per account
- Same transaction fee as normal transaction!
  - Same latency (1 block every 4.5s), same throughput (1MB block)
- Can be combined with stateless smart contracts to hold Algos & assets

Layer-1 Stateful Smart Contracts

Examples

- Vote:
  - Store globally the tally
  - Store in each account that votes what they voted for
- Crowdfunding:
  - Users can fund a project
  - If funding goal is not reached, funds are reimbursed to users
- AlgoSwap: (equivalent of UniSwap)
  - Decentralized exchange
  - Price is determined by the liquidity provided by users

Layer-1 Features

Conclusion

- Layer-1 Smart Contracts,
  Algorand Standard Assets,
  Atomic Transfers
- Sufficient for many applications
  - Stable coins, rewards, tokenization, …
  - Simple market, voting, crowdfunding, simple decentralized exchange, …
- Same transaction fee, same latency (< 4.5s), same throughput as Algos transactions
- More expressive than they may appear: how expressive? Open question
- But sometimes, not convenient enough

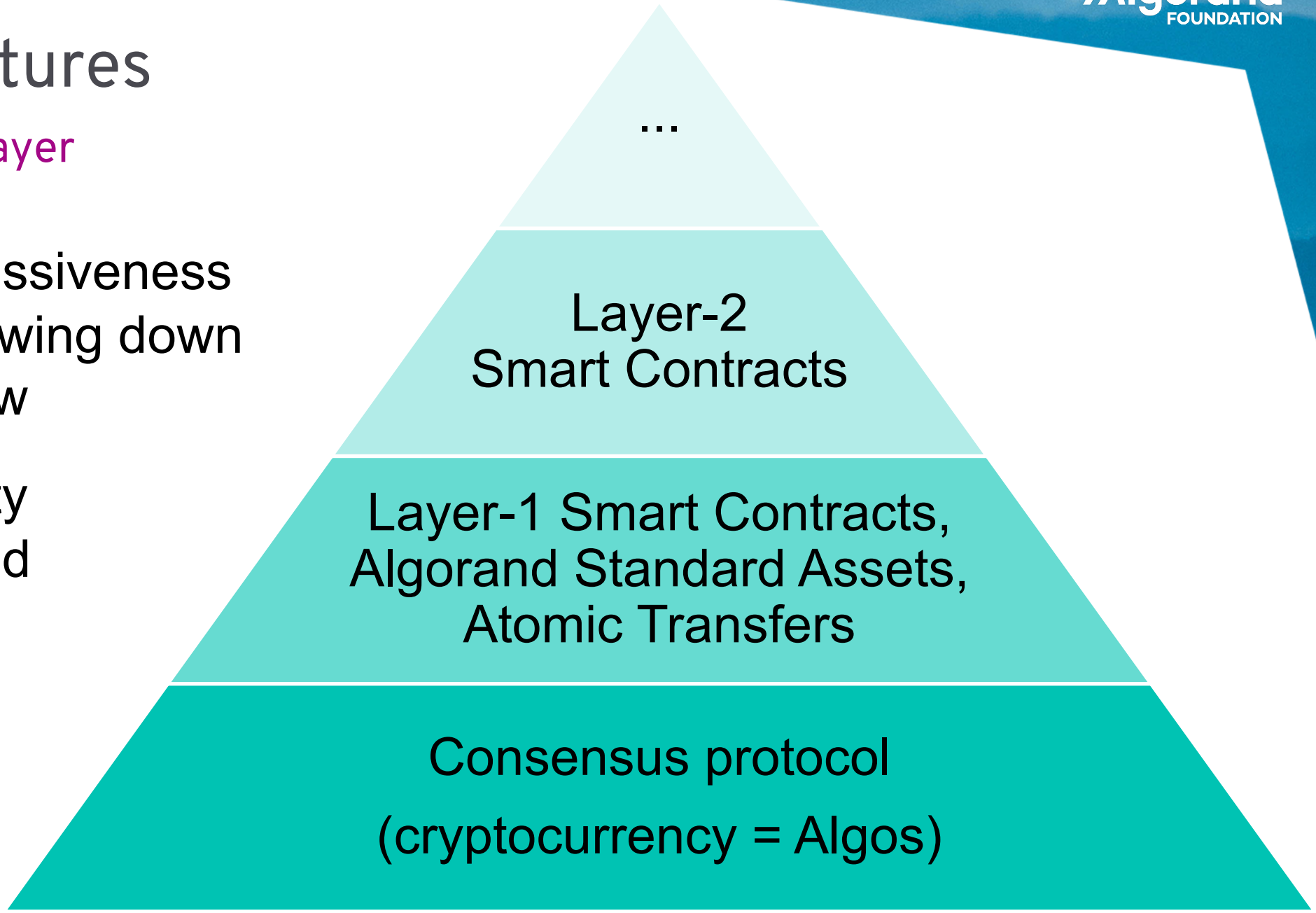
Algorand Features

Building layer by layer

1. More expressiveness
2. Without slowing down layers below

1. Same safety
2. Same speed
3. Same cost

Solving the Blockchain Trilemma



Layer-2 Smart Contracts (Work in Progress)

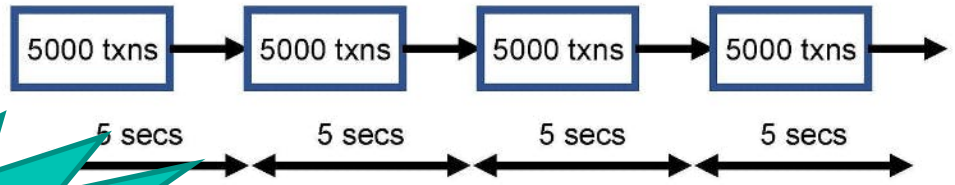
Goal

- For contracts that are:
  - Potentially too complex to write with layer-1: need for higher-level language
  - Too computationally demanding: e.g., zkSNARK
- Goal: Allow such contracts while:
  - Not slowing down the blockchain (latency & throughput)
  - Keeping the blockchain secure

Layer-2 Smart Contracts (Work in Progress)

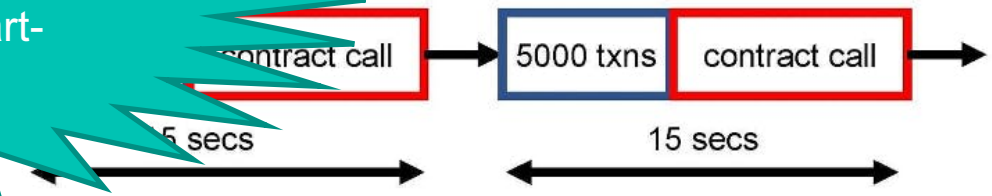
Architecture

- Layer-1 currently:



Learn more
https://www.algorand.com/resources/blog/algorand-smart-contract-architecture

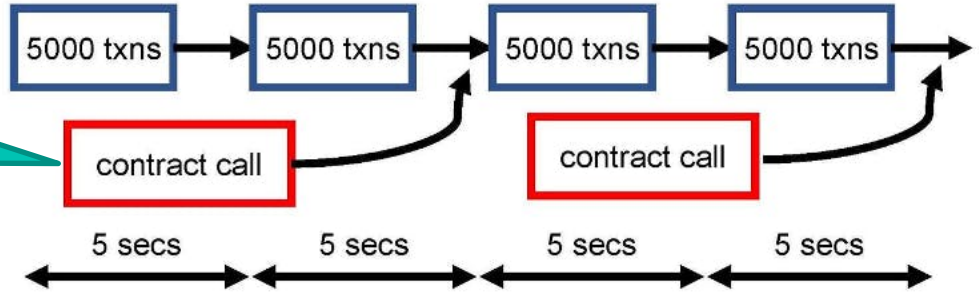
- Ethereum-like



- Algorand Layer-2 Smart Contracts

Contract Execution Committee

Selected as the
Consensus Committee using a VRF



Compact Certificates (Work in Progress)

- Allow other blockchains to efficiently check Algorand's blocks
  - With a short certificate
  - That does not require VRF or complex cryptographic tools
- Facilitate interoperability between blockchains
  - Example: replace hash-time lock contracts to transfer assets between chains

Start Building on Algorand

Many tools to start developing right now

- **Block explorer:** goalseeker.purestake.io, algoexplorer.io
- **Online tools:** algodesk.io — create your first token & first smart contract
- **Interactive tutorial:** algorand.rockx.com
- Official **SDK:** JS, Python, Go, Java + community SDK: C#, Rust, …
- **IDE:** VSCode, IntelliJ Idea, Algorand Studio, …
- **Free API services:** algoexplorer.io, purestake.io (equivalent to Infura)
- **Wallet for DApps:** AlgoSigner (equivalent of MetaMask)
- **Simplify DApp writing:** reach.sh (bonus: same code works on Ethereum)
- Automate development of smart contracts and assets: Algorand Builder

Start Building on Algorand

Resources

- Visit https://developer.algorand.org
  - Getting started article: https://developer.algorand.org/articles/getting-started-algorand
  - Tutorials, solutions, …
  - Full documentation
  - Source code: https://github.com/algorand
- Questions:
  - Discord server: https://discord.gg/YgPTCVk
  - Q&A: https://forum.algorand.org
  - Office hours: https://www.algorand.com/developers

Join the Algorand Community!

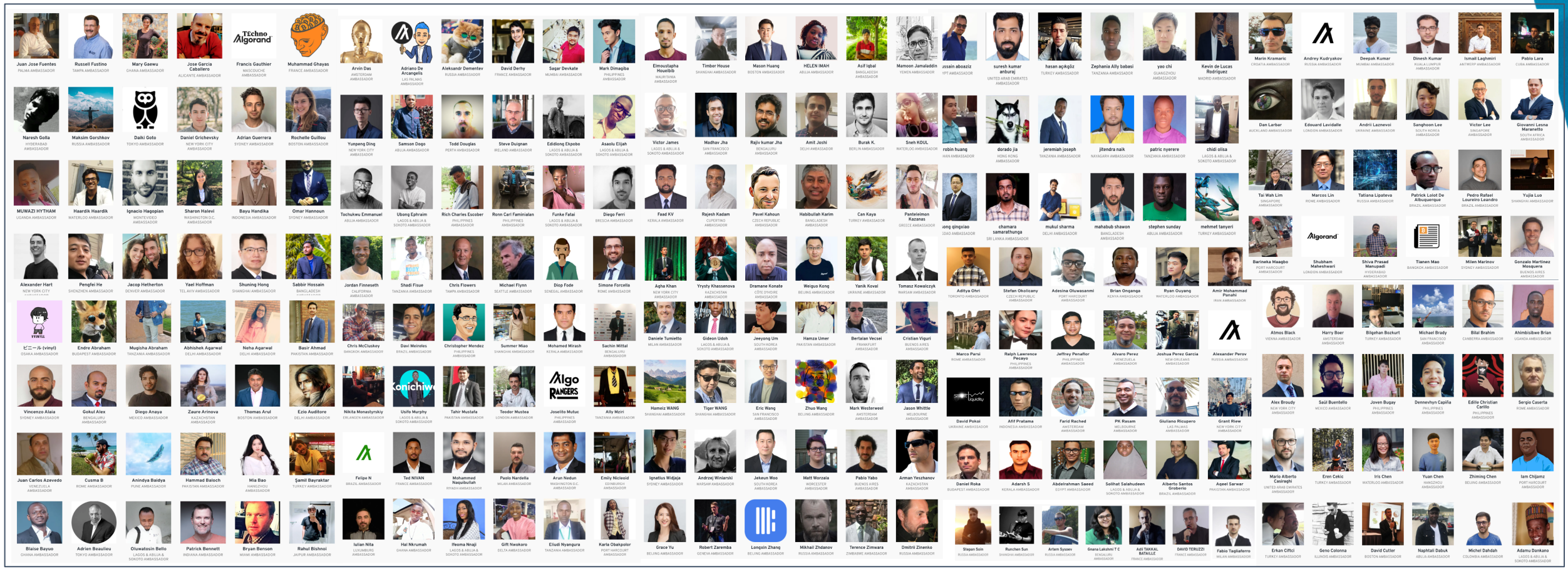
Contribute and get rewards!

- Become an Algorand ambassador
  - https://algorand.foundation/2020-ambassador-rewards-program
- Write tutorials and articles for Algorand (devAmbassador):
  - https://algorand.foundation/dev-ambassadors
- Get bounties:
  - https://github.com/algorandfoundation/grow-algorand
- Development awards for a tool / application you developed
  - https://algorand.foundation/developer-incentive-awards-program
- Apply for a grant
  - https://algorand.foundation/grants-program
- Join a pre-accelerator or an accelerator
  - https://algorand.foundation/ecosystem/accelerator

Algorand FOUNDATION

Join the Algorand Community!

400+ Ambassadors From 66+ countries




THANK
YOU