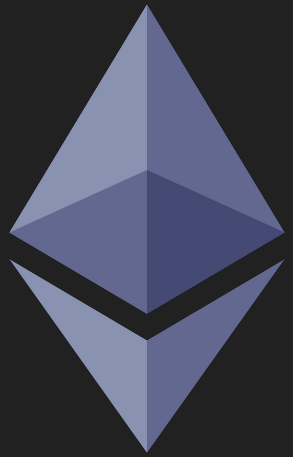
Ethereum 2.0

Prysmatic Labs

Preston Van Loon
@preston_vanloon

Prysmatic Labs

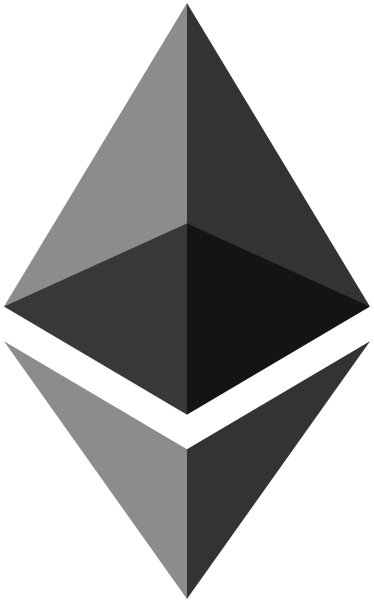


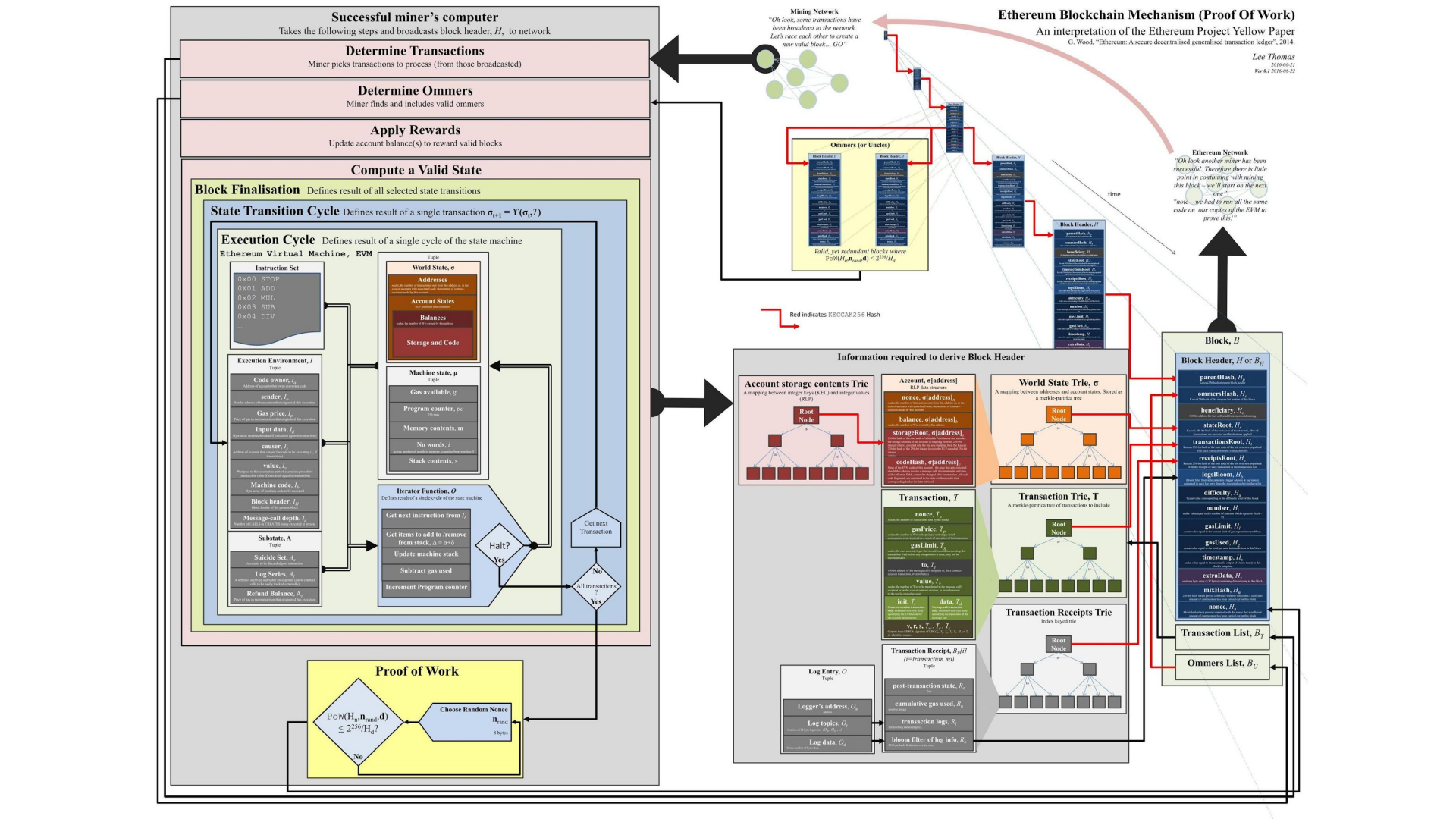
Implementing Ethereum Serenity
with Proof of Stake + Sharding

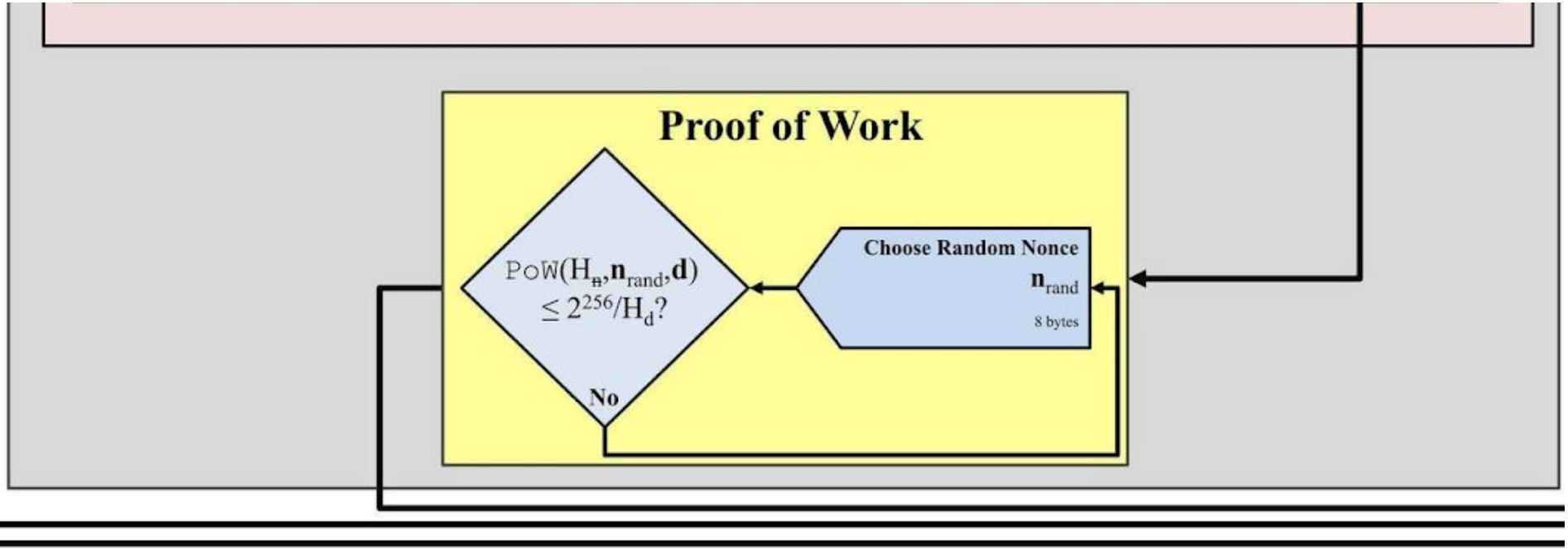
What is Ethereum?

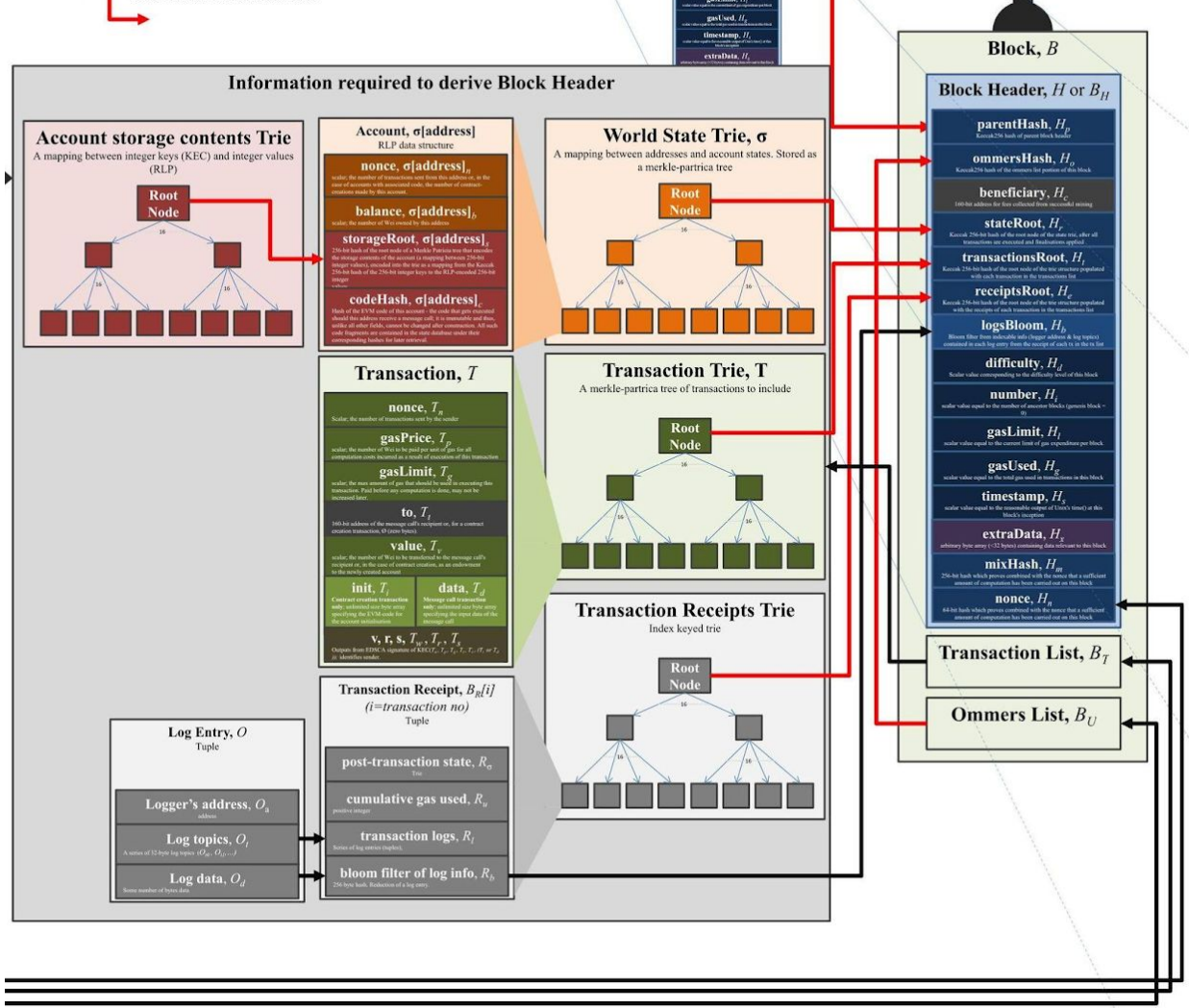
Ethereum - A Decentralized World Computer

- Open source blockchain
- Decentralized global virtual machine
- Consisting of tens of thousands of nodes
- Unlimited possibility of use cases
  - DAOs
  - ERC Tokens
  - DApps




Successful miner's computer
Takes the following steps and broadcasts block header, H, to network
Determine Transactions
Miner picks transactions to process (from those broadcasted)
Determine Ommers
Miner finds and includes valid ommers
Apply Rewards
Update account balance(s) to reward valid blocks
Compute a Valid State
Block Finalisation Defines result of all selected state transitions
State Transition Cycle Defines result of a single transaction σt+1 = Υ(σt,T)
Execution Cycle Defines result of a single cycle of the state machine
Ethereum Virtual Machine, EVM
Instruction Set
0x00 STOP
0x01 ADD
0x02 MUL
0x03 SUB
0x04 DIV
Execution Environment, I
Tuple
Code owner, Ia
Address of account that owns executing code
sender, Io
Sender address of transaction that originated this execution
Gas price, Ip
Price of gas in the transaction that originated this execution
Input data, Id
Byte array (transaction data if execution agent is transaction)
causer, Is
Address of account that caused the code to be executing (I, if
value, Iv
Wei pass to this account as part of execution procedure
(transaction value, if execution agent is transaction)
Machine code, Ib
Byte array of machine code to be executed
Block header, IH
Block header of the present block
Message-call depth, Ie
Number of CALLs or CREATEs being executed at present
Substate, A
Tuple
Suicide Set, As
Accounts to be discarded post-transaction
Log Series, Al
A series of archival indexable checkpoints (allow contract
author to efficiently include messages)
Refund Balance, Ar
Price of gas in the transaction that originated this execution
Tuple
World State, σ
Addresses
Account States
Balances
Storage and Code
Machine state, μ
Tuple
Gas available, g
Program counter, pc
Memory contents, m
No words, i
Stack contents, s
Iterator Function, O
Defines result of a single cycle of the state machine
Get next instruction from Ii
Get items to add to /remove from stack, Δ = α+δ
Update machine stack
Subtract gas used
Increment Program counter
Halt?
Yes
No
Get next Transaction
All transactions ?
Yes
No
Proof of Work
PoW(Hn,nrand,d) ≤ 2256/Hd?
No
Choose Random Nonce
nrand
8 bytes
Mining Network
"Oh look, some transactions have been broadcast to the network. Let's race each other to create a new valid block... GO"
Ethereum Blockchain Mechanism (Proof Of Work)
An interpretation of the Ethereum Project Yellow Paper
G. Wood, "Ethereum: A secure decentralised generalised transaction ledger", 2014.
Lee Thomas
2016-06-21
Ver 0.1 2016-06-22
Ommers (or Uncles)
Valid, yet redundant blocks where PoW(Hn,nrand,d) < 2256/Hd
Block Header, H
time
Ethereum Network
"Oh look another miner has been successful. Therefore there is little point in continuing with mining this block – we'll start on the next one."
"note – we had to run all the same code on our copies of the EVM to prove this!"
Red indicates KECCAK256 Hash
Information required to derive Block Header
Account storage contents Trie
A mapping between integer keys (KEC) and integer values (RLP)
Root Node
Account, σ[address]
RLP data structure
nonce, σ[address]n
balance, σ[address]b
storageRoot, σ[address]s
codeHash, σ[address]c
World State Trie, σ
A mapping between addresses and account states. Stored as a merkle-patricia tree
Transaction, T
nonce, Tn
gasPrice, Tp
gasLimit, Tg
to, Tt
value, Tv
init, Ti
data, Td
v, r, s, Tw, Tr, Ts
Transaction Trie, T
A merkle-partrica tree of transactions to include
Transaction Receipts Trie
Index keyed trie
Transaction Receipt, BR[i]
(i=transaction no)
Tuple
post-transaction state, Rσ
cumulative gas used, Rg
transaction logs, Rl
bloom filter of log info, Rb
Log Entry, O
Tuple
Logger's address, Oa
Log topics, Ot
Log data, Od
Block, B
Block Header, H or BH
parentHash, HP
ommersHash, HO
beneficiary, HC
stateRoot, HR
transactionsRoot, HT
receiptsRoot, HE
logsBloom, HB
difficulty, HD
number, HI
gasLimit, HL
gasUsed, HG
timestamp, HS
extraData, HX
mixHash, Hm
nonce, Hn
Transaction List, BT
Ommers List, BU


Proof of Work
PoW(H_n, n_rand, d) ≤ 2^256/H_d?
No
Choose Random Nonce
n_rand
8 bytes


gasUsed, H_g
timestamp, H_s
extraData, H_x
Block, B
Block Header, H or B_H
parentHash, H_p
Keccak256 hash of parent block header
ommersHash, H_O
Keccak256 hash of the ommers list portion of this block
beneficiary, H_c
160-bit address for fees collected from successful mining
stateRoot, H_r
Keccak 256-bit hash of the root node of the state trie, after all transactions are executed and finalisations applied
transactionsRoot, H_t
Keccak 256-bit hash of the root node of the trie structure populated with each transaction in the transactions list
receiptsRoot, H_e
Keccak 256-bit hash of the root node of the trie structure populated with the receipts of each transaction in the transactions list
logsBloom, H_b
Bloom filter from indexable info (logger address & log topics) contained in each log entry from the receipt of each tx in the tx list
difficulty, H_d
Scalar value corresponding to the difficulty level of this block
number, H_i
scalar value equal to the number of ancestor blocks (genesis block = 0)
gasLimit, H_l
scalar value equal to the current limit of gas expenditure per block
gasUsed, H_g
scalar value equal to the total gas used in transactions in this block
timestamp, H_s
scalar value equal to the reasonable output of Unix's time() at this block's inception
extraData, H_x
arbitrary byte array (<32 bytes) containing data relevant to this block
mixHash, H_m
256-bit hash which proves combined with the nonce that a sufficient amount of computation has been carried out on this block
nonce, H_n
64-bit hash which proves combined with the nonce that a sufficient amount of computation has been carried out on this block
Transaction List, B_T
Ommers List, B_U
Information required to derive Block Header
Account storage contents Trie
A mapping between integer keys (KEC) and integer values (RLP)
Root Node
16
Account, σ[address]
RLP data structure
nonce, σ[address]$_n$
balance, σ[address]$_b$
storageRoot, σ[address]$_s$
codeHash, σ[address]$_c$
World State Trie, σ
A mapping between addresses and account states. Stored as a merkle-partrica tree
Transaction, T
nonce, T_n
Scalar; the number of transactions sent by the sender
gasPrice, T_p
gasLimit, T_g
to, T_t
160-bit address of the message call's recipient or, for a contract creation transaction, Ø (zero bytes).
value, T_v
init, T_i
Contract creation transaction only: unlimited size byte array specifying the EVM-code for the account initialisation
data, T_d
Message call transaction only: unlimited size byte array specifying the input data of the message call
v, r, s, T_w, T_r, T_s
Outputs from ECDSA signature of KEC(T_n, T_p, T_g, T_t, T_v, T_i or T_d)
ii identifies sender.
Transaction Trie, T
A merkle-partrica tree of transactions to include
Transaction Receipts Trie
Index keyed trie
Transaction Receipt, $B_R[i]$
(i=transaction no)
Tuple
post-transaction state, R_o
cumulative gas used, R_u
transaction logs, R_l
bloom filter of log info, R_b
Log Entry, O
Tuple
Logger's address, O_a
Log topics, O_t
Log data, O_d

What Does It Mean To Scale Ethereum?

Today’s Transaction Maximum Throughput

7 tx/s

Average

3 tx/s

27 tx/s

Average

12 tx/s

24,000+ tx/s

Average

1,667 tx/s

Today’s Blocktimes

10 minutes

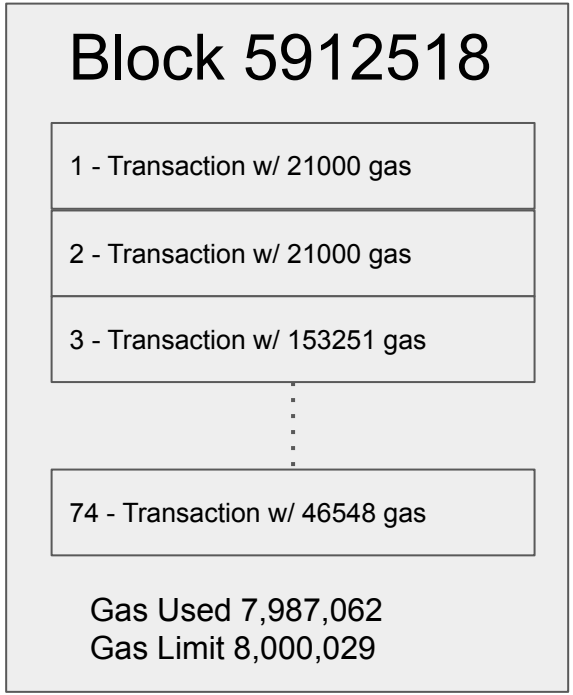
14 seconds

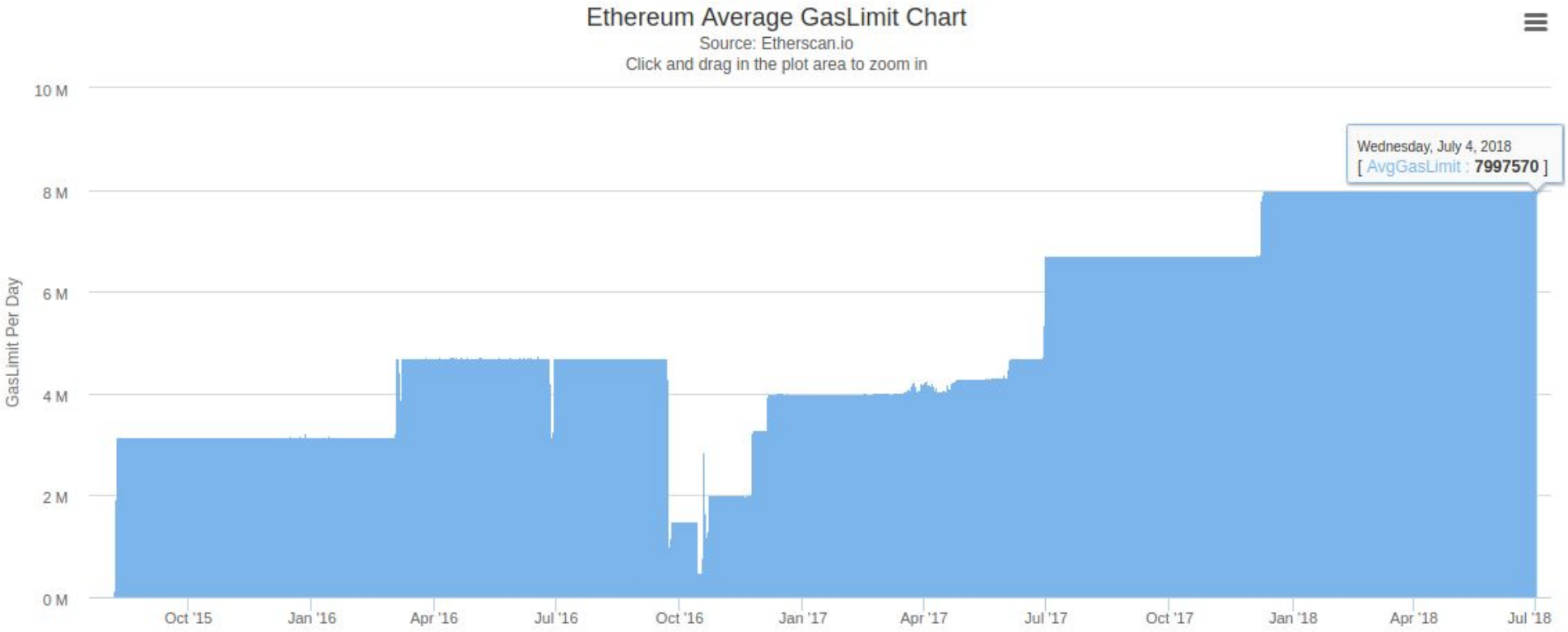
VISA

?

How Transactions Fit Into Blocks

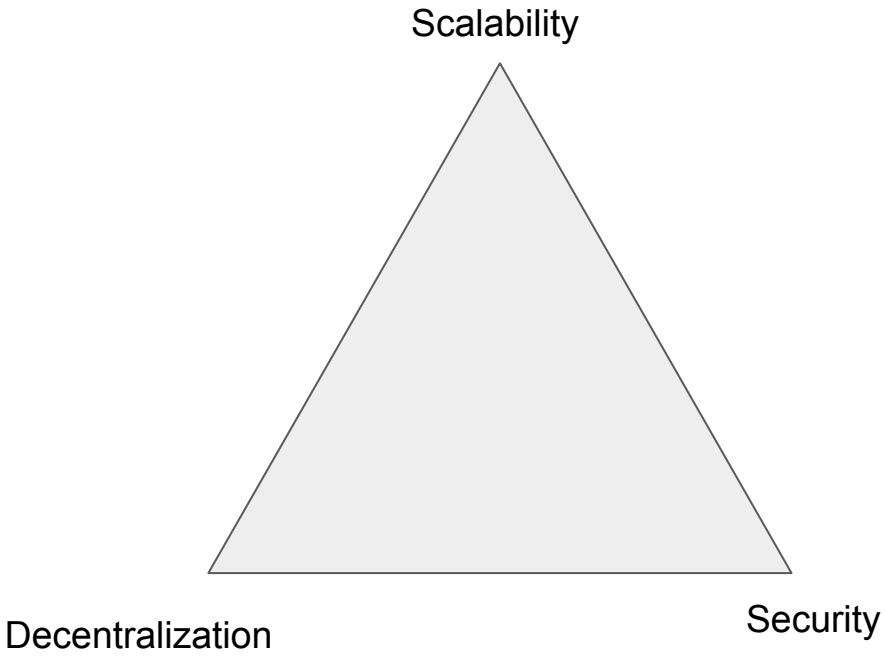
- Blocks typically consist of the highest paying transactions that fit within a block gas limit
- Miners mine transactions and collect gas fees
- Miners vote on the gas limit
- Current default algorithm for gas limit calculation is at least 4.7M but targeting 150% of recent 1024 block exponential moving average. Changes are limited by a factor of 1/1024 in either direction.




Ethereum Average GasLimit Chart
Source: Etherscan.io
Click and drag in the plot area to zoom in
GasLimit Per Day
10 M
8 M
6 M
4 M
2 M
0 M
Oct '15
Jan '16
Apr '16
Jul '16
Oct '16
Jan '17
Apr '17
Jul '17
Oct '17
Jan '18
Apr '18
Jul '18
Wednesday, July 4, 2018
[AvgGasLimit : 7997570]


How Can We Scale?

Blockchain Trilemma



Idea: Increase the Block Gas Limit!

Issues to consider

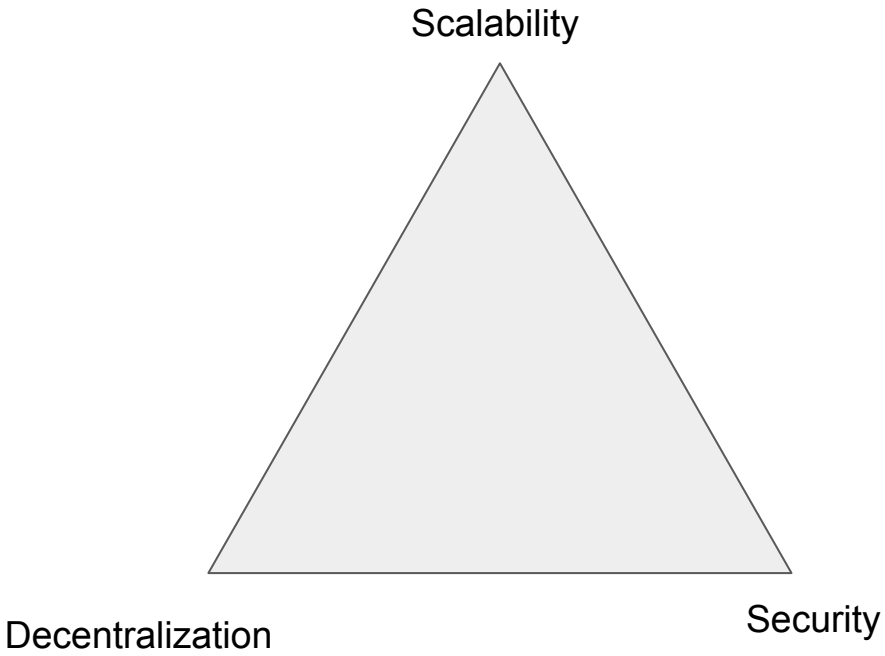
- Bigger blocks means each block requires more computational power
- Full nodes require more resources to verify blocks
- Less decentralized

Idea: Reduce the Time Between Blocks!

Issues to consider

- Fast blocks means higher probability of forks
- More forks makes blockchains vulnerable to attacks
- Less secure

Blockchain Trilemma



Two Types of Scaling Solutions

Layer 1 - On chain

- Higher throughput on the protocol layer
- More difficult to implement
- Satisfies the trilemma
- Benefits layer 2

Layer 2 - Off chain

- Higher throughput enabled by less on-chain operations
- Easier to implement
- More flexible and customizable
- May not be as secure or decentralized as layer 1


Ethereum 2.0

What is Ethereum 2.0?

> "A big, multi-year long, upgrade to massively increase the blockchain's scalability with **sharding**, increase security with **proof of stake**, and improve its **programmability** by changing a bunch of technical things we got wrong the first time."

– Vitalik Buterin, Creator of Ethereum

Phase 0
Beacon Chain

Validator Registry

- 1 way deposit via deposit contract
- 32 ETH minimum to join
- 18 ETH ejection balance
- Exits / Withdraws

Shuffling / Randomness

- Calculated during epoch transition
- RANDAO model
- Randomly distributed validator pool
- Verifiable delay function (soon™)

Beacon Chain

Reward / Penalties

- Calculated every epoch
- Validator slashing
- Liveness penalty
- Participation reward

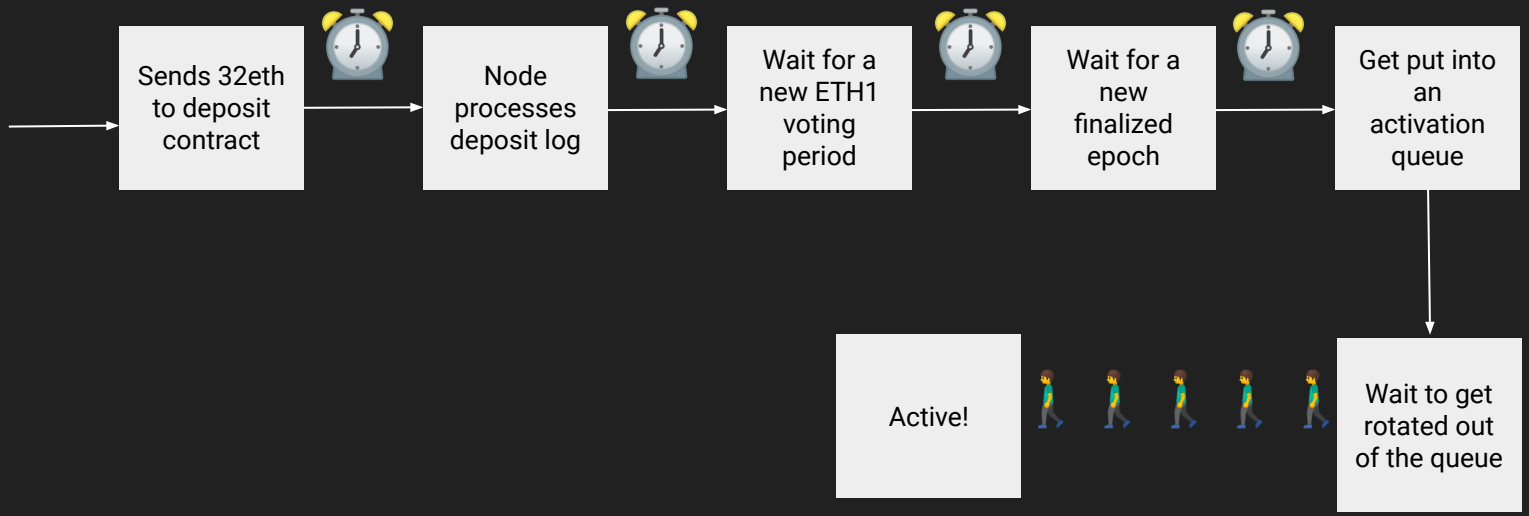
Proof of Stake Finalization

- Block justification via Casper FFG
- Allows finalization of ETH 1.x

Casper - Friendly Finality Gadget

- Validators have ETH at stake
- Energy efficient consensus mechanism
- Finalized checkpoints
- Lower barrier to entry

Becoming a Validator



Minimum activation time ~2.134 hours

Validator Responsibilities

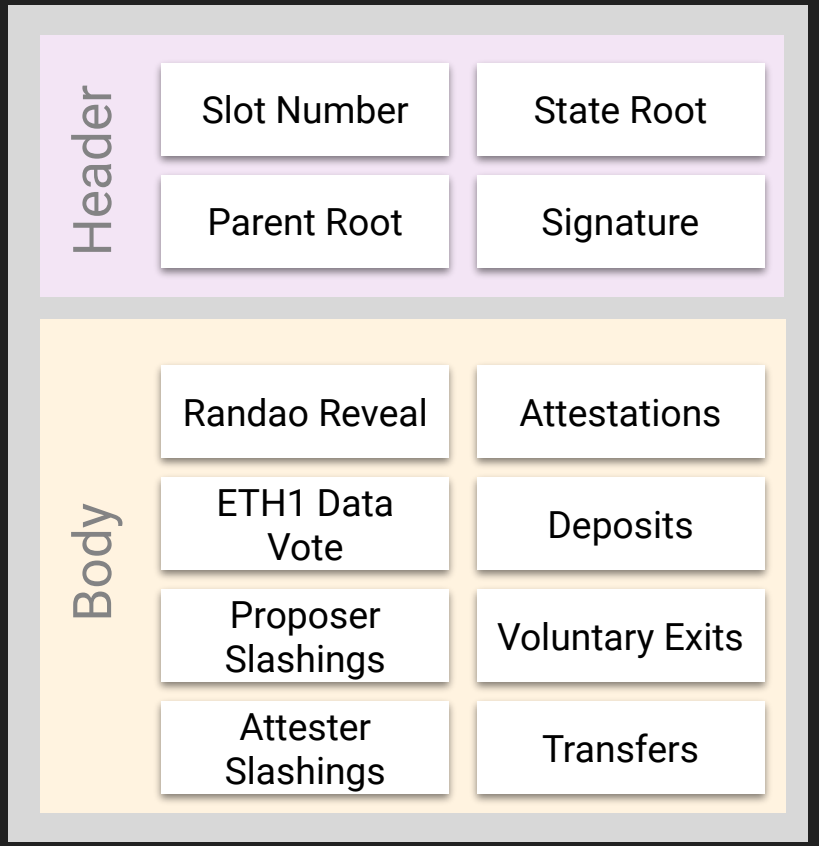
Proposer - A validator selected to create a beacon chain block

Attester - A validator that is part of the committee that creates attestation and creates crosslink to a recent shard block on a shard chain

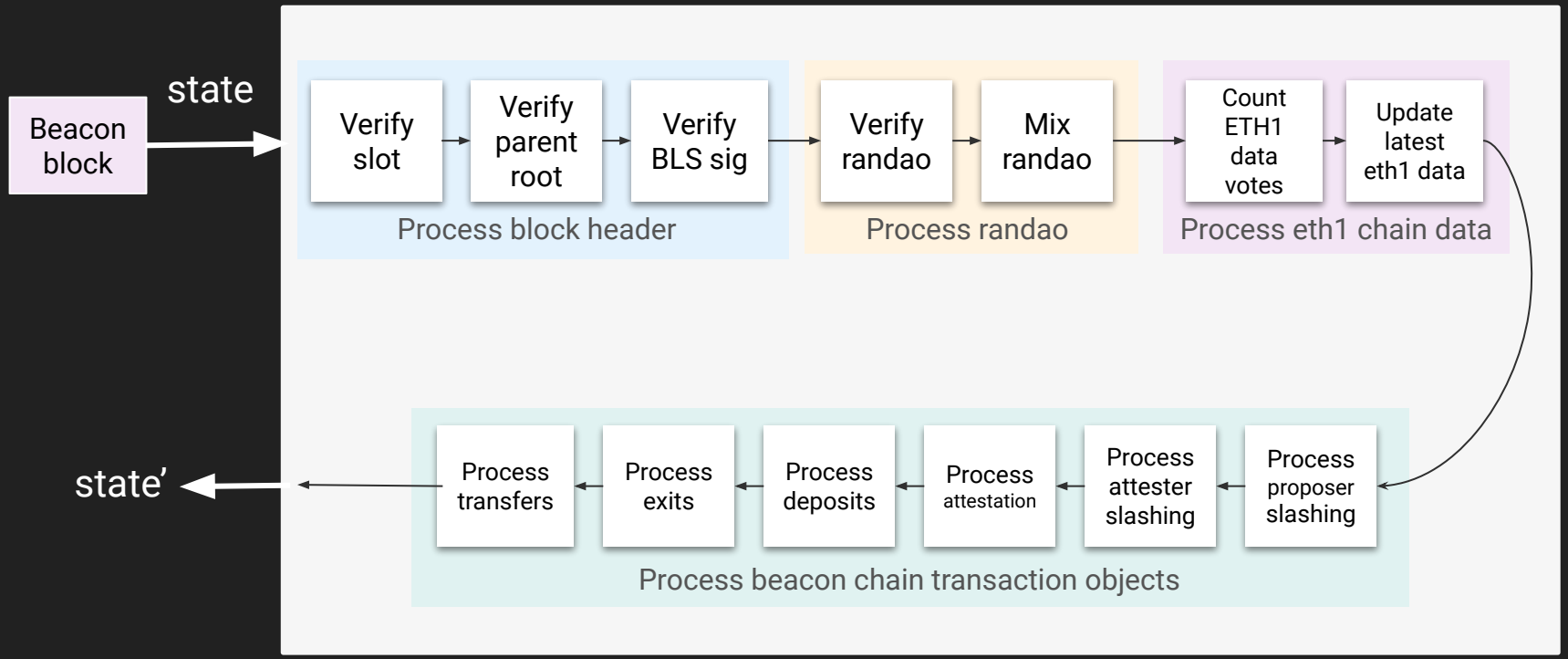
Committee - A randomly sampled subset of validators

Proposing a Beacon Block

1. Assemble the block body
2. Execute the state transition
3. Sign the block
4. Broadcast to network



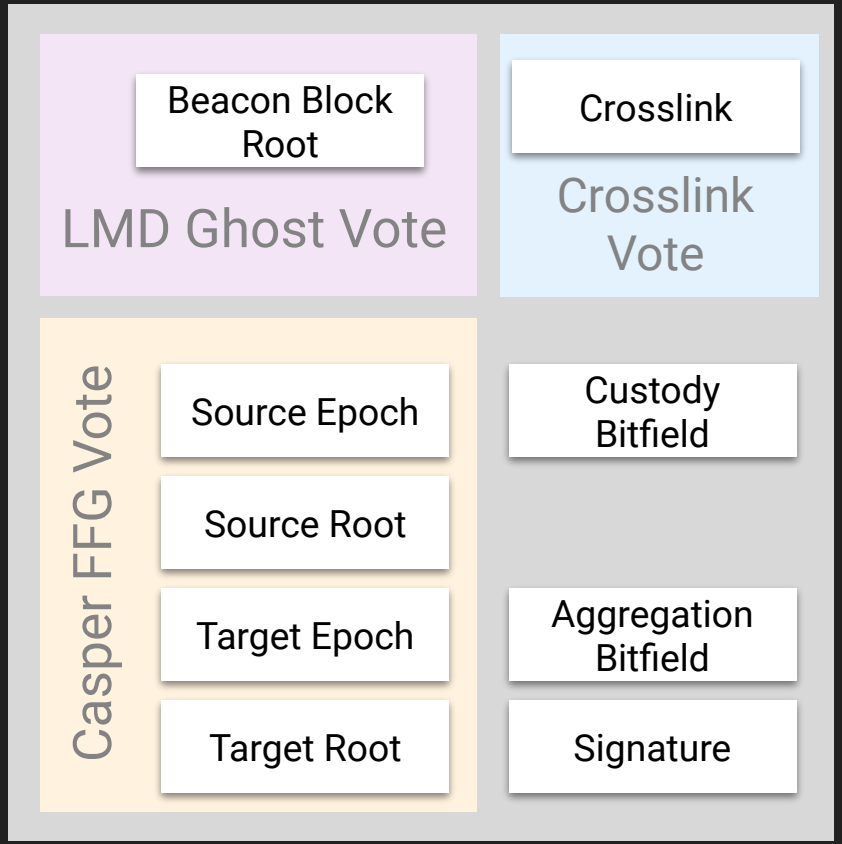
Beacon Block Processing



Attesting

1. Determine Casper FFG,

   Crosslink, and LMD Ghost votes
2. Aggregate similar attestations
3. Sign the attestation
4. Broadcast to network



Validator Rewards and Penalties

- Proposing a block yields higher reward than attestations
- Rewards and penalties are calculated every epoch
- Penalties increase exponentially when finality has not occured for more than four epochs

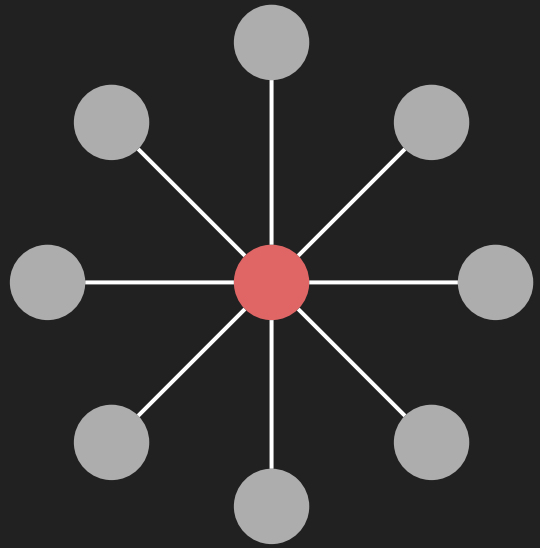
Total ETH validating	Max annual issuance	Max annual network issuance	Max annual return rate
1MM	181,019	0.17%	18.10%
3MM	313,534	0.30%	10.45%
10MM	572,433	0.54%	5.72%
30MM	991,483	0.94%	3.30%
100MM	1,810,193	1.71%	1.81%

2,097,152 ETH required to start ETH 2.0

Phase 1
Shard Chains

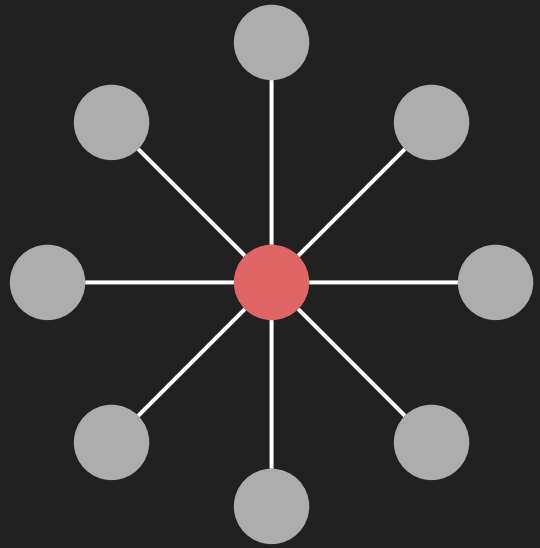
Shard Chains

- Introduces the parallel shard chains
- 64 shards, data only
- Shard chains are linked to the beacon chain by crosslinks once per epoch
- Expected to come to consensus on 10Mb/s of data



Use Cases

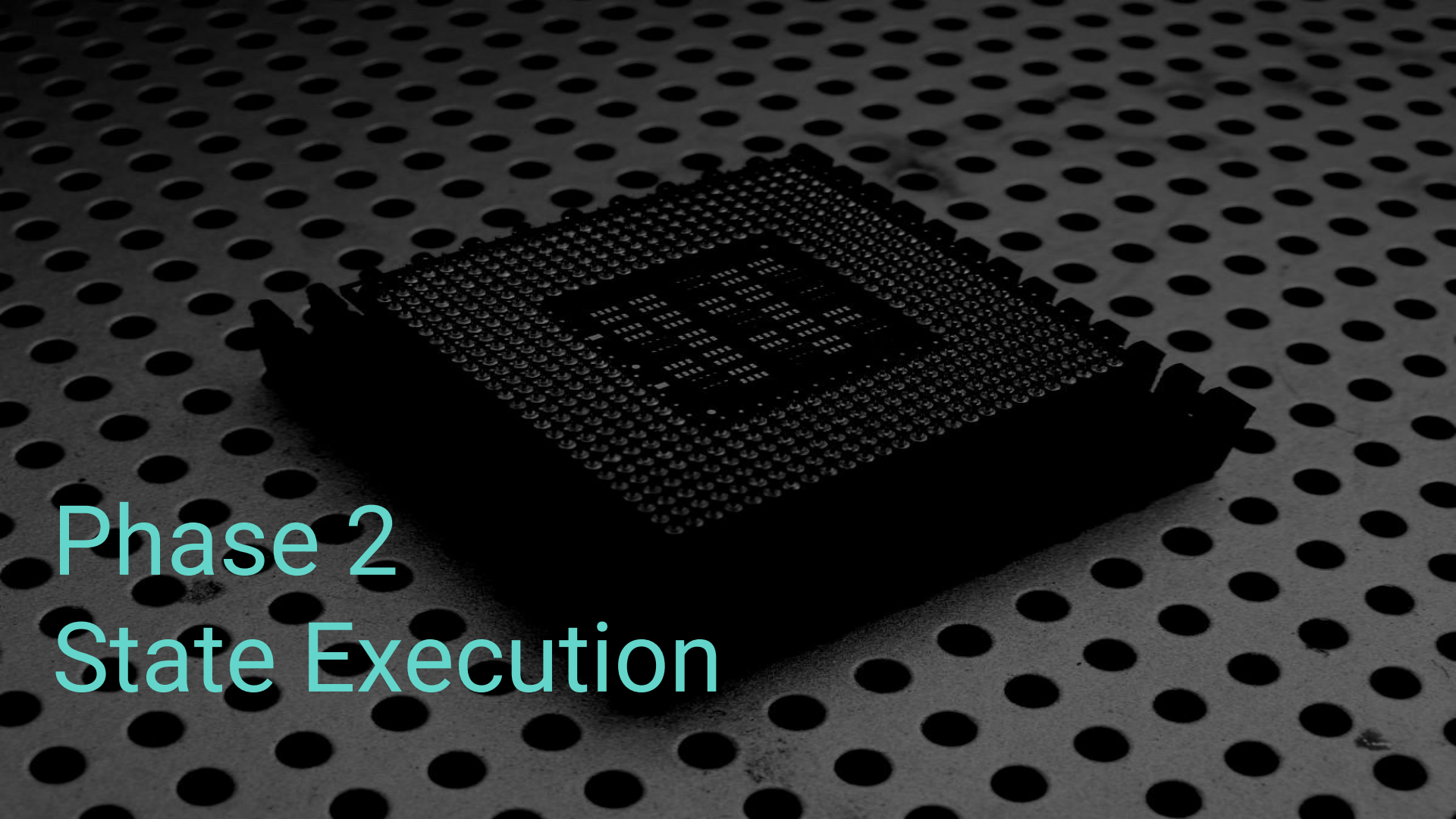
- ZK Rollup
- ZK Rollup Rollup
- Decentralized twitter
- GPG key server
- Website hosting
- Data layer for private/enterprise blockchains
- Generalized small / medium amounts of storage




Phase 1.5
Merging eth1 & eth2

Phase 1.5

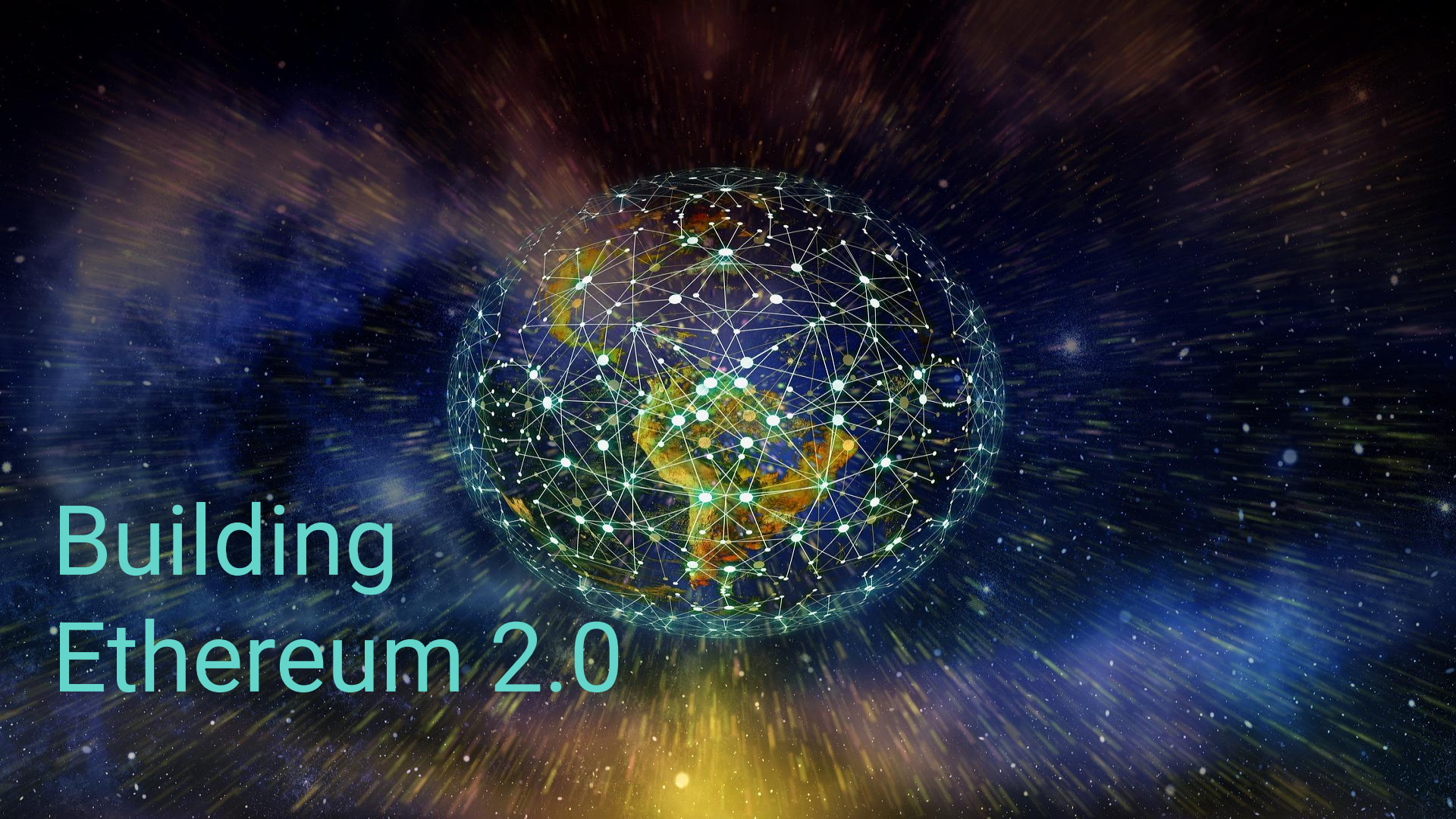
- Until phase 1.5, the Ethereum we use today on mainnet will continue as a proof-of-work blockchain and transactions will continue to be processed by miners
- Starting in phase 1.5, eth1 will officially become a shard and transition to proof-of-stake
- For end users and dapps, this change should be **seamless**



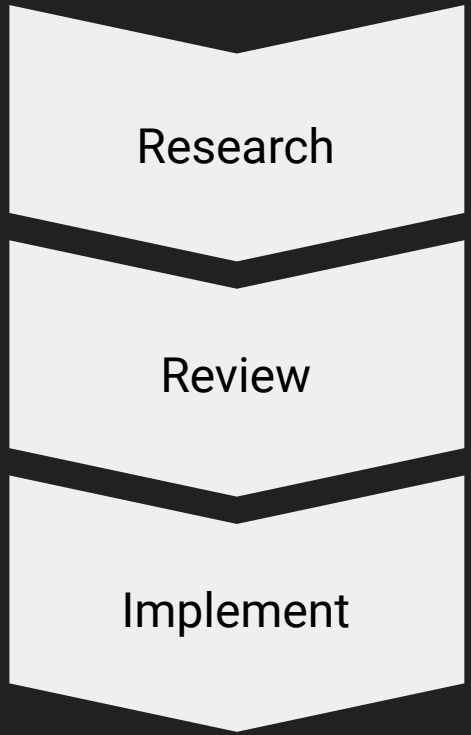
Phase 2
State Execution

State Execution

- Replace EVM with eWASM
- Asynchronous cross shard transactions
- Contract yanking (migrating shards)
- Ethereum 2.0 becomes useful to average contract developer / users
- In research and design phase, development likely to start early 2020
- Development can start in parallel to phase 0 and phase 1


Building
Ethereum 2.0

From Research to Implementation



- Explore new ideas
- Collaborate on ethrear.ch, in person, online channels
- Propose changes to the Ethereum 2.0 specification

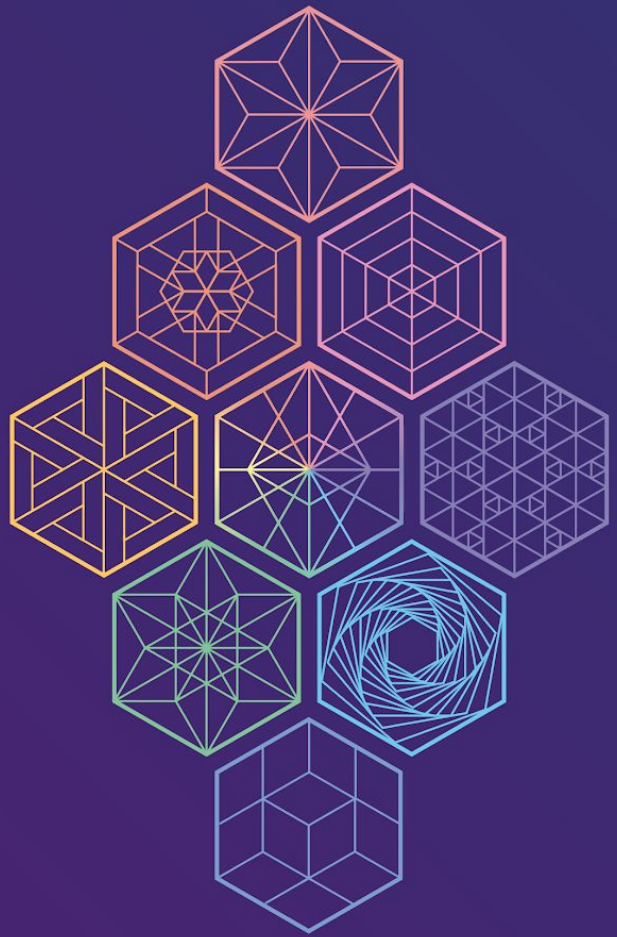
- The spec changes are reviewed by other researchers and implementation teams
- Spec release targets are tagged

- Implementation teams design new features
- Features are proposed in github and reviewed within the team

Prysm Feature Lifecycle

- Design document
- Tracking issues
- Implementation
- Pull request review
- Canary analysis / testing
- Release

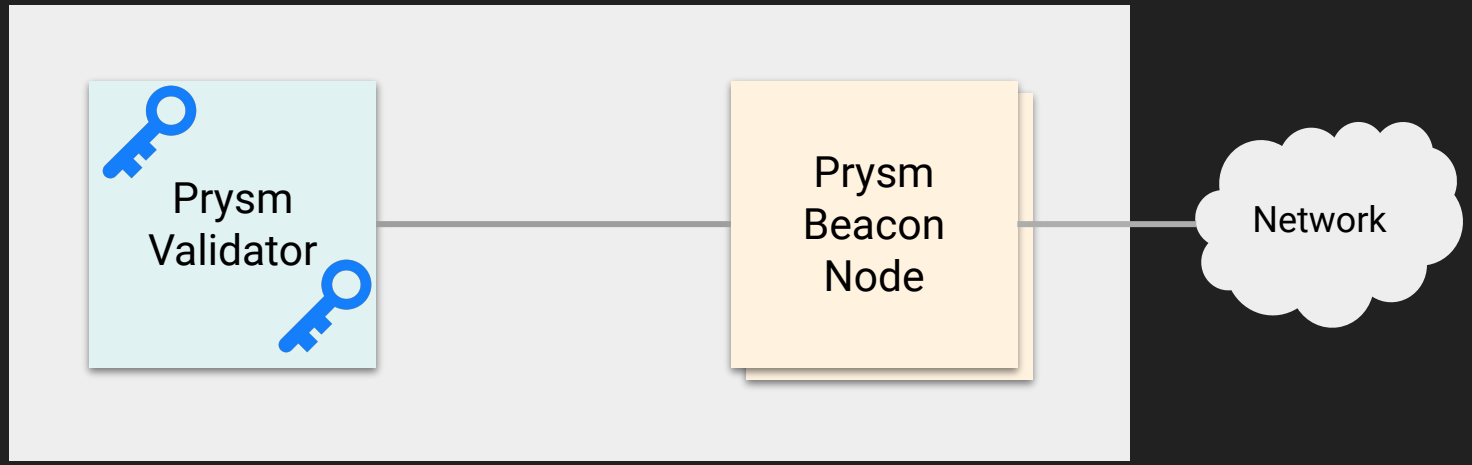
```
                        // be greater than 2/3 of the total balance.
                        if 3*attestedBalance >= 2*totalBalance {
                                state.CurrentCrosslinks[shard] = crosslink
                        }
                }
        }
        return state, nil
}

// ProcessRewardsAndPenalties processes the rewards and penalties of individual validator.
//
// Spec pseudocode definition:
//  def process_rewards_and_penalties(state: BeaconState) -> None:
//    if get_current_epoch(state) == GENESIS_EPOCH:
//        return
//
//    rewards1, penalties1 = get_attestation_deltas(state)
//    rewards2, penalties2 = get_crosslink_deltas(state)
//    for i in range(len(state.validator_registry)):
//        increase_balance(state, i, rewards1[i] + rewards2[i])
//        decrease_balance(state, i, penalties1[i] + penalties2[i])
func ProcessRewardsAndPenalties(state *pb.BeaconState) (*pb.BeaconState, error) {
        // Can't process rewards and penalties in genesis epoch.
        if helpers.CurrentEpoch(state) == 0 {
                return state, nil
        }
        attsRewards, attsPenalties, err := attestationDelta(state)
        if err != nil {
                return nil, errors.Wrap(err, "could not get attestation delta")
        }
        clRewards, clPenalties, err := crosslinkDelta(state)
        if err != nil {
                return nil, errors.Wrapf(err, "could not get crosslink delta")
        }
        for i := 0; i < len(state.Validators); i++ {
                state = helpers.IncreaseBalance(state, uint64(i), attsRewards[i]+clRewards[i])
                state = helpers.DecreaseBalance(state, uint64(i), attsPenalties[i]+clPenalties
        }
        return state, nil
}

// ProcessRegistryUpdates rotates validators in and out of active pool.
// the amount to rotate is determined churn limit.
//
// Spec pseudocode definition:
//   def process_registry_updates(state: BeaconState) -> None:
//    # Process activation eligibility and ejections
//    for index, validator in enumerate(state.validator_registry):
//        if (
//            validator.activation_eligibility_epoch == FAR_FUTURE_EPOCH and
//            validator.effective_balance >= MAX_EFFECTIVE_BALANCE
//        ):
//            validator.activation_eligibility_epoch = get_current_epoch(state)
```

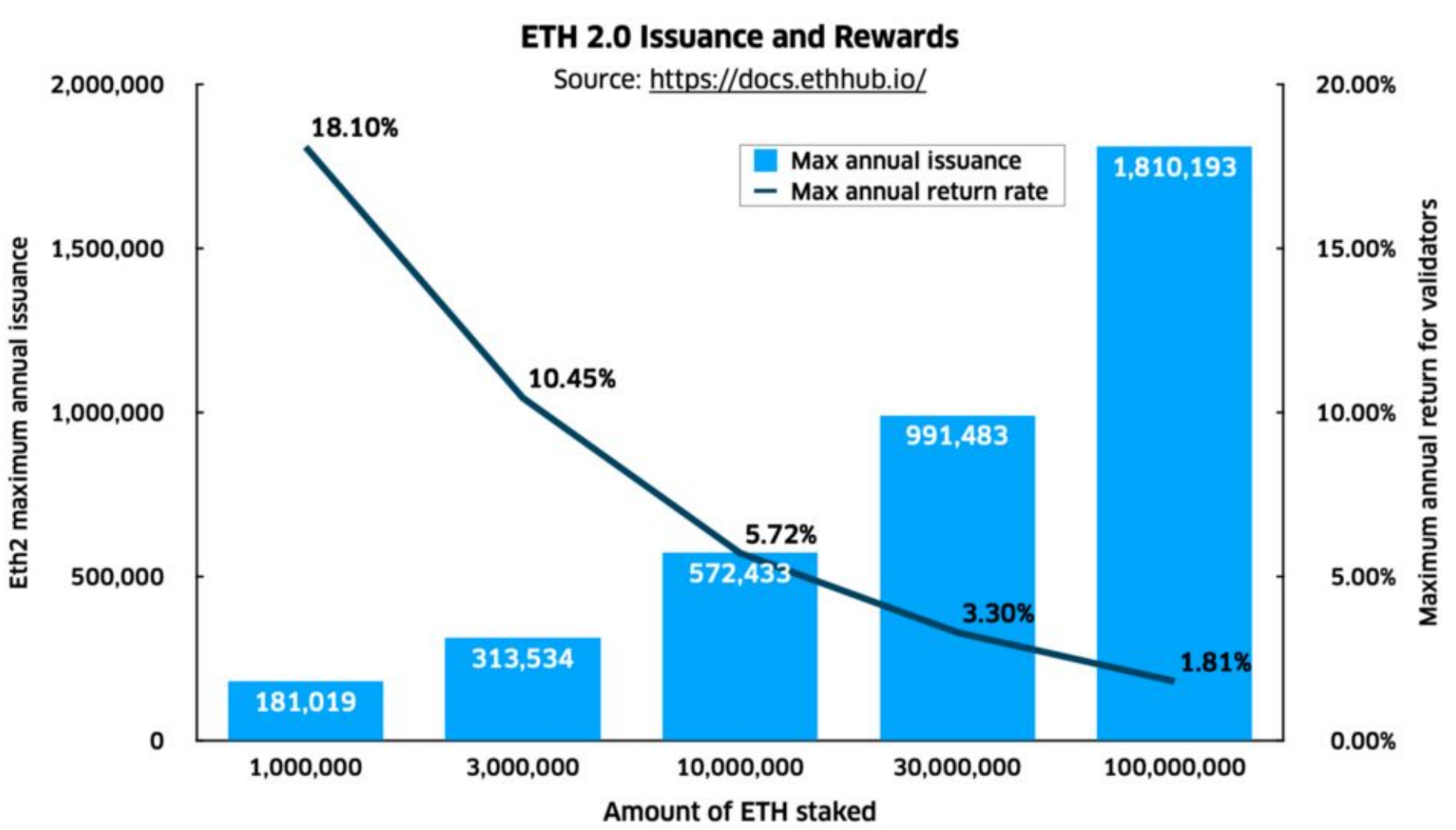


Running Validators

This is NOT investment advice!

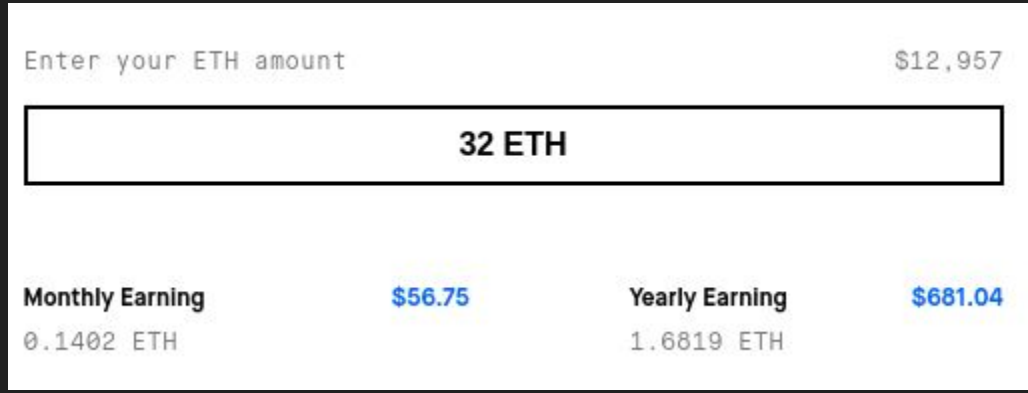
Prysm Client Design




ETH 2.0 Issuance and Rewards
Source: https://docs.ethhub.io/
Max annual issuance
Max annual return rate
Eth2 maximum annual issuance
Maximum annual return for validators
Amount of ETH staked
2,000,000
1,500,000
1,000,000
500,000
0
20.00%
15.00%
10.00%
5.00%
0.00%
1,000,000
3,000,000
10,000,000
30,000,000
100,000,000
181,019
313,534
572,433
991,483
1,810,193
18.10%
10.45%
5.72%
3.30%
1.81%

Validator profits/yield

- Costs are low and do not scale linearly with the number of validators in operator
- Long term commitment: cannot unlock funds until phase 2
- Liveness penalties can cost up to half of validator balance (16 ETH)
- Rewards are higher for early adopters
- Staking is not without **risk!**



Based on 10MM at stake and ETH price at $405

Become a validator and help secure eth2.

Earn continuous rewards for providing a public good to the community.

GET STARTED

- https://medalla.launchpad.ethereum.org/
- https://prylabs.net

Recap

- Ethereum 2.0 introduces proof of stake and blockchain sharding
- Ethereum 2.0 is a new blockchain; not a hard fork
- Ethereum 2.0 is a phased rollout, expected to complete in 2021
- Ethereum 2.0 phase 0 is available to test today, launching this Q4 2020

Prysmatic Labs

@prylabs