


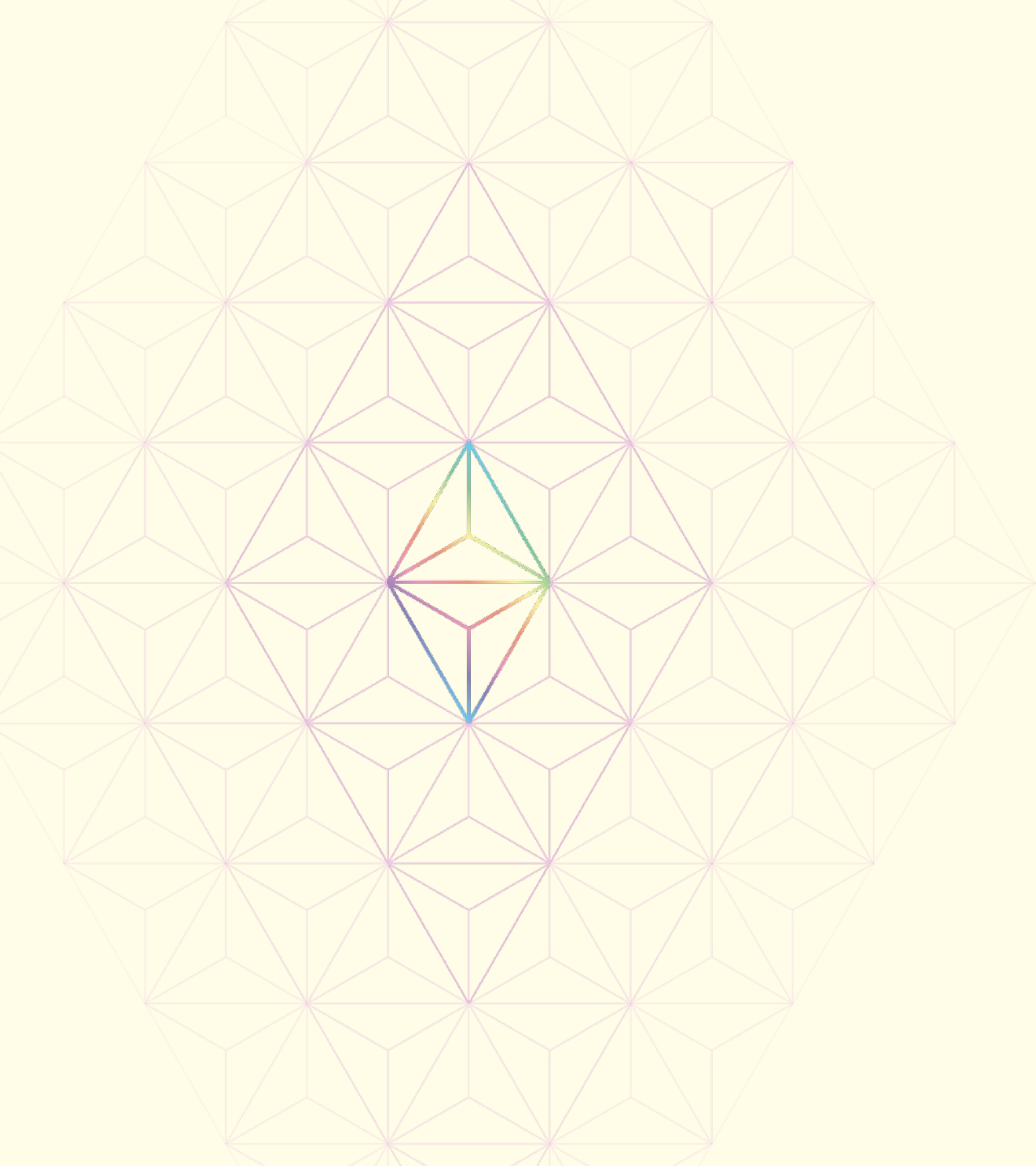


Consensus clients security

Mamy Ratsimbazafy
Ethereum / Nimbus developer @ Status.im

 m_ratSIM

 mratsim



Consensus Layer milestones

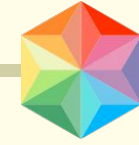
2018, refining consensus layer design

Nov 2017-Jan
2018



Proof-of-Stake
Hybrid Casper
testnet

Jan 2018



Sharding phase
1.
Prismatic Labs,
“sharded Geth”

Apr 2018



Initial sharding
plan deprecated
and reworked

Jun-Sep 2018

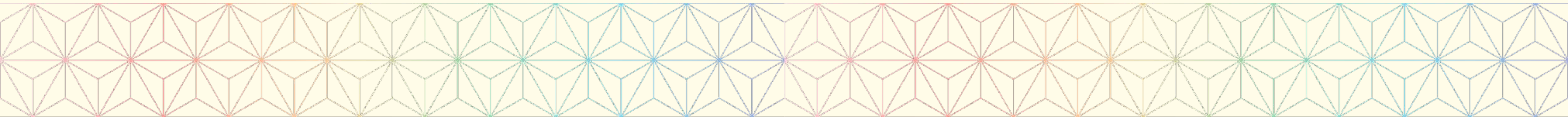


Vitalik’s HackMD
“Casper + Sharding
spec”

Sep 2018-Feb
2019



Commit-driven
research at
ethereum/eth2-specs

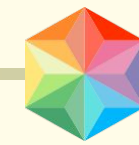


2019, Interop

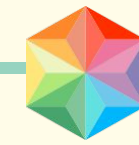
Jan 2019



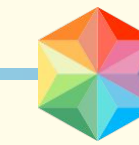
Mar 2019



Apr 2019



May 2019



Sep 2019



First spec
pre-release v0.1
Phase 0 feature
complete (except
networking)
Phase 1 now
research focus

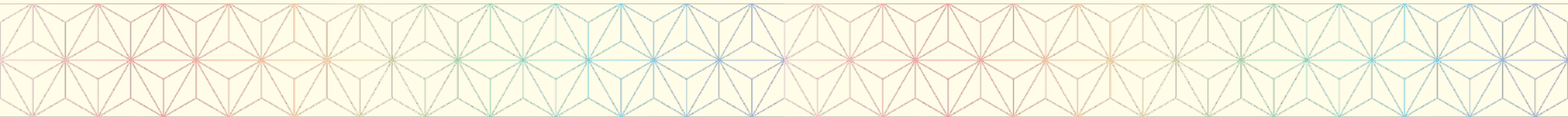
Executable spec and
test vectors.

Single-client testnets

Continuous
Integration of spec

Beacon Fuzz

The Interop Lock-In
event



2020, multicient testnets and ...

Mar 2020



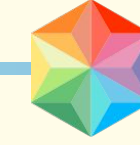
Apr 2020



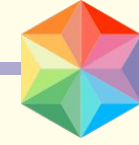
May 2020



July 2020



Q2 to Q4



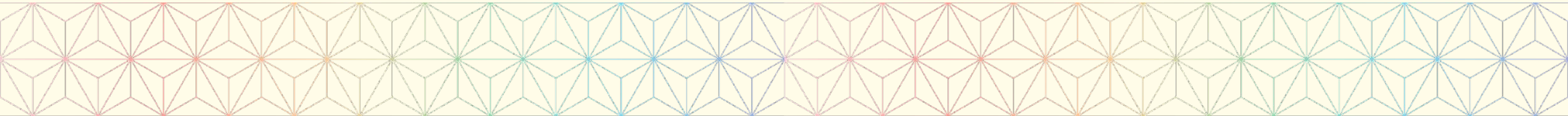
Phase 0 spec audit
completed

First multi-client
testnet: Schlesi

Formal verification
found a critical bug:
an unbounded
number of blocks
could be created for
the same slot.

Public attacknets

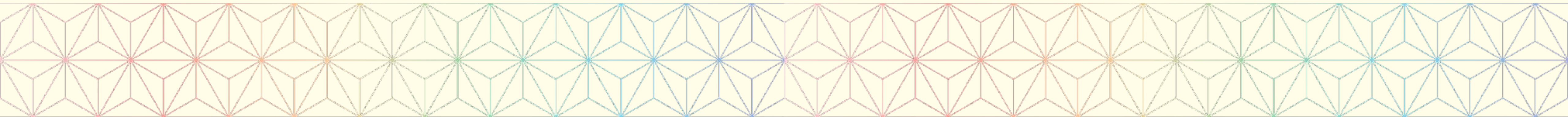
All clients undergo
security audits

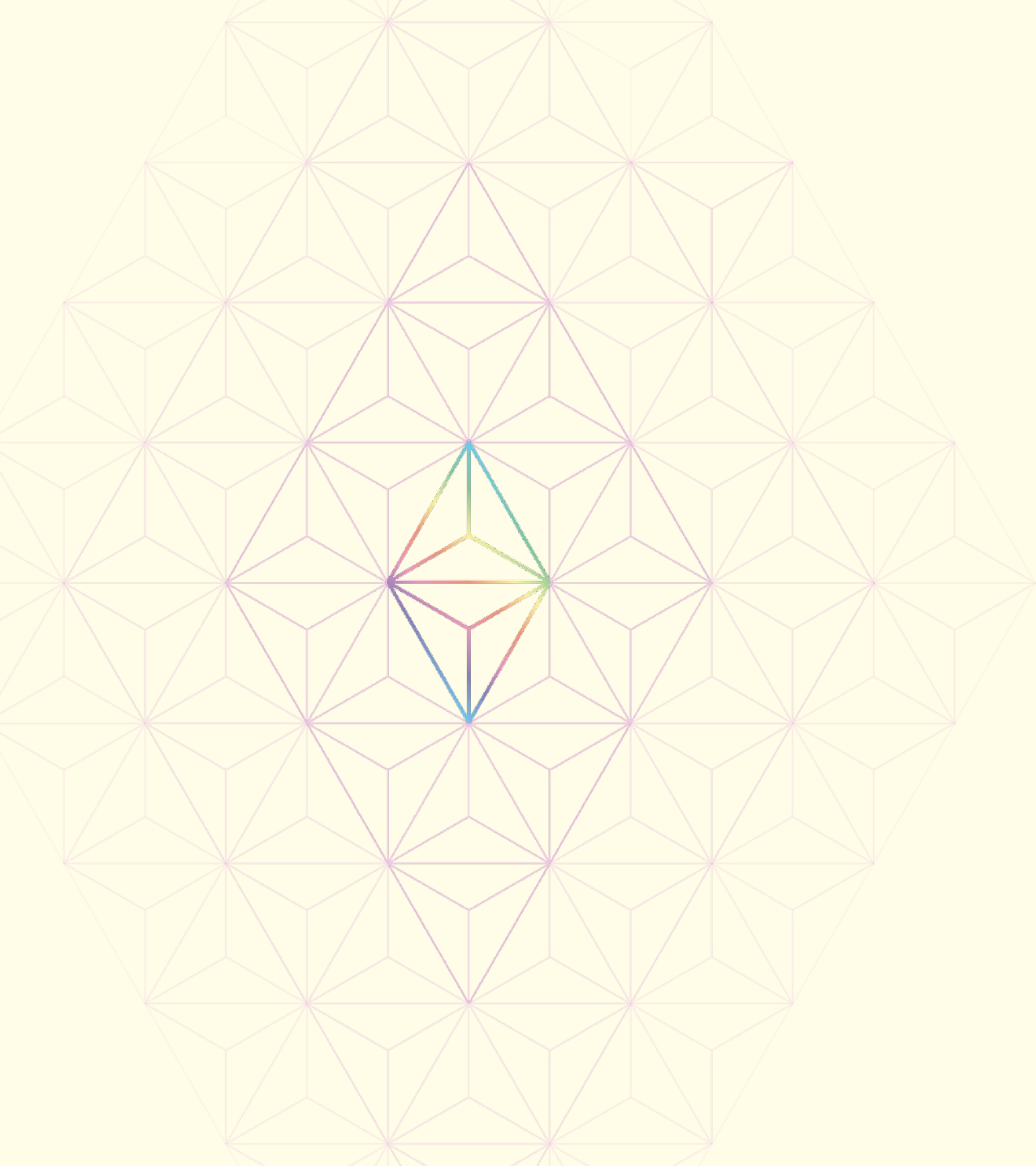


2020, multicient testnets
and ...



Mainnet launch, on Dec 1st, 2020

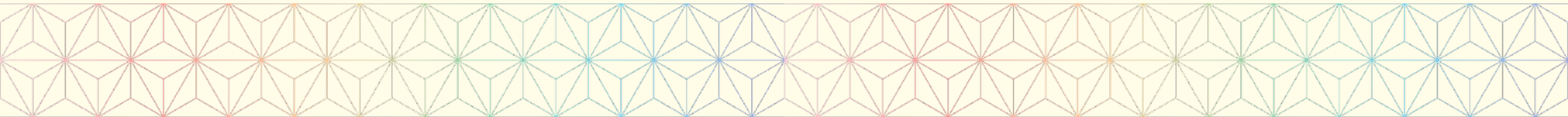


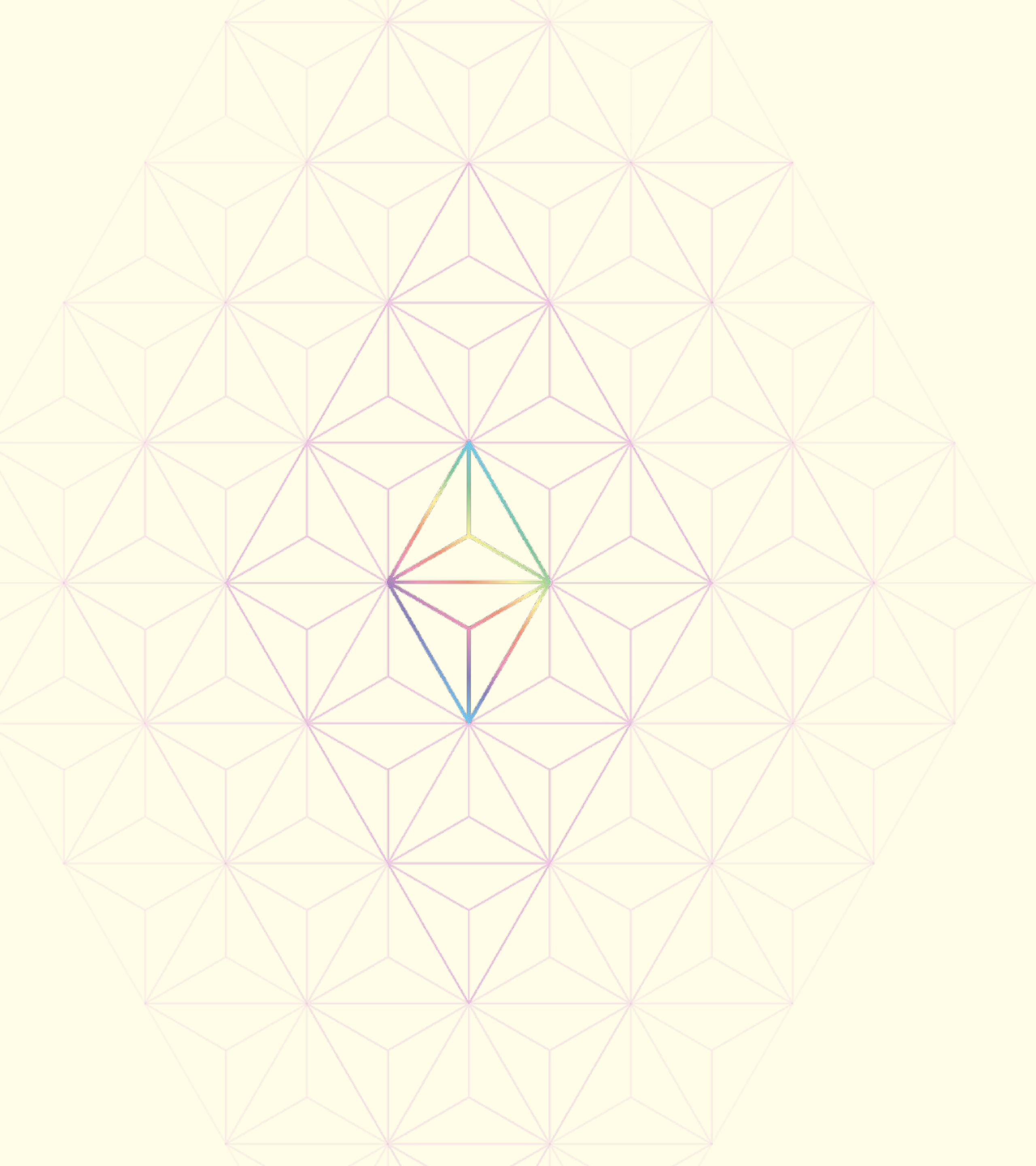


What's at Stake

What's at Stake

- Reputation
- 610k+ ETH in the deposit contract at Genesis (~20k validators, 0.8% of that time supply, ~\$400M)
- 1M+ ETH after a week
- 11.2M+ ETH today (~350k validators, ~\$34.5B)
- \$23M Ethereum transaction fees / day (source: [cryptoforesight.com](https://cryptoforesight.com/ethereum-transaction-fees/))
- 7.5M+ ETH locked in Defi (source: [Defipulse](https://defipulse.com/))





**What could go
wrong?**

What could go wrong?

Cryptography

Signature forgeries

Networking

Denial of Service (discovery, gossipsub)

Eclipse attacks

Wire format

Consensus

State transition & block processing

Fork choice & Finalization

Rewards, penalties slashings

Sync

Engineering

Implementation bugs

Dependencies bugs (including time)

User Interface issues (keystores, RPC, DB migration)

What could go wrong?

A Shuffling example

Code	4
Commits	35
Issues	73
Wikis	0
States	
Closed	65
Open	8

[Advanced search](#)[Cheat sheet](#)

73 issues in ethereum/eth2.0-specs

Use custom types in data structures

Opened by Nashatyrev 9 days ago • 4 comments

fix committee assignment bugs

2 bugs: - We were modifying current_ **shuffling** _epoch and then calling get_current_epoch_committee_count expecting that it was still getting the committee value of ...

Opened by djrtwo 2 days ago • 5 comments

[WIP] Break up crosslink_committees_at_slot

should we add an assert to check **shuffling** _epoch out of bound condition? assert get_previous_epoch(state) <= **shuffling** _epoch <= get_current_epoch(state)+1

Opened by djrtwo 7 hours ago • 8 comments

committee shufflings take at least 2 epochs to change

... validating and not all shards are represented in each **shuffling**). - This slows dc activations and exits by a factor of 2.

Opened by djrtwo on Jan 14 • 3 comments

Light client proposal

... , maintained similarly to the randao roots - To compute the seed used in get_shi sha3(get_randao_mix(slot), get_active_index_root(slot)) where get_active_index_r

Opened by vbuterin on Jan 17 • 3 comments

get_crosslink_committees reads previous_seed during genesis_epoch

Problem Summary During GENESIS_EPOCH, get_crosslink_committees_at_slot(state.previous_ **shuffling** _epoch when epoch == current_epoch. Detail get_crosslink_committees_at_slot(...) first ...

Opened by paulhauner 13 days ago

Fix out-of-bounds in `get_shuffling`

What Change get_ **shuffling** (...) so it gives shuffles using an index of active_valida value of active_validator_indices. Why The following fails with a IndexError: list inde

Remove Record suffix

no, get_ **shuffling** just returns the **shuffling** split across slots. get_shard_committees_at_slot returns an array of arrays of (validators, shard) tuples. Arguably, the naming and return value of these ...

Opened by sifnoc on Jan 13 • 24 comments

Keep latest 2^n RANDAO mixes in the state

Also ensures that we have the seed of the current **shuffling** in state. Right now, we discard the current **shuffling** seed immediately after performing the **shuffling** . Not very friendly for reconstructing the **shuffling** outside the specific state transition.

Opened by JustinDrake on Dec 12, 2018 • 4 comments

"proposer_slots" -> "proposer_nonce"

Opened by paulhauner on Jan 24 • 1 comment

RFC: drop shard_and_committee_for_slots from state

Opened by ametheduck on Nov 29, 2018 • 4 comments

Understanding the `get_new_shuffling()` function

For the past several weeks, as part of my hackternship, I have been trying to understand the get_new_ **shuffling** () function. In its current iteration, the get_new_ **shuffling** () function is in charge of ...

Opened by Mikerah on Nov 14, 2018 • 3 comments

initial assignment of `state.persistent_committees`

Issue We are currently assigning state.persistent_committees as a **shuffling** of ValidatorRecords rather than just validator_indices persistent_committees=split(shuffle(initial_validator_registry ...

Opened by djrtwo on Dec 5, 2018 • 5 comments

Remove MIN_VALIDATOR_REGISTRY_CHANGE_INTERVAL

... in times of non-finality (attacks, short range forks, etc), I worry we end up making the **shuffling** extremely subjective and ultimately an attack vector. (for example: easier to construct reasonable looking blocks when the proposer for that slot is not entirely certain)

Opened by vbuterin on Dec 19, 2018 • 5 comments

assertion in `get_active_index_root` too strong

#515

Issue In validator registry and **shuffling** seed data we set state.current_calculation_epoch = next_epoch and then do state.current_epoch_seed = generate_seed(state, state.current_calculation_epoch ...

Opened by djrtwo on Jan 29

Delay exits with penalty

#350

Delaying exits with penalty by 1+epsilon epochs ensures that self-slashing single validators does not change the **shuffling** for the next epoch and so cannot (normally) be used as a way of manipulating the **shuffling** .

Opened by vbuterin on Dec 21, 2018 • 8 comments

Introduce swap-or-not shuffle

#576

See #563 for discussion. Here is a more efficient implementation for **shuffling** an entire set; it can live here until we come up with an explicit "efficient implementation" doc: def shuffle ...

Opened by vbuterin 23 days ago • 20 comments

Mitigating attacks on light clients

#403

... **shuffling** . Note that alternative **shuffling** algos do not fix this problem, because the step of filtering out inactive validators still requires a pass through the entire validator set. Second, it is ...

Opened by vbuterin on Jan 7 • 8 comments

helpers and notes for **shuffling** lookahead

#520

beacon chain spec changes: - update get_crosslink_committees_at_slot to be able to get potential committees for slots from the next epoch. add registry_change param to get next epoch committees ...

Opened by djrtwo on Jan 30 • 7 comments

Possible alternative numer-theoretic **shuffling** algorithm

#323

Motivation Construct a **shuffling** algorithm where you can compute the value in the shuffled list at any specific position relatively cheaply without computing all of the other values at the same time ...

Opened by vbuterin on Dec 14, 2018 • 13 comments

non-determinism in **shuffling** from `SEED_LOOKAHEAD`

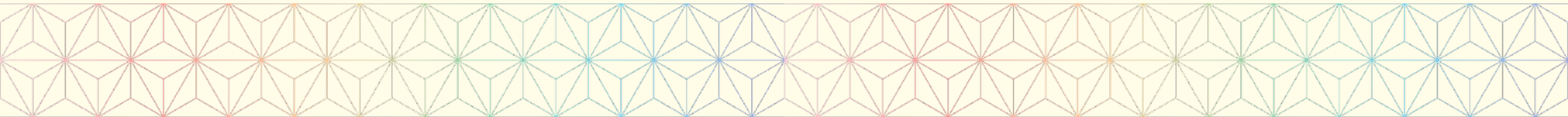
#405

Issue shufflings are calculated using a seed from SEED_LOOKAHEAD slots ago get_ **shuffling** (state.latest_randao_mixes[(state.slot - SEED_LOOKAHEAD) % LATEST_RANDAO_MIXES_LENGTH ...

Opened by djrtwo on Jan 7 • 3 comments

An array of mitigation solutions

- Multi-client development (up to 10 clients)
 - Only NASA is more stringent: teams must be completely firewalled
- Specs: audited, executable, formally verified *twice*
- Each client has been tested, fuzzed, audited
- Many many devnets, testnets, attacknets, mergenets, ...



Outcomes

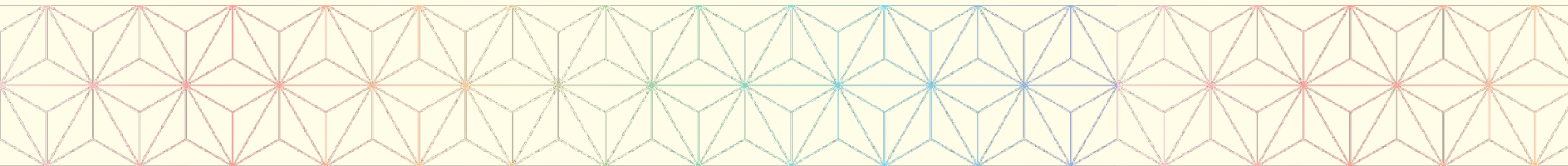
Each client is one of the highest quality software in their programming language.

Ethereum CL has been pushing standards:

- Cryptography (BLS signature and hashing to elliptic curve)
- Networking (LibP2P, Discv5)

Both research and engineering prowess.

All issues so far (fingers crossed) are usage-related!






Let's BUIDL!

Consensus Clients Security

Mamy Ratsimbazafy

Ethereum / Nimbus developer @ Status.im

 m_ratSIM

 mratsim

