

# Tightly Coupled DeFi in ETH2



pg

## Agenda

1. What do we mean by tightly coupled Ethereum?
2. What are our options for tightly coupling ETH2?
3. How will DeFi look on scalable Ethereum?

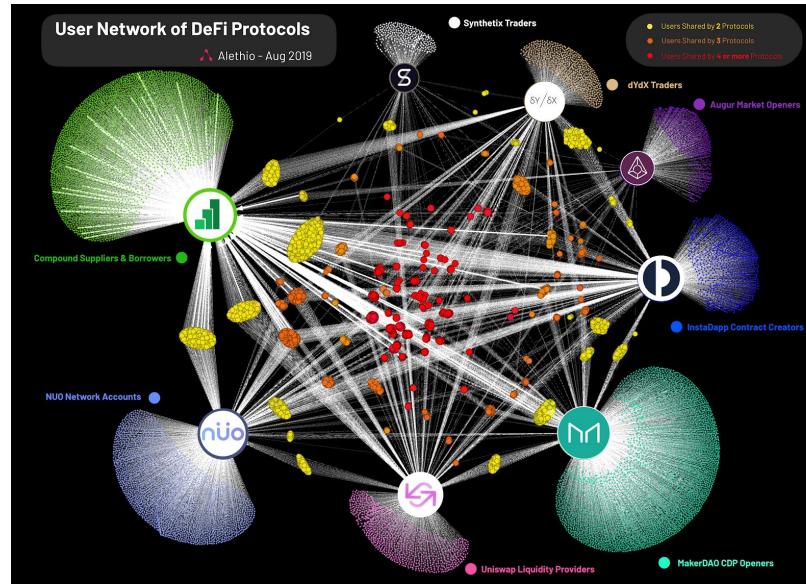
## Agenda

1. What do we mean by tightly coupled Ethereum?
2. What are our options for tightly coupling ETH2?
3. How will DeFi look on scalable Ethereum?

What does it mean that DeFi is tightly coupled?

# What does it mean that DeFi is tightly coupled?

- Single state space.
- Atomic transactions.



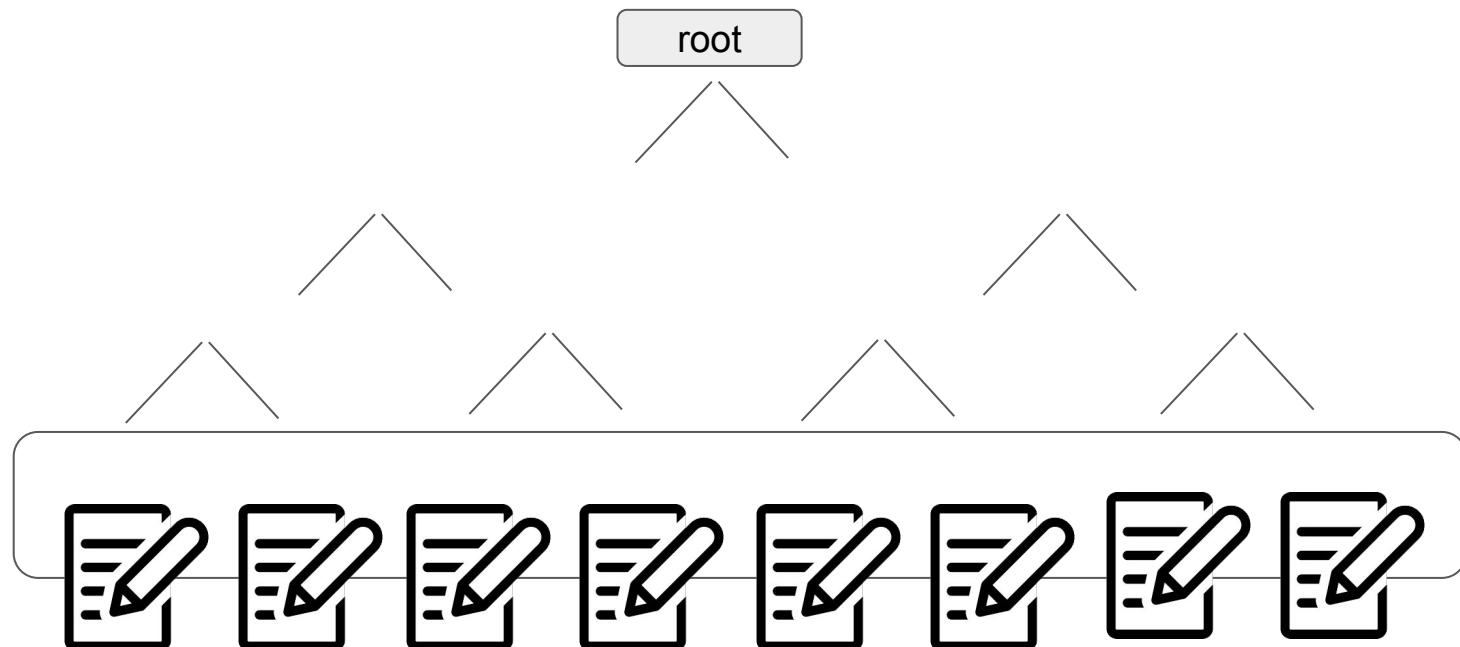
What does it mean that DeFi is tightly coupled?

- Single state space.
- Atomic transactions.

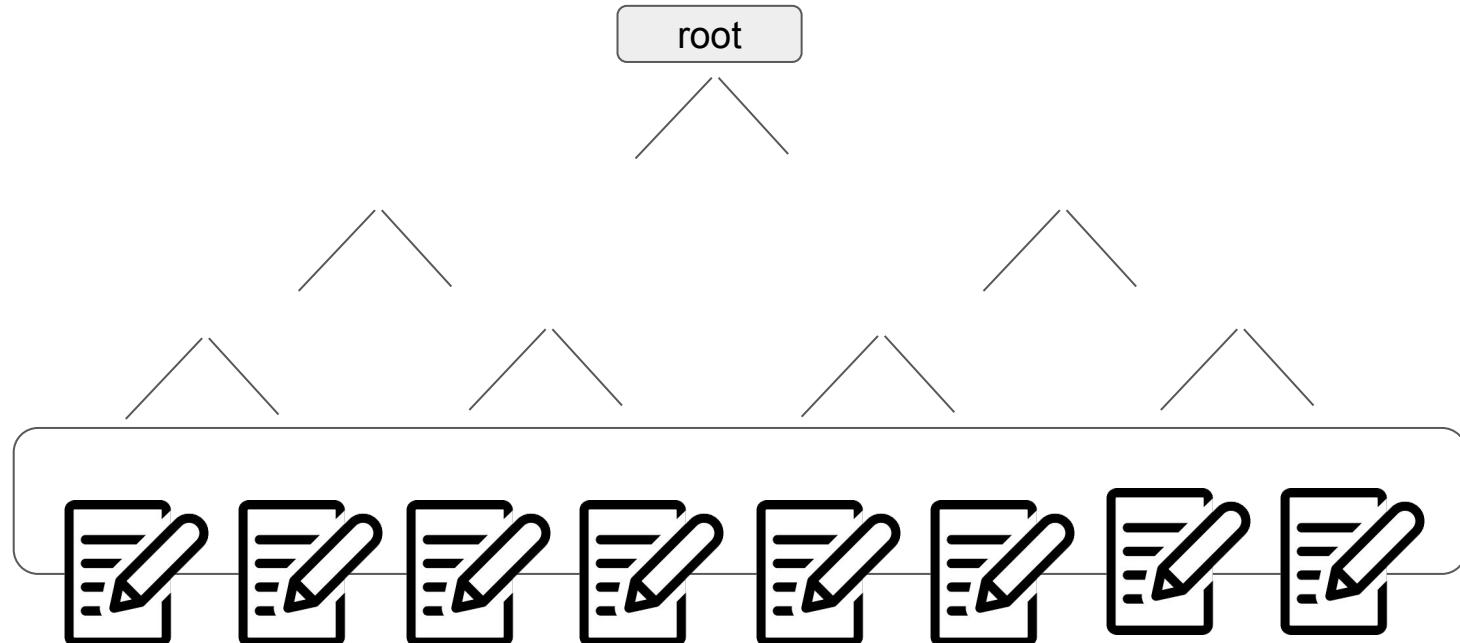


# Single State Space

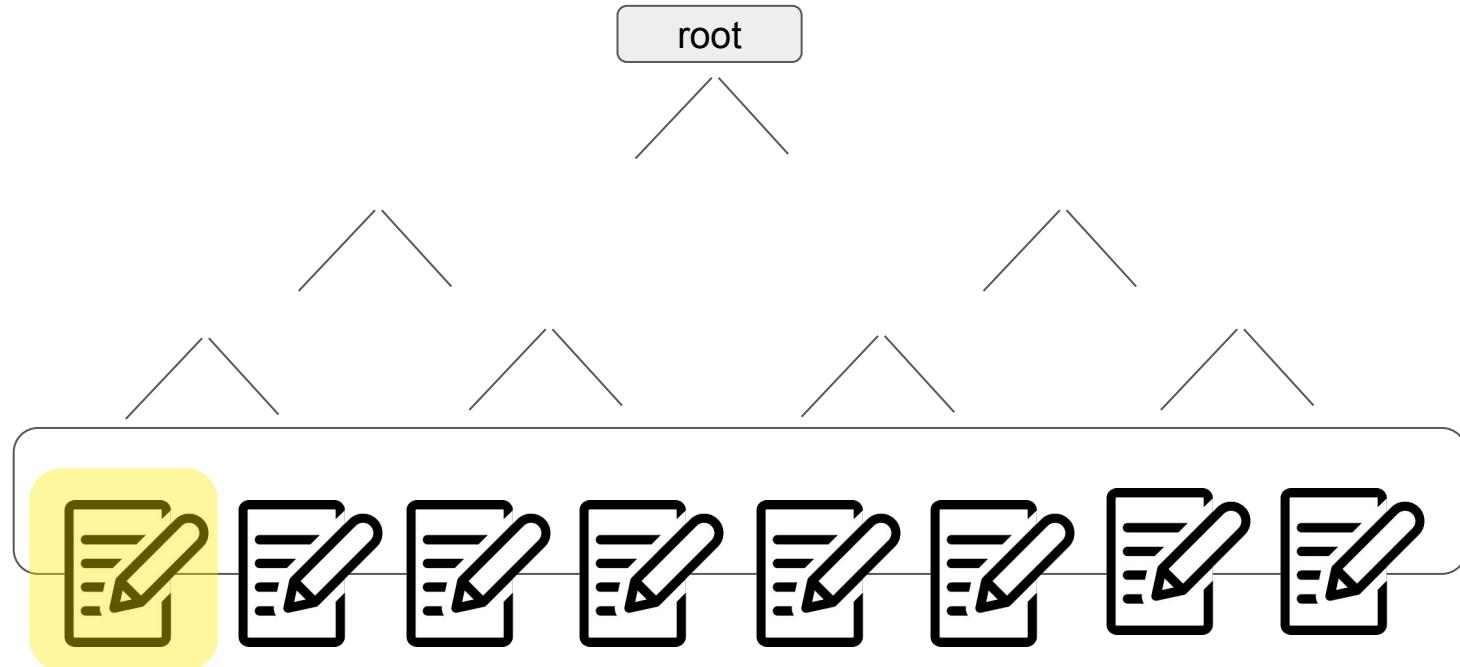
# Single State Space



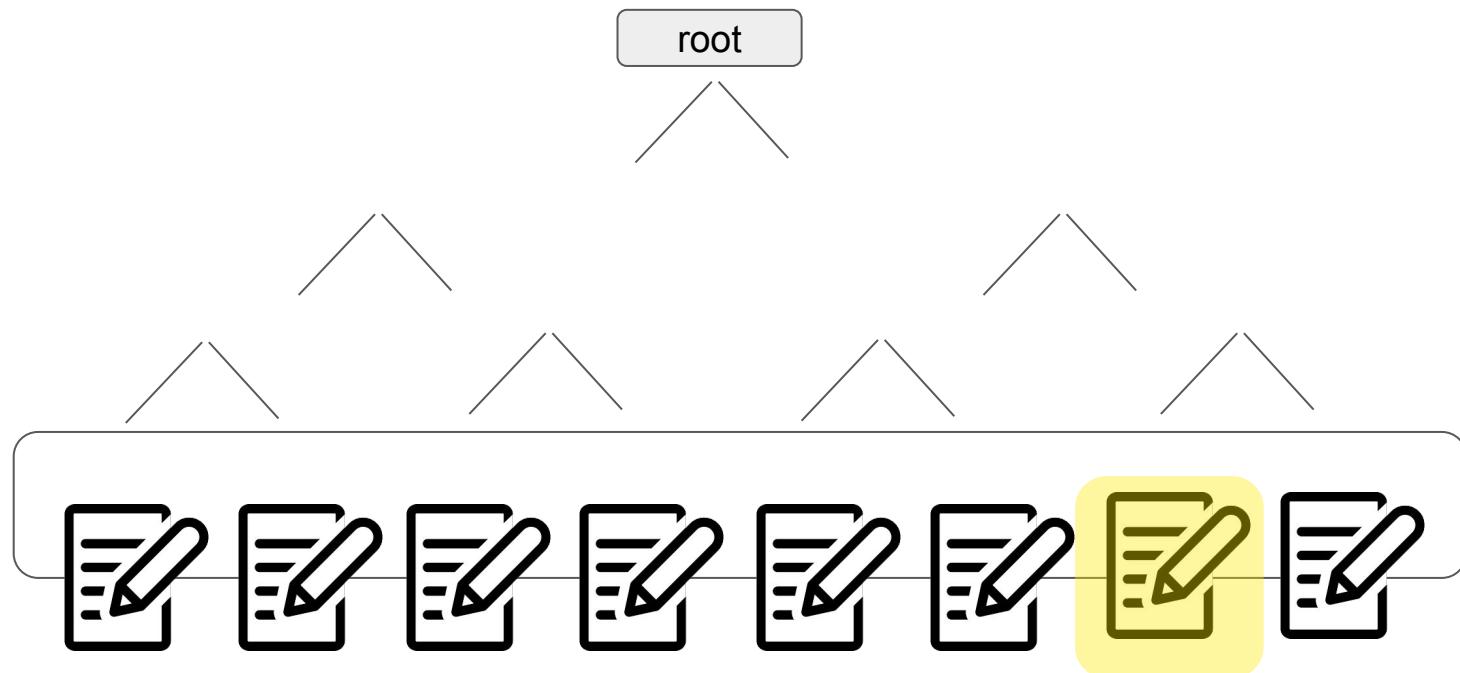
# Single State Space



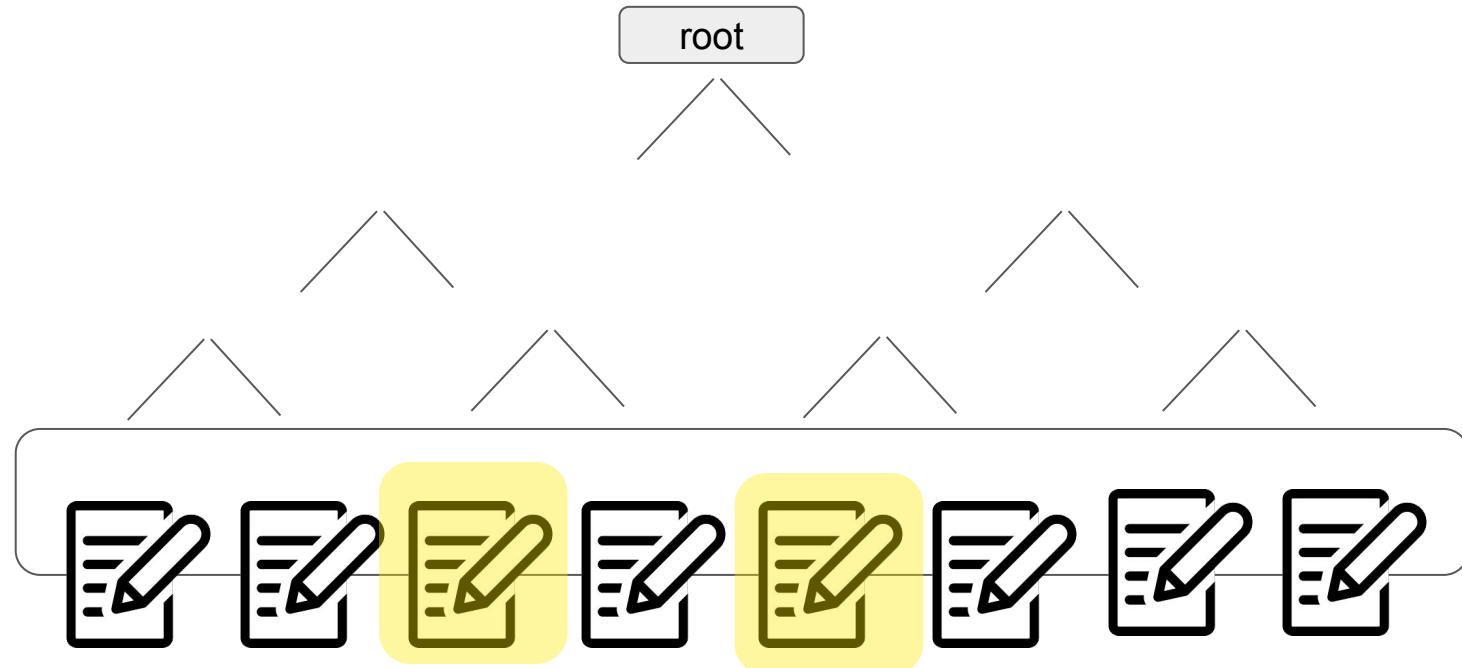
# Single State Space



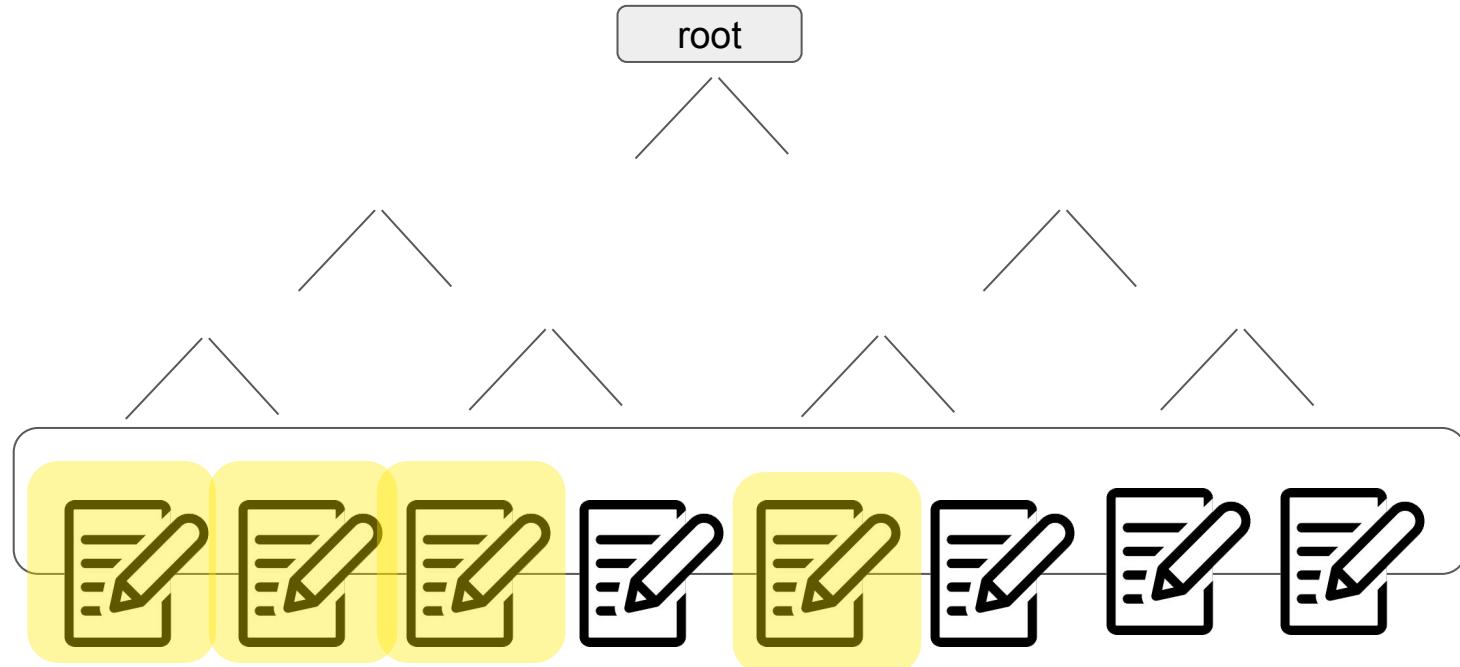
# Single State Space



# Single State Space



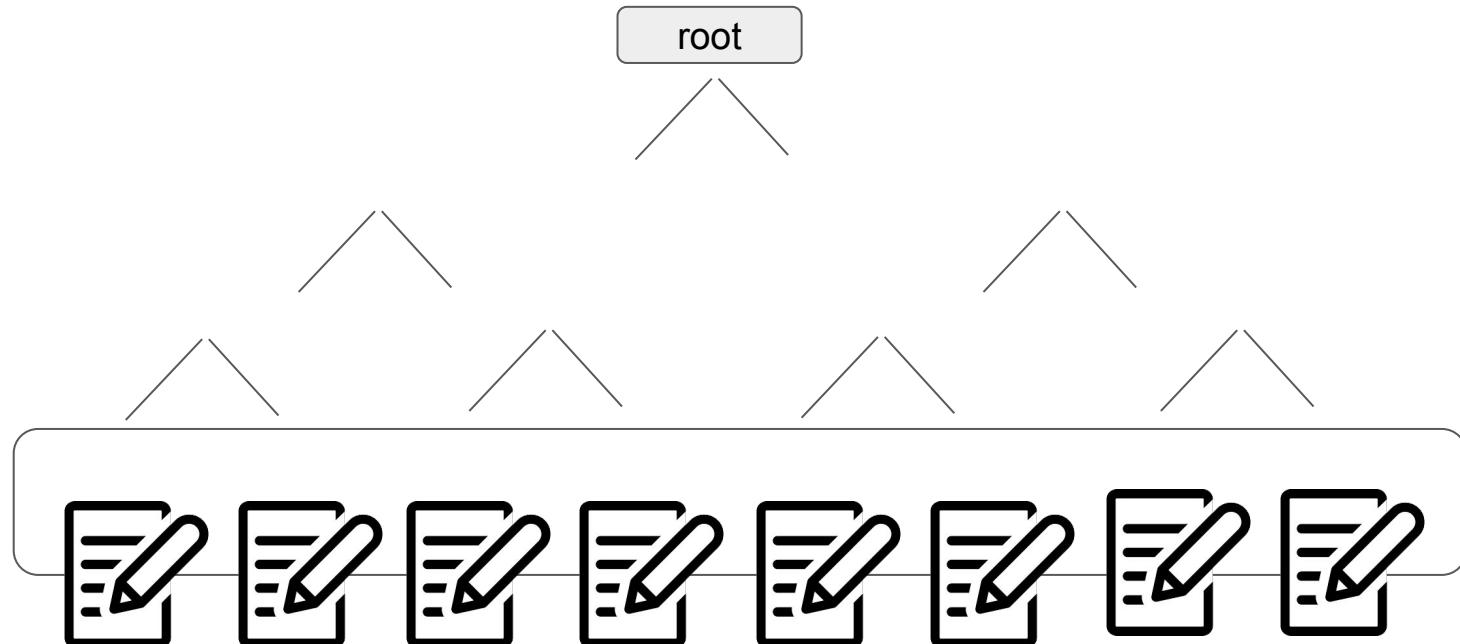
# Single State Space



## Single State Space

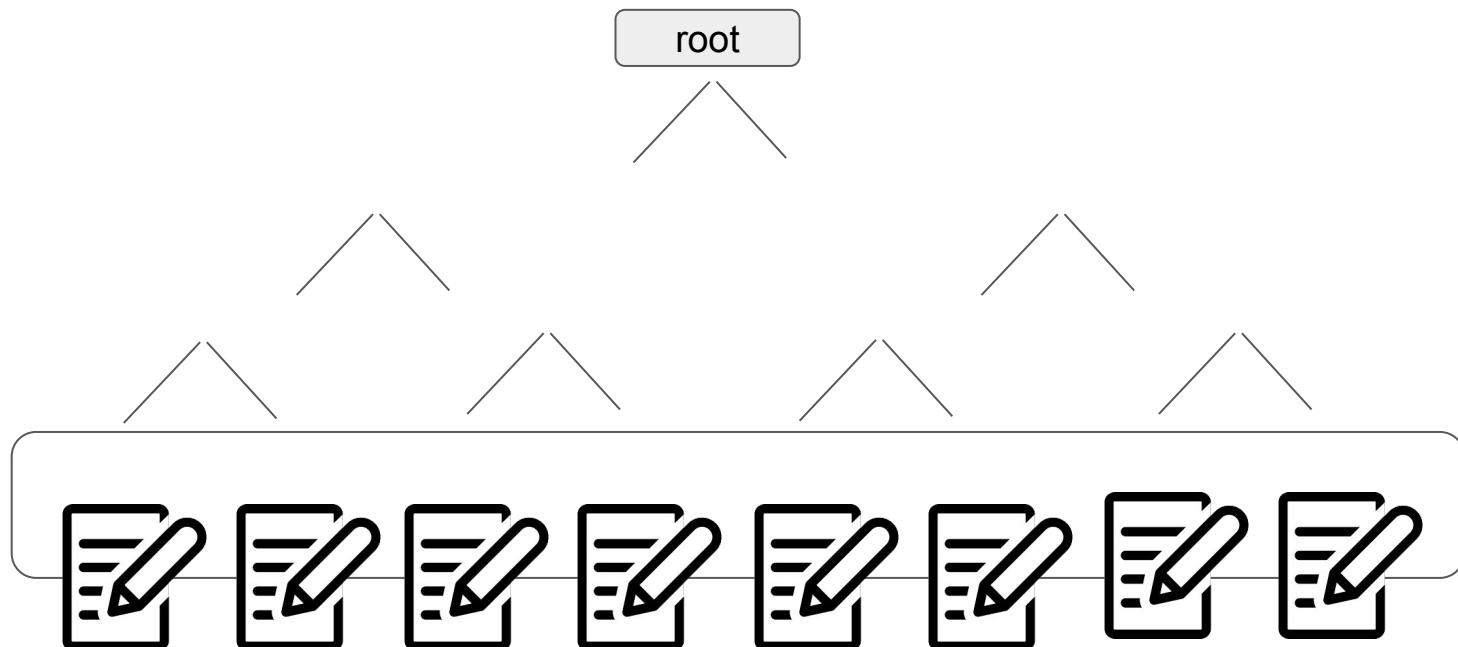


All contracts  
are accessible  
in every block.



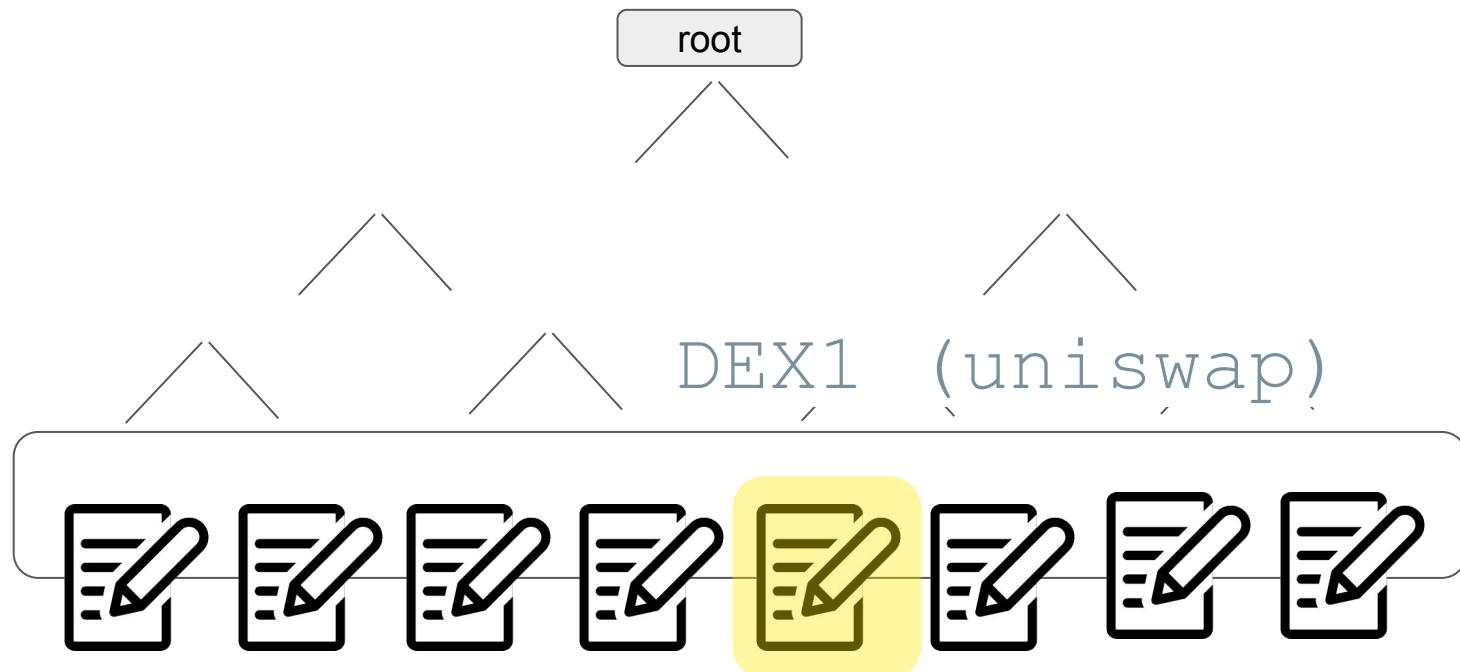
# Atomic Transactions

# Atomic Transactions



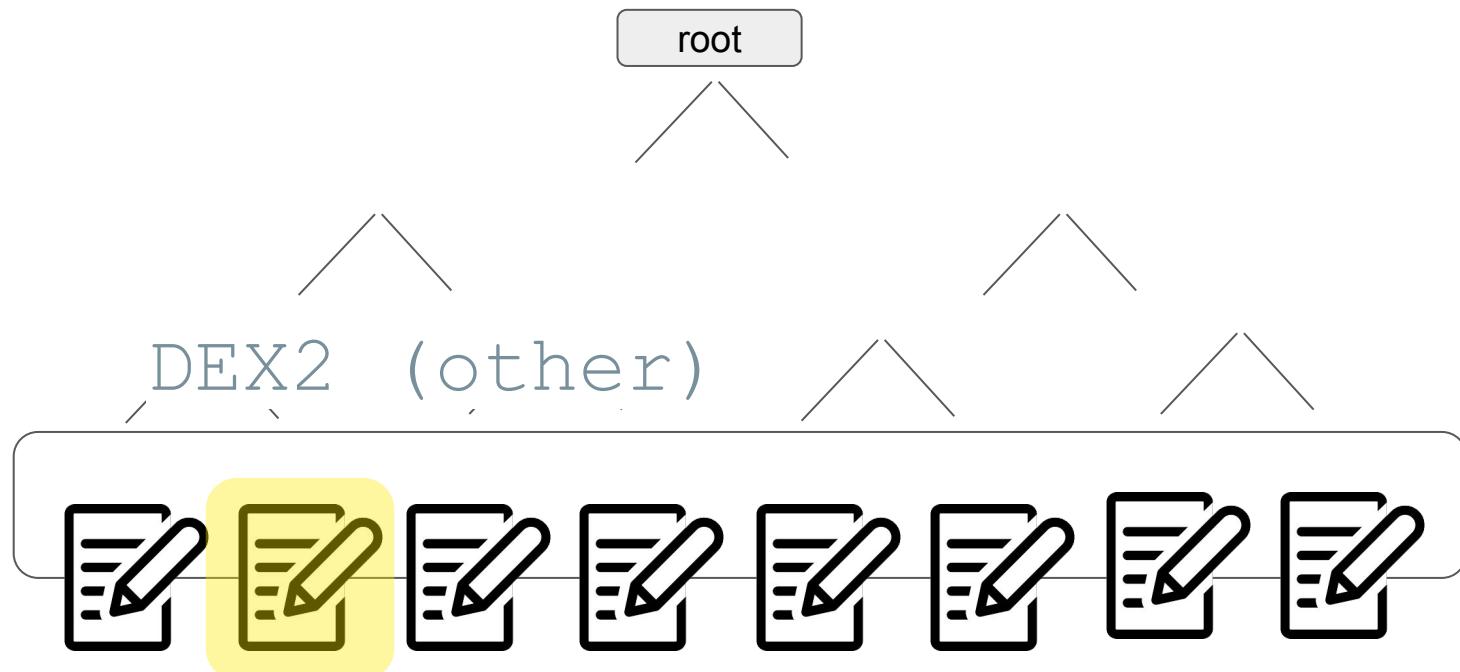
Atomic Transactions

Selling token  
for \$10

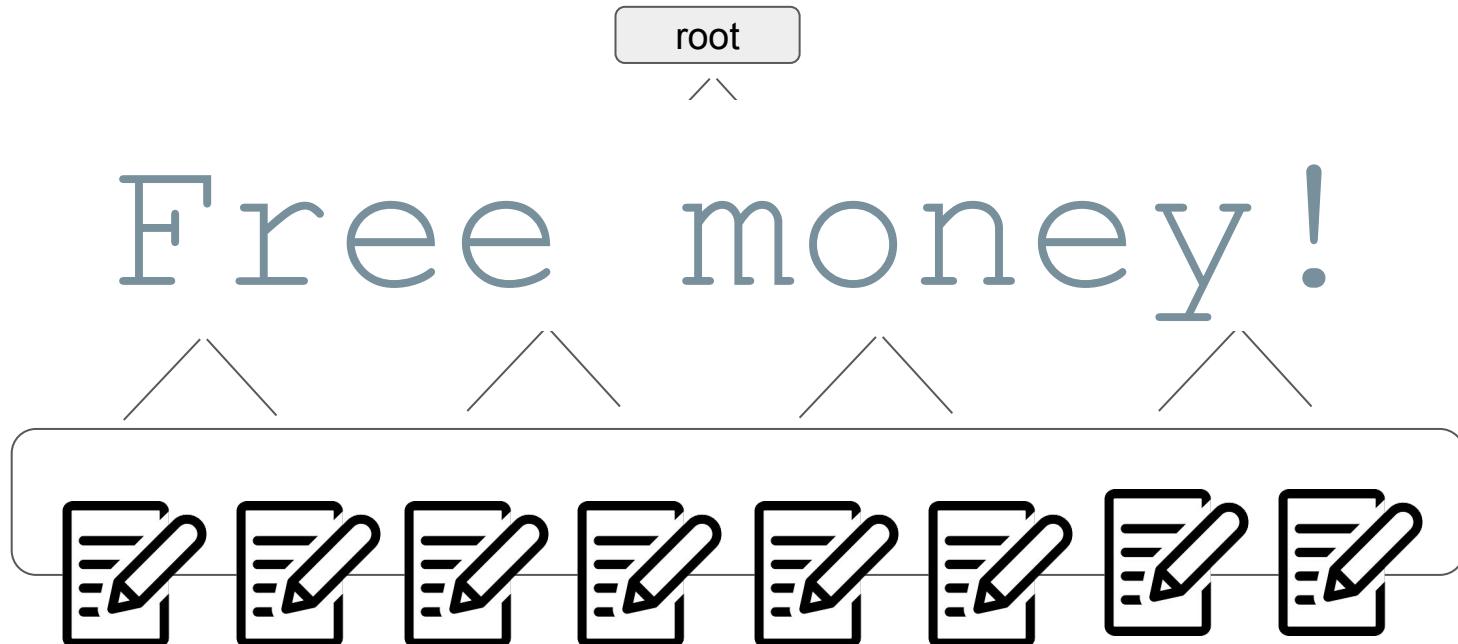


Atomic Transactions

Buying token  
for \$11



## Atomic Transactions



# Atomic Transactions

Atomic transaction:

[buy token on Uniswap,  
sell token on OtherDex]



## Atomic Transactions

All or nothing!

Atomic transaction:

[buy token on Uniswap,  
sell token on OtherDex]

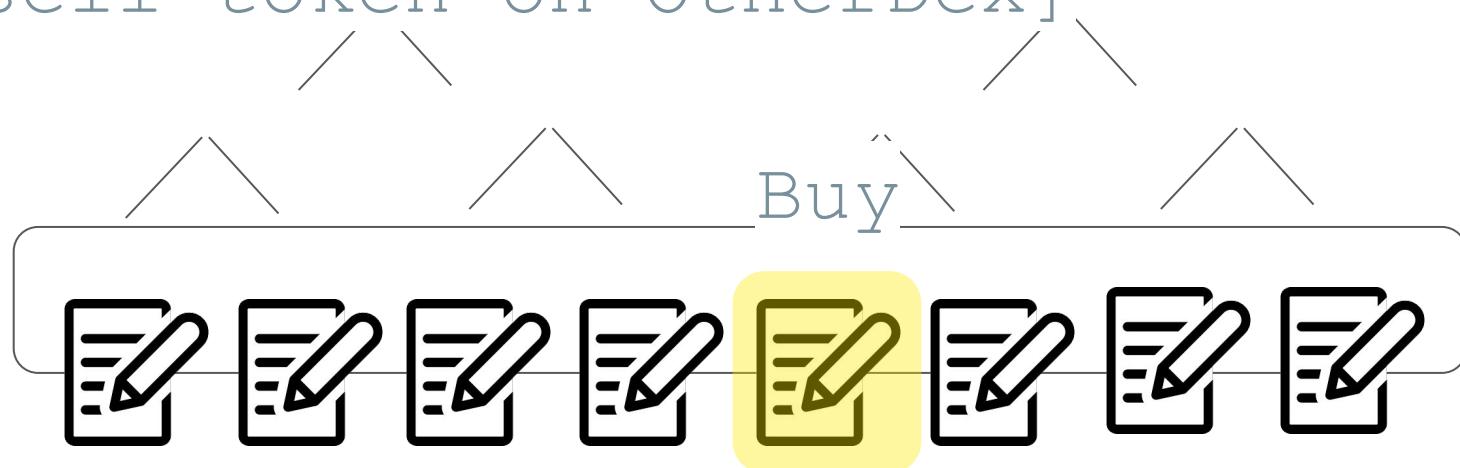


## Atomic Transactions

All or nothing!

Atomic transaction:

[buy token on Uniswap,  
sell token on OtherDex]

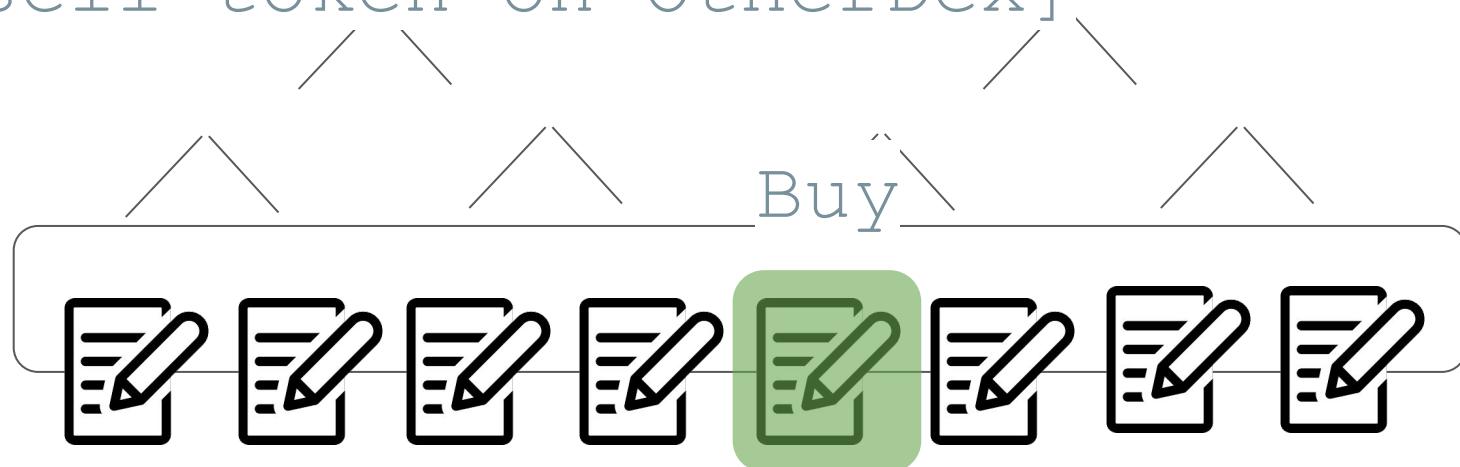


## Atomic Transactions

All or nothing!

Atomic transaction:

[buy token on Uniswap,  
sell token on OtherDex]



## Atomic Transactions

All or nothing!

Atomic transaction:

[buy token on Uniswap,  
sell token on OtherDex]



## Atomic Transactions

All or nothing!

Atomic transaction:

[buy token on Uniswap,  
sell token on OtherDex]



## Atomic Transactions

All or nothing!

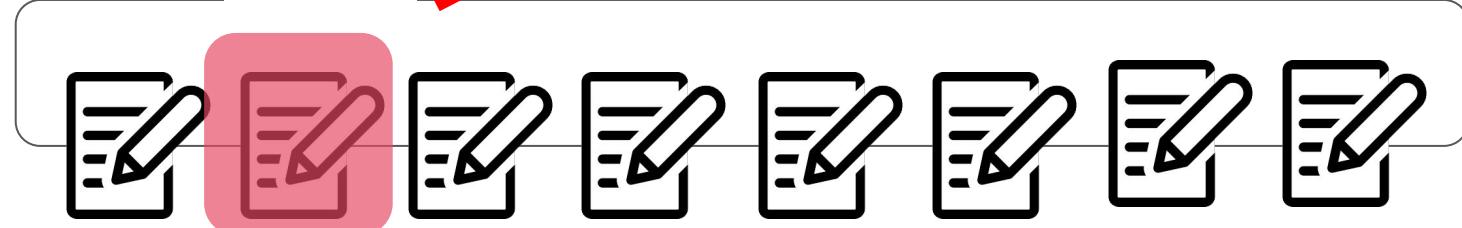
Atomic transaction

[buy token on H

sell token

REVERT

Sell



# Atomic Transactions

Google free money

All Images News Maps Videos More Settings Tools Collections SafeSearch ▾

clipart government printable scholarship roblox background cash casino real vec

HERE'S SOME FREE MONEY

Yo! FREE MONEY!!

FREE MONEY!

FREE MONEY

50 Ways to Get Free Money : There's a ...  
sidehustlenation.com

11 Easy Ways To Get Free Money Today ...  
wealthyturtle.com

4 Easy Ways To Make FREE Money! - YouT...  
youtube.com

Get Free Money: 10 Ways to Earn ...  
everybuckcounts.com

Get FREE Money NOW! Automatic Income ...  
ezraslayton.com

Easiest Ways to Get Free Money ...  
listenmoneymatters.com

40 Easy Ways To Get Free Money Fast ...  
biblemoneymatters.com

50 Ways to Get Free Mon...  
sidehustlenation.com

Cash Bonuses and Free Money Pro...  
maximizingmoney.com

How to Get Free Money Online in 2019 ...  
swiftsalary.com

## Agenda

1. What do we mean by tightly coupled Ethereum?
2. What are our options for tightly coupling ETH2?
3. How will DeFi look on scalable Ethereum?

## Agenda

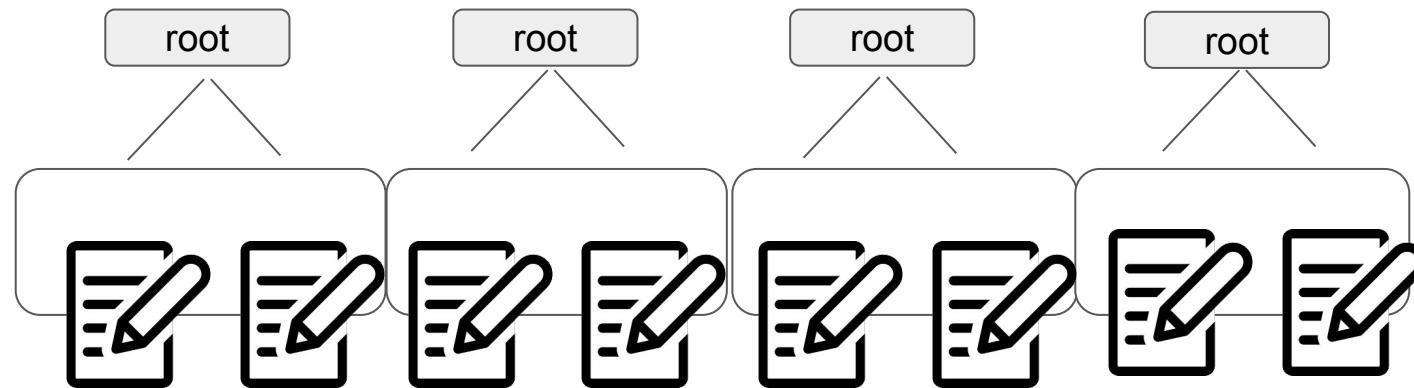
1. What do we mean by tightly coupled Ethereum?
2. What are our options for tightly coupling ETH2?
3. How will DeFi look on scalable Ethereum?

DeFi in ETH2: Why are we worried?

## DeFi in ETH2: Why are we worried?

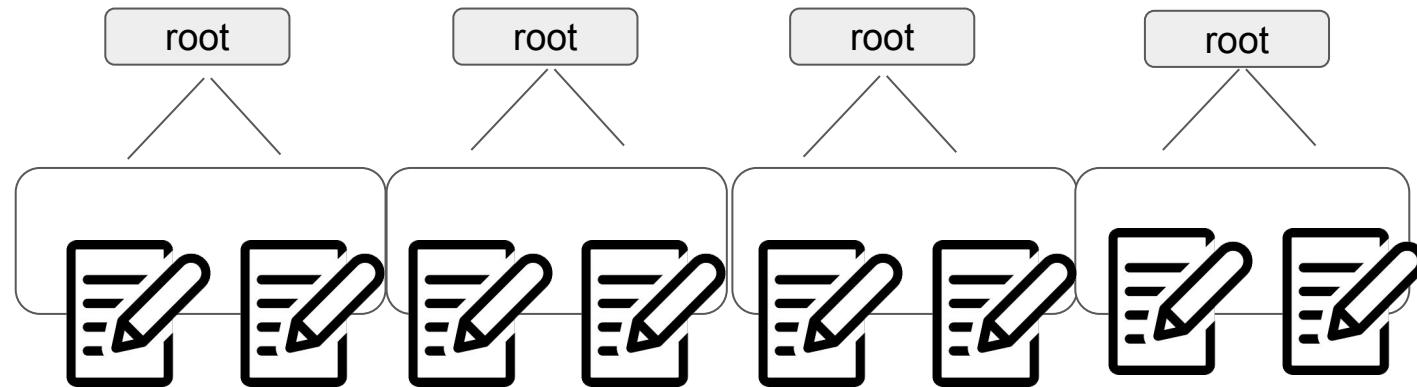
- **Multiple state spaces.**
  - Shard\_a, shard\_b, shard\_c, etc.
  - What do I deploy to?

# Multiple State Spaces



# Multiple State Spaces

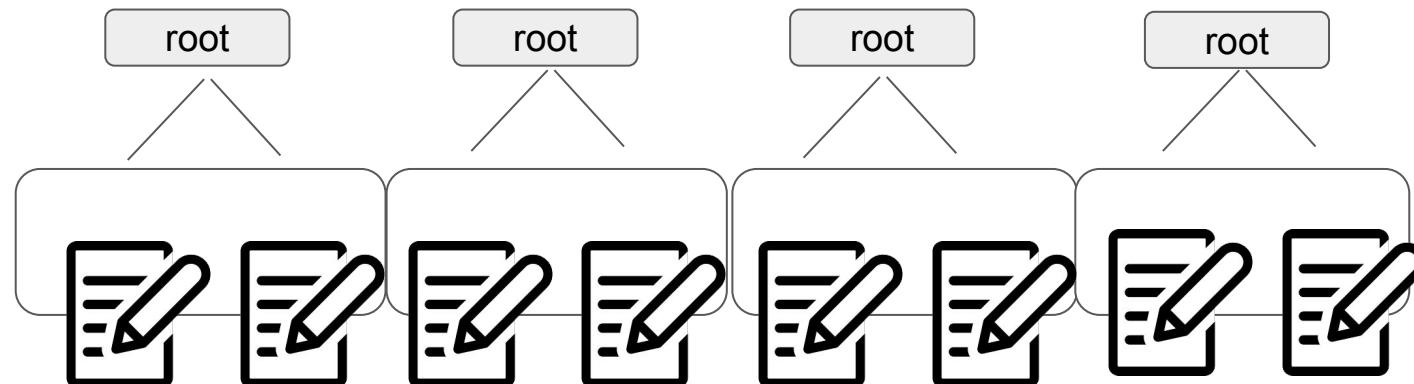
Shard\_A    Shard\_B    Shard\_C    Shard\_D



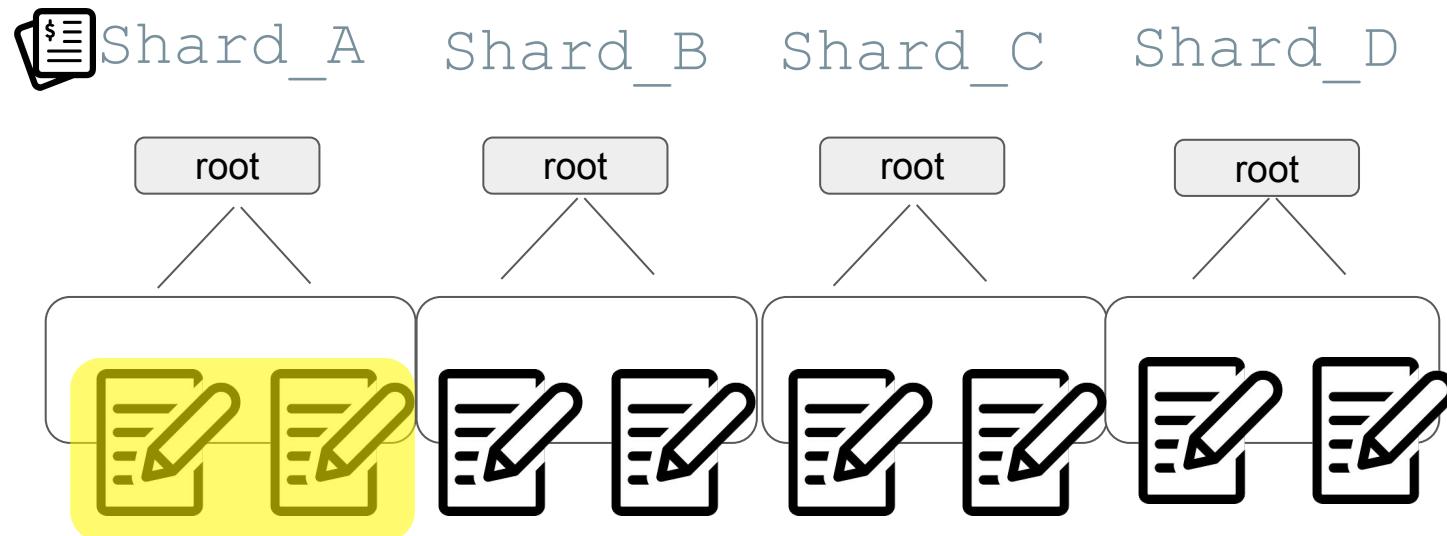
# Multiple State Spaces



Shard\_A    Shard\_B    Shard\_C    Shard\_D



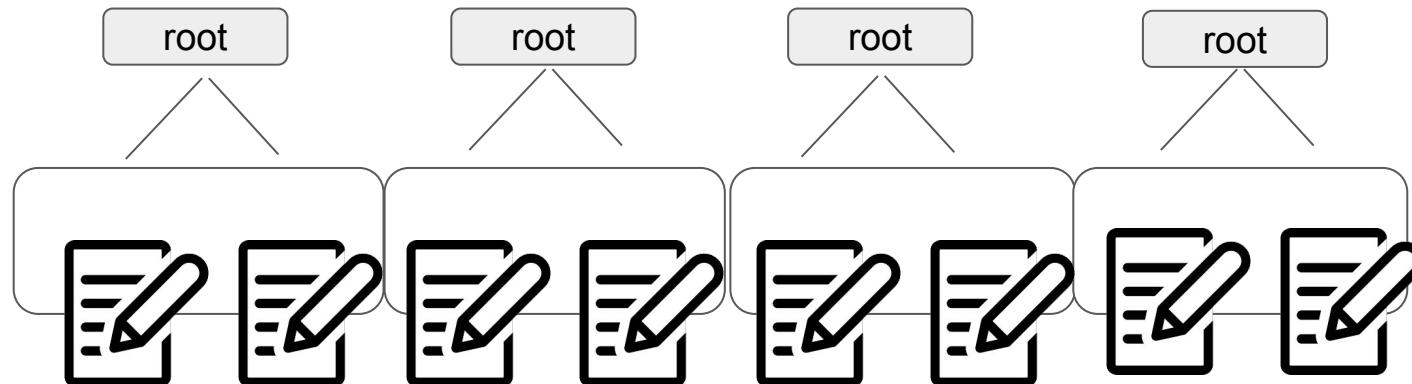
# Multiple State Spaces



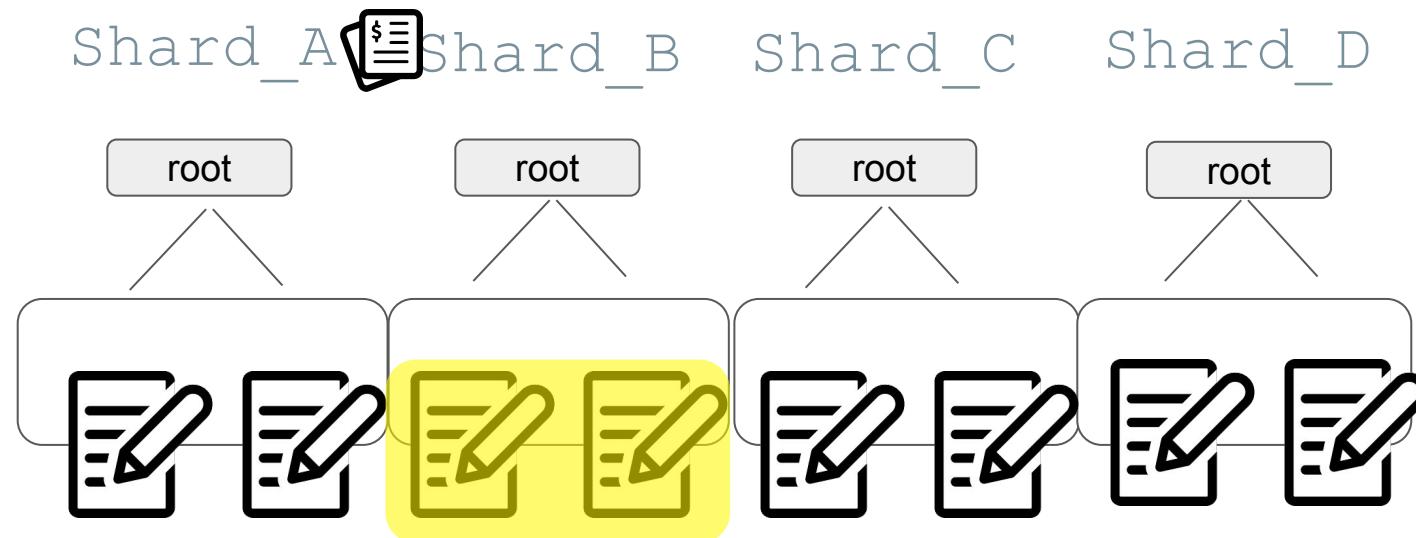
# Multiple State Spaces



Shard\_A    Shard\_B    Shard\_C    Shard\_D



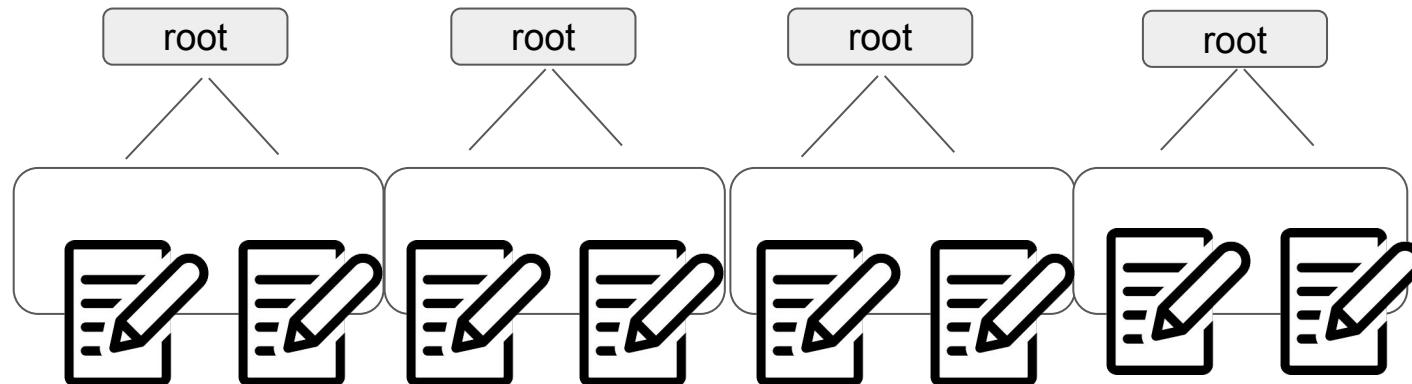
# Multiple State Spaces



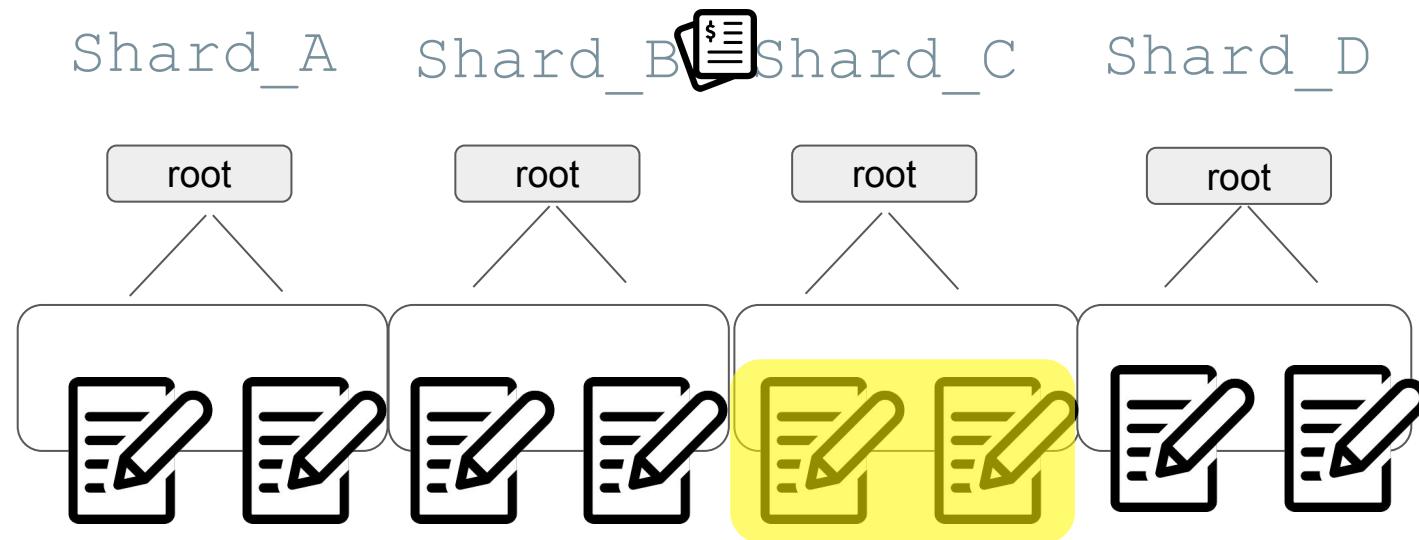
# Multiple State Spaces



Shard\_A    Shard\_B    Shard\_C    Shard\_D



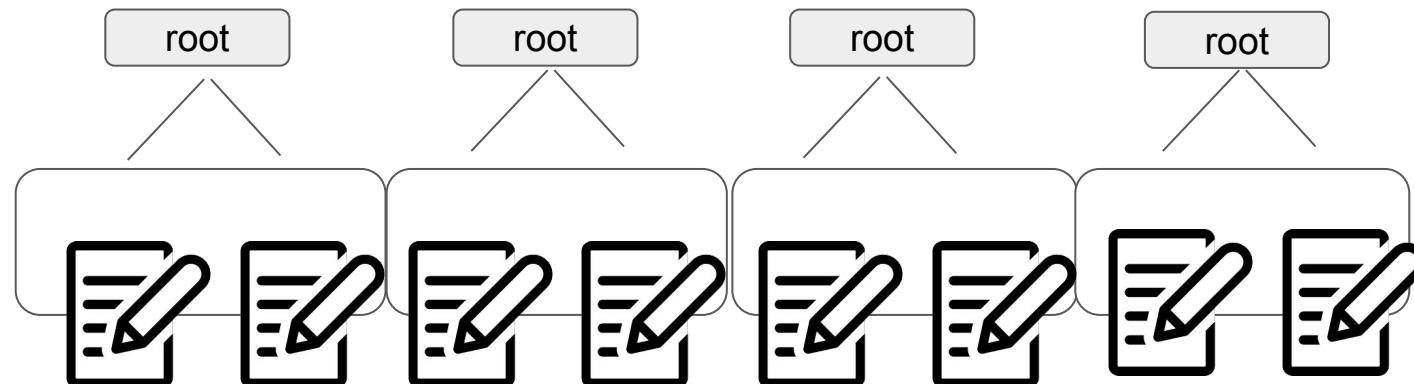
# Multiple State Spaces



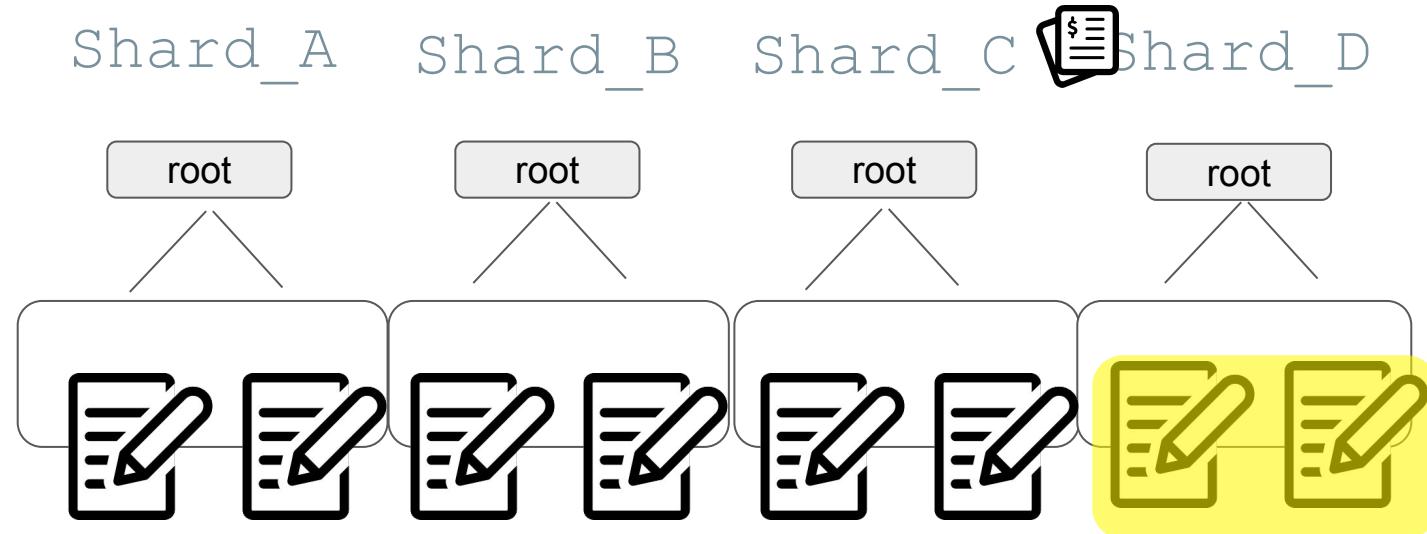
# Multiple State Spaces



Shard\_A    Shard\_B    Shard\_C    Shard\_D



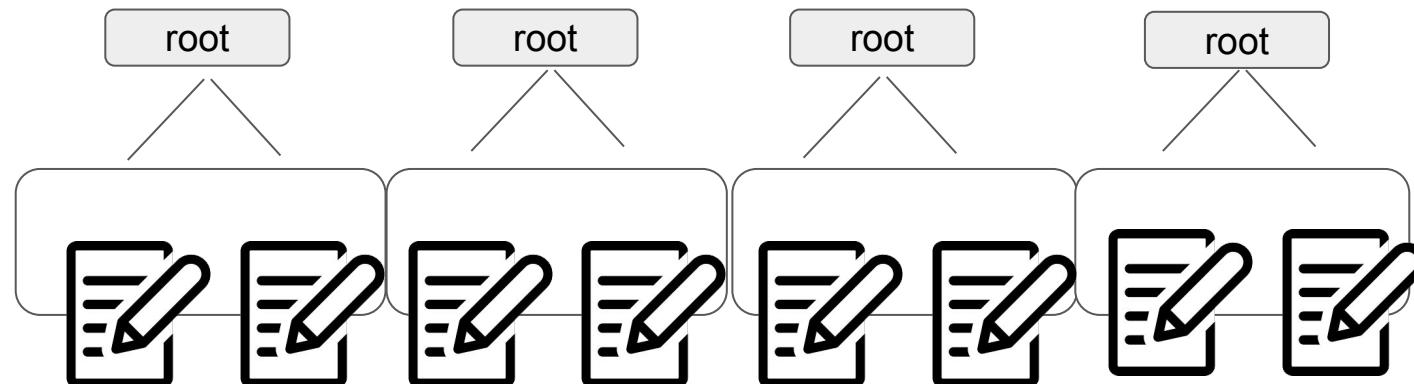
# Multiple State Spaces



# Multiple State Spaces

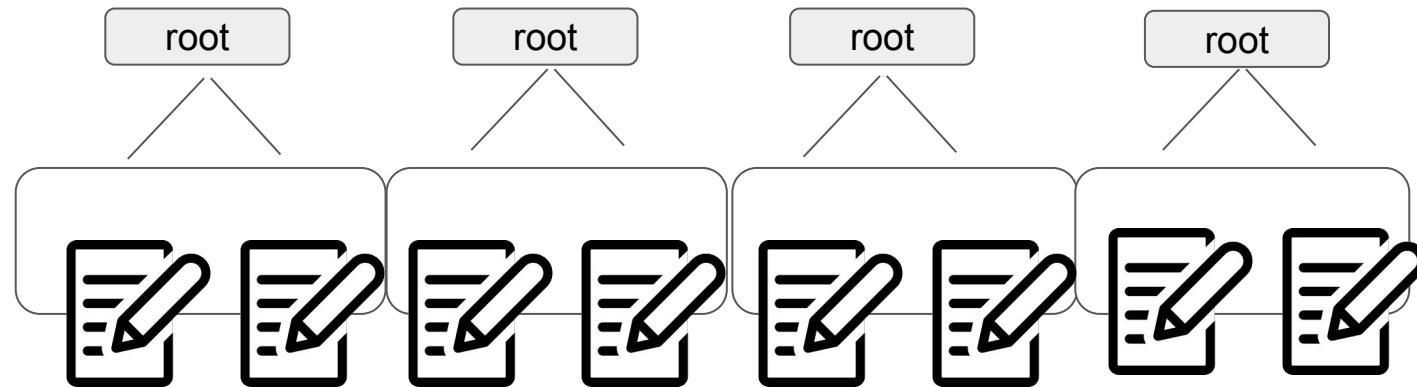


Shard\_A    Shard\_B    Shard\_C    Shard\_D



# Multiple State Spaces

Shard\_A    Shard\_B    Shard\_C    Shard\_D



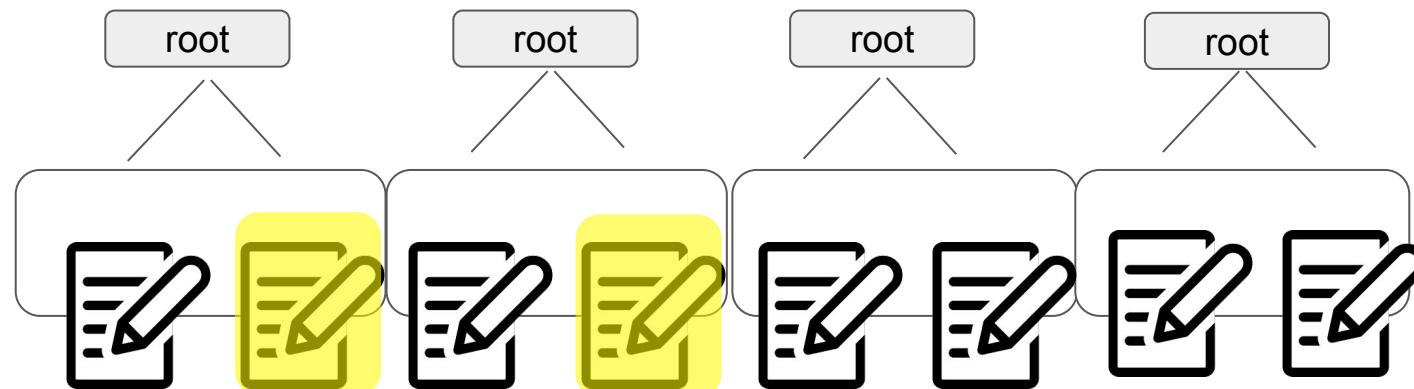
## DeFi in ETH2: Why are we worried?

- **Multiple state spaces.**
  - Shard\_a, shard\_b, shard\_c, etc.
  - What do I deploy to?
- **Atomic txs have added complexity.**
  - Do we enforce in L1?
  - Do we enforce in L2?

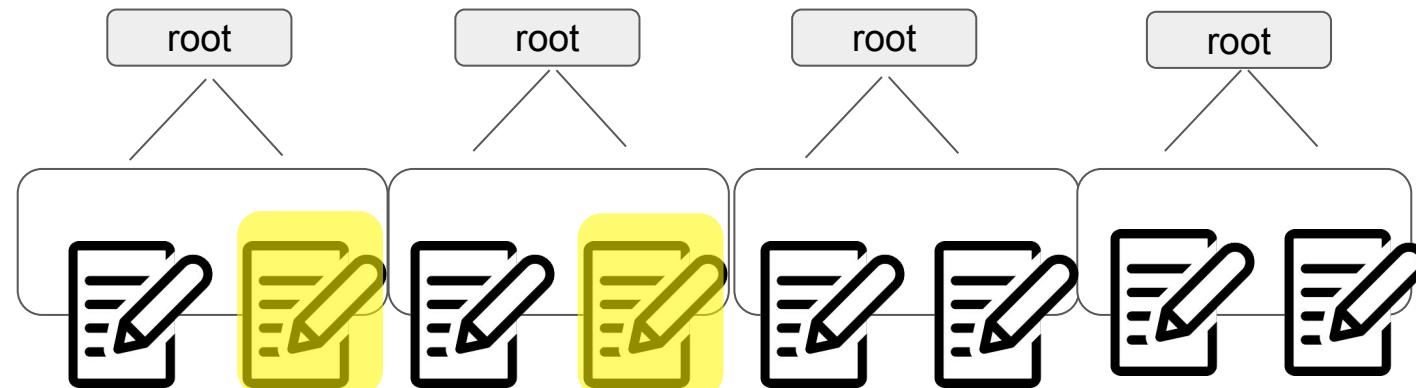
# Atomic Cross Shard Communication



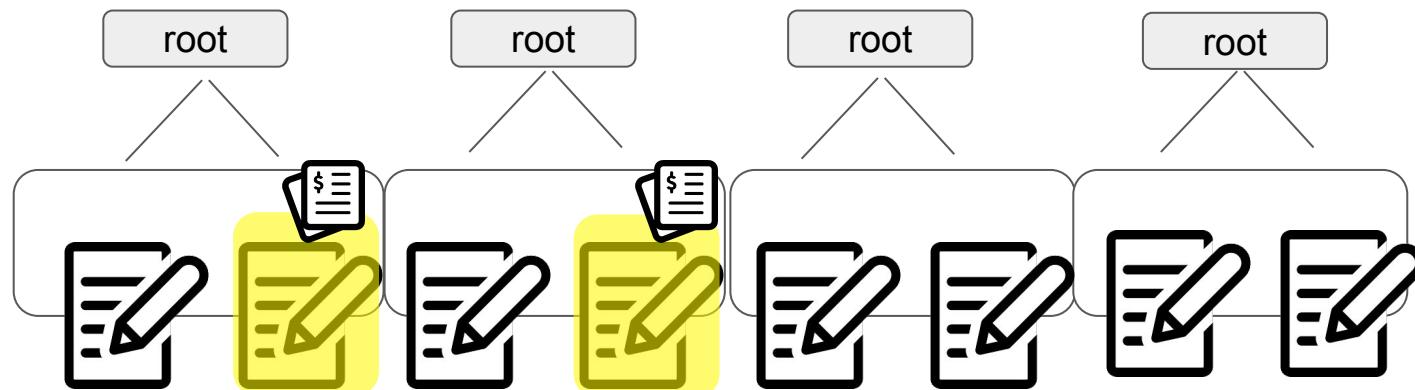
# Atomic Cross Shard Communication



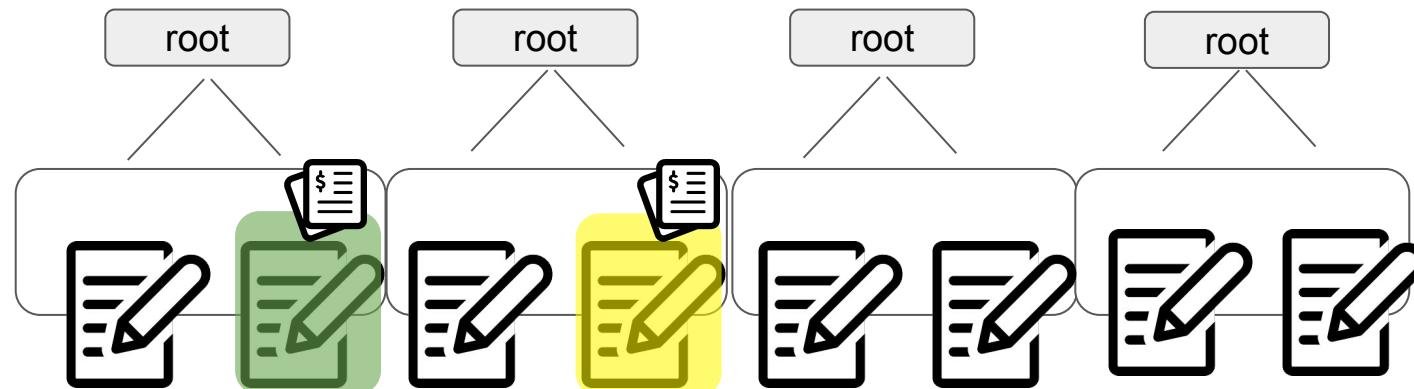
# Atomic Cross Shard Communication



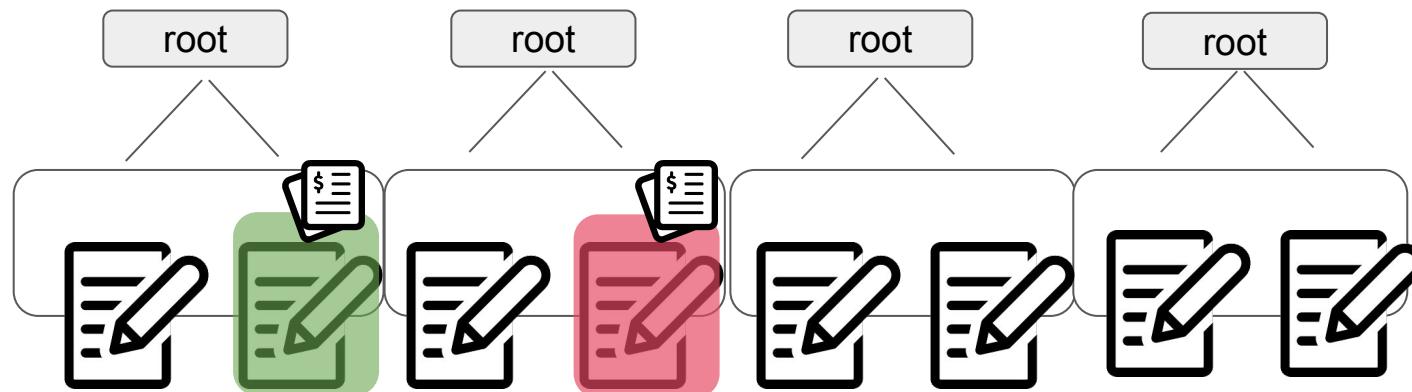
# Atomic Cross Shard Communication



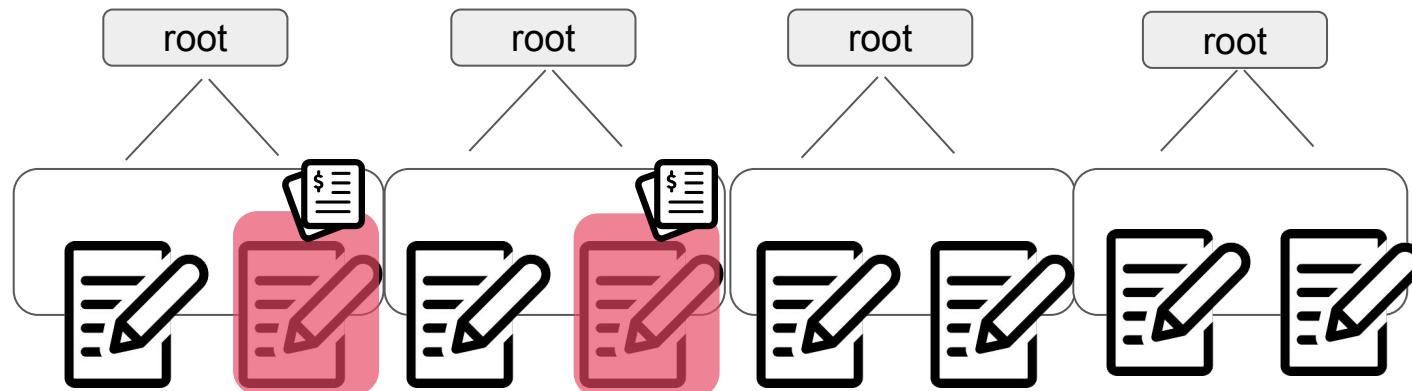
# Atomic Cross Shard Communication



# Atomic Cross Shard Communication

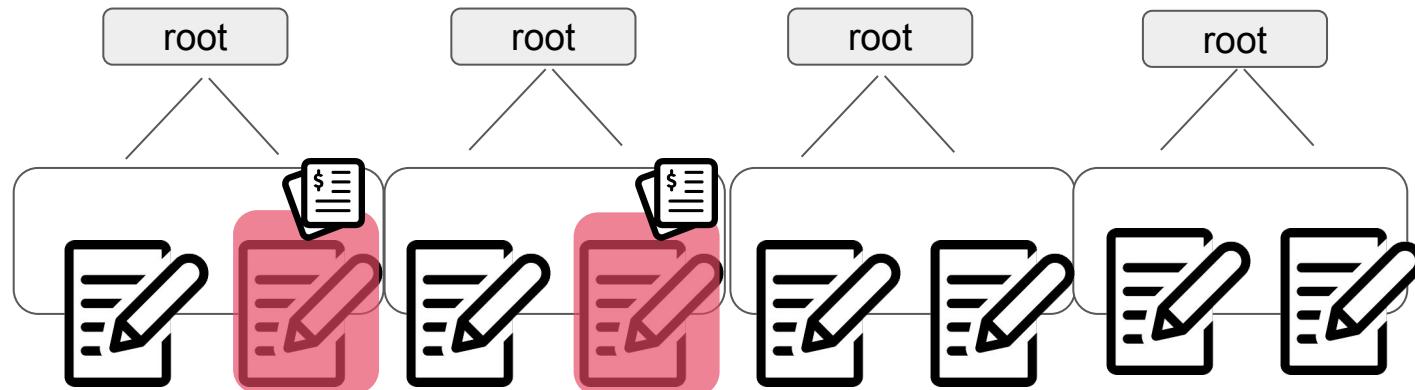


# Atomic Cross Shard Communication



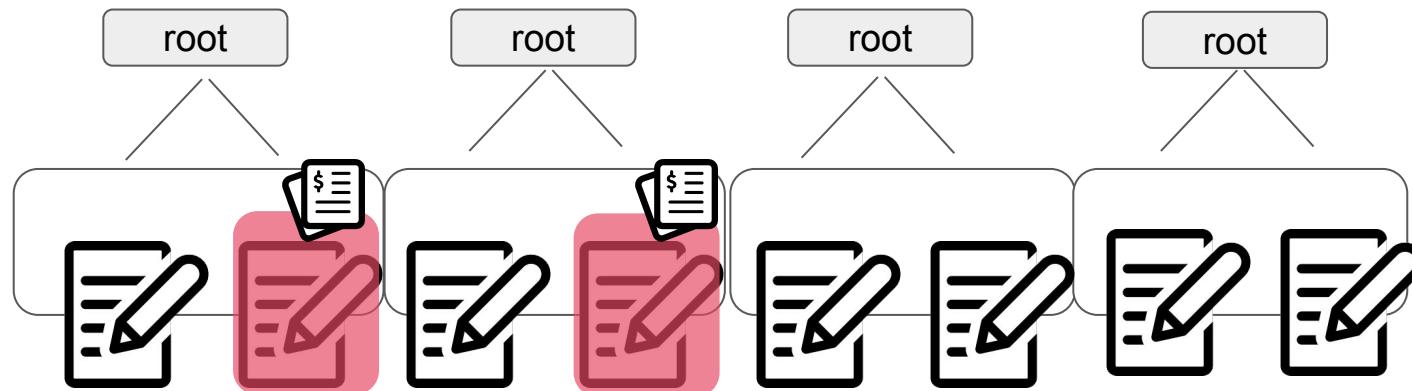
# Atomic Cross Shard Communication

Train & hotel problem!



# Atomic Cross Shard Communication

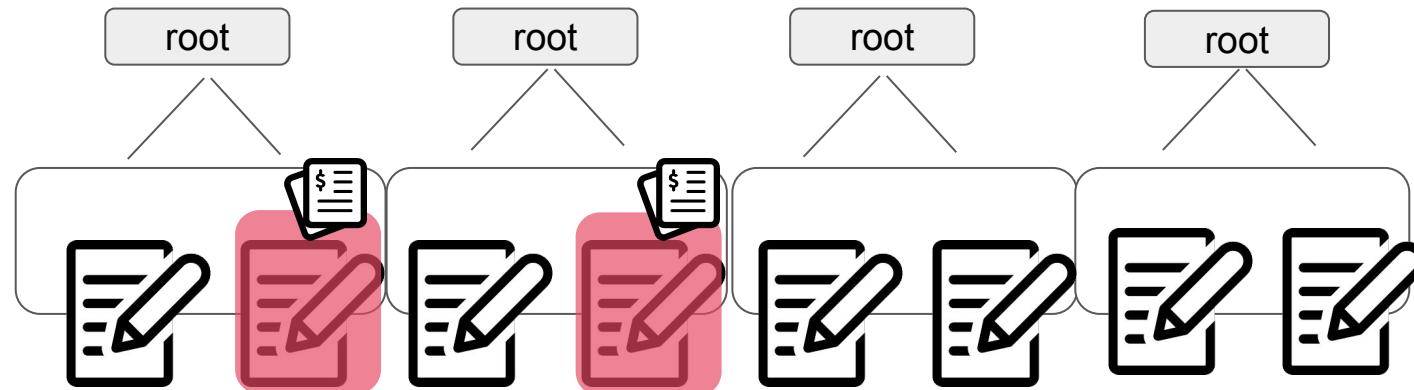
~~Train & hotel problem!~~



# Atomic Cross Shard Communication

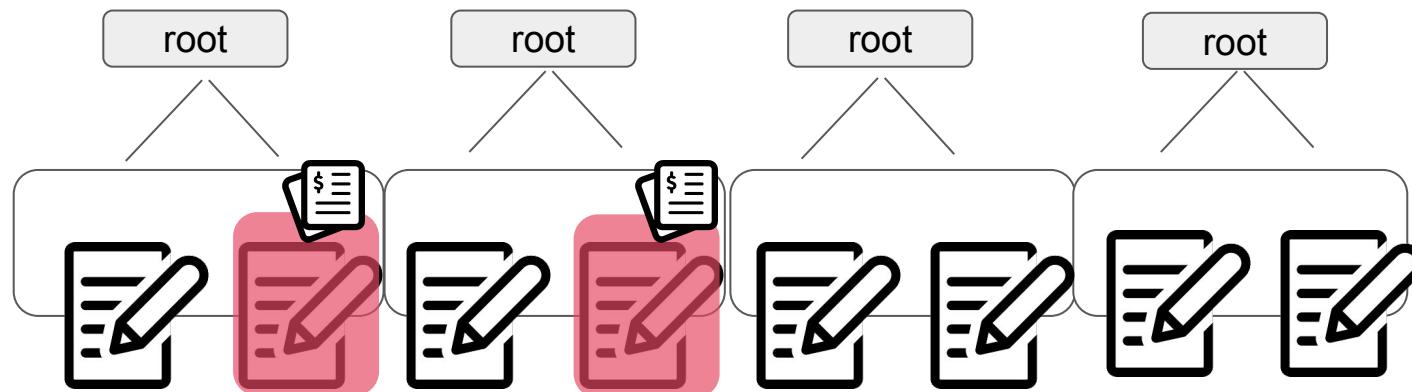
Risk-free arbitrage problem!

~~Train & hotel problem!~~



# Atomic Cross Shard Communication

How do we achieve this?



# Atomic Cross Shard Communication

How do we achieve this?



# Atomic Cross Shard Communication

How do we achieve this?

- 1) Layer 1, or<sub>(and)</sub> 2) Layer 2



# Layer 1 Cross Shard



# Receipts!

## Cross-shard contract yanking

Sharding    cross-shard



vbuterin

1 Mar '18

*Special thanks to Piper Merriam for helping to come up with this idea*

This is a generalization of [cross-shard locking schemes](#) and similar techniques to enabling cross-shard activity to solve train-and-hotel problems. Philosophically speaking, “locking” a contract that exists on shard A effectively means freezing its state on shard A, saving the state into a receipt, importing the contract to shard B, performing some operation between that contract and some other object on shard B, then using another receipt to send the contract back to shard A where it then continues its existence.

We can simplify and generalize this by changing the mechanism from “locking” to “yanking”. We add an opcode to the EVM, YANK (one stack argument: target\_shard), which deletes the contract from the state and issues a receipt that contains the state of the contract and the target\_shard; this receipt can then be processed in the target\_shard to instantiate the same contract in that shard. Contracts are free to specify their own conditions for when they get yanked.

# Honorable Mention: Merge Blocks



## Merge blocks and synchronous cross-shard state execution

Sharding

cross-shard



vbuterin

Feb '18

We know that with [merge blocks and clever use of fork choice rules](#) 100, we can have a sharded chain where a block on one shard is atomic with a block on another shard; that is, either both blocks are part of the final chain or neither are, and within that block there can be transactions that make synchronous calls between the two shards (which are useful for, among other things, solving [train-and-hotel problems](#) 43). However, there remains a challenge: how do we do state execution for such blocks? That is, suppose that we have block A on shard 1 and block B on shard 2, which are merged: how does a node

# Layer 2 Cross Shard



## Fast cross-shard transfers via optimistic receipt roots

Sharding ■ phase-2 ■ cross-shard



vbuterin

4 🖊 Apr 21

**TLDR:** given a base layer blockchain that offers slow asynchronous cross-shard transactions, we can emulate fast asynchronous cross-shard transactions as a layer on top.

This is done by storing “conditional states”, for example if Bob has 50 coins on shard B, and Alice sends 20 coins to Bob from shard A, but shard B does not yet know the state of shard A and so cannot fully authenticate the transfer, Bob’s account state temporarily becomes “70 coins if the transfer from Alice is genuine, else 50 coins”. Clients that have the ability to authenticate shard A and shard B can be sure of

# Layer 2 Cross Shard



## Fast cross-shard transfers via optimistic receipt roots

Sharding ■ phase-2 ■ cross-shard



vbuterin

4 🖊 Apr 21

**TLDR:** given a base layer blockchain that offers slow asynchronous cross-shard transactions, we can emulate fast asynchronous cross-shard transactions as a layer on top.

This is done by storing “conditional states”, for example if Bob has 50 coins on shard B, and Alice sends 20 coins to Bob from shard A, but shard B does not yet know the state of shard A and so cannot fully authenticate the transfer, Bob’s account state temporarily becomes “70 coins if the transfer from Alice is genuine, else 50 coins”. Clients that have the ability to authenticate shard A and shard B can be sure of

Honorable Mentions . . .

# Honorable Mentions . . .



## Phase 2 pre-spec: cross-shard mechanics

Eth2 phase 2 ■ phase-2 ■ cross-shard



vbutterin

5 Feb 11

### THIS IS A WORK IN PROGRESS!

The goal of this post is to provide a rough outline of what phase 2 might look like, to help make the discussion about state and execution more concrete as well as to give us an idea of the level and types of complexity that would be involved in implementing it. This section focuses on withdrawals from the beacon chain and cross-shard calls (which use the same mechanism); rent is unspecified but note that a hibernation can be implemented as a forced cross-shard call to the same shard.

Topics covered:

- **Addresses**
- **Cross-shard receipt creation and inclusion**
- **Withdrawals from the beacon chain**

# Honorable Mentions . . .



## Phase 2 pre-spec



Eth2 phase 2 ■ phase-2 ■ cross-shard



vbutterin

### THIS IS A WORK IN PROGRESS

The goal of this post is to discuss about state of complexity that would be introduced by beacon chain and cross-shard communication. Hibernation can be introduced to make the system more robust.

Topics covered:

- Addresses
- Cross-shard requests
- Withdrawals from shards

## research

# Simple synchronous cross-shard transaction proposal

### Sharding

■ cross-shard



vbutterin

Status: an idea I've already talked with some people before, but still deserves to be written down in one place and hasn't yet.

We already know that it is relatively easy to support asynchronous cross-shard communication, so we can extend that to solve train-and-hotel problems via cross-shard yanking.

# Honorable Mentions . . .



Phase

Eth2 phase



T

T

di

of

be

a

Tc



Sign In

## Synchronous cross-shard transactions with consolidated concurrency control and consensus (or how I rediscovered Chain Fibers)

Sharding    cross-shard



cdetrio

Jun '18

I'd like to argue against a widespread perception that under the initial sharding protocol, the only way to do cross-shard transactions will be with asynchronous messages (i.e. contract locking/yanking, and so on). I do agree that if we expect the least amount of innovation from protocol engineers, then asynchronous transactions are what we'll get (asynchronous transactions are the solution if protocol engineers shirk it off and leave it as a problem to be solved by users, contract authors and dapp devs). But it would be a shame to accept a "worse is better" fate. Instead, I want to highlight some insight why solving the problem at the protocol layer, enabling synchronous cross-shard transactions that would we can extend that to solve train-and-hotel problems via cross-shard yanking

15

## Honorable Mentions . . .



### Layer 2 state schemes

PI

Sharding

Eth



vbuterin

Jul

*Special thanks to Karl Floersch and Ben Jones for ideas around Plasma with on-chain publication*

An interesting class of constructions that is starting to appear for eth2 is that where the blockchain is used as a data and computation layer, but there is not a trivial direct mapping between the blockchain state and the actual state (eg. account balances) that users care about.

One example of this is [A layer 2 computing model using optimistic state roots](#) 51, where we create a higher-layer scheme to pretend that cross-shard transactions happen much more quickly than they actually “settle” by storing dependency graphs as part of the state and letting clients figure out what the state *will* resolve to when all the dependencies settle

## Honorable Mentions . . .



### Layer 2 state schemes

PI

Sharding

Eth



vbuterin

*Special thanks to Ka*

An interesting class of protocols used as a data and computation layer between the blockchain state and the actual application logic.

One example of this higher-layer scheme is the "Layer 2" state that actually "settles" by settling the state of the underlying blockchain.

### Various kinds of scaling, execution models, a

Sharding



ryanreich

tl;dr: We mean a lot of things when we say “scalability”, and we can’t ignore the fact that if we want to support a particular execution model scalably, we are missing something. One way to do this could be realized by formulating a more ideal execution model. Ultimately, we can gain a lot of benefits from this, such as better security, faster transaction times, and gain cool stuff like multi-chain communication and separate execution environments.

Links that appear below:

- Phase One and Done 7

# Honorable Mentions . . .

research

## Layer 2 state schemes

PI Sharding

Eth



vbuterin

*Special thanks to Ka*

An interesting class of protocols used as a data and computation layer between the state and the actual application.

One example of this higher-layer scheme is the "Layer 2" system that actually "settles" by state, will resolve to a

Various kinds of scaling, execution models, a

Sharding



symantech

# TOO MUCH ! !

tl;dr: We mean a lot of things when we say “scalability”, and we can’t ignore the fact that if we want to support a particular execution model scalably, we are missing something. In fact, what we could be doing is trying to realize what we could be realized by formulating a more ideal execution model. Ultimately, we can gain a lot of things, and gain cool stuff like multi-chain communication and separate execution environments.

Links that appear below:

- Phase One and Done ↗

So what's the difference?

- **Layer 1**
  - Simple(r)
  - Can be slower (based on cross-link time)
- **Layer 2**
  - Complex gas payments!
    - Can require DEX between shards
  - Extremely flexible

## Agenda

1. What do we mean by tightly coupled Ethereum?
2. What are our options for tightly coupling ETH2?
3. How will DeFi look on scalable Ethereum?

## Agenda

1. What do we mean by tightly coupled Ethereum?
2. What are our options for tightly coupling ETH2?
3. How will DeFi look on scalable Ethereum?

What is the future?



SEEMS LEGIT

What is the future?



SEEMS LEGIT

What is the future?

What is the future?

Layer 1

Layer 2

What is the future?

Layer 1

Layer 2



What is the future?

## **Layer 1**

Simplest cross  
shard tx scheme  
adopted.

## **Layer 2**

# What is the future?

## Layer 1

Simplest cross shard tx scheme adopted.



## Layer 2

# What is the future?

## Layer 1

Simplest cross shard tx scheme adopted.



## Layer 2

Optimistic rollup & plasma adopt fun L2 cross shard TXs schemes.

# What is the future?

## Layer 1

Don't ask<sup>cross</sup>  
permission.<sup>heme</sup>  
adopted.



You won't need  
forgiveness.



## Layer 2

Optimistic rollup  
& plasma adopt  
fun L2 cross  
shard TXs  
schemes.

# What is the future?

## Layer 1

Don't ask<sup>cross</sup>  
permission.<sup>heme</sup>  
adopted.



You won't need

Wait, what's

“Optimistic Rollup”?

## Layer 2

Optimistic rollup  
& plasma adopt  
fun L2 cross  
shard TXs  
schemes.

What is the future?

Layer 1  
Don't panic.  
So glad you asked!  
on. heme  
pted.

You won't need

Wait, what's

“Optimistic Rollup”?

so glad you asked! ; )

**Layer 2**

Optimistic rollup  
& plasma adopt  
fun L2 cross  
shard TXs  
schemes.

Optimistic Rollup! Unipig! Yay!

Optimistic Rollup! Unipig! Yay!



Optimistic Rollup! Unipig! Yay!



Optimistic Rollup! Unipig! Yay!

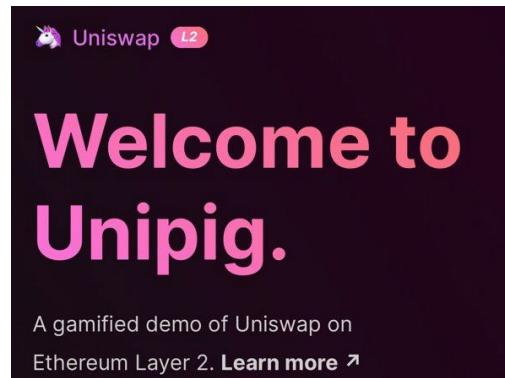


Uniswap

# Optimistic Rollup! Unipig! Yay!



# Optimistic Rollup! Unipig! Yay!



# Uniswap

Uniswap implemented on Ethereum L2  
using **Optimistic Rollup!**

DeFi in L2!

## **DeFi in L2!**

Chat with us  
if you want to  
join Uniswap &  
expand into  
L2!

**We're all in  
this together!**

**DeFi in L2!**

Chat with us  
if you want to  
join Uniswap &  
expand into  
L2!

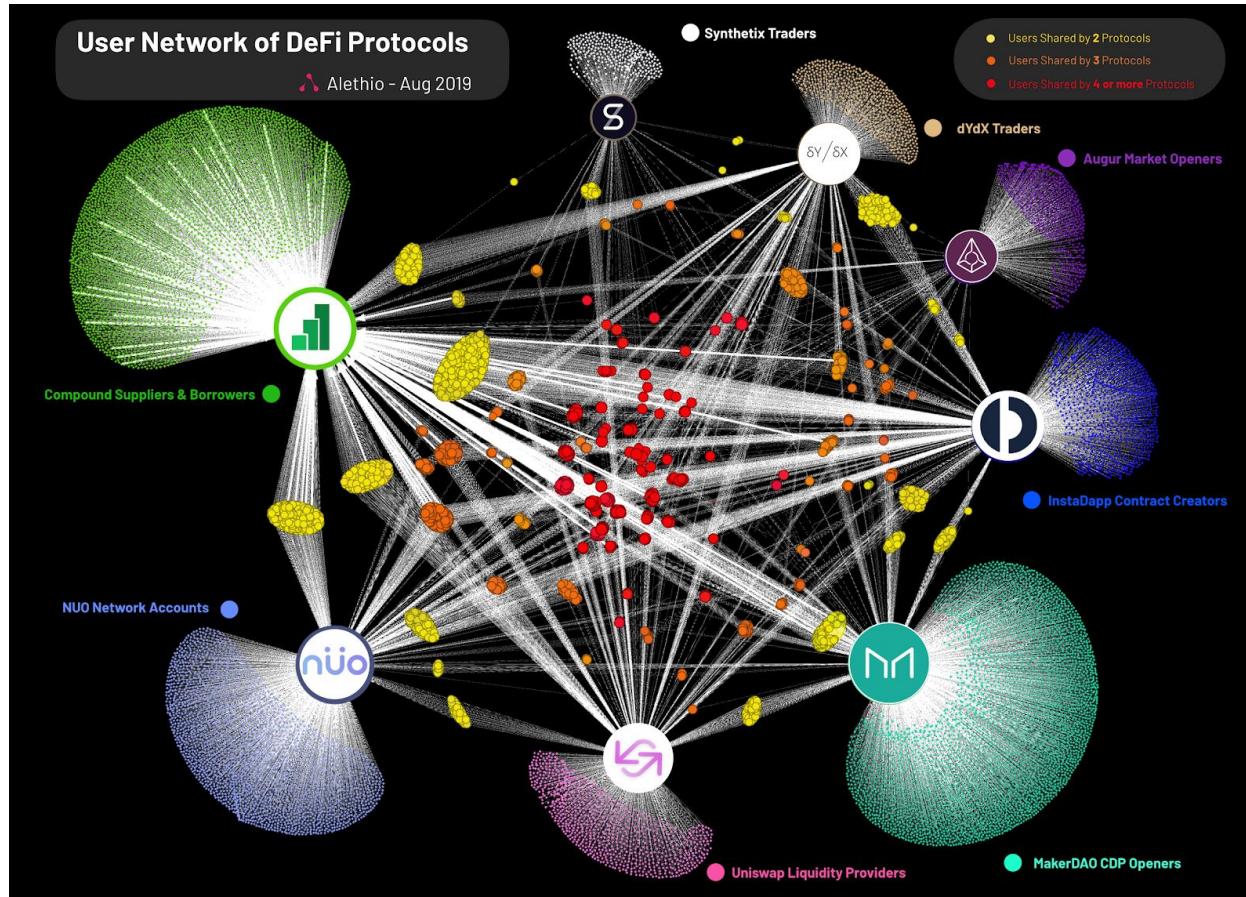
**We're all in  
this together!**



**I WANT YOU**

To Defend the Rights of Animals.

# DeFi in L2!



# DeFi in L2!

---



Uniswap Liquidity Providers



MakerDAO CDP Openers

# Thank you!



plasma.group

# Thank you!



plasma.group

# Thank you!

Let's Chat!

[plasma.group](https://plasma.group)