While introductions to

EigenLayerare often focused on

restaking, what got me interested was learning how this was just the means to an end to accomplish something exciting and powerful, and that I'm still wrapping my head around.

EigenLayer enables developers to build arbitrary distributed systems on top of the Ethereum trust network.

It helps developers launch chains, decentralized networks, and Proof of Stake (PoS) systems at 10x speed, ease, security, and distribution than possible before.

It brings the power of Ethereum outside the EVM to additional apps on the network, making it an order of magnitude easier to build these systems by extending the programmability of Ethereum.

What does EigenLayer solve

Developers who build decentralized infrastructure face the challenge of establishing their own economic security.

While developers can use protocols like Ethereum to provide economic security for smart contracts,

infrastructures like bridges, sequencers, oracles, and specialized data availability layers require their own economic security

.

EigenLayer solves this problem by enabling

any

service, regardless of its composition (e.g. EVM-compatibility), to tap into the pooled security of Ethereum's stakers.

To put it simply, EigenLayer makes it much easier and more economically viable to build chains, distributed systems, or any PoS network.

How blockchain networks have evolved

To understand this better, let's take a quick look at how blockchains have evolved over the past 15 years.

Application-specific blockchains

The first blockchain networks, like Bitcoin and Litecoin, were application-specific.

If someone wanted to build a slightly different network, they had to build the entire protocol from the ground up.

Therefore, blockchain developers at this time were literally building or maintaining the blockchains themselves, there was not concept of a "blockchain application" in the sense of running an app on one of these networks.

Programmable blockchains

With Ethereum and the EVM, blockchains became programmable.

Developers could now easily write a smart contract and deploy it to an existing general purpose chain, inheriting the liquidity and security of the underlying network.

This made developing apps an order of magnitude easier, and lowered the barrier to entry for developers wanting to build.

Programmable blockchains ushered in the current state of innovation, decentralized apps, and web3 — it was now simple to experiment with new ideas without dedicating a large amount of resources in doing so.

Ethereum's rollup-centric roadmap & scalability

As more apps and users have come online, it's become obvious that the initial architecture of Ethereum and the EVM just doesn't scale.

When building software there are two approaches to scalability, vertical and horizontal. Vertical scaling involves increasing compute resources (CPU, RAM, storage, etc.), while horizontal involves adding more servers / instances to distribute load - e.g. adding more app servers behind a load balancer.

Ethereum and the EVM are taking both approaches, with the network itself incorporating a rollup-centric roadmap to accomplish horizontal scaling.

Rollups outsource execution and transaction processing off of the main Ethereum (or L1) network, while still inheriting Ethereum security guarantees.

However, networks and protocols not proven by or deployed to the EVM cannot inherit these security and trust guarantees. Examples include sidechains, oracles, bridges, data availability layers, decentralized databases, decentralized storage, and sequencers among other things.

Bootstrapping a network from scratch

There are major challenges in building these types of networks and services from scratch.

Potential stakers have to first be identified and educated about what you are building.

These stakers also must forego taking part in, and receiving rewards from, other existing, trusted, and well known opportunities. They must be convinced instead to invest a significant amount of money and opportunity cost to take part in your new network, usually by buying the new network's native token, which is generally volatile and hard to get.

Building in this way also results in an undesirable and low trust model for apps using your service. The cost of corruption of the app will now be the minimum cost needed to compromise its weakest infrastructural dependency.

Introducing EigenLayer

Like Ethereum enabled smart contract developers to easily build applications that inherit the security and trust of Ethereum, EigenLayer enables developers to build

any

distributed system or decentralized network, while still leveraging the same security and trust guarantees of Ethereum.

This is a paradigm shift in how decentralized protocols and Proof of Stake networks are built. EigenLayer is already ushering in a new era of permissionless innovation and I believe will unleash a new generation of apps.

At EigenLayer we call these type of services an AVS (Actively Validated Service).

So what are the types of

servicesthat can be built? The design space is infinite, but here are some verticals that developers are already focusing on:

Specialization and experimentation

Like smart contracts drastically lowered the barrier to entry and opportunity cost of new experimentation for blockchain developers,

EigenLayer does the same for developers building any type of chain, distributed system, or PoS network

.

This means developers will be more likely to try new things because, even if they fail, the cost has been low enough for it to make sense.

There is also a higher incentive for specialization, because the opportunity cost is much lower and the incentive mechanisms much more streamlined and efficient.

Just like in the “web2” world we see deep specialization in countless verticals like arbitrary compute, databases, and serverless platforms, we’ll begin to see more specialization for decentralized infrastructure as the cost continues to go down for developers building in this space.

Specialization and experimentation are accelerated when the amount of work needed to build something is reduced dramatically. When you pair that with a better monetary value proposition for builders, there is the potential for unlocking a wave of new innovation.

For instance, to build a PoS network from scratch there are a lot of requirements (thanks to

@13yearoldvc for the diagrams):

With EigenLayer, much of this architecture and handled for you:

Dual Staking

One of the most powerful features of EigenLayer is

dual staking.

Problem

One of the reasons bootstrapping a new PoS network is hard is because it involves issuing a new token, then persuading people to buy and stake the token.

These tokens are often volatile, more risky, and hard to access.

Because the value of the new native token is closely tied to the system's overall security, early PoS networks also face the potential "death spiral".

A decrease in token value weakens the chain's security, resulting in capital flight (decrease in TVL), resulting decrease in native token price.

Solution

With Dual staking, two tokens can be used to secure the same PoS network.

Instead of requiring stakers to buy and hold a new, more volatile token, they can instead stake ETH - a token with lower volatility, deeper liquidity, and more access. Alongside staking ETH,

they can also stake the network's native token

.

You can learn more about how dual staking works

here.

EigenLayer Ecosystem and resources

Interesting AVSs and rollups building on EigenLayer

There are already many AVSs and chains being built on EigenLayer. Check out the EigenLayer

ecosystem pageto learn more.

Hello World

The easiest way to get started building an AVS is to start with

Hello World.

Additional Resources

To learn more about EigenLayer, I've created a curated list of my own favorite learning resources

here.