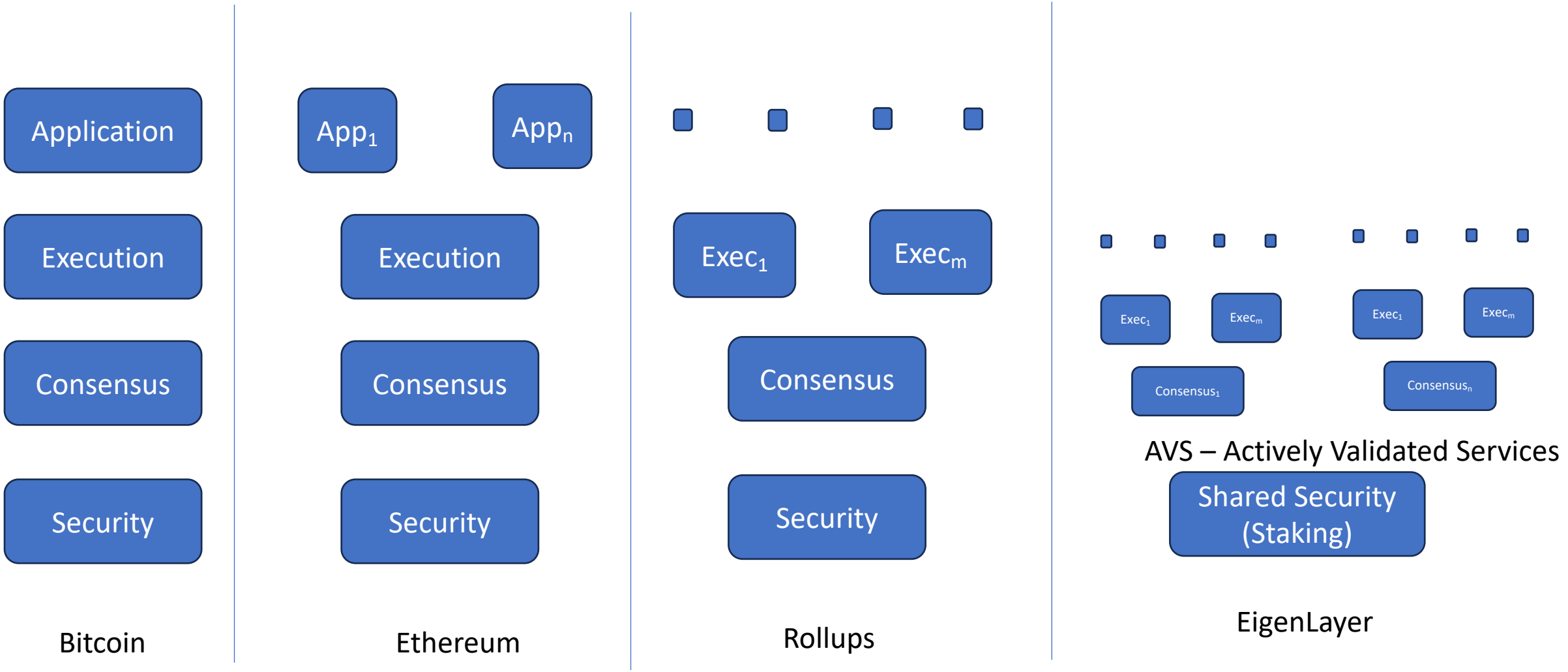
Strong Cryptoeconomic Security for Arbitrary Validation Tasks

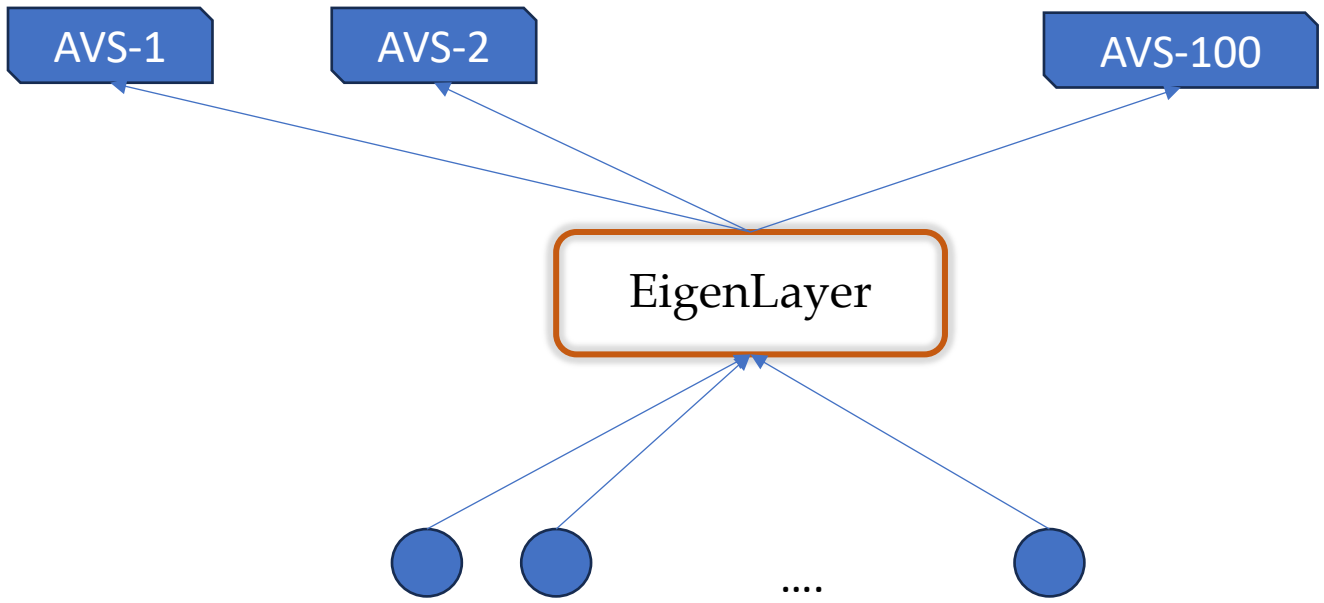
Sreeram Kannan

The Four Levels of Open Innovation



Eigen Layer

Core Problem: Cryptoeconomic security



Validators commit stake

1. Validators commit stake

2. Opt in to performing tasks

3. These tasks are called “AVS”:
   Actively Validated Services

4. Key question:
   How to ensure nodes perform the tasks correctly?

5. Cryptoeconomic security
   If nodes do not perform the task correctly, they will lose a measurable amount of deposit.

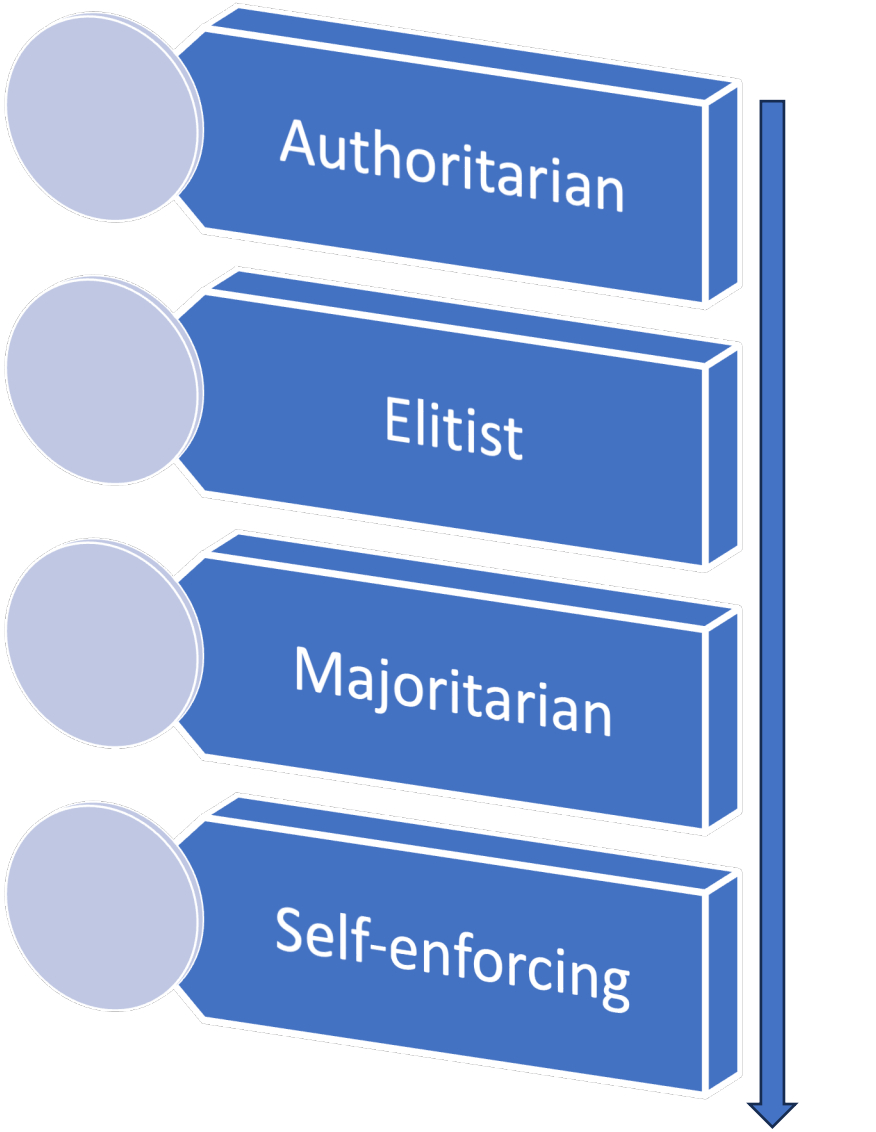
Eigen Layer

Attributability and security

Faults in digital tasks	Attributability	Examples	Security model
Objective faults	Proactively attributable	Deterministic Validity	ZK, CE
	Retroactively attributable	Reorgs	CE
Intersubjective faults	Retroactively attributable	Oracle price feed, validity (without slashing contracts)	CE-New
	Concurrently attributable	Data withholding, Censorship	CE-New
Non-attributable faults	None	Revealing secret shares	

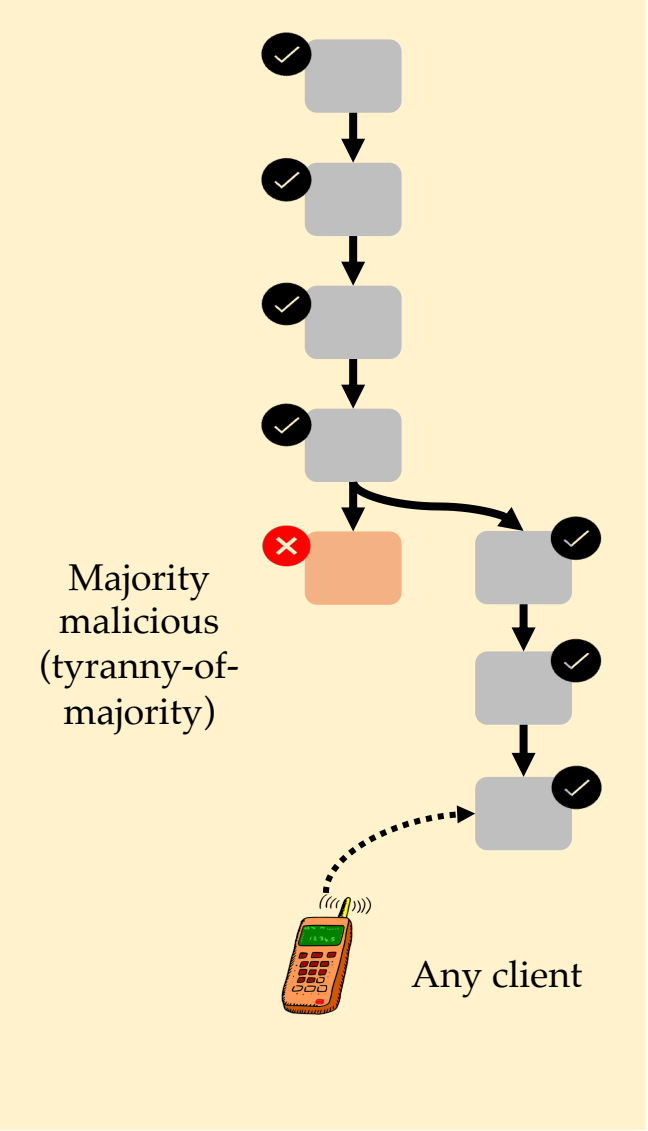
Eigen Layer

Four Types of Coordination Enforcement



Eigen Layer

Slashing by forking in a Blockchain: Self-Enforcing



1. If a majority of validators sign an invalid block this is a problem.
2. The core solution to this problem: clients do not accept invalid blocks.
3. Thus the majority-signed block is forked, and the malicious validators are slashed (they lose their stake).
4. Note: this is only possible for the execution code that the clients validate.
5. Note: slashing only works for assets for which the blockchain is the final ledger of record.

Eigen Layer

Cryptoeconomics for Arbitrary Tasks: Open Problem

Current mechanisms for resolving intersubjectively attributable faults

Majority voting

Slash the stake of operators who diverged from the majority

Committee

Use a "committee" to slash the operators whose responses are not "true"

Slashing-by-forking

Slash the stake by forking the chain

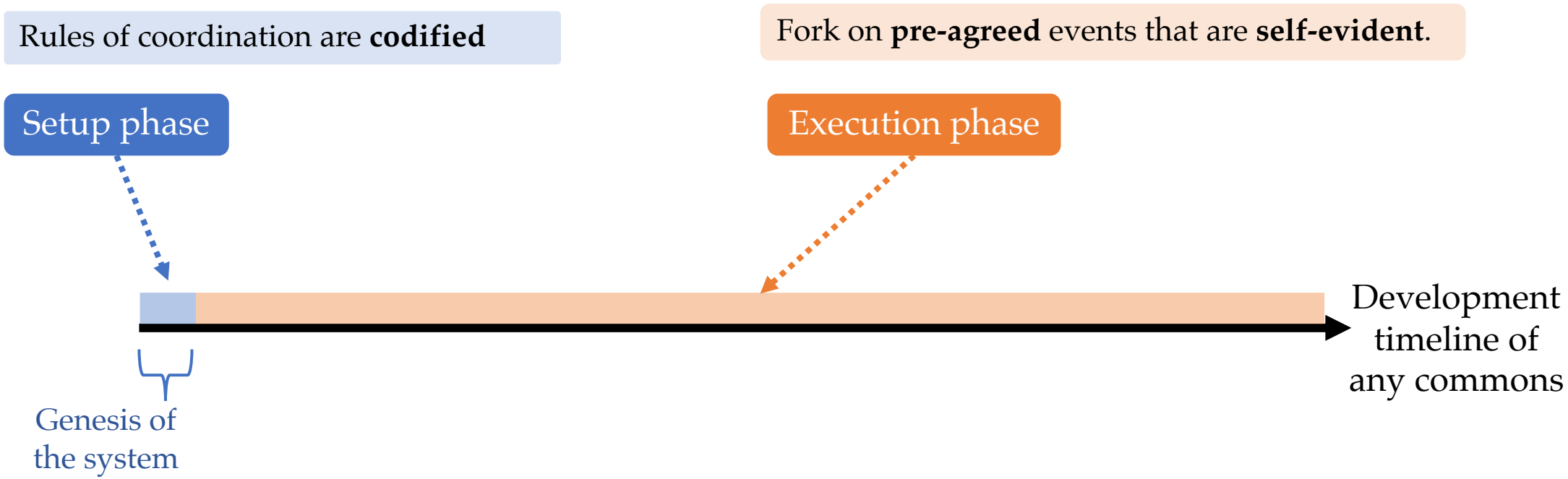
Social consensus: an *unsized* and *limitless* group

Vulnerable to tyranny-of-majority

Open Problem: How do we extend the cryptoeconomics to any intersubjectively attributable fault without forking the chain?

EigenLayer

Core Idea 1: Setup & Execution Phase



Eigen Layer

Core Idea 2: Token forking

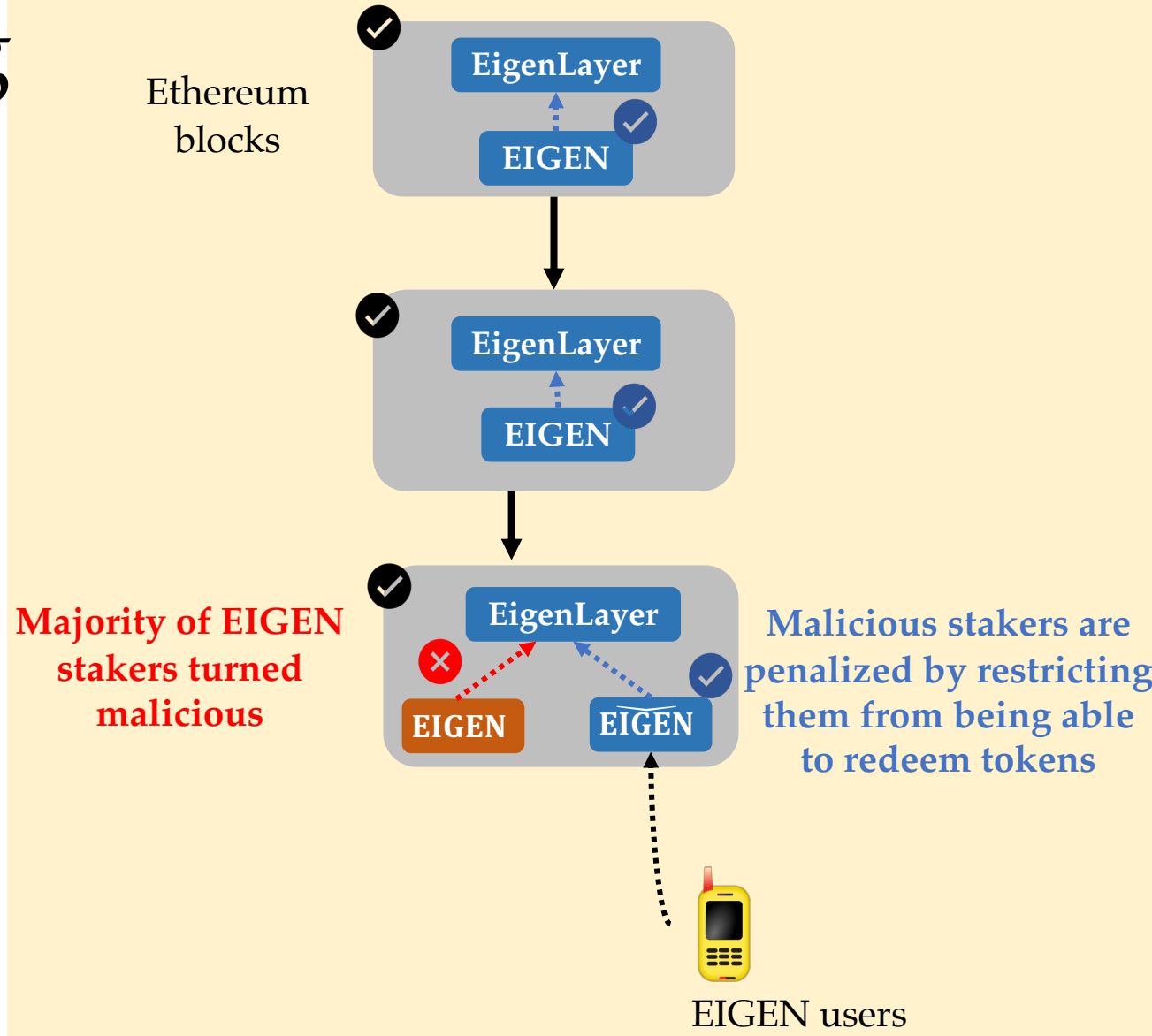
Observation

Value of a token arises from social consensus considering it as valuable

Idea: Fork only token

Fork only token without forking the chain state and use social consensus to induce value to the token fork

Doesn't overload Ethereum's social consensus



EigenLayer

Prior work on token forking

Vitalik Buterin (*circa* 2014-2015)

Augur: a Decentralized Oracle and Prediction Market Platform (v2.0

Jack Peterson, Joseph Krug, Micah Zoltu, Austin K. Williams, and Stephanie Alexander
Forecast Foundation
(Dated: November 1, 2019)

Shortcomings

Specialization to measure profit-from-corruption for only prediction markets.

Every holder of REP token, even if not participating in the market, has to be **fork-aware**

Possible to build **parasitic prediction markets**, thus, making profit-from-corruption unknown

EigenLayer

EIGEN: The Universal Intersubjective Work Token

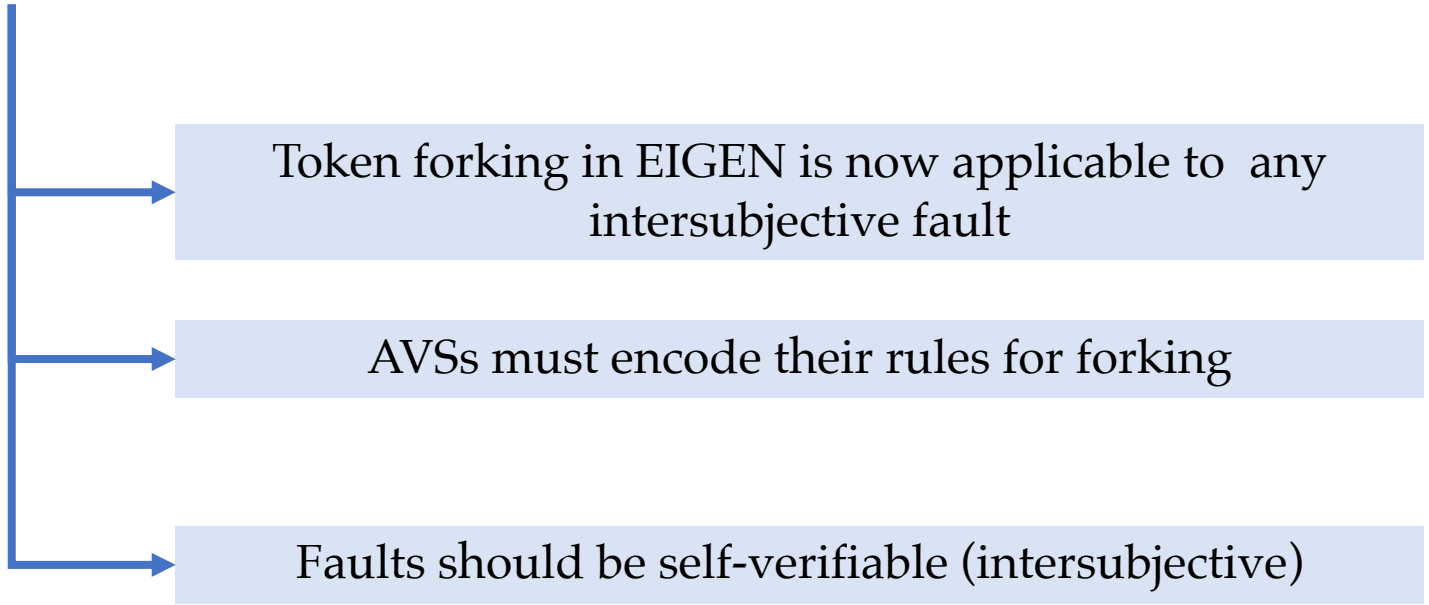
Universality	Applicable to all intersubjective tasks
Isolation	Forking leads to externalities on DeFi => Need isolation between defi and staking / forking
Metering	Forking leads to social cost => meter and charge the social cost
Compensation	Malicious tasks lead to harm for dependent apps => Slash the malicious stakers and redistribute to harmed parties.

Solves long-standing open problems in crypto!

Eigen Layer

Core feature 1: Universality

Setup phase for EIGEN stipulates that:



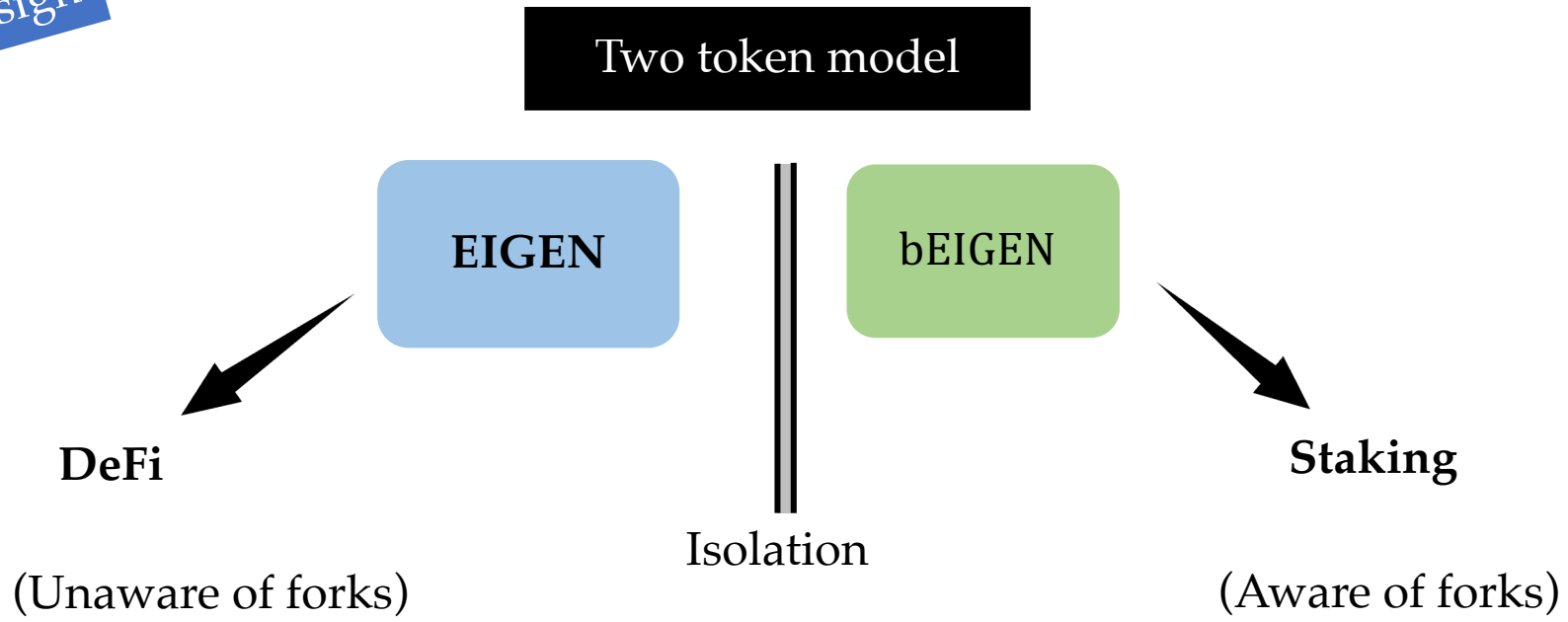
Eigen Layer

Core feature 2: Isolation

Undesirable design

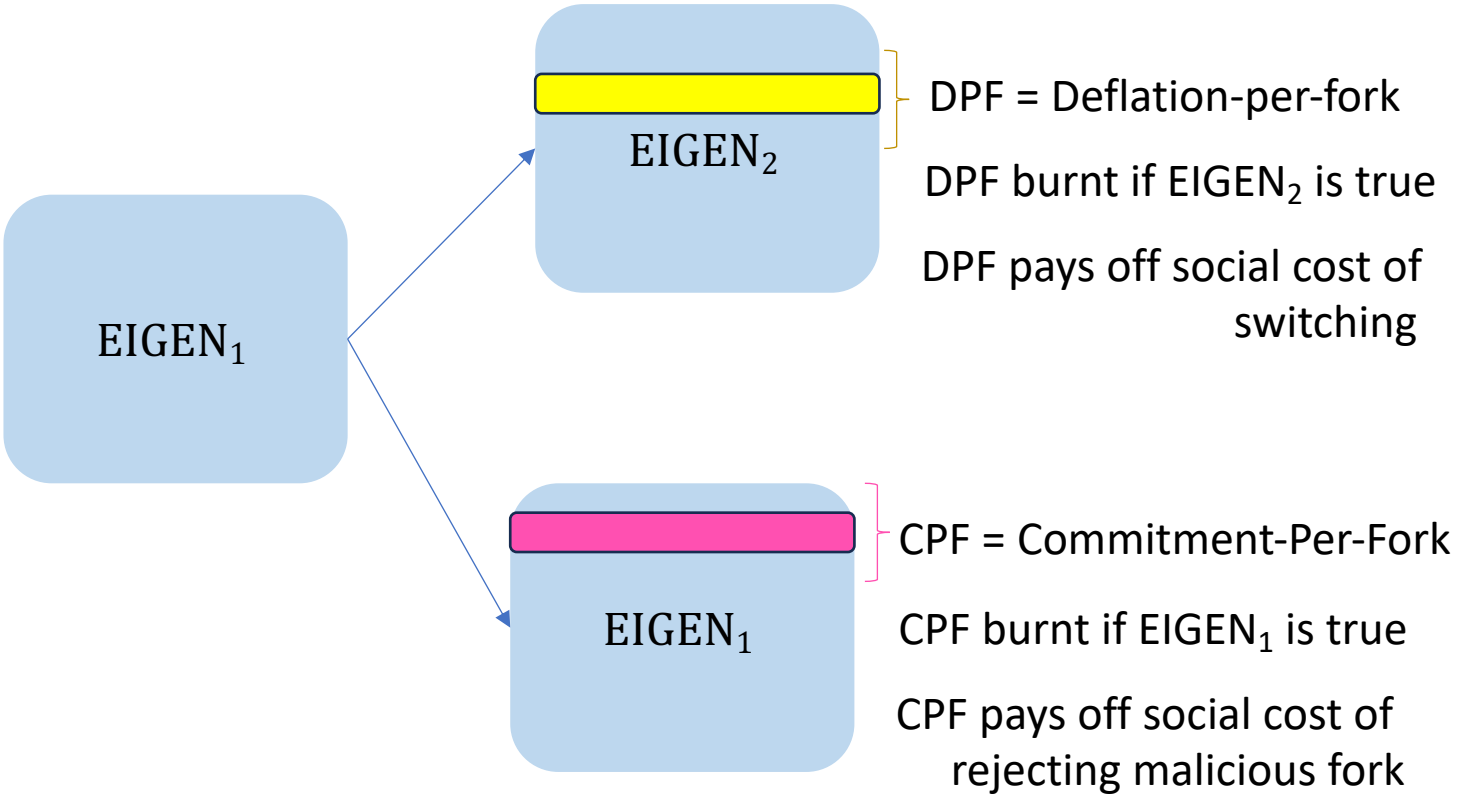
Fork in token → DeFi markets have to be aware of fork → Token not usable for long-term positions

EIGEN's design



Eigen Layer

Core feature 3: Metering



Eigen Layer

Core feature 4: Compensation

Traditional definition

Cryptoeconomic Security

For any attacker, the maximal profit
smaller than the minimum cost enfo

STAKESURE: Proof of Stake Mechanisms with Strong Cryptoeconomic Safety.

Soubhik Deb
EigenLabs
soubhik@eigenlabs.org

Robert Raynor
EigenLabs
rraynor@eigenlabs.org

Sreeram Kannan
EigenLabs
sreeram@eigenlabs.org

Problems

- No way to measure profit-from-corr
- Adversary can engage in parasitic behavior outside the system's locus of measurement
- Harmed user doesn't get compensated for the value lost due to the attack

New definition

Strong Cryptoeconomic Security

For any user, Harm from Corruption < Insured Security

EIGEN: The Universal Intersubjective Work Token

Universality	Applicable to all intersubjective tasks
Isolation	Forking leads to externalities on DeFi => Need isolation between defi and staking / forking
Metering	Forking leads to social cost => meter and charge the social cost
Compensation	Malicious tasks lead to harm for dependent apps => Slash the malicious stakers and redistribute to harmed parties.

Solves long-standing open problems in crypto!

Eigen Layer

What new can you build now with EIGEN?

Any service that involves writing complex fraud proofs

- Gaming VMs
- Databases
- Intent, Order Matching, MEV engines

and many more.

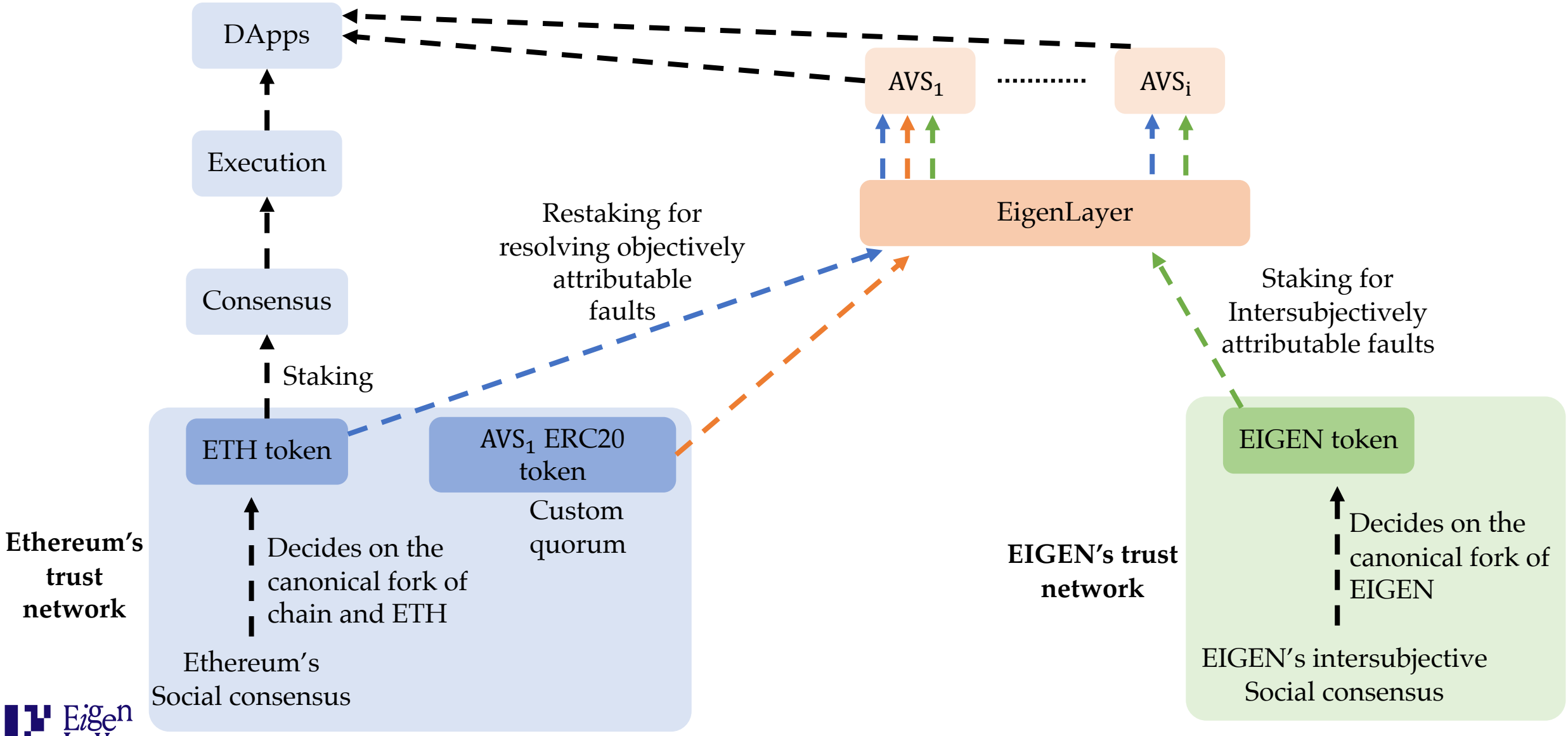
Any service where faults are only observable from outside

- Data availability
- Prediction markets
- Ordering service
- AI Training, Benchmarking, Inference

and many more.

EigenLayer

AVSs can mix-and-match ETH and EIGEN quorums



Eigen Layer

Thanks!!!

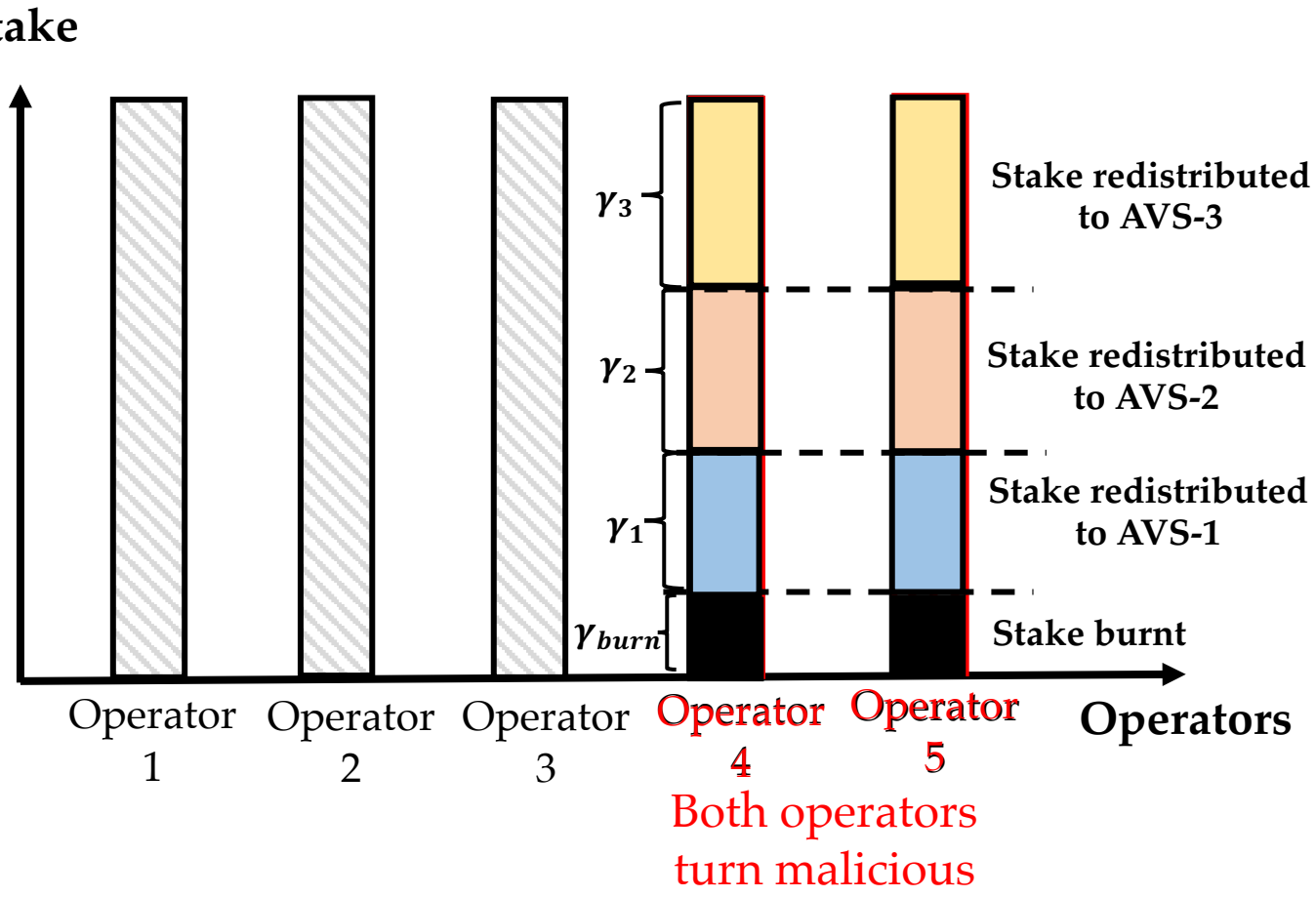
Mechanics of compensation

Assumptions (for simplicity of explanation)

- All operators have equal stake
- They are all opted into 3 AVSs
- Operator 4 and 5 behave maliciously and **succeeds in harming** AVS-1

Protocol for compensation:

- All stake of operator 4 and operator 5 will be slashed as part of token forking.
- A fraction of slashed stake is burnt.
- A fraction of slashed stake gets redistributed to each of the AVSs to which the operators were opted into.

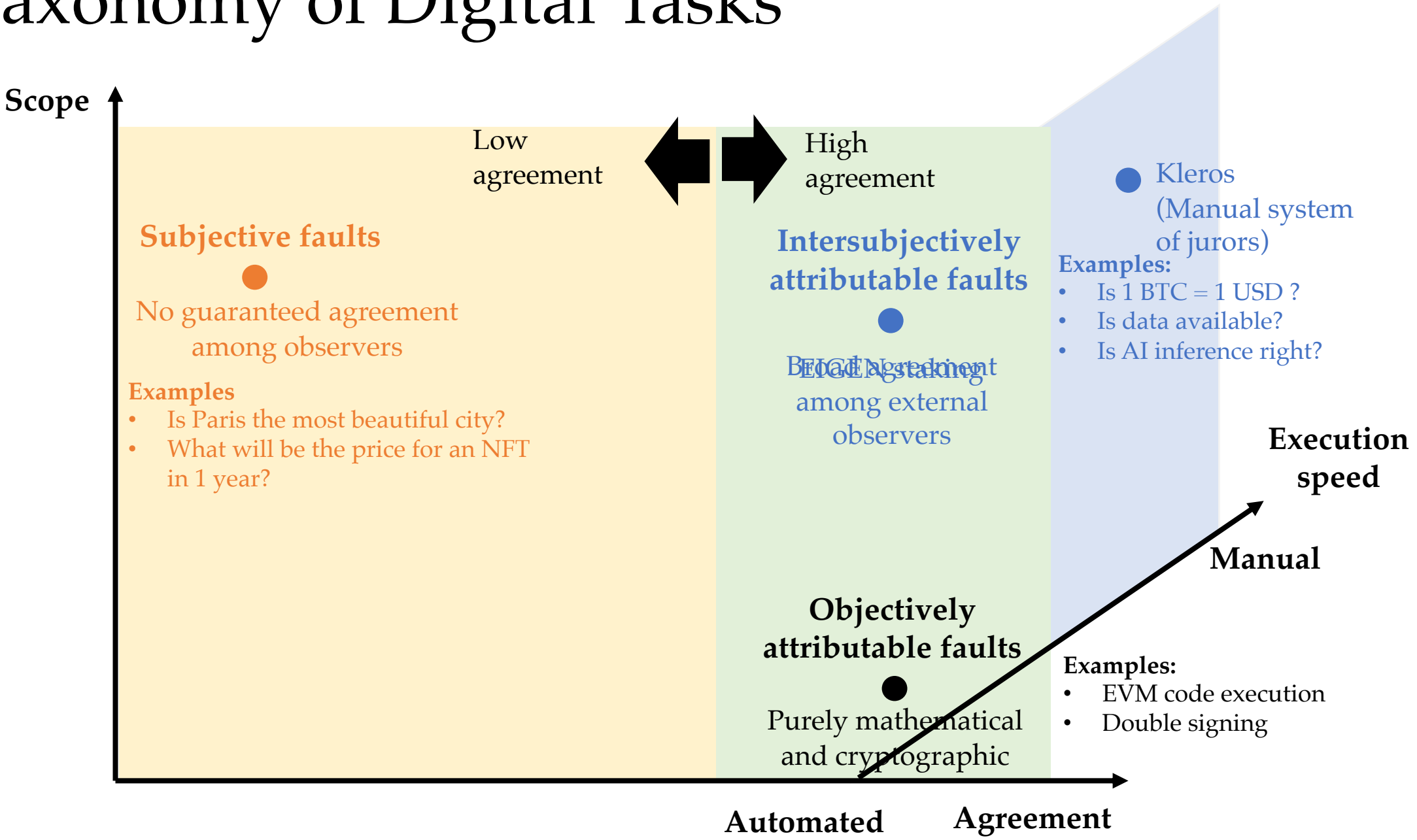


Burning ⟹ Protection against griefing

Redistribution ⟹ Compensation to harmed parties

Eigen Layer

Taxonomy of Digital Tasks



Examples of Setup & Execution Phase

Phases	US Gov.	PoW consensus	Weak subjectivity	Rollup	Sovereign Rollup
Setup phase: Pre-agree on a rule	US Constitution	Longest-chain rule (LCR)	Weak subjectivity checkpoint rule	Follow bridge contract	Example: Follow social consensus to revert hacks
Execution phase: Execute the rule	Laws passed compliant with constitution	Decide on latest block (per LCR)	Computation of weak subjectivity checkpoint	Decide on current rollup block (using bridge state)	Decide on current rollup block (using social consensus)