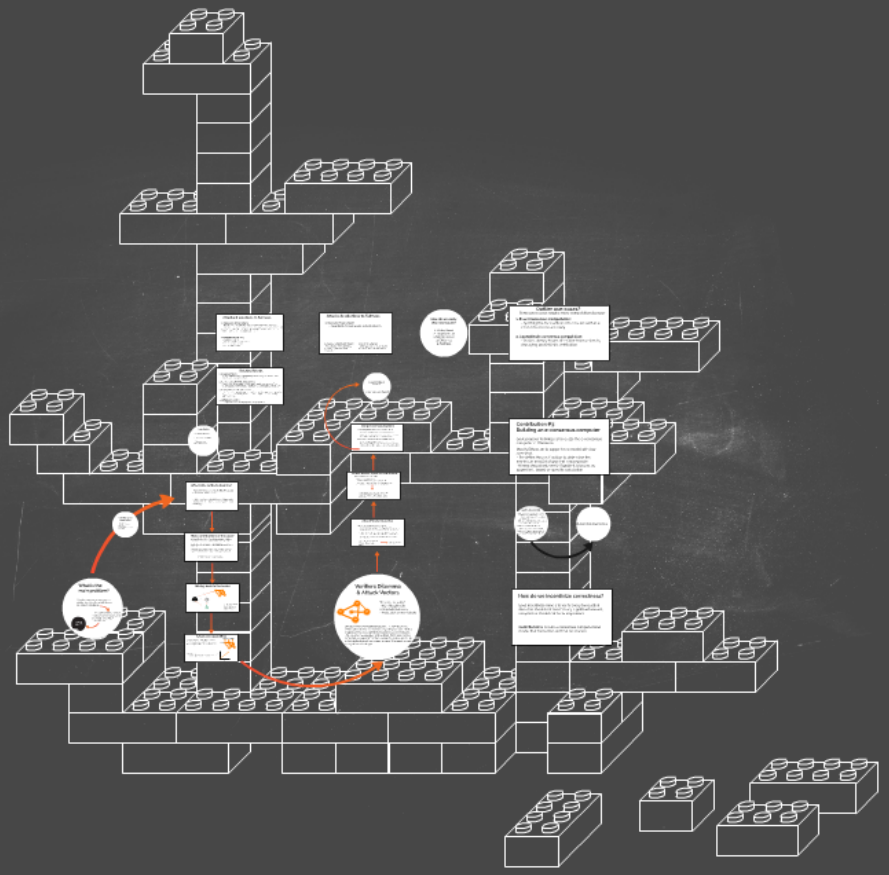
**Blockchain:**
**"Demystifying Incentives in the Consensus Computer"**

**By:**
Loi Luu, Jason Teutsch, Prateek Saxena, Raghav Kulkarni

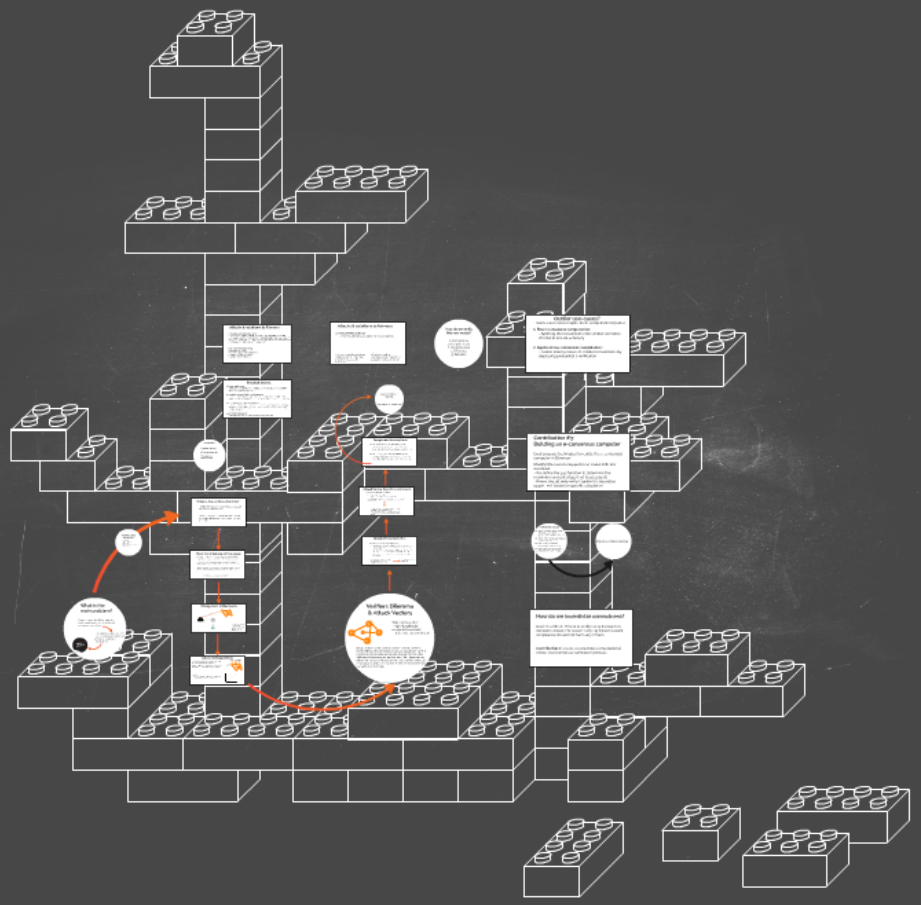
**Reviewed by:**
John Podboy



**Blockchain:**
**"Demystifying Incentives in the Consensus Computer"**

**By:**
Loi Luu, Jason Teutsch, Prateek Saxena, Raghav Kulkarni

**Reviewed by:**
John Podboy



# Blockchain: "Demystifying Incentives in the Consensus Computer"

**By:**

Loi Luu, Jason Teutsch, Prateek Saxena, Raghav Kulkarni

# Reviewed by:

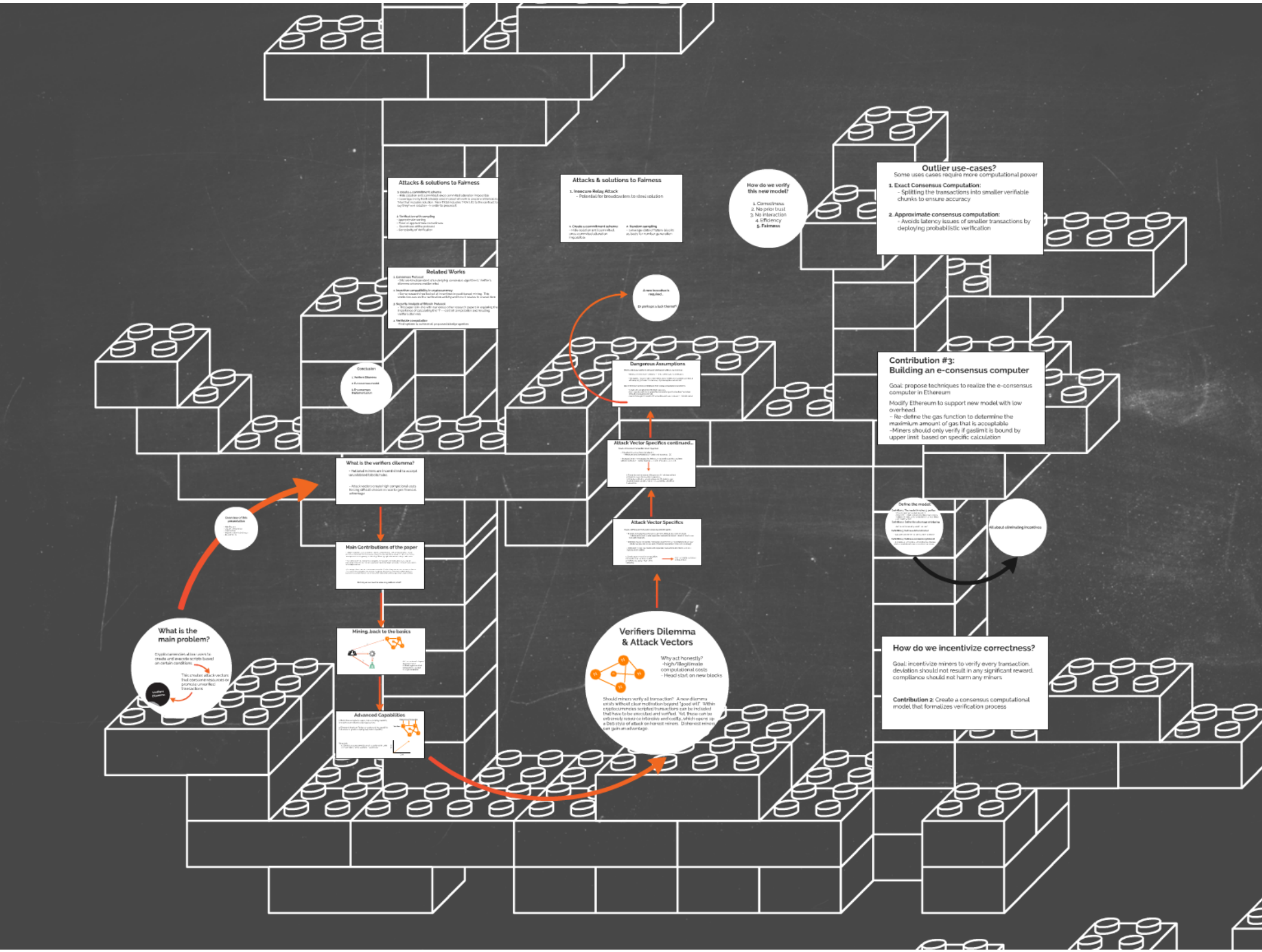
# "Demystifying Incentives in the Consensus Computer"

**By:**

Loi Luu, Jason Teutsch, Prateek Saxena, Raghav Kulkarni

**Reviewed by:**

John Podboy


Attacks & solutions to Fairness
Related Works
Attacks & solutions to Fairness
1. Insecure Relay Attack
- Potential for broadcasters to steal solution
How do we verify this new model?
1. Correctness
2. No prior trust
3. No interaction
4. Efficiency
5. Fairness
Outlier use-cases?
Some uses cases require more computational power
1. Exact Consensus Computation:
- Splitting the transactions into smaller verifiable chunks to ensure accuracy
2. Approximate consensus computation:
- Avoids latency issues of smaller transactions by deploying probabilistic verification
Dangerous Assumptions
Contribution #3:
Building an e-consensus computer
Goal: propose techniques to realize the e-consensus computer in Ethereum
Modify Ethereum to support new model with low overhead.
- Re-define the gas function to determine the maximum amount of gas that is acceptable
-Miners should only verify if gaslimit is bound by upper limit based on specific calculation
Attack Vector Specifics continued...
Attack Vector Specifics
Define the model
All about eliminating incentives
What is the verifiers dilemma?
Main Contributions of the paper
Mining..back to the basics
Advanced Capabilities
What is the main problem?
Verifiers Dilemma & Attack Vectors
Why act honestly?
-high/illegitimate computational costs
- Head start on new blocks
How do we incentivize correctness?
Goal: incentivize miners to verify every transaction. deviation should not result in any significant reward. compliance should not harm any miners
Contribution 2: Create a consensus computational model that formalizes verification process

# What is the main problem?

Cryptocurrencies allow users to create and execute scripts based on certain conditions.

This creates attack vectors that consume resources or promote unverified transactions

Verifiers Dilemma

# Overview of this presentation

- Main Problem
- Cryptocurrency Basics
- Contributions
- Implementation & Challenges
- Related Works

# What is the verifiers dilemma?

- Rational miners are incentivized to accept unvalidated blockchains.

- Attack vectors create high computional costs forcing difficult choices in race to gain financial advantage

# Main Contributions of the paper

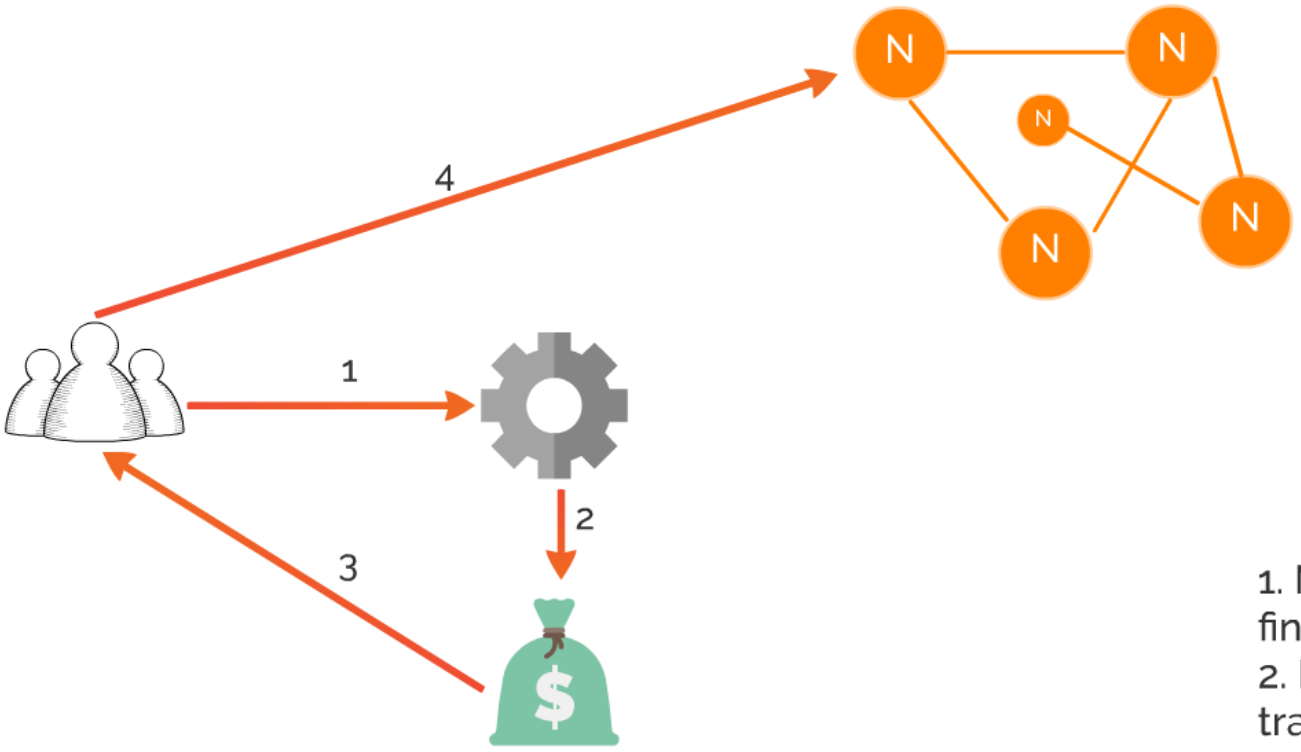
1. Verifier's dilemma & attack vectors. Miners are forced into a difficult choice where attack vectors can force verifiers to consume and waste computational resources. This can result in unscrupulous miners gaining an advantage **&** forcing legitimate miners to skip verifications.

2. Security model for a Consensus computer. Formulation and verification of an accurate consensus computer that creates appropriate incentive structures, reduces threats and address scalability concerns.

3. Techniques to realize an e-consensus computer. Practical implementation and verification of an e-consensus computer that promotes appropriate incentive structures within Ethereum ecosystem. Demonstrated ease of implementation while addressing certain attack vectors.

**But why do we need to solve any problem at all?**

# Mining..back to the basics

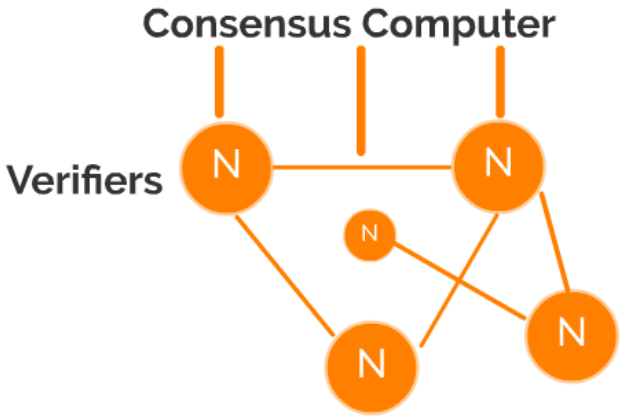



1. Miners mine with hope of financial reward
2. Miners agree on what transactions to accept through consensus

# Advanced Capabilities

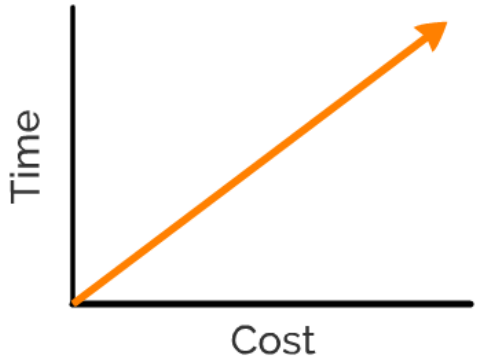
1. Blockchain ecosystems support some scripting capability to enable decentralized scripts/applications.

2. Ethereum introduces Turing-complete scripting capability that allows for greater scripting/application capability.

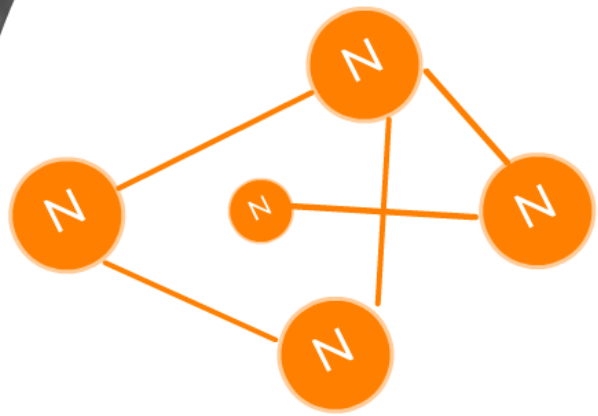



Miners job:

1. Check blocks are correctly constructed (proof of work)
2. Check validity of transactions in each block



# Verifiers Dilemma & Attack Vectors




Why act honestly?
-high/illegitimate computational costs
- Head start on new blocks

Should miners verify all transaction? A new dilemma exists without clear motivation beyond "good will". Within cryptocurrencies scripted transactions can be included that have to be executed and verified. Yet, these can be extremely resource intensive and costly...which opens up a DoS style of attack on honest miners. Dishonest miners can gain an advantage.

# Attack Vector Specifics

**Attack #1 Resource Exhaustion attack by problem givers**

- If miners honestly follow the protocol--they verify all blocks in the chain
  -if dishonest honers create expensive transactions honest miners will be forced to waste resources

- Ethereum has some control mechanism to prevent this--concept known as "Gas"
  -While Gas provides some level of control/prevention it does not do enough

- Dishonest miners can create/add expensive transactions into their own block that has to be verified

1. Create super expensive computation.
2. Execute script so it looks valid.
3. Verifiers eat up significant time validating.

1. Miners waste resources
2. Step ahead

# Attack Vector Specifics continued...

**Attack #2 Incorrect transaction attack by prover**

- This attack is a direct result of attack 1
  - Miners are afraid of wasted computational resources = $$

- Increased time ---> increases the likehood miners will accept transactions without verification. Puzzle Giver (G) -- (A x B). Include C---> C=AxB

1. Execution can include anything as result (malicious action)
2. Health & trust of ecosystem disappears
3. Skipping verification allows a jump start & yields longer chains in system of dishonest miners containing unverified transactions

# Dangerous Assumptions

**Miners will always perform adequate verification without any incentive**

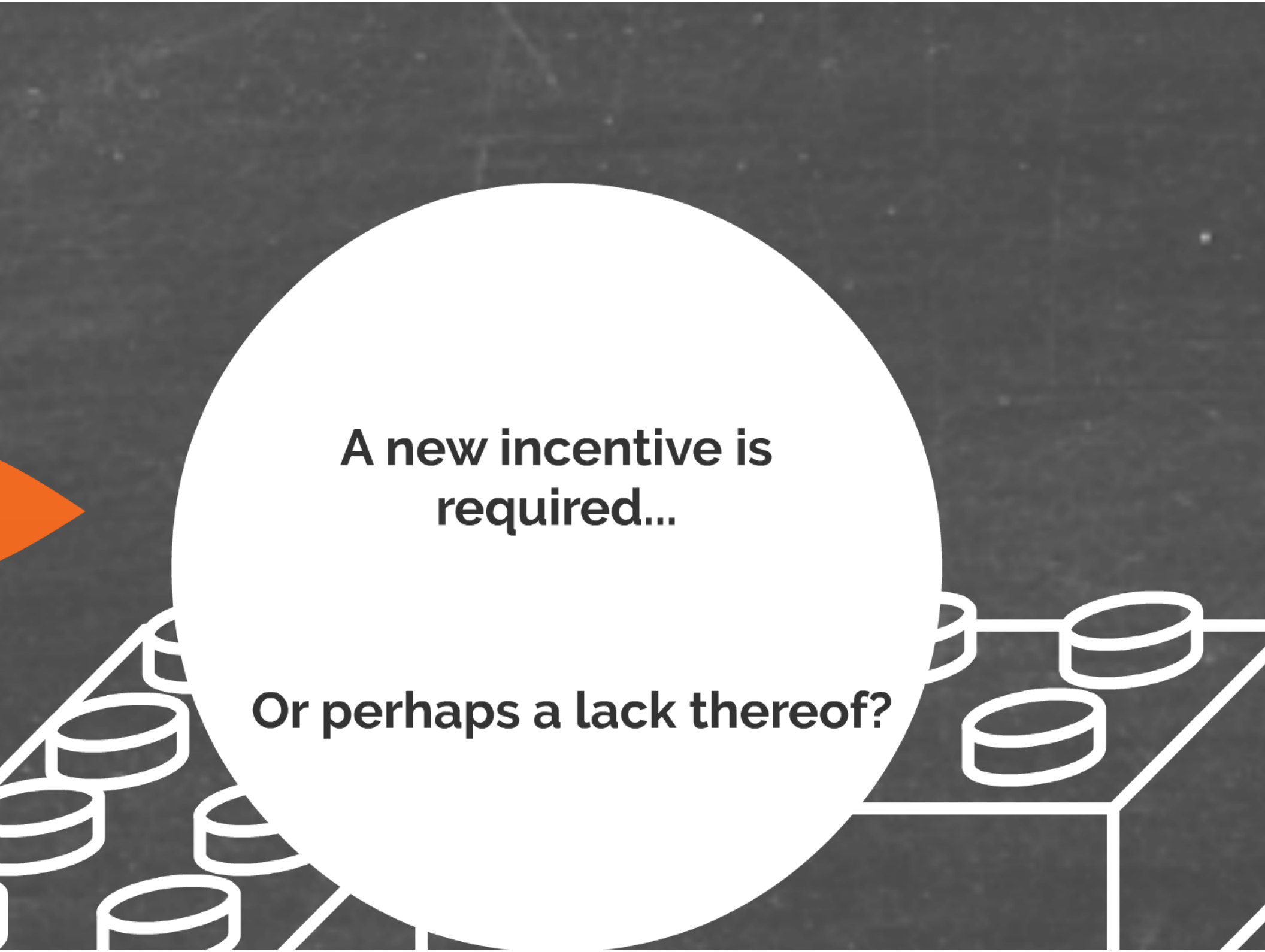
- Miners are ONLY incentivized to mine for currency & not verification
- Anticipated impact of dishonest miners and unverified transactions can reduce efficiency by 30% even if super majority of ecosystem are honest.

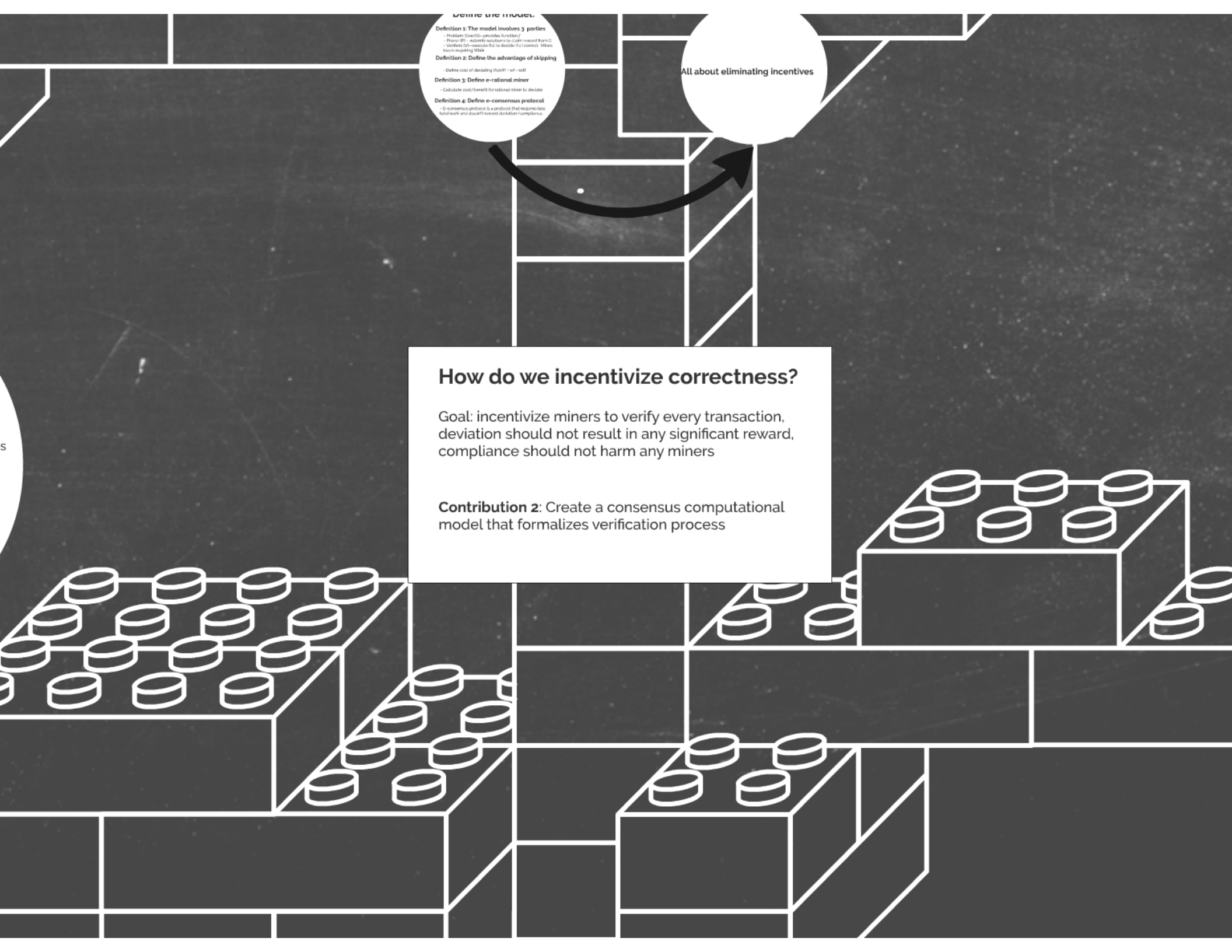
**Gas in Ethereum protects individuals from costly computational problems.**

- Scripts are compiled into Ethereum opcodes.
- Each opcode costs some predefined amount of gas (transactions fee) when executed/charged to Sender
- Gas isn't charged to creator of transaction and can circumvent intended desire

A new incentive is required...

r perhaps a lack thereof?


A new incentive is required...
Or perhaps a lack thereof?


Define the model:
Definition 1: The model involves 3 parties
Definition 2: Define the advantage of skipping
Definition 3: Define e-rational miner
Definition 4: Define e-consensus protocol
All about eliminating incentives
How do we incentivize correctness?
Goal: incentivize miners to verify every transaction, deviation should not result in any significant reward, compliance should not harm any miners
Contribution 2: Create a consensus computational model that formalizes verification process

# How do we incentivize correctness?

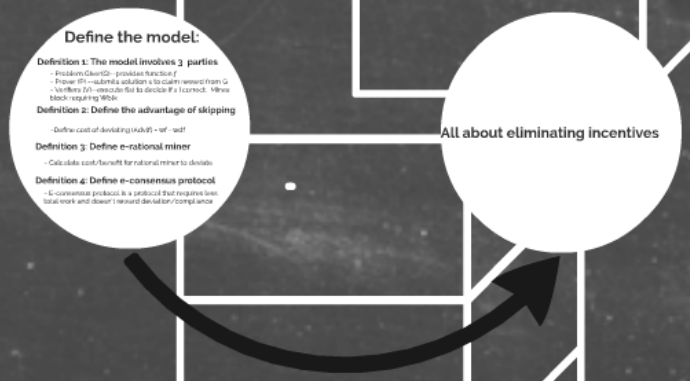
Goal: incentivize miners to verify every transaction, deviation should not result in any significant reward, compliance should not harm any miners

**Contribution 2**: Create a consensus computational model that formalizes verification process

## Building an e-consensus computer

Goal: propose techniques to realize the e-consensus computer in Ethereum

Modify Ethereum to support new model with low overhead.
~ Re-define the gas function to determine the maximium amount of gas that is acceptable
~Miners should only verify if gaslimit is bound by upper limit based on specific calculation




## How do we incentivize correctness?

Goal: incentivize miners to verify every transaction, deviation should not result in any significant reward, compliance should not harm any miners

**Contribution 2**: Create a consensus computational model that formalizes verification process

# Define the model:

## finition 1: The model involves 3 parti

~ Problem Giver(G)--provides function *f*
~ Prover (P) --submits solution s to claim reward
~ Verifiers (V)--execute f(s) to decide if *s* i correct.
block requiring Wblk

# Define the model:

## Definition 1: The model involves 3 parties

~ Problem Giver(G)--provides function *f*
~ Prover (P) --submits solution s to claim reward from G
~ Verifiers (V)--execute f(s) to decide if *s* i correct. Mines block requiring Wblk

## Definition 2: Define the advantage of skipping

~Define cost of deviating (Adv(f) = wf - wdf

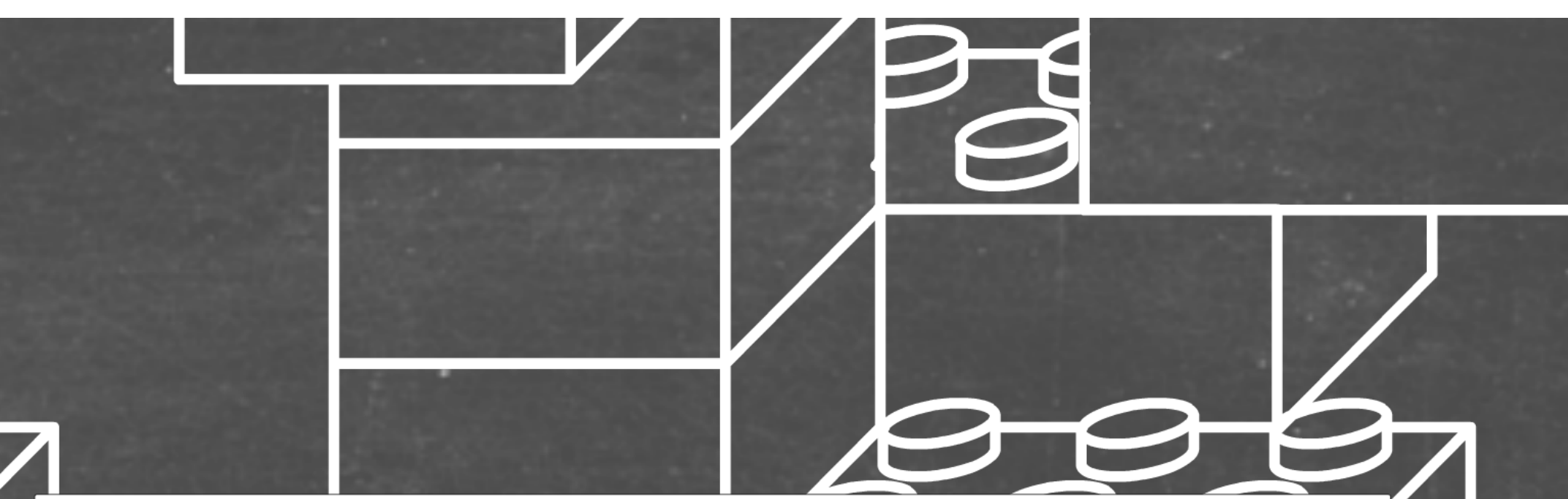
## Definition 3: Define e-rational miner

~ Calculate cost/benefit for rational miner to deviate

## Definition 4: Define e-consensus protocol

~ E-consensus protocol is a protocol that requires less total work and doesn't reward deviation/compliance

# All about eliminating incentives



# Contribution #3: Building an e-consensus computer

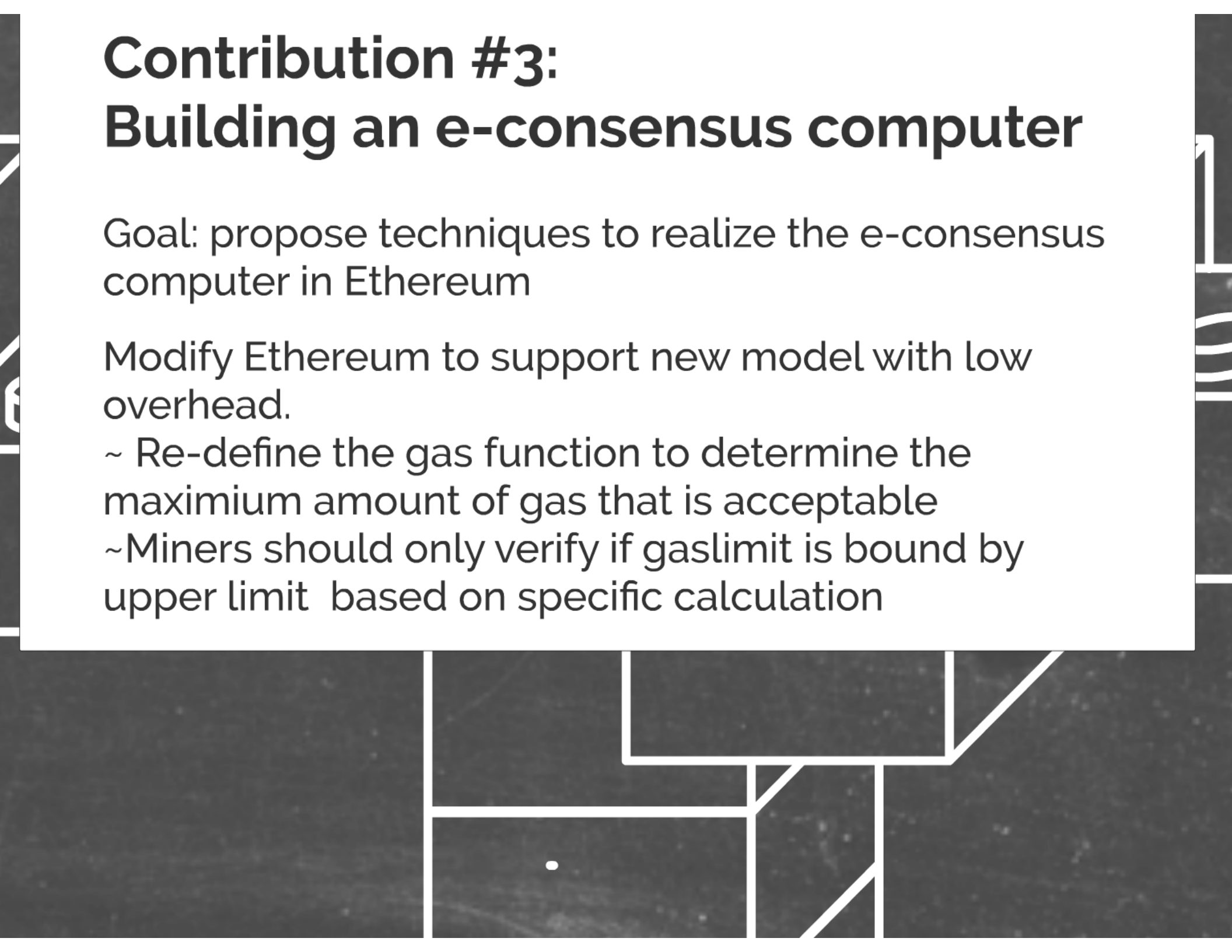
Goal: propose techniques to realize the e-consensus computer in Ethereum

Modify Ethereum to support new model with low overhead.
~ Re-define the gas function to determine the maximium amount of gas that is acceptable
~Miners should only verify if gaslimit is bound by upper limit, based on specific calculation

# Contribution #3: Building an e-consensus computer

Goal: propose techniques to realize the e-consensus computer in Ethereum

Modify Ethereum to support new model with low overhead.
~ Re-define the gas function to determine the maximium amount of gas that is acceptable
~Miners should only verify if gaslimit is bound by upper limit  based on specific calculation

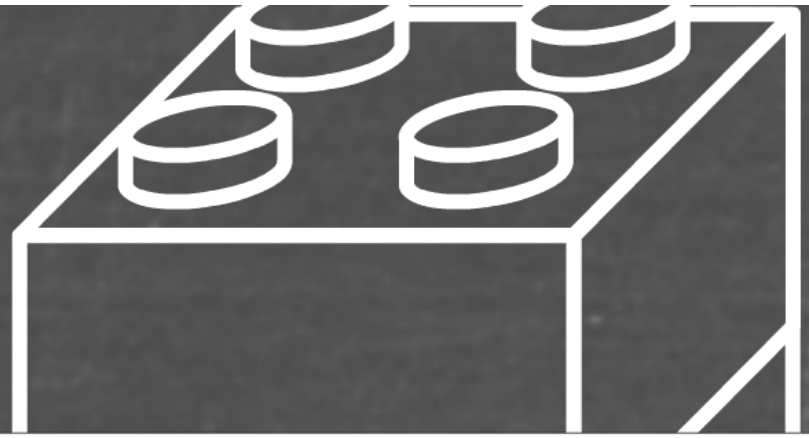


# Outlier use-cases?

Some uses cases require more computational power

1. **Exact Consensus Computation:**
   - Splitting the transactions into smaller verifiable chunks to ensure accuracy

2. **Approximate consensus computation:**
   - Avoids latency issues of smaller transactions by deploying probabilistic verification



# Outlier use-cases?

Some uses cases require more computational power

1. **Exact Consensus Computation:**
   - Splitting the transactions into smaller verifiable chunks to ensure accuracy

2. **Approximate consensus computation:**
   - Avoids latency issues of smaller transactions by deploying probabilistic verification

# Outlier use-cases?

Some uses cases require more computational power

1. **Exact Consensus Computation:**
   - Splitting the transactions into smaller verifiable chunks to ensure accuracy

2. **Approximate consensus computation:**
   - Avoids latency issues of smaller transactions by deploying probabilistic verification

# How do we verify this new model?

1. Correctness
2. No prior trust

1. Correctness
2. No prior trust
3. No interaction
4. Efficiency
5. **Fairness**

# Attacks & solutions to Fairness

**1. Insecure Relay Attack**

- Potential for broadcasters to steal solution

**1. Create a commitment scheme**

- Hide solution until commited, once commited alteration impossible

**2. Random sampling**

- Leverage data of future blocks as basis for number generation

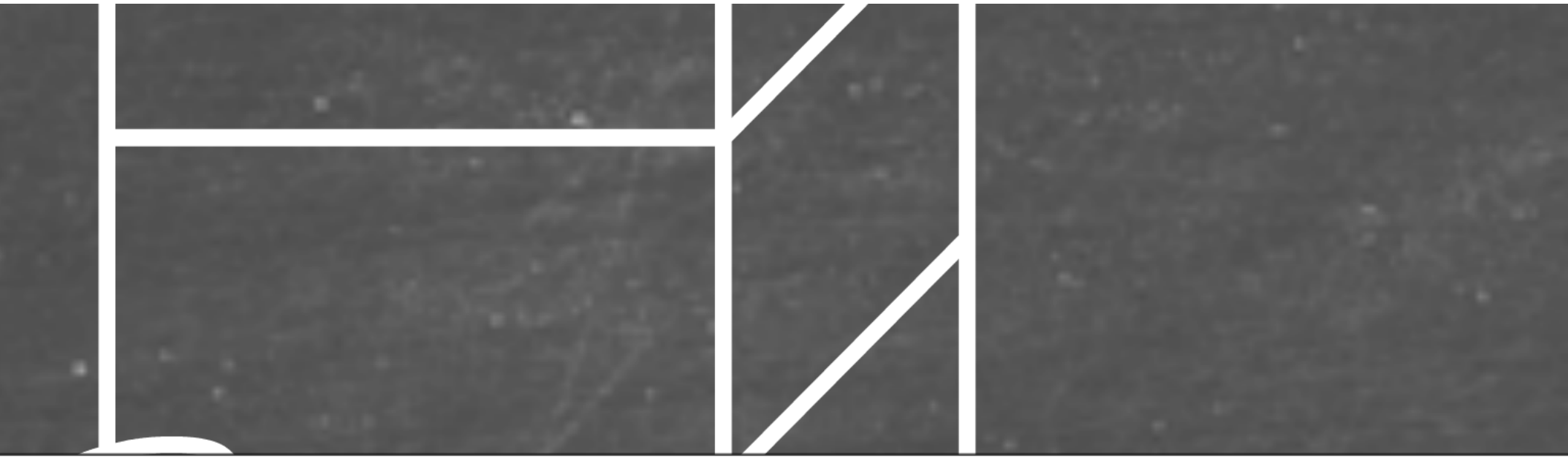
# Attacks & solutions to Fairness

**1. Create a commitment scheme**

- Hide solution until commited, once commited alteration impossible
- Leverage 1-way hash already used in proof of work to prepare a transaction Tx(a) that includes solution. New TX(b) includes TX(A)'s ID to the contract to say they have solution--in order to process it

**2. Verification with sampling**

- Approximate sorting
- Proof of approximate correctness
- Soundness of the protocol
- Complexity of Verification



# Related Works

**ol**

ndent of underlying consensus algorithm's.
matter what

**bility in cryptocurrency**

# Related Works

**1. Consensus Protocol**

- this work independent of underlying consensus algorithm's. Verifier's dilemma arises no matter what

**2. Incentive compatibility in cryptocurrency**

- Some research has looked at incentives in pool-based mining. This works focuses on the verification activity and how it relates to overall flow

**3. Security Analysis of Bitcoin Protocol**

- This paper is in-line with numerous other research papers in exploring the importance of calculating the "f" ---cost of computation and resulting verifiers dilemma

**4. Verifiable computation**

-First system to achieve all proposed ideal properties

# Conclusion

## 1. Verifiers Dilemma

1. Verifiers Dilemma

2. E-consensus model

3. E-consensus implementation

**Blockchain:**
**"Demystifying Incentives in the Consensus Computer"**

**By:**
Loi Luu, Jason Teutsch, Prateek Saxena, Raghav Kulkarni

**Reviewed by:**
John Podboy