Ethereum and MEV status: “It’s complicated”

Barnabé Monnot
Robust Incentives Group (RIG), Ethereum Foundation
Main themes

MEV is value — we shouldn’t be too quick to prevent its emergence!

But MEV is *mismatched* value — we should develop better mechanisms to channel it productively.
Validators as protocol operators
Who runs the network?

Ethereum requires **consensus** over state of the chain
This is done with **Proof-of-Stake-based mechanism**

**Validators are first-class protocol operators**
Responsible for **maintaining a single view of the ledger**
Produce blocks, are accountable for safety faults
How to become a validator

Lock up **32 ETH** in the deposit contract, wait for activation

- **Block proposals**: Containing consensus and execution data
- **Attestations**: Provide view of block tree, finalise blocks
Protocol specifies rewards and penalties:

- **Rewards**
  - Block reward for block proposal/correct voting
  - Transaction fees from execution payload

- **Penalties**: Inactivity penalty + Slashing
Protocol lets validators-as-block-producers consume resources. Supply constrained to guarantee low verification costs.

Validators produce blocks, meeting demand for transactions with supply of resources.
Ethereum protocol

Network of validators

- Rewards
- Gas
- Penalties
Ethereum protocol

Network of validators

- Rewards
- Gas
- Penalties

Block building
Towards minimum rent for validators
Validator privilege

Validators have a *privileged* position on the network.

**In this talk**

**MEV =** All revenue achievable by validator *from this position*

Includes “revenue achievable by re-ordering, inserting or censoring transactions”
Rent 1: Congestion pricing

Validators pack blocks, but block space is scarce ⇒ Users express inclusion preferences via fees

Monopoly without a monopolist (Huberman, Leshno, Moallemi, 2021) Operators cannot enforce monopoly pricing (Bitcoin-type TFMs)

Ethereum with EIP-1559 fee market (Roughgarden, 2021) Fees / Congestion costs are internalised by the protocol

Data point: ~6 billion USD captured and removed since EIP-1559 (Aug. ‘21)
Rent 2: Validator privilege

 Validators include user transactions in the blocks they make

 Last look ⇒ Validators capture value from externalities

称之为 Arbitrage

 User makes a swap order for token A against token B on a market 1
  ⇒ Creates price imbalance with another market 2

 Validator buys B low on 1 ⇒ Validator sells B high on 2
  ⇒ Price imbalance is resolved, Validator pockets the difference
Rent 2: Validator privilege

Validators include user transactions in the blocks they make

Last look ⇒ Validators extract value from users

“Sandwich” attack

User makes a swap order for token A against token B

Validator places: 1) Order for A/B before user swap

+ 2) Order for B/A after user swap

_validator buys low ⇒ _user buys high ⇒ _validator sells high

Permissionless [validators + programmability] ⇒ No “outlawing”

+ Sandwiches may create surplus! (Kulkarni, Diamandis, Chitra, 2022)
Rent 2: Validator privilege

Maximising extractable value for validator is hard
Requires sophistication and/or access to exclusive order flow

Division of labor: Validators source their blocks from builders
~ Procurement auction, builders extract value, bid it away

Future: Protocol is the auctioneer, permissionless auction
Bid values are captured and internalised ⇒ Minimum rent
Recovering max user welfare

Protocol captures validator rent, but user is still hurt 🥪😔🥪

Question: How to protect user, without hurting coordination?

Tensions
Permissionless programmability ⇒ Max coordination value
Defensive “protections” add constraints ⇒ May destroy value

Are we lost?
Recovering max user welfare

**Operator** may have last look, but **user** has commitment power!

**Examples**

- Order Flow Auctions (OFAs): User sells order to bidders
- Contextual execution
- ??? ⇒ Permissionless innovation in cryptography and mechanism design

This is the most exciting place to do research in!

[mevconometrics.wtf] ⇒ 7 hours of great content :)


Thank you!

Go further:

- [ethereum.github.io/rig](ethereum.github.io/rig)
- [barnabe.substack.com](barnabe.substack.com)
- [mevconomics.wtf](mevconomics.wtf)

Get in touch! [barnabe@ethereum.org](mailto:barnabe@ethereum.org)