by Conor McMenamin, Nethermind Research

This article is part 2 of a series (part 1 available here) on the economic viability of preconfirmations. The series has received funding from a LEGO grant. Many thanks to Lin and Michal from Nethermind, Fra and Lorenzo from Chainbound, and Bruno for their reviews and comments.

Abstract

We show that there exists a class of executionpreconfirmation protocols called dependendent sub-slot auctions

(DSSAs) that increase a proposer's expected revenue compared to the expected revenue from runningMEV-Boost, a block-building protocol standard used by over 90% of Ethereum proposers. In DSSAs, each preconfirmation is made dependent on future expected block and preconfirmation value. Specifically, the builder/proposer preconfirming the N^{th}

transaction in a slot is the same person who benefits from the expected revenue of all preconfirmations and block-building revenue after the N^{th}

transaction in the same slot. We show the expected proposer revenue through DSSAs is greater than or equal to the expected revenue from just running end-of-slot MEV-Boost, regardless of any preconfirmation tips.

Organisation of the Paper

We begin by introducing all of the background subject matter needed to reason about the preconfirmation protocols we introduce and the results we present. After this, we introduce a set of preconfirmation classifications. We then analyze the expected proposer revenue of these classifications. We conclude with a discussion on the consequences, limitations and future directions that can be taken from our work.

Preliminaries

This section introduces existing terminology and techniques that are used throughout the article.

Ethereum

The results in this paper are provided in the context of Ethereum, but can be extended to any blockchain progressing in discrete time periods with a deterministic leader election mechanism, and the leader known ahead of time. Specifically, we consider Ethereum as progressing in time periods of 12s, known as slots, with each slot having a single designated proposer elected at some point before the slot begins.

Preconfirmations

This article is interested in protocols where proposers, whether they be L1 or L2 proposers, confirm transactions for users throughout the sub-slots. Consensus on transactions is only initiated at the end of a slot, so confirmations provided at sub-slots before the end of the slot have a weaker guarantee of being finalized. For this reason, and because "confirmation" is a term typically associated with classical consensus, these sub-slot confirmations where consensus is not taking place will hereafter be referred to as preconfirmations.

To reason about preconfirmations, we can consider each Ethereum slot as being splitable into sub-slots. Without loss of generality, when we consider a slot with N

sub-slots, we assume each sub-slot is of equal duration, i.e. 12/N

seconds. In this framework, Ethereum slots correspond to there being 1 sub-slot.

Given a number of sub-slots N

, we are interested in the blockchain state at the end of each sub-slot. We let $S_{i}, \ 0 \leq i \leq n$

signify the state at the end of each sub-slot, with S_0

being the state at the beginning of the first sub-slot, equivalent to the state at the beginning of the entire slot.

To properly define the preconfirmation protocols we are interested in, we also need to introduce the concept of an active proposer

at each sub-slot, denoted by $P_i, \ 0 \leq i \leq n. \ P_0$

is the proposer as defined within the Ethereum protocol. Generally, P_i

is the entity who owns the right to propose the remainder of the Ethereum block at sub-slot i

. If $P_i \neq P_0$

, we assume that this right to propose a block has been transferred to P_i

for sub-slot i

in a secure manner. This means that during sub-slot i

, if P_i

proposes a (sub-)block, this block will be considered valid by the Ethereum network equivalent to P_0

proposing a block in the conventional Ethereum network. More than this, if $P_i \neq P_{j}$, $i<j$

, and P_i

does not propose a block before sub-slot $i+1$

begins, P_i

cannot propose a (sub-)block in sub-slot j

.

The exact mechanism for transferring proposal rights and enforcing no out-of-sub-slot proposals is omitted. Such a mechanism may leverage one or all of collateralized commitments, a reputation-based system, multi-party computation, threshold encryption, or trusted execution environments.

Lastly, we need to introduce the concept of reachable states.

Given some starting state S_{i-1}

for sub-slot i

, the set of reachable states for sub-slot i

describes the set of states that builders can transition to during sub-slot i

. This set of reachable states is the result of all combinations of available transactions in the mempool, as well as all possible actions builders can take given those transactions during slot i

.

Utility and Expectation

With respect to our framing of active proposers, we need to distinguish between slot revenue and sub-slot revenue. When P_0

is considering whether or not to offer preconfirmations, or which preconfirmation protocol to use, P_0

is only interested in maximizing their own expected slot revenue, that is, the total expected revenue from their slot. Slot revenue of P_0

will be crucial when comparing protocols. To reason about slot revenue, we will need to consider sub-slot revenue too. This is the revenue the active proposer receives in a particular sub-slot. It should be noted that P_0

's slot revenue is not necessarily equal to the sum of sub-slot revenues throughout the slot. This is because some preconfirmation protocols involve changing the active proposer. The function $E(x)$

is used to represent the expected value of some variable x

.

MEV-Boost, and Auctions

Much of our analysis is done in the context of MEV-Boost. MEV-Boost is a proposer-builder separation protocol which allow a proposer to outsource block-building among a sophisticated set of builders. These builders compete to be selected to build the Ethereum block through an auction. This auction is conducted by a relayer, with the relayer trusted not to reveal builder payloads before the auction concludes, as well as to publish all builder bids to the proposer, and release the payload of the winning builder when the bid is selected by the proposer. Over 90% of blocks on Ethereum are sourced through MEV-Boost so we see this as a relevant standard against which to compare the preconfirmation classifications we introduce in this article. MEV-Boost is seen as a way for proposers to maximize their revenue from the proposer role due to the valuable

nature of block-building and the competition induced through the auction.

Generally, we assume a set of rational proposers and rational homogeneous builders (all have the same capabilities) builders, as well as a trusted auctioneer, in-line with the assumptions of MEV-Boost. For any number of sub-slots N>0

considered in this paper, at any sub-slot i

, builders are able to observe S_{i-1}

and accurately price the value of their bid for the sub-slot auction at slot i

, before needing to submit bids to the auction for sub-slot i

. Any auctions we consider are first-price unsealed bid, orEnglish Auctions as they are commonly known (e.g. MEV-Boost). We also assume that the auctioneer can submit a bid into their own auction.

Classifying Preconfirmation Protocols

This section provides a classification of preconfirmation protocols that allow for clear analysis and comparison. Although the classifications presented may not be complete, we believe the classifications we provide cover the majority of protocols being discussed and developed. We look forward to additional classifications emerging, and extending our analysis to incorporate them, if/when they arise.

Inclusion vs Execution Preconfirmations

The distinction between inclusion preconfirmations and execution preconfirmations is perhaps one of the more fundamental distinctions of preconfirmations. This article is primarily focused on execution preconfirmations, although we touch on inclusion preconfirmations in the conclusion section, as well as providing a small result on their revenue in the accompanying Appendix.

Execution preconfirmations guarantee a specific execution of the transaction. Although execution preconfirmations are likely to be provided by committing to the exact sequence of transactions on which they depend (thus guaranteeing the execution), more exotic/elaborate forms of execution preconfirmations may emerge. Without loss of generality, unless otherwise stated, for the remainder of the article preconfirmations should be taken to mean execution preconfirmations.

Dependent vs Independent Preconfirmations

Towards outlining a broad class of preconfirmation protocols that is economically viable we make the distinction between what we refer to as dependent and independent execution preconfirmations.

- Dependent preconfirmations are offered with the goal of maximizing the revenue of the active proposer for the entire slot.
- Independent preconfirmations maximize the active proposer's revenue in the upcoming sub-slot, but not necessarily for the entire slot.

To illustrate the differences between the two classes, we will define both with respect to sub-slots and sub-slot auctions held to provide preconfirmations at each of the sub-slots.

Independent preconfirmations, or independent sub-slot auctions (ISSAs), are defined as follows:

1. For each sub-slot i \in [1,…, n], \ P_0

auctions off the right to build a sub-block on the state S_{i-1}

.

1. After the auction for sub-slot n

is completed, P_0

concatenates the sub-blocks from each sub-slot in order to form a block. This block is then proposed to the network.

Dependent preconfirmations, or dependent sub-slot auctions (DSSAs), are defined as follows:

1. For each sub-slot i \in [1,…, n], P_{i-1}

auctions off the single right to both:

a. build a sub-block on the state S_{i-1}.

b. become P_{i}

.

1. After the auction for sub-slot n is completed, P_0

concatenates the sub-blocks from each sub-slot in order to form a block. This block is then proposed to the network.

Notice that the only difference between DSSAs and ISSAs is the transfer of active proposer rights that occurs in DSSAs at every-sub-slot. With this subtle difference, the auction winner of DSSA sub-slot k

is not only paying for the right to build the sub-block for slot k

, but also for the right to receive the revenue from the DSSA at sub-slot k+1

.

[

ISSA vs DSSA Flow

879×849 98 KB

](https://europe1.discourse-cdn.com/flex013/uploads/lido/original/2X/b/b023f9548968df9ac3fb711876368a39763b51f3.jpeg)

An example of how ISSAs and DSSAs progress. Note that DSSAs auction off the sub-block building rights for some sub-slot i and the active proposer rights for sub-slot i+1.

In a previous article in this series we presented simulation-based results on the expected proposer revenue from running ISSAs. The results presented in that article indicate a large expected drop off in proposer revenue versus just running MEV-Boost through lost MEV revenue. That being said, it is still possible that ISSAs emerge as a popular preconfirmation protocol, replacing and potentially even improving on this lost revenue through preconfirmation tips.

In the Analysis section of this article, we demonstrate that even without preconfirmation tips, DSSAs increase the expected proposer revenue versus just running MEV-Boost.

Implementation

To help with intuition, we briefly describe examples of these sub-slot auctions, and/or how to implement them in the wild. A sub-slot auction among sub-block builders would require a mutually trusted third party like the relayer in MEV-Boost.

mev-boost has the role of the mutually trusted relay

. Anyone can be a relay, and they will compete on reputation and service to both builders and validators. While this is a strict improvement to the trust model compared to Stage 1 PBS, relays can still be a risk to both builders and validators.

Alternatively, sub-slot auctions can be run by the sub-slot proposer themselves, e.g.priority gas auctions. This however may raise issues related to fair exchange that mutually trusted relays circumvent.

Dependent Sub-Slot Auctions

DSSAs are a rather theoretical concept. To run a DSSA, as we have defined it, the presence of a relayer or gateway who is trusted by the majority of competitive builders and the proposer to credibly run the auction for each active proposer or transfer proposer rights is a requirement. Replicating the trust of a relayer/gateway in a decentralized manner would require complex crypto-economic machinery, such as through an ideal implementation of SUAVE.

A more realistic implementation of a DSSA would be for the proposer to delegate active proposer rights to some affiliated builder, and for that builder to be the only participant in the DSSA. This is not an ISSA as the assumption of a builder effectively owning proposal rights will mean sub-blocks are conditioned on maximizing the revenue of the entire slot.

A nice property of this single-delegated builder is that the builder will always bid truthfully to themselves (proof omitted), so the choice of sub-block at any given sub-block is always chosen with the intention of maximizing the revenue of the entire slot. This revenue would be increased in expectancy by allowing many builders to participate in the auction. However, this would require the ability to transfer active proposer rights, which is non-trivial.

Independent Sub-Slot Auctions

ISSAs are relatively straightforward to implement. Each sub-slot, some relayer(s) run a MEV-Boost style auction to build the sub-block for that slot. This maps almost directly to the infrastructure and trust assumptions of normal MEV-Boost. Multi-round MEV Boost is an example of an ISSA.

Analysis

This section provides a series of results regarding the expected proposer revenue from running DSSAs. We prove that within our model and classifications, the expected proposer revenue from running DSSAs is optimal. We provide the results and their proofs here for completeness. Feel free to skip to the proceeding section for a less formal discussion about our results.

Describing the Revenue of Sub-Slot Auctions

To help describe the revenue from sub-slot auctions, let's introduce two variables V_i

and W_{i}

.

First, we define V_i

as the realised value of the winning builder's block from running a sub-slot auction at sub-slot i

. We differentiate between the winning bid and the realized value of the sub-block, as when i<n

, at least in the case of DSSAs, bidders realize some value from the sub-block through e.g. fees, MEV, but also buy the right to be the next active proposer.

When i=n

, the realised value of the sub-block n

equals the bid value. As such, V_n

can be thought of as the value of the winning bid in a MEV-Boost auction, with the caveat that the inital transactions in all of these MEV-Boost blocks must be the sequence resulting from concatenating the sub-blocks from sub-slot 1

to n-1.

Given V_i

, we define W_i

recursively for i \in [1, … n]

as:

- For i=n

, let W_n=V_n

, the value from running a MEV-Boost auction at sub-slot n

on S_{n-1}

.

- For 1 ≤ i <n

, let W_i= V_i + E(W_{i+1})

, where V_i

is the realised value from the sub-block winning the auction at sub-slot i

on S_{i-1}

, and E(W_{i+1})

is the expected value of W_{i+1}

given the state resulting from applying the winning sub-block from sub-slot i.

In the context of a specific auction, we can alternatively write W_i= \sum_{k=i}^n E(V_k)

, that is, the expected value of the realised builders' values in each sub-slot from i

to n

.

Although we believe thinking about W_i

as the sum of some realizeable value V_i

and the revenue of proceeding auctions W_{i+1}

makes for a more intuitive definition (especially in the context of DSSAs), thinking only in terms of expected realizeable values, the V_i

s, is also fine.

To help with the understanding of this these variables, we provide the following example:

Example:

Consider a set of two transactions [tx_1, tx_2]

as the only transactions that will exist during a slot, which has two sub-slots. Let tx_1

pay a tip of 2 gwei if preconfirmed in sub-slot 1, 0 if preconfirmed in sub-slot 2. Contrarily, let tx_2

have a value of 1 gwei if preconfirmed in sub-slot 1, and an expected value of 3 gwei if preconfirmed in sub-slot 2 (e.g. tx_2

could be a CEX-DEX arbitrage tx where only 1 CEX-DEX transaction is allowed per full slot).

- For an ISSA, V_1=4

and E(W_2)=0

, as the winning builder in sub-slot 1 will include both transactions in sub-slot 1, leaving no additional value to be captured in sub-slot 2.

- For a DSSA, preconfirming tx_1

during sub-slot 1, and delaying confirmation of tx_2

until sub-slot 2 results in a V_1 = 3, E(W_2)=3

, meaning W_1=6

. This represents the value of winning the auction given only tx_1

is preconfirmed during sub-slot 1 for a present value of 3 gwei, as the winning builder expects to capture a further 3 gwei in sub-slot 2.

With these variables hand, we now state a result about the revenue of DSSAs.

Theorem 1

At any sub-slot i<n

, consider the set of all reachable states at sub-slot i

as X_i

. Let there be some S'_i \in X_i

resulting in V'_i

such that W'*i= V'_i + E(W*{i+1}|S'_i)

is maximized over X_i

. A DSSA at sub-slot i

produces S'_i

, with expected revenue of at least W’_i

for P_{i-1}

.

Proof

: We know that in a DSSA, the winning builder producing some state S_i

corresponding to W_i=V_i+E(W_{i+1}|S_i)

realises V_i

for themselves and expects to capture E(W_{i+1}| S_i)

in the proceeding sub-slot auction. As any builder can produce S’_i

, this means all builders value winning the auction at sub-slot i

at W'*i= V'_i + E(W*{i+1}|S'_i)

. By definition, as W'_i

is the maximal W_i

achievable during sub-slot i

, builders are forced to produce S'_i

. By the revenue equivalence theorem the winning bid is at least W'_i.

With this theorem, we effectively have it that DSSAs maximize a proposer’s expected revenue given proposer’s are willing to participate in preconfirmations through sub-slot auctions. The next sub-section will present similar results of DSSAs maximizing revenue in the context of normal MEV-Boost and ISSA expected revenues.

[

Revenue progression in ISSAs vs DSSAs

1475×743 103 KB

](https://europe1.discourse-cdn.com/flex013/uploads/lido/original/2X/0/0a1e9b3caa5725b88cf3a25642d0095e72f8e420.jpeg)

A simplified representation of how the realised value and the max expected value of a slot change in ISSAs and DSSAs. The graph is normalized to keep the “max value attainable” point constant. In reality, this will change throughout the slot as the mempool and external information changes.

Comparing Sub-Slot Auction Revenue to MEV-Boost Revenue

Towards demonstrating that DSSAs increase the expected revenue for the Ethereum proposer versus just running MEV-Boost, we start off with some preliminary results with respect to the terminology we’ve introduced so far.

Corollary 2

If no sub-blocks are built, from the perspective of the proposer E(V_i)=0

and S_i=S_0

for all i

.

Proof:

This follows from the definitions of V_i

and S_i

.

- V_i =0

: As no sub-blocks are built, there is no realised revenue for the sub-slot.

- S_i=S_0

: As no sub-blocks are built, the starting state at each sub-block is the same as the starting state at the beginning of the slot, namely S_0

.

Corollary 3

If no sub-blocks are built, from the perspective of the Ethereum proposer P_0

, E(W_i)=E(W_n)

for all i

.

Lemma 4

Let E_i()

be the expected value function at the start of slot i

. For every S_{i-1}, i ≥ 1

, and for every builder participating in the DSSAs, there exists at least one set of transactions Tx_i

such that given:

- S'_{i}

is the state resulting from executing Tx_i

on S_{i-1}, W'_{i+1}

corresponding to S'_{i}

.

- S''_{i}

is the state resulting from executing nothing on S_{i-1}. \ W''_{i+1}

corresponding to S''_{i}

.

then: E_i (W'*{i+1}) ≥ E_i (W''*{i+1})

.

Proof:

Trivial as Tx_i

can be the null set.

Lemma 5

The expected revenue for an Ethereum proposer P_0

running a DSSA is greater than or equal to the expected revenue from running MEV-Boost without preconfirmations.

Proof:

Applying no transactions in each sub-slot and then running MEV-Boost at the end of the slot is equivalent to running MEV-Boost without preconfirmations from Corollary 2. From Lemma 1, this has less than or equal expected revenue to running the DSSA at every sub-slot.

Towards the main result regarding DSSAs, we introduce the concept of a mixed-strategy sub-slot auction (MSSA)

to cover any auction run over any number of sub-slots, where at each sub-slot the proposer can either:

1. run an independent sub-slot auction.
2. run a dependendent sub-slot auction.

3. wait to run MEV-Boost at the end of the slot.

Toward the optimality of DSSAs among considered execution preconfirmation strategies, we now provide weak bounds on DSSAs being dominant.

Lemma 6

At any sub-slot, the expected revenue for the active proposer from running a DSSA is greater than or equal to the revenue from running an MSSA.

Proof:

To get this result, we will compare the expected revenue from running a DSSA vs each of the 3 options in an MSSA, and show that the revenue from a DSSA is greater than or equal to the revenue from running an MSSA.

1. Building an independent sub-block at each sub-slot is one of the possible strategies in a DSSA. As such, the revenue from an ISSA is less than or equal to the revenue of a DSSA.
2. Trivial, as both options are DSSAs.
3. Building an empty sub-block and self-bidding to win the DSSA is equivalent to an active proposer doing nothing. As this is a possible DSSA strategy, the revenue from doing nothing is less than or equal to the revenue of running a DSSA.

The result follows.

Finally, we prove that running a DSSA is a dominant sub-slot auction strategy for an Ethereum proposer seeking to maximize their block building revenue.

Theorem 7

For an Ethereum proposer P_0

running an MSSA over N>0

sub-slots, the DSSA run over N

sub-slots has greater than or equal to expected revenue than the MSSA.

Proof:

Follows from Lemma 3 and induction over N sub-slots.

Conclusion

The most important result from the previous section is that there exists a classification of execution preconfirmation protocol that improves on the expected revenue for proposers compared to the current MEV-Boost block proposal paradigm. This results is independent of any additional preconfirmation tips that may be paid. This definitively proves that given any demand for execution preconfirmations expressed through the willingness to pay tips for preconfirmations, there exists profitable execution preconfirmation protocols.

Improving on Revenue Bounds

All of the bounds we formally derive are weak bounds, with expected preconfirmation revenues only greater than or equal to expected MEV-Boost revenues. Preconfirmations provide a considerably larger action space to proposers and as well as potential for increased tips due to the vastly improved user experience that sub-slot confirmations provide. We can incorporate the latter as an assumption, e.g. “all transactions pay some additional tip to be preconfirmed”. With this we could prove strict increases in expected proposer revenue. These results would require careful treatment of expected transaction sets that would only add marginally to the strength of our results. We omit a direct comparison between inclusion and execution preconfirmation protocols, as we expect the demand for inclusion and execution preconfirmations to be rather independent. This “independence” comes from the fact that inclusion preconfirmations are agnostic to their execution. Once we see both protocol classes emerge, we hope to revisit such analysis with a better understanding of any relationship between these classes of preconfirmations. In the meantime, we encourage industry and academia to pursue such results.

Effect of Preconfirmation Tips on Optimality Results

Interestingly, the presence of fees may affect the dominant strategy nature of DSSAs compared to MSSAs[Earlier in this series we demonstrated that ISSAs without fees are expected to reduce proposer revenue] vs. MEV-Boost. However, if ISSAs are the only acceptable preconfirmation solution from a user perspective due to trust and/or centralization concerns

regarding DSSAs, ISSAs may in fact emerge as the optimal block-building strategy. At the very least the results presented in this article enable us as a community to confidently build towards a preconfirmation future.

Preconfirmation Risks

Benefits aside, caution must be exercised when putting preconfirmation protocols into the block-building pipeline. All of the classifications outlined in this article, which we believe represent a significant subset set of preconfirmation protocols, lead to increased sophistication, capital, and risk-tolerance requirements from proposers, as well as additional trust and complexity requirements from users, relayers and builders. In other words, the risk of centralization, latency wars, monopolies and single points of failure all increase. Work must be done to identify and analyze these risks in the context of preconfirmations. We at Nethermind are undertaking this work with the joint support of Lido and the PBS Foundation, but we encourage the community to take part in this research too.

Appendix

Inclusion Preconfirmations

Inclusion preconfirmations guarantee that a transaction will be included somewhere in the block produced at the end of the slot. However, inclusion preconfirmations say nothing about where in the block they will be executed, what their output will be, or even that they don't revert. In this sense, inclusion preconfirmations are best suited to uncontentious state updates which can be order agnostic, e.g. standalone token transfers, rollup batches.

Example: MEV Boost + inclusion preconfirmations.

In this variation of MEV-Boost, proposers can choose to offer inclusion preconfirmations throughout the sub-slots they control. At the end of the slot, the proposer runs a MEV-Boost auction with the added restriction that builders must adhere to these inclusion preconfirmations for blocks to be valid in the MEV-Boost auction. Chainbound's Bolt V1 can be leveraged to provide these inclusion preconfirmations. This can then be coupled with something like commit-boost to run MEV-Boost with any inclusion preconfirmations provided as additional constraints on blocks to be considered valid.

Analysis

We start our analysis by providing a straightforward yet important lower bound on the expected proposer revenue from MEV-Boost + inclusion preconfirmations.

Lemma

The expected proposer revenue from running MEV-Boost + inclusion preconfirmations is greater than or equal to the expected revenue from running MEV-Boost.

Proof:

A proposer has two options:

1. Offer no inclusion preconfirmations: This is equivalent to running MEV-Boost, which has revenue equal to MEV-Boost.
2. Offer 1 or more inclusion preconfirmations: Any decision by the proposer to offer an inclusion preconfirmation must increase, although not necessarily strictly increase, the expected revenue of the proposer compared to not offering inclusion preconfirmations.

The result follows.

Implementing ISSAs and DSSAs

This paper doesn't explicitly define the implementation of the preconfirmation protocols we introduce. We aim to generalize these classes as much as possible, although formally covering all possible protocols is outside the scope of this paper. For each class there are many possible implementations:

1. Dependent preconfirmations:
2. Proposer provides preconfirmations themselves.
3. Listens to the mempool, and selectively preconfirms transactions
4. Commits to running an end-of-slot auction among a set of gateways, with these gateways providing preconfirmations. The preconfirmations from any one gateway are thus probabilistic, with a preconfirmation in this setting only guaranteed to be confirmed when all gateways preconfirm the transaction and/or one of the gateways who preconfirmed the transaction wins the end-of-slot auction. An example of this form of preconfirmation protocol is mev-

commit from primev.

5. Listens to the mempool, and selectively preconfirms transactions
6. Commits to running an end-of-slot auction among a set of gateways, with these gateways providing preconfirmations. The preconfirmations from any one gateway are thus probabilistic, with a preconfirmation in this setting only guaranteed to be confirmed when all gateways preconfirm the transaction and/or one of the gateways who preconfirmed the transaction wins the end-of-slot auction. An example of this form of preconfirmation protocol is mev-commit from primev.
7. Proposer outsources preconfirmation rights to a trusted gateway.
8. Proposer provides preconfirmations themselves.
9. Listens to the mempool, and selectively preconfirms transactions
10. Commits to running an end-of-slot auction among a set of gateways, with these gateways providing preconfirmations. The preconfirmations from any one gateway are thus probabilistic, with a preconfirmation in this setting only guaranteed to be confirmed when all gateways preconfirm the transaction and/or one of the gateways who preconfirmed the transaction wins the end-of-slot auction. An example of this form of preconfirmation protocol is mev-commit from primev.
11. Listens to the mempool, and selectively preconfirms transactions
12. Commits to running an end-of-slot auction among a set of gateways, with these gateways providing preconfirmations. The preconfirmations from any one gateway are thus probabilistic, with a preconfirmation in this setting only guaranteed to be confirmed when all gateways preconfirm the transaction and/or one of the gateways who preconfirmed the transaction wins the end-of-slot auction. An example of this form of preconfirmation protocol is mev-commit from primev.
13. Proposer outsources preconfirmation rights to a trusted gateway.
14. Independent preconfirmations:
15. Independent sub-slot auctions run by the proposer among a set of builders. This can be anything from periodically auctioning off fixed size sub-blocks a là MEV-Boost, or periodically running a priority gas auction.
16. Independent sub-slot auctions run by the proposer among a set of builders. This can be anything from periodically auctioning off fixed size sub-blocks a là MEV-Boost, or periodically running a priority gas auction.

Each of these protocols have many possible trust assumptions too. For this phase of research, a comparison of the trust assumptions is beyond the scope. We merely aim to exhibit that some implementable preconfirmation protocols are clearly economically viable for proposers, while others require further analysis.