Rollups are a fascinating primitive. They are

set to become Ethereum's preferred way of scaling moving forward and offer a wide design space for their operations. On all counts, rollups

extend

the protocol they build on, preserving most of its properties. All that matters is to guarantee

correctness

of the off-chain execution, as well as

availability

of the data behind the execution. How to achieve either is up to the designer.

Lately I've become interested in understanding rollups from an economic perspective. This is not just a theoretical concern. Fees on the base layer (L1) are high and we should aggressively work on providing spaces where users can transact at an affordable rate. Better economics means

better pricing means

happier users.

So how should we think about

rollup

economics specifically? We'll first identify the parties interacting in the rollup system, their roles and responsibilities. Once done, we tease out various costs, revenues and fees irrigating the rollup system, offering us clues on how to navigate this wide design space.

Since writing this piece, I presented more about Rollup Economics at EthCC in 2022. Find the video here!

And now, back to the piece…

Players in the rollup game

For simplicity, we'll idealise three distinct roles:

- The users:

Users send their transactions on the L2 network as they would on L1. They own assets on L2 and interact with contracts deployed on the rollup.

- The rollup operator:

This role subsumes many others. The rollup operator represents all the infrastructure necessary to process transactions received on the L2 network. We have in reality sequencers publishing transaction batches, executors posting assertions, challengers reporting fraud proofs, provers computing validity proofs (

see also here for details)…

- The base layer:

A protocol securing the data published by rollups. It is enough to consider a protocol that guarantees data availability with reduced functionality for settlement (e.g., "template" bridge contracts deployed on their execution layer) but you can also think of a general purpose settlement engine, like Ethereum's.

The users:

Users send their transactions on the L2 network as they would on L1. They own assets on L2 and interact with contracts deployed on the rollup.

The rollup operator:

This role subsumes many others. The rollup operator represents all the infrastructure necessary to process transactions received on the L2 network. We have in reality sequencers publishing transaction batches, executors posting assertions, challengers reporting fraud proofs, provers computing validity proofs (

see also here for details)…

The base layer:

A protocol securing the data published by rollups. It is enough to consider a protocol that guarantees data availability with reduced functionality for settlement (e.g., “template” bridge contracts deployed on their execution layer) but you can also think of a general purpose settlement engine, like Ethereum’s.

Rollup costs

In this post we adopt a “systems” view of the rollup, focusing on

costs, revenues

, and

fees

. Running a system induces

costs

, which are “energy sinks”, value flowing from inside the system to outside. The system might also receive

revenues

, which are the opposite, “energy sources”, value flowing from outside the system to inside.

Fees

bridge sources to sinks, transfer value throughout the system components so that each performs its function appropriately.

For instance, in a

previous piece on EIP-1559, I explained how to decompose the transaction fee paid by the user:

- Some of the fee goes to the operator including the transaction: in the Proof-of-Work base layer case, the miner fee compensates miners for the increased uncle risk. This fee makes whole the operator for the cost of publishing the block. This is the 1 or 2 Gwei you default to when sending a transaction today.
- The rest of the fee pays for the right to be included before others, pricing congestion. This fee makes whole the network and its users who suffer from your extra bit of congestion. This is what the L1 basefee aims to be, under normal conditions.

Some of the fee goes to the operator including the transaction: in the Proof-of-Work base layer case, the miner fee compensates miners for the increased uncle risk. This fee makes whole the operator for the cost of publishing the block. This is the 1 or 2 Gwei you default to when sending a transaction today.

The rest of the fee pays for the right to be included before others, pricing congestion. This fee makes whole the network and its users who suffer from your extra bit of congestion. This is what the L1 basefee aims to be, under normal conditions.

The good news is we will use some of the same thinking to understand costs on rollups. EIP-1559 costs involved only two parties: the

user

and the

base layer

. With a different architecture and the operator sitting between the user and the base layer, we must disassociate costs specific to the operator from costs specific to the base layer, which we do in the following sections.

L2 operator costs

Rollups must find operators willing to expend the

computational resources

to process the rollup data, i.e., maintain a transaction pool, sequence batches, compute state roots/state diffs/validity proofs, etc. This is a tangible cost, quantifying the dollars and cents necessary to run the infrastructure behind operators.

L1 data publication costs

It’s worth spending time on the

data publication cost

as this is truly the new cost in the rollup economy. Once an operator has assembled a large enough set of transactions, they are expected to post a compressed summary of the set to the base layer. Today, it is not done in a particularly elegant way: data is published by simply posting it as “CALLDATA”, a transaction attribute which allows the sender to add an arbitrary sequence of bytes.

Typically CALLDATA holds the reference to the smart contract method called by the transaction, with the inputs to that method. It may be helpful to think of rollup operators as calling some “registerCompressedData” method of the L1 “rollup chain” smart contract, with the blob of bytes representing the compressed transactions as its argument. Here is

an examplefrom Optimism’s “Canonical Transaction Chain” contract.

The cost of publishing data is incurred by the base layer. To publish data on Ethereum, the current market price of data is governed by EIP-1559, where each non-zero byte of CALLDATA consumes 16 gas, while each zero byte consumes 4 gas. With multi-dimensional EIP-1559, e.g., as instantiated in Vitalik’s simple

Sharding-format blob-carrying transactions, the price of CALLDATA could be found on its own EIP-1559 market, pricing the data market

separatelyfrom the traditional execution market.

In this

Dune analytics dashboard, I tried collating the data published by a few major rollups. It’s not clear to me that I am capturing all of it, especially for zk-rollups. If you notice discrepancies, please let me know! :)

See also Aditi’s recent post

Ethereum Rollup Call Data Pricing Analysisand

L2fees.info

L2 congestion costs

There is a third, more intangible cost. Whenever the supply of rollup blockspace cannot address existing demand, scarce resources must be allocated. Included users are included to the detriment of users who are not. In a spherical cow world of time-insensitive users, users simply queue and wait for their turn. There is no loss of value from a congested system. But when users incur costs for waiting, they will attempt to minimise their delay as much as possible. For users at the back of the queue, their welfare decrease is a cost on the rollup system as a whole.

It is typical (at least in the Ethereum world) to rely on fee markets to operate this allocation,

making this cost explicit.

1

Without a fee market or some form of congestion pricing, users either “pay in time” (are included later), bribe block proposers off-chain for inclusion or repeatedly resend their transaction to guarantee one of them will be picked up from the mempool. In all these instances, users express their loss of utility from congestion by expending resources to prevent this loss.

Rollup revenues

Now that we are clear about the costs of a rollup, we’ll try and reason about system revenues. Here we distinguish two main sources:

transaction value

and

issuance

.

Transaction value

Users obtain some

value

from transacting on a rollup rather than elsewhere, and are thus prepared to expend fees for the services they obtain. Value here refers to the

utility

a user obtains from having their transaction included on the rollup. If I have $50 utility for inclusion, I am willing to pay up to that amount so that my transaction is included. If I end up paying $2, my surplus is worth $48. The revenue of whoever receives the fee is $2, but from the rollup

system

perspective, the initial inflow is $50 worth of value.

Second, whenever the transaction contains positive

MEV

, e.g., the transaction is a sandwich-able swap on some DEX, this quantity is added to our notion of transaction value. It does not matter at this point who receives this value, whether it's the sequencer extracting the value, a user sandwiching the transaction or something else. The only thing that matters here is that our initial transaction brought more value to the system as a whole than the original user would receive from it. This gives us:

Transaction value = User value + MEV

Issuance

A second source of revenue is

issuance

. On the base layer, block producers receive revenue in the form of newly-minted coins, issuance of the cryptoasset native to the network they help maintain. This revenue offsets their infrastructure costs, with more block producers joining as long as it is profitable for them to do so.

Assume a rollup is able to mint their own token, and this token has non-zero value. The rollup may pay for some of its operations by issuing new tokens to cover its obligations. Here the model is fuzzier, with various ways of spending that revenue source on the rollup costs. For now, let's only consider that it is possible to issue valuable tokens, and by this act bring more value into the system.

Pass the buck: rollup edition

Summarising, a rollup system includes three parties: the users, the rollup operators and the base layer. Running the system incurs three types of costs: running cost of operations, data publication costs on the base layer and congestion costs. The system receives revenue in two forms: transaction value and issuance.

Now it is simply a game of matching who pays for what, and when. Some pairs are easy to dispatch. The operator

must

pay the L1 data publication fee to the base layer. They must pay for it exactly when they publish the data, and at the rate quoted by the base layer.

[2]

When fees are dynamic, priced in a fee market, L2 congestion costs are also immediate. Users observe the current demand for rollup blockspace and adjust their fees given the available supply. For instance, rollups may want to deploy an EIP-1559-style market mechanism on top of their network to govern inclusion of L2 transactions. An L2 basefee is then available to enable easy estimation of current L2 congestion costs by the users.

Budget balance: a limit to passing the buck

Let's add a new constraint to our system,

operator

budget balance

. We assume that the rollup operator cannot operate at a loss, i.e., they must at least receive a revenue equal to or greater than their costs. This assumption may not always hold, yet it appears critical to me if we care about a sufficiently

decentralised and open set of operators in a future where operators are decentralised. Operators who operate at a loss may price out less well-capitalised players and restrict the set of operators.

3

A smaller set of operators decrease censorship-resistance guarantees, where in the worst case a user must force include their transaction via the base layer, incurring high fees.

Issuance comes in handy as a slack variable to ensure budget balance. Whenever operators are “too unprofitable”, they leave the system, increasing the share of issuance received by remaining operators until balance is achieved again. Similarly when operators are “too profitable”, new entrants compete to eat a share of the profits until operators are once again budget-balanced.

Maintaining operator budget balance with delayed payments

With the budget balance rule, we must consider how operators maintain a non-negative balance. Their principal outflows, L1 data publication fees, are variable, and charged separately from their principal inflow, transaction fees. We make the assumption that operators have perfect knowledge of their L2 operator costs, and quote an exact price for it to the user at transaction time (akin to

their knowledge of uncle rates and the corresponding miner fee compensating them). But how should they quote users for the eventual L1 data publication cost, realised later in time?

Today, rollups apply heuristics to hedge themselves against L1 data publication cost variability. In one instance, the rollup observes the current L1 basefee (thanks,

EIP-3198!) and pad the rate with an extra buffer, overcharging the user early in case operators must pay more later to publish the data. In another, users are charged a function of the L1 basefee’s running average, to even out fluctuations over the long term.

In my view, the natural solution is to invoke derivatives, e.g.,

simple L1 basefee future contracts. At transaction time, the user is charged a fee defraying the cost of locking in a future price for the publication of data on the base layer. By reducing pessimistic overpayment, savings are passed on to the users. Investigating the optimal design of such derivatives remains an open question.

What to do with congestion fees?

Assuming the rollup perfectly prices the cost of congestion when users transact, there is now revenue available in the form of congestion fees. Today, on the Ethereum base layer, these fees are burned. The first reason for doing so is incentive-compatibility: should congestion fees return to the block producer, the protocol-quoted basefee would no longer be binding, destroying the purpose of EIP-1559. But

burning is not the only option preserving incentive-compatibility.

It has been proposed to direct all “rollup exhaust”, fees arising from economic externalities, e.g., congestion or MEV, towards public goods funding.

4

This is not a bad solution. Congestion pricing in cities is typically ear-marked for improvements of the public transportation system, i.e., it is money spent on compensating for the negative externalities which, when priced accordingly, provide this revenue.

Notice that I slipped MEV in there… why should we think about it the same way we think about congestion fees? First, because like congestion, MEV is an externality. The simple act of emitting an MEV-carrying transaction creates a positive externality to those who can capture it.

5

Externalities are “unmatched value”, i.e., they arise from some original economic activity that balanced useful work with payment for that work (e.g., users pay for the L2 operator costs; operators pay for the L1 publication costs), but they produce or destroy some extra bit of value in process.

This is most clearly articulated in the concept of

MEV auctions. In this design, operators compete for the right to make a block based on how much value they can extract from it. Implicit in this notion of value are congestion costs, which users express by bidding against one another. More explicit is the MEV itself, which would-be operators compete on. Once again, assuming no operator is allowed to operate at loss, their bids must reflect their true capacity of extracting value from the block, i.e., an operator would bid the sum of user fees in their batch

plus

the MEV they extract from the batch. Meanwhile, assuming all operators must pay an equal amount in L2 operator costs and L1 data publication costs are charged precisely to the user, we obtain:

- User fee

= L1 data publication fee + L2 operator fee + L2 congestion fee

- Operator cost

= L2 operator cost + L1 data publication cost

- Operator revenue =

User fees + MEV

- Operator profit = Operator revenue - Operator cost

= L2 congestion fee + MEV

User fee

= L1 data publication fee + L2 operator fee + L2 congestion fee

Operator cost

= L2 operator cost + L1 data publication cost

Operator revenue =

User fees + MEV

Operator profit = Operator revenue - Operator cost

= L2 congestion fee + MEV

In a world where operators compete in an efficient market to win the right to propose a block, the operators must bid away the entirety of their profits, i.e., exactly the congestion fees and the MEV available in their batch.

6

This is value that “slipped” into the system: the first from users protecting against losses due to congestion, the second from ripple effects caused by the initial transaction. This value was never anyone’s to begin with, so why shouldn’t it be captured and redistributed somehow?

Unboxing

Many assumptions were made in the piece. For instance, I have assumed “operator budget balance”, as I believe the community should consider rollups who operate at a loss with a critical eye, potentially less decentralisation-ready. Issuance helps to re-establish budget balance, though it relies on an exogenous price signal (the token value) to harmonise operator incentives. In this view, it remains preferable for operators to price as precisely as possible what they can price, e.g., their L2 operator costs and L1 data publication costs. This avoids future revenue mismatches, where an operator expected a higher token price to cover their operations.

7

But this is not a call to advocate for a specific form of rollup economics. The design space remains wide open. Unboxing L2 operator costs reveals more complexity that we haven’t explored here, e.g., the market institutions supporting a

decentralised infrastructure to generate validity proofs for zk-rollups Focusing on specific types of users on the rollup, e.g., fast withdrawal services from L2 to L1 or cross-L2 bridges, would also reveal different facets of user demand. With clear ideas about costs, revenues and fees, it is now hopefully easier to reason about the outcomes and business objectives a rollup should achieve, and the means to achieve them.

Other resources

- John Adler’s

Wait, it’s all resource pricing?(slides, video

here) gives more context to the L2 operator costs and the separability of execution and data availability costs.

- Patrick McCorry, Chris Buckland, Bennet Yee, Dawn Song,

SoK: Validating Bridges as a Scaling Solution for Blockchains

John Adler's

Wait, it's all resource pricing?(slides, video

here) gives more context to the L2 operator costs and the separability of execution and data availability costs.

Patrick McCorry, Chris Buckland, Bennet Yee, Dawn Song,

SoK: Validating Bridges as a Scaling Solution for Blockchains

Many thanks to Anders Elowsson, Vitalik Buterin, Fred Lacs and Alex Obadia for many useful comments, and Michael Silberling for his insights and discussion.

Vitalik also suggested that this cost is an

opportunity cost

to the blockspace supplier. In this reading, if you are included, you ought to pay the supplier at least what they could have won from including

someone else

instead.

This means we can unbox our model further. The data publication cost is quoted by evaluating congestion on the base layer and making whole base layer operators, i.e., block producers!

Centralisation of the rollup operator set may not be quite as bad as centralisation of the base layer block producer set, but evaluating decentralisation trade-offs to the rollup network is left for a future piece.

At the time of writing, even mismatched fees such as those received from users to hedge against data publication costs are directed to some public goods funding in the case of Optimism's retroPGF initiative.

Interestingly, with gas price auctions, the fight to capture this positive externality creates negative externalities to the network as a whole, who must deal with greater congestion!

Note that market efficiency is no longer a fair assumption in a world where some operators obtain private transaction order flow, or some are engaged in

cross-domain MEV(h/t Alex Obadia!) In the latter case, market efficiency among cross-domain extractors could re-establish market efficiency in single-domain builder auctions.

This model is not so terrible by the way! This is mostly the model under which miners have operated all this while. Yet we must bear in mind that any additional risk taken on by the operator is a fresh pressure to centralise, barring the availability of risk management primitives, e.g., derivatives. Even in the presence of such options to hedge against risk, the know-how necessary to conduct good business may be high, deterring less sophisticated operators.