

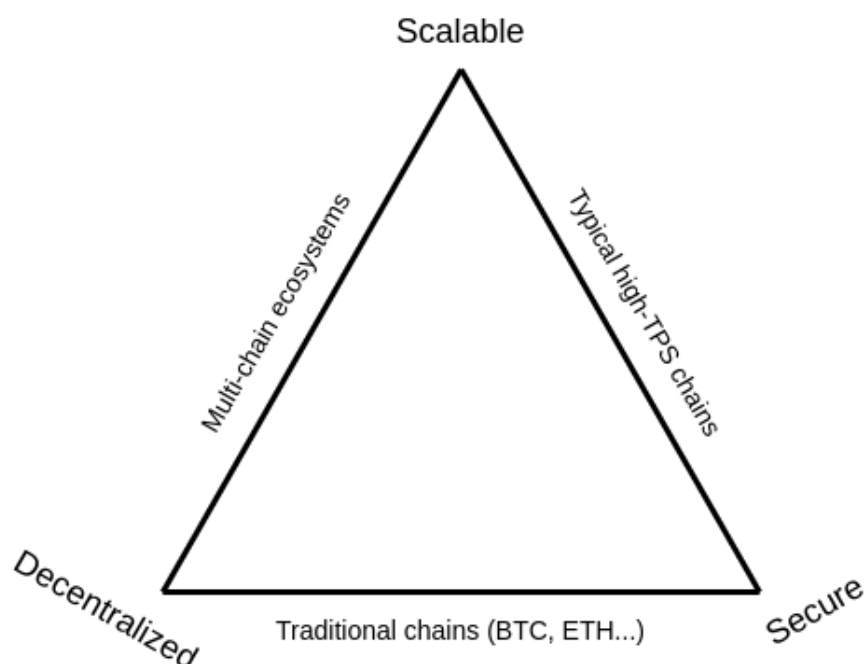
Lesson 7

Today's topics

- Scalability
- Layer 1 solutions
- Layer 2 theory
 - Rollups
 - Comparison of rollup types
 - Fractal Scaling
- State Channels
- zkEVM solutions
- Ethereum Scalability directions

Scalability Introduction

The scalability trilemma



"The decentralization of a system is determined by the ability of the weakest node in the network to verify the rules of the system." - Georgios Konstantopoulos

"For a blockchain to be decentralized, it's crucially important for regular users to be able to run a node, and to have a culture where running nodes is a common activity." - Vitalik [article](#)

On Ethereum there is a goal to keep the hardware requirements low.

There is a useful guide to scalability in their [documentation](#)

Approaches to Scalability



FIGURE 2. Taxonomy and comparison of blockchain scalability solutions.

From Scaling Blockchains: A Comprehensive Survey by Hafid et al.

Layer 1 Solutions

Tackling scalability at the L1 level is designing or redesigning your protocol to improve throughput, latency and finality.

Choice of Consensus Mechanism

Using a voting approach such as in BFT can have a negative impact on scalability. This is caused by the increase in the amount of messages being passed as the number of validators increases.

Therefore chains adopting a BFT based approach tend to use additional mechanisms to offset this.

On Ethereum validators form committees and votes are aggregated by committee.

Reducing transaction broadcasts

Solana moved away from using a gossip protocol to get transactions to the relevant parties. They reasoned that only the leader (block producer) needs to receive transactions, so on receiving transactions, nodes will send them to the leader rather than other nodes.

Parallel processing of transactions

In Ethereum transactions are ordered by the block producer and then executed sequentially. This has the benefit of simplicity, but leads to

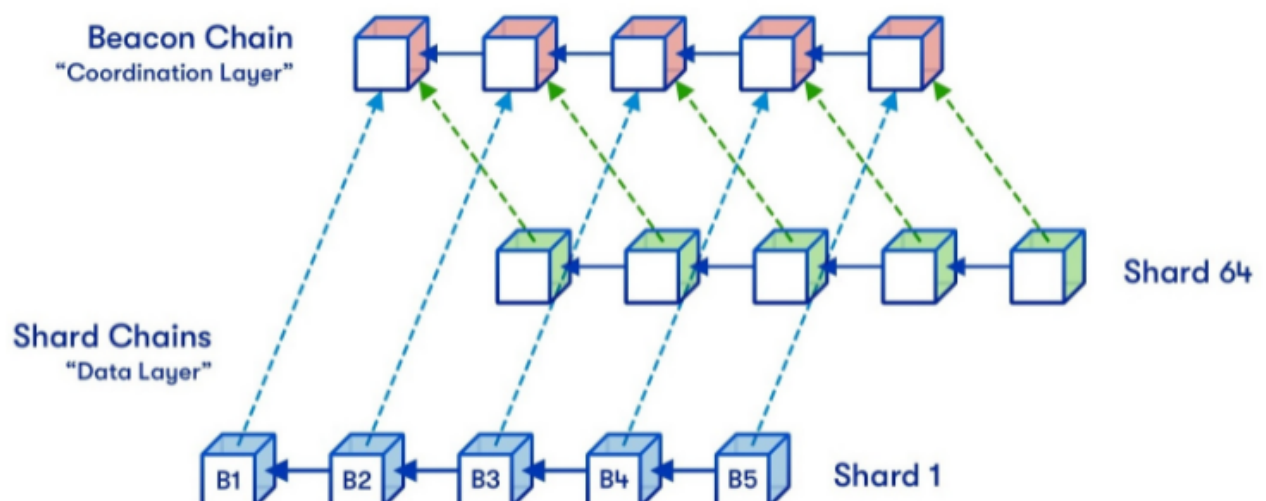
- MEV
- Poor horizontal scaling

More recent chains such as Solana / Aptos / Sui allow parallel processing of transactions.

They rely on an understanding of what areas of state will be changed by a transaction, and therefore a dependency among transactions. From this they can allow 'non dependent' transactions to be processed in parallel.

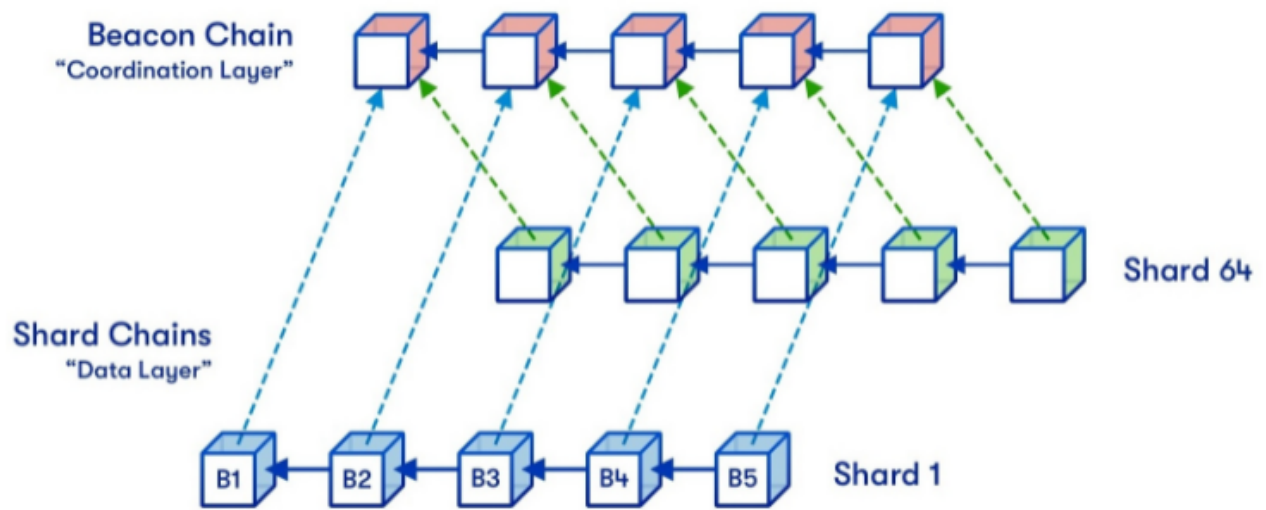
Sharding

Ethereum plans to introduce 64 new shard chains, to spread the network load.



Vitalik's [overview](#)

[Introduction](#)



Introduction of Sharding

Vitalik sees 3 options

- Shards remain as data depots
- A subset of the 64 shards will allow smart contracts
- Wait until increased use of ZKPs allows private transactions

Points from an [article](#) by Vitalik

There are three key limitations to a full node's ability to process a large number of transactions:

- **Computing power**: what % of the CPU can we safely demand to run a node?
- **Bandwidth**: given the realities of current internet connections, how many *bytes* can a block contain?

- **Storage**: how many gigabytes on disk can we require users to store? Also, how quickly must it be readable? (ie. is HDD okay or do we need SSD)

Many of the scalability approaches we see concentrate on the issue of computing power and how that can be made sufficient.

























Off chain Scaling

Generally speaking, transactions are submitted to these layer 2 nodes instead of being submitted directly to layer 1 (Mainnet). For some solutions the layer 2 instance then batches them into groups before anchoring them to layer 1, after which they are secured by layer 1 and cannot be altered.

A specific layer 2 instance may be open and shared by many applications, or may be deployed by one project and dedicated to supporting only their application.

Ethereum Layer 2 Ecosystem

thirdweb

Optimistic Rollups	<div><div> BASE</div><div> OP Mainnet</div><div> ZORA NETWORK</div><div> ARBITRUM</div><div> METIS</div></div>
Zero-Knowledge (ZK) Rollups	<div><div> polygon zkEVM</div><div> Linea</div><div> STARKNET</div><div> ImmutableX</div><div> zkSync</div><div> era</div><div> Aztec</div><div> Scroll</div><div> LOOPRING</div></div>
Sidechains	<div><div> polygon PoS</div><div> GnosisCHAIN</div><div> SKALE</div><div> loom</div></div>
Validiums	<div><div> polygon 2.0</div><div> STARKEx</div></div>
Channels	<div><div> connext</div><div> RAIDEN</div><div> PERUN</div><div> K</div></div>

Rollups are solutions that have

- transaction execution outside layer 1
- data or proof of transactions is on layer 1
- a rollup smart contract in layer 1 that can enforce correct transaction execution on layer 2 by using the transaction data on layer 1

The main chain holds funds and commitments to the side chains

The side chain holds state and performs execution

There needs to be some proof, either a fraud proof (optimistic) or a validity proof (zk)

Rollups require "operators" to stake a bond in the rollup contract. This incentivises operators to verify and execute transactions correctly.

There are currently 2 types of rollups

- Zero Knowledge Proof rollups
 - Optimistic rollups
-

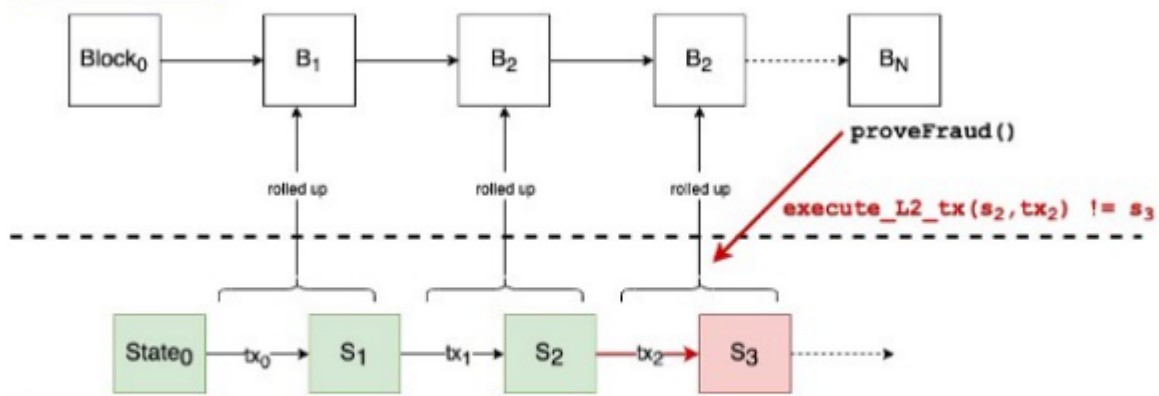
These rollups rely on a proof of the correctness of the execution that produces the rollup block state transition being supplied to a validator contract on L1.

The state transition on L2 will not be regarded as valid unless this proof has been validated. Although we use a zero knowledge proof, the zero knowledge aspect is usually ignored, the inputs and data involved is usually public, the focus is on the correctness of computation. For this reason some people prefer the term validity proof.

Optimistic Rollups

The name Optimistic Rollups originates from how the solution works. 'Optimistic' is used because aggregators publish only the bare minimum information needed with no proofs, assuming the aggregators run without committing frauds, and only providing proofs in case of fraud. 'Rollups' is used because transactions are committed to main chain in bundles (that is, they are rolled-up).

L1 (Ethereum)



L2 (Rollup)

Optimistic execution scales because L2 transactions can be replayed on L1 — but only when necessary!

See this [article](#) for further discussion of the differences between these types of rollups.

Rollups in detail

From Ethereum [Docs](#)

Optimistic rollup operators bundle multiple off-chain transactions together in large batches before submitting to Ethereum. This approach enables spreading fixed costs across multiple transactions in each batch, reducing fees for end-users. Optimistic rollups also use compression techniques to reduce the amount of data posted on Ethereum.

If the fraud proof succeeds, the rollup protocol re-executes the transaction(s) and updates the rollup's state accordingly. The other effect of a successful fraud proof is that the sequencer responsible for including the incorrectly executed transaction in a block receives a penalty.

If the rollup batch remains unchallenged (i.e., all transactions are correctly executed) after the challenge period elapses, it is deemed valid and accepted on Ethereum. Others can continue to build on an unconfirmed rollup block, but with a caveat: transaction results will be reversed if

based on an incorrectly executed transaction published previously.

Process

- Developer sends transaction off-chain to a bonded aggregator
- Anyone with a bond may become an aggregator.
- There are multiple aggregators on the same chain.
- Fees are paid however the aggregator wants (account abstraction / meta transactions).
- Developer gets an instant guarantee that the transaction will be included or else the aggregator loses their bond.
- Aggregator locally applies the transaction & computes the new state root.
- Aggregator submits an Ethereum transaction (paying gas) which contains the transaction & state root (an optimistic rollup block).
- If anyone downloads the block & finds that it is invalid, they may prove the invalidity

with `verify_state_transition(prev_state, block, witness)` which:

- Slashes the malicious aggregator & any aggregator who built on top of the invalid block.
- Rewards the prover with a portion of the aggregator's bond.

From [explanation](#) by Kelvin Fichter

A level 1 fault proof system is a system has an admin that can upgrade the system within the challenge window and can only be used by the admin.

A level 2 fault proof system still has an admin that can upgrade within the challenge window but is permissioned to allow a few others (besides the admin/team itself) to run the fault proof.




A level 3 fault proof is permissionless but still has an admin that can execute an upgrade within the challenge window. At level 3, you only need to trust that the admin won't do anything malicious and won't be compromised.

Level 4 proofs are completely permissionless and cannot be upgraded before users have a chance to withdraw their funds.

Level 4 fault proofs are the holy grail for an Optimistic Rollup.

OPTIMISM

TOOLS ▾DEVELOPER ▾COMMUNITY ▾INTEGRATIONS



Use live apps on Optimistic Ethereum today.

Transact in milliseconds, save 10-100x on fees.

DEPOSIT NOW

USER GUIDE

2.2M+

Transactions Processed

150+

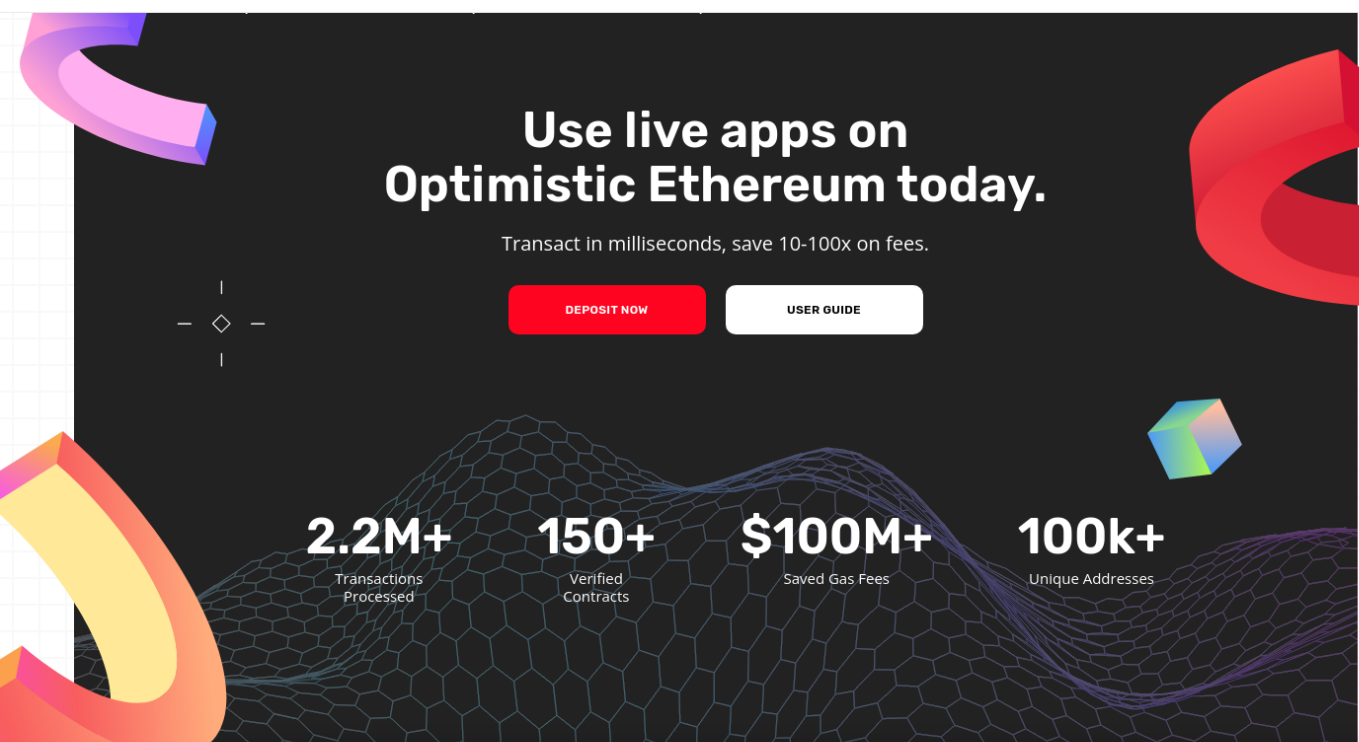
Verified Contracts

\$100M+

Saved Gas Fees

100k+

Unique Addresses



Building Arbitrum for Secure Ethereum Dapps.

Experience economical efficiency of the blockchain without limits.







ARBITRUM

Comparison of the rollup types

Property	Optimistic rollups	ZK rollups
Fixed gas cost per batch	~40,000 (a lightweight transaction that mainly just changes the value of the state root)	~500,000 (verification of a ZK-SNARK is quite computationally intensive)
Withdrawal period	~1 week (withdrawals need to be delayed to give time for someone to publish a fraud proof and cancel the withdrawal if it is fraudulent)	Very fast (just wait for the next batch)
Complexity of technology	Low	High (ZK-SNARKs are

Property	Optimistic rollups	ZK rollups
		very new and mathematically complex technology)
Generalizability	Easier (general-purpose EVM rollups are already close to mainnet)	Harder (ZK-SNARK proving general-purpose EVM execution is much harder than proving simple computations, though there are efforts (eg. Cairo) working to improve on this)
Per-transaction on-chain gas costs	Higher	Lower (if data in a transaction is only used to verify, and not to cause state changes, then this data can be left out,

Property	Optimistic rollups	ZK rollups
		whereas in an optimistic rollup it would need to be published in case it needs to be checked in a fraud proof)
Off-chain computation costs	Lower (though there is more need for many full nodes to redo the computation)	Higher (ZK-SNARK proving especially for general-purpose computation can be expensive, potentially many thousands of times more expensive than running the computation directly)

See this [article](#) from Starkware comparing the types of proofs

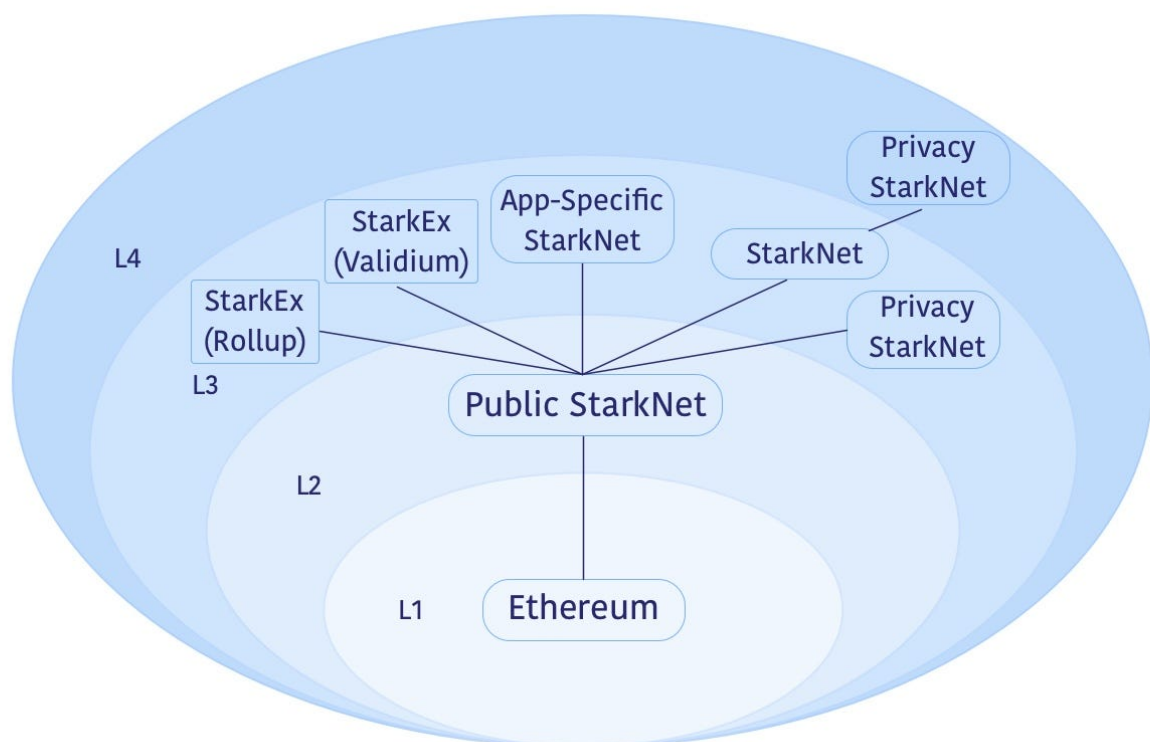
Fractal Scaling

The rollup process can be extended further by the use of recursive proofs, essentially you can create a proof that proves a secondary proof (that proves another proof)

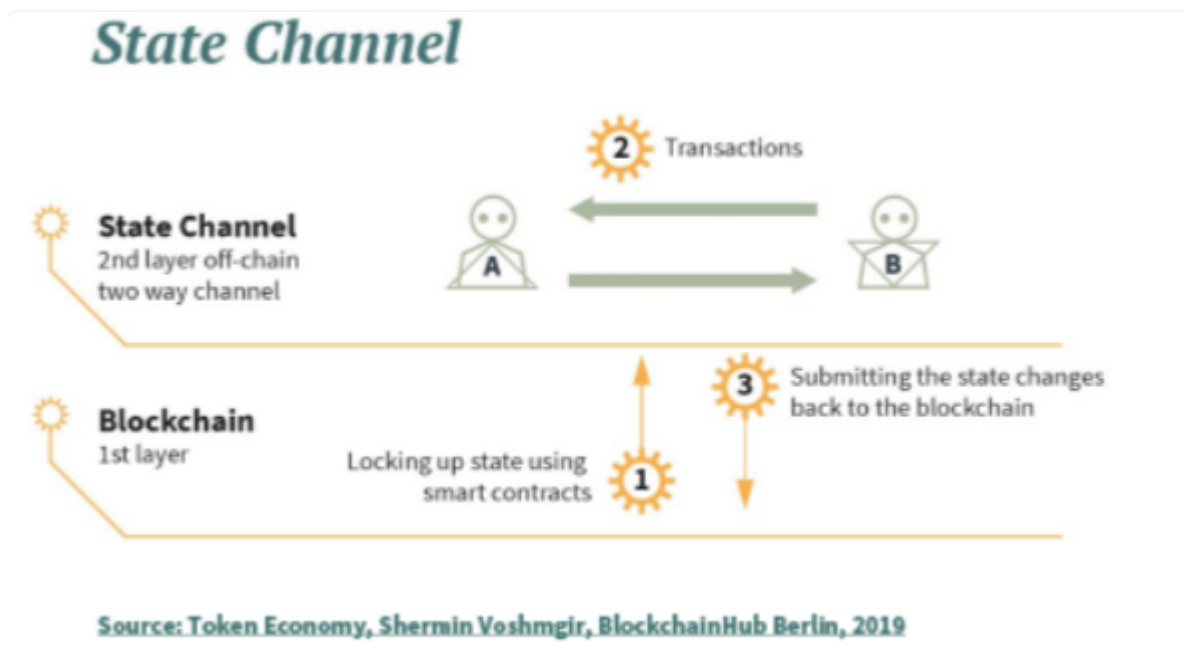
There is now [talk](#) about L3 for specific applications, where L3 a bundle of L3 proofs can be sent as a proof to L2, which will be part of the bundle of proofs sent to L1.

This gives a further boost to scalability.

For example with Starknet



State Channels



State channels

Payment channels are a specialised form of state channel

State channels allow participants to transact many off-chain while but only require 2 transactions on the L1 blockchain, one at the start and one at the end. An ideal use case for this is micropayments.

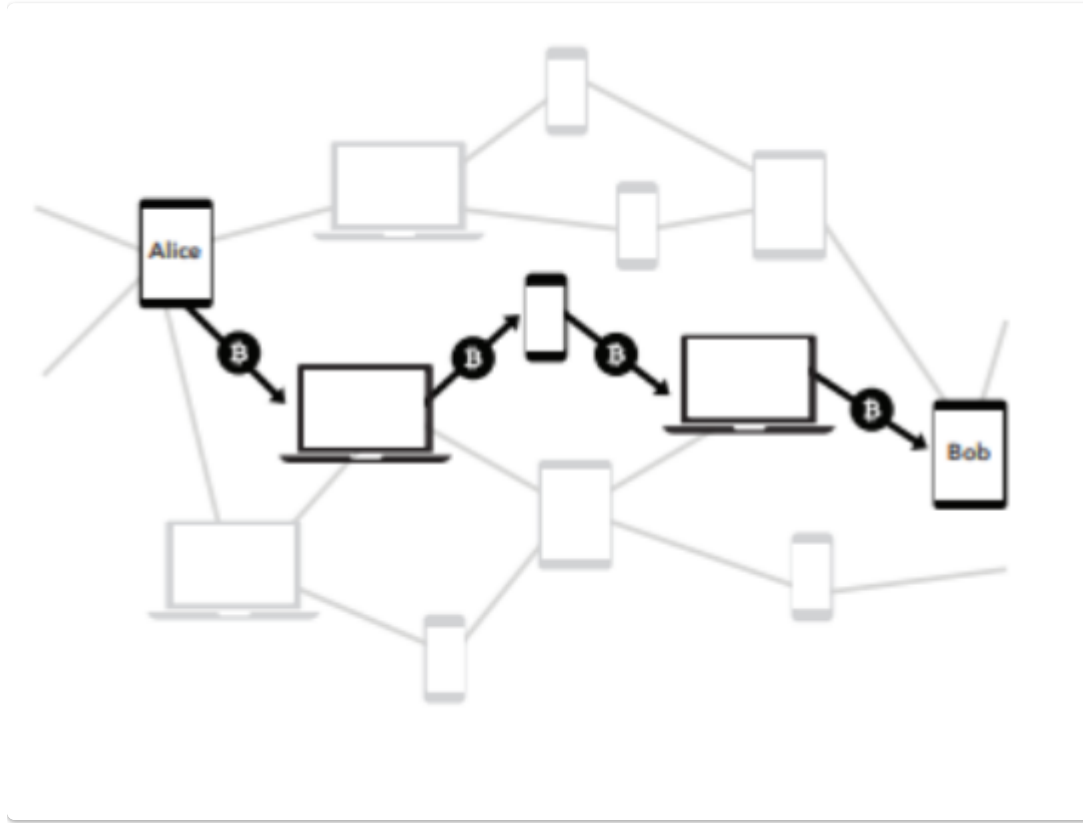
Participants must lock a portion of Ethereum's state, like an ETH deposit, into a multisig contract.

Locking the state in this way is the first transaction and opens up the channel. The participants can then transact quickly and freely off-chain. When the interaction is finished, a

final on-chain transaction is submitted, unlocking the state.

Examples

Lightning network



Funds are placed into a two-party, multisignature "channel" bitcoin address. This channel is represented as an entry on the bitcoin public ledger. In order to spend funds from the channel, both parties must agree on the new balance. The current balance is stored as the most recent transaction signed by both parties, spending from the channel address. To make a payment, both parties sign a new exit transaction spending from the channel address.

All old exit transactions are invalidated by doing so. The Lightning Network does not require cooperation from the counterparty to exit the channel. Both parties have the option to unilaterally close the channel, ending their relationship. Since all parties have multiple multisignature channels with many different users on this network, one can send a payment to any other party across this network.

Advantages

- Instant Payments.

Bitcoin aggregates transactions into blocks spaced ten minutes apart. Payments are widely regarded as secure on bitcoin after confirmation of six blocks, or about one hour. On the Lightning Network, payments don't need block confirmations, and are instant and atomic. Lightning can be used at retail point-of-sale terminals, with user device-to-device transactions, or anywhere instant payments are needed

- Micropayments.

New markets can be opened with the possibility of micropayments. Lightning

enables one to send funds down to 0.00000001 bitcoin without custodial risk. The bitcoin blockchain currently enforces a minimum output size many hundreds of times higher, and a fixed per-transaction fee which makes micropayments impractical. Lightning allows minimal payments denominated in bitcoin, using actual bitcoin transactions.

zkEVM Solutions

Rollup Recap

Rollups are solutions that have

- transaction execution outside layer 1
- transaction data and proof of transactions is on layer 1
- a rollup smart contract in layer 1 that can enforce correct transaction execution on layer 2 by using the transaction data on layer 1

The main chain holds funds and commitments to the side chains

The side chain holds additional state and performs execution

There needs to be some proof, either a fraud proof (Optimistic) or a validity proof (zk)

Rollups require “operators” to stake a bond in the rollup contract. This incentivises operators to verify and execute transactions correctly.

Introductory video

[zkEVMs](#)

[zkEVM overview - Dune dashboard](#)

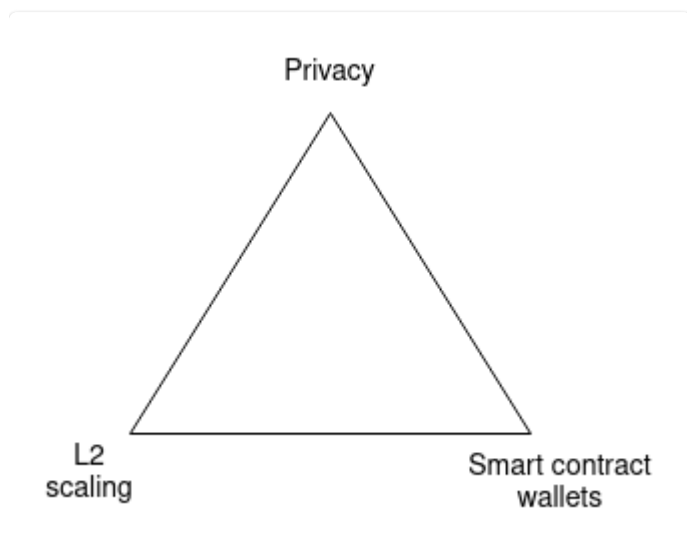
Ethereum Scalability directions

See [article](#) by Vitalik

Vitalik sees three major transitions that need to occur in Ethereum

- **The L2 scaling transition** - everyone [moving to rollups](#)
- **The wallet security transition** - everyone moving to [smart contract wallets](#)
- **The privacy transition** - making sure privacy-preserving funds transfers are available, and making sure all of the *other* gadgets that are being developed (social recovery, identity, reputation) are privacy-preserving

As a tie in with the scalability trilemma, here you can pick 3 out of 3



Without the first, Ethereum fails because each transaction costs \$3.75 (\$82.48 if we have another bull run), and every product aiming for the mass market inevitably forgets about the chain and adopts centralized workarounds for everything.

Without the second, Ethereum fails because users are uncomfortable storing their funds (and non-financial assets), and everyone moves onto centralized exchanges.

Without the third, Ethereum fails because having all transactions (and POAPs, etc) available publicly for literally anyone to see is far too high a privacy sacrifice for many users, and everyone moves onto centralized solutions that at least somewhat hide your data.