Lesson 2

Week 1

Lesson 1 - Introduction to Blockchain and Layer 1

Lesson 2 - Why Scalability

Lesson 3 - Introduction to Layer 2

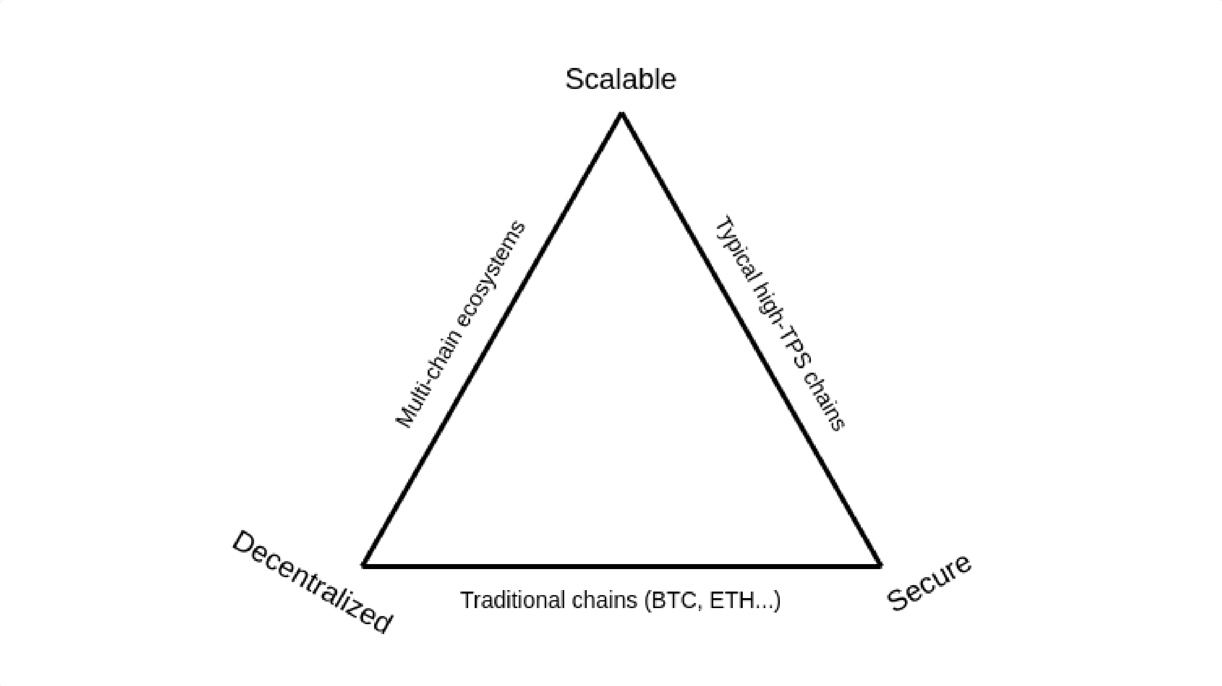
Lesson 4 - Maths and Cryptography

Today's Topics

- Introduction to scalability
- Possible approaches
- Layer 1 solutions
- Off chain (Layer 2) solutions
- State channels
- Rollups and Plasma chains
- Side chains
- Future 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]

> **Measuring performance in blockchains**
>
> To assess how well different blockchains scale, we need to measure their performance, but this can be ill defined and different chains may have different features and aims.
> Metrics such as TPS (transactions per second) should always have context, and although useful for promotional material may have little real value.
> This article has an insightful look at the problem.
> As a rough guide, Bitcoin and Ethereum are thought to be 2-3 orders of magnitude slower than centralised commercial systems such as Visa.
> More recent Layer 1s such as Avalanche / Solana / Sui may be closer to centralised systems.

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

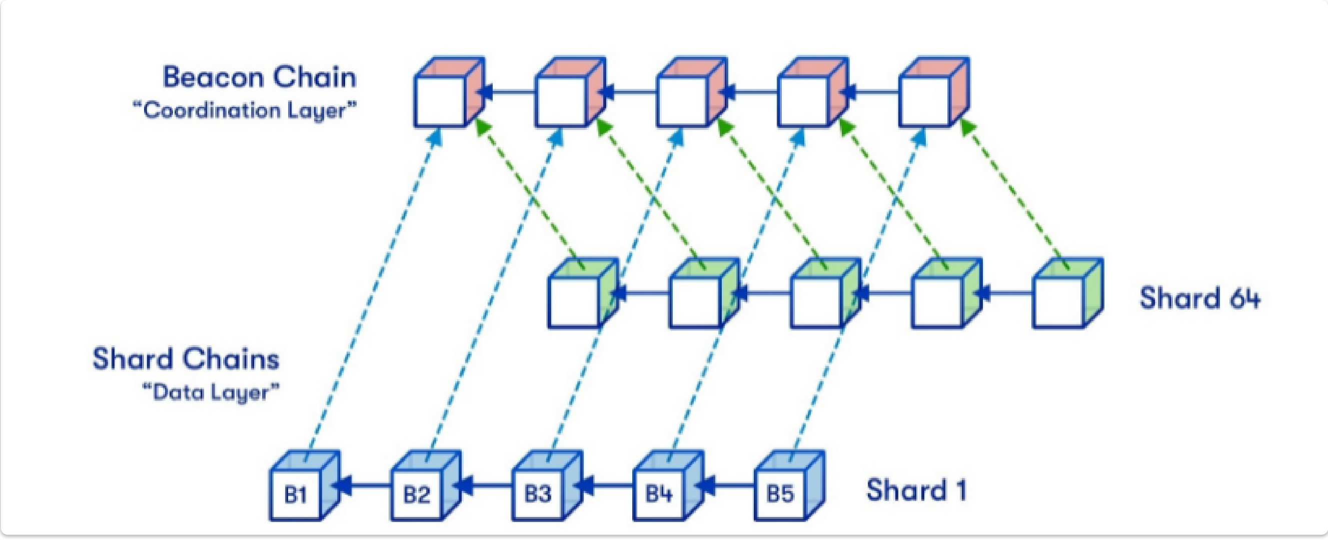
In Ethereum transactions are ordered by the block producer and then executed sequentially.
This has the benefit of simplicity, but leads to

- 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

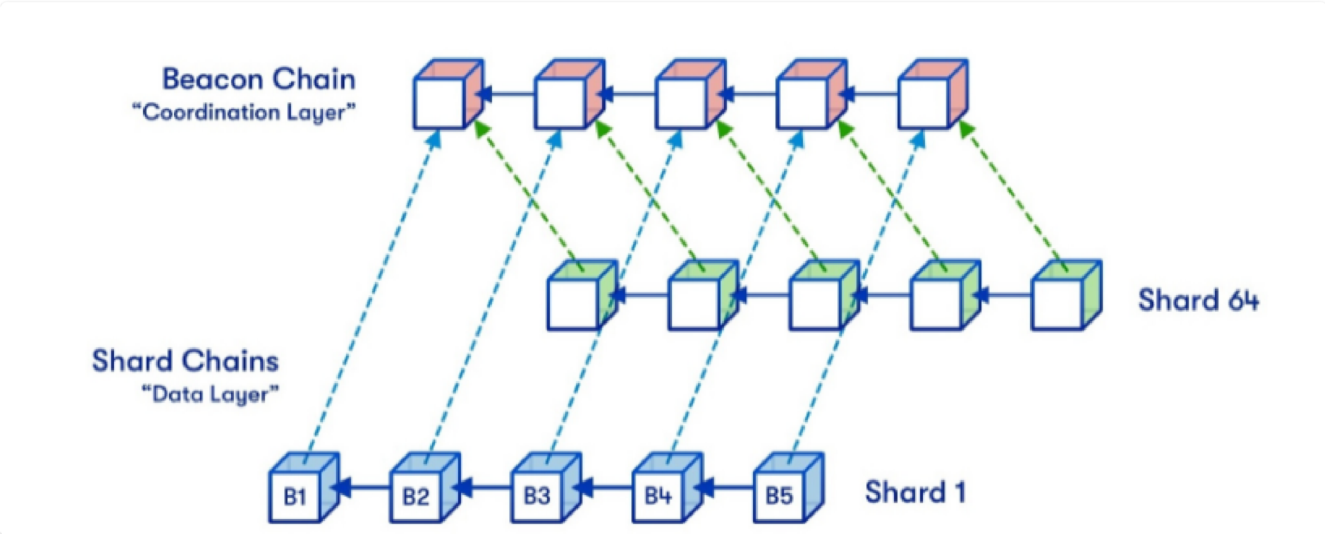
Ethereum plans to introduce 64 new shard chains, to spread the network



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

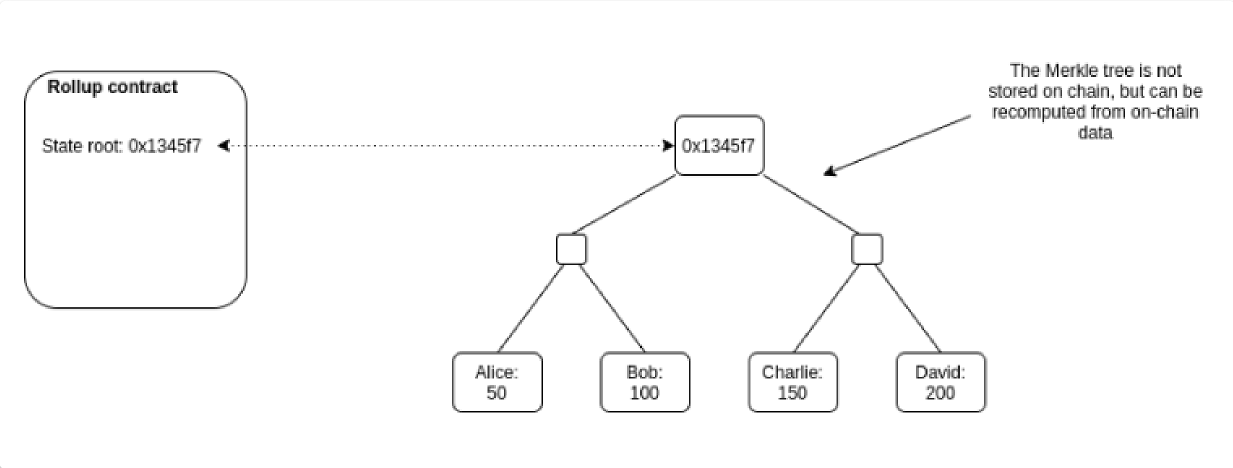
Points from an article by Vitalik

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

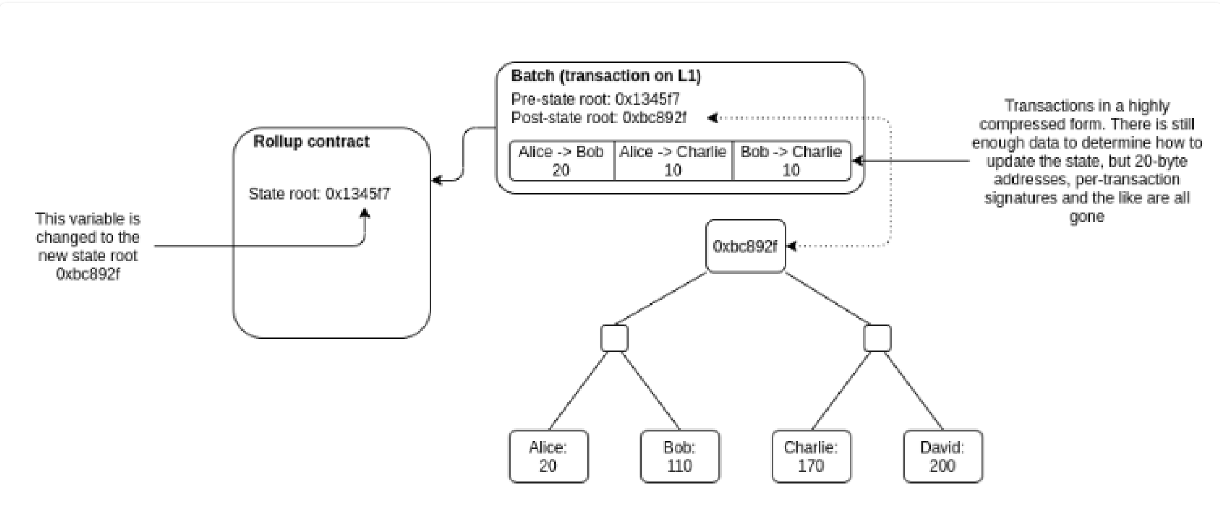
- Computing power: what % of the CPU can we safely demand to run a node?

> - **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.

TODO

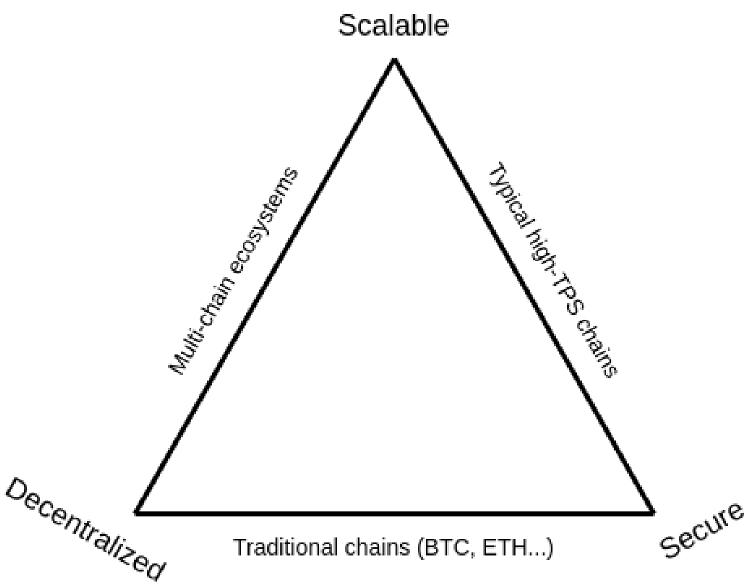


Anyone can publish a batch, a collection of transactions in a highly compressed form together with the previous state root and the new state root (the Merkle root after processing the transactions). The contract checks that the previous state root in the batch matches its current state root; if it does, it switches the state root to the new state root.



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

Measuring performance in blockchains

To assess how well different blockchains scale, we need to measure their performance, but this can be ill defined and different chains may have different features and aims.
Metrics such as TPS (transactions per second) should always have context, and although useful for promotional material may have little real value.
This article has an insightful look at the problem.
As a rough guide, Bitcoin and Ethereum are thought to be 2-3 orders of magnitude slower than centralised commercial systems such as Visa. More recent Layer 1s such as Avalanche / Solana / Sui may be closer to centralised systems.

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

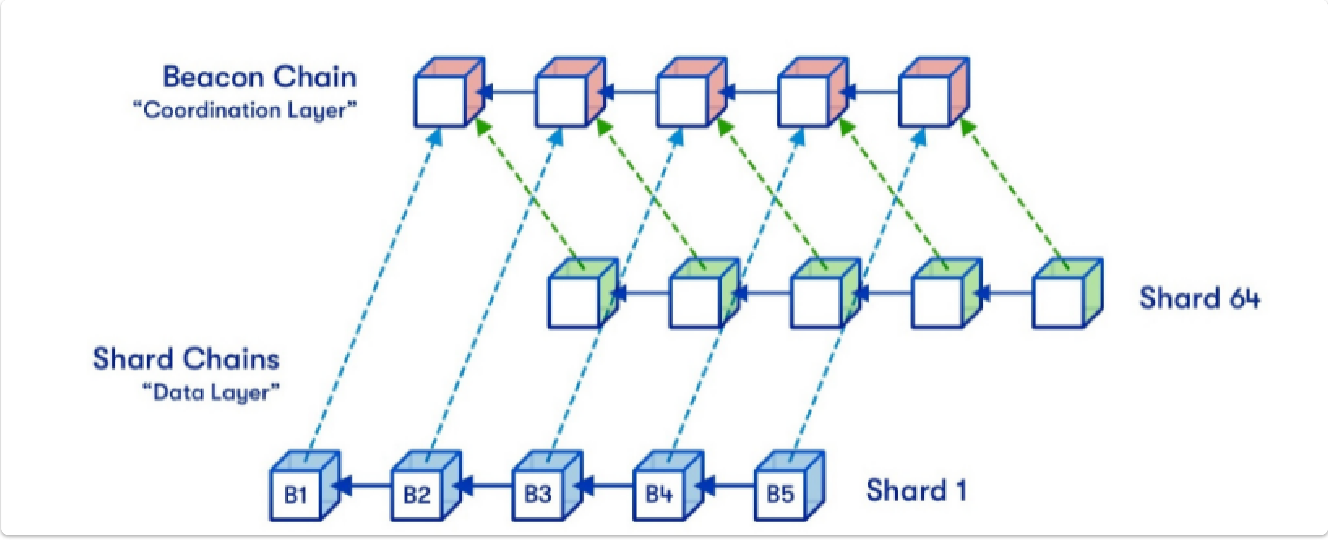
In Ethereum transactions are ordered by the block producer and then executed sequentially.
This has the benefit of simplicity, but leads to

- 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

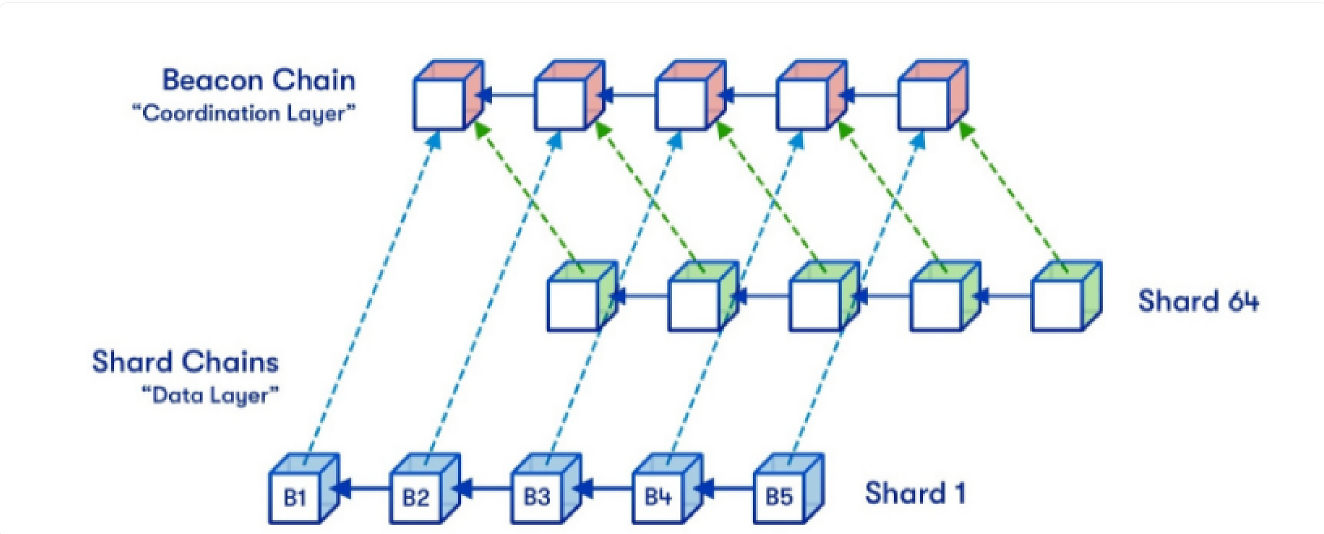
Ethereum plans to introduce 64 new shard chains, to spread the network



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?

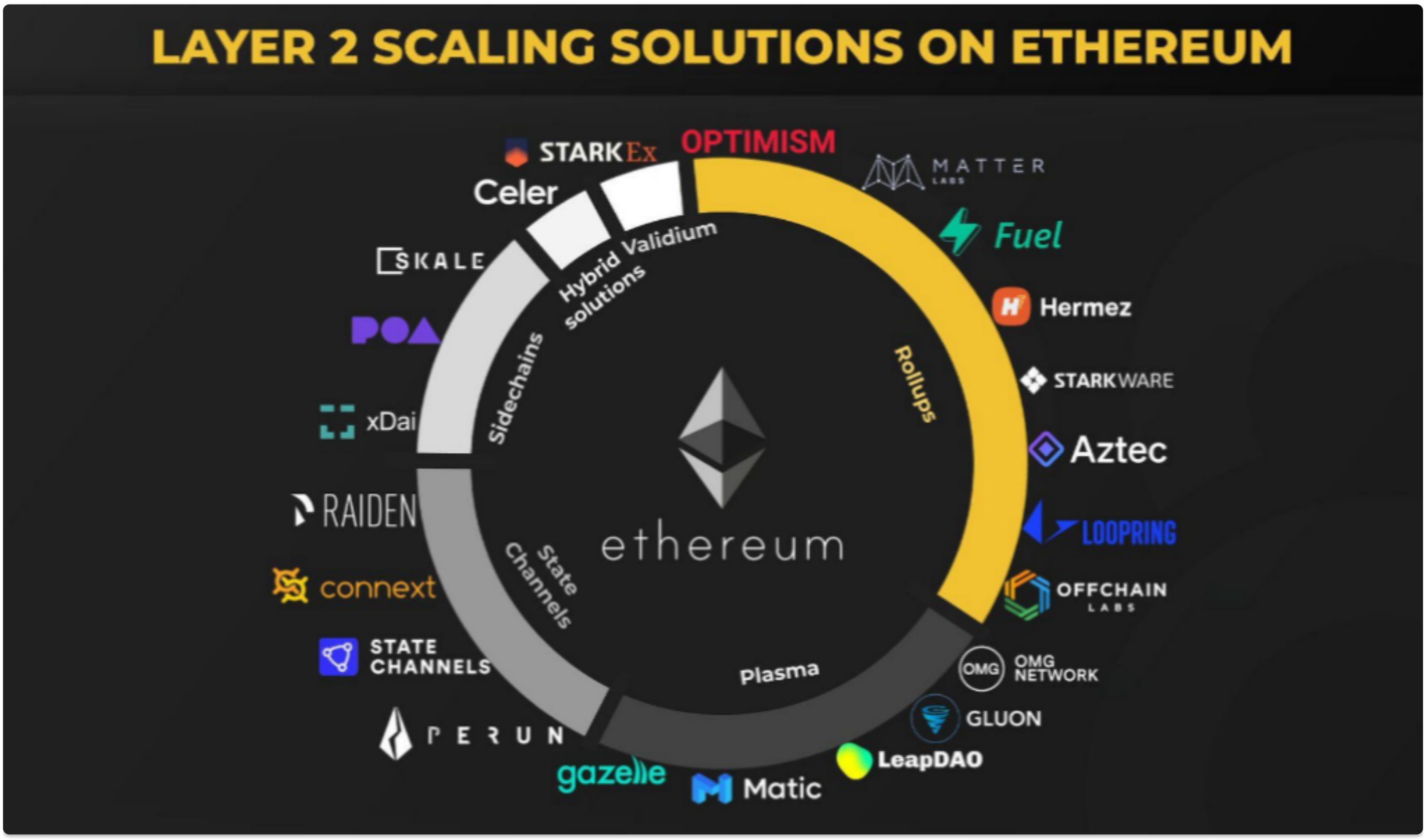
- **Bandwidth**: given the realities of current internet connections, how many *bytes* can a block contain?
- **Storage**: how many gigabytes on disk can we require users to store? Also, how quickly must it be readable? (ie. is HDD okay or do we need SSD)

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.

Early Layer 2 Solutions on Ethereum



Rollups

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

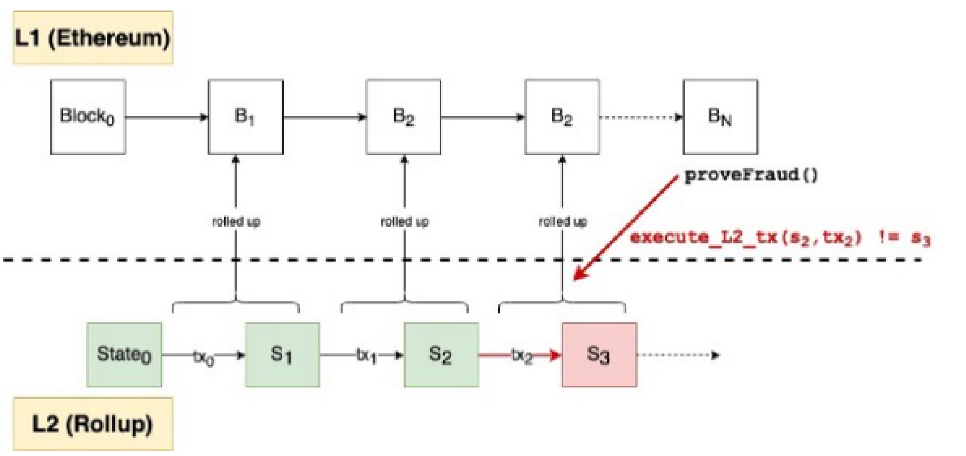
- Zero Knowledge Proof rollups
- Optimistic rollups

ZKP or validity 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 commiting frauds, and only providing proofs in case of fraud. 'Rollups' is used because transactions are commited to main chain in bundles (that is, they are rolled-up).

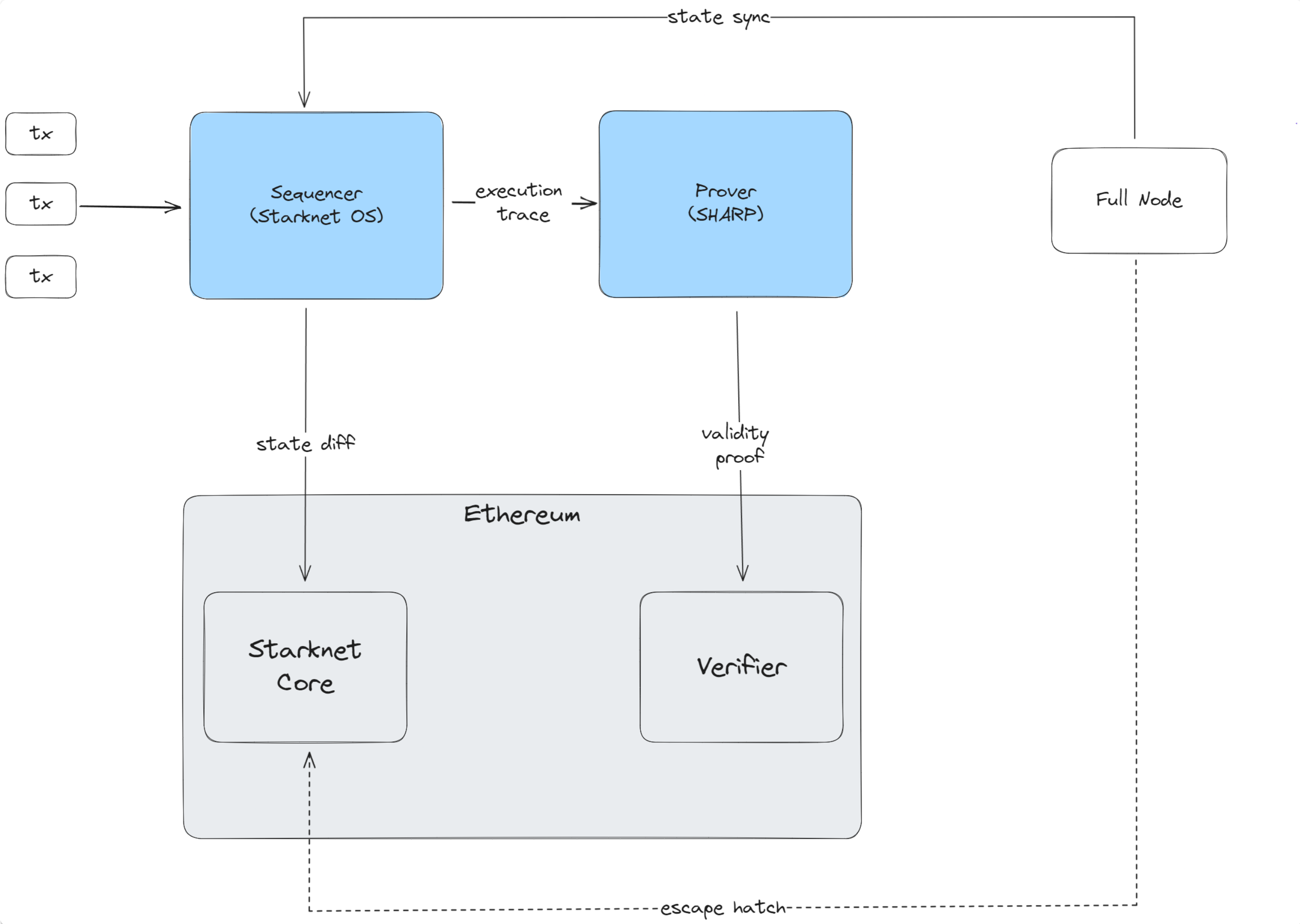


Optimistic execution scales because L2 transactions can be replayed on L1 — but only when necessary!

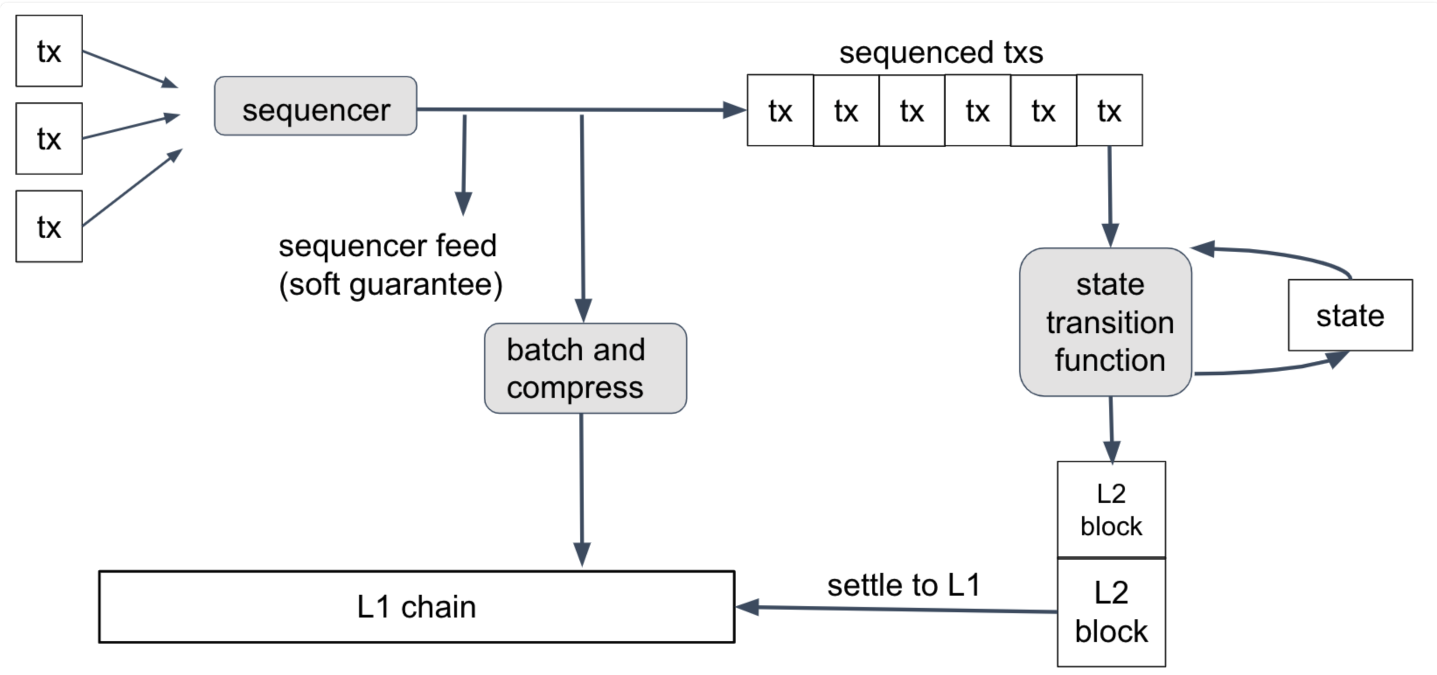
See this article for further discussion of the differences between these types of rollups.

Typical Rollup Architecture

This is an overview of the Starknet architecture



Here is Arbitrum's design



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

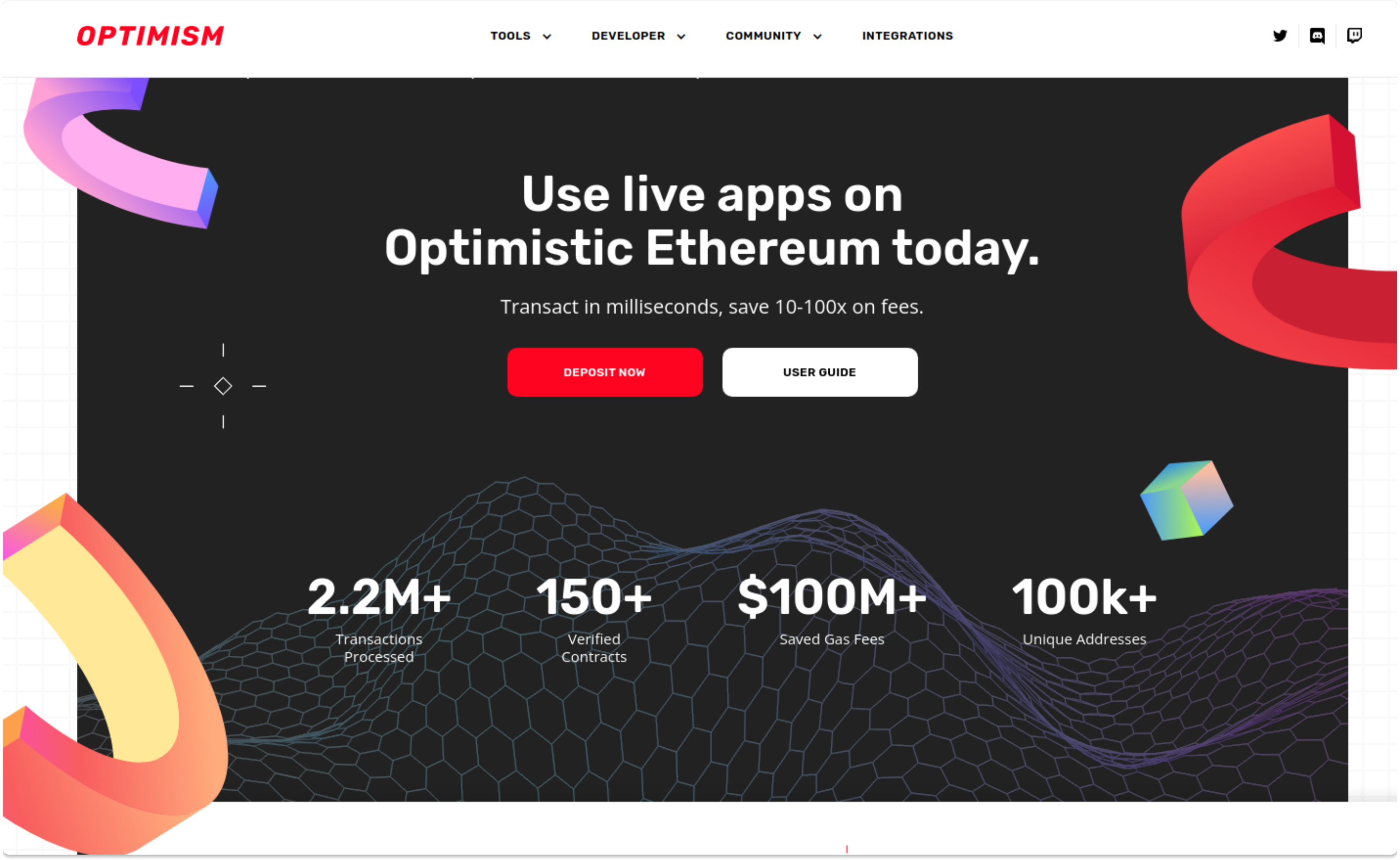
A level 1 fault proof system is a system has an admin that can upgrade the system within the challenge window and can only be used by the admin.

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.

Example Projects



Zero Knowledge Proof Rollups

See Ethworks Report

Note that in this context the proofs produced are often referred to as validity proofs since the zero knowledge aspect is not required.

ZK Rollup Process

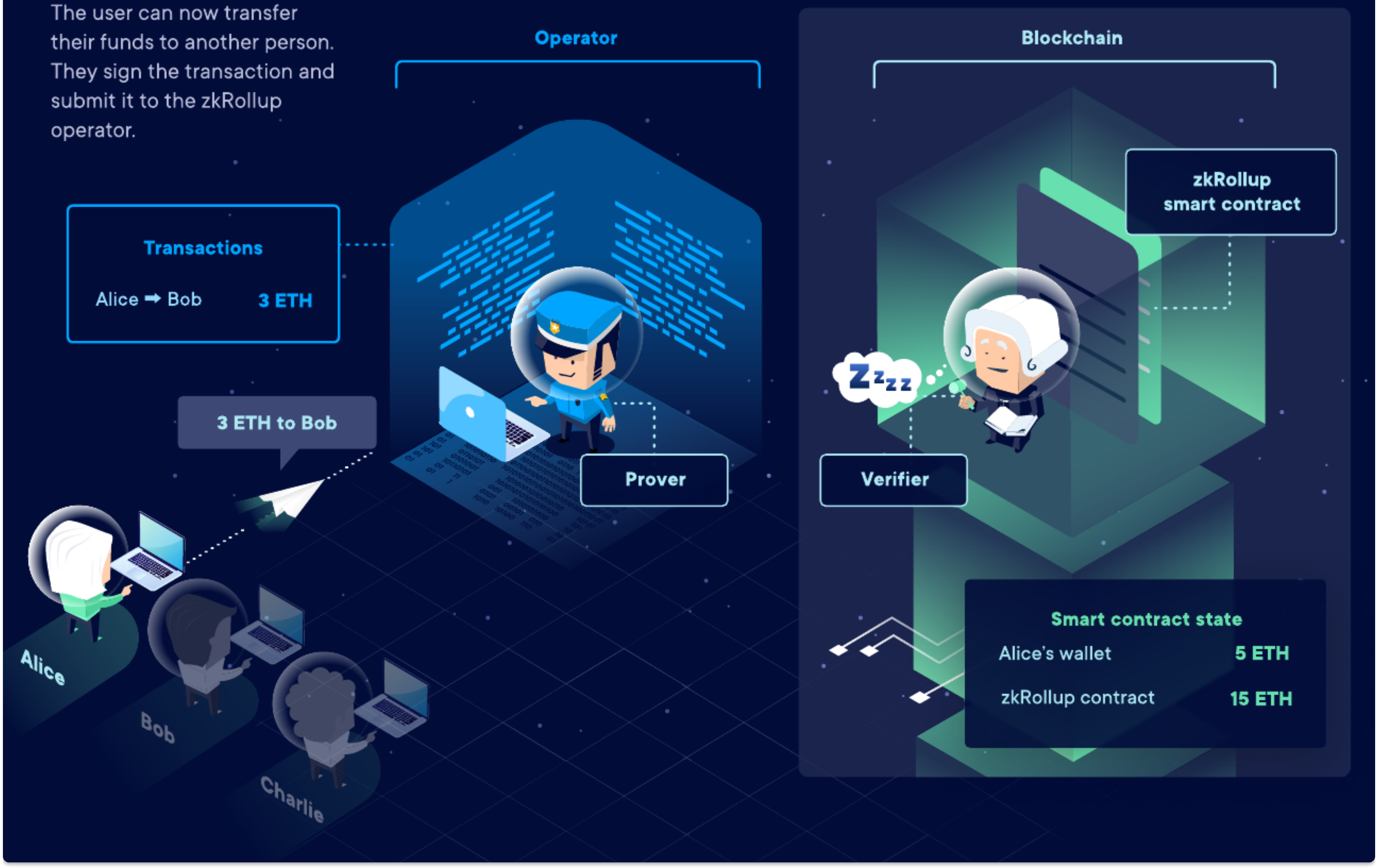
From Ethworks

02. Alice's Enter

To enter the system, the user needs to transfer their funds to the zkRollup. The assets are sent to a smart contract.

03. Alice's Transfer

The user can now transfer their funds to another person. They sign the transaction and submit it to the zkRollup operator.



04. Bob's Transfer

05. Charlie's Exit

If a user wishes to withdraw their funds from the zkRollup, they can submit their exit request to the operator any time.

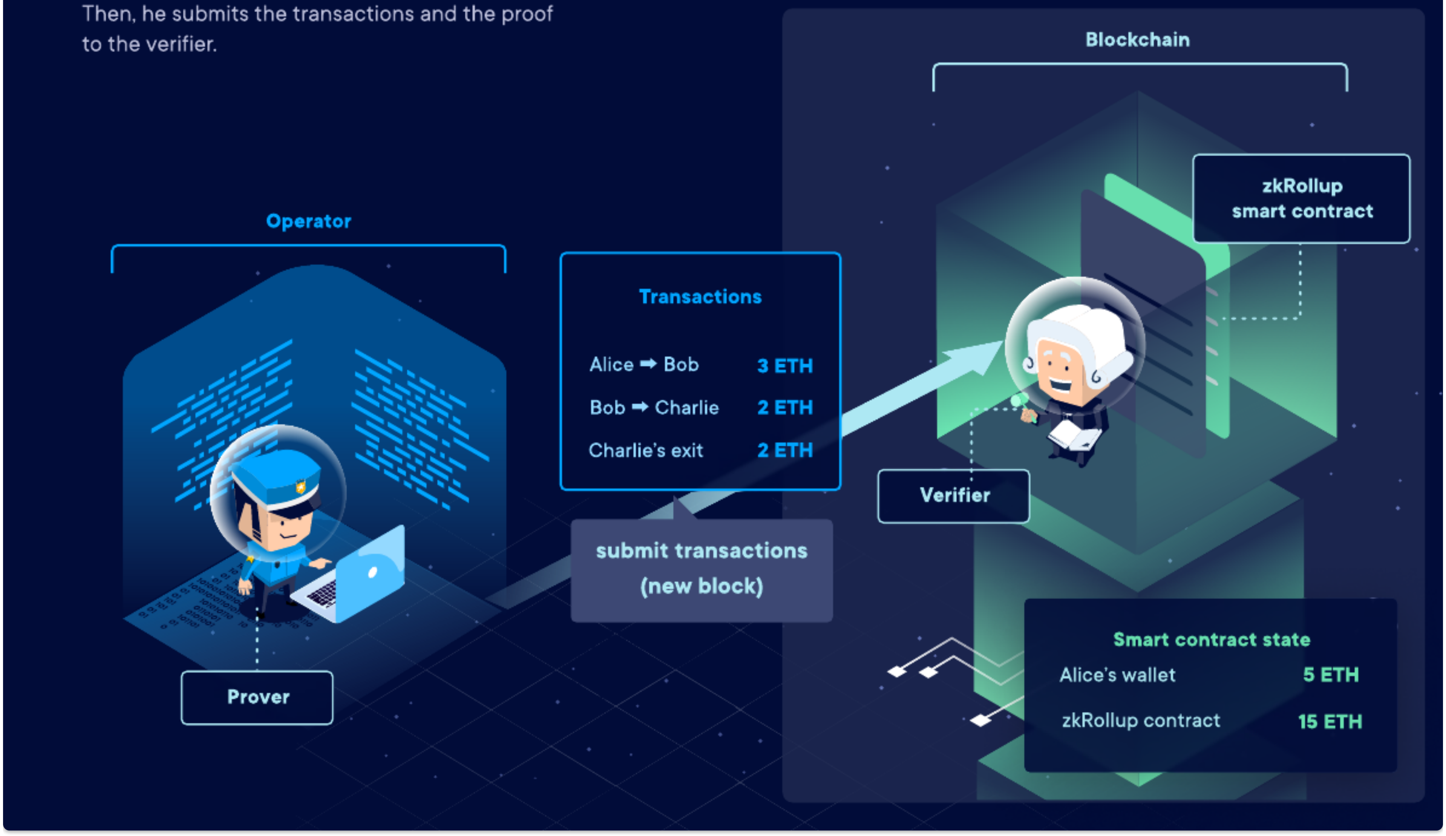
06. Collecting Transactions

In the meantime, the operator collects transactions and exit requests from many users.

* Note that even if Bob and Charlie didn't have any funds on the zkRollup, they could still receive transfers from other users.

07. Submitting Transactions

Once in a while, the operator bundles the collected transactions together and generates a ZK proof. Then, he submits the transactions and the proof to the verifier.




08. Submitting ZK Proof
The smart contract verifies the transactions and the proof. Once it's done, the transactions are finalized.
Operator
Blockchain
zkRollup smart contract
submit ZK proof (new block)
Verifier
Prover
Calldata
Alice ➡ Bob 3 ETH
Bob ➡ Charlie 2 ETH
Charlie's exit 2 ETH
Smart contract state
Alice's wallet 5 ETH
Charlie's wallet 2 ETH
zkSync contract 13 ETH

How does compression work?

A simple Ethereum transaction (to send ETH) takes ~110 bytes. An ETH transfer on a rollup, however, takes only ~12 bytes:

Parameter	Ethereum	Rollup
Nonce	~3	0
Gasprice	~8	0-0.5
Gas	3	0-0.5
To	21	4
Value	~9	~3
Signature	~68 (2 + 33 + 33)	~0.5
From	0 (recovered from sig)	4
Total	~112	~12

Part of this is simply superior encoding: Ethereum's RLP wastes 1 byte per value on the length of each value. But there are also some very clever compression tricks that are going on:

Data Availability

In order to re create the state, transaction data is needed, the data availability question is where this data is stored and how to make sure it is available to the participants in the system.

	Validity Proofs		Fault Proofs
Data On-Chain	Volition	ZK-Rollup	Optimistic Rollup
Data Off-Chain		Validium	Plasma

STARKWARE

See Docs

StarkNet is currently in ZK-Rollup mode (see above). This means that upon the acceptance of a state update on-chain, the state diff between the previous and new state is sent as calldata to Ethereum.

This data allows anyone that observes Ethereum to reconstruct the current state of StarkNet. Note that to update the StarkNet state on L1, it suffices to send a valid proof — without information on the transactions or particular changes that this update caused. Consequently, more information must be provided in order to allow other parties to locally track StarkNet's state.

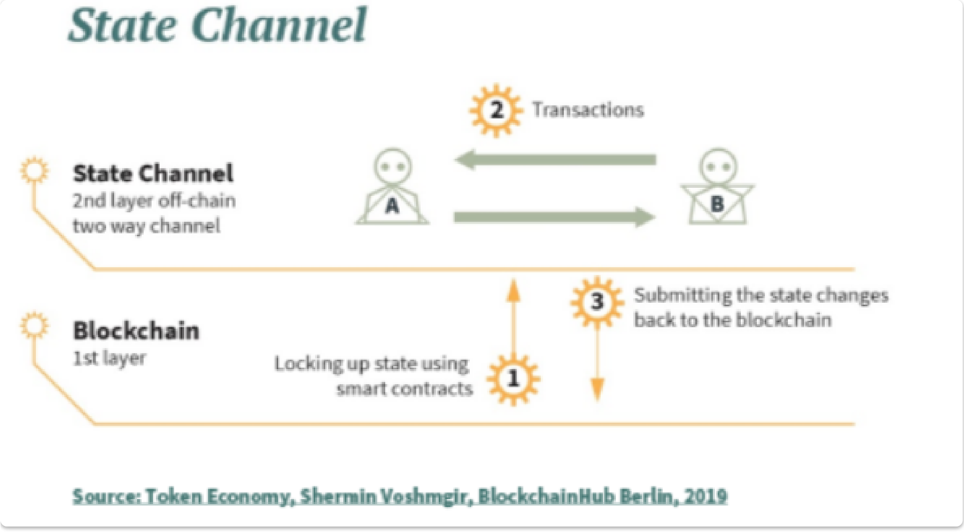
Comparison of the rollup types

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)

Property	Optimistic rollups	ZK rollups
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 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, 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

State Channels



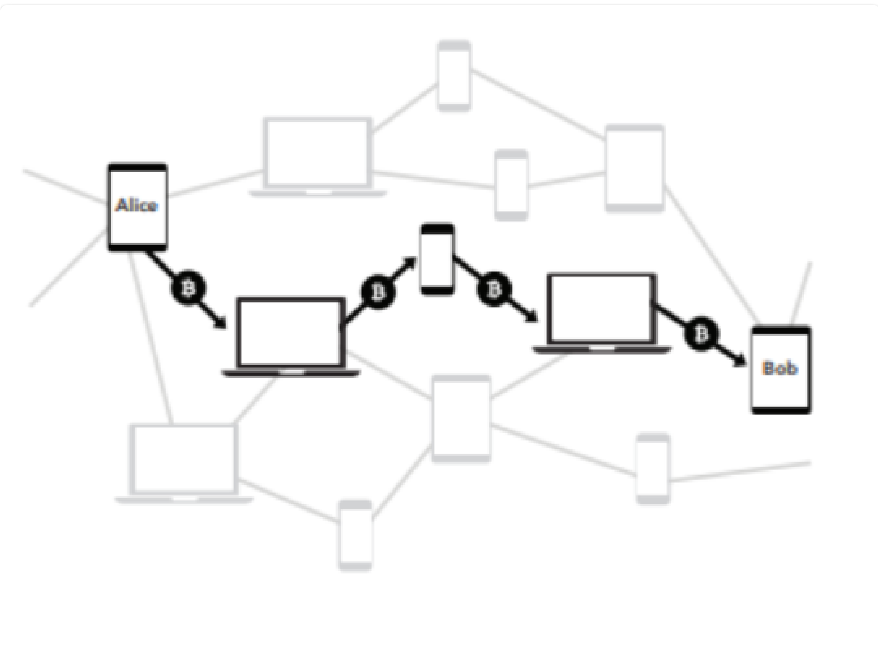
State channels

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.

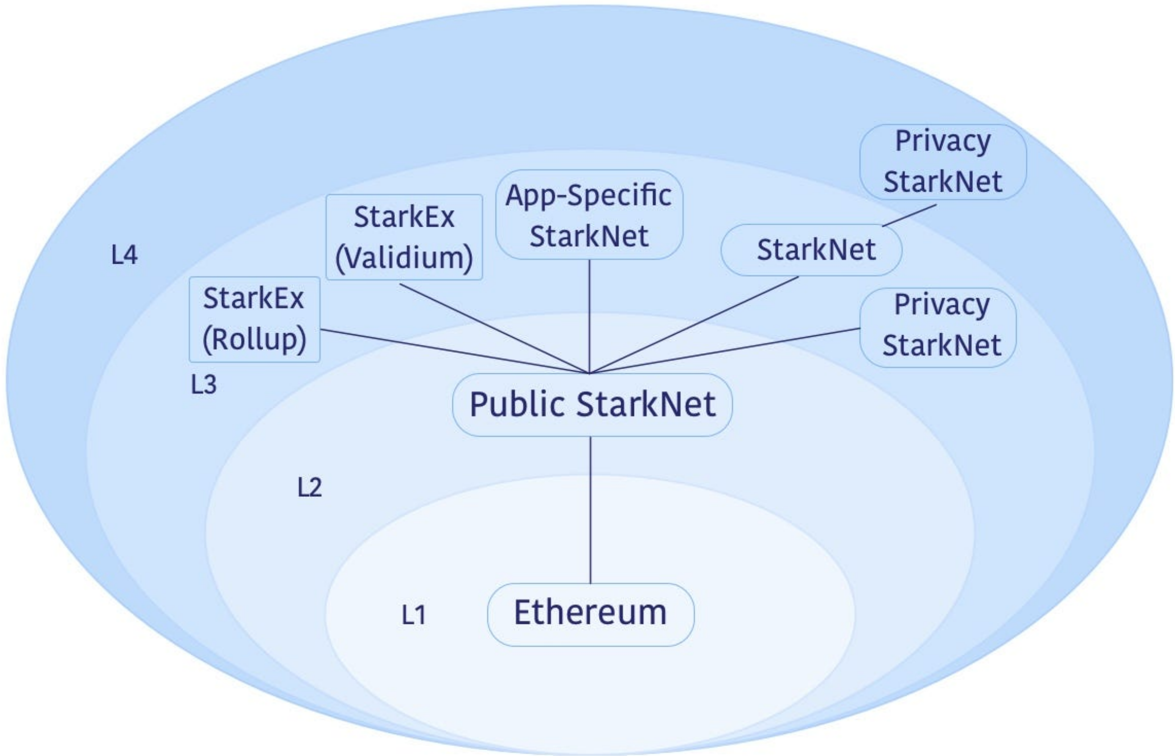
Advantages

- 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.

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



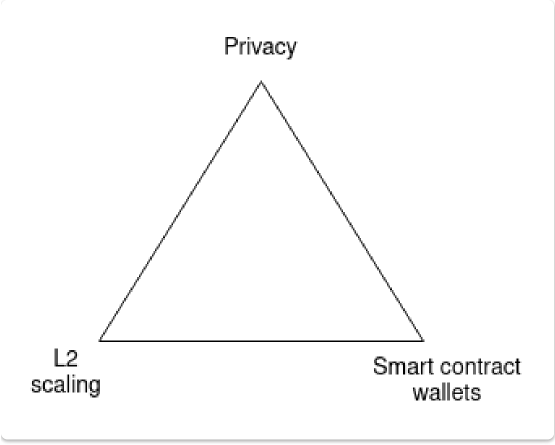
Ethereum Scalability directions

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.

BNB Sidechain

Architecture

BNB Sidechain serves as a framework within the BNB Smart Chain (BSC) ecosystem, designed for the creation of sidechains.
This infrastructure supports developers and node operators in constructing and operating their tailor-made blockchains, which function as internal value systems for numerous users while maintaining strong ties to the BSC.
BNB Sidechain's primary objective is to enable project developers to deploy their distinct blockchains, complete with unique specifications and validator sets, while remaining connected to the BSC infrastructure.
Depending on the BNB Sidechain deployer, the validator set may operate with fewer validators than the BNB Chain.

Application owners or community stakeholders can run these validators, introducing greater flexibility and decentralisation to BNB Sidechain.

Developers and teams can create customised blockchains featuring their own business regulations and economies, and most importantly, expand the existing functionality of the BNB Chain.

At its core, BNB Sidechain consists of a collection of smart contracts that can be coded in any programming language. There are no specific requirements for the contract executor, allowing for flexibility in implementation and the freedom to use any desired programming language or API standards.

BNB Sidechain primarily determines the fundamental structure and configuration of the blockchain through the use of specialised templates. These templates are pre-built blockchain solutions that are already integrated with the BNB Smart Chain infrastructure. As a result, developers are granted instant access to a staking system, block explorer, SDK, API gateways, and governance interfaces.

While the current BNB Sidechain implementation is built upon a customised version of BNB Smart Chain, it is important to note that BNB Sidechain can function on top of any blockchain platform.

Modules

BNB Sidechain offers programmable and customisable modules that developers can use or modify to achieve their business objectives, such as:

- Cross Chain — BNB Sidechain enables cross-chain functionality for native assets. As BAS developers manage native assets, they can adjust token supply or mint/burn tokens.
- Staking & Staking Pool — BNB Sidechain supports an on-chain staking system using the PoSA (proof-of-stake-of-authority) model. Users can delegate tokens to specific validators and share rewards based on the total staked amount.

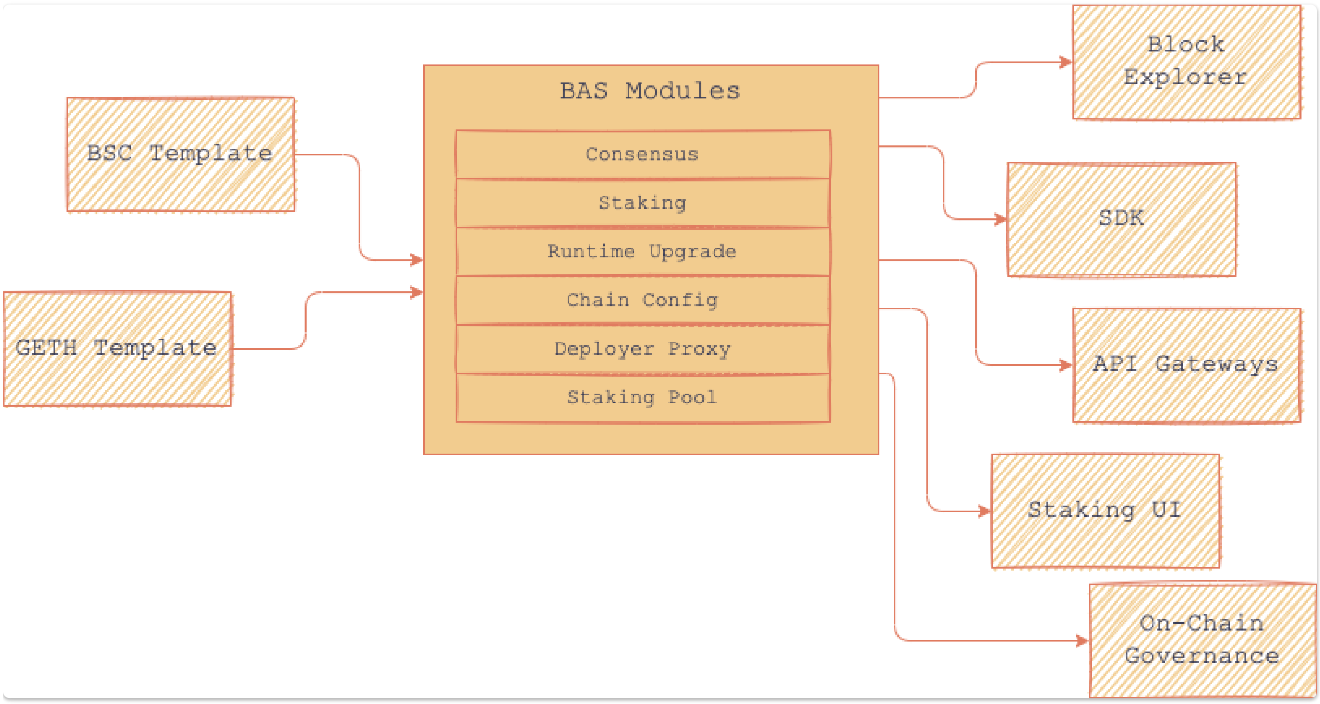
- Runtime Upgrade — The runtime upgrade system smart contract allows modification of existing bytecode for system smart contracts, but not user smart contracts. Users must create proposals for changes, which can only be implemented once a governance quorum is reached. This process is simpler than hard forks, as validators do not need to upgrade their nodes.
- Blockchain & EVM — BNB Sidechain allows for block production and EVM transaction execution, with the potential to define custom runtime execution environments, such as WebAssembly, in the future.
- Web3 API — BNB Sidechain is compatible with the Web3 ecosystem, including MetaMask and other applications.
- Transaction Pool — BNB Sidechain manages transaction filtering and system operation fees through internal policies.

Building a BNB Sidechain with Ankr

From Ankr Documentation

BNB Sidechain specifies the primary structure and configuration of the blockchain, using special templates. A template is a ready-made blockchain solution that is already integrated into the BNB Smart Chain infrastructure. With this integration, developers automatically get access to products like a ready-made staking system, block explorer, SDK, API gateways, interfaces for governance, etc.

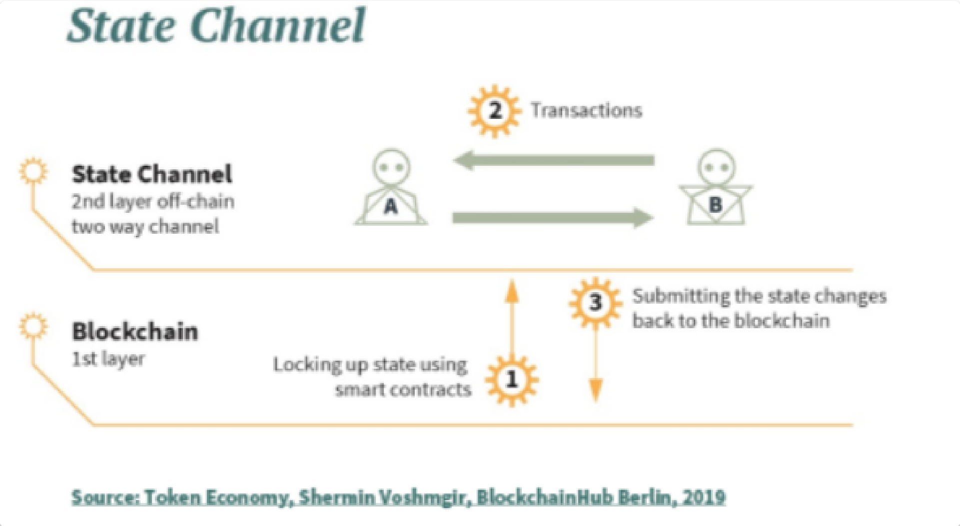
After applying templates, BNB Sidechain can be customized using programmable and configurable modules.



Bridges vs Layer 2

Bridges between chains are typically less functional and more specialised than Layer 2 solutions.
They usually provide the ability to transfer tokens atomically between chains.
One mechanism used for this is 'lock and mint' bridging, in which tokens are locked or burned on the sending chain and an equivalent value of tokens is minted on the receiving chain.
Bridging is quite widespread, for example see the Wormhole projectwhich is used by over 25 chains.

State Channels



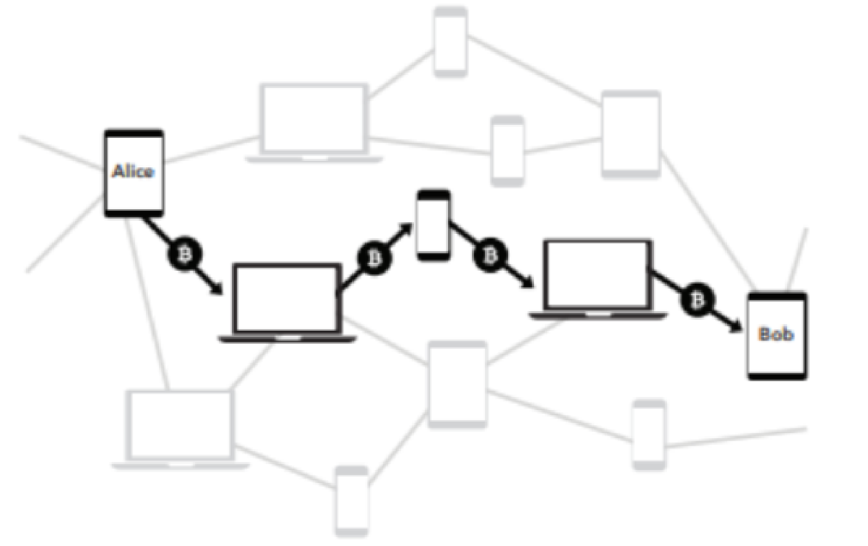
State channels

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.

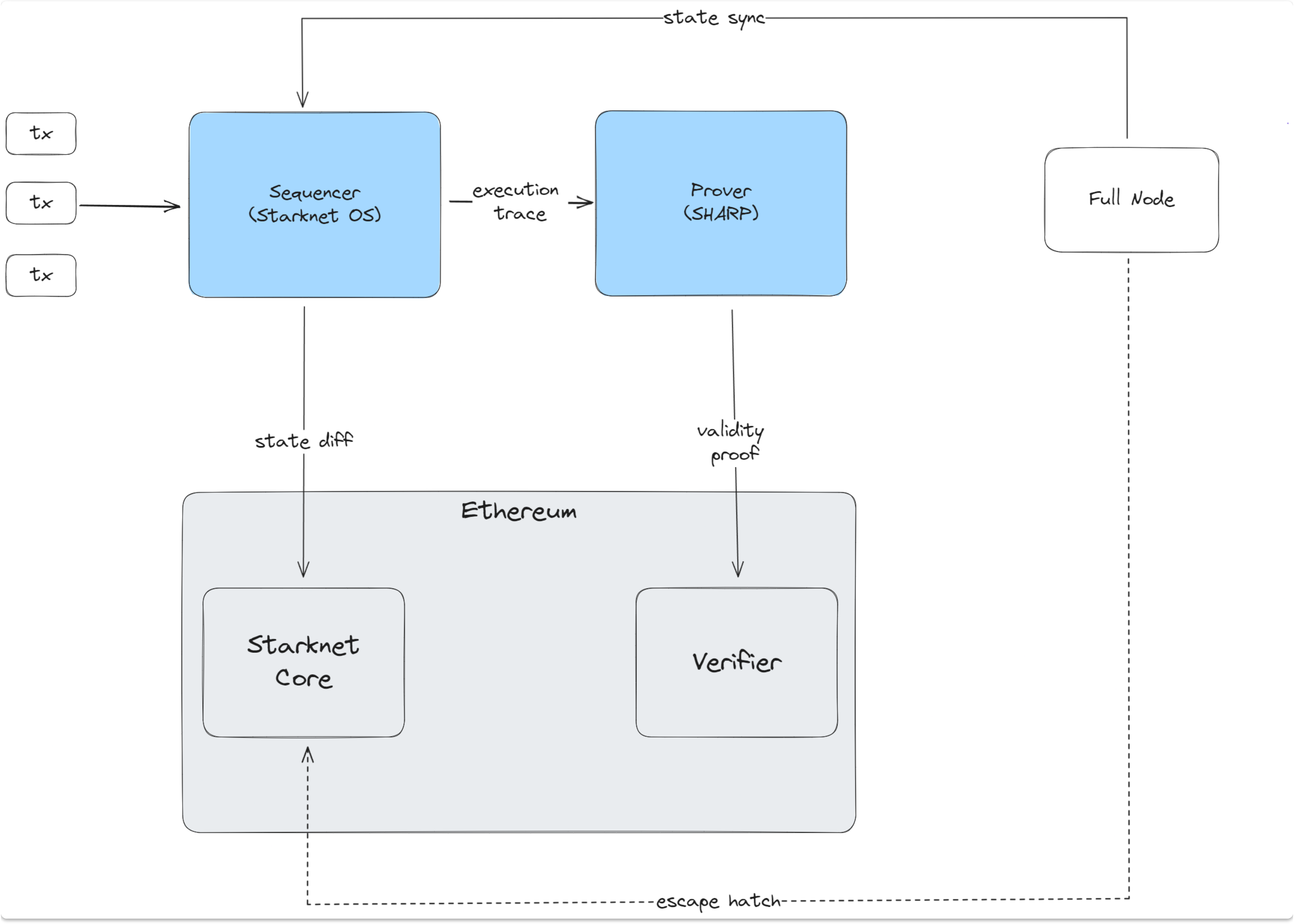
Advantages

- 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.

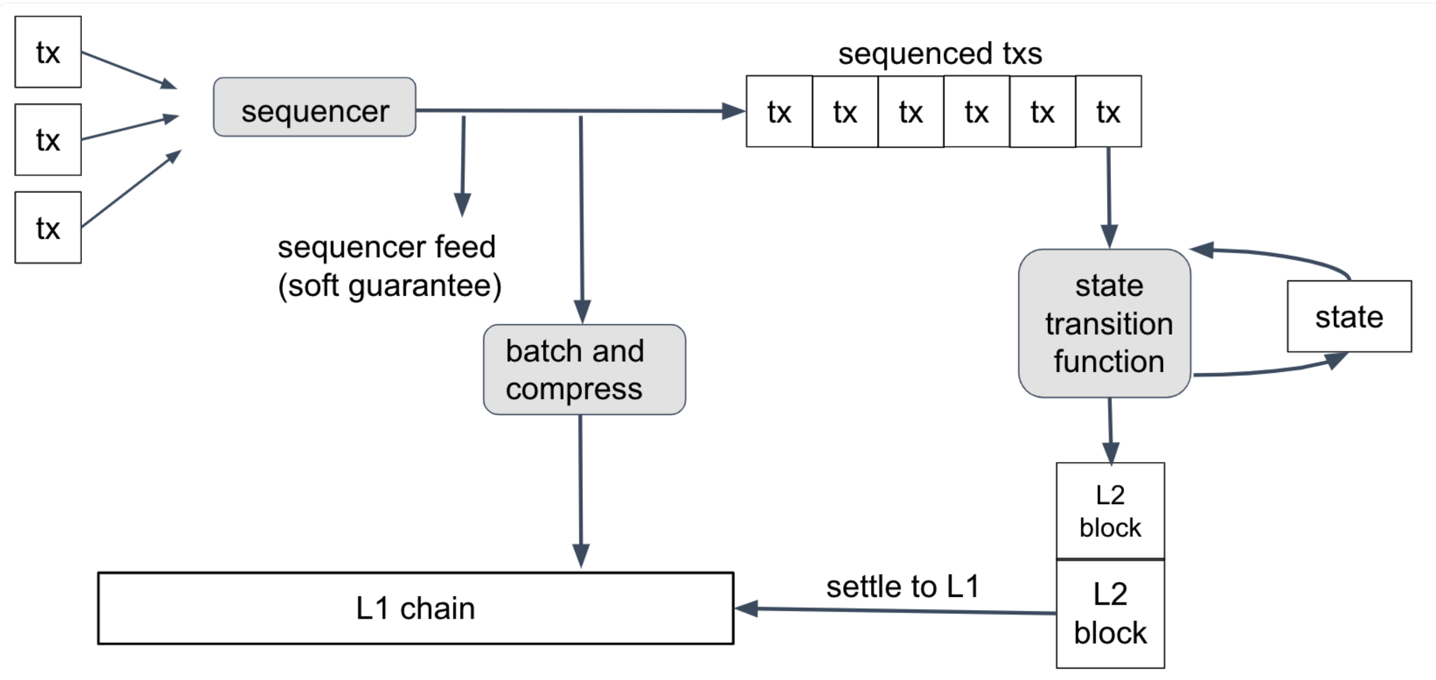
> Lightning enables one to send funds down to 0.00000001 bitcoin without custodial risk. The bitcoin blockchain currently enforces a minimum output size many hundreds of times higher, and a fixed per-transaction fee which makes micropayments impractical. Lightning allows minimal payments denominated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

Typical Rollup Architecture

This is an overview of the Starknet architecture



Here is Arbitrum's design



Plasma Chains

Resources

[Ethereum Docs]
Vitalik [article]

Plasma chains are blockchains anchored to Ethereum's main chain.
They utilise fraud proofs, such as Optimistic rollups, to resolve disputes.
Often labeled as "child" chains, they act as mini versions of the Ethereum Mainnet.
Using Merkle trees, these chains can be stacked indefinitely, helping reduce the load on the parent chains, including the Mainnet.
Their security hinges on fraud proofs, and each has a unique block validation method.

Plasma suggests that Ethereum Mainnet need not validate every transaction. Instead, transactions can be processed off Mainnet, sparing nodes from verifying every single one.

Because they prioritize speed and cost, Plasma chains often use a single "operator" for transaction management.
Having just one entity confirm transactions means Plasma chains generally outpace the Ethereum Mainnet in speed.

This operator, key to producing blocks on a Plasma chain, is obliged to periodically post "state commitments" on Ethereum. These commitments take the form of "Merkle roots", sent regularly to the Plasma contract on Ethereum.

On Ethereum, Plasma operates a master contract that handles user entries/exits, monitors state commitments, and penalises dishonest actions using fraud proofs.

The Mass Exit Problem

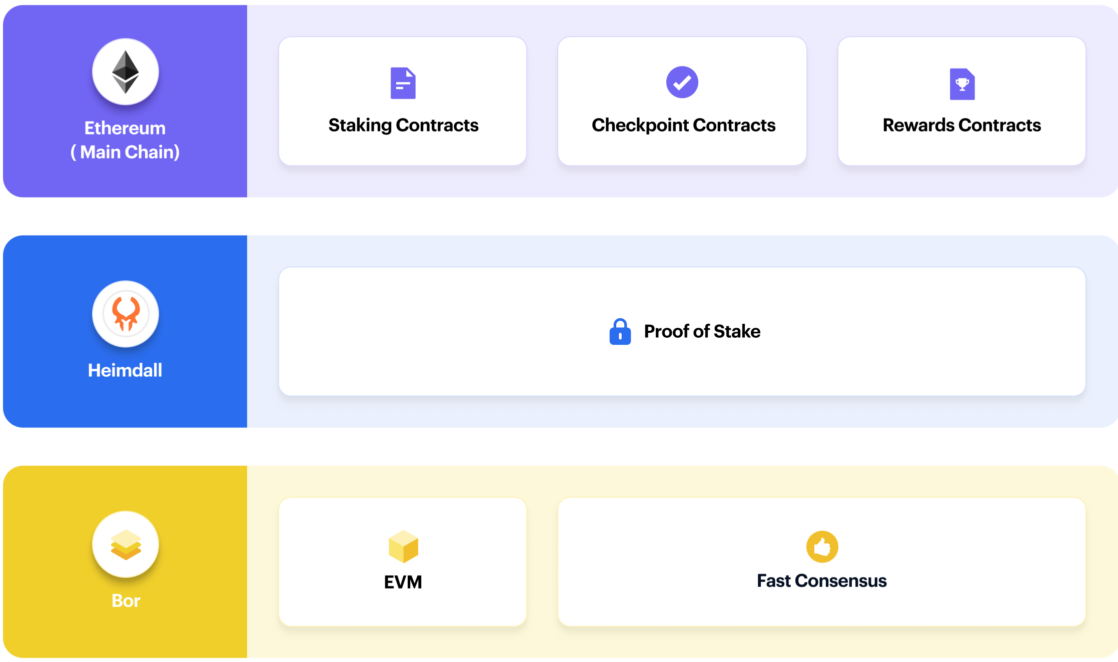
Because Plasma chains can expand without limits and Ethereum blocks are often near their full capacity, transferring the entire content of a Plasma

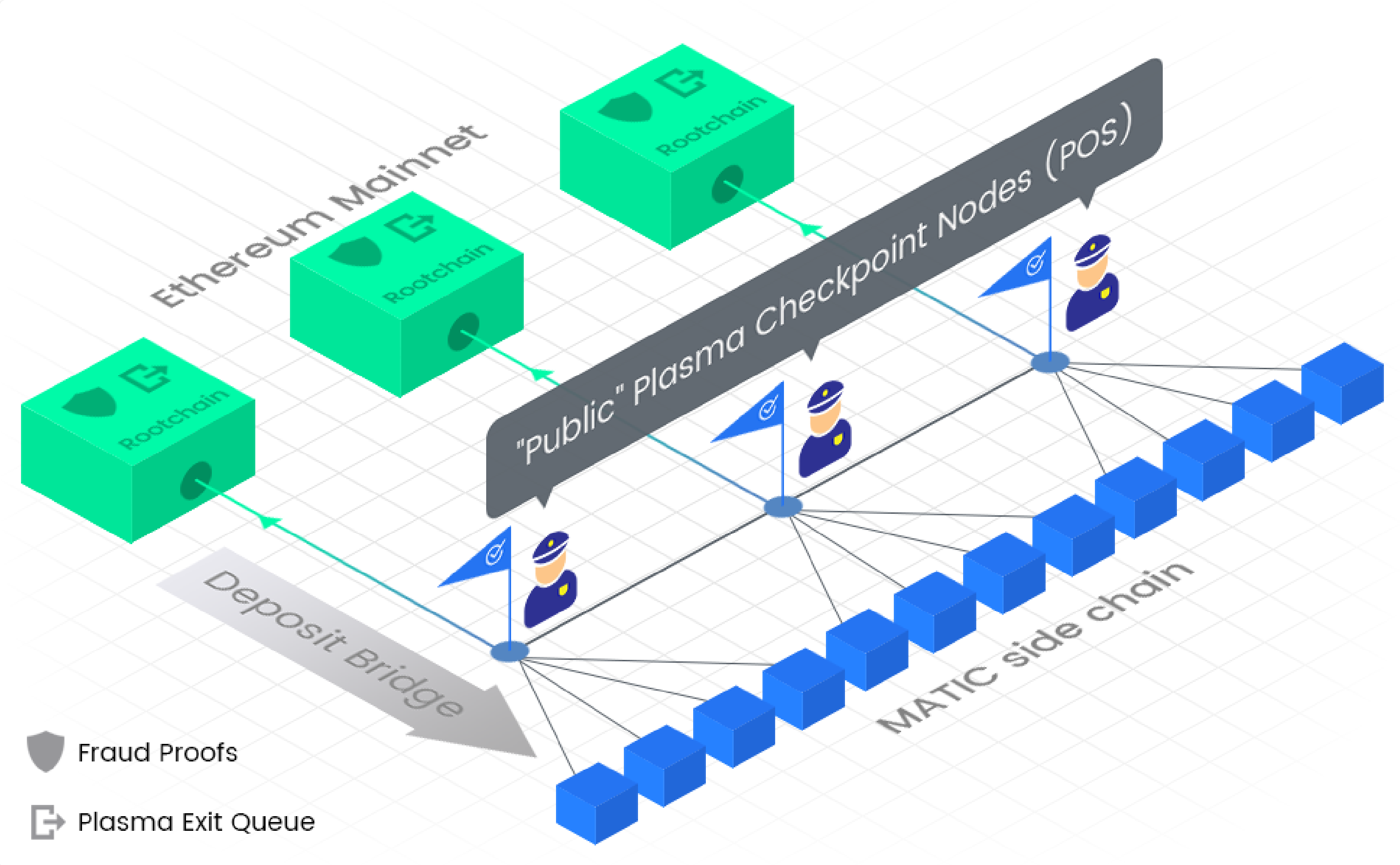
> chain to the Ethereum mainnet would likely be unfeasible. Consequently, a rush to exit could likely cause congestion on Ethereum. This situation is termed the mass exit problem.

Polygon PoS

See Wiki
Architecture Overview
From the wiki :
The Polygon PoS Network has a three-layer architecture:

- Ethereum layer — a set of contracts on the Ethereum mainnet.
- Heimdall layer — a set of proof-of-stake Heimdall nodes running parallel to the Ethereum mainnet, monitoring the set of staking contracts deployed on the Ethereum mainnet and committing the Polygon Network checkpoints to the Ethereum mainnet. Heimdall is based on Tendermint.
- Bor layer — a set of block-producing Bor nodes shuffled by Heimdall nodes. Bor is based on Go Ethereum.

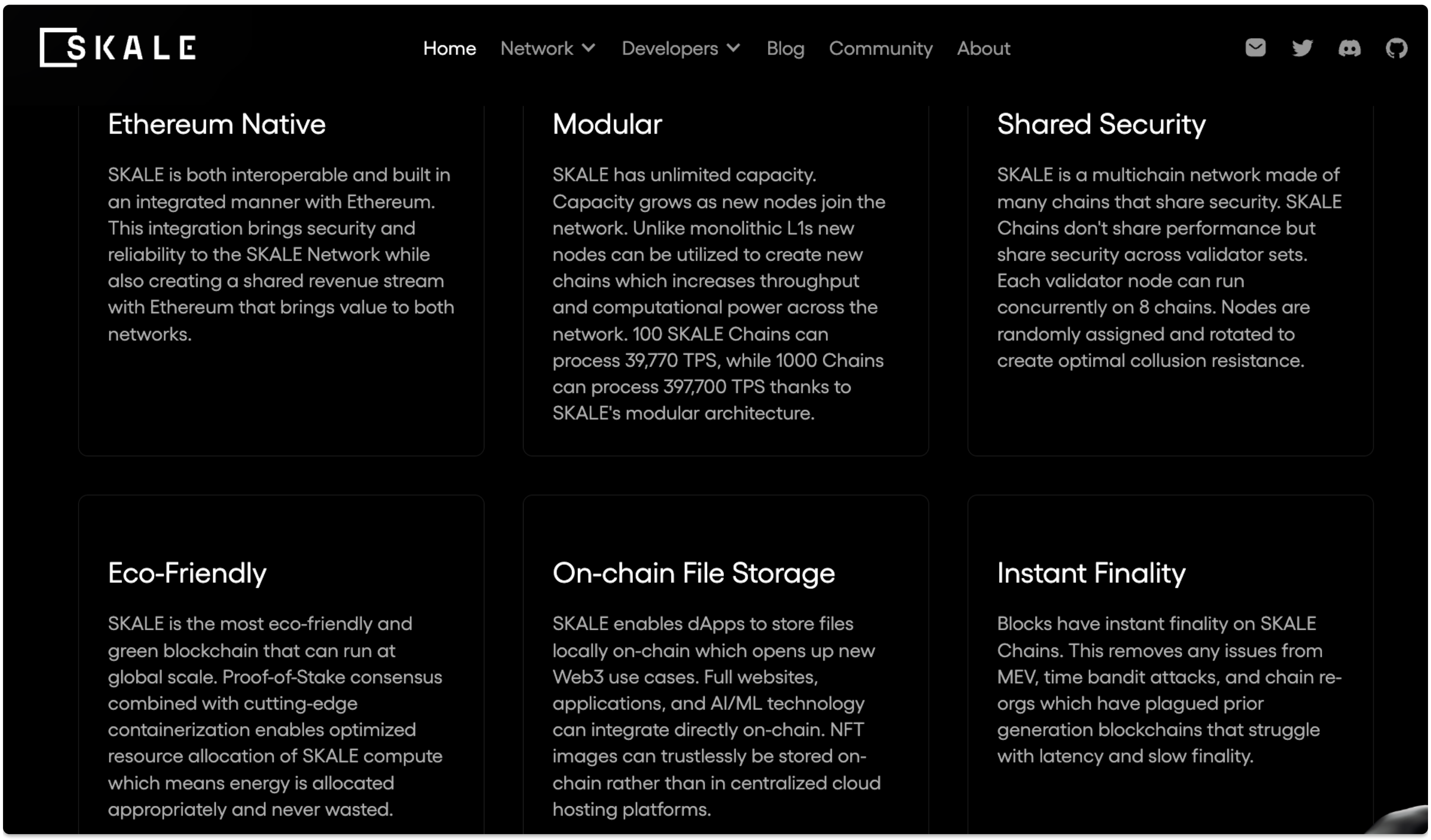


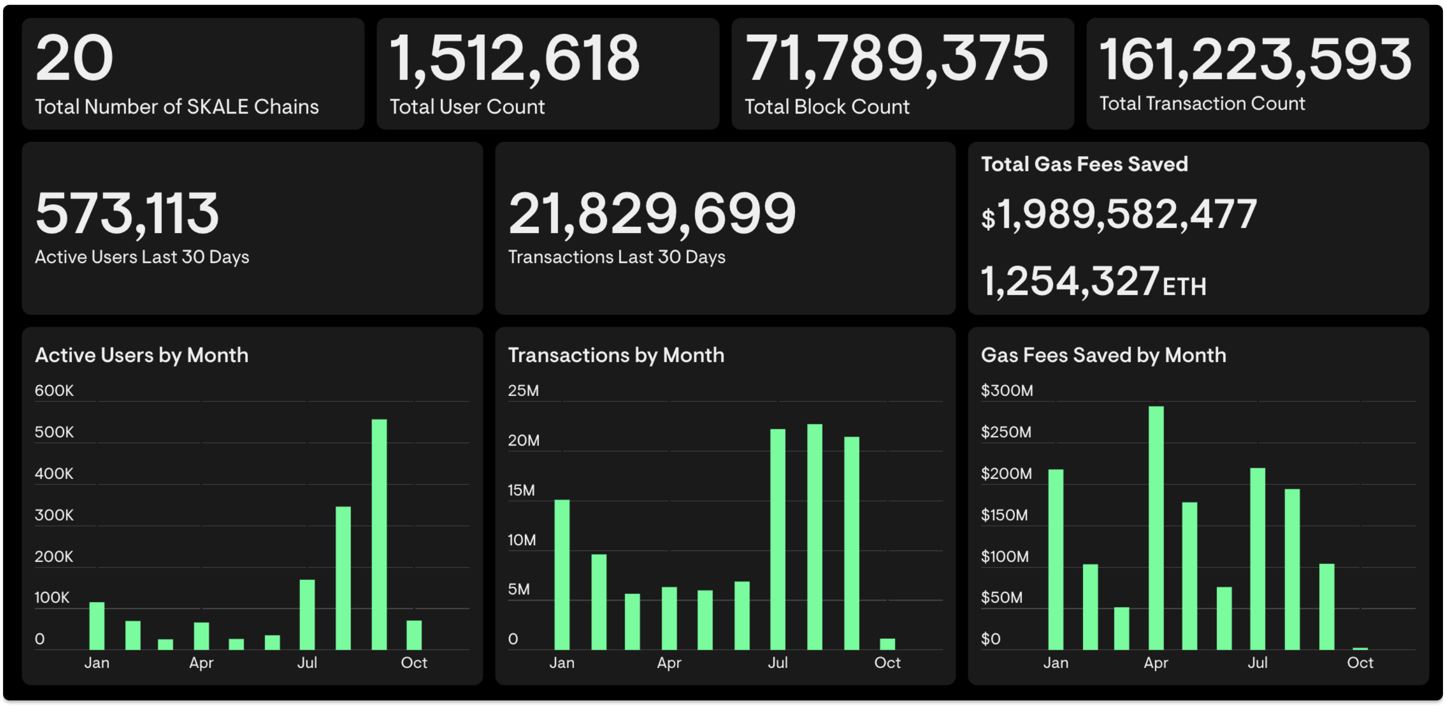


Resources

- Bor Architecture
- Heimdall Architecture
- Checkpoint Mechanism

Skale


SKALE
Home
Network
Developers
Blog
Community
About
Ethereum Native
SKALE is both interoperable and built in an integrated manner with Ethereum. This integration brings security and reliability to the SKALE Network while also creating a shared revenue stream with Ethereum that brings value to both networks.
Modular
SKALE has unlimited capacity. Capacity grows as new nodes join the network. Unlike monolithic L1s new nodes can be utilized to create new chains which increases throughput and computational power across the network. 100 SKALE Chains can process 39,770 TPS, while 1000 Chains can process 397,700 TPS thanks to SKALE's modular architecture.
Shared Security
SKALE is a multichain network made of many chains that share security. SKALE Chains don't share performance but share security across validator sets. Each validator node can run concurrently on 8 chains. Nodes are randomly assigned and rotated to create optimal collusion resistance.
Eco-Friendly
SKALE is the most eco-friendly and green blockchain that can run at global scale. Proof-of-Stake consensus combined with cutting-edge containerization enables optimized resource allocation of SKALE compute which means energy is allocated appropriately and never wasted.
On-chain File Storage
SKALE enables dApps to store files locally on-chain which opens up new Web3 use cases. Full websites, applications, and AI/ML technology can integrate directly on-chain. NFT images can trustlessly be stored on-chain rather than in centralized cloud hosting platforms.
Instant Finality
Blocks have instant finality on SKALE Chains. This removes any issues from MEV, time bandit attacks, and chain re-orgs which have plagued prior generation blockchains that struggle with latency and slow finality.


20
Total Number of SKALE Chains
1,512,618
Total User Count
71,789,375
Total Block Count
161,223,593
Total Transaction Count
573,113
Active Users Last 30 Days
21,829,699
Transactions Last 30 Days
Total Gas Fees Saved
$1,989,582,477
1,254,327 ETH
Active Users by Month
600K
500K
400K
300K
200K
100K
0
Jan
Apr
Jul
Oct
Transactions by Month
25M
20M
15M
10M
5M
0
Jan
Apr
Jul
Oct
Gas Fees Saved by Month
$300M
$250M
$200M
$150M
$100M
$50M
$0
Jan
Apr
Jul
Oct

How SKALE Works

Harmonizing Speed, Security, and Decentralization

SKALE's advanced cryptography and pooled security model enables speed and decentralization without sacrificing security, allowing developers to deliver an exceptional experience to end-users free of any gas fees or latency.

Virtualized subnodes & fully decentralized blockchain

Ethereum-native chain orchestration method

10M+ transactions per day per chain

Efficient & high-performing pooled validation model

Secured by random & regular node rotation

Deploy Ethereum-based smart contracts cost-effectively

The Tech Behind SKALE

Leaderless BFT Consensus

The consensus model used for block creation and commitment for each chain is a variant of the Asynchronous Binary Byzantine Agreement (ABBA) protocol, limiting subnode downtime by identifying slow links.

Interchain Messaging: BLS Threshold Signatures

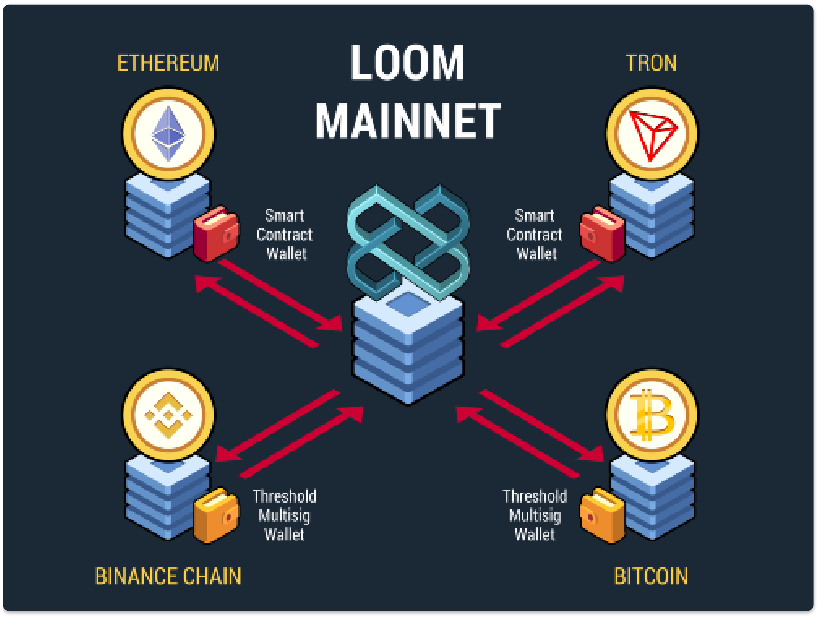
Virtualized subnodes for each chain can validate a transaction signed and committed by the subnodes in another chain using its group signature once it's published to the Ethereum mainnet.

Node Monitoring

A Node Monitoring Service (NMS) runs on each SKALE Node and facilitates the performance tracking of a certain number of other nodes in the network. Performance tracking measures uptime and latency through a regular process that pings peer nodes and logs the measurements.

Loom

See Site



> Loom Network is a multichain interop platform live in production since early 2018. Optimized for scaling high-performance dapps that require a fast and smooth user experience, the network allows dapps to offer a UX comparable to traditional applications and onboard new users without the friction of needing to download crypto wallet software.
>
> Loom also has integrations to Bitcoin, Ethereum, Binance Chain, and Tron (with EOS and Cosmos coming soon). This allows developers to integrate assets from all major chains, as well as build a dapp only once and offer it to users on all platforms simultaneously.

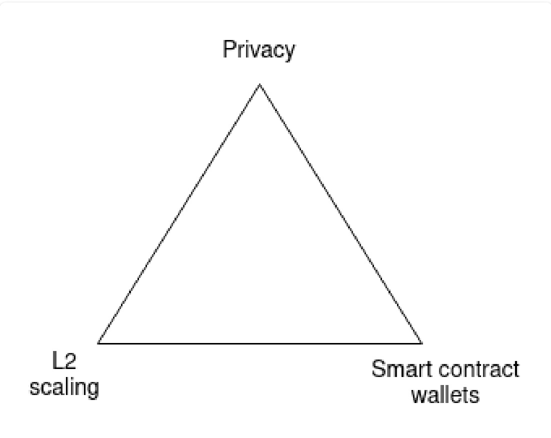
Ethereum Scalability directions

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.