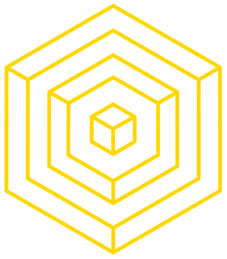
Lecture 09:
Scaling Blockchains

Akash Khosla

ethereum

BLOCKCHAIN
AT BERKELEY



LECTURE OUTLINE

1. CURRENT STATE OF BLOCKCHAINS
2. SCALING WITH LAYER 1
3. STATE CHANNELS AND PLASMA
4. MINIMUM VIABLE PLASMA

1 THE STATUS QUO

Highly scalable blockchain based payment network using Plasma

Author by Joseph Poon and Vitalik Buterin

omisego

LOOKING AT SCALING

SCALING

- **Effective throughput** <= maximum block size / block interval
- block interval := time it takes to confirm a block
- Can we reach the performance of a mainstream payment processor?
- For Bitcoin especially, the community has put forth various proposals to modify the system parameters of block size and block intervals (**reparameterization**)
- If the transaction rate exceeds more than 90% effective throughput, then 10% of the nodes in the network would be unable to keep up, potentially resulting in denied services to users along with the network's mining power, and overall decentralization

LOOKING AT SCALING

SCALING

- Block size should not exceed 4MB given today's 10 minute average block interval at 27 txn/s (**Throughput Limit**)
- The block interval should not be smaller than 12s if full utilization of the network's bandwidth is to be achieved (**Latency Limit**)
- Upper bounds on scaling proposed by researchers

BITCOIN'S REALITY CHECK

SCALING

- **Max throughput** is 3.3 - 7 transactions/sec
    - Constrained by maximum block size and inter block time
- **Latency** - Time for a transaction to confirm, a transaction is confirmed when it is included in a block, roughly 10 minutes in expectation
- **Bootstrap time** - the time is takes a new node to download and process the history necessary to validate the current system state. Currently takes **4+ days** on average
- **Cost per Confirmed Transaction (CPCT):** Cost in USD of resources consumed by the entire Bitcoin system to confirm a single transaction

COST PER CONFIRMED TRANSACTION

WHAT COMPOSES IT?

- **Mining**: Expended by miners in generating the proof of work for each block
- **Transaction validation**: The cost of computation necessary to validate that a transaction can spend the outputs referenced by its inputs, dominated by cryptographic verifications
- **Bandwidth**: The cost of network resources required to receive and transmit transactions, blocks, and metadata
- **Storage**: The cost of storing all currently spendable transactions, which is necessary for miners and full nodes to perform transaction validation, and of storing the blockchain's (much larger) historical data, which is necessary to bootstrap new nodes that join the network

COST PER CONFIRMED TRANSACTION

WHAT COMPOSES IT?

	at max throughput		at *de facto* throughput	
	cost/tx	percentage	cost/tx	percentage
Mining: proof-of-work	~ \$0.8 – \$1.7	~ 56%	~ \$3.6	~ 56%
Mining: hardware	~ \$0.6 – \$1.3	~ 42%	~ \$2.7	~ 42%
Transaction validation	~ \$0.002	~ 0.2%	~ \$0.008	~ 0.2%
Bandwidth	~ \$0.02	~ 2%	~ \$0.08	~ 2%
Storage (running cost)	~ \$0.0008/5years			

Table 1: **Bitcoin cost breakdown.** Includes cost incurred by all nodes.

Source

DOES ETHEREUM FACE THE SAME PROBLEM?

SHORT ANSWER: YES

COURTESY CRYPTOKITTIES

DOES ETHEREUM FACE THE SAME PROBLEM?

SHORT ANSWER: YES

- Ethereum is stuck at 15 TPS
- Popular DApps like Cryptokitties and the occasional ICO slow down the network and cause gas prices to increase
- The core limitation is that public blockchains like ethereum require every transaction to be processed by every single node in the network
    - This is by design—it's part of what makes public blockchains authoritative. Nodes don't have to rely on someone else to tell them what the current state of the blockchain is—they figure it out for themselves
- This puts a fundamental limit on ethereum's **transaction throughput**: it cannot be higher than what we are willing to require from an individual node

1.1 ADJUSTING THE STATUS QUO

SCALING WORKAROUND?

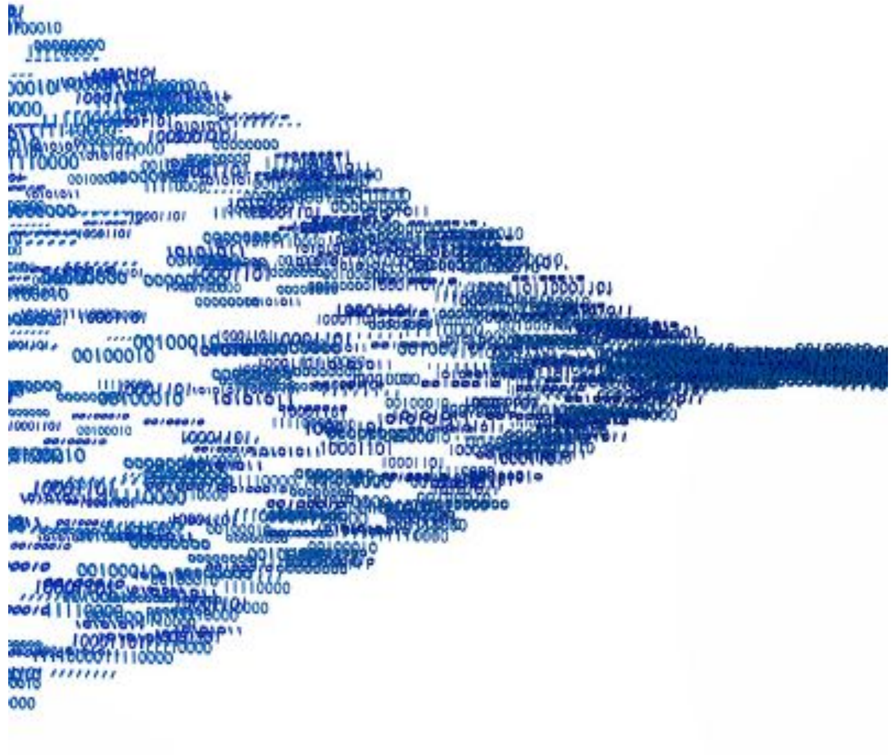
SCALING ETHEREUM

- We could ask every individual node to do more work
- If we doubled the block size (i.e., the block gas limit), it would mean that each node is doing roughly double the amount of work processing each block
- But this comes at the cost of decentralization: requiring more work from nodes means that less powerful computers (like consumer devices) may drop out of the network, and mining becomes more centralized in powerful node operators
  - Defeats the purpose of blockchains if we get all centralized
  - Instead, we need a way for blockchains to do more useful stuff without increasing the workload on individual nodes

NETWORK SCALING

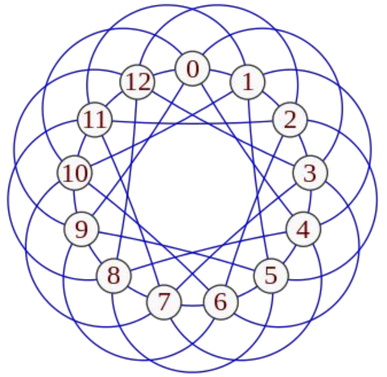
BOTTLENECKS & TRADEOFFS

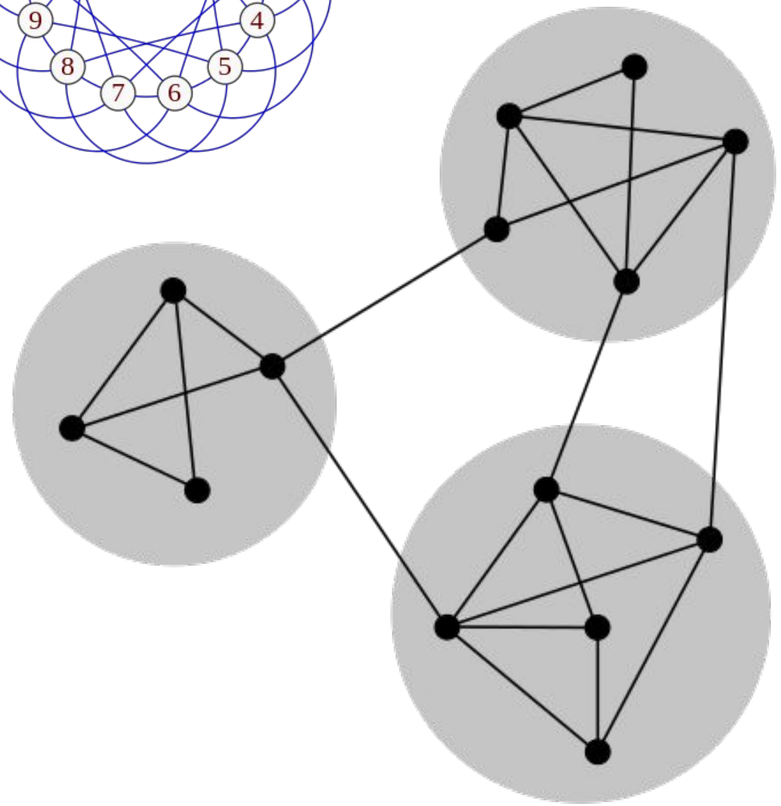
- Generally, BTC's network is bottlenecked at the packet transmission level (ignoring consensus-based drawbacks)
  - Sending transactions twice: once for the originating tx and one for when its included in a block
    - tx propagation localized but block propagation remains global
    - Can potentially fix this by having miners get tx's they don't have from new blocks and reconcile them with their own set
  - There is also the need to validate every transaction that passes through each node



NETWORK SCALING

MORE EFFECTIVE NETWORK TOPOLOGIES





- Dynamic P2P systems are difficult to build for
  - Most static networks use **expander graph** models
  - These are graphs with a high amount of neighbors per subset
- Low diameters between nodes means reduced transmission times between neighbors
  - May reduce the need to revalidate every tx if neighbors are trusted
- Options to **randomize peer selection** as a defense mechanism against DoS and Eclipse attacks
- **Rate-limiting** also an option

NETWORK SCALING

LACKING INCENTIVE LAYER

- Fundamental problem:
  - Nobody is incentivized to propagate blocks
  - Mostly relies on voluntary participation
  - No disincentives against continual propagation leading to flooding and DoS attacks

CONSENSUS SCALING

RECAP

- The role of consensus is to designate a globally accepted set of transactions for processing, as well as a total or partial order on these transactions
- This plane ingests messages from the Network Plane and outputs transactions for insertion into the system ledger
- In BTC, the Consensus Plane is the functionality that mines blocks and reaches consensus on their integration into the blockchain

More reading here

CONSENSUS SCALING

POW & POS

- "Artificially" bottlenecked; the network plane is the physical bottleneck
- Basic tradeoffs: tx latency and throughput vs. computation, bandwidth, storage
  - Stronger trust assumptions built in at the consensus protocol level mean that the network can achieve more efficient tradeoffs.
- **Proof of Work**
  - GHOST protocol (or similar fork reconciliation solutions) can be used to tackle security/fairness issues that arise from block-generation speed (speed you reach consensus)
- **Proof of Stake**
  - Very few convergence guarantees

CONSENSUS SCALING

CONSORTIUM CONSENSUS & SHARDING

- **Consortium Consensus**
  - BFT protocol executed by small group of trusted entities (consortium blockchains) Distributing trust naturally incur performance costs
  - Way better than BTC but would still suffer at scale without sharding
- **Sharding**
  - "Split up the task of consensus among ... nodes"
  - Reduced per-node storage and computing
  - Used with centralized databases too! (MongoDB, MySQL, etc.)
  - The issue is that these are centralized models; in blockchains operations may span multiple shards
  - Cross-shard consensus has been proposed to be solved by separate consensus (e.g. Paxos) but the overhead incurred is huge

CONSENSUS SCALING

SIDECHAINS

- **Delegation of trust and hierarchy of sidechains**
- Hierarchy of low-tier consensus instances (sidechains) with a lower degree of decentralization

  Individual side chains must be independent of main blockchain
  - This is one of the differences with sharding, as shards generally are extensions to the main chain
  - Can be federated as long as there is trust in the federation
  - Merged mining (incentive model for miners to do work on multiple chains) requires miner coordination
  - Some chains may become more dominant than others
  - Blind anchoring also an option to replace BTC checkpoints and increase security

CONSENSUS SCALING

SIDECHAINS (CONT.)

- **Delegation of trust and hierarchy of sidechains**
- Increased burden of redundant tx
  - tx's that originate at side chain A and have a destination in side chain B must travel through the main chain
  - This means that two tx's are necessary
- Confirmation times are longer
  - Now every new block require a sequence of block confirmations across chains to ensure block finality

STORAGE SCALING

THE PROBLEM

- Global read/write ledger for the network
  - Ingests operations from the consensus layer
- Also stores state structures (e.g. smart contracts, etc.)
- BTC's storage mechanism does not allow deletes and "reading" requires the download of the entire chain (> 4 days)
  - Highly inefficient as it requires redundant storage of the entire chain across all nodes

STORAGE SCALING

THE SOLUTIONS

- Alternative storage models involve UTXO data sharding as an interface for storage
  - Schemes like sending parent tx's with their Merkle branches
    - The UTXO needs to be constructed here in such a way that the information provided with transactions is always enough to update the new committed UTXO hash
    - Enables a bandwidth/storage tradeoff. E.g. if you're willing to store all the data yourself, you could tell your peers not to send you all the proof data which you can simply construct on your own
    - **THIS SCHEME IS NOT SAFE (DOUBLE-SPENDABLE)**
- Other proposals involve cryptographic accumulators and distributed hash tables as a means of not necessitating the storage of the entire state

VIEW SCALING

WHAT IS A VIEW?

- The interface that abstracts away from the chain's entire state and only focuses on relevant information
  - For example, the UTXO set in BTC as opposed to every tx
  - For example, contracts involved in new tx in ETH as opposed to every contract
- As an optimization, view may be stored in storage
  - BTC doesn't use this which is why miners need to store the entire ledger to be able to build this view from scratch

VIEW SCALING

INNOVATIONS ON THE VIEW

- Views are determined by the consensus plane and authenticable by anyone
  - **Views via Replication** is the standard practice. All consensus nodes verify all tx's and update their views based on the computation (transition function) introduced
  - **Outsourcing views via cryptography.** nodes can send their updates to a third-party provider. This provider then provides the base "truth" of the update via a cryptographic digest (e.g. Merkle root) and a proof of its correctness (e.g. using verifiable computation techniques like SNARKS)
    - Nodes can store these or simply allow them to persist with the provider which saves on space in storage

2 SHARDING

REMOVING REDUNDANT EXECUTIONS

SCALING ETHEREUM

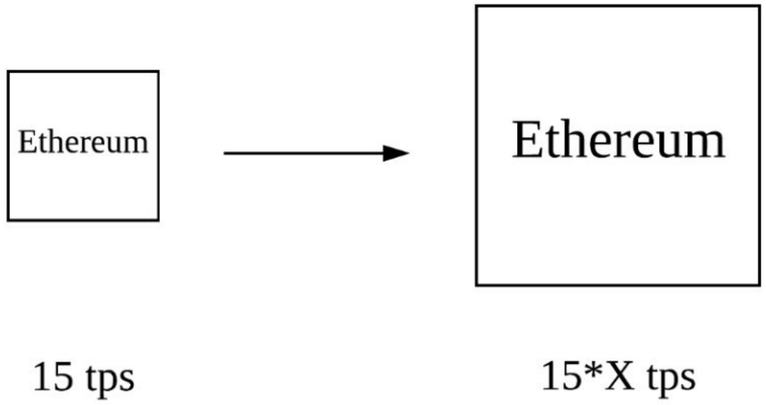
- What if we could build a blockchain where every node didn't have to process every operation?
- What if, instead, the **network was divided into two sections**, which could operate **semi-independently**?
  - Section A could process one batch, while Section B does another batch of transactions
- Txn throughput can be **doubled**, and if we divide this up into more sections, then we can increase the throughput of a blockchain by many multiples
- **Sharding**: Split a blockchain up into multiple sections where each section can independently process transactions

AUTHOR: AKASH KHOSLA
SOURCE

LAYER 1 SCALING

SCALING ETHEREUM

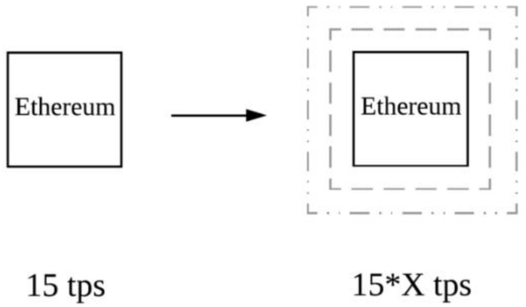
- Layer 1 is the base consensus layer of the Ethereum protocol
- Sharding is Layer 1
- Layer 1 solutions increase Ethereum's txn throughput by increasing the capacity of the base blockchain
- Usually requires a hard fork to implement



LAYER 2 SCALING

SCALING ETHEREUM

- What if we could do more things with the capacity we already have?
- The throughput of the base layer (the main chain) remains the same, but would like to be able to do more operations that are useful to people and applications, like txns, state updates between fixed or changing parties or computations
- Off chain technologies like state channels, plasma and truebit guarantee a sufficient level of security and finality
- They are Layer 2 because they are built on top of the Ethereum main chain, essentially running as smart contracts that interact with off-chain software (therefore no hard fork required)



AUTHOR: AKASH KHOSLA

SOURCE

3

LAYER 2 SCALING

L2 SOLUTIONS ARE CRYPTOECONOMIC

SCALING

- Much of the power in public blockchains relies on **cryptoeconomic consensus**
- By carefully aligning incentives and securing them with software and cryptography, we can create networks of computers that reliably come to agreement about the internal state of a system
- Cryptoeconomic consensus gives us a **strong security guarantee (core kernel of certainty)** of the main chain
    - Unless something extreme like a 51% attack happens, we know that on-chain operations—like payments, or smart-contracts—will execute as written

AUTHOR: AKASH KHOSLA
SOURCE

L2 SOLUTIONS ARE CRYPTOECONOMIC

SCALING

- The insight behind layer 2 solutions is that we can use this core kernel of certainty as an **anchor** for ultimate finality
    - **Anchor**: a fixed point to which we attach additional economic mechanisms.
- Lets us have interactions off of the blockchain that can still reliably refer back to that core kernel if necessary
- These layers built “on top” of ethereum won’t always have the same guarantee as on-chain operations
    - But they can still be sufficiently final and secure to be very useful— slight decrease in finality lets us perform operations much faster or with lower overhead costs

AUTHOR: AKASH KHOSLA
SOURCE

3.1 GENERAL STATE CHANNELS

STATE CHANNELS

OFF CHAIN SOLUTIONS

- Technique for performing transactions and other state updates “off chain”
- Things that happen inside still retain a strong degree of finality and security
- If things go wrong, we have the option of referring back to the anchor in the main chain
- Payment channels have been around for a while, specifically on the lightning network
- State channels are generalized payment channels used for any arbitrary state update on the blockchain like variables inside a smart contract

AUTHOR: AKASH KHOSLA

SOURCE

STATE CHANNELS

TIC TAC TOE

- Imagine that Alice and Bob want to play a game of tic tac toe, where the winner recieves 1 eth
- Naive implementation:
  - Create a smart contract on ethereum that implements the rules of tic tac toe and keeps track of each players' moves.
  - Every time a player wants to make a move, they send a transaction to the contract
  - When one player wins, as determined by the rules, the contract pays out the 1 eth to the winner
- The entire Ethereum network is processing their game which is overkill for what they need
- They have to pay gas costs every time a player wants to make a move and wait for blocks to be mined before making the next move

STATE CHANNELS

LETS TRY DOING THINGS OFF CHAIN

- We create a smart contract **“Judge”** on the ethereum main-chain that understands the rules of tic-tac-toe, can identify Alice and Bob as the two players in our game and holds the 1 eth prize
- Alice and Bob begin playing the game
  - Alice creates and signs a transaction describing her first move, and sends it to Bob, who also signs it, sends back the signed version, and keeps a copy for himself.
  - Then Bob creates and signs a transaction describing his first move, and sends it to Alice, who also signs it, sends it back, and keeps a copy.
- **Each time, they are updating the current state of the game between them**
  - Each transaction contains a “nonce”, which simply means that we can always tell later in what order the moves happened

AUTHOR: AKASH KHOSLA

SOURCE

STATE CHANNELS

LETS TRY DOING THINGS OFF CHAIN

- Alice and Bob are simply sending transactions to each other over the internet, but **nothing is hitting the blockchain yet**
- However, all of the transactions could be sent to the Judge contract—in other words, they are valid ethereum transactions.
  - You can picture this as two people writing a series of blockchain-certified cheques back and forth to each other
- **No money has actually been deposited or withdrawn from a bank, but each has a stack of cheques that they could deposit whenever they want**

AUTHOR: AKASH KHOSLA
SOURCE

STATE CHANNELS

LETS TRY DOING THINGS OFF CHAIN

- When Alice and Bob are done playing the game—perhaps because Alice has won—they can close the channel by submitting the final state (e.g. a list of transactions) to the Judge contract, paying only a single transaction fee
- The Judge ensures that this “final state” is signed by both parties, and **waits a period of time to ensure that no one can legitimately challenge the result**, and then pays out the 1 eth award to Alice

AUTHOR: AKASH KHOSLA
SOURCE

STATE CHANNELS

CHALLENGE PERIOD

- Why do we need this “challenge period” where the Judge contract waits?
- Imagine that instead of sending the real final state to the Judge, Bob sent an old version of the state—one where he was winning ahead of Alice
  - The Judge is just a dumb contract—on its own, it has no way of knowing whether this is the most recent state or not
  - The challenge period gives Alice a chance to prove to the Judge contract that Bob has lied about the final state of the game
  - If there is a more recent state, then she will have a copy of the signed transactions, and she can submit those to the Judge. The Judge can tell that Alice’s version is more recent by checking the nonce, and Bob’s attempt to steal the win is rejected

AUTHOR: AKASH KHOSLA
SOURCE

STATE CHANNELS

FEATURES AND LIMITATIONS

- **State channels rely on availability**
  - If Alice lost her internet connection during a challenge she might not be able to respond before the challenge period ends
  - However, Alice can pay someone else to keep a copy of her state and maintain availability on her behalf
- **Useful when participants are going to be exchanging many state updates over a long period of time**
  - This is because there is an initial cost to creating a channel in deploying the Judge contract. But once it is deployed, the cost per state update inside that channel is extremely low

STATE CHANNELS

FEATURES AND LIMITATIONS

- **State channels are best used for applications with a defined set of participants**
  - This is because the Judge contract must always know the entities (i.e. addresses) that are part of a given channel
  - We can add and remove people, but it requires a change to the contract each time
- **State channels have strong privacy properties**, because everything is happening “inside” a channel between participants, rather than broadcast publicly and recorded on-chain
  - Only the opening and closing transactions must be public
- **State channels have instant finality**, meaning that as soon as both parties sign a state update, it can be considered final
  - Both parties have a very high guarantee that they can “enforce” that state on-chain

STATE CHANNELS

IMPLEMENTATIONS TO LOOK OUT FOR

- Counterfactual
- Raiden (network of payment channels)
- Funfair
- Spankhain

AUTHOR: AKASH KHOSLA

SOURCE

3.2 PLASMA

PLASMA

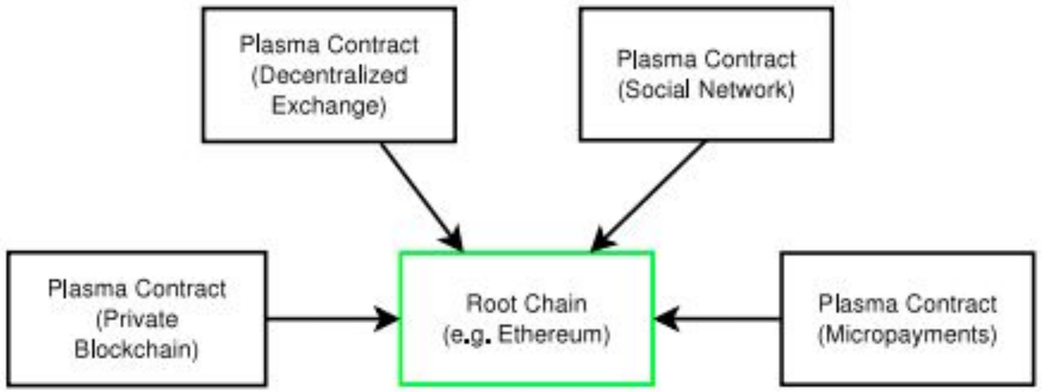
AUTONOMOUS SMART CONTRACTS

- Like state channels, **Plasma** is a technique for conducting off-chain transactions while relying on the underlying ethereum blockchain to ground its security
- But Plasma takes the idea in a new direction, by allowing for the creation of “child” blockchains attached to the “main” ethereum blockchain
- These child-chains can, in turn, spawn their own child-chains, who can spawn their own child-chains, and so on
- If you’re familiar with sidechains, this is an example of a an implementation
- As a result, we can perform complex operations at the child chain, i.e. running applications with thousands of users with minimal interaction of the main chain

AUTHOR: AKASH KHOSLA
SOURCE

PLASMA
EXAMPLE

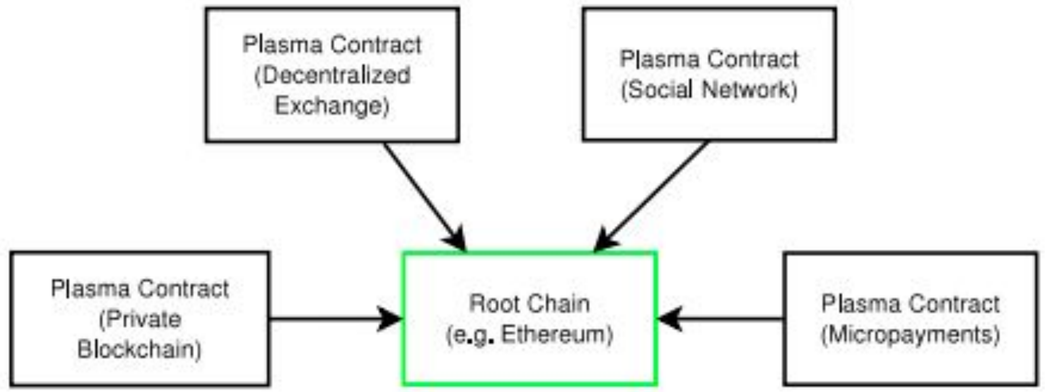
- Let’s imagine that you’re creating a trading-card game on ethereum
- The cards will be ERC 721 non-fungible tokens (like Cryptokitties), but have certain features and attributes that lets users play against each other—like in Hearthstone, or Magic the Gathering
- These kinds of complex operations are expensive to do on-chain, so you decide to use Plasma instead for your application



AUTHOR: AKASH KHOSLA
SOURCE

PLASMA
WALKTHROUGH

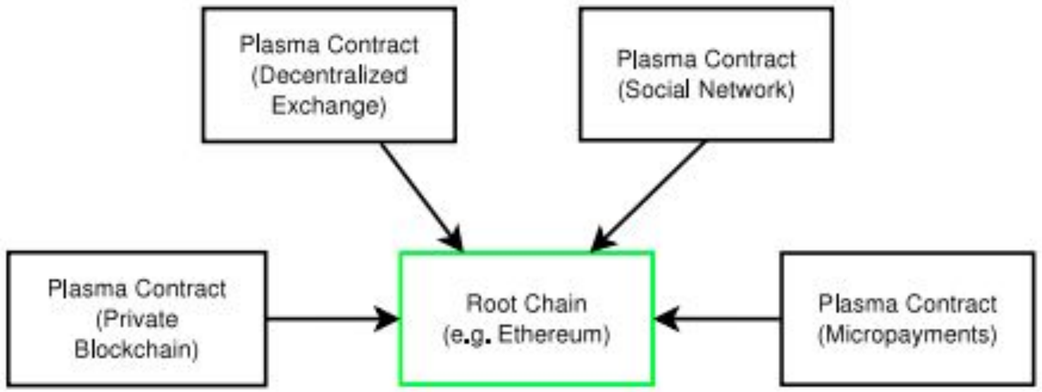
- First, we create a set of smart-contracts on ethereum main-chain that serve as the “Root” of our Plasma child-chain.
- The Plasma root:
  - contains the basic “state-transition rules” of our child chain (things like “transactions cannot spend assets that have already been spent”)
  - records hashes of the child-chain’s state
  - serves as a kind of “bridge” that lets users move assets between the ethereum main-chain and the child-chain



AUTHOR: AKASH KHOSLA
SOURCE

PLASMA
WALKTHROUGH

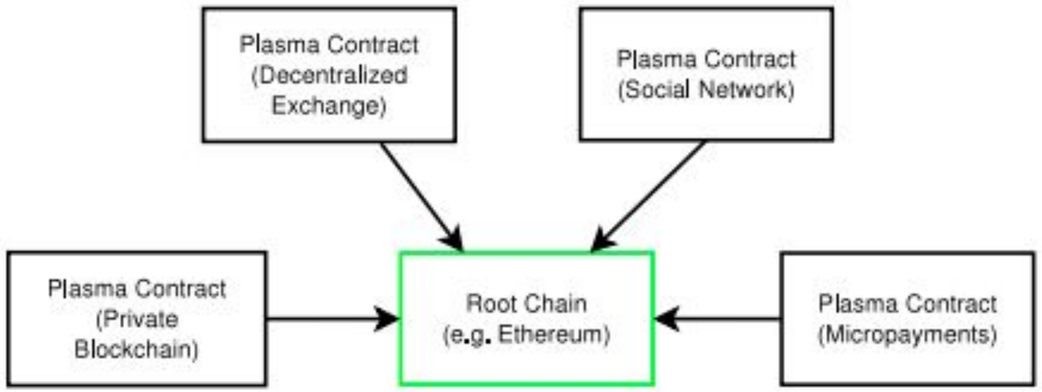
- Then, we create our child-chain
  - can have its own consensus algorithm — let's say that it uses Proof of Authority (PoA), a simple consensus mechanism that relies on trusted block producers (i.e. validators).
  - Let's keep our example simple, and say that you (the company that created the game) are the only entity that is creating blocks—i.e. your company runs a few nodes that are the block producers for our child-chain.



AUTHOR: AKASH KHOSLA
SOURCE

PLASMA
WALKTHROUGH

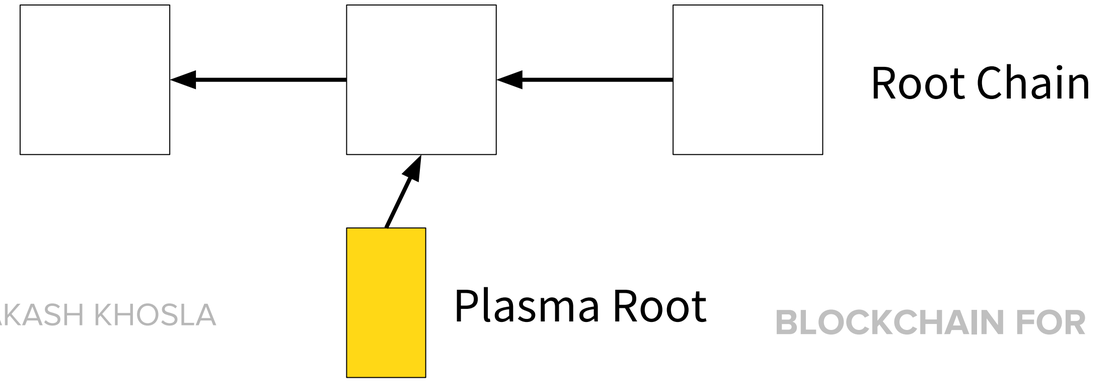
- Once the child-chain is created and active, the block producers make periodic commitments to the root contract
  - This means they are effectively saying “I commit that the most recent block in the child-chain is X”
  - These commitments are recorded on-chain in the Plasma root as a proof of what has happened in the child-chain.



AUTHOR: AKASH KHOSLA
SOURCE

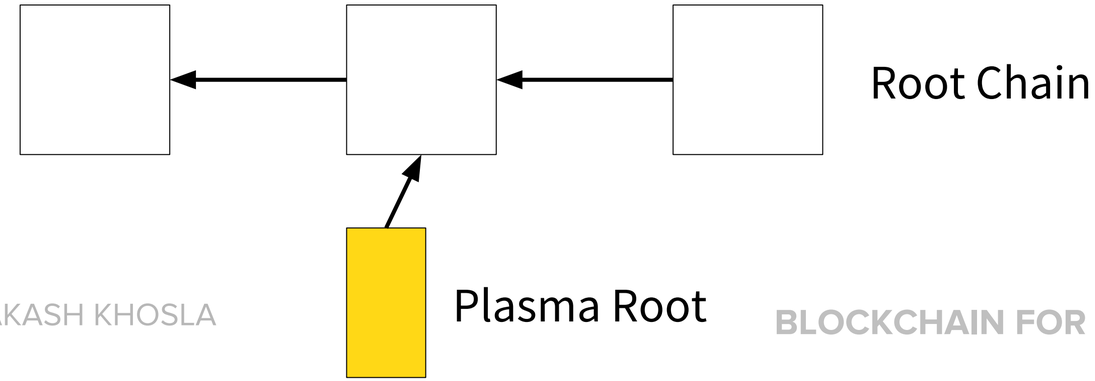
PLASMA
WALKTHROUGH

- Now that the child-chain is ready, we can create the basic components of our trading card game.
- The cards themselves are ERC721's, initially created on the ethereum main-chain, and then moved onto the child-chain through the plasma root.
  - This introduces a crucial point: Plasma lets us scale interactions with blockchain-based digital assets, but those assets should be created first on the ethereum-main chain.
  - Then, we deploy the actual game application smart-contracts on the child-chain, which contains all of the game logic and rules.



PLASMA
WALKTHROUGH

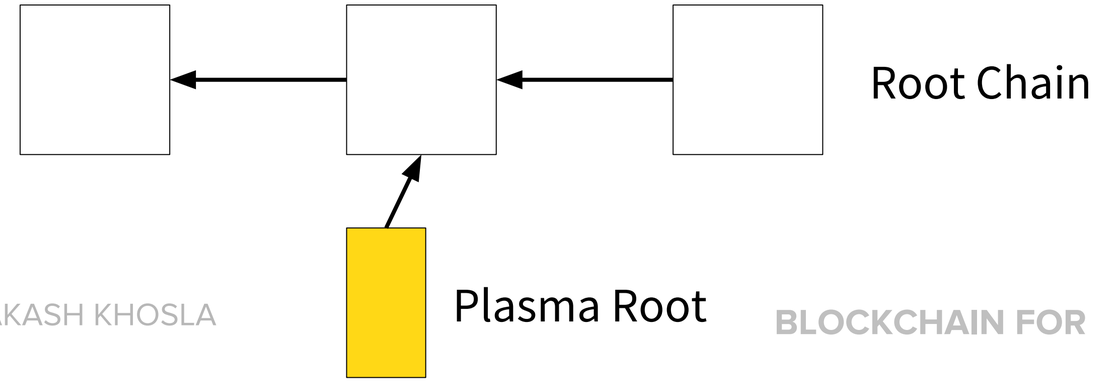
- When a user wants to play our game, they are only interacting with the child chain
  - They can hold assets (the ERC721 cards), buy and trade them for ether, play rounds of the game against other users—whatever our game lets them do—without ever interacting directly with the main-chain
- Because only a much smaller number of nodes (i.e. block producers) have to process transactions, fees can be much lower and operations can be faster



HOW IS PLASMA SAFE?

PLASMA

- So far we know that we can move operations off the main chain into a child chain to perform more
- But are all transactions on the child chain considered final? And Proof of Authority is lame because a single entity in this case is controlling block production on the child chain.. Pretty centralized right?
  - Therefore, the company can steal your funds or collectible cards when it wants?

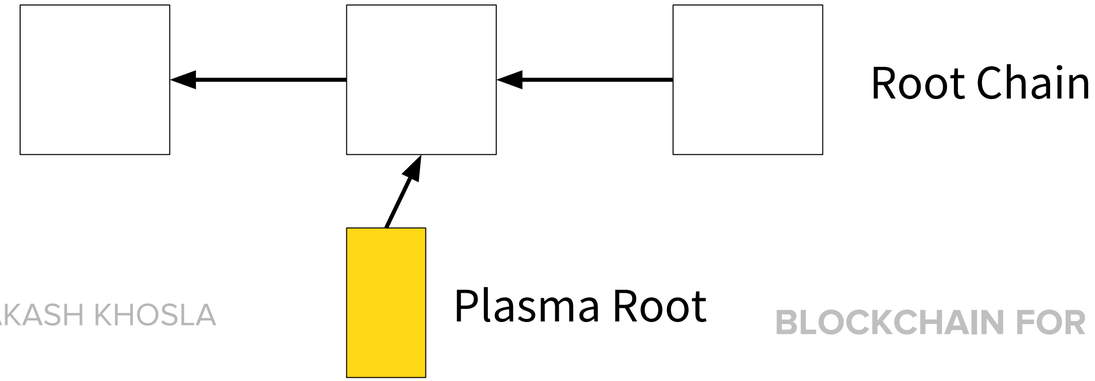


AUTHOR: AKASH KHOSLA
SOURCE

HOW IS PLASMA SAFE?

PLASMA

- Even in a scenario where a single entity controls 100% of block production on a child chain, Plasma gives you a basic guarantee that you can always withdraw your funds and assets back onto the main chain
- If a block producer starts acting maliciously, worst that happens is you **exit** the Plasma chain
- Plasma also has incentivized fraud proofs to prevent cheating and reward those who catch it
- But censorship resistance is lost in the case of PoA because the block producers can censor the users of child chain, i.e. by not including the transactions

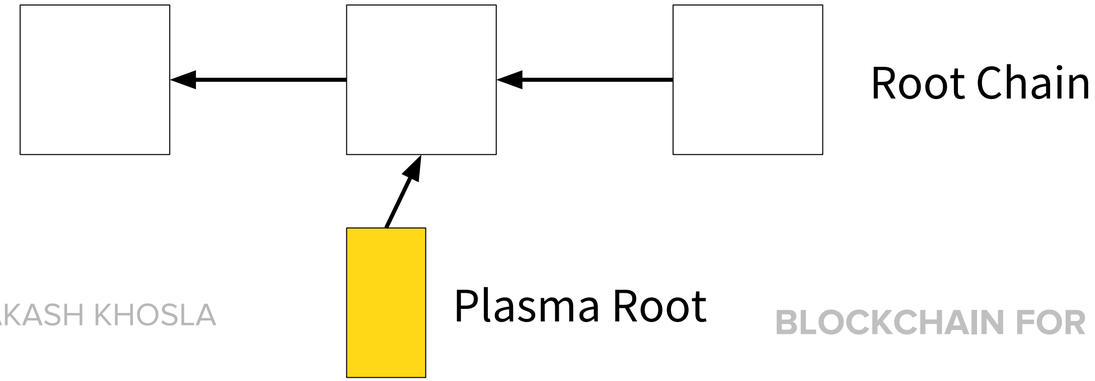


AUTHOR: AKASH KHOSLA
SOURCE

RISK OF WITHDRAWALS

PLASMA

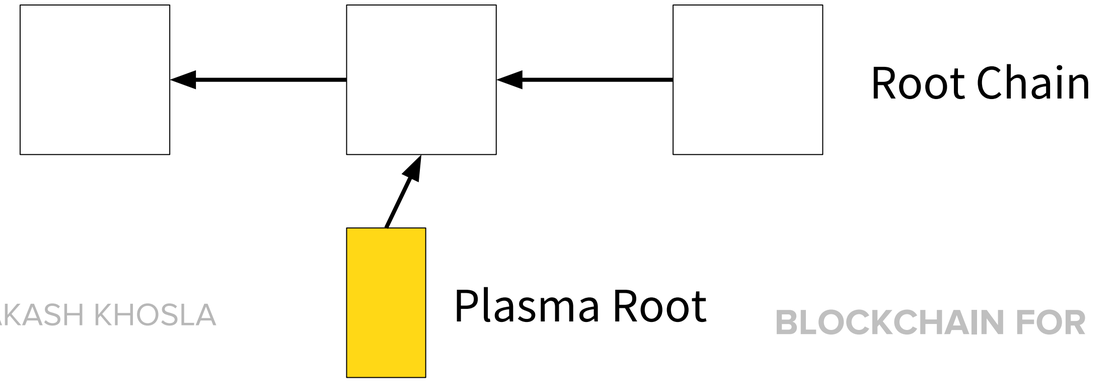
- One concern is what would happen if everyone using a child-chain tried to withdraw at the same time
  - In the case of a mass withdrawal, there might not be enough capacity on the ethereum main-chain to process everyone's transactions within the challenge period, meaning users could lose funds
  - There are many possible techniques for preventing this, e.g. by extending the challenge period in a way that is responsive to demand for withdrawals



AUTHOR: AKASH KHOSLA
SOURCE

NOTE ABOUT PROOF OF AUTHORITY BASED PLASMA

- It’s worth noting that it doesn’t need to be the case that all block producers are controlled by one entity—this is simply the extreme case in our example
  - We can create child-chains whose block production is spread among many different entities—i.e. actually decentralized in a way that is more similar to public blockchains
  - In those cases, there is less risk that block producers would interfere in the way described above, and so less risk that a user would have to move their assets back to the ethereum main-chain



AUTHOR: AKASH KHOSLA
SOURCE

PLASMA VS STATE CHANNELS

PLASMA

- State channels can perform instant withdrawals when all of the parties in the channel consent to the withdrawal
  - If Alice and Bob agree to close out a channel and withdraw their funds, so long as they both agree to the final state they can get their assets out of the channel immediately
  - On Plasma users must always go through a withdrawal process that involves a challenge period
- State channels should also be less expensive per transaction than Plasma, and be faster
  - This means that we will likely build state channels on Plasma child-chains
    - For example, in an application where two users are exchanging a series of small transactions
  - Building a state channel at the child-chain level should be cheaper and faster than performing each of those transactions on the child-chain directly

AUTHOR: AKASH KHOSLA
SOURCE

4 MINIMUM VIABLE PLASMA

MINIMUM VIABLE PLASMA

OVERVIEW

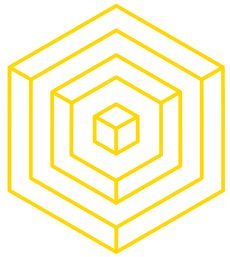
- Plasma chain should be secure as the root chain, which is why we can have a single plasma operator (block proposer in the plasma chain) because it is a constrained authority with proof of authority
- **Security guarantees**
  - Tokens cannot be double spent
  - Tokens cannot be withheld
  - Tokens can always be redeemed on the root chain (people should be able to exit the plasma chain and go back to the root chain)

MINIMUM VIABLE PLASMA

OVERVIEW

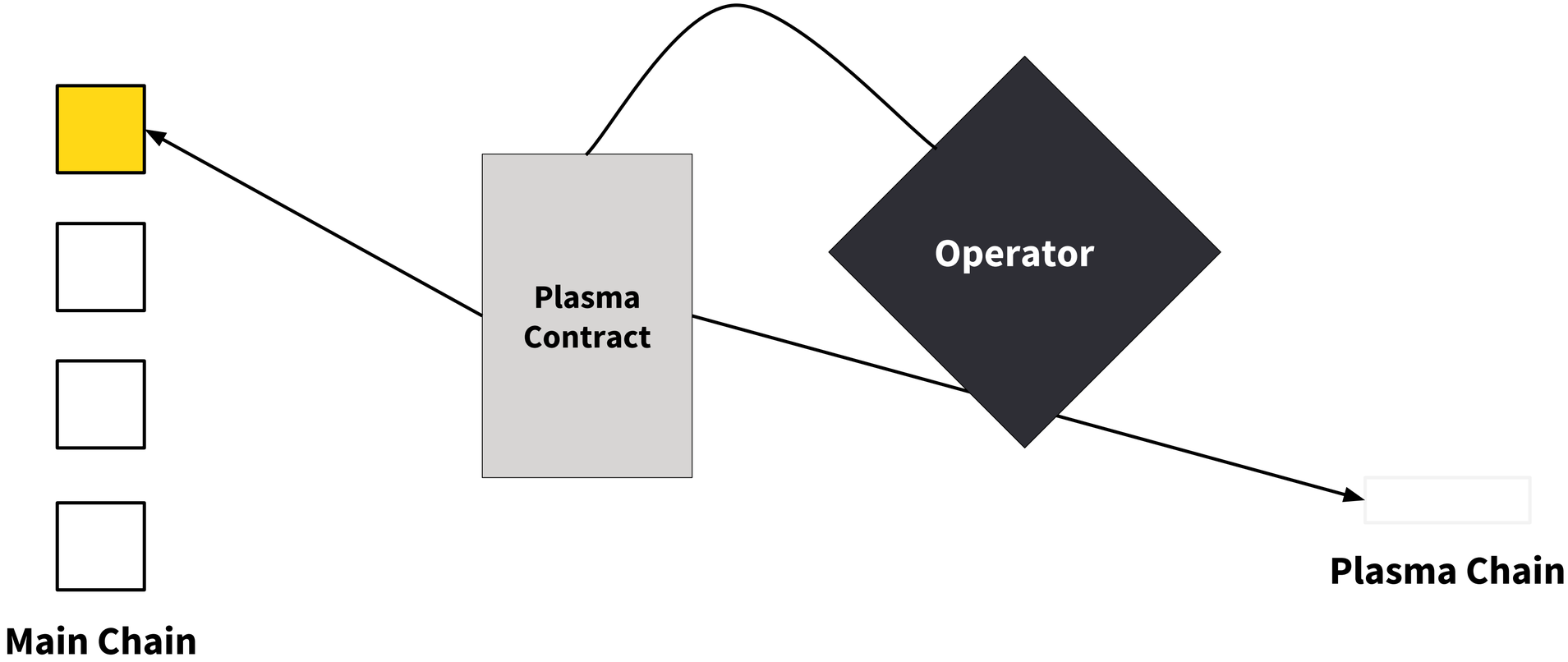
- **Plasma Exit -** How you pull coins off of the main chain
- This is coded in a smart contract on the root chain, which in this case in mainnet Ethereum
- The MVP is designed for token transfers
  - Can be designed for ERC 721, general state transitions, etc.
- MVP uses POA with a single operator pushing blocks to the main chain
- (Thanks to Karl Floersch for the next portions, they are based on his explanation)

4.1 OPERATOR CREATES A BLOCK



MINIMUM VIABLE PLASMA

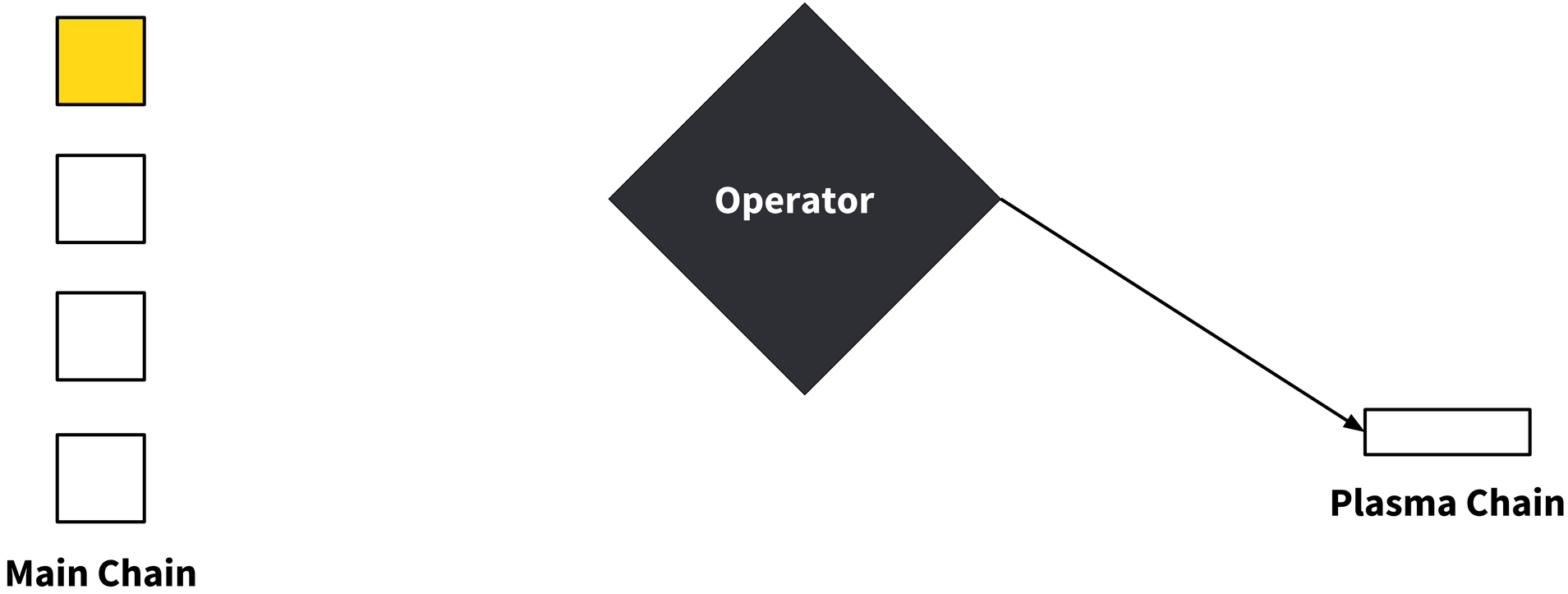
OVERVIEW



Plasma contract is deployed on the main chain and it manages the plasma blocks, which exists in the plasma operator's database

MINIMUM VIABLE PLASMA

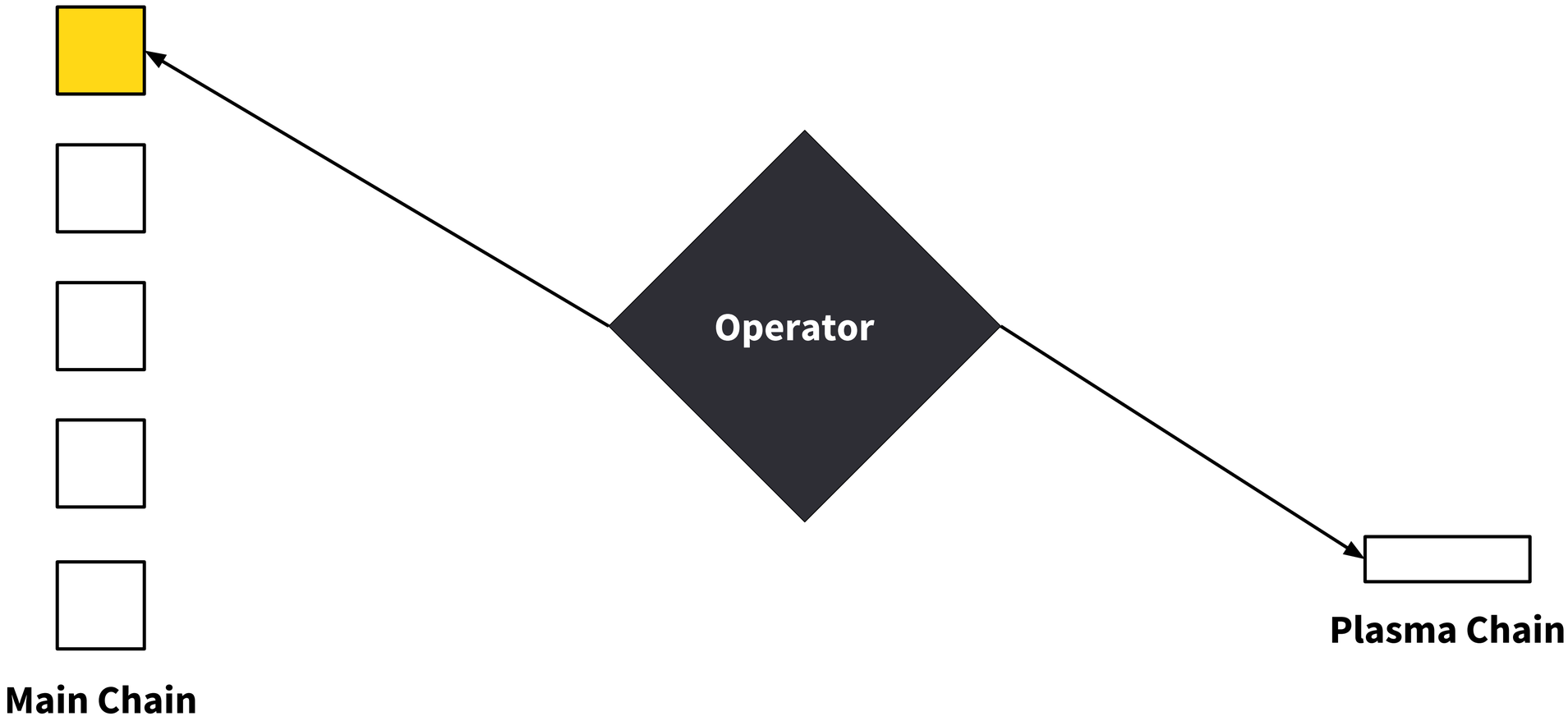
OVERVIEW



Operator creates a block, which is a plasma block inside the operator's database, and that's not included in the root chain

MINIMUM VIABLE PLASMA

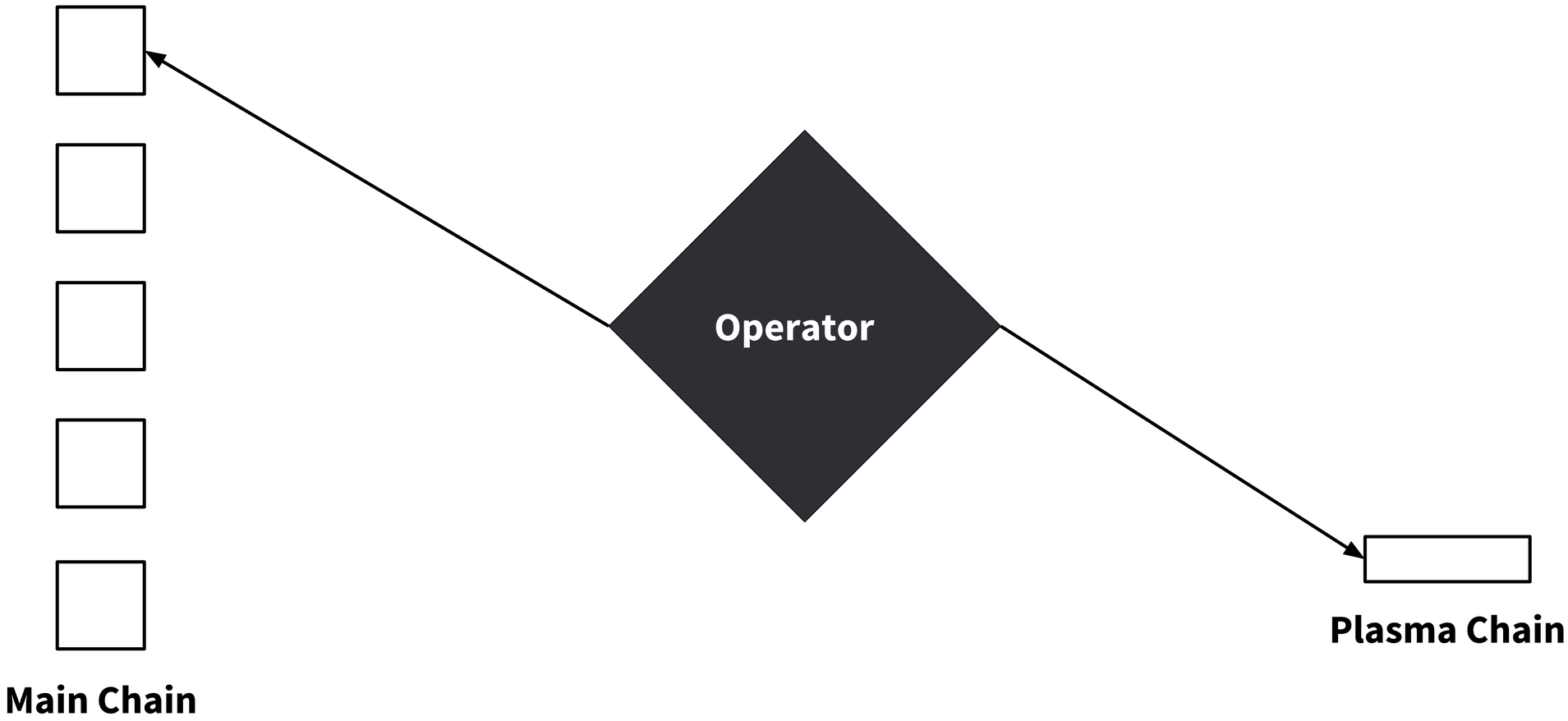
OVERVIEW



Operator sends transaction to create plasma block (contains a merkle root of the transactions and state) to main chain

MINIMUM VIABLE PLASMA

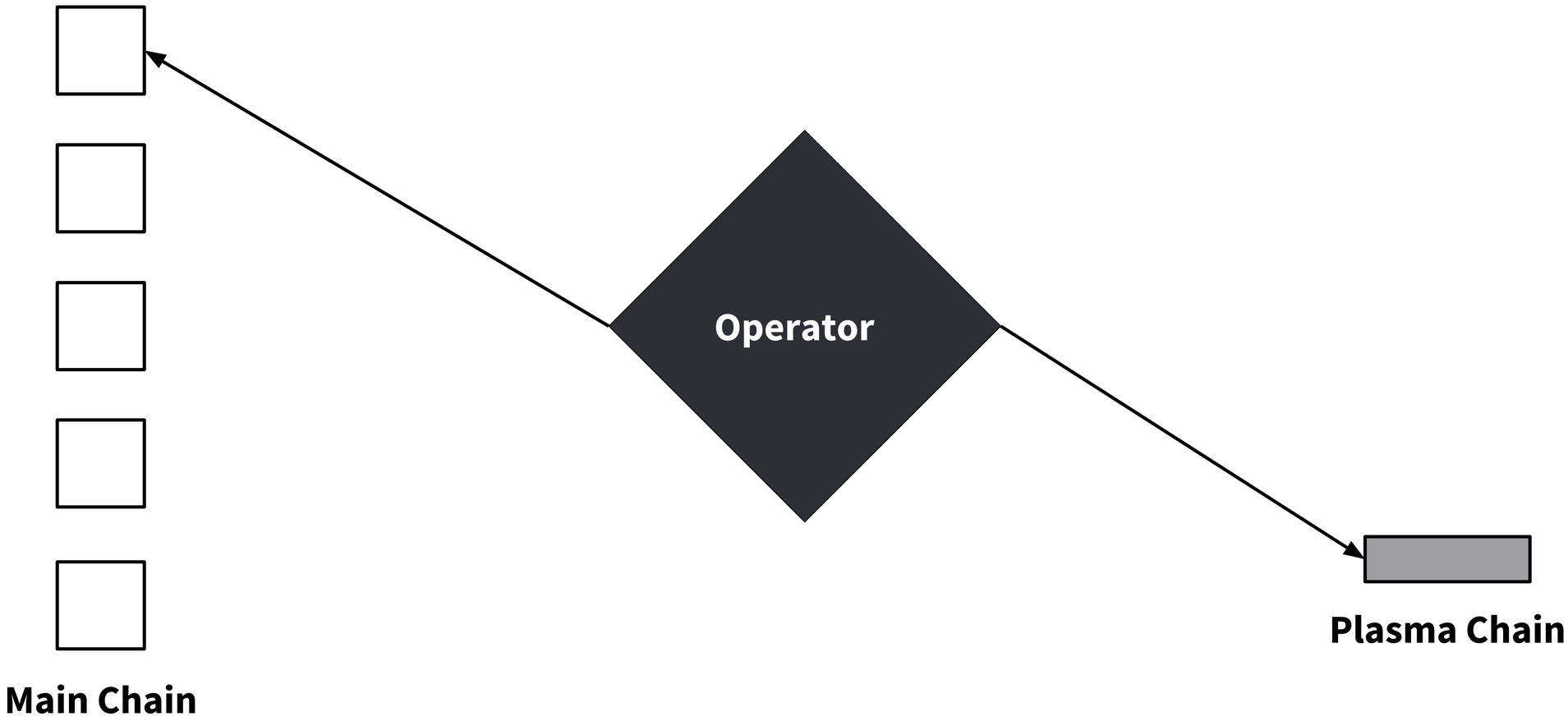
OVERVIEW



Miner receives it and creates a block on the main chain

MINIMUM VIABLE PLASMA

OVERVIEW

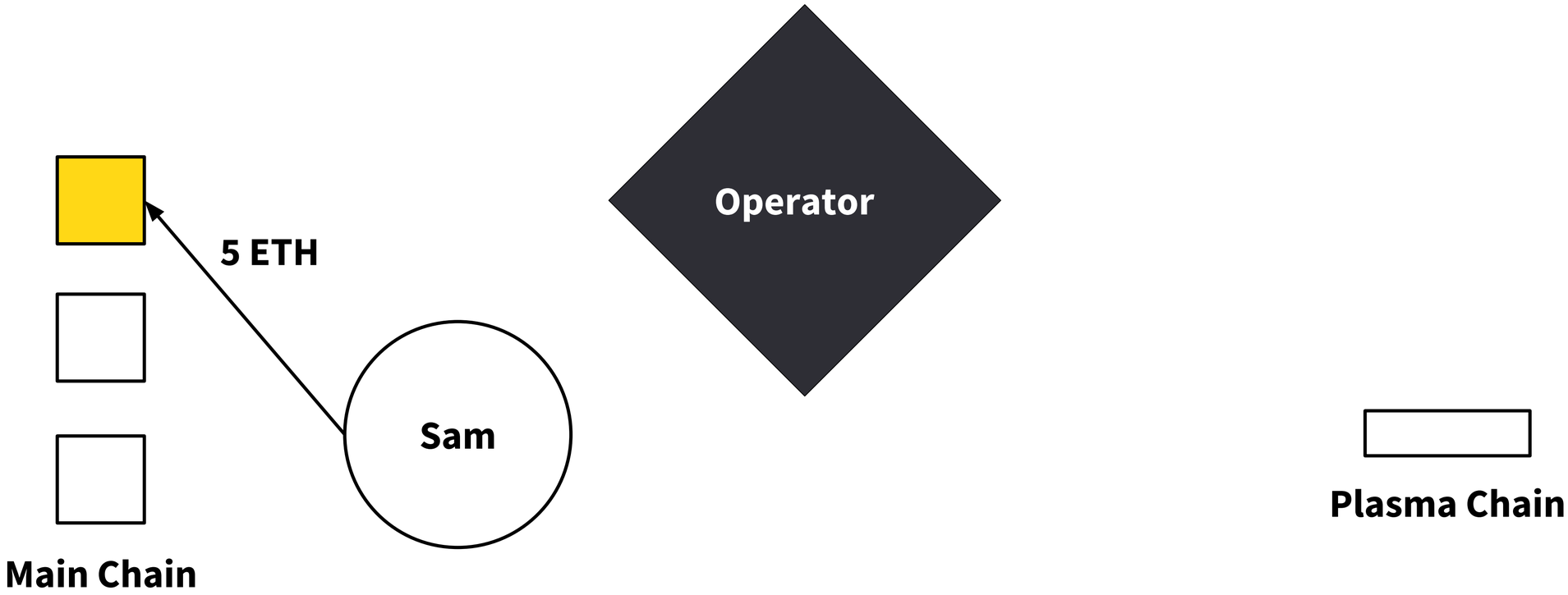


Plasma chain then gets confirmed once it is on the main chain

4.2 SAM DEPOSITS ETH INTO THE PLASMA CHAIN

MINIMUM VIABLE PLASMA

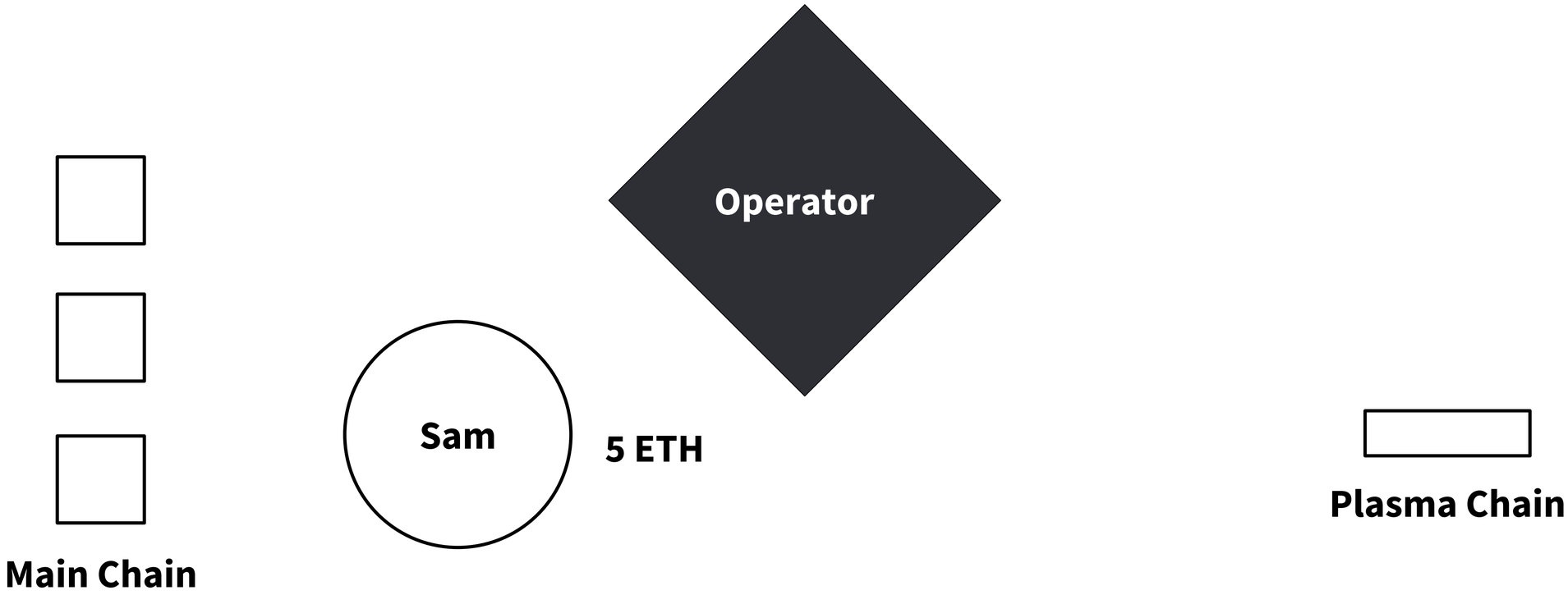
OVERVIEW



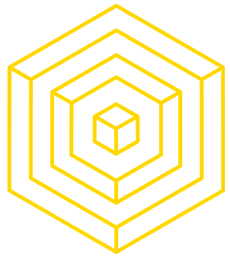
Sam deposits some ETH which immediately creates a block on the main chain

MINIMUM VIABLE PLASMA

OVERVIEW

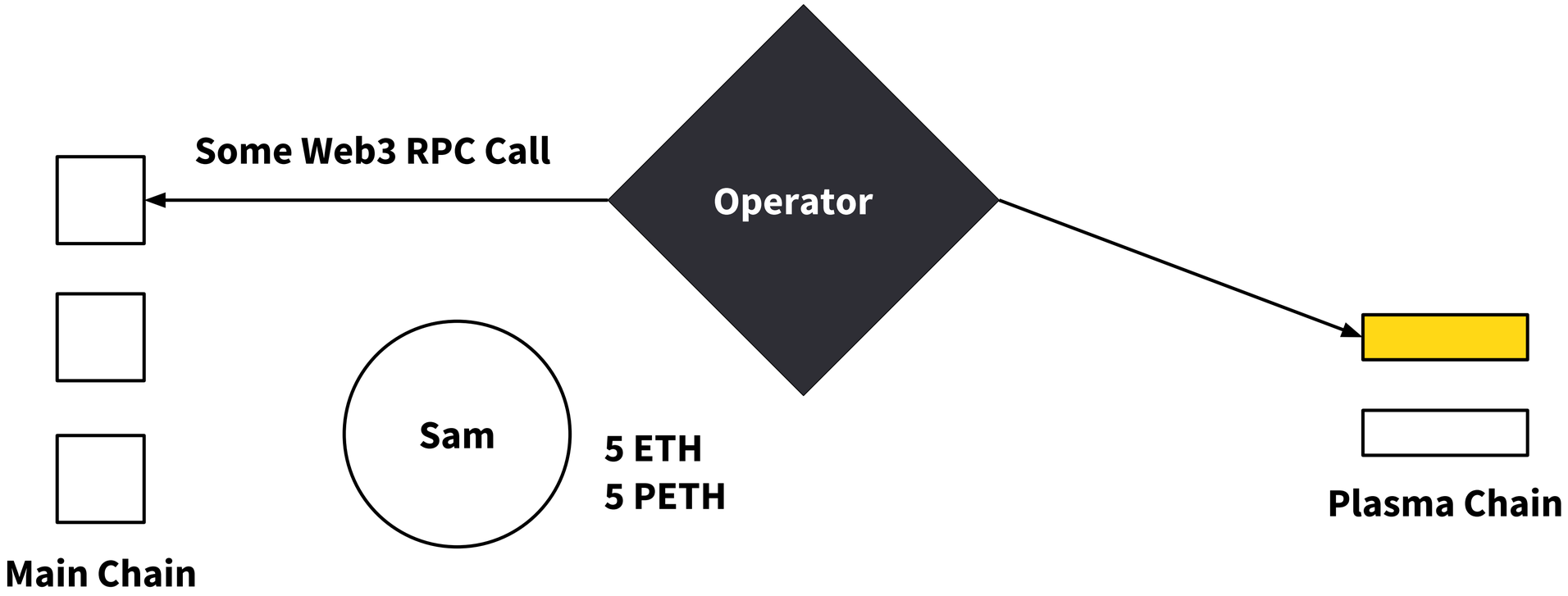


Block on the main chain says Sam has whatever ETH he deposited



MINIMUM VIABLE PLASMA

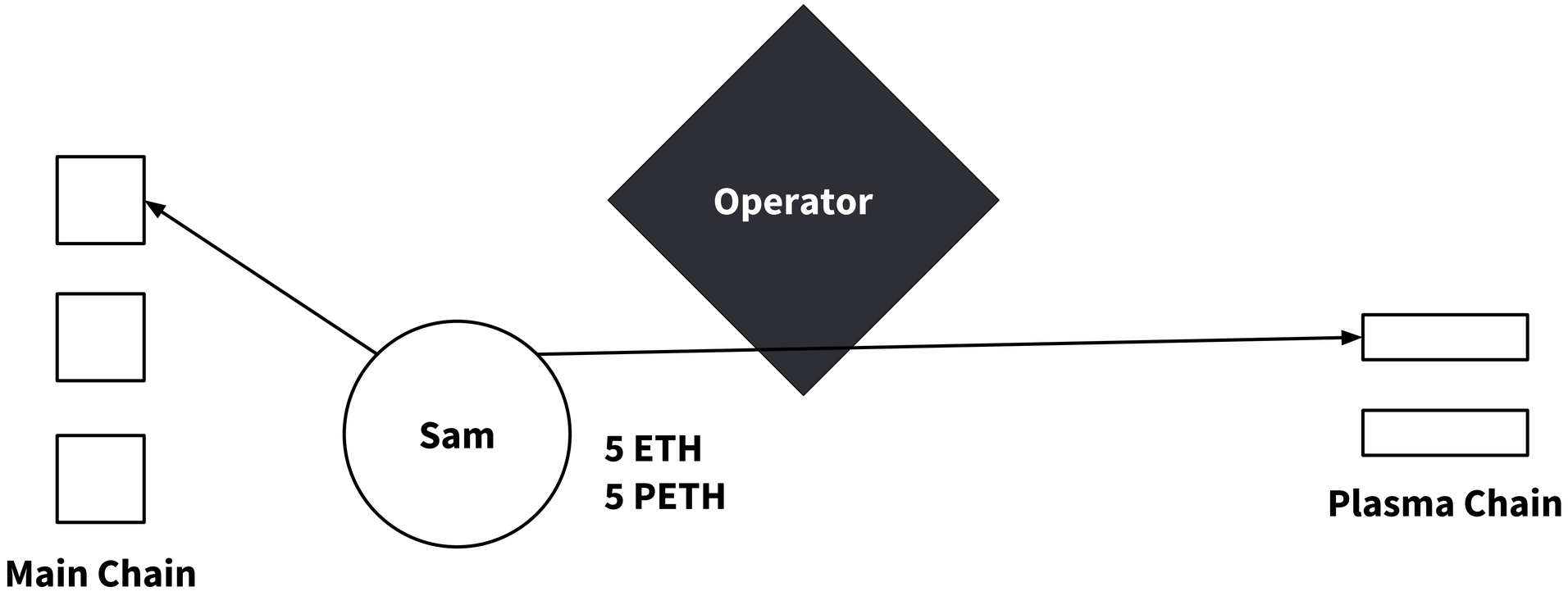
OVERVIEW



Plasma operator then syncs with the main chain, which is free of cost, because the ETH network basically took care of it

MINIMUM VIABLE PLASMA

OVERVIEW

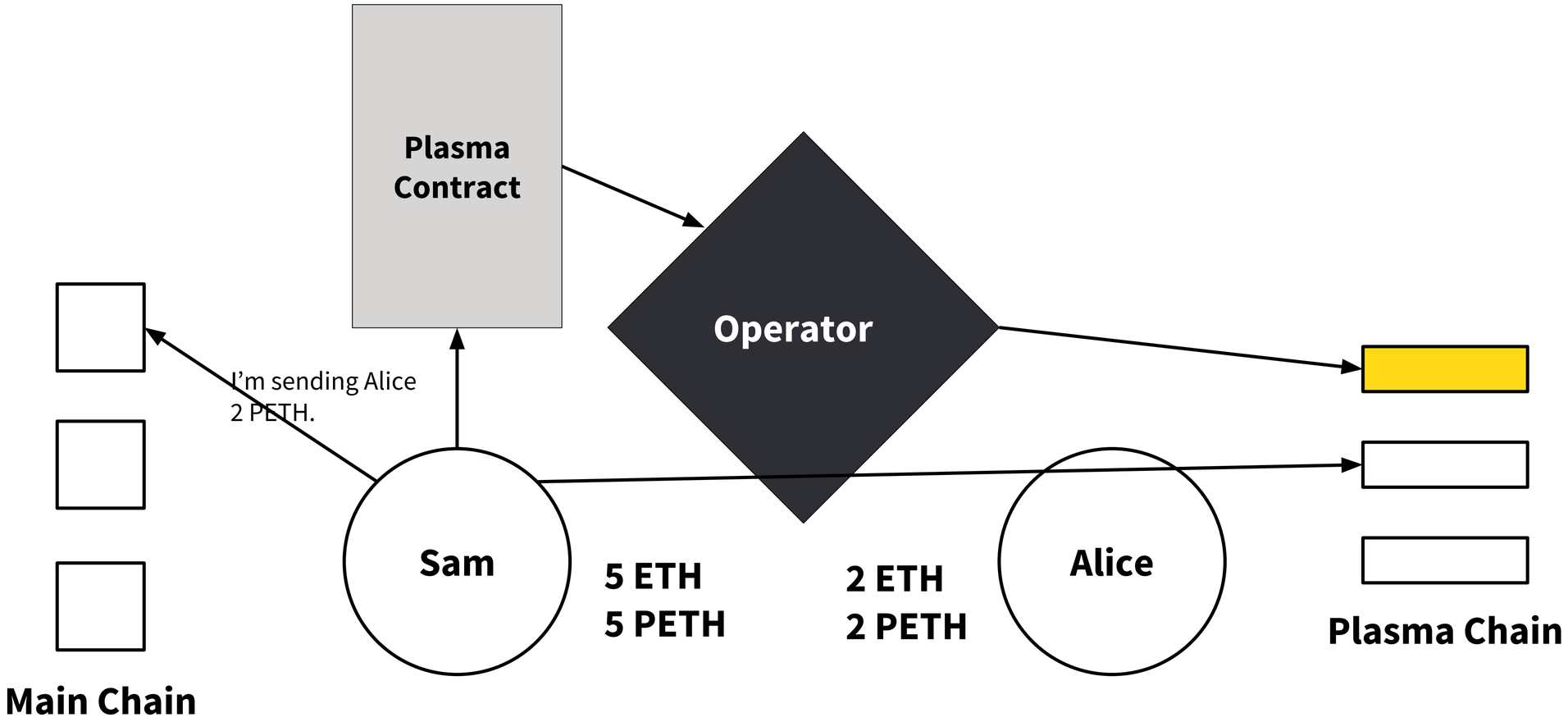


Sam will now watch both the main chain and plasma chain, looking for misbehavior on the plasma chain

4.3 SAM SENDS A BUNCH OF TXNS TO ALICE

MINIMUM VIABLE PLASMA

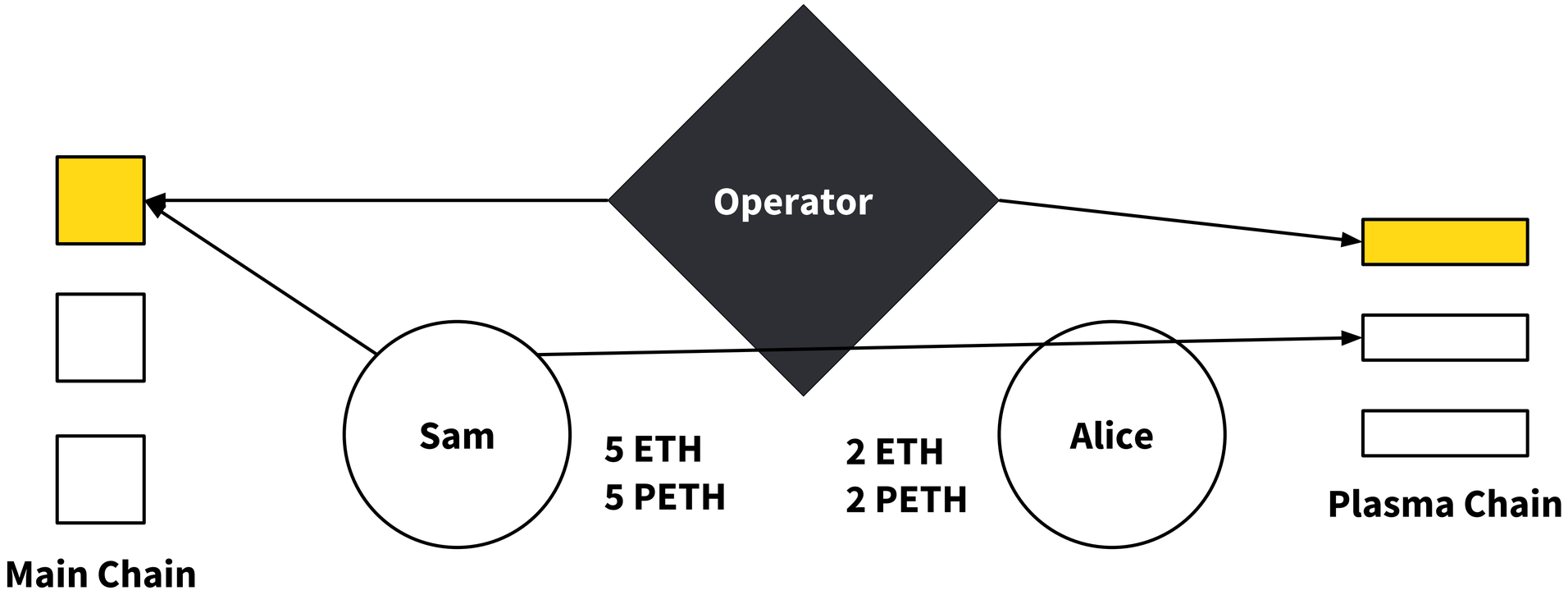
OVERVIEW



We create a block with Sam's transactions on the plasma chain, and note that the block has not been confirmed

MINIMUM VIABLE PLASMA

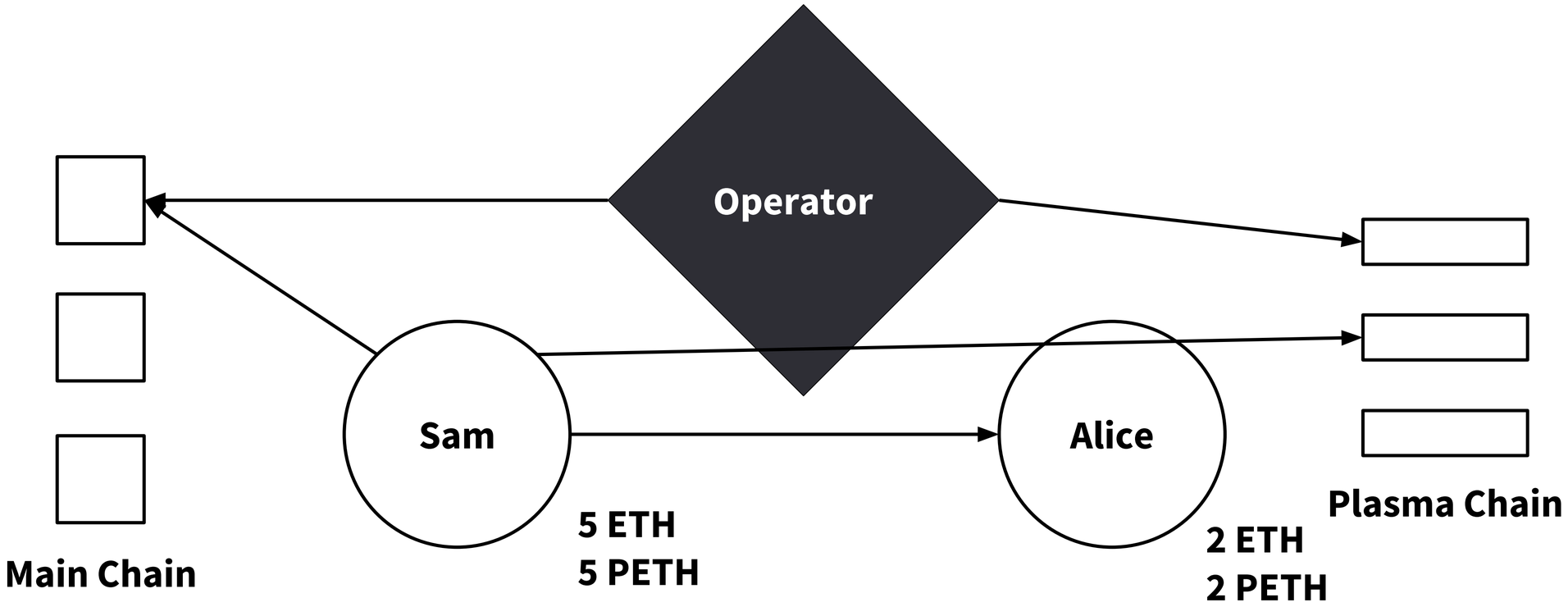
OVERVIEW



The block will be submitted into the main chain by the operator. Needs to be confirmed so balances not updated

MINIMUM VIABLE PLASMA

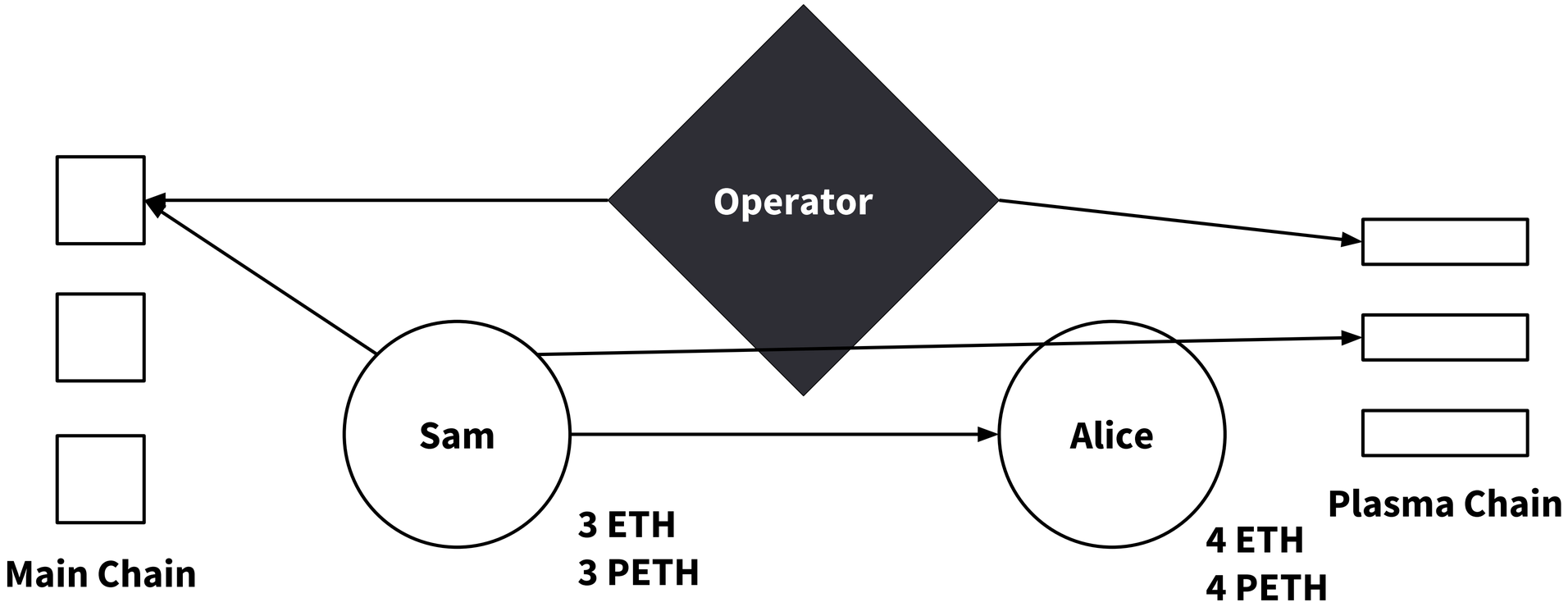
OVERVIEW



Sam will notice the two chains and see that the transactions were included. Sam sends a confirm message to Alice acknowledging his transaction is in the main chain.

MINIMUM VIABLE PLASMA

OVERVIEW



Balances update, Alice can spend those coins.

4.4 SAM ATTEMPTS TO EXIT WITH ALL 5 PETH BUT FAILS

MINIMUM VIABLE PLASMA

OVERVIEW

- Sam submits an exit transaction to the main chain, which is the final arbiter that keeps everyone in line
  - Remember than Sam no longers has these coins in this scenario, he sent them all to Alice
  - Exit will have a challenge period.
  - Challenge period - if your exit is invalid, main chain is accepting any transaction from anyone to challenge this, and anyone who challenges it sucessfully will receive a security deposit bond (basically a bounty for securing the system)!

MINIMUM VIABLE PLASMA

OVERVIEW

- Alice can send the challenge before the challenge period ends with the merkle proof of the txn tree and the state tree showing Sam spent the coins he tried to withdraw and the contract evaluates that and checks to see Sam's exit is invalid and therefore can cancel Sam's exit.
  - Alice gets paid a security deposit

4.5

OPERATOR CREATES AN INVALID BLOCK AND ALL USERS EXIT

MINIMUM VIABLE PLASMA

OPERATOR CREATES AN INVALID BLOCK

- Operator tries to print money by creating an invalid block, perhaps by stealing from the users
- Clients notice bad behavior and submit exit transactions before their money gets stolen by the operator
- Well, the plasma operator can try and exit to steal all the PETH
- However, exists based on older transactions are processed first
- Therefore the plasma operator's exit will come later, it's based on a newer txn

4.6

CONCLUDING THOUGHTS

ISSUES
OVERVIEW

- It’s the responsibility of everyone holding value on side chain to notice if something goes wrong, and to exit
- There could be congestion if everyone tried to exit at the same time, and doubly bad if there was a busy ICO happening at the same time
- Ultimate finality is on the main chain
- In a POA network you need to explicitly trust the authorities
- Tooling can be an issue, although at least one team (POA Network) is working on an open source block explorer
- Can’t offer transfers as quickly as state channels do (although allow for many-to-many transactions)

CONSTRAINTS OF DESIGN PATTERN

OVERVIEW

- Trying to scale blockchains while not losing decentralization
- We use deposits to constrain bad actors
- Trust is established by limiting possible actions, and imposing costs for bad behaviors
- Store minimal information on chain, plasma does this by avoiding any storage of intermediate state, it only stores incoming & outgoing balances

SEE YOU NEXT TIME

ZK-SNARKs

libsnark focused, with

Howard Wu