By Stephane Gosselin and Ankit Chiplunkar on behalf of Frontier Research

For most recent information about OneBalance, please visitfrontier.tech.

Motivation

Web3 and the crypto ecosystem more broadly has historically had a chain centric worldview. We believe this is an outdated framework originating from a perceived scarcity in blockspace due to bundling of credible commitment machines with global consensus.

We believe an ecosystem wide transition towards an account centric worldview will enable the consolidation of a fragmented user experience. With this proposal for a new account model, we hope to provide a missing component of the web3 stack that will help onboard the first billion humans onto crypto.

TLDR

OneBalance is a framework for creating and managing accounts on credible commitment machines.

These accounts allow users to conveniently and reliably request state transitions on any chain.

This is achieved through the introduction of two key concepts:

1. Accounts hosted on credible commitment machines, and
2. Resource locks

The credible commitment machine is responsible for securing the account, issuing resource locks over the state it holds, and validating the fulfillment of such locks.

Such an architecture requires a transition from a transaction supply chain built around Externally Owned Accounts (EOAs), the JSON-RPC API, and transactions, to one built around credible commitment machines and resource locks.

By introducing OneBalance, we hope to accelerate the transition of the ecosystem towards such an architecture.

A OneBalance account can:

- Combine token balances from every chain
- Abstract gas on any chain
- Swap and send tokens to and from any chain
- Issue complex permissions over any subset of user state
- Incentivize and enforce atomic asynchronous composability across multiple chains
- Fast confirmations through separation of fulfillment from settlement

Many use cases are unlocked with these new capabilities. Users can spend any token to pay for state transitions. They can aggregate liquidity that lives both on-chain and off-chain. They even have the building blocks to build a decentralized Fireblocks, or a non-custodial Binance.

The EOA Account Model

The web3 market structure equilibrium, as defined by Ethereum, is the use of public-private key-pair, aka, Externally Owned Account (EOA) to manage all aspects of a user's state.

EOAs play three crucial roles:

- Authentication: a user can generate authenticated state transition requests because they have the private key associated with a given public key
- On-chain identity: a user can store state at the address associated with their public key
- Permissions: a user can issue permissions to modify their state by signing transactions which are interpretable by the chain

But they have limitations:

- No way to authenticate with SSO, passkeys, application keys, etc.
- Users can’t flexibly delegate attenuated control of their state
- The key pair is tied and limited to their respective signing ecosystem

This makes EOAs, and the transaction supply chain more broadly, incapable of providing the guarantees required for allowing solvers to act as third party executors without risk of being griefed through equivocation or double spending.

The existing account model is holding back a lot of use-cases that would benefit users. So what would a new account model look like?

A next generation account model should have the following characteristics:

- Support the use of modern authentication (FIDO, passkeys, etc)
- Enable granular permissions to modify account state, even when the use is offline
- Issue commitments which solvers can rely on to take finality risk in cross chain execution
- Allow users to move seamlessly between levels of custody, enabling progressive self-custody

We call our proposed solution the OneBalance account model.

The OneBalance Account Model

Each OneBalance account can be thought of as its own rollup. The account wraps individual user state from all chains and replicates it in a virtual environment. This virtual environment issues state transition requests as “resource locks” and fulfills those state transitions through cross-chain execution proofs. This virtual environment is secured by a credible commitment machine.

Crucially, a OneBalance account can create an arbitrary number of sub accounts across any number of chains and manage any state present on those chains. It is backwards compatible with all chains, smart contracts, and assets including ERC20s, NFTs, DAOs, multisigs, defi protocols, and deposits or points programs.

[

OneBalance Account Model

2666×1500 480 KB

](https://ethresear.ch/uploads/default/original/3X/c/b/cbf20c4700ee30fc7099a71d5dea684cac7ed8f8.png)

Resource Locks

A resource lock is a credible commitment made by a user to escrow some state conditional on particular conditions fulfilled, or an expiry time.

An example could be a cross-chain request to purchase an NFT on Solana using USDC deposited in the OneBalance account from Ethereum.

resource_lock: { lock: 1500 USDC, fulfill: DeGods #12345, expiry: Solana block 245547084 }

Resource locks are necessary to prevent solvers from being griefed by a user through double spending or equivocating their request during execution.

Since the user makes a commitment not to overwrite their request within a time window, it removes the uncertainty solvers typically incur between a transaction being signed and the finalized chain state.

A lock is analogous to depositing funds in a smart contract, or issuing an ERC20 approval, but without spending gas or waiting for on-chain finality since it is done within the account itself.

The lock expiry needs to be of long enough duration to provide solvers the chance to execute the requested state transition on the destination chain and submit a proof of fulfillment to the fulfillment engine.

Crucially, this introduces a separation between fulfillment time and settlement time. Since the account provides local assurance of the lock, solvers can bring a requested state transition on a destination chain without waiting for finality on the origin chain.

This allows users to buy SOL with ETH at the speed of Solana without being constrained by the speed of Ethereum. This

fulfillment speed can be extended to execution of any state transition such as sniping an NFT, sending money to your grandmother, or anything else users do on blockchains.

Resource locks can implement constraints which sit anywhere along the spectrum of permissions.

[

Permission Spectrum

3474×1151 200 KB

](https://ethresear.ch/uploads/default/original/3X/9/c/9ca35b639e52c67b62c207b50b1850c50ca2037c.png)

Permissions could be stateful or stateless. For example:

- Scoped session keys

: allows an app like a Telegram bot to take arbitrary actions on subsets of a user's token balances

- Circuit breaker

: sell all open positions if there is no account activity or market volatility above a predefined threshold

- Limit order:

post an order if a pair reaches a certain price on a DEX

- MFA

: post a transaction if two valid authentication methods are provided

Credible Commitment Machine

A credible commitment machine is a secure computer on which the account lives and is trusted to provide assurances over the valid issuance of resource locks and the validation of their fulfillment.

We present here four possible architectures of credible commitment machines which provide for secure issuance and enforcement of locks: Trusted Execution Environments (TEEs), Multi-Party Computation / chain signatures (MPC), Smart Contract Accounts (SCAs), and in protocol virtual machine changes.

These mechanisms are being developed and refined as we speak, it is likely that the ideal architecture today will look vastly different than the one five years form now.

This is why OneBalance account model is providing a framework which is composable and adaptable with the evolving parts of the ecosystem.

[

Credible Commitment Machines

2430×1645 232 KB

](https://ethresear.ch/uploads/default/original/3X/3/3/336a55816553061931fc80230793ec6330c071c2.png)

Roadmap

OneBalance v1:

- add support for transactions and swaps of any token on any chain
- add support for session keys for trusted applications
- add support for user rage quit through exit hatch

OneBalance v2:

- add support for stateless policies
- add support for arbitrary transactions
- add support for authentication modules

OneBalance v3:

- add support for stateful policies

OneBalance v4:

- add liveness guarantees through account replication

EOA

OneBalance v1

OneBalance v2

OneBalance v3

OneBalance v4

self-custody

transfers

swaps

session keys

transactions

stateless policies

auth modules

stateful policies

liveness guarantees

Acknowledgements

Thank you to the collective consciousness of the crypto ecosystem for fostering a fertile ground for innovation.

In no particular order, thank you to the following collaborators for the many stimulating discussions which lead to the creation of this proposal:

Murat Akdeniz, Ahmed Al-Balaghi, Viktor Bunin, Jonah Burian, Vitalik Buterin, Philippe Castonguay, Vaibhav Chellani, Valery Cherepanov, Jasper De Gooijer, Nicolas Della Penna, Justin Drake, Brendan Farmer, Ben Fisch, Mattia Gagliardi, Johnny Gannon, Matt Garnett, Ivo Georgiev, Christopher Goes, Pedro Gomes, Mason Hall, Sam Hart, Connor Howe, Sreeram Kannan, Hart Lambur, Zaki Manian, Robert Miller, Alex Obadia, Puja Ohlhaver, Anatolii Padenko, Nick Pai, Illia Polosukhin, Karthik Senthil, Tomasz Stanczak, Henri Stern, Alex Stokes, Caleb Tebbe, Dror Tirosh, Dean Tribble, Drew Van der Werff, Alex Watts, Yoav Weiss, Nathan Worsley, Evgeny Yurtae, Philipp Zentner, Noah Zinsmeister, apriori, jxom, awkweb.

We look forward to continued conversation on this topic.

Discussion

Why are you doing this?

Our mission is to help the web3 ecosystem to transition to an account centric worldview in order to bring web3 to the first 1 billion people.

We believe non-coercive credible commitment machines are essential for human coordination in the digital age.

We believe the chain centric worldview is an outdated framework originating from the historical scarcity in blockspace.

Much like the shift from the Geocentric worldview to the Heliocentric worldview unlocked a wealth of discoveries, we believe the shift from a chain centric worldview to an account centric worldview will unlock the full potential of web3.

What does this mean from a user perspective?

Users don’t need to care about which chains they are interacting with + can get close to instant fulfillment.

Lets take a complex, yet common example:

User wants to buy an NFT traded on Solana with a price of 10 SOL, but only has USDC on Ethereum.

This state transition request requires the following sequential operations to take place:

1. Generate: Create a new Ed25519 keypair in a solana wallet
2. Swap: USDC for ETH to pay for gas on Ethereum
3. Bridge: Send USDC to bridge contract on Ethereum and get USDC minted on Solana
4. Swap: USDC for SOL to pay for gas on Solana
5. Swap: USDC for SOL to purchase NFT
6. Execute: Execute calldata on marketplace to purchase the NFT

Today, users are required to manually perform each of these actions and wait for the previous one to be settled or finalized before performing the following one. Some of these operations are even technically impossible in a non-custodial way using EOAs without chain level gas abstraction.

The critical path of execution here requires waiting for the settlement of two Ethereum transactions and two Solana transactions + waiting for the finality of Ethereum. With Ethereum's current block finality time, we are looking at a minimum of 15min to complete execution.

Lets look at the equivalent using a OneBalance account:

User wants to buy an NFT traded on Solana with a price of 10 SOL, but only has USDC on OneBalance.

This state transition request requires the following sequential operations to take place:

1. Create a resource lock on OneBalance as follows:

resource_lock: { lock: 1500 USDC, fulfill: DeGods #12345, expiry: Solana block 245547084 }

There are no additional steps for the user to take.

Behind the scenes, a solver purchases the NFT and credits it to the OneBalance proxy account of the user on Solana, and claims the resources in the lock.

Since a OneBalance account separates fulfillment from settlement, the user gets execution at the speed of transaction execution on their destination network, in this case Solana. The user can perform any operation on any chain using any token they hold in their OneBalance account.

How does this relate to 4337 style approach to account abstraction?

wip

How do you guarantee atomic asynchronous execution across chains?

We can't. But we can incentivize it.

Accounts can enforce two types of constraints:

- Constraints over what states of the world it requires in order to issue a lock
- Constraints over what states of the world it requires in order to fulfill a lock

OneBalance can incentivize

atomicity, but it cannot guarantee it.

- For example: Take a lock that requests sequencing of state transitions across chain A and chain B, with a desired atomicity such that B is executed if and only if A is executed. If the lock is public and the only condition on inclusion of B is the existence of a valid signature from the requesting account, then B could be included regardless of A if it exposes sufficient MEV to compensate its inclusion.

However! Since executing B without A invalidates the fulfillment conditions of the lock, then the solver cannot extract any value from the lock. This means that from the user perspective, atomicity is maintained. Some people refer to this as "economic atomicity". For locks with complex multi chain atomicity requirements, Solvers take on the risk of non-execution.

This “lock leaking” due to MEV problem can be resolved at the routing layer by routing the state transition request through a secure OFA that prevents information leakage.

Novel mechanisms are being developed at the settlement layer to help solvers manage their settlement risk using things like pre-confirmations or proof aggregation. Importantly, these help manage solver execution risk and therefore help minimize non-execution risk for users of a OneBalance account, but the OneBalance account already has economic atomicity guarantees.

Is OneBalance just HTLCs for Ethereum?

I prefer to call it turbo HTLCs

The design of intent bridges work similarly to HTLCs in that they lock user funds on the originating chain until a proof is provided of the completion of a state transition on the destination chain.

Instead of being constrained by the speed of the originating chains, OneBalance accounts are constrained by the speed of the credible commitment engine in generating these locks. This means that on a TEE architecture, locks can be issued at clock speed of a single server, hence turbo HTLCs!

How does this thing scale to one billion concurrent users?

OneBalance accounts create “local” locks, whereas regular accounts can only create “global” locks. Global locks require locking the state of all accounts in the execution environment during sequencing, whereas local locks only require locking the state of accounts which are party to the lock.

Unlike global locks, local locks provide the opportunity for lock sequencing to be parallelized on distinct machines.

Where does the account live?

The account lives in a secure computer of the users choice that can make credible commitments about what messages it will and won’t sign.

The four architectures for credible commitment enforcement presented above provide such environments, but each have their tradeoffs. OneBalance accounts are not opinionated to the type of credible commitment enforcement mechanism used.

Crucially, as the sophistication of these architectures evolve over time, the tradeoff space will change and so will user preferences. As such, OneBalance accounts must remain flexible to supporting different architectures.

Is OneBalance a standard?

No.

OneBalance is a framework for building accounts on top of credible commitment machines. A standard is being developed by the CAKE working group to expose an API to connect to any credible commitment machine based accounts.

Are you providing pre-confirmations?

No. OneBalance provides resource locks.

Lets explore the difference between the two.

Pre-confirmations is a mechanism being actively developed by several teams in the ecosystem to offer better inclusion guarantees for entities sending transactions to the blockchain.

Resource locks is a mechanism for offering guarantees to solvers that a user cannot double spend or equivocate on their state transition request.

Both mechanisms are aimed at reducing the execution risk of solvers, pre-confirmations are guarantees provided by proposers, resource locks are guarantees provided by users.

Both pre-confirmations and resource locks can be enforced by the same credible commitment mechanism.

For example: If Ethereum validators were to deposit their staked ETH in a OneBalance account, they could create resource locks which specify slashing rules if certain transaction inclusion commitments are invalidated. This could be implemented with a restaking contract like eigenlayer.

[

image

3587×1114 138 KB

](https://ethresear.ch/uploads/default/original/3X/4/c/4c122bf52622b902e0be90e45e5ec787ab5e0b89.png)

How does a lock turn into a state transition on chain?

OneBalance accounts are responsible for issuing and enforcing resource locks over state transition requests.

Routing of the request in order to provide fulfillment is a downstream module which the upstream app / user must define. For certain kinds of requests, it may be beneficial to route directly to a chain's transaction pool, other requests may benefit from using a solver network or orderflow auction, and others from specifying an exclusive solver.

OneBalance is unopinionated about the routing mechanism.

Is OneBalance a competitor to (insert my bags here)?

Probably not.

OneBalance is providing a framework for orchestration of stateful accounts. Our goal is to displace the industry wide reliance on imperative state transition requests issued by user managed EOAs. OneBalance is not opinionated on the architecture of the stateful accounts or the credible commitment mechanisms used to secure them.

OneBalance uses a modular architecture that allows for the following components to be integrated:

1. Orderflow sources: wallets / apps / tg bots / waas
2. Fulfillment engines: orderflow auctions / solvers / solver networks / market makers / intent networks / bridges
3. Settlement engines: any L1 or L2 (yes, even BTC)

The design for OneBalance emerged from our work with members of theCAKE working group.

Looking at the CAKE framework, OneBalance sits between the permissioning layer and the solver layer and is compatible with all the other necessary components of the cake.

[

image

1763×2267 147 KB

](https://ethresear.ch/uploads/default/original/3X/a/a/aac5be2963c62330d3ee339ff53a64b1a09f9139.jpeg)

Are OneBalance accounts custodial?

No.

All OneBalance accounts are issued withdrawal receipts which allow them to permissionlessly exit their assets back to the settlement chain where the assets originate.

This means at any point, a user is able to "rage quit" and recover their assets on the source chain.

This mechanism is implemented differently depending on the accounting engine used, but essentially boils down to the same outcome: users can withdraw their funds by submitting withdrawal receipts to their proxy accounts on origination chains.

I firmly believe freedom of exit is an essential characteristic of non-oppressive human coordination system design. I have never and will never design a system without freedom of exit.

OneBalance accounts really compatible with any chain?

wip

Are OneBalance accounts vulnerable to vampire attacks?

Yes. This is a good thing.

If someone can build a system which delivers higher combined utility (functionality + economic incentive) to users than OneBalance accounts, users should be able to exit. This is necessary to avoid anti-competitive monopolies that censor innovation.

To whoever wants to try, I say this; bring it on.

Is this a Keystore Rollup?

No.

As they are defined today, a keystore rollup solves the problem of having a central source of truth for account permissions, across all chains. It doesn't offer cross-chain guarantees to solvers. A OneBalance account does not need a keystore rollup since on chain keys don't change.

If a Keystore Rollup was to attempt to issue locks, they could only be communicated at L1 finality speed.

How do you prevent sequencer DOS?

wip

What is the trust model?

Instead of requiring mutual trust between users and solvers, the OneBalance model requires each party to trust the credible commitment mechanism used to issue and enforce resource locks.

Show me some sequence diagrams.

Ok.

Lets walkthrough a few examples on how interoperability is achieved under different conditions:

1. The user has EOA account and wants to do a cross-chain contract call while not having gas on target chain
2. The user signs two transactions using their EOA, the first transfers gas amount from the origin chain to the solver escrow address and the second calls the contract on the target chain.
3. As soon as the solver simulates these two transactions, they have a guarantee that the user will pay them the correct gas amount (user's commitment is enforced).
4. The solver instantly funds user's EOA account on the target chain, and executes the contract call transaction without waiting for settlement or finality of first transaction.
5. The solver can include the first transaction (gas payment) within the expiry window of commitment.
6. The user signs two transactions using their EOA, the first transfers gas amount from the origin chain to the solver escrow address and the second calls the contract on the target chain.
7. As soon as the solver simulates these two transactions, they have a guarantee that the user will pay them the correct gas amount (user's commitment is enforced).
8. The solver instantly funds user's EOA account on the target chain, and executes the contract call transaction without waiting for settlement or finality of first transaction.
9. The solver can include the first transaction (gas payment) within the expiry window of commitment.

[

diagram-20

2006×1032 137 KB

](https://ethresear.ch/uploads/default/original/3X/1/a/1af6c767e76f17d368ddd9f171ee4745538f19bf.png)

1. The user has Smart Contract account and wants to do a cross-chain swap while not having gas on target chain.
2. The user signs a UserOp authenticating the transfer of required tokens to the solvers escrow address.
3. As soon as the solver simulates the transaction, they have a guarantee that the user will pay them the required tokens (user's commitment is enforced).
4. The solver procures the required tokens on the target chain and deposits them into the users account on the target chain without waiting for settlement or finality.
5. The solver can include the UserOp (token deposit in escrow) eventually.

6. The user signs a UserOp authenticating the transfer of required tokens to the solvers escrow address.
7. As soon as the solver simulates the transaction, they have a guarantee that the user will pay them the required tokens (user's commitment is enforced).
8. The solver procures the required tokens on the target chain and deposits them into the users account on the target chain without waiting for settlement or finality.
9. The solver can include the UserOp (token deposit in escrow) eventually.

[

diagram-21

1747×880 32.9 KB

](https://ethresear.ch/uploads/default/original/3X/9/a/9a635cedc8a0c6c682e3e59d57418e58e098077e.png)

As we can see in the above examples a user commitment via nonce lock is same as delegating eventual state update to solvers. This reduces latency introduced by finality especially in a cross-chain setting.

What are the latency and consistency guarantees of this account model?

wip

Does OneBalance remove the need for bridges?

No.

Bridges are necessary mechanisms to provide fulfillment of locks and settlement of inventory outside of the critical path of user request execution.

What is the relationship between Frontier Research, OneBalance, and the CAKE Working Group?

Frontier Research is and remains an independent research and advisory group formed to bridge the gap between fundamental research and commercial products.

Frontier founded the CAKE Working Group along with other collaborators such as Anoma, Across, and Ethereum Foundation members to foster conversation around common interfaces and language to accelerate the development of chain abstraction technology.

Frontier is spinning out OneBalance as an independent project to accelerate the transition of an account centric worldview for web3.

All three groups of humans will continue to follow their individual missions moving forward.