OneBalance

Speed running - Chain Abstraction ERCs

Outline

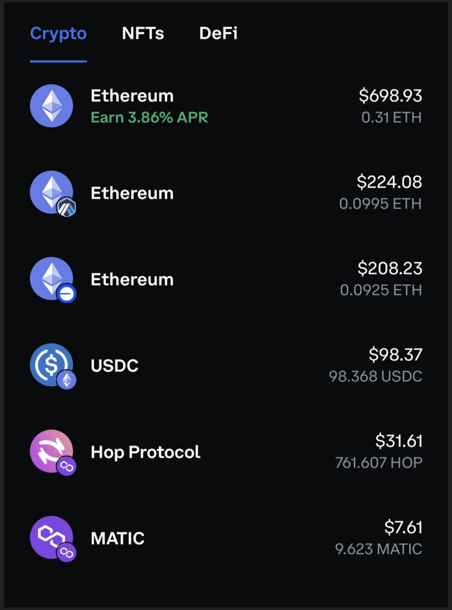
- The United Blockspaces of Ethereum
- Permission Layer ERCs
- dAPP <> Permission Layer ERCs
- Permission Layer <> Solver Layer ERCs
- Solver ERCs <> Settlement Layer ERCs

The United Blockspaces of Ethereum

The United Blockspaces of Ethereum

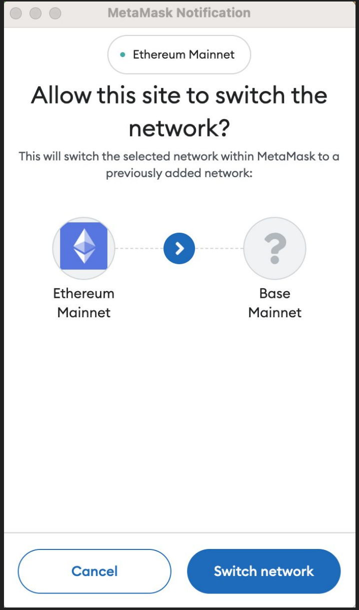
- L2 centric roadmap brought us many blockspaces of Ethereum
- Some are red and some are blue, but all are ethereum
- But they don't feel like it

The United Blockspaces of Ethereum - but they don't feel like it



Crypto NFTs DeFi

Ethereum Earn 3.86% APR	$698.93 0.31 ETH
Ethereum	$224.08 0.0995 ETH
Ethereum	$208.23 0.0925 ETH
USDC	$98.37 98.368 USDC
Hop Protocol	$31.61 761.607 HOP
MATIC	$7.61 9.623 MATIC



MetaMask Notification

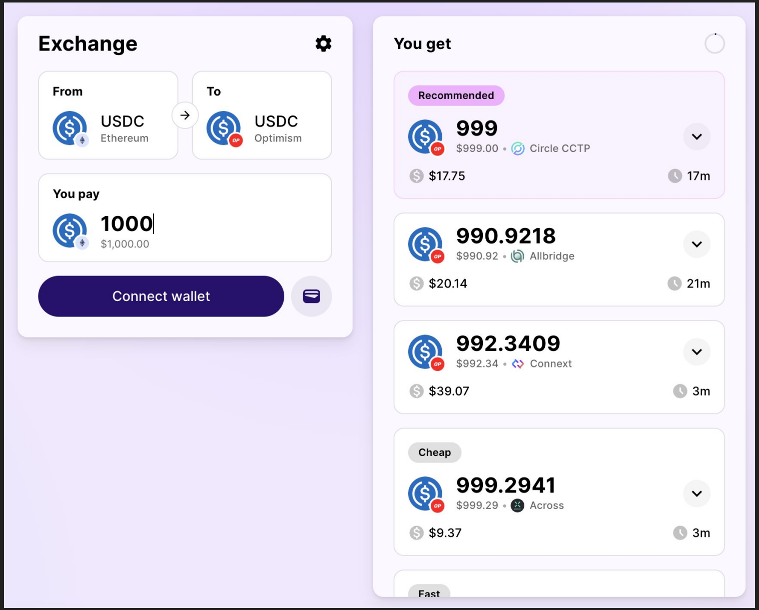
Ethereum Mainnet

Allow this site to switch the network?

This will switch the selected network within MetaMask to a previously added network:

Ethereum Mainnet → Base Mainnet

Cancel | Switch network



Exchange

From: USDC Ethereum → To: USDC Optimism

You pay: 1000 ($1,000.00)

Connect wallet

You get

Recommended
999
$999.00 • Circle CCTP
$17.75 · 17m

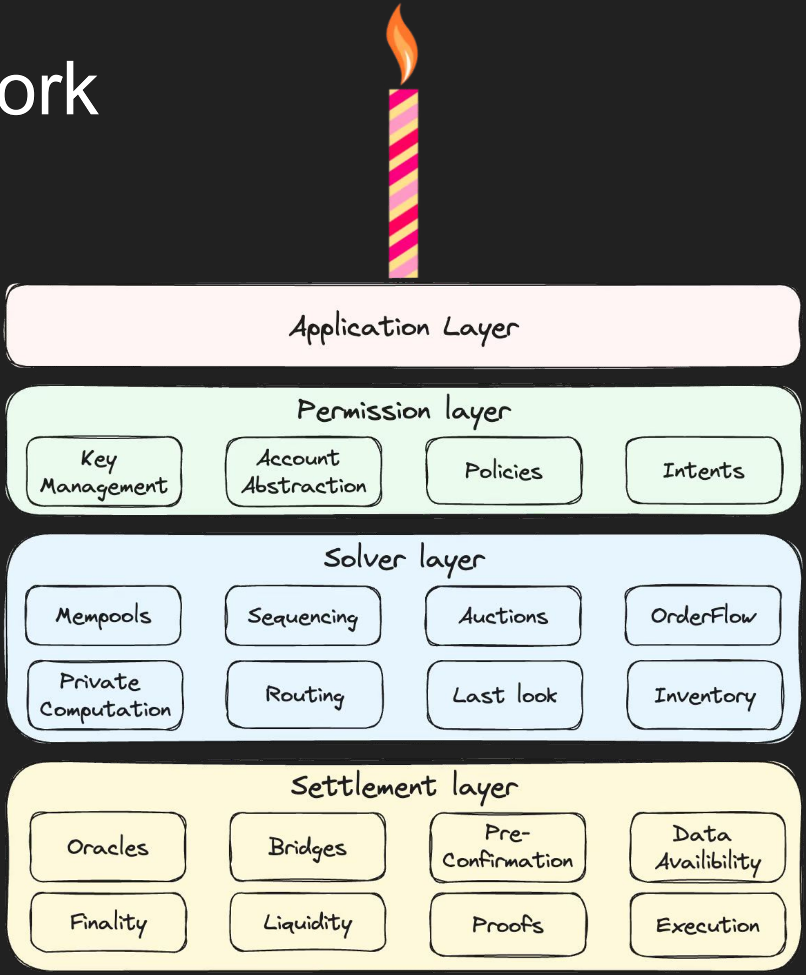
990.9218
$990.92 • Allbridge
$20.14 · 21m

992.3409
$992.34 • Connext
$39.07 · 3m

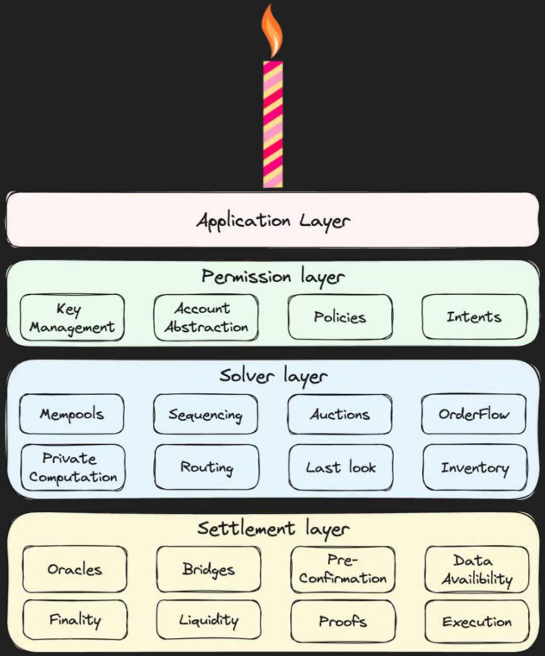
Cheap
999.2941
$999.29 • Across
$9.37 · 3m

Fast

CAKE framework



Core trade-offs



	Core trade-off
Permission layer	UX vs Agency
Solver layer	Optimal routing vs Execution Guarantee
Settlement layer	Low fees vs Execution Speed

Chain abstraction - expected end state

- Remove the words gas, chain and bridges from the user journey
- dApp requests effective balance on the target chain
- Dapp constructs the call data to execute on the target chain
- Other parts of the stack figure out how to bring the expected funds on the target chain
- Users can hold funds on different chains and bring them to target chain at execution time

Permission Layer ERCs

ERC-4337

- Decreased off-chain validation complexity
- Gas sponsorship is specified by the Paymaster
- Msg.sender remains the user
- DoS protection
  - A too high gasLimit is penalized
  - Specific opcodes are forbidden cause their values depend on state transitions happening.
  - Throttling up to banning
  - Stake as sybil protective measurement

- *IEntryPoint* (All UserOps are submitted here)
  - handleOps(...)
  - getNonce(...)
  - addStake(...)
  - unlockStake(...)
  - withdrawStake(...)
  - depositTo(...)
  - balanceOf(...)
  - withdrawTo(...)
  - simulateValidation(...)
- *IAccount*
  - validateUserOp(...)
  - executeUserOp(...)
- *IPaymaster*
  - validatePaymasterUserOp(...)
  - postOp(...)

Modularity

ERC-7579

- Highest adoption
- Module registry (*ERC-7484*)
- Optional hooks
- *ERC-1271* support
- Simplification of *ERC-6900*

ERC-6900

- First modular SCA standard
- Higher complexity as *ERC-7579*
- Optional hooks
- *ERC-1271* support

Key concepts:

- Validator module
- Executor module
- Pre & Post hook

Permissions

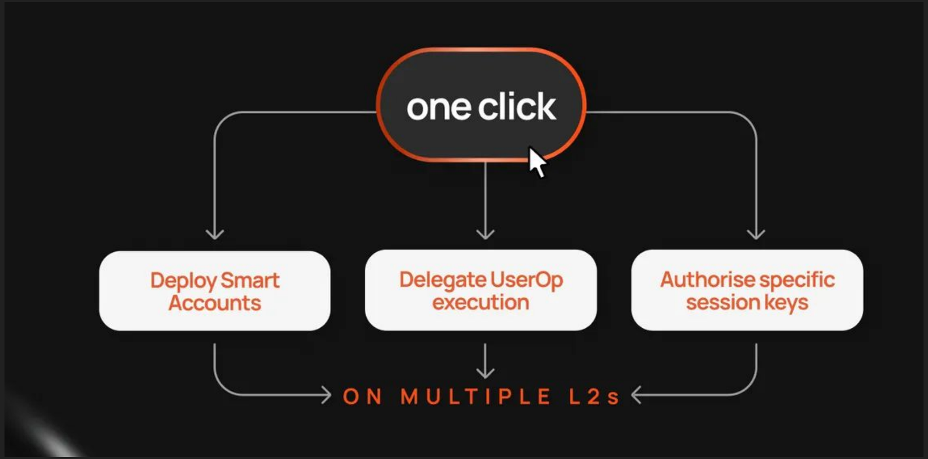
ERC-7710

- On-chain permission management
  - In essence an executor with validation logic.
- Simple interface
  - *ERC7710.redeemDelegation(...)*
  - Adding and removing of delegators and pre-/post-hooks is not specified.

Session Keys

- *ERC-7763*
  - Issue app specific keys
  - Key derivation standard
- Policies
  - Off-chain:
    - Admin key approves an app key PK through a signature including a policy.
  - On-chain:
    - App-Key will get registered on chain together with a policy.

Merkelized signatures - Not a standard



dAPP <> Permission Layer ERCs

Chain specific addresses - No standard yet

- Asset centric view, i.e. a bridge is as easy as a transfer. Across ecosystems.
- No selecting chain id: Solved via chain specific addresses: 0xaddresshash@optimism.eth
- Every chain has an ENS address, wallets can resolve the chain information via that ENS information

Authentication

ERC-7555

- The goal of this standard is to prevent the fragmentation of accounts because an application couldn’t identify an already existing SCA.

- This problem appears if other key pairs as *secp256k1* ones are used for the signing process.

- It specifies the same flow as we know from current SSO providers.
  - Authenticate and redirect back to the app.
  - Send transaction and redirect back to the app.

Routes

- /auth
  - Parameters
    - redirect_uri
    - chain_id
  - Response
    - smart_contract_address
  - **provider.io/auth/?redirect_uri=””&chain_id=””**
- /sendTransaction
  - Parameters
    - redirect_uri
    - chain_id
    - transaction
  - Response
    - smart_contract_address
    - tx_hash
  - **provider.io/sendTransaction/?smart_account_address=””&tx_hash=””**

Sessions

CAIP-25

- Standardizes for chain agnostic wallets the creation of a session and keeping the authorization persistent.
- It is required to pass scopes defined in *CAIP-217*.
- JSON-RPC method:
  - wallet_createSession
    - requiredScopes
    - optionalScopes
    - scopedProperties
    - sessionProperties

CAIP-285

- Standard to revoke an active *CAIP-25* session.
- JSON-RPC method:
  - wallet_revokeSession
    - sessionId (optional; *CAIP-171*)

Sessions

CAIP-311

- Standardizes an event to inform the caller about a changed *CAIP-25* session.
- JSON-RPC method:
  - wallet_sessionChanged
    - sessionId (optional; *CAIP-171*)
    - sessionScopes
      - Updated scopes.

CAIP-312

- Standardizes the method to retrieve the details of an active *CAIP-25* session.
- JSON-RPC method:
  - wallet_getSession
    - sessionId (optional; *CAIP-171*)

Calls

EIP-5792

- This standards introduces new wallet JSON-RPC methods that allow to batch calls and define the chainID for each call.
- It does make the wallet<> DApp interaction chain abstracted through it.

- wallet_sendCalls
  - Group an arbitrary amount of calls into one request.
  - Capabilities feature
    - paymasterService (*ERC-7677*)
    - Atomic execution
    - …
- wallet_getCallsStatus
- wallet_showCallsStatus
  - Triggers the wallet to show a status
- wallet_getCapabilities
  - Returns a list of wallet features/capabilities

Calls

CAIP-27

- This standard defines a JSON-RPC method to invoke methods together with a *sessionID* and a valid *scopeObject* (*CAIP-25*).
- In short: It enables chain-abstracted interactions with a wallet.

- wallet_invokeMethod
  - **sessionId -** The *CAIP-25* session ID
  - **Scope -** The *CAIP-2* identifier
  - **request -** JSON-RPC request
    - method
    - params

Assets

ERC-7811

- Allows to request a wallet for the users balances for specific assets.

- Introduces a new JSON-RPC method wallet_getAssets:
    - account - Address
    - requiredAssets - Address[]

- Returns a list of objects including:
    - address
    - balance
    - type
    - metadata
        - name
        - symbol
        - decimals

Permission layer <> Solver layer ERCs

Intent framework

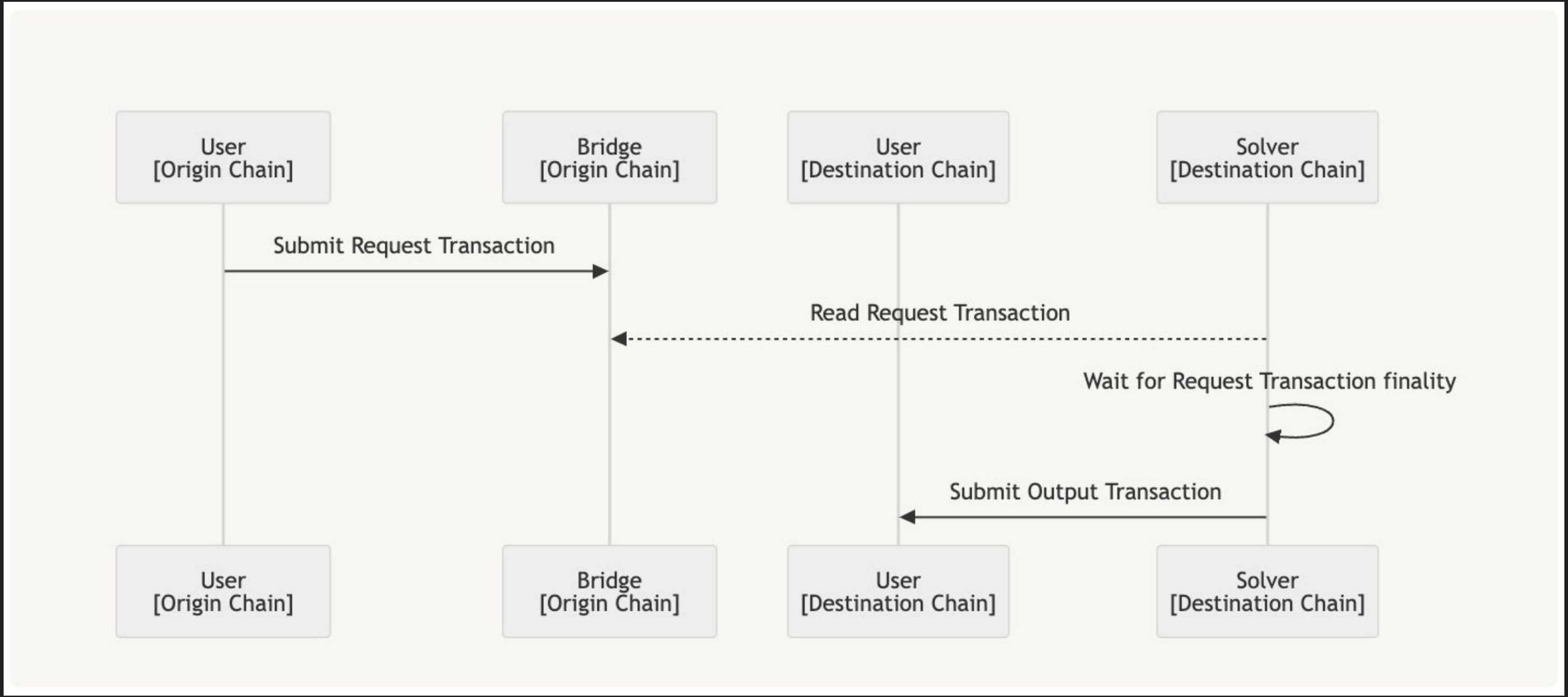
ERC-7683

- Lets wallet specify a set of input and output conditions on their request
- Limited to the cross-chain value transfer use case
- Defines input, output, success conditions and any and subsequent calls to be made

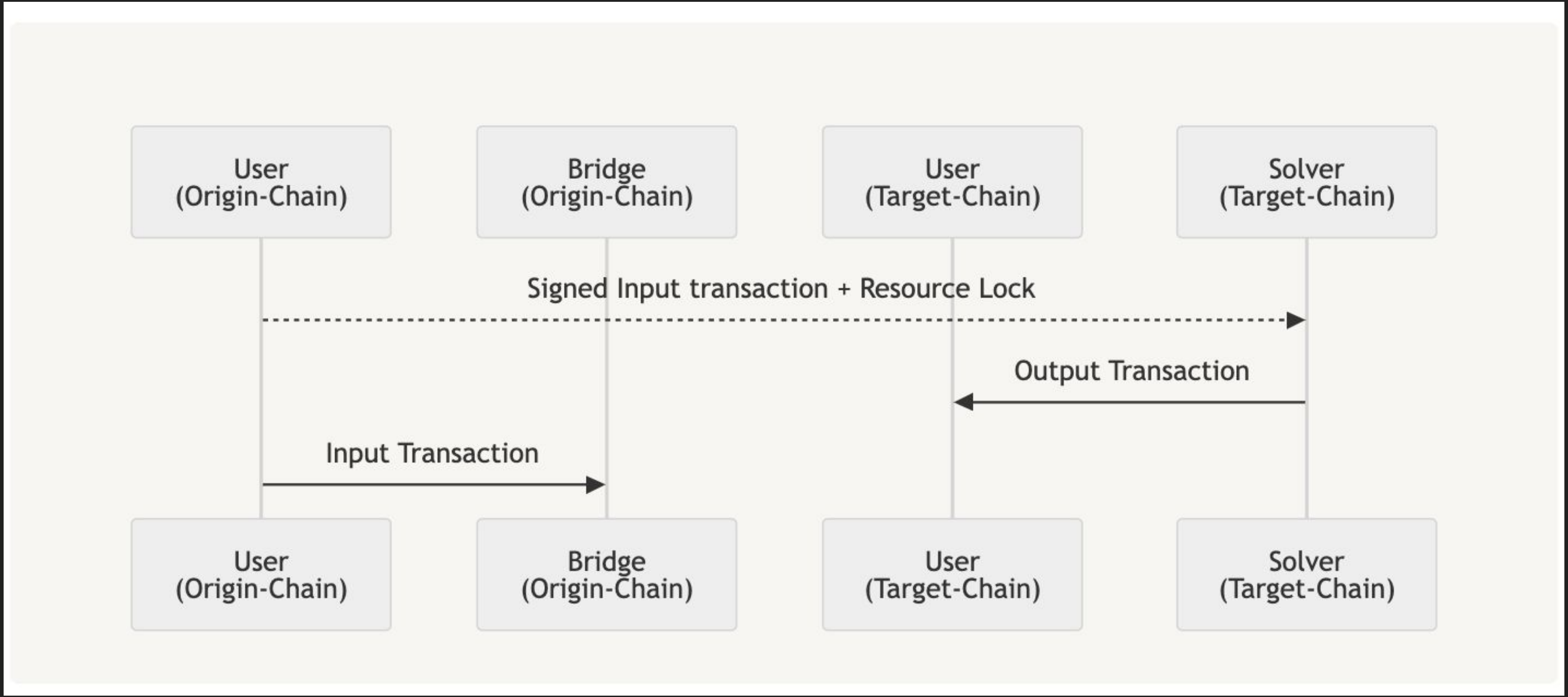
Standard

```
●     struct ResolvedCrossChainOrder {
●             address user;
●             uint256 originChainId;
●             uint32 openDeadline;
●             uint32 fillDeadline;
●             bytes32 orderId;
●             Output[] minReceived;
●             FillInstruction[] fillInstructions;
●     }
      struct Output {
●             bytes32 token;
●             uint256 amount;
●             bytes32 recipient;
●             uint256 chainId;
●     }
●
```

Resource locks - No standard yet



Resource locks - No standard yet



- Asynchronous cross-chain calls

Demo time

https://x.com/ankitchiplunkar/status/1854945676951797772

https://teleport.onebalance.io/

Solver Layer <> Permission layer standards

Bridge

ERC-7786

- An abstract messaging protocol respectively standard for contracts to improve the interoperability and composability.

- It will enable the possibility to have cross-chain deployed protocols with global state. While using N bridge solutions.

- Provides the possibility to aggregate bridge oracle networks which increases security.

- *IERC7786GatewaySource*
  - sendMessage(...)
- *IERC7786Receiver*
  - executeMessage(...)
- *IERC7786GatewayDestinationPassive*
  - setMessageExecuted(...)

Bridge

ERC-7802

- Standard to implement the mint/burn logic of bridges.
- This standard will extend *ERC20* and token issuers can approve bridges.
- *IERC7802*
  - crosschainMint(...)
  - crosschainBurn(...)

ERC-7281

- Standard to implement bridging of tokens while keeping the canonical representation of it known.
- *IXERC20*
  - setLockbox(...)
  - setLimits(...)
  - mintingMaxLimitOf(...)
  - burningMaxLimitOf(...)
  - mintingCurrentLimitOf(...)
  - burningCurrentLimitOf(...)
  - mint(...)
  - burn(...)
- *ILockbox*
  - deposit(...)
  - withdraw(...)

chain abstraction toolkit

for apps

- *Chain / Bridge / Gas free UX*
- *Aggregate liquidity everywhere*
- *Teleport users between chains*
- *Monetize your orderflow*
- *Modular WaaS*

Try chain abstracted swaps:
app.onebalance.io

resource locks as a service

for infra

- *Reorg and equivocation protection*
- *Trust minimized destination chain execution*
- *Composable intents*
- *Modular Solvers*

Try resource lock bridging:
teleport.onebalance.io

Thank you