GPv2Settlement

Architecture

A settlement comprises of:

- A list of traded tokens with their corresponding price in the batch
- A list of trades to execute
- A list of interactions

A solver monitors on-chain liquidity and on receiving batch-auction instances from the Protocol, it determines which orders can be matched, the clearing prices of the settlement, and what extra liquidity is necessary from the blockchain.

Normally, all orders in a settlement are settled withuniform clearing prices , which means that every user receives the same price for the same token.

If the transaction does not revert, each order will be executed during a settlement and the user will receive the desired token after the settlement.

Guarantees and Invariants

The parameters of the order are verified for validity:

- The signature for the order matches the user's address
- The order is not expired
- The order was not previously filled
- The current prices are equal to or better than what is specified in the order

Trades

Trades contain a description of the users' orders, a signature for verifying its validity, and the executed amount (for partially fillable orders).

The contract decodes the order parameters from the trade.

Interactions

Interactions allow solvers to execute arbitrary calls to any on-chain contract. Normally, they are used to interact with other on-chain liquidity providers, for example, to make a swap call to Uniswap.

Protocol fee collection Interactions are also used for accounting / bookkeeping purposes as well. As trades are executed, the Protocol collects a fee from each trade and stores this in the settlement contract (known asinternal buffers). At regular intervals, the Protocol withdraws the fees from the settlement contract to the Protocol's treasury Safe. tip Solvers areNOT able to directly accessuser funds through interactions. There are strict guarantees enforced by the settlement contract over movement of user funds. caution Allowing interactions by solvers, creates the possibility of malicious solvers that can steal funds from the settlement contract. This is why the Protocol only allows interactions to be executed by allow-listed solvers, and requires that the solver post a bond to the Protocol before it can be allowed.

Data Types and Storage

GPv2Order.Data

struct

TheGPV2Order.Data is the one of the most important data structures in the Protocol. It:

- defines the parameters of an order
- is the basis of which the order digest is determined usingEIP-712
- and subsequentlysigned
- by the user

In code, it is defined as:

struct

Data

{ IERC20 sellToken ; IERC20 buyToken ; address receiver ; uint256 sellAmount ; uint256 buyAmount ; uint32 validTo ;

bytes32 appData ; uint256 feeAmount ; bytes32 kind ; bool partiallyFillable ; bytes32 sellTokenBalance ; bytes32 buyTokenBalance ; } Field Description sellToken ERC-20 token sell buyToken ERC-20 token to buy receiver The address that will receive the proceedings of the trade. If this field isaddress(0) (i.e. the zero address0x00...0), then the user who signed the trade is going to receive the funds. sellAmount Amount ofsellToken that is sold in wei. buyAmount Amount ofbuyToken that is bought in wei validTo UNIX timestamp (in seconds) until which the order is valid appData Extra information about the order. Not enforced by the smart contract outside of signature verification (may be used for referrals etc). feeAmount Amount of fees paid insellToken wei kind buy orsell partiallyFillable partially fillable (true) or fill-or-kill (false) sellTokenBalance From where thesellToken balance is withdrawn buyTokenBalance Where thebuyToken is deposited Balance locations ThesellTokenBalance andbuyTokenBalance fields thekeccak256 hash of the balance location. The following table describes the possible values and their meaning:

Value Description erc20 User'sERC-20 balance via approvals given to the GPv2VaultRelayer (default) external User'sERC-20 balance via approvals given to the Balancer vault internal User's internal Balancer vault balance

orderUid

TheorderUid is a unique identifier for an order. It is 56 bytes and defined as:

orderUid = orderDigest ‖ owner ‖ validTo Where:

- orderDigest
- is theEIP-712
- digest of theGPv2Order.Data struct
- (32 bytes)
- owner
- is the address of the order owner (20 bytes)
- validTo
- is the timestamp until which the order is valid (4 bytes)
- ‖
- is the concatenation operator

filledAmounts

ThefilledAmounts mapping stores the amount of an order that has been filled. It is defined as:

mapping (bytes

=>

uint256)

public filledAmounts ; The key is theorderUid and the value is the amount of the order that has been filled.

caution DoNOT rely on thefilledAmounts mapping to determine if an historical order has been filled. This is because the respective slot in the mapping may be freed by a solver after the order has expired to save gas.

Functions

This section will cover the main functions that are used by users and solvers.

For users

setPreSignature

This function allows a user to pre-sign an order, which can be used if for some reason the user is unable to sign witheth_sign ,EIP-712 , orERC-1271 . This is most useful for smart contracts that have neither a private key, nor implementERC-1271 , but still want to use CoW Protocol.

function

setPreSignature (bytes

calldata orderUid ,

bool signed)

external ;

invalidateOrder

This function allows a user to invalidate (cancel) an order:

function

invalidateOrder (bytes

calldata orderUid)

external ;

For solvers

settle

This function is permissioned and can only be called by solvers passing the allow-list authentication. It executes a settlement:

function

settle (IERC20 [ ]

calldata tokens , uint256 [ ]

calldata clearingPrices , GPv2Trade . Data [ ]

calldata trades , GPv2Interaction . Data [ ] [3]

calldata interactions)

external nonReentrant onlySolver ;

Indexing

Events that are indexed are:

- Trade
-
  - on any trade
- Interaction
-
  - on any interaction
- Settlement
-
  - which solver executed the settlement
- OrderInvalidated
-
  - when an order is invalidated
- PreSignature
-
  - when pre-signing or revoking a pre-signed signature

Off-chain

As this is the main contract of CoW Protocol, all the off-chain infrastructure is built around it. This includes:

- The Protocol
- Solvers Edit this page Previous Core Next GPv2VaultRelayer