Kwenta Paymaster Audit v1.0.0

Audit by: Jared Borders

Detailed Findings

The following sections include in-depth descriptions of the audit findings.

(C) Critical Risk

> *The issue must be fixed immediately, as significant funds/assets are at risk.*

(H) High Risk

> *The issue must be addressed promptly. Failure to do so could result in the loss of funds/assets or deviation from the provided specifications.*

(M) Medium Risk

> *It is strongly recommended to fix the issue, as its implications could significantly impact the project, though not to an existential extent.*

(L) Low Risk

> *The risk is minor and potentially unlikely or irrelevant, but it is noteworthy.*

(L-1) `percentageMarkup` Sanitization

Recommendation

If `percentageMarkup` is set to a value less than `100`, the `getCostOfGasInUSDC()` function will return a value strictly less than the actual cost of gas. This could lead to a situation where the user is charged less than the actual cost of gas. To prevent this, the `percentageMarkup` setter should be restricted to values greater than or equal to `100`.

(Q) Code Quality

> *There is no immediate risk, but fixing the issue would improve code quality, better conform to standards, and potentially reduce unperceived future risks.*

(Q-1) NatSpec: MarginPaymaster Immutables

Recommendation

Add NatSpec describing each immutable/constant value.

(Q-2) Separate Immutables and Constants

Recommendation

Add a `CONSTANTS` section.

(Q-3) NatSpec: MarginPaymaster Functions

Recommendation

Add NatSpec describing each function, its parameters, and return values.

(Q-4) NatSpec: MarginPaymaster Modifiers

Recommendation

Add NatSpec describing the modifier(s).

(Q-5) `getUSDCAvailableInWallet()`'s `balance` And `amount` Not Used

Recommendation

Remove `balance` and `amount` from `getUSDCAvailableInWallet()`.

(Q-6) `USDC_TO_SUSDC_DECIMALS_INCREASE` Duplicate Constant

Recommendation

Use Zap's `_DECIMALS_FACTOR`, which is already defined.

(Q-7) Add Event to `postOp()`

Recommendation

Add an event to `postOp()` that emits only when `withdrawn` is zero; this will surely help in debugging/monitoring later.

(Q-8) Naked Value in `getCostOfGasInUSDC()`

Recommendation

Define a constant for `100` in `getCostOfGasInUSDC()` that represents "100%."

(Q-9) Inaccurate Documentation

Recommendation

Update the documentation in `getWalletAccountId()` to reflect the actual behavior (i.e., multiple accounts are now supported).

(I) Informational

> *These are warnings and considerations for protocol operation. No immediate action is required.*

(I-1) Inconsistent Formatting

Recommendation

Run `forge fmt`.

(I-2) TickMath Library Unused Function(s)

Recommendation

Remove `getTickAtSqrtRatio()` from the `TickMath` library; it is not used in the project.

(I-3) OracleLibrary Unused Function(s)

- Remove `getOldestObservationSecondsAgo()` from `OracleLibrary`; it is not used in the project.
- Remove `getBlockStartingTickAndLiquidity()` from `OracleLibrary`; it is not used in the project.
- Remove `getWeightedArithmeticMeanTick()` from `OracleLibrary`; it is not used in the project.
- Remove `getChainedPrice()` from `OracleLibrary`; it is not used in the project.

(I-4) OracleLibrary Unused Struct(s)

- Remove `WeightedTickData` from `OracleLibrary`; it is not used in the project.

(I-5) FullMath Unused Function(s)

- Remove `mulDivRoundingUp` from `FullMath`; it is not used in the project.

(G) Gas Optimizations

> *The suggested optimization would save a significant amount of gas, making it worthwhile to implement despite the development cost.*

(G-1) `getCostOfGasInUSDC()`'s `gasPrice` And `gasUsed` Not Used

Recommendation

Update `if (withdrawableMargin < 0) return 0;` to check `<=` for early exit.