IDEX B-1

Security Audit

February 12, 2024
Version 1.0.0

Presented by 0xMacro

Table of Contents

Introduction

This document includes the results of the security audit for IDEX's smart contract code as found in the section titled ‘Source Code’. The security audit was performed by the Macro security team from January 12 to January 16, and on February 2, 2024.

The purpose of this audit is to review the source code of certain IDEX Solidity contracts, and provide feedback on the design, architecture, and quality of the source code with an emphasis on validating the correctness and security of the software in its entirety.

Disclaimer: While Macro’s review is comprehensive and has surfaced some changes that should be made to the source code, this audit should not solely be relied upon for security, as no single audit is guaranteed to catch all possible bugs.

Overall Assessment

The following is an aggregation of issues found by the Macro Audit team:

Severity	Count	Acknowledged	Won't Do	Addressed
Medium	1	-	-	1
Low	3	1	-	2
Code Quality	2	1	-	1

IDEX was quick to respond to these issues.

Specification

Our understanding of the specification was based on the following sources:

- Discussions on Slack with the IDEX team.
- Available documentation in the respective PRs.

Source Code

The following source code was reviewed during the audit:

- **PR 85:** /idex-contracts-ikon/pull/85
- Commit Hash: `9d53426e2508587671ca891602f3262b76703d63`
- **PR 86:** /idex-contracts-ikon/pull/86
- Commit Hash: `6c713066070ae62e163150585431096779030289`
- **PR 87:** /idex-contracts-ikon/pull/87
- Commit Hash: `cc2a38c738828d945348c90421e44bd201edaf55`
- **PR 88:** /idex-contracts-ikon/pull/88
- Commit Hash: `57f90117f408f3dccf17fb28a15be18bc52b9b69`
- **PR 89:** /idex-contracts-ikon/pull/89
- Commit Hash: `789f07db872d53776a2dc29fbbacbf60bbefe1ab`
- **PR 90:** /idex-contracts-ikon/pull/90
- Commit Hash: `612e842fca36031f7423e0497a0eededa8561b4a`
- **PR 91:** /idex-contracts-ikon/pull/91
- Commit Hash: `9fce821850ef0f981e055ec05f3d8dc3f649bd1d`
- **PR 93:** /idex-contracts-ikon/pull/93
- Commit Hash: `691e9b9e8f181edc1686569c5bac1509b8a4b14a`
- **PR 94:** /idex-contracts-ikon/pull/94
- Commit Hash: `c6d611f5739b826606813bca5d4bda7815b445e0`

Changes for PR 85

Contract	SHA256
contracts/libraries/BalanceTracking.sol	5e41d4d36f4392603c161b2cb0158ad249654ad02cb02fb9ed6f7f1d349b68e5
contracts/libraries/IndexPriceMargin.sol	f676496498b45adfb30b7b6a60d0156d01f95b01a9829f96ab38b1c101034585
contracts/libraries/Trading.sol	a731c7341ca15f2e0b3795f9f1079c9a4422d9c1ad623ea8558a32cc7995ad6e

Changes for PR 86

Contract	SHA256
contracts/bridge-adapters/ExchangeStargateAdapter.sol	a181ec6fe79ccfb4fb5fcc9f55a9c77be926174897e218eeaa9ee4bf0c3f5208
contracts/libraries/Constants.sol	80650b18d45190907521cfdf0a7c34dd4cccbaada979c61321b1ca9078093db8
contracts/libraries/Hashing.sol	67e0fd2b891d02e0367dc5097abb47833385ee37c2b5fbe8029057c4b83e378f
contracts/libraries/IndexPriceMargin.sol	f7c94e762fbf3e74af52940e43ed4f866e341f1e105bcd4d135ceca1245098ed
contracts/libraries/PositionBelowMinimumLiquidation.sol	b917dee7d20ed67705d7d48a93054ee44040af2f278c3ed0cc16901fc6c85018
	5056b4269f687d03dc2330916a4147c

contracts/libraries/Structs.sol	912809d611fe8b433d46fbc44b0e9cdb2
contracts/libraries/TradeValidations.sol	c46b8e6b678a2f2ef735932e9fca526de5e89f8b05ee4b79135769dd17eb4c34
contracts/libraries/Withdrawing.sol	2608c3a67e834986526ae6d672009960bf80a582fae08e871f1d780a99b0a1b6

Changes for PR 87

Contract	SHA256
contracts/Exchange.sol	220e0628420a810b0926c01e295206edb1fd40a614e3cedd381f7537542a4786
contracts/libraries/Depositing.sol	1cd0480b5dd633e680ec5c39864e856d750ffe044bcc52ee9baf4186d02ddb32
contracts/libraries/IndexPriceMargin.sol	9489e5bb38976aa081d2feccb536df167e9a03fb54f99c76e4709976a04946d9
contracts/libraries/OraclePriceMargin.sol	4872aa093b59e1a054ff111d7eeff75490c57251ece62fcad5f3b0a27014eae3
contracts/libraries/Withdrawing.sol	aa241dd4dc79a78983795b97cdffce3382879a8fcfa51f4b35fb868c0d0675f5
contracts/util/ExchangeWalletStateAggregator.sol	5f98e19a1c2566963c48f821822e6f90c140f39303b0d36c766e10dd0993af06

Changes for PR 88

Contract	SHA256
contracts/libraries/IndexPriceMargin.sol	d73b0d1b1e39c7f41ab3e05d58ec12a651c79daa37e12bae6511e92536be351a
contracts/libraries/LiquidationValidations.sol	e30eeb22fd47afbc8513d03853940de6d666d965f6814185c00e9450c9c1e76b
contracts/libraries/Structs.sol	5f1cb7d6a231a275203886c46cb5e795008f3ecb74c470a946981f506141c2e0
contracts/libraries/WalletExitAcquisitionDeleveraging.sol	4226aaf2e8a2254459d0d62b35e1c175ffc06ea7c8a8a62183af180969f6b628
contracts/libraries/WalletInMaintenanceAcquisitionDeleveraging.sol	3e9a1b812ce2cba3de80d798958422ab56f7f80dd73226a1171c8a902c9405c5

Changes for PR 89

Contract	SHA256
contracts/bridge-adapters/ExchangeStargateAdapter.sol	a31a0b12a38bcb037887743ce6cf24501af9a60cc37902bbf22c95aa299e432c

Changes for PR 90

Contract	SHA256
contracts/bridge-adapters/ExchangeStargateAdapter.sol	a31a0b12a38bcb037887743ce6cf24501af9a60cc37902bbf22c95aa299e432c

contracts/libraries/WalletExitAcquisitionDeleveraging.sol	4f3c56a05fcd515a614344f4d9dd0097e0f5ed3dacc1cbc60e630d71c7cd767c
contracts/libraries/WalletInMaintenanceAcquisitionDeleveraging.sol	20a79c89f9074605d6a8aa8d740b9b0412e5a46b4f893109ca1bf42060c9a324

Changes for PR 91

Contract	SHA256
contracts/Exchange.sol	2125b3c945e5ab69507efb5805b54f9ea04eb895d96e58f43a35492f6343c0c6

Changes for PR 93

Contract	SHA256
contracts/EarningsEscrow.sol	7e2b6ee63afbb30a8e9251f73e22395b17d483600314858f1ecaf18f494a75bc

Changes for PR 94

Contract	SHA256
contracts/Custodian.sol	8c7dac3d90ce8cddb10a9868092c5313aab77d36b3c69b4b0f57dffecb5bc31e
contracts/EarningsEscrow.sol	485937713f9224bfa493d83a2d05ebf8a0172506f05c2e9581a559b2a5b88481

contracts/Exchange.sol	042df087b1f0cc3cac3baa1f56675d90b44f57ef2da0b4375a4ec478928ab76f
contracts/Governance.sol	78224a4abe8c13da328cc879e7efac635c0e70715073ea71267faf79dd37444e
contracts/Owned.sol	47f001901e3f0ca64439373348ca806439863498b170a6fd73bdb8420704dd1f
contracts/asset-migrators/USDCeMigrator.sol	44ad7e9832f6e2c47e7a5d52da3041f161824698e0a5f4bc7a0f41c760444dbe
contracts/bridge-adapters/ExchangeStargateAdapter.sol	134e8eafb6aaf33c59a9c63c79adecc2da83100c3735ab24d4fb7d4593131a41

Issue Descriptions and Recommendations

Click on an issue to jump to it, or scroll down to see them all.

Security Level Reference

We quantify issues in three parts:

1. The high/medium/low/spec-breaking **impact** of the issue:
    - How bad things can get (for a vulnerability)
    - The significance of an improvement (for a code quality issue)
    - The amount of gas saved (for a gas optimization)
2. The high/medium/low **likelihood** of the issue:
    - How likely is the issue to occur (for a vulnerability)
3. The overall critical/high/medium/low **severity** of the issue.

This third part – the severity level – is a summary of how much consideration the client should give to fixing the issue. We assign severity according to the table of guidelines below:

Severity	Description
(C-x) Critical	We recommend the client **must** fix the issue, no matter what, because not fixing would mean **significant funds/assets WILL be lost.**
(H-x) High	We recommend the client **must** address the issue, no matter what, because not fixing would be very bad, *or* some funds/assets will be lost, *or* the code's behavior is against the provided spec.
(M-x) Medium	We recommend the client to **seriously consider** fixing the issue, as the implications of not fixing the issue are severe enough to impact the project significantly, albiet not in an existential manner.
(L-x) Low	The risk is small, unlikely, or may not relevant to the project in a meaningful way. Whether or not the project wants to develop a fix is up to the goals and needs of the project.
(Q-x) Code Quality	The issue identified does not pose any obvious risk, but fixing could improve overall code quality, on-chain composability, developer ergonomics, or even certain aspects of protocol design.
(I-x) Informational	Warnings and things to keep in mind when operating the protocol. No immediate action required.
(G-x) Gas Optimizations	The presented optimization suggestion would save an amount of gas significant enough, in our opinion, to be worth the development cost of implementing it.

Issue Details

~~M-1~~ Quote token migration can be griefed

TOPIC	STATUS	IMPACT	LIKELIHOOD
Griefing	Fixed	Medium	Medium

Reference: PR94#R83

Description

Quote token migration allows for the migration of the quote token from the initial source token to a different destination token. The protocol's intention is to initially use BWUSDC and later migrate it to the USDC.e representation, once it is in its final state.

However, the current implementation is vulnerable to a griefing attack, which allows an attacker to break the entire migration process. The problematic part lies in the `Custodian.migrateAssets` function, specifically in the following `require` statement:

```
// Entire balance must be migrated
require(
  IERC20(destinationAsset).balanceOf(address(this)) == quantityInAssetU
  "Balance was not completely migrated"
);
```

Since the balances of `destinationAsset` and the quantity sent to the `assetMigrator` must match exactly, an attacker can transfer a small amount (1 unit is enough) of the destination token to the `Custodian` contract. This action

breaks the above check and causes the transaction to revert.

The issue is classified as medium severity, as it can be remediated by upgrading (with a 3-day delay) the `USDCeMigrator` contract.

L-1 Fee-on-transfer tokens not supported in `EarningsEscrow`

TOPIC	STATUS	IMPACT	LIKELIHOOD
Protocol Design	Fixed	Medium	Low

Reference: PR93#R197

Description

The `_transferTo` function in *EarningsEscrow.sol* does not account for tokens applying transfer tax, which results in less value transferred to the `walletOrContract` address than requested.

As a result, transferring out tokens from the contract will fail on the require statement on line L196:

```
require(
  balanceAfter - balanceBefore == quantityInAssetUnits,
  "Token contract returned transfer success without expected balance ch
);
```

This is because (balanceAfter - balanceBefore) will be smaller than the quantity transferred.

Remediations to Consider

Consider handling fee-on-transfer tokens correctly by calculating `balanceAfter`

– `balanceBefore` of the EarningsEscrow contract itself as instead to the receiving wallet.

~~L-2~~ Use low-level `call` instead of `transfer` / `send` for withdrawing tokens

TOPIC	STATUS	IMPACT	LIKELIHOOD
Protocol Design	Fixed	Medium	Low

Reference: PR93#R188, PR93#L266

EarningsEscrow.sol supports earnings provided in native tokens when `assetAddress` is set to `0x0`. If so, tokens are withdrawn using the `.send` method:

```
if (asset == address(0x0)) {
  require(walletOrContract.send(quantityInAssetUnits), "ETH transfer fa
} else {
...
```

However, `send` sends a fixed amount of gas, and assumes the address it sends the token to doesn't have a `fallback` or `receive` function that runs code. If the address is a contract with a custom fallback function, like a multi-sig, then the `send` will fail, and the receiving address will be prevented from withdrawing their funds.

The same issue applies with *ExchangeStargateAdapter's* `withdrawNativeAsset` function, which uses the `transfer` function.

Remediations to Consider

Use a low-level `call` to transfer native tokens; doing so will allow any contract

or EOA to withdraw without issue.

L-3 Missing `chainId` in signature for distributing escrow earnings

TOPIC	STATUS	IMPACT	LIKELIHOOD
Protocol Design	Acknowledged	High	Low

Reference: PR93#R171-R177

The signature used in *EarningsEscrow* doesn't include a `chainId` parameter. This makes the protocol susceptible to replay attacks when the protocol is deployed to multiple chains or when a hard fork of the chain happens.

Remediations to Consider

Consider including the `chainId` to the signature message.

RESPONSE BY IDEX

> EarningEscrow contracts are only deployed by an IDEX admin with the operational practice of using a different wallet per target chain. Our operational guide for this release includes:
>
> "The EarningsEscrow contract replay mechanism only includes the contract's address. When deploying a new contract, confirm that the new contract address is not already used elsewhere, for example on another chain."

~~Q-1~~ Margin requirement change on trades is not updated in documentation

TOPIC	STATUS	QUALITY IMPACT
Documentation	Fixed	Low

Reference: PR85#R268-R302

Due to new business requirements, the margin requirements have changed. The change requires to meet `initialMarginRequirement` on trades when the buy or sell position is expanded and requires `maintenanceMarginRequirement` on trades when the buy or sell position is reduced.

Consider updating the documentation properly to reflect the changed behavior.

Q-2 Declare functions as `external` instead of `public` when not used internally

TOPIC	STATUS	QUALITY IMPACT
Best Practice	Acknowledged	Medium

Throughout the code base, a lot of functions (e.g. public functions in *Custodian*, *Exchange*, *ExchangeStargateAdapter*, ...) can be marked as `external` instead of `public` as they are not used internally. Consider changing them from `public` to `external` for clarity sake.

Disclaimer

Macro makes no warranties, either express, implied, statutory, or otherwise, with respect to the services or deliverables provided in this report, and Macro specifically disclaims all implied warranties of merchantability, fitness for a particular purpose, noninfringement and those arising from a course of dealing, usage or trade with respect thereto, and all such warranties are hereby excluded to the fullest extent permitted by law.

Macro will not be liable for any lost profits, business, contracts, revenue, goodwill, production, anticipated savings, loss of data, or costs of procurement of substitute goods or services or for any claim or demand by any other party. In no event will Macro be liable for consequential, incidental, special, indirect, or exemplary damages arising out of this agreement or any work statement, however caused and (to the fullest extent permitted by law) under any theory of liability (including negligence), even if Macro has been advised of the possibility of such damages.

The scope of this report and review is limited to a review of only the code presented by the IDEX team and only the source code Macro notes as being within the scope of Macro's review within this report. This report does not include an audit of the deployment scripts used to deploy the Solidity contracts in the repository corresponding to this audit. Specifically, for the avoidance of doubt, this report does not constitute investment advice, is not intended to be relied upon as investment advice, is not an endorsement of this project or team, and it is not a guarantee as to the absolute security of the project. In this report you may through hypertext or other computer links, gain access to websites operated by persons other than Macro. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such websites' owners. You agree that Macro is not responsible for the content or operation of such websites, and that Macro shall have no liability to your or any other person or entity for the use of third party websites. Macro assumes no responsibility for the use of third party software and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software.