Smart Contract Final Audit Report for Volmex Finance

Document Properties

Client	Volmex Finance
Title	V2 protocol Migration - Smart Contract Final Audit Report
Version (code freeze hash)	24c381b92b4b9cb271e8441f48ce3edb414fae00 d8a7a394a8f62cfa7750d941e5b6b9a2188b63ba
Author	Manmeet Singh Parmar
Auditor	Manmeet Singh Parmar

Overview

The audited commit is `24c381b92b4b9cb271e8441f48ce3edb414fae00` and all Solidity contract in the Volmexfinance/volmex-amm/contracts/protocol folder were in scope.

Project Status

Volmex Finance is a decentralized platform that implies the volatility of crypto assets using options contracts from multiple sources.
Currently, the team is planning to migrate from V1 volatility tokens to V2 volatility tokens. The amount of collateral to mint the tokens in v2 is *400* stablecoins, compared to *250* in v1.

Timeline

From 2022-11-28 To 2022-11-23

Languages

Solidity

Audit Update

The Volmex team applied several fixes and acknowledge a few issues based on the recommendations. The audit fix is `d8a7a394a8f62cfa7750d941e5b6b9a2188b63ba` and contracts are good to deploy on the public Mainnet chain.

Security Assessment

The focus of the audit was to verify that the Smart Contract System is secure, resilient, and working according to the specifications. The audit activities can be classified as

Issues

1. CRITICAL: Bugs leading to Ether or token theft, fund access locking, or any other loss of Ether/tokens to be transferred to any party (for example, dividends)

2. HIGH: Bugs that can trigger a contract failure. Further recovery is possible only by manual modification of the contract state or replacement

3. WARNINGS: Bugs that can break the intended contract logic or expose it to DoS attacks.

4. NOTES & ADDITIONAL INFORMATION: Other issues and recommendations

Security Assessment Methodology

Goals

- Testing the functionality of the Smart Contract to determine whether proper logic has been followed throughout the process.
- Analyzing the complexity of the code in-depth and detailed, manual review of the code, line-by-line.
- Analyzing failure preparations to check how the Smart Contract performs in case of any bugs and vulnerabilities.
- Checking whether all the libraries used in the code are on the latest version.
- Analyzing the security of the on-chain data.
- A full review of the contract source code. The primary areas of focus include
    - Accuracy
    - Readability
    - Sections of code with high complexity
    - Quantity and quality of test coverage

Stages of audit

- Structural Analysis
- Static Analysis
- Code Review / Manual Analysis

General Feedback

Happy to see such a quality codebase of Volmex Finance. I didn't find any critical vulnerability and was glad to have robustness across the contracts even with novel designs.

Findings List

Level	Amount
CRITICAL	0
HIGH	0
WARNING	0
COMMENTS	3 - Fixed and 2 - Acknowledged

Detected Issues

CRITICAL

Not Found

HIGH

Not Found

WARNING

Not Found

NOTES & ADDITIONAL INFORMATION

[IN01] Transfer collateral tokens to itself when calling to migrateToV2

When the **migrateToV2** function is called via the caller there is a collateral token transfer from volmexProtocol to volmexProtocol which result in waste of gas. The same transfer can

be seen in the transaction below:

[0x3b7a3237640820ae7585be125ab28b0fa561aacce81d8b6c7bcb8a1c3aa53d22]

Recommendation:
The transfer of token should be restricted in the **_redeem** function when **redeemSettled** is called via **migrateToV2**.

Status: **Fixed**

[IN02] Minting of new EVIV and IEVIV tokens are done via VolmexProtocolWithPrecision contract at the time of migrateToV2

The new EVIV and IEVIV tokens are minted to the user via *volmexProtocolWithPrecision*. When resulting in four external calls:

- EVIV and IEVIV Tokens get minted to volmexProtocolWithPrecision.
- volmexProtocolWithPrecision transfer EVIV and IEVIV to the caller.

The transfer of tokens can be done in two external calls.

Recommendation:
In **migrateToV2** function instead of calling **collateralize** function of the V2 protocol there should be a call to a new method **collateralizeFor(**collQtyToBeRedeemed, receiver) which will be similar to **collateralize** but the minting of new tokens (EVIV and IEVIV) will be sent to the *receiver* address, not to *msg.sender.*

This help to reduce the transfer of tokens via volmexProtocolWithPrecision and results in gas reduction.

Status: **Acknowledged**

[IN03] Natspec comments missing one input param

volmexProtocolV1.redeemSettled method has three input params but in Natspec only two params are been explained.

Recommendation:
Explain the third param *receiver* definition in the Natspec comments.

Status: Fixed

[IN04] Unnecessary safeMath Wrappers in division operation

unnecessary use of inbuilt safeMath in **_redeem** and **collateralize** while dividing by 10000 which results in more gas utilization.

Recommendation:
Always use an unchecked flag for the division by constant to remove safeMath wrapper.

Status: Fixed

[IN05] Upgrade Openzepplin contract to new stable version

The old Openzepplin version - 4.4.0 been used in the volmexIndexFactory contract and the new version - 4.8.0 comes with many fixes and gas optimization in various library/contracts like clones, ERC20, etc.

These libraries have dependencies in the volmexIndexFactory and have been used to deploy the volatility and InverseVolatility tokens.

Recommendation:
Recommend using the new version - 4.8.0 dependencies for the contracts which result in a reduction of the gas cost of transactions.

Status: Acknowledged

Conclusion

No critical, High, and warnings were found. Changes proposed for comments are implemented or acknowledged by the Volmex Team.