SOLIDProof
Bring trust into your projects

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY

Audit

Security Assessment
26. November, 2021

For

Balincer

Disclaimer

SolidProof.io reports are not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team. SolidProof.io do not cover testing or auditing the integration with external contract or services (such as Unicrypt, Uniswap, PancakeSwap etc’...)

SolidProof.io Audits do not provide any warranty or guarantee regarding the absolute bug- free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. SolidProof Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

SolidProof.io Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology. Blockchain technology and cryptographic assets present a high level of ongoing risk. SolidProof’s position is that each company and individual are responsible for their own due diligence and continuous security. SolidProof in no way claims any guarantee of security or functionality of the technology we agree to analyze.

Version	Date	Description
1.0	26. November 2021	• Layout project • Automated- /Manual-Security Testing • Summary

Network
Binance Smart Chain (BEP20)

Website
https://www.balincer.network/

Telegram
https://t.me/Balincer_Global

Twitter
https://twitter.com/BalincerNetwork

Github
https://github.com/balincer-network

Medium
https://medium.com/@balincer247

Description

Balincer is the first AMM-based decentralized margin trading platform on multi-chains, where users can easily earn interest through lending and perform leveraged trading. Powered by its own margin pool and by integrating with external AMM's like Uniswap, Balincer can significantly increase your trade efficiency and asset utilization. With Balincer, simplified margin trading and visualized position management are available for traders in the DeFi space.

Margin positions are realized through a decentralized margin pool. By putting up a margin deposit on Balincer, traders are able to open a position with up to 5X leverage. Other than serving as collateral, your margin deposit will also earn interest for you, which makes Balincer different from centralized exchanges.

Project Engagement

During the 23rd of November 2021, **Balincer Network Team** engaged Solidproof.io to audit smart contracts that they created. The engagement was technical in nature and focused on identifying security flaws in the design and implementation of the contracts. They provided Solidproof.io with access to their code repository and whitepaper.

Logo

Contract Link

v1.0

TBA

Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
Critical	9 - 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
High	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon aspossible.
Medium	4 – 6.9	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
Low	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
Informational	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
    - i) Review of the specifications, sources, and instructions provided to SolidProof to make sure we understand the size, scope, and functionality of the smart contract.
    - ii) Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
    - iii) Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to SolidProof describe.

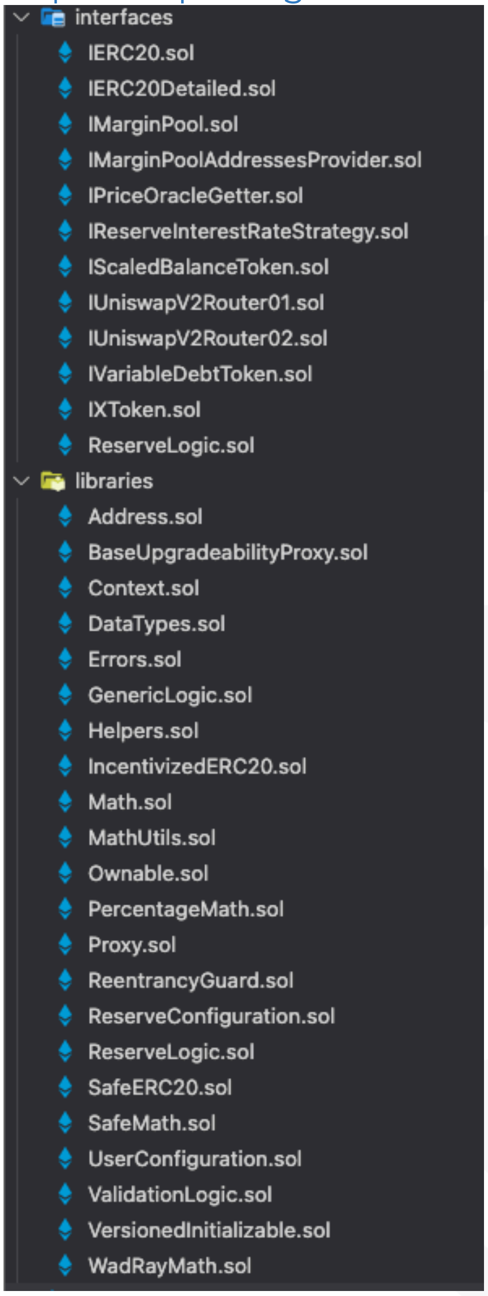
2. Testing and automated analysis that includes the following:
    - i) Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
    - ii) Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.

3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.

Used Code from other Frameworks/Smart Contracts (direct imports)

Imported packages:



interfaces
- IERC20.sol
- IERC20Detailed.sol
- IMarginPool.sol
- IMarginPoolAddressesProvider.sol
- IPriceOracleGetter.sol
- IReserveInterestRateStrategy.sol
- IScaledBalanceToken.sol
- IUniswapV2Router01.sol
- IUniswapV2Router02.sol
- IVariableDebtToken.sol
- IXToken.sol
- ReserveLogic.sol

libraries
- Address.sol
- BaseUpgradeabilityProxy.sol
- Context.sol
- DataTypes.sol
- Errors.sol
- GenericLogic.sol
- Helpers.sol
- IncentivizedERC20.sol
- Math.sol
- MathUtils.sol
- Ownable.sol
- PercentageMath.sol
- Proxy.sol
- ReentrancyGuard.sol
- ReserveConfiguration.sol
- ReserveLogic.sol
- SafeERC20.sol
- SafeMath.sol
- UserConfiguration.sol
- ValidationLogic.sol
- VersionedInitializable.sol
- WadRayMath.sol

Tested Contract Files

This audit covered the following files listed below with a SHA-1 Hash.

A file with a different Hash has been modified, intentionally or otherwise, after the security review. A different Hash could be (but not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of this review.

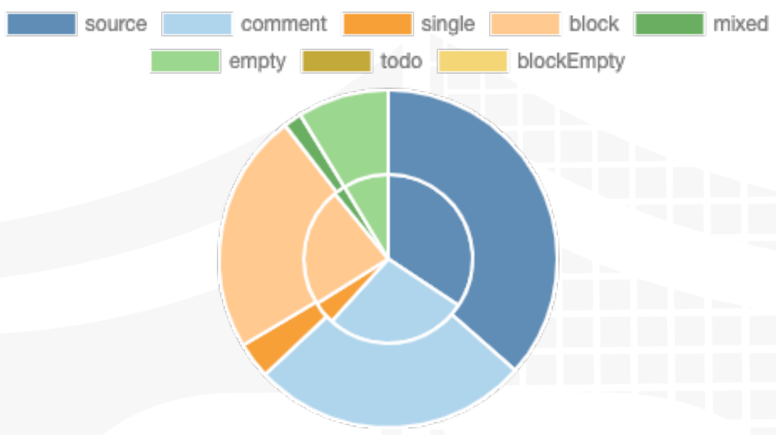
v1.0

File Name	SHA-1 Hash
contracts/MarginPoolStorage.sol	41dfb1dd0aaa79c7dcff81b0e1302aaa15d0cd83
contracts/OrderBook.sol	aab3d26051beb1c7c36d66f798bde091d6a48f87
contracts/ValidationLogic.sol	e38cfc752000b7401d058034b4c3d708795e67df
contracts/default-reserveIRS.sol	21fd7471db7615e37ee1160d0f8509d33e5b069c
contracts/balincer-token.sol	7acbf5d0c8aa526a645f03120d6c9704513b08e8
contracts/MarginPoolConfigurator.sol	191049146dcea65a2331b03b067cd08810c953af
contracts/MarginPoolCollateralManager.sol	a13e5a5aa98da1ed3a36e49e6d11f0e0421dc8c8
contracts/balincer-marginpool.sol	56f0ddd974294713d11adaa1f97959ebc60d385f
contracts/twitter-pool.sol	be378ae24b59082b8f9440eccb40ffa677f8aee1
contracts/balincer-ltoken.sol	6b1e1542d19fda98cd4169c900654453f03ecc20
contracts/ReserveLogic.sol	793cfac7411bc329db6e5efc3f52383dcfb55ba8
contracts/GenericLogic.sol	0eacc0fde3e599be87f5652e8ca13c2dec7ff2bf
contracts/balincer-xtoken.sol	41d8f986b67d943a293d70ccf423be8fe1267e72
contracts/balincer-marginpooladdressprovider.sol	a7073e548f30ebe1e43406257bce256a18197ab4

Metrics

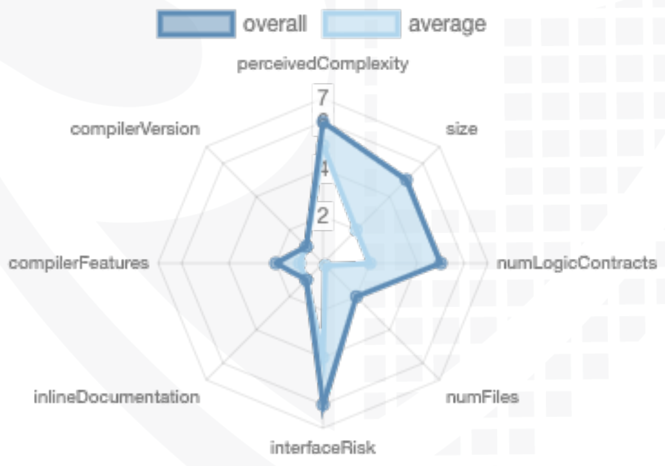
Source Lines

v1.0



Risk Level

v1.0



Capabilities

Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	28	104	71	11

Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

Version	Public	Payable
1.0	776	20

Version	External	Internal	Private	Pure	View
1.0	691	1161	20	361	474

State Variables

Version	Total	Public
1.0	1098	808

Capabilities

Version	Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	`^0.6.12`		`yes`	`yes` (22 asm blocks)	

Version	Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	ECRecover	New/ Create/ Create2
1.0	`yes`		`yes`	`yes`	`yes`	`yes` `→ NewContract:InitializableImmutableAdminUpgradeabilityProxy`

Scope of Work

The above token Team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract (usual the same name as team appended with .sol).

We will verify the following claims:

1. Correct implementation of Token standard
2. Deployer cannot mint any new tokens
3. Deployer cannot burn or lock user funds
4. Deployer cannot pause the contract
5. Overall checkup (Smart Contract Security)

Inheritance Graph

v1.0

Verify Claims

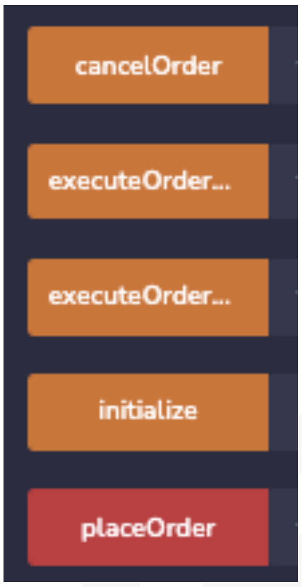
Correct implementation of Token standard

Tested	Verified
✓	✓

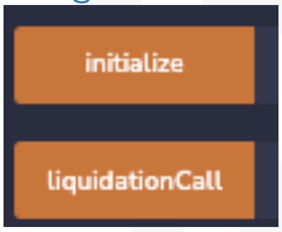
Function	Description	Exist	Tested	Verified
TotalSupply	provides information about the total token supply	✓	✓	✓
BalanceOf	provides account balance of the owner's account	✓	✓	✓
Transfer	executes transfers of a specified number of tokens to a specified address	✓	✓	✓
TransferFrom	executes transfers of a specified number of tokens from a specified address	✓	✓	✓
Approve	allow a spender to withdraw a set number of tokens from a specified account	✓	✓	✓
Allowance	returns a set number of tokens from a spender to the owner	✓	✓	✓

Write functions of contract

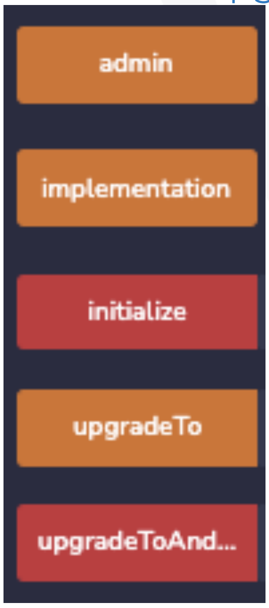
Orderbook



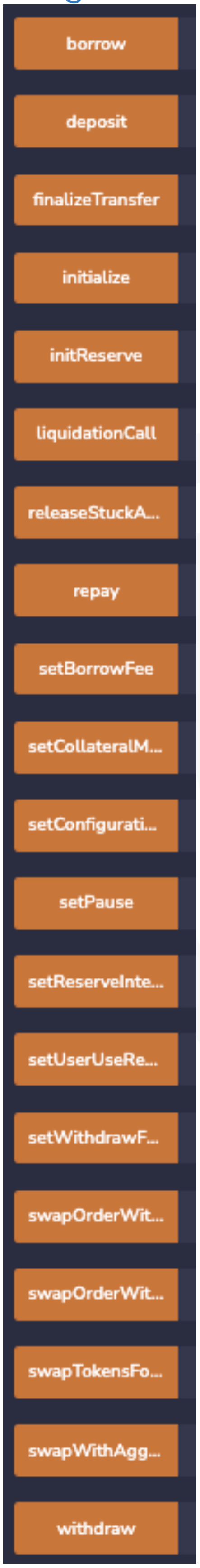
MarginPoolCollateralManager



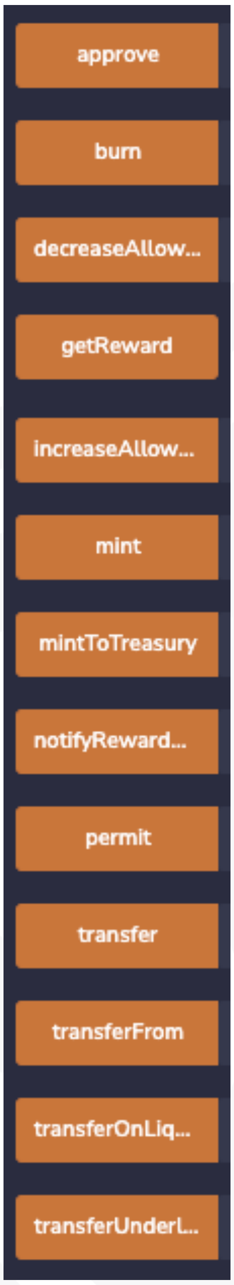
InitializableUpgradeabilityProxy



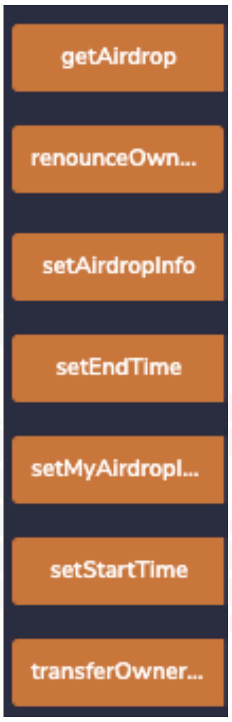
MarginPool



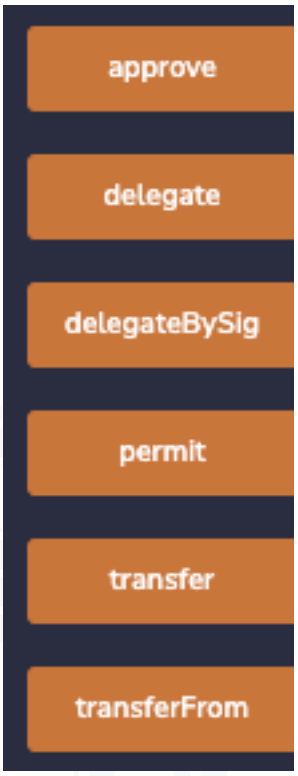
XToken



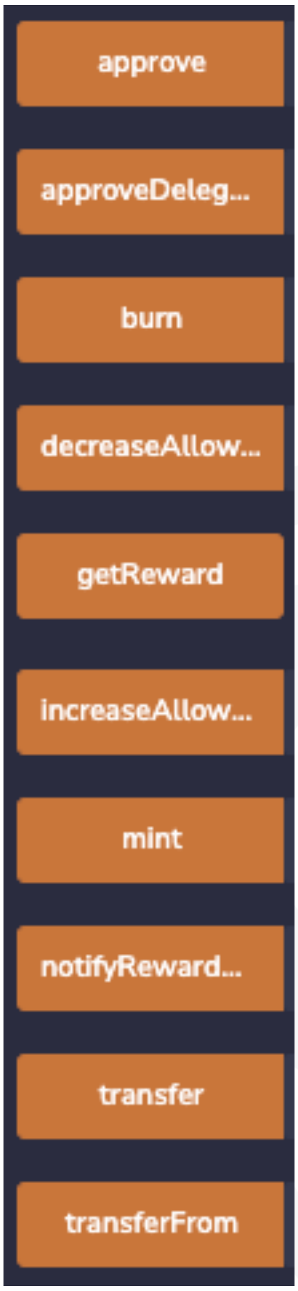
TwitterPool



BalincerToken



VariableDebtToken



approve

approveDeleg...

burn

decreaseAllow...

getReward

increaseAllow...

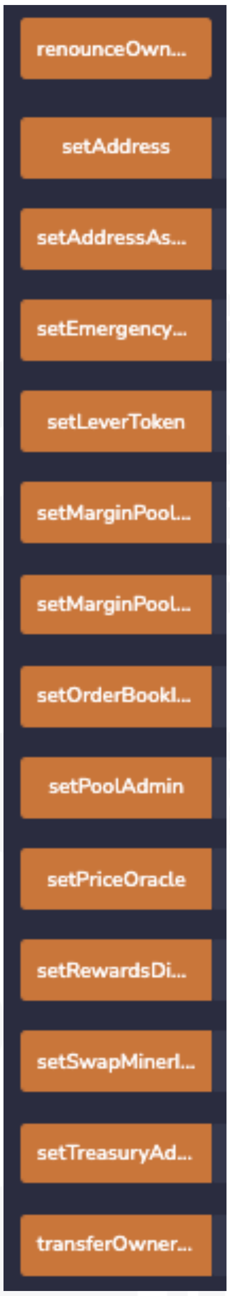
mint

notifyReward...

transfer

transferFrom

MarginPoolAddressProvider



renounceOwn...

setAddress

setAddressAs...

setEmergency...

setLeverToken

setMarginPool...

setMarginPool...

setOrderBookl...

setPoolAdmin

setPriceOracle

setRewardsDi...

setSwapMinerl...

setTreasuryAd...

transferOwner...

Deployer cannot mint any new tokens

File	Name	Exist	Tested	Verified
Orderbook	cannot mint	–	–	–
MarginPoolCollateralManager	cannot mint	–	–	–
InitializableUpgradeabilityProxy	cannot mint	–	–	–
MarginPool	cannot mint	–	–	–
XToken	cannot mint	✓	✓	✗
TwitterPool	cannot mint	–	–	–
BalincerToken	cannot mint	–	–	–
VariableDebtToken	cannot mint	✓	✓	✗
MarginPoolAddressProvider	cannot mint	–	–	–

Max / Total Supply:

Comments:

v1.0

- VariableDebtToken
    - OnlyMarginPool can mint
- XToken
    - OnlyMarginPool can mint
    - Can mint to treasury

Deployer cannot burn or lock user funds

File	Name	Exist	Tested	Verified
Orderbook	cannot lock	–	–	–
	cannot burn	–	–	–
MarginPoolCollateralManager	cannot lock	–	–	–
	cannot burn	–	–	–
InitializableUpgradeabilityProxy	cannot lock	–	–	–
	cannot burn	–	–	–
MarginPool	cannot lock	–	–	–
	cannot burn	–	–	–
XToken	cannot lock	✓	✓	✓
	cannot burn	✓	✓	✗
TwitterPool	cannot lock	–	–	–
	cannot burn	–	–	–
BalincerToken	cannot lock	✓	✓	✓
	cannot burn	–	–	–
VariableDebtToken	cannot lock	✓	✓	✓
	cannot burn	✓	✓	✗
MarginPoolAd	cannot lock	–	–	–

dressProvider	cannot burn	–	–	–

Comments:

v1.0

- VariableDebtToken
  - Being non transferrable, the debt token does not implement any of the standard ERC20 functions for transfer and allowance.
- XToken
  - onlyMarginpool can burn token

Deployer cannot pause the contract

File	Name	Exist	Tested	Verified
Orderbook	cannot pause	–	–	–
MarginPoolCollateralManager	cannot pause	–	–	–
InitializableUpgradeabilityProxy	cannot pause	–	–	–
MarginPool	cannot pause	✓	✓	✗
XToken	cannot pause	–	–	–
TwitterPool	cannot pause	✓	✓	✗
BalincerToken	cannot pause	–	–	–
VariableDebtToken	cannot pause	–	–	–
MarginPoolAddressProvider	cannot pause	–	–	–

Comments:

v1.0

- MarginPool
  - onlyMarginPoolConfigurator can set pause
    - In MarginPoolConfigurator onlyEmergencyAdmin can set pause of pool with setPoolPause
- TwitterPool
  - onlyOwner can set start time without any limitations and can lock some functions
    - getAirdrop
    - setMyAirdropInfo
    - setAirdropInfo

Overall checkup (Smart Contract Security)

Tested	Verified
✓	✓

Legend

Attribute	Symbol
Verfified / Checked	✓
Partly Verified	⚑
Unverified / Not checked	✗
Not available	–

Modifiers

Orderbook

- Initializer
    - Initialize

MarginPoolCollateralManager

- Initializer
    - Initialize

InitializableUpgradeabilityProxy

- Initializer
    - Initialize

MarginPool

- Initializer
    - Initialize
- onlyMarginConfigurator
    - setCollateralManager
    - setBorrowFee
    - setWithdrawFee
    - initReserve
    - setReserveInterestRateStrategyAddress
    - setConfiguration
    - setPause

- whenNotPaused
    - Deposit
    - Withdraw
    - Borrow
    - liquidationCall
    - swapTokensForTokens
    - swapOrderWithUni
    - swapOrderWithAggregation
    - repay
    - setUserUseReserveAsCollateral
    - finalizeTransfer
    - reDeposit

- onlyOrderBook
    - swapOrderWithUni
    - swapOrderWithAggregation

XToken

- onlyMarginPool

    - Burn
    - Mint
    - mintToTreasury
    - transferOnLiquidation
    - transferUnderlyingTo
- onlyRewardDistributor
    - notifyRewardAmount

- UpdateReward
    - Burn
    - Mint
    - mintToTreasury
    - transferOnLiquidation
    - getReward
    - notifyRewardAmount

TwitterPool
- onlyOwner
    - setStartTime
    - setEndTime
    - setAirdropInfo

- pauseGateway
    - getAirdrop
    - setMyAirdropInfo
    - setAirdropInfo

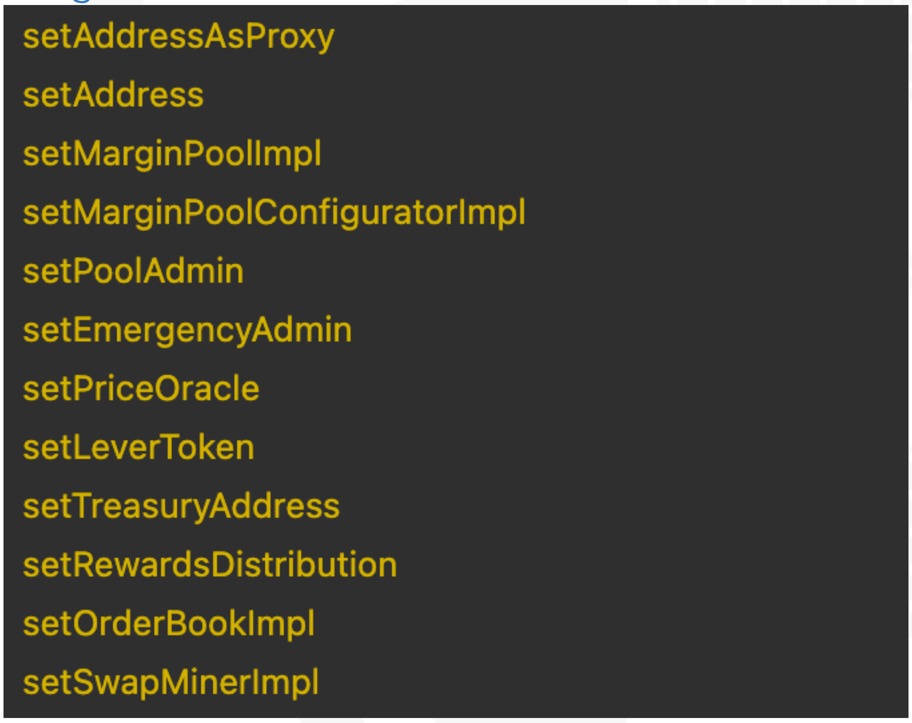
- validNewMember
    - getAirdrop

BalincerToken
- -

VariableDebtToken

- onlyMarginPool
    - Mint
    - Burn
- updateReward
    - mint
    - Burn
    - getReward
    - notifyRewardAmount
- onlyRewardsDistribution
    - notifyRewardAmount

MarginPoolAddressProvider

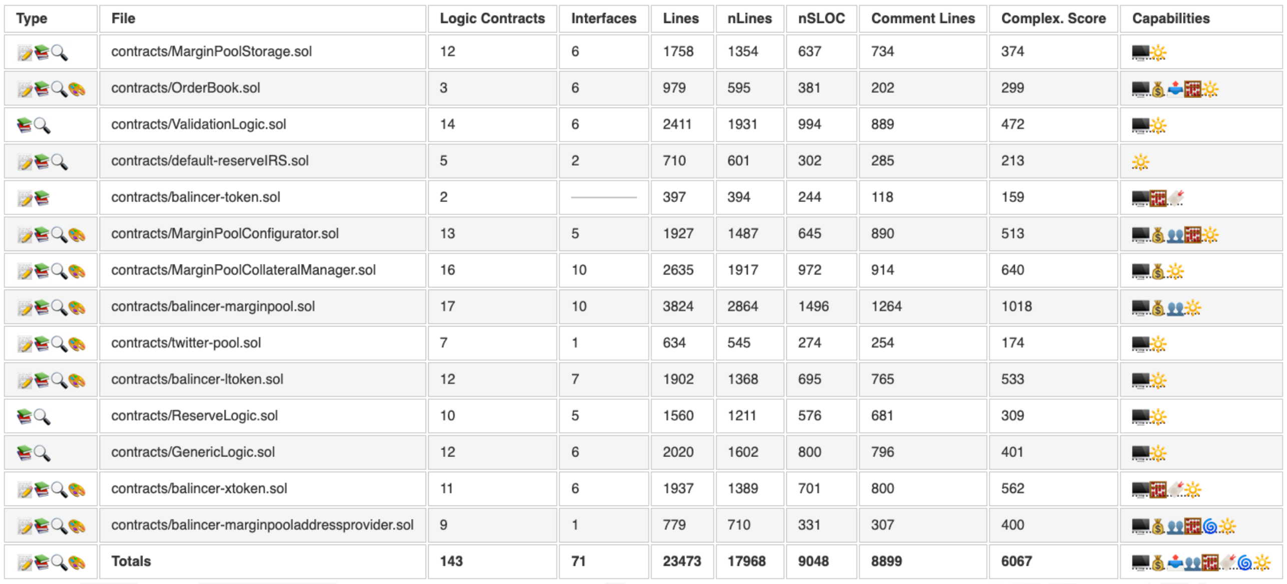


```
setAddressAsProxy
setAddress
setMarginPoolImpl
setMarginPoolConfiguratorImpl
setPoolAdmin
setEmergencyAdmin
setPriceOracle
setLeverToken
setTreasuryAddress
setRewardsDistribution
setOrderBookImpl
setSwapMinerImpl
```

CallGraph

Source Units in Scope

v1.0



Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📝📚🔍	contracts/MarginPoolStorage.sol	12	6	1758	1354	637	734	374	🖥🔆
📝📚🔍🎨	contracts/OrderBook.sol	3	6	979	595	381	202	299	🖥💰📤🧮🔆
📚🔍	contracts/ValidationLogic.sol	14	6	2411	1931	994	889	472	🖥🔆
📝📚🔍	contracts/default-reserveIRS.sol	5	2	710	601	302	285	213	🔆
📝📚	contracts/balincer-token.sol	2	—	397	394	244	118	159	🖥🧮🔖
📝📚🔍🎨	contracts/MarginPoolConfigurator.sol	13	5	1927	1487	645	890	513	🖥💰👥🧮🔆
📝📚🔍🎨	contracts/MarginPoolCollateralManager.sol	16	10	2635	1917	972	914	640	🖥💰🔆
📝📚🔍🎨	contracts/balincer-marginpool.sol	17	10	3824	2864	1496	1264	1018	🖥💰👥🔆
📝📚🔍🎨	contracts/twitter-pool.sol	7	1	634	545	274	254	174	🖥🔆
📝📚🔍🎨	contracts/balincer-ltoken.sol	12	7	1902	1368	695	765	533	🖥🔆
📚🔍	contracts/ReserveLogic.sol	10	5	1560	1211	576	681	309	🖥🔆
📚🔍	contracts/GenericLogic.sol	12	6	2020	1602	800	796	401	🖥🔆
📝📚🔍🎨	contracts/balincer-xtoken.sol	11	6	1937	1389	701	800	562	🖥🧮🔖🔆
📝📚🔍🎨	contracts/balincer-marginpooladdressprovider.sol	9	1	779	710	331	307	400	🖥💰👥🧮🌀🔆
📝📚🔍🎨	**Totals**	**143**	**71**	**23473**	**17968**	**9048**	**8899**	**6067**	🖥💰📤👥🧮🔖🌀🔆

Legend

Attribute	Description
Lines	total lines of the source unit
nLines	normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
nSLOC	normalized source lines of code (only source-code lines; no comments, no blank lines)
Comment Lines	lines containing single or block comments
Complexity Score	a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

Audit Results

AUDIT PASSED

Critical issues

- no critical issues found -

High issues

- no high issues found -

Medium issues

- no medium issues found -

Low issues

Issue	File	Type	Line	Description
#1	All files	Contract doesn't import npm packages from source (like OpenZeppelin etc.)	-	We recommend to import all packages from npm directly without flatten the contract. Functions could be modified or can be susceptible to vulnerabilities
#2	All files	A floating pragma is set	1	The current pragma Solidity directive is „"^0.6.12"".
#3	Margin PoolCo nfigurat or	Missing Zero Address Validation (missing-zero-check)	75, 110, 135	Check that the address is not zero
#4	OrderB ook	Missing Zero Address Validation (missing-zero-check)	99	Check that the address is not zero
#5	Balincer -ltoken	Missing Zero Address Validation (missing-zero-check)	208	Check that the address is not zero
#6	Balincer -xtoken	Missing Zero Address Validation (missing-zero-check)	85	Check that the address is not zero

#7	Balincer-margin pool	Missing Zero Address Validation (missing-zero-check)	125, 292,	Check that the address is not zero
#8	Balancer-margin pool	Missing Events Arithmetic	142, 147	Emit an event for critical parameter changes
#9	balincer-margin pool	State variable visibility is not set	1802, 1803	It is best practice to set the visibility of state variables explicitly
#10	balincer-margin pooladdressprovider.	State variable visibility is not set	105	It is best practice to set the visibility of state variables explicitly
#11	Twitter-pool	State variable visibility is not set	14, 84	It is best practice to set the visibility of state variables explicitly

Informational issues

Issue	File	Type	Line	Description
#1	balincer-token	State variables that could be declared constant (constable-states)	116	Add the `constant` attributes to state variables that never change
#2	balincer-xtoken	State variables that could be declared constant (constable-states)	55	Add the `constant` attributes to state variables that never change
#3	balincer-token	SPDX license identifier not provided	-	Use "SPDX-License-Identifier: UNLICENSED" for non-open-source code
#4	All files	Multiple SPDX license identifiers found	-	Use "AND" or "OR" to combine multiple licenses
#5	balincer-xtoken	Experimental ABIEncoderV2 is missing to use DataTypes	-	Add pragma experimental ABIEncoderV2 into source file

#6	balincer-ltoken	Error message missing in require statement	416	Add an error message in require statement
#7	balincer-xtoken	Error message missing in require statement	421	Add an error message in require statement
#8	Helpers	Functions that are not used	25	Remove unused functions
#9	IncentivizedERC20	Functions that are not used	222, 215, 218,	Remove unused functions
#10	Math	Functions that are not used	26, 11	Remove unused functions
#11	OrderBook	Functions that are not used	186, 233	Remove unused functions
#12	ReserveConfiguration	Functions that are not used	196, 315, 173, 102, 52, 290,	Remove unused functions
#13	ReserveLogic	Functions that are not used	131	Remove unused functions
#14	SafeMath	Functions that are not used	67, 116, 76, 140, 155, 96, 80, 85, 19	Remove unused functions
#15	ValidationLogic	Functions that are not used	288	Remove unused functions
#16	Balincer-ltoken	Functions that are not used	376	Remove unused functions
#17	WadRayMath	Functions that are not used	38, 45, 116, 31, 55	Remove unused functions
#18	balincer-ltoken	Redundant statements	157, 117, 135, 146, 128, 116, 145, 144, 129, 134, 156, 167, 168	Remove redundant statements if they congest code but offer no value
#19	ReserveLogic	Redundant statements	120, 121	Remove redundant statements if they congest code but offer no value
#20	Balincer-margin pool	Unused state variables	116, 47	Remove unused state variables

	balincer - margin pooladdressprovider	Unused state variables	146	Remove unused state variables

Commented Code exist

There are some instances of code being commented out in the following files that should be removed:

File	Line	Comment

Balincer-token	33, 35	// assert(b > 0); // Solidity automatically throws when dividing by 0
		
		// assert(a == b * c + a % b); // There is no case in which this doesn't hold

Recommendation

Remove the commented code, or address them properly.

Audit Comments

26. November 2021:

- For more information read report

SWC Attacks

ID	Title	Relationships	Status
SWC-136	Unencrypted Private Data On-Chain	CWE-767: Access to Critical Private Variable via Public Method	**PASSED**
SWC-135	Code With No Effects	CWE-1164: Irrelevant Code	**PASSED**
SWC-134	Message call with hardcoded gas amount	CWE-655: Improper Initialization	**PASSED**
SWC-133	Hash Collisions With Multiple Variable Length Arguments	CWE-294: Authentication Bypass by Capture-replay	**PASSED**
SWC-132	Unexpected Ether balance	CWE-667: Improper Locking	**PASSED**
SWC-131	Presence of unused variables	CWE-1164: Irrelevant Code	**PASSED**
SWC-130	Right-To-Left-Override control character (U+202E)	CWE-451: User Interface (UI) Misrepresentation of Critical Information	**PASSED**
SWC-129	Typographical Error	CWE-480: Use of Incorrect Operator	**PASSED**
SWC-128	DoS With Block Gas Limit	CWE-400: Uncontrolled Resource Consumption	**PASSED**

SWC-127	Arbitrary Jump with Function Type Variable	CWE-695: Use of Low-Level Functionality	**PASSED**
SWC-125	Incorrect Inheritance Order	CWE-696: Incorrect Behavior Order	**PASSED**
SWC-124	Write to Arbitrary Storage Location	CWE-123: Write-what-where Condition	**PASSED**
SWC-123	Requirement Violation	CWE-573: Improper Following of Specification by Caller	**PASSED**
SWC-122	Lack of Proper Signature Verification	CWE-345: Insufficient Verification of Data Authenticity	**PASSED**
SWC-121	Missing Protection against Signature Replay Attacks	CWE-347: Improper Verification of Cryptographic Signature	**PASSED**
SWC-120	Weak Sources of Randomness from Chain Attributes	CWE-330: Use of Insufficiently Random Values	**PASSED**
SWC-119	Shadowing State Variables	CWE-710: Improper Adherence to Coding Standards	**PASSED**
SWC-118	Incorrect Constructor Name	CWE-665: Improper Initialization	**PASSED**
SWC-117	Signature Malleability	CWE-347: Improper Verification of Cryptographic Signature	**PASSED**

SWC-116	Timestamp Dependence	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	PASSED
SWC-115	Authorization through tx.origin	CWE-477: Use of Obsolete Function	PASSED
SWC-114	Transaction Order Dependence	CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	PASSED
SWC-113	DoS with Failed Call	CWE-703: Improper Check or Handling of Exceptional Conditions	PASSED
SWC-112	Delegatecall to Untrusted Callee	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	PASSED
SWC-111	Use of Deprecated Solidity Functions	CWE-477: Use of Obsolete Function	PASSED
SWC-110	Assert Violation	CWE-670: Always-Incorrect Control Flow Implementation	PASSED
SWC-109	Uninitialized Storage Pointer	CWE-824: Access of Uninitialized Pointer	PASSED
SWC-108	State Variable Default Visibility	CWE-710: Improper Adherence to Coding Standards	NOT PASSED
SWC-107	Reentrancy	CWE-841: Improper Enforcement of Behavioral Workflow	PASSED
SWC-106	Unprotected SELFDESTRUCT Instruction	CWE-284: Improper Access Control	PASSED

SWC-105	Unprotected Ether Withdrawal	CWE-284: Improper Access Control	PASSED
SWC-104	Unchecked Call Return Value	CWE-252: Unchecked Return Value	PASSED
SWC-103	Floating Pragma	CWE-664: Improper Control of a Resource Through its Lifetime	NOT PASSED
SWC-102	Outdated Compiler Version	CWE-937: Using Components with Known Vulnerabilities	PASSED
SWC-101	Integer Overflow and Underflow	CWE-682: Incorrect Calculation	PASSED
SWC-100	Function Default Visibility	CWE-710: Improper Adherence to Coding Standards	PASSED

Solid Proofed

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY