Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project: Scrub Finance Protocol
Platform: Cronos Blockchain
Language: Solidity
Date: March 28th, 2022

Table of contents

Email: audit@EtherAuthority.io

THIS IS SECURITY AUDIT REPORT DOCUMENT AND WHICH MAY CONTAIN INFORMATION WHICH IS CONFIDENTIAL. WHICH INCLUDES ANY POTENTIAL VULNERABILITIES AND MALICIOUS CODES WHICH CAN BE USED TO EXPLOIT THE SOFTWARE. THIS MUST BE REFERRED INTERNALLY AND ONLY SHOULD BE MADE AVAILABLE TO THE PUBLIC AFTER ISSUES ARE RESOLVED.

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by the Scrub Finance team to perform the Security audit of the Scrub Finance Protocol smart contracts code. The audit has been performed using manual analysis as well as using automated software tools. This report presents all the findings regarding the audit performed on March 28th, 2022.

The purpose of this audit was to address the following:

- Ensure that all claimed functions exist and function correctly.
- Identify any security vulnerabilities that may be present in the smart contract.

Project Background

The Scrub Finance Contracts have functions like stake, withdraw, epoch, claimReward, distributeReward, burn, add new pool, etc. The Scrub Finance contracts also inherits ERC20Burnable, Math, IERC20, SafeERC20, ReentrancyGuard, SafeMath standard smart contracts from the openzepelin library.

Audit scope

Name	Code Review and Security Analysis Report for Scrub Finance Protocol Smart Contracts
Platform	**Cronos / Solidity**
File 1	Scrub.sol
File 1 MD5 Hash	9564259D43E0E0F9C224EAB74D8442C5
File 2	LBond.sol
File 2 MD5 Hash	4E3F2A179BB7DC86F6D39A038E4C8C2C
File 3	Lion.sol
File 3 MD5 Hash	D64082C2269102AF48CB48063FA8E76A
File 4	Oracle.sol
File 4 MD5 Hash	EAF11EC3474020A77AB343D97A681059
File 5	Tiger.sol
File 5 MD5 Hash	757F6EEC03D7B27976DE46FF41439FD0
File 6	Treasury.sol
File 6 MD5 Hash	FAA81C6BABFCA92A368EFD808B1008F7
File 7	TigerRewardPool.sol
File 7 MD5 Hash	178D735ECCA38C1179A9032C372C29C8
Audit Date	March 28th,2022
Revise Audit Date	April 4th,2022

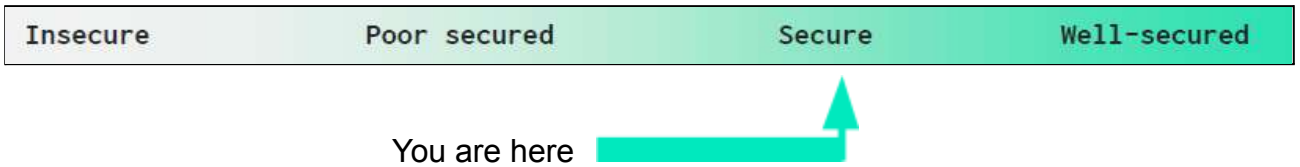
Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
File 1 Scrub.sol • Withdraw Lockup Epochs: 6 epochs • Reward Lockup Epochs: 3 epochs	**YES, This is valid.**
File 2 Oracle.sol • Oracle can update 1-day EMA price from Uniswap.	**YES, This is valid.**
File 3 LBond.sol • Name: Lion Bonds • Symbol: LBOND • Decimals: 18	**YES, This is valid.**
File 4 Lion.sol • Name: LION • Symbol: LION • Decimals: 18 • Burn Threshold: 1.1 LION • Initial Launchpad Distribution: 0.4 million LION • Total Supply: 400001 LION	**YES, This is valid.**
File 5 Treasury.sol • Period: 8 hours • Bond supply for depletion floor: 100% • Seigniorage Expansion Floor Percent: 35% • Maximum Supply Contraction Percent: 3% • Maximum Debt Ratio Percent: 35% • Premium Threshold: 1.1 • Premium Percent: 70% • Maximum Supply Expansion Percent: 4%	**YES, This is valid.** **Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**

File 6 Tiger.sol • Name: Lion Shares • Symbol: TIGER • Tax Rate: 1% • Farming Pool Reward Allocation: 35,000 TIGER • Community Fund Pool Allocation: 5000 TIGER • Dev Fund Pool Allocation: 5000 TIGER • Digits Dao Allocation: 5000 TIGER • Initial Pool Supply: 100 TIGER • Maximum Tax Rate: 1%	**YES, This is valid.**
File 7 TigerRewardPool.sol • Total Rewards: 35,000 TIGER • Running Time: 365 days	**YES, This is valid.**

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **"Secured"**. These contracts do contain owner control, which does not make them fully decentralized.



We used various tools like Slither, Solhint and Remix IDE. At the same time this finding is based on critical analysis of the manual audit.
All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the Audit overview section. General overview is presented in AS-IS section and all identified issues can be found in the Audit overview section.

We found 0 critical, 0 high, 0 medium and 1 low and some very low level issues.
All these issues have been resolved / acknowledged.

Investors Advice: Technical audit of the smart contract does not guarantee the ethical nature of the project. Any owner controlled functions should be executed by the owner with responsibility. All investors/users are advised to do their due diligence before investing in the project.

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Passed
	Integer overflow/underflow	Passed
	Function input parameters lack of check	Passed
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Passed
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	N/A
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Passed
	Other programming issues	Passed
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Passed
Gas Optimization	“Out of Gas” Issue	Passed
	High consumption ‘for/while’ loop	Passed
	High consumption ‘storage’ storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage not set	Passed
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: **PASSED**

Code Quality

This audit scope has 7 smart contract files. Smart contracts contain Libraries, Smart contracts, inherits and Interfaces. This is a compact and well written smart contract.

The libraries in the Scrub Finance Protocol are part of its logical algorithm. A library is a different type of smart contract that contains reusable code. Once deployed on the blockchain (only once), it is assigned a specific address and its properties / methods can be reused many times by other contracts in the Scrub Finance Protocol.

The Scrub Finance Protocol team has not provided unit test scripts, which would have helped to determine the integrity of the code in an automated way.

Code parts are **not** well commented on smart contracts.

Documentation

We were given a Scrub Finance Protocol smart contract code in the form of a File. The hash of that code is mentioned above in the table.

As mentioned above, code parts are **not well** commented. So it is not easy to quickly understand the programming flow as well as complex code logic. Comments are very helpful in understanding the overall architecture of the protocol.

Use of Dependencies

As per our observation, the libraries are used in this smart contracts infrastructure that are based on well known industry standard open source projects.

Apart from libraries, its functions are used in external smart contract calls.

AS-IS overview

Scrub.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	onlyOperator	modifier	Passed	No Issue
3	masonExists	modifier	Passed	No Issue
4	updateReward	modifier	Passed	No Issue
5	notInitialized	modifier	Passed	No Issue
6	initialize	write	Passed	No Issue
7	setOperator	external	access only Operator	No Issue
8	setLockUp	external	access only Operator	No Issue
9	latestSnapshotIndex	read	Passed	No Issue
10	getLatestSnapshot	internal	Passed	No Issue
11	getLastSnapshotIndexOf	read	Passed	No Issue
12	getLastSnapshotOf	internal	Passed	No Issue
13	canWithdraw	external	Passed	No Issue
14	canClaimReward	external	Passed	No Issue
15	epoch	external	Passed	No Issue
16	nextEpochPoint	external	Passed	No Issue
17	getLionPrice	external	Passed	No Issue
18	rewardPerShare	read	Passed	No Issue
19	earned	read	Passed	No Issue
20	stake	write	access only One Block	No Issue
21	withdraw	write	access only One Block	No Issue
22	exit	external	Passed	No Issue
23	claimReward	write	Passed	No Issue
24	allocateSeigniorage	external	access only Operator	No Issue
25	governanceRecoverUnsupported	external	access only Operator	No Issue
26	totalSupply	read	Passed	No Issue
27	balanceOf	read	Passed	No Issue
28	stake	write	Passed	No Issue
29	withdraw	write	Passed	No Issue
30	checkSameOriginReentranted	internal	Passed	No Issue
31	checkSameSenderReentranted	internal	Passed	No Issue
32	onlyOneBlock	modifier	Passed	No Issue

LBond.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	mint	write	access only Operator	No Issue
3	burn	write	Passed	No Issue
4	burnFrom	write	Passed	No Issue
5	owner	read	Passed	No Issue
6	onlyOwner	modifier	Passed	No Issue
7	renounceOwnership	write	access only Owner	No Issue
8	transferOwnership	write	access only Owner	No Issue
9	_transferOwnership	internal	Passed	No Issue
10	operator	read	Passed	No Issue
11	onlyOperator	modifier	Passed	No Issue
12	isOperator	read	Passed	No Issue
13	transferOperator	write	access only Owner	No Issue
14	_transferOperator	internal	Passed	No Issue

Lion.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	burn	write	Passed	No Issue
3	burnFrom	write	Passed	No Issue
4	owner	read	Passed	No Issue
5	onlyOwner	modifier	Passed	No Issue
6	renounceOwnership	write	access only Owner	No Issue
7	transferOwnership	write	access only Owner	No Issue
8	_transferOwnership	internal	Passed	No Issue
9	operator	read	Passed	No Issue
10	onlyOperator	modifier	Passed	No Issue
11	isOperator	read	Passed	No Issue
12	transferOperator	write	access only Owner	No Issue
13	_transferOperator	internal	Passed	No Issue
14	onlyTaxOffice	modifier	Passed	No Issue
15	onlyOperatorOrTaxOffice	modifier	Passed	No Issue
16	getTaxTiersTwapsCount	read	Passed	No Issue
17	getTaxTiersRatesCount	read	Passed	No Issue
18	isAddressExcluded	read	Passed	No Issue
19	setTaxTiersTwap	write	access only Tax Office	No Issue
20	setTaxTiersRate	write	access only Tax Office	No Issue
21	setBurnThreshold	write	access only Tax Office	No Issue
22	_getLionPrice	internal	Passed	No Issue
23	_updateTaxRate	internal	Passed	No Issue
24	enableAutoCalculateTax	write	access only Tax Office	No Issue

25	disableAutoCalculateTax	write	access only Tax Office	No Issue
26	setOracle	write	access only Operator Or Tax Office	No Issue
27	setTaxOffice	write	access only Operator Or Tax Office	No Issue
28	setTaxCollectorAddress	write	access only Tax Office	No Issue
29	setTaxRate	write	access only Tax Office	No Issue
30	setBurnTax	write	access only Tax Office	No Issue
31	excludeAddress	write	access only Operator Or Tax Office	No Issue
32	includeAddress	write	access only Operator Or Tax Office	No Issue
33	mint	write	access only Operator	No Issue
34	burn	write	Passed	No Issue
35	burnFrom	write	access only Operator	No Issue
36	transferFrom	write	Passed	No Issue
37	_transferWithTax	internal	Passed	No Issue
38	distributeReward	external	access only Operator	No Issue
39	governanceRecoverUnsupported	external	access only Operator	No Issue

Oracle.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	operator	read	Passed	No Issue
8	onlyOperator	modifier	Passed	No Issue
9	isOperator	read	Passed	No Issue
10	transferOperator	write	access only Owner	No Issue
11	_transferOperator	internal	Passed	No Issue
12	checkStartTime	modifier	Passed	No Issue
13	checkEpoch	modifier	Passed	No Issue
14	getCurrentEpoch	read	Passed	No Issue
15	getPeriod	read	Passed	No Issue
16	getStartTime	read	Passed	No Issue
17	getLastEpochTime	read	Passed	No Issue
18	nextEpochPoint	read	Passed	No Issue
19	setPeriod	external	access only Operator	No Issue
20	setEpoch	external	access only Operator	No Issue
21	update	external	Passed	No Issue
22	consult	external	Passed	No Issue

23	twap	external	Passed	No Issue

Tiger.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	operator	read	Passed	No Issue
8	onlyOperator	modifier	Passed	No Issue
9	isOperator	read	Passed	No Issue
10	transferOperator	write	access only Owner	No Issue
11	_transferOperator	internal	Passed	No Issue
12	onlyTaxOffice	modifier	Passed	No Issue
13	onlyOperatorOrTaxOffice	modifier	Passed	No Issue
14	setTreasuryFund	external	access only Operator	No Issue
15	setDevFund	external	Passed	No Issue
16	unclaimedTreasuryFund	read	Passed	No Issue
17	unclaimedDevFund	read	Passed	No Issue
18	unclaimedDigitsDaoFund	read	Passed	No Issue
19	claimRewards	external	Passed	No Issue
20	transferFrom	write	Passed	No Issue
21	_transferWithTax	internal	Passed	No Issue
22	setTaxRate	write	access only Operator Or Tax Office	No Issue
23	excludeAddress	write	access only Operator Or Tax Office	No Issue
24	includeAddress	write	access only Operator Or Tax Office	No Issue
25	setTaxOffice	write	access only Operator Or Tax Office	No Issue
26	setTaxCollectorAddress	write	access only Tax Office	No Issue
27	distributeReward	write	access only Operator	No Issue
28	burn	write	Passed	No Issue
29	governanceRecoverUnsupported	external	access only Operator	No Issue

Treasury.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue

2	onlyOperator	modifier	Passed	No Issue
3	checkCondition	modifier	Passed	No Issue
4	checkEpoch	modifier	Passed	No Issue
5	checkOperator	modifier	Passed	No Issue
6	notInitialized	modifier	Passed	No Issue
7	isInitialized	read	Passed	No Issue
8	nextEpochPoint	read	Passed	No Issue
9	getLionPrice	read	Passed	No Issue
10	getLionUpdatedPrice	read	Passed	No Issue
11	getReserve	read	Passed	No Issue
12	getBurnableLionLeft	read	Passed	No Issue
13	getRedeemableBonds	read	Passed	No Issue
14	getBondDiscountRate	read	Passed	No Issue
15	getBondPremiumRate	read	Passed	No Issue
16	initialize	write	Passed	No Issue
17	setOperator	external	access only Operator	No Issue
18	setScrub	external	access only Operator	No Issue
19	setLionOracle	external	access only Operator	No Issue
20	setLionPriceCeiling	external	access only Operator	No Issue
21	setMaxSupplyExpansionPercents	external	access only Operator	No Issue
22	setSupplyTiersEntry	external	access only Operator	No Issue
23	setMaxExpansionTiersEntry	external	access only Operator	No Issue
24	setBondDepletionFloorPercent	external	access only Operator	No Issue
25	setMaxSupplyContractionPercent	external	access only Operator	No Issue
26	setMaxDebtRatioPercent	external	access only Operator	No Issue
27	setBootstrap	external	access only Operator	No Issue
28	setExtraFunds	external	access only Operator	No Issue
29	setMaxDiscountRate	external	access only Operator	No Issue
30	setMaxPremiumRate	external	access only Operator	No Issue
31	setDiscountPercent	external	access only Operator	No Issue
32	setPremiumThreshold	external	access only Operator	No Issue
33	setPremiumPercent	external	access only Operator	No Issue
34	setMintingFactorForPayingDebt	external	access only Operator	No Issue
35	_updateLionPrice	internal	Passed	No Issue
36	getLionCirculatingSupply	read	access only Operator	No Issue
37	buyBonds	external	access only One Block	No Issue
38	redeemBonds	external	access only One Block	No Issue
39	_sendToScrub	internal	Passed	No Issue
40	_calculateMaxSupplyExpansionPercent	internal	Passed	No Issue

41	allocateSeigniorage	external	access only One Block	No Issue
42	excludeFromTotalSupply	external	access only Operator	No Issue
43	includeToTotalSupply	external	access only Operator	No Issue
44	governanceRecoverUnsupported	external	access only Operator	No Issue
45	ScrubSetOperator	external	access only Operator	No Issue
46	ScrubSetLockUp	external	access only Operator	No Issue
47	ScrubAllocateSeigniorage	external	access only Operator	No Issue
48	ScrubGovernanceRecoverUnsupported	external	access only Operator	No Issue
49	checkSameOriginReentranted	internal	Passed	No Issue
50	checkSameSenderReentranted	internal	Passed	No Issue
51	onlyOneBlock	modifier	Passed	No Issue

TigerRewardPool.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	onlyOperator	modifier	Passed	No Issue
3	checkPoolDuplicate	internal	Passed	No Issue
4	add	write	access only Operator	No Issue
5	set	write	access only Operator	No Issue
6	getGeneratedReward	read	Passed	No Issue
7	pendingTIGER	external	Passed	No Issue
8	massUpdatePools	write	Passed	No Issue
9	updatePool	write	Passed	No Issue
10	deposit	write	Passed	No Issue
11	withdraw	write	Passed	No Issue
12	emergencyWithdraw	write	Passed	No Issue
13	safeTShareTransfer	internal	Passed	No Issue
14	setOperator	external	access only Operator	No Issue
15	governanceRecoverUnsupported	external	access only Operator	No Issue

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to token loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Email: audit@EtherAuthority.io

Audit Findings

Critical Severity

No Critical severity vulnerabilities were found.

High Severity

No High severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

(1) Critical operation lacks event log: **TigerRewardPool.sol**

Missing event log for:

- add
- set
- setOperator

Resolution: Write an event log for listed events.

Status: Fixed

Very Low / Informational / Best practices:

(1) Variables should be made immutable:

Variables that are defined within the constructor but further remain unchanged should be marked as immutable to save gas and to ease the reviewing process of third-parties.

Treasury.sol

lionPriceOne , startTime, lion , tiger , lbond

Tiger.sol

startTime, endTime, communityFundRewardRate , devFundRewardRate , digitsDaoRewardRate, digitsDaoFund.

Scrub.sol

lion, share, treasury

TigerRewardPool.sol

poolEndTime, poolStartTime, feeAddress

Resolution: We suggest setting these variables as immutable.
Status: Acknowledged

(2) Make variables constant: **PowderRewardPool.sol**

runningTime, tSharePerSecond

Resolution: We suggest setting these variables as constant.
Status: Acknowledged

Centralization

This smart contract has some functions which can be executed by the Admin (Owner) only. If the admin wallet private key would be compromised, then it would create trouble. Following are Admin functions:

- setOperator: The Scrub Operator can set the operator address.
- setLockUp: The Scrub Operator can set reward lockup epochs and withdraw lockup epochs.
- allocateSeigniorage: The Scrub Operator can allocate Seigniorage amount.
- governanceRecoverUnsupported: The Scrub Operator can transfer the amount to governance to recover unsupported addresses.
- mint: The LBond Operator can mints basis bonds to a recipient.
- burnFrom: The LBond Operator can burn an amount from an account.
- mint: The Lion Operator can mint LION to a recipient.
- burnFrom: The Lion Operator can burn an amount from an account.
- distributeReward: The Lion Operator can distribute to the launchpad.
- governanceRecoverUnsupported: The Lion Operator can transfer the amount to governance to recover unsupported addresses.
- distributeReward: The Tiger Operator can distribute to the reward pool.

- governanceRecoverUnsupported: The Tiger Operator can transfer the amount to governance to recover unsupported addresses.
- setOperator: The Treasury Operator can set the operator address.
- setScrub: The Treasury Operator can set a Scrub address.
- setLionOracle: The Treasury Operator can set a lion oracle address.
- setLionPriceCeiling: The Treasury Operator can set a lion price ceiling.
- setMaxSupplyExpansionPercents: The Treasury Operator can set maximum supply expansion percentages.
- setSupplyTiersEntry: The Treasury Operator can set supply tiers entry value and index.
- setMaxExpansionTiersEntry: The Treasury Operator can set Maximum expansion tiers entry.
- setBondDepletionFloorPercent: The Treasury Operator can set bond depletion floor percentage.
- setMaxSupplyContractionPercent: The Treasury Operator can set maximum supply contraction percentage.
- setMaxDebtRatioPercent: The Treasury Operator can set maximum debt ratio percentage.
- setBootstrap: The Treasury Operator can set bootstrap epoch.
- setExtraFunds: The Treasury Operator can set dao funds, dev funds.
- setMaxDiscountRate: The Treasury Operator can set maximum Discount Rate.
- setMaxPremiumRate: The Treasury Operator can set maximum Premium Rate.
- setDiscountPercent: The Treasury Operator can set a discount percentage.
- setPremiumThreshold: The Treasury Operator can be the premium threshold.
- setPremiumPercent: The Treasury Operator can be premium percentages.
- setMintingFactorForPayingDebt: The Treasury Operator can set the minting factor for paying debt value.
- buyBonds: The Treasury Operator can buy bonds.
- allocateSeigniorage: The Treasury Operator can allocate Seigniorage.
- excludeFromTotalSupply: The Treasury Operator can exclude from total supply.
- includeToTotalSupply: The Treasury Operator can include total supply.
- governanceRecoverUnsupported: The Treasury Operator can transfer the amount to governance to recover unsupported addresses.
- scrubSetOperator: The Treasury Operator can set Scrub operator address.

- scrubSetLockUp: The Treasury Operator can withdraw Lockup Epochs value, reward Lockup Epochs value.
- scrubAllocateSeigniorage: The Treasury Operator can set Scrub allocate seigniorage amount.
- scrubGovernanceRecoverUnsupported: The Treasury Operator can transfer the Scrub governance to recover unsupported addresses.
- governanceRecoverUnsupported: The TigerRewardPool Operator can transfer the amount to governance to recover unsupported addresses.
- set: The TigerRewardPool Operator can update the given pool's tSHARE allocation point.
- add: The TigerRewardPool Operator can add a new lp to the pool.

Conclusion

We were given a contract code in the form of files. And we have used all possible tests based on given objects as files. We have not observed any major issues in the smart contracts. So, **it's good to go to production**.

Since possible test cases can be unlimited for such smart contracts protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan everything.

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart Contract's high-level description of functionality was presented in the As-is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security state of the reviewed contract, based on standard audit procedure scope, is **"Secured"**.

Email: audit@EtherAuthority.io

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review:

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis:

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

Documenting Results:

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system.

Suggested Solutions:

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Disclaimers

EtherAuthority.io Disclaimer

EtherAuthority team has analyzed this smart contract in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report, (Source Code); the Source Code compilation, deployment and functionality (performing the intended functions).

Due to the fact that the total number of test cases are unlimited, the audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only. We also suggest conducting a bug bounty program to confirm the high level of security of this smart contract.

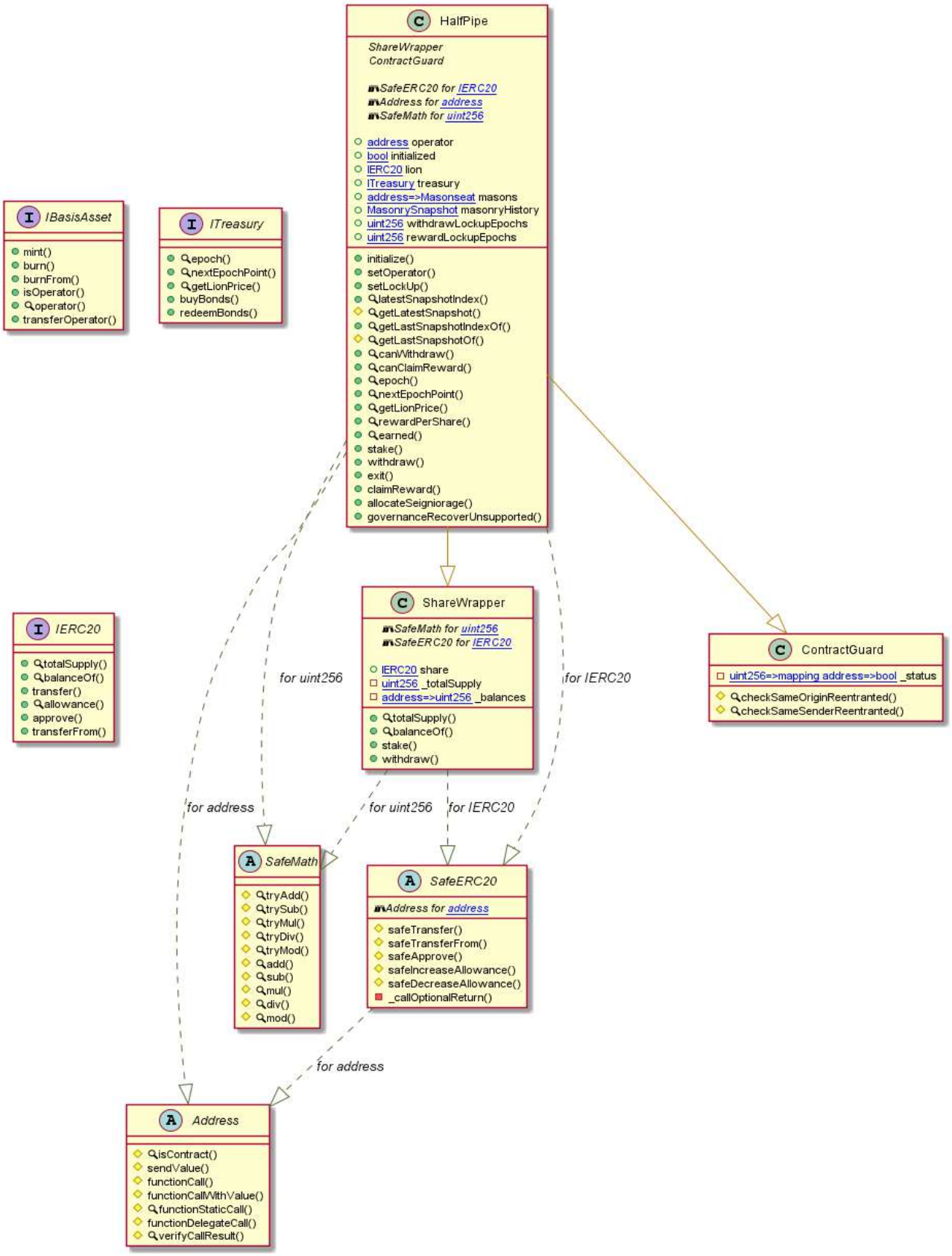
Technical Disclaimer

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can't guarantee explicit security of the audited smart contracts.

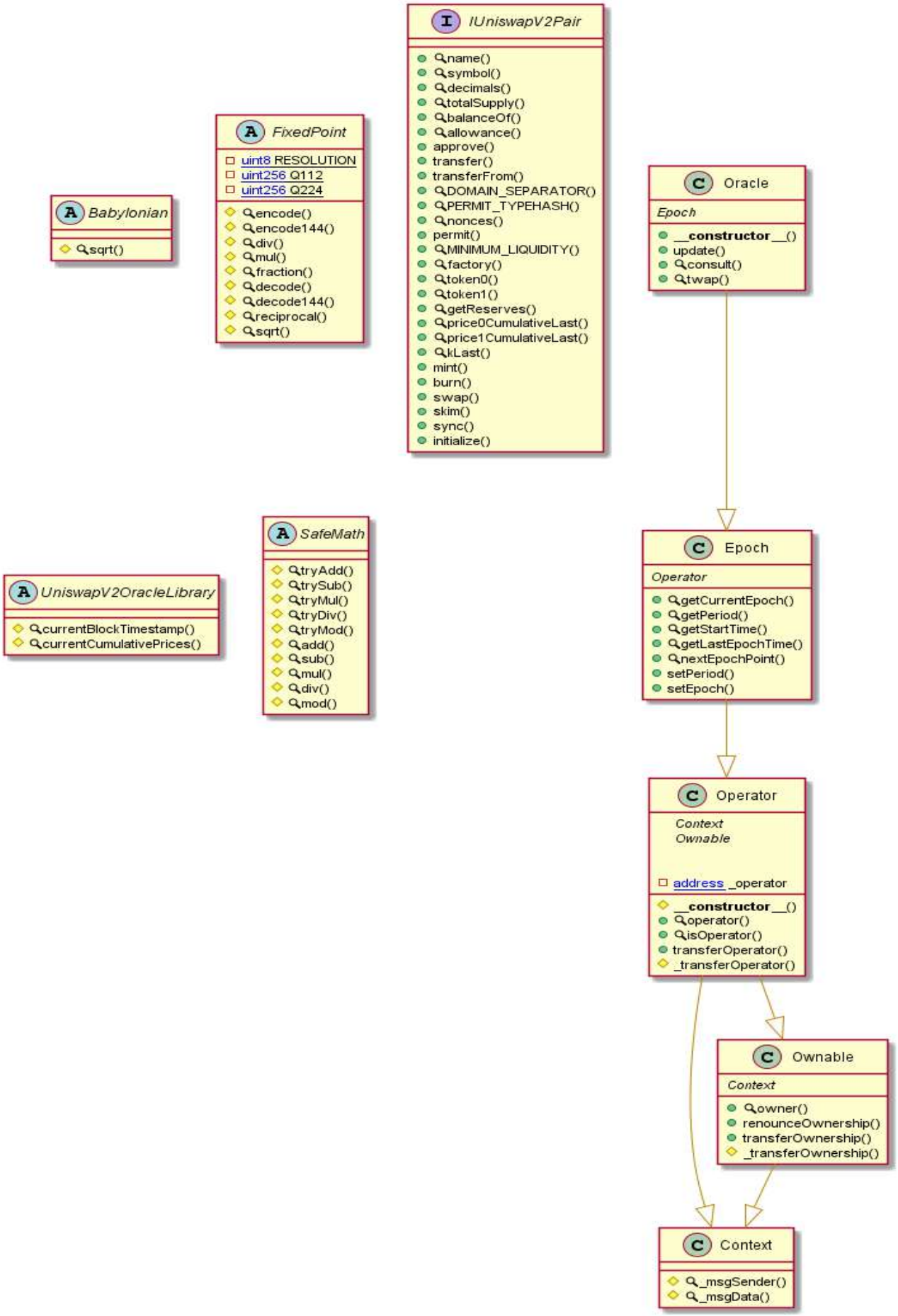
Appendix

Code Flow Diagram - Scrub Finance Protocol

Scrub Diagram

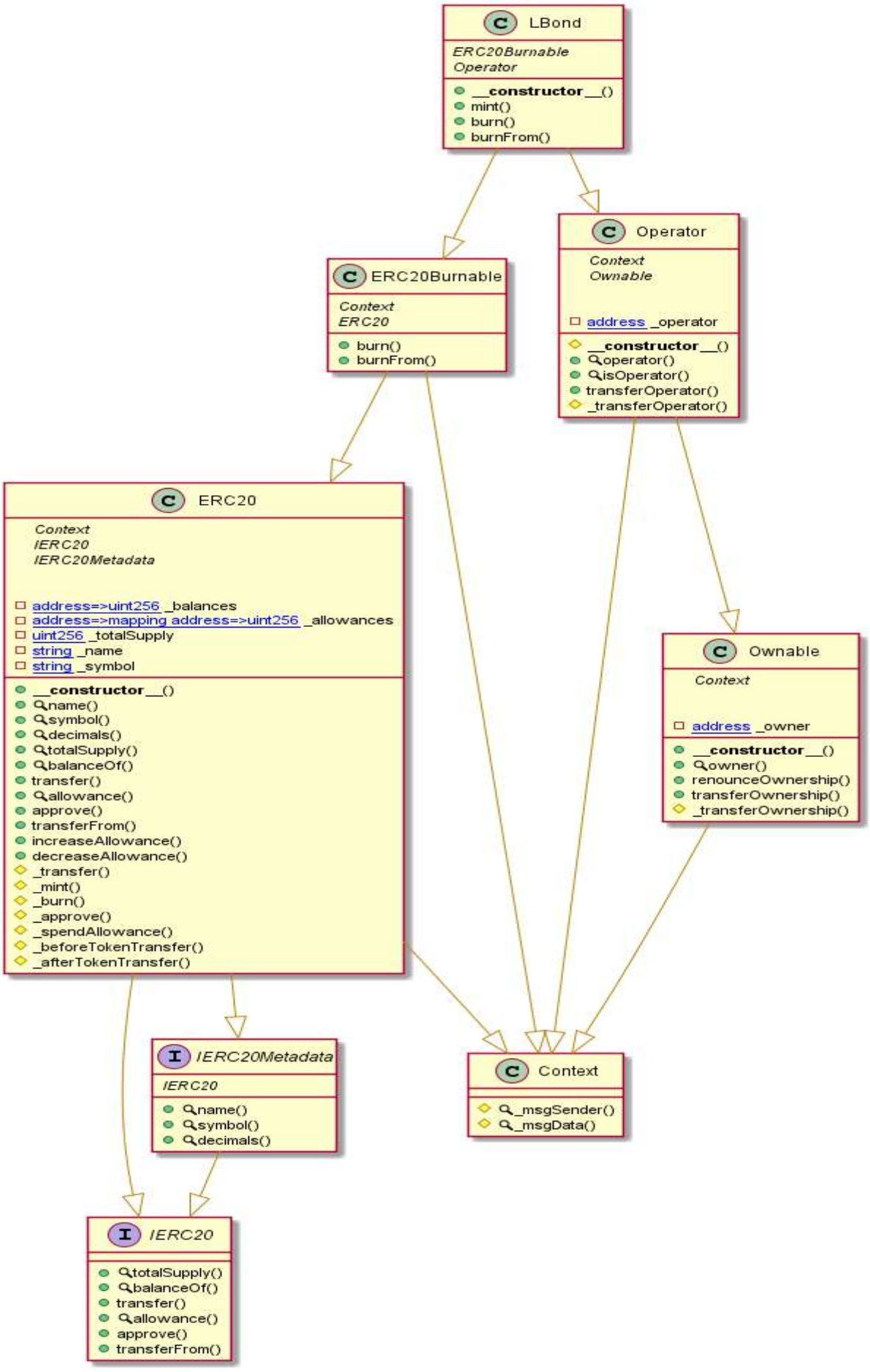


Oracle Diagram



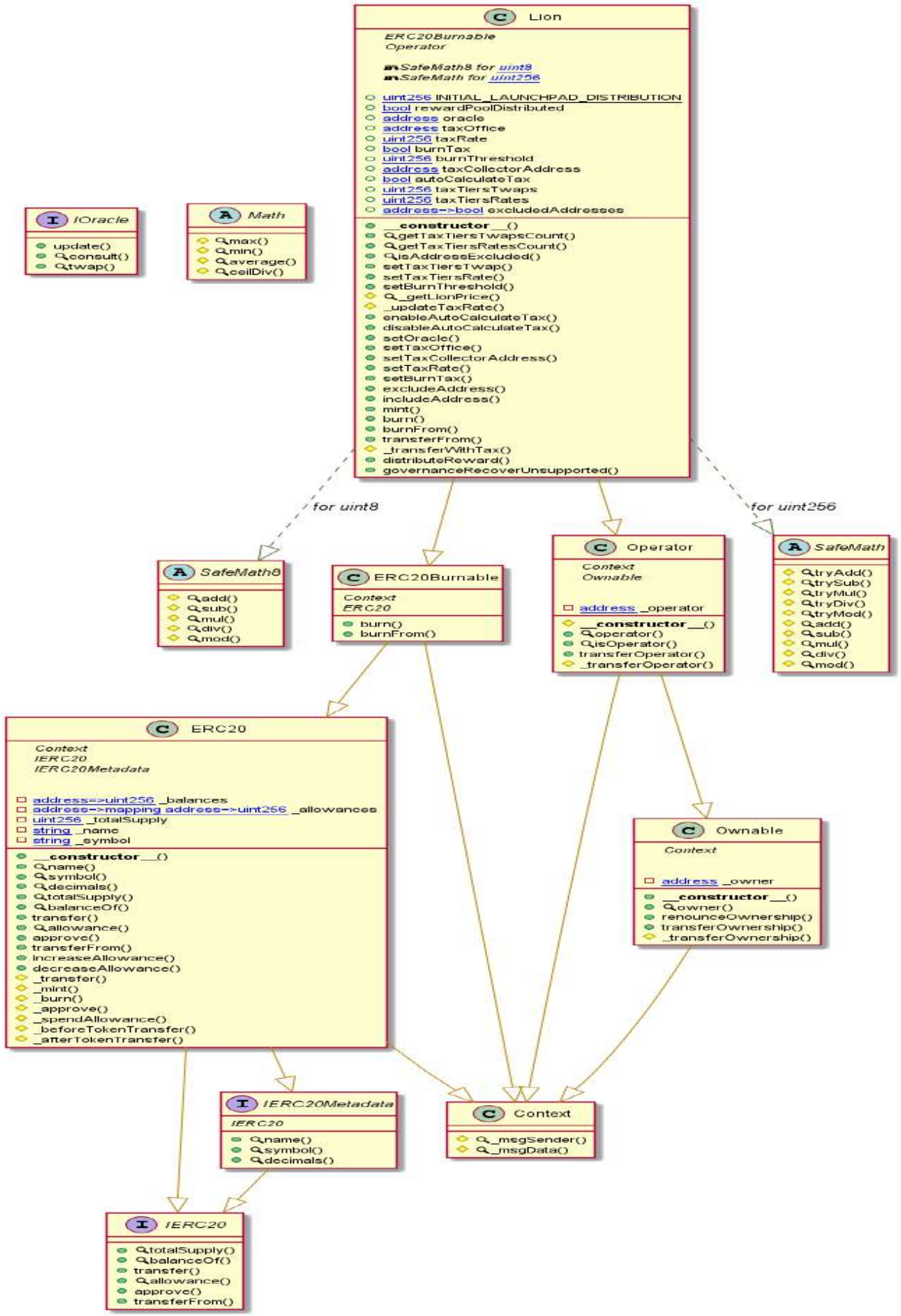
Email: audit@EtherAuthority.io

LBond Diagram

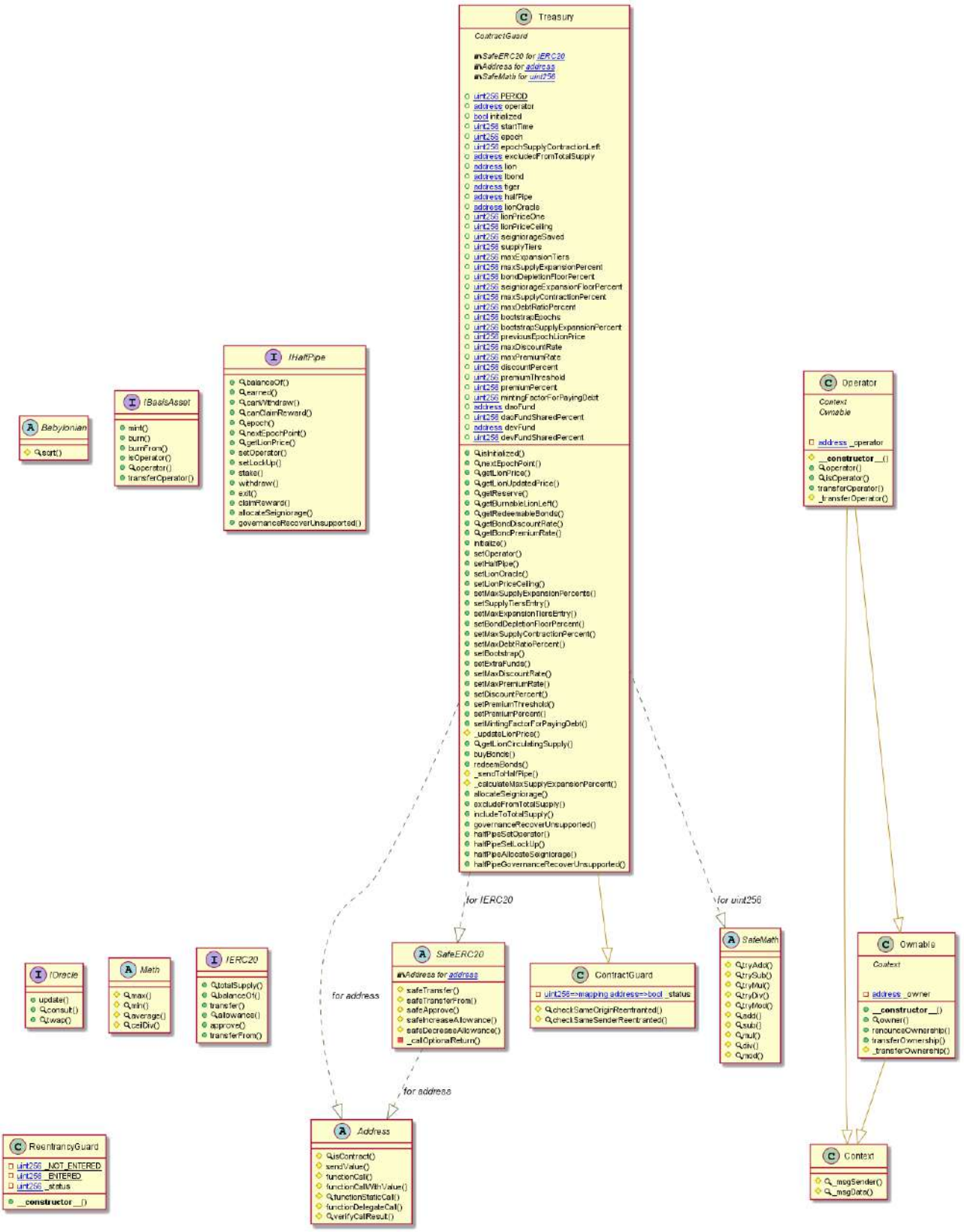


Email: audit@EtherAuthority.io

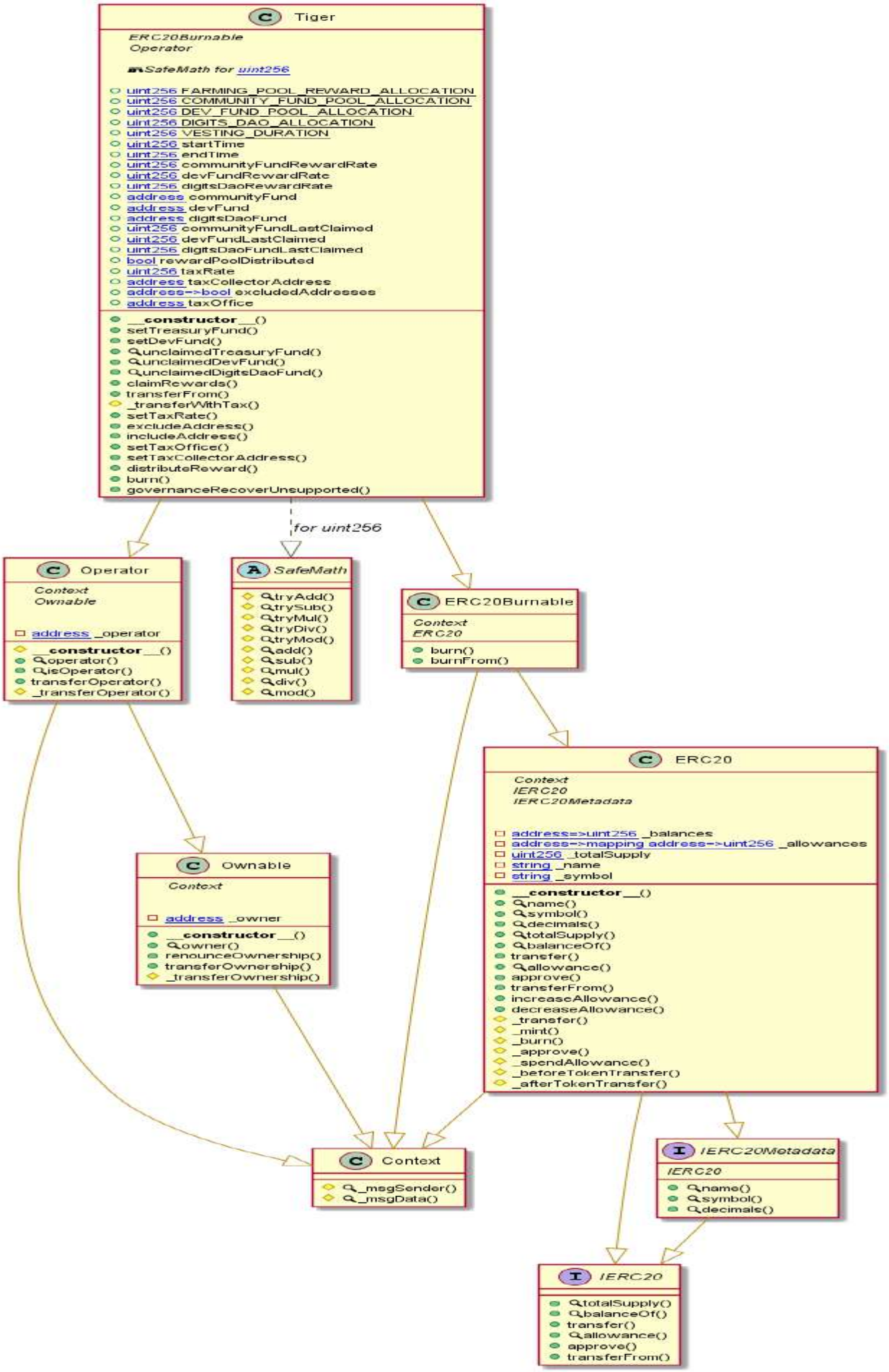
Lion Diagram



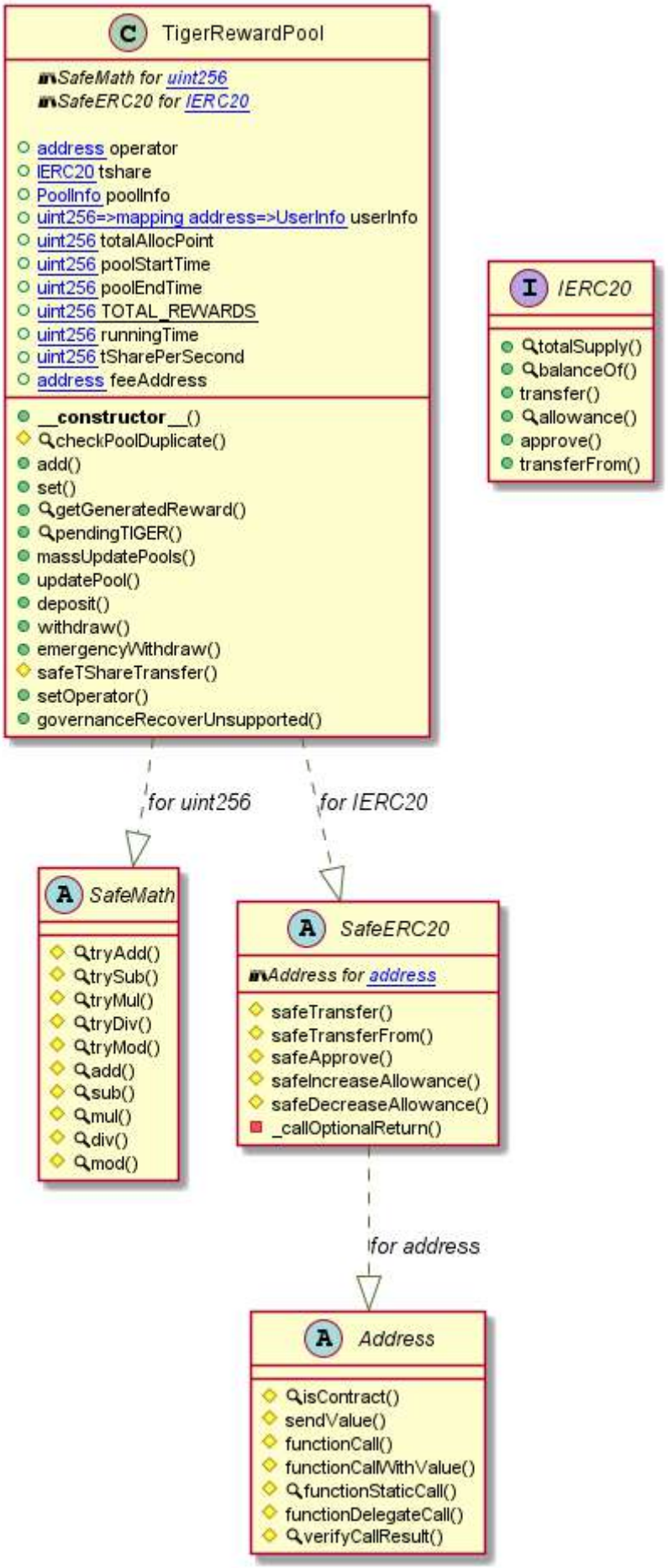
Treasury Diagram



Tiger Diagram



TigerRewardPool Diagram



Slither Results Log

Slither log >> Scrub.sol

```
INFO:Detectors:
HalfPipe.setOperator(address) (HalfPipe.sol#766-768) should emit an event for:
        - operator = _operator (HalfPipe.sol#767)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
HalfPipe.setLockUp(uint256,uint256) (HalfPipe.sol#770-774) should emit an event for:
        - withdrawLockupEpochs = _withdrawLockupEpochs (HalfPipe.sol#772)
        - rewardLockupEpochs = _rewardLockupEpochs (HalfPipe.sol#773)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
HalfPipe.setOperator(address)._operator (HalfPipe.sol#766) lacks a zero-check on :
                - operator = _operator (HalfPipe.sol#767)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in HalfPipe.allocateSeigniorage(uint256) (HalfPipe.sol#861-881):
        External calls:
        - lion.safeTransferFrom(msg.sender,address(this),amount) (HalfPipe.sol#866)
        State variables written after the call(s):
        - masonryHistory.push(newSnapshot) (HalfPipe.sol#878)
Reentrancy in ShareWrapper.stake(uint256) (HalfPipe.sol#634-640):
        External calls:
        - share.safeTransferFrom(msg.sender,address(this),amount) (HalfPipe.sol#636)
        State variables written after the call(s):
        - _balances[msg.sender] = _balances[msg.sender].add(depositAmount) (HalfPipe.sol#639)
        - _totalSupply = _totalSupply.add(depositAmount) (HalfPipe.sol#638)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in HalfPipe.allocateSeigniorage(uint256) (HalfPipe.sol#861-881):
        External calls:
        - lion.safeTransferFrom(msg.sender,address(this),amount) (HalfPipe.sol#866)
        Event emitted after the call(s):
        - RewardAdded(msg.sender,amount) (HalfPipe.sol#880)
Reentrancy in HalfPipe.claimReward() (HalfPipe.sol#850-859):
        External calls:
        - lion.safeTransfer(msg.sender,reward) (HalfPipe.sol#856)
        Event emitted after the call(s):
        - RewardPaid(msg.sender,reward) (HalfPipe.sol#857)
```

```
Reentrancy in HalfPipe.stake(uint256) (HalfPipe.sol#831-836):
        External calls:
        - super.stake(amount) (HalfPipe.sol#833)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (HalfPipe.sol#609)
                - share.safeTransferFrom(msg.sender,address(this),amount) (HalfPipe.sol#636)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
        External calls sending eth:
        - super.stake(amount) (HalfPipe.sol#833)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
        Event emitted after the call(s):
        - Staked(msg.sender,amount) (HalfPipe.sol#835)
Reentrancy in HalfPipe.withdraw(uint256) (HalfPipe.sol#838-844):
        External calls:
        - claimReward() (HalfPipe.sol#841)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (HalfPipe.sol#609)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
                - lion.safeTransfer(msg.sender,reward) (HalfPipe.sol#856)
        - super.withdraw(amount) (HalfPipe.sol#842)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (HalfPipe.sol#609)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
                - share.safeTransfer(msg.sender,amount) (HalfPipe.sol#647)
        External calls sending eth:
        - claimReward() (HalfPipe.sol#841)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
        - super.withdraw(amount) (HalfPipe.sol#842)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
        Event emitted after the call(s):
        - Withdrawn(msg.sender,amount) (HalfPipe.sol#843)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Address.verifyCallResult(bool,bytes,string) (HalfPipe.sol#223-243) uses assembly
        - INLINE ASM (HalfPipe.sol#235-238)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (HalfPipe.sol#107-109) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (HalfPipe.sol#136-142) is never used and should be removed
Address.functionDelegateCall(address,bytes) (HalfPipe.sol#196-198) is never used and should be removed
```

```
Address.functionDelegateCall(address,bytes,string) (HalfPipe.sol#206-215) is never used and should be removed
Address.functionStaticCall(address,bytes) (HalfPipe.sol#169-171) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (HalfPipe.sol#179-188) is never used and should be removed
Address.sendValue(address,uint256) (HalfPipe.sol#82-87) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (HalfPipe.sol#561-574) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (HalfPipe.sol#585-596) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (HalfPipe.sol#576-583) is never used and should be removed
SafeMath.div(uint256,uint256,string) (HalfPipe.sol#496-505) is never used and should be removed
SafeMath.mod(uint256,uint256) (HalfPipe.sol#456-458) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (HalfPipe.sol#522-531) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (HalfPipe.sol#473-482) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (HalfPipe.sol#327-333) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (HalfPipe.sol#369-374) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (HalfPipe.sol#381-386) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (HalfPipe.sol#352-362) is never used and should be removed
SafeMath.trySub(uint256,uint256) (HalfPipe.sol#340-345) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (HalfPipe.sol#82-87):
        - (success) = recipient.call{value: amount}() (HalfPipe.sol#85)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (HalfPipe.sol#150-161):
        - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
Low level call in Address.functionStaticCall(address,bytes,string) (HalfPipe.sol#179-188):
        - (success,returndata) = target.staticcall(data) (HalfPipe.sol#186)
Low level call in Address.functionDelegateCall(address,bytes,string) (HalfPipe.sol#206-215):
        - (success,returndata) = target.delegatecall(data) (HalfPipe.sol#213)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
```

```
INFO:Detectors:
Parameter HalfPipe.initialize(IERC20,IERC20,ITreasury)._lion (HalfPipe.sol#747) is not in mixedCase
Parameter HalfPipe.initialize(IERC20,IERC20,ITreasury)._share (HalfPipe.sol#748) is not in mixedCase
Parameter HalfPipe.initialize(IERC20,IERC20,ITreasury)._treasury (HalfPipe.sol#749) is not in mixedCase
Parameter HalfPipe.setOperator(address)._operator (HalfPipe.sol#766) is not in mixedCase
Parameter HalfPipe.setLockUp(uint256,uint256)._withdrawLockupEpochs (HalfPipe.sol#770) is not in mixedCase
Parameter HalfPipe.setLockUp(uint256,uint256)._rewardLockupEpochs (HalfPipe.sol#770) is not in mixedCase
Parameter HalfPipe.governanceRecoverUnsupported(IERC20,uint256,address)._token (HalfPipe.sol#883) is not in mixedCase
Parameter HalfPipe.governanceRecoverUnsupported(IERC20,uint256,address)._amount (HalfPipe.sol#883) is not in mixedCase
Parameter HalfPipe.governanceRecoverUnsupported(IERC20,uint256,address)._to (HalfPipe.sol#883) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
initialize(IERC20,IERC20,ITreasury) should be declared external:
        - HalfPipe.initialize(IERC20,IERC20,ITreasury) (HalfPipe.sol#746-764)
rewardPerShare() should be declared external:
        - HalfPipe.rewardPerShare() (HalfPipe.sol#818-820)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:HalfPipe.sol analyzed (9 contracts with 75 detectors), 46 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> LBond.sol

```
INFO:Detectors:
Context._msgData() (LBond.sol#104-106) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (LBond.sol#137-139)
symbol() should be declared external:
        - ERC20.symbol() (LBond.sol#145-147)
decimals() should be declared external:
        - ERC20.decimals() (LBond.sol#162-164)
totalSupply() should be declared external:
        - ERC20.totalSupply() (LBond.sol#169-171)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (LBond.sol#188-192)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (LBond.sol#211-215)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (LBond.sol#233-242)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (LBond.sol#256-260)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (LBond.sol#276-285)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (LBond.sol#520-522)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (LBond.sol#528-531)
operator() should be declared external:
        - Operator.operator() (LBond.sol#553-555)
isOperator() should be declared external:
        - Operator.isOperator() (LBond.sol#562-564)
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (LBond.sol#566-568)
mint(address,uint256) should be declared external:
        - LBond.mint(address,uint256) (LBond.sol#589-595)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:LBond.sol analyzed (8 contracts with 75 detectors), 16 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Lion.sol

```
INFO:Detectors:
Lion.setBurnThreshold(uint256) (Lion.sol#1070-1072) should emit an event for:
        - burnThreshold = _burnThreshold (Lion.sol#1071)
Lion.setTaxRate(uint256) (Lion.sol#1118-1122) should emit an event for:
        - taxRate = _taxRate (Lion.sol#1121)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Variable 'Lion._getLionPrice()._price (Lion.sol#1075)' in Lion._getLionPrice() (Lion.sol#1074-1080) potentially used before declaration
uint256(_price) (Lion.sol#1076)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables
INFO:Detectors:
Lion._updateTaxRate(uint256) (Lion.sol#1082-1092) has costly operations inside a loop:
        - taxRate = taxTiersRates[tierId] (Lion.sol#1087)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop
INFO:Detectors:
Context._msgData() (Lion.sol#290-292) is never used and should be removed
Math.average(uint256,uint256) (Lion.sol#31-34) is never used and should be removed
Math.ceilDiv(uint256,uint256) (Lion.sol#42-45) is never used and should be removed
Math.max(uint256,uint256) (Lion.sol#16-18) is never used and should be removed
Math.min(uint256,uint256) (Lion.sol#23-25) is never used and should be removed
SafeMath.add(uint256,uint256) (Lion.sol#838-840) is never used and should be removed
SafeMath.div(uint256,uint256,string) (Lion.sol#936-945) is never used and should be removed
SafeMath.mod(uint256,uint256) (Lion.sol#896-898) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Lion.sol#962-971) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (Lion.sol#767-773) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (Lion.sol#809-814) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (Lion.sol#821-826) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (Lion.sol#792-802) is never used and should be removed
SafeMath.trySub(uint256,uint256) (Lion.sol#780-785) is never used and should be removed
SafeMath8.add(uint8,uint8) (Lion.sol#59-64) is never used and should be removed
SafeMath8.div(uint8,uint8) (Lion.sol#133-135) is never used and should be removed
SafeMath8.div(uint8,uint8,string) (Lion.sol#149-155) is never used and should be removed
SafeMath8.mod(uint8,uint8) (Lion.sol#169-171) is never used and should be removed
SafeMath8.mod(uint8,uint8,string) (Lion.sol#185-188) is never used and should be removed
SafeMath8.mul(uint8,uint8) (Lion.sol#107-119) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

```
INFO:Detectors:
Parameter Lion.isAddressExcluded(address)._address (Lion.sol#1046) is not in mixedCase
Parameter Lion.setTaxTiersTwap(uint8,uint256)._index (Lion.sol#1050) is not in mixedCase
Parameter Lion.setTaxTiersTwap(uint8,uint256)._value (Lion.sol#1050) is not in mixedCase
Parameter Lion.setTaxTiersRate(uint8,uint256)._index (Lion.sol#1063) is not in mixedCase
Parameter Lion.setTaxTiersRate(uint8,uint256)._value (Lion.sol#1063) is not in mixedCase
Parameter Lion.setBurnThreshold(uint256)._burnThreshold (Lion.sol#1070) is not in mixedCase
Parameter Lion.setOracle(address)._oracle (Lion.sol#1102) is not in mixedCase
Parameter Lion.setTaxOffice(address)._taxOffice (Lion.sol#1107) is not in mixedCase
Parameter Lion.setTaxCollectorAddress(address)._taxCollectorAddress (Lion.sol#1113) is not in mixedCase
Parameter Lion.setTaxRate(uint256)._taxRate (Lion.sol#1118) is not in mixedCase
Parameter Lion.setBurnTax(bool)._burnTax (Lion.sol#1124) is not in mixedCase
Parameter Lion.excludeAddress(address)._address (Lion.sol#1128) is not in mixedCase
Parameter Lion.includeAddress(address)._address (Lion.sol#1134) is not in mixedCase
Parameter Lion.governanceRecoverUnsupported(IERC20,uint256,address)._token (Lion.sol#1222) is not in mixedCase
Parameter Lion.governanceRecoverUnsupported(IERC20,uint256,address)._amount (Lion.sol#1223) is not in mixedCase
Parameter Lion.governanceRecoverUnsupported(IERC20,uint256,address)._to (Lion.sol#1224) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Lion.slitherConstructorConstantVariables() (Lion.sol#974-1229) uses literals with too many digits:
        - INITIAL_LAUNCHPAD_DISTRIBUTION = 30000000000000000000000 (Lion.sol#979)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (Lion.sol#323-325)
symbol() should be declared external:
        - ERC20.symbol() (Lion.sol#331-333)
decimals() should be declared external:
        - ERC20.decimals() (Lion.sol#348-350)
totalSupply() should be declared external:
        - ERC20.totalSupply() (Lion.sol#355-357)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (Lion.sol#374-378)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (Lion.sol#397-401)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (Lion.sol#419-428)
```

```
        - Lion.transferFrom(address,address,uint256) (Lion.sol#1162-1186)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (Lion.sol#442-446)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (Lion.sol#462-471)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Lion.sol#705-707)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Lion.sol#713-716)
operator() should be declared external:
        - Operator.operator() (Lion.sol#738-740)
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (Lion.sol#751-753)
isAddressExcluded(address) should be declared external:
        - Lion.isAddressExcluded(address) (Lion.sol#1046-1048)
setTaxTiersTwap(uint8,uint256) should be declared external:
        - Lion.setTaxTiersTwap(uint8,uint256) (Lion.sol#1050-1061)
setTaxTiersRate(uint8,uint256) should be declared external:
        - Lion.setTaxTiersRate(uint8,uint256) (Lion.sol#1063-1068)
setBurnThreshold(uint256) should be declared external:
        - Lion.setBurnThreshold(uint256) (Lion.sol#1070-1072)
enableAutoCalculateTax() should be declared external:
        - Lion.enableAutoCalculateTax() (Lion.sol#1094-1096)
disableAutoCalculateTax() should be declared external:
        - Lion.disableAutoCalculateTax() (Lion.sol#1098-1100)
setOracle(address) should be declared external:
        - Lion.setOracle(address) (Lion.sol#1102-1105)
setTaxOffice(address) should be declared external:
        - Lion.setTaxOffice(address) (Lion.sol#1107-1111)
setTaxCollectorAddress(address) should be declared external:
        - Lion.setTaxCollectorAddress(address) (Lion.sol#1113-1116)
setTaxRate(uint256) should be declared external:
        - Lion.setTaxRate(uint256) (Lion.sol#1118-1122)
setBurnTax(bool) should be declared external:
        - Lion.setBurnTax(bool) (Lion.sol#1124-1126)
includeAddress(address) should be declared external:
        - Lion.includeAddress(address) (Lion.sol#1134-1138)
```

```
mint(address,uint256) should be declared external:
        - Lion.mint(address,uint256) (Lion.sol#1146-1152)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Lion.sol analyzed (12 contracts with 75 detectors), 73 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Oracle.sol

```
INFO:Detectors:
UniswapV2OracleLibrary.currentCumulativePrices(address) (Oracle.sol#188-212) uses timestamp for comparisons
        Dangerous comparisons:
        - blockTimestampLast != blockTimestamp (Oracle.sol#203)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Babylonian.sqrt(uint256) (Oracle.sol#6-18) is never used and should be removed
Context._msgData() (Oracle.sol#433-435) is never used and should be removed
FixedPoint.decode(FixedPoint.uq112x112) (Oracle.sol#70-72) is never used and should be removed
FixedPoint.div(FixedPoint.uq112x112,uint112) (Oracle.sol#49-52) is never used and should be removed
FixedPoint.encode(uint112) (Oracle.sol#39-41) is never used and should be removed
FixedPoint.encode144(uint144) (Oracle.sol#44-46) is never used and should be removed
FixedPoint.reciprocal(FixedPoint.uq112x112) (Oracle.sol#80-83) is never used and should be removed
FixedPoint.sqrt(FixedPoint.uq112x112) (Oracle.sol#86-88) is never used and should be removed
SafeMath.div(uint256,uint256) (Oracle.sol#334-336) is never used and should be removed
SafeMath.div(uint256,uint256,string) (Oracle.sol#390-399) is never used and should be removed
SafeMath.mod(uint256,uint256) (Oracle.sol#350-352) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Oracle.sol#416-425) is never used and should be removed
SafeMath.mul(uint256,uint256) (Oracle.sol#320-322) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (Oracle.sol#367-376) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (Oracle.sol#221-227) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (Oracle.sol#263-268) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (Oracle.sol#275-280) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (Oracle.sol#246-256) is never used and should be removed
SafeMath.trySub(uint256,uint256) (Oracle.sol#234-239) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

```
INFO:Detectors:
Struct FixedPoint.uq112x112 (Oracle.sol#24-26) is not in CapWords
Struct FixedPoint.uq144x112 (Oracle.sol#30-32) is not in CapWords
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (Oracle.sol#117) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (Oracle.sol#119) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (Oracle.sol#138) is not in mixedCase
Parameter Epoch.setPeriod(uint256)._period (Oracle.sol#598) is not in mixedCase
Parameter Epoch.setEpoch(uint256)._epoch (Oracle.sol#603) is not in mixedCase
Parameter Oracle.consult(address,uint256)._token (Oracle.sol#670) is not in mixedCase
Parameter Oracle.consult(address,uint256)._amountIn (Oracle.sol#670) is not in mixedCase
Parameter Oracle.twap(address,uint256)._token (Oracle.sol#679) is not in mixedCase
Parameter Oracle.twap(address,uint256)._amountIn (Oracle.sol#679) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable UniswapV2OracleLibrary.currentCumulativePrices(address).price0Cumulative (Oracle.sol#192) is too similar to UniswapV2OracleLib
ry.currentCumulativePrices(address).price1Cumulative (Oracle.sol#193)
Variable Oracle.price0Average (Oracle.sol#624) is too similar to Oracle.price1Average (Oracle.sol#625)
Variable Oracle.twap(address,uint256).price0Cumulative (Oracle.sol#680) is too similar to Oracle.update().price1Cumulative (Oracle.sol#
9)
Variable Oracle.update().price0Cumulative (Oracle.sol#649) is too similar to Oracle.update().price1Cumulative (Oracle.sol#649)
Variable Oracle.twap(address,uint256).price0Cumulative (Oracle.sol#680) is too similar to Oracle.twap(address,uint256).price1Cumulative
Oracle.sol#680)
Variable Oracle.price0CumulativeLast (Oracle.sol#622) is too similar to Oracle.price1CumulativeLast (Oracle.sol#623)
Variable Oracle.update().price0Cumulative (Oracle.sol#649) is too similar to Oracle.twap(address,uint256).price1Cumulative (Oracle.sol#
0)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Oracle.sol#471-473)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Oracle.sol#479-482)
isOperator() should be declared external:
        - Operator.isOperator() (Oracle.sol#513-515)
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (Oracle.sol#517-519)
getCurrentEpoch() should be declared external:
        - Epoch.getCurrentEpoch() (Oracle.sol#576-578)
```

```
getPeriod() should be declared external:
        - Epoch.getPeriod() (Oracle.sol#580-582)
getStartTime() should be declared external:
        - Epoch.getStartTime() (Oracle.sol#584-586)
getLastEpochTime() should be declared external:
        - Epoch.getLastEpochTime() (Oracle.sol#588-590)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Oracle.sol analyzed (10 contracts with 75 detectors), 48 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Tiger.sol

```
INFO:Detectors:
Tiger.setTaxOffice(address) (Tiger.sol#958-961) should emit an event for:
        - taxOffice = _taxOffice (Tiger.sol#960)
        - taxOffice = _taxOffice (Tiger.sol#960)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
Tiger.setTaxRate(uint256) (Tiger.sol#941-944) should emit an event for:
        - taxRate = _taxRate (Tiger.sol#943)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Tiger.setTreasuryFund(address)._communityFund (Tiger.sol#857) lacks a zero-check on :
                - communityFund = _communityFund (Tiger.sol#859)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Tiger.unclaimedTreasuryFund() (Tiger.sol#868-873) uses timestamp for comparisons
        Dangerous comparisons:
        - _now > endTime (Tiger.sol#870)
        - communityFundLastClaimed >= _now (Tiger.sol#871)
Tiger.unclaimedDevFund() (Tiger.sol#875-880) uses timestamp for comparisons
        Dangerous comparisons:
        - _now > endTime (Tiger.sol#877)
        - devFundLastClaimed >= _now (Tiger.sol#878)
Tiger.unclaimedDigitsDaoFund() (Tiger.sol#882-887) uses timestamp for comparisons
        Dangerous comparisons:
        - _now > endTime (Tiger.sol#884)
        - digitsDaoFundLastClaimed >= _now (Tiger.sol#885)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Context._msgData() (Tiger.sol#10-12) is never used and should be removed
SafeMath.add(uint256,uint256) (Tiger.sol#257-259) is never used and should be removed
SafeMath.div(uint256,uint256,string) (Tiger.sol#355-364) is never used and should be removed
SafeMath.mod(uint256,uint256) (Tiger.sol#315-317) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Tiger.sol#381-390) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (Tiger.sol#186-192) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (Tiger.sol#228-233) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (Tiger.sol#240-245) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (Tiger.sol#211-221) is never used and should be removed
```

```
SafeMath.trySub(uint256,uint256) (Tiger.sol#199-204) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Parameter Tiger.setTreasuryFund(address)._communityFund (Tiger.sol#857) is not in mixedCase
Parameter Tiger.setDevFund(address)._devFund (Tiger.sol#862) is not in mixedCase
Parameter Tiger.setTaxRate(uint256)._taxRate (Tiger.sol#941) is not in mixedCase
Parameter Tiger.excludeAddress(address)._address (Tiger.sol#946) is not in mixedCase
Parameter Tiger.includeAddress(address)._address (Tiger.sol#952) is not in mixedCase
Parameter Tiger.setTaxOffice(address)._taxOffice (Tiger.sol#958) is not in mixedCase
Parameter Tiger.setTaxCollectorAddress(address)._taxCollectorAddress (Tiger.sol#963) is not in mixedCase
Parameter Tiger.distributeReward(address,address)._farmingIncentiveFund (Tiger.sol#971) is not in mixedCase
Parameter Tiger.distributeReward(address,address)._tigerGenesisRewardPool (Tiger.sol#971) is not in mixedCase
Parameter Tiger.governanceRecoverUnsupported(IERC20,uint256,address)._token (Tiger.sol#984) is not in mixedCase
Parameter Tiger.governanceRecoverUnsupported(IERC20,uint256,address)._amount (Tiger.sol#985) is not in mixedCase
Parameter Tiger.governanceRecoverUnsupported(IERC20,uint256,address)._to (Tiger.sol#986) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Tiger.sol#48-50)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Tiger.sol#56-59)
operator() should be declared external:
        - Operator.operator() (Tiger.sol#81-83)
```

```
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (Tiger.sol#94-96)
name() should be declared external:
        - ERC20.name() (Tiger.sol#440-442)
symbol() should be declared external:
        - ERC20.symbol() (Tiger.sol#448-450)
decimals() should be declared external:
        - ERC20.decimals() (Tiger.sol#465-467)
totalSupply() should be declared external:
        - ERC20.totalSupply() (Tiger.sol#472-474)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (Tiger.sol#479-481)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (Tiger.sol#491-495)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (Tiger.sol#514-518)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (Tiger.sol#536-545)
        - Tiger.transferFrom(address,address,uint256) (Tiger.sol#910-923)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (Tiger.sol#559-563)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (Tiger.sol#579-588)
burnFrom(address,uint256) should be declared external:
        - ERC20Burnable.burnFrom(address,uint256) (Tiger.sol#783-786)
setTaxRate(uint256) should be declared external:
        - Tiger.setTaxRate(uint256) (Tiger.sol#941-944)
includeAddress(address) should be declared external:
        - Tiger.includeAddress(address) (Tiger.sol#952-956)
setTaxOffice(address) should be declared external:
        - Tiger.setTaxOffice(address) (Tiger.sol#958-961)
setTaxCollectorAddress(address) should be declared external:
        - Tiger.setTaxCollectorAddress(address) (Tiger.sol#963-966)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Tiger.sol analyzed (9 contracts with 75 detectors), 48 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Treasury.sol

```
INFO:Detectors:
Treasury.setOperator(address) (Treasury.sol#1116-1118) should emit an event for:
        - operator = _operator (Treasury.sol#1117)
Treasury.setHalfPipe(address) (Treasury.sol#1120-1122) should emit an event for:
        - halfPipe = _halfPipe (Treasury.sol#1121)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
Treasury.setLionPriceCeiling(uint256) (Treasury.sol#1128-1131) should emit an event for:
        - lionPriceCeiling = _lionPriceCeiling (Treasury.sol#1130)
Treasury.setMaxSupplyExpansionPercents(uint256) (Treasury.sol#1133-1136) should emit an event for:
        - maxSupplyExpansionPercent = _maxSupplyExpansionPercent (Treasury.sol#1135)
Treasury.setBondDepletionFloorPercent(uint256) (Treasury.sol#1159-1162) should emit an event for:
        - bondDepletionFloorPercent = _bondDepletionFloorPercent (Treasury.sol#1161)
Treasury.setMaxDebtRatioPercent(uint256) (Treasury.sol#1169-1172) should emit an event for:
        - maxDebtRatioPercent = _maxDebtRatioPercent (Treasury.sol#1171)
Treasury.setBootstrap(uint256,uint256) (Treasury.sol#1174-1179) should emit an event for:
        - bootstrapEpochs = _bootstrapEpochs (Treasury.sol#1177)
        - bootstrapSupplyExpansionPercent = _bootstrapSupplyExpansionPercent (Treasury.sol#1178)
Treasury.setExtraFunds(address,uint256,address,uint256) (Treasury.sol#1181-1195) should emit an event for:
        - daoFundSharedPercent = _daoFundSharedPercent (Treasury.sol#1192)
        - devFundSharedPercent = _devFundSharedPercent (Treasury.sol#1194)
Treasury.setMaxDiscountRate(uint256) (Treasury.sol#1197-1199) should emit an event for:
        - maxDiscountRate = _maxDiscountRate (Treasury.sol#1198)
Treasury.setMaxPremiumRate(uint256) (Treasury.sol#1201-1203) should emit an event for:
        - maxPremiumRate = _maxPremiumRate (Treasury.sol#1202)
Treasury.setDiscountPercent(uint256) (Treasury.sol#1205-1208) should emit an event for:
        - discountPercent = _discountPercent (Treasury.sol#1207)
Treasury.setPremiumThreshold(uint256) (Treasury.sol#1210-1214) should emit an event for:
        - premiumThreshold = _premiumThreshold (Treasury.sol#1213)
Treasury.setPremiumPercent(uint256) (Treasury.sol#1216-1219) should emit an event for:
        - premiumPercent = _premiumPercent (Treasury.sol#1218)
Treasury.setMintingFactorForPayingDebt(uint256) (Treasury.sol#1221-1224) should emit an event for:
        - mintingFactorForPayingDebt = _mintingFactorForPayingDebt (Treasury.sol#1223)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Treasury.initialize(address,address,address,address,address,uint256)._lion (Treasury.sol#1077) lacks a zero-check on :
                - lion = _lion (Treasury.sol#1084)
```

```
Treasury.initialize(address,address,address,address,address,uint256)._lbond (Treasury.sol#1078) lacks a zero-check on :
                - lbond = _lbond (Treasury.sol#1085)
Treasury.initialize(address,address,address,address,address,uint256)._tiger (Treasury.sol#1079) lacks a zero-check on :
                - tiger = _tiger (Treasury.sol#1086)
Treasury.initialize(address,address,address,address,address,uint256)._lionOracle (Treasury.sol#1080) lacks a zero-check on :
                - lionOracle = _lionOracle (Treasury.sol#1087)
Treasury.initialize(address,address,address,address,address,uint256)._halfPipe (Treasury.sol#1081) lacks a zero-check on :
                - halfPipe = _halfPipe (Treasury.sol#1088)
Treasury.setOperator(address)._operator (Treasury.sol#1116) lacks a zero-check on :
                - operator = _operator (Treasury.sol#1117)
Treasury.setHalfPipe(address)._halfPipe (Treasury.sol#1120) lacks a zero-check on :
                - halfPipe = _halfPipe (Treasury.sol#1121)
Treasury.setLionOracle(address)._lionOracle (Treasury.sol#1124) lacks a zero-check on :
                - lionOracle = _lionOracle (Treasury.sol#1125)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Treasury.getLionCirculatingSupply() (Treasury.sol#1232-1240) has external calls inside a loop: balanceExcluded = balanceExcluded.add(li
Erc20.balanceOf(excludedFromTotalSupply[entryId])) (Treasury.sol#1237)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
INFO:Detectors:
Variable 'Treasury.getLionPrice().price (Treasury.sol#994)' in Treasury.getLionPrice() (Treasury.sol#993-999) potentially used before d
laration: uint256(price) (Treasury.sol#995)
Variable 'Treasury.getLionUpdatedPrice().price (Treasury.sol#1002)' in Treasury.getLionUpdatedPrice() (Treasury.sol#1001-1007) potentia
y used before declaration: uint256(price) (Treasury.sol#1003)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables
```

```
INFO:Detectors:
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1332-1372):
        External calls:
        - _updateLionPrice() (Treasury.sol#1333)
                - IOracle(lionOracle).update() (Treasury.sol#1229)
        State variables written after the call(s):
        - _mse = _calculateMaxSupplyExpansionPercent(lionSupply).mul(1e14) (Treasury.sol#1346)
                - maxSupplyExpansionPercent = maxExpansionTiers[tierId] (Treasury.sol#1325)
        - previousEpochLionPrice = getLionPrice() (Treasury.sol#1334)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in Treasury._sendToHalfPipe(uint256) (Treasury.sol#1297-1320):
        External calls:
        - IBasisAsset(lion).mint(address(this),_amount) (Treasury.sol#1298)
        - IERC20(lion).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1303)
        Event emitted after the call(s):
        - DaoFundFunded(now,_daoFundSharedAmount) (Treasury.sol#1304)
Reentrancy in Treasury._sendToHalfPipe(uint256) (Treasury.sol#1297-1320):
        External calls:
        - IBasisAsset(lion).mint(address(this),_amount) (Treasury.sol#1298)
        - IERC20(lion).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1303)
        - IERC20(lion).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1310)
        Event emitted after the call(s):
        - DevFundFunded(now,_devFundSharedAmount) (Treasury.sol#1311)
Reentrancy in Treasury._sendToHalfPipe(uint256) (Treasury.sol#1297-1320):
        External calls:
        - IBasisAsset(lion).mint(address(this),_amount) (Treasury.sol#1298)
        - IERC20(lion).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1303)
        - IERC20(lion).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1310)
        - IERC20(lion).safeApprove(halfPipe,0) (Treasury.sol#1316)
        - IERC20(lion).safeApprove(halfPipe,_amount) (Treasury.sol#1317)
        - IHalfPipe(halfPipe).allocateSeigniorage(_amount) (Treasury.sol#1318)
        Event emitted after the call(s):
        - HalfPipeFunded(now,_amount) (Treasury.sol#1319)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1332-1372):
        External calls:
        - _updateLionPrice() (Treasury.sol#1333)
```

```
                - IOracle(lionOracle).update() (Treasury.sol#1229)
        - _sendToHalfPipe(lionSupply.mul(bootstrapSupplyExpansionPercent).div(10000)) (Treasury.sol#1338)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Treasury.sol#476)
                - IBasisAsset(lion).mint(address(this),_amount) (Treasury.sol#1298)
                - (success,returndata) = target.call{value: value}(data) (Treasury.sol#240)
                - IERC20(lion).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1303)
                - IERC20(lion).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1310)
                - IERC20(lion).safeApprove(halfPipe,0) (Treasury.sol#1316)
                - IERC20(lion).safeApprove(halfPipe,_amount) (Treasury.sol#1317)
                - IHalfPipe(halfPipe).allocateSeigniorage(_amount) (Treasury.sol#1318)
        External calls sending eth:
        - _sendToHalfPipe(lionSupply.mul(bootstrapSupplyExpansionPercent).div(10000)) (Treasury.sol#1338)
                - (success,returndata) = target.call{value: value}(data) (Treasury.sol#240)
        Event emitted after the call(s):
        - DaoFundFunded(now,_daoFundSharedAmount) (Treasury.sol#1304)
                - _sendToHalfPipe(lionSupply.mul(bootstrapSupplyExpansionPercent).div(10000)) (Treasury.sol#1338)
        - DevFundFunded(now,_devFundSharedAmount) (Treasury.sol#1311)
                - _sendToHalfPipe(lionSupply.mul(bootstrapSupplyExpansionPercent).div(10000)) (Treasury.sol#1338)
        - HalfPipeFunded(now,_amount) (Treasury.sol#1319)
                - _sendToHalfPipe(lionSupply.mul(bootstrapSupplyExpansionPercent).div(10000)) (Treasury.sol#1338)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1332-1372):
        External calls:
        - _updateLionPrice() (Treasury.sol#1333)
                - IOracle(lionOracle).update() (Treasury.sol#1229)
        - _sendToHalfPipe(_savedForHalfPipe) (Treasury.sol#1363)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Treasury.sol#476)
                - IBasisAsset(lion).mint(address(this),_amount) (Treasury.sol#1298)
                - (success,returndata) = target.call{value: value}(data) (Treasury.sol#240)
                - IERC20(lion).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1303)
                - IERC20(lion).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1310)
                - IERC20(lion).safeApprove(halfPipe,0) (Treasury.sol#1316)
                - IERC20(lion).safeApprove(halfPipe,_amount) (Treasury.sol#1317)
                - IHalfPipe(halfPipe).allocateSeigniorage(_amount) (Treasury.sol#1318)
        External calls sending eth:
        - _sendToHalfPipe(_savedForHalfPipe) (Treasury.sol#1363)
                - (success,returndata) = target.call{value: value}(data) (Treasury.sol#240)
        Event emitted after the call(s):
```

```
                - _sendToHalfPipe(_savedForHalfPipe) (Treasury.sol#1363)
        - DevFundFunded(now,_devFundSharedAmount) (Treasury.sol#1311)
                - _sendToHalfPipe(_savedForHalfPipe) (Treasury.sol#1363)
        - HalfPipeFunded(now,_amount) (Treasury.sol#1319)
                - _sendToHalfPipe(_savedForHalfPipe) (Treasury.sol#1363)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1332-1372):
        External calls:
        - _updateLionPrice() (Treasury.sol#1333)
                - IOracle(lionOracle).update() (Treasury.sol#1229)
        - _sendToHalfPipe(_savedForHalfPipe) (Treasury.sol#1363)
                - IBasisAsset(lion).mint(address(this),_amount) (Treasury.sol#1298)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Treasury.sol#476)
                - IERC20(lion).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1303)
                - (success,returndata) = target.call{value: value}(data) (Treasury.sol#240)
                - IERC20(lion).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1310)
                - IERC20(lion).safeApprove(halfPipe,0) (Treasury.sol#1316)
                - IERC20(lion).safeApprove(halfPipe,_amount) (Treasury.sol#1317)
                - IHalfPipe(halfPipe).allocateSeigniorage(_amount) (Treasury.sol#1318)
        - IBasisAsset(lion).mint(address(this),_savedForBond) (Treasury.sol#1367)
        External calls sending eth:
        - _sendToHalfPipe(_savedForHalfPipe) (Treasury.sol#1363)
                - (success,returndata) = target.call{value: value}(data) (Treasury.sol#240)
        Event emitted after the call(s):
        - TreasuryFunded(now,_savedForBond) (Treasury.sol#1368)
Reentrancy in Treasury.buyBonds(uint256,uint256) (Treasury.sol#1242-1269):
        External calls:
        - IBasisAsset(lion).burnFrom(msg.sender,_lionAmount) (Treasury.sol#1262)
        - IBasisAsset(lbond).mint(msg.sender,_bondAmount) (Treasury.sol#1263)
        - _updateLionPrice() (Treasury.sol#1266)
                - IOracle(lionOracle).update() (Treasury.sol#1229)
        Event emitted after the call(s):
        - BoughtBonds(msg.sender,_lionAmount,_bondAmount) (Treasury.sol#1268)
Reentrancy in Treasury.redeemBonds(uint256,uint256) (Treasury.sol#1271-1295):
        External calls:
        - IBasisAsset(lbond).burnFrom(msg.sender,_bondAmount) (Treasury.sol#1289)
        - IERC20(lion).safeTransfer(msg.sender,_lionAmount) (Treasury.sol#1290)
        - _updateLionPrice() (Treasury.sol#1292)
```

```
                - IOracle(lionOracle).update() (Treasury.sol#1229)
        Event emitted after the call(s):
        - RedeemedBonds(msg.sender,_lionAmount,_bondAmount) (Treasury.sol#1294)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Address.verifyCallResult(bool,bytes,string) (Treasury.sol#304-324) uses assembly
        - INLINE ASM (Treasury.sol#316-319)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Treasury._calculateMaxSupplyExpansionPercent(uint256) (Treasury.sol#1322-1330) has costly operations inside a loop:
        - maxSupplyExpansionPercent = maxExpansionTiers[tierId] (Treasury.sol#1325)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop
INFO:Detectors:
Address.functionCall(address,bytes) (Treasury.sol#188-190) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (Treasury.sol#217-223) is never used and should be removed
Address.functionDelegateCall(address,bytes) (Treasury.sol#277-279) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (Treasury.sol#287-296) is never used and should be removed
Address.functionStaticCall(address,bytes) (Treasury.sol#250-252) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (Treasury.sol#260-269) is never used and should be removed
Address.sendValue(address,uint256) (Treasury.sol#163-168) is never used and should be removed
Babylonian.sqrt(uint256) (Treasury.sol#7-19) is never used and should be removed
Context._msgData() (Treasury.sol#555-557) is never used and should be removed
Math.average(uint256,uint256) (Treasury.sol#96-99) is never used and should be removed
Math.ceilDiv(uint256,uint256) (Treasury.sol#107-110) is never used and should be removed
Math.max(uint256,uint256) (Treasury.sol#81-83) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (Treasury.sol#452-463) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (Treasury.sol#443-450) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (Treasury.sol#412-419) is never used and should be removed
SafeMath.div(uint256,uint256,string) (Treasury.sol#825-834) is never used and should be removed
SafeMath.mod(uint256,uint256) (Treasury.sol#785-787) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Treasury.sol#851-860) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (Treasury.sol#802-811) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (Treasury.sol#656-662) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (Treasury.sol#698-703) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (Treasury.sol#710-715) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (Treasury.sol#681-691) is never used and should be removed
SafeMath.trySub(uint256,uint256) (Treasury.sol#669-674) is never used and should be removed
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (Treasury.sol#163-168):
        - (success) = recipient.call{value: amount}() (Treasury.sol#166)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (Treasury.sol#231-242):
        - (success,returndata) = target.call{value: value}(data) (Treasury.sol#240)
Low level call in Address.functionStaticCall(address,bytes,string) (Treasury.sol#260-269):
        - (success,returndata) = target.staticcall(data) (Treasury.sol#267)
Low level call in Address.functionDelegateCall(address,bytes,string) (Treasury.sol#287-296):
        - (success,returndata) = target.delegatecall(data) (Treasury.sol#294)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter Treasury.initialize(address,address,address,address,address,uint256)._lion (Treasury.sol#1077) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._lbond (Treasury.sol#1078) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._tiger (Treasury.sol#1079) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._lionOracle (Treasury.sol#1080) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._halfPipe (Treasury.sol#1081) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._startTime (Treasury.sol#1082) is not in mixedCase
Parameter Treasury.setOperator(address)._operator (Treasury.sol#1116) is not in mixedCase
Parameter Treasury.setHalfPipe(address)._halfPipe (Treasury.sol#1120) is not in mixedCase
```

```
INFO:Detectors:
Variable Treasury.setExtraFunds(address,uint256,address,uint256)._daoFundSharedPercent (Treasury.sol#1183) is too similar to Treasury.s
ExtraFunds(address,uint256,address,uint256)._devFundSharedPercent (Treasury.sol#1185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
Treasury.initialize(address,address,address,address,address,uint256) (Treasury.sol#1076-1114) uses literals with too many digits:
        - supplyTiers = (0,200000000000000000000000,300000000000000000000000,600000000000000000000000,1200000000000000000000000,180
0000000000000000000000,2400000000000000000000000,6000000000000000000000000) (Treasury.sol#1095)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Treasury.sol#593-595)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Treasury.sol#601-604)
operator() should be declared external:
        - Operator.operator() (Treasury.sol#626-628)
isOperator() should be declared external:
        - Operator.isOperator() (Treasury.sol#635-637)
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (Treasury.sol#639-641)
isInitialized() should be declared external:
        - Treasury.isInitialized() (Treasury.sol#983-985)
getLionUpdatedPrice() should be declared external:
        - Treasury.getLionUpdatedPrice() (Treasury.sol#1001-1007)
getReserve() should be declared external:
        - Treasury.getReserve() (Treasury.sol#1010-1012)
getBurnableLionLeft() should be declared external:
        - Treasury.getBurnableLionLeft() (Treasury.sol#1014-1026)
getRedeemableBonds() should be declared external:
        - Treasury.getRedeemableBonds() (Treasury.sol#1028-1037)
initialize(address,address,address,address,address,uint256) should be declared external:
        - Treasury.initialize(address,address,address,address,address,uint256) (Treasury.sol#1076-1114)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Treasury.sol analyzed (15 contracts with 75 detectors), 134 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> TigerRewardPool.sol

```
INFO:Detectors:
TigerRewardPool.setOperator(address) (TigerRewardPool.sol#872-874) should emit an event for:
        - operator = _operator (TigerRewardPool.sol#873)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
TigerRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256) (TigerRewardPool.sol#669-713) should emit an event for:
        - totalAllocPoint = totalAllocPoint.add(_allocPoint) (TigerRewardPool.sol#711)
TigerRewardPool.set(uint256,uint256,uint16,uint16) (TigerRewardPool.sol#716-729) should emit an event for:
        - totalAllocPoint = totalAllocPoint.sub(pool.allocPoint).add(_allocPoint) (TigerRewardPool.sol#722-724)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
```

```
INFO:Detectors:
TigerRewardPool.constructor(address,uint256,address)._feeAddress (TigerRewardPool.sol#646) lacks a zero-check on :
                - feeAddress = _feeAddress (TigerRewardPool.sol#652)
TigerRewardPool.setOperator(address)._operator (TigerRewardPool.sol#872) lacks a zero-check on :
                - operator = _operator (TigerRewardPool.sol#873)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in TigerRewardPool.deposit(uint256,uint256) (TigerRewardPool.sol#791-815):
        External calls:
        - safeTShareTransfer(_sender,_pending) (TigerRewardPool.sol#799)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (TigerRewardPool.sol#582)
                - tshare.safeTransfer(_to,_tshareBal) (TigerRewardPool.sol#865)
                - (success,returndata) = target.call{value: value}(data) (TigerRewardPool.sol#346)
                - tshare.safeTransfer(_to,_amount) (TigerRewardPool.sol#867)
        External calls sending eth:
        - safeTShareTransfer(_sender,_pending) (TigerRewardPool.sol#799)
                - (success,returndata) = target.call{value: value}(data) (TigerRewardPool.sol#346)
        Event emitted after the call(s):
        - RewardPaid(_sender,_pending) (TigerRewardPool.sol#800)
Reentrancy in TigerRewardPool.deposit(uint256,uint256) (TigerRewardPool.sol#791-815):
        External calls:
        - safeTShareTransfer(_sender,_pending) (TigerRewardPool.sol#799)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (TigerRewardPool.sol#582)
                - tshare.safeTransfer(_to,_tshareBal) (TigerRewardPool.sol#865)
```

```
TigerRewardPool.pendingTIGER(uint256,address) (TigerRewardPool.sol#746-757) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp > pool.lastRewardTime && tokenSupply != 0 (TigerRewardPool.sol#751)
TigerRewardPool.massUpdatePools() (TigerRewardPool.sol#760-765) uses timestamp for comparisons
        Dangerous comparisons:
        - pid < length (TigerRewardPool.sol#762)
TigerRewardPool.updatePool(uint256) (TigerRewardPool.sol#768-788) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp <= pool.lastRewardTime (TigerRewardPool.sol#770)
TigerRewardPool.governanceRecoverUnsupported(IERC20,uint256,address) (TigerRewardPool.sol#876-887) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < poolEndTime + 7776000 (TigerRewardPool.sol#877)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.verifyCallResult(bool,bytes,string) (TigerRewardPool.sol#410-430) uses assembly
        - INLINE ASM (TigerRewardPool.sol#422-425)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (TigerRewardPool.sol#294-296) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (TigerRewardPool.sol#323-329) is never used and should be removed
Address.functionDelegateCall(address,bytes) (TigerRewardPool.sol#383-385) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (TigerRewardPool.sol#393-402) is never used and should be removed
Address.functionStaticCall(address,bytes) (TigerRewardPool.sol#356-358) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (TigerRewardPool.sol#366-375) is never used and should be removed
Address.sendValue(address,uint256) (TigerRewardPool.sol#269-274) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (TigerRewardPool.sol#534-547) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (TigerRewardPool.sol#558-569) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (TigerRewardPool.sol#549-556) is never used and should be removed
SafeMath.div(uint256,uint256,string) (TigerRewardPool.sol#180-189) is never used and should be removed
SafeMath.mod(uint256,uint256) (TigerRewardPool.sol#140-142) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (TigerRewardPool.sol#206-215) is never used and should be removed
```

```
SafeMath.tryMul(uint256,uint256) (TigerRewardPool.sol#36-46) is never used and should be removed
SafeMath.trySub(uint256,uint256) (TigerRewardPool.sol#24-29) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
TigerRewardPool.tSharePerSecond (TigerRewardPool.sol#635) is set pre-construction with a non-constant function or state variable:
        - TOTAL_REWARDS / runningTime
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state-variables
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (TigerRewardPool.sol#269-274):
        - (success) = recipient.call{value: amount}() (TigerRewardPool.sol#272)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (TigerRewardPool.sol#337-348):
        - (success,returndata) = target.call{value: value}(data) (TigerRewardPool.sol#346)
Low level call in Address.functionStaticCall(address,bytes,string) (TigerRewardPool.sol#366-375):
        - (success,returndata) = target.staticcall(data) (TigerRewardPool.sol#373)
Low level call in Address.functionDelegateCall(address,bytes,string) (TigerRewardPool.sol#393-402):
        - (success,returndata) = target.delegatecall(data) (TigerRewardPool.sol#400)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter TigerRewardPool.checkPoolDuplicate(IERC20)._token (TigerRewardPool.sol#661) is not in mixedCase
Parameter TigerRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256)._allocPoint (TigerRewardPool.sol#670) is not in mixedCase
Parameter TigerRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256)._token (TigerRewardPool.sol#671) is not in mixedCase
```

```
Parameter TigerRewardPool.set(uint256,uint256,uint16,uint16)._allocPoint (TigerRewardPool.sol#716) is not in mixedCase
Parameter TigerRewardPool.set(uint256,uint256,uint16,uint16)._depositFeeBP (TigerRewardPool.sol#716) is not in mixedCase
Parameter TigerRewardPool.set(uint256,uint256,uint16,uint16)._withdrawFeeBP (TigerRewardPool.sol#716) is not in mixedCase
Parameter TigerRewardPool.getGeneratedReward(uint256,uint256)._fromTime (TigerRewardPool.sol#732) is not in mixedCase
Parameter TigerRewardPool.getGeneratedReward(uint256,uint256)._toTime (TigerRewardPool.sol#732) is not in mixedCase
Parameter TigerRewardPool.pendingTIGER(uint256,address)._pid (TigerRewardPool.sol#746) is not in mixedCase
Parameter TigerRewardPool.pendingTIGER(uint256,address)._user (TigerRewardPool.sol#746) is not in mixedCase
Parameter TigerRewardPool.updatePool(uint256)._pid (TigerRewardPool.sol#768) is not in mixedCase
Parameter TigerRewardPool.deposit(uint256,uint256)._pid (TigerRewardPool.sol#791) is not in mixedCase
Parameter TigerRewardPool.deposit(uint256,uint256)._amount (TigerRewardPool.sol#791) is not in mixedCase
Parameter TigerRewardPool.withdraw(uint256,uint256)._pid (TigerRewardPool.sol#818) is not in mixedCase
Parameter TigerRewardPool.withdraw(uint256,uint256)._amount (TigerRewardPool.sol#818) is not in mixedCase
Parameter TigerRewardPool.emergencyWithdraw(uint256)._pid (TigerRewardPool.sol#844) is not in mixedCase
Parameter TigerRewardPool.safeTShareTransfer(address,uint256)._to (TigerRewardPool.sol#861) is not in mixedCase
Parameter TigerRewardPool.safeTShareTransfer(address,uint256)._amount (TigerRewardPool.sol#861) is not in mixedCase
Parameter TigerRewardPool.setOperator(address)._operator (TigerRewardPool.sol#872) is not in mixedCase
Parameter TigerRewardPool.governanceRecoverUnsupported(IERC20,uint256,address)._token (TigerRewardPool.sol#876) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
TigerRewardPool.runningTime (TigerRewardPool.sol#634) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
set(uint256,uint256,uint16,uint16) should be declared external:
        - TigerRewardPool.set(uint256,uint256,uint16,uint16) (TigerRewardPool.sol#716-729)
deposit(uint256,uint256) should be declared external:
        - TigerRewardPool.deposit(uint256,uint256) (TigerRewardPool.sol#791-815)
withdraw(uint256,uint256) should be declared external:
        - TigerRewardPool.withdraw(uint256,uint256) (TigerRewardPool.sol#818-841)
emergencyWithdraw(uint256) should be declared external:
        - TigerRewardPool.emergencyWithdraw(uint256) (TigerRewardPool.sol#844-858)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:TigerRewardPool.sol analyzed (5 contracts with 75 detectors), 81 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Software analysis result:

These software reported many false positive results and some are informational issues. So, those issues can be safely ignored.

Email: audit@EtherAuthority.io

Email: audit@EtherAuthority.io