EtherAuthority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project:	Snow Thrive Protocol
Platform:	Avalanche
Language:	Solidity
Date:	March 11th, 2022

Table of contents

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by the Snow Thrive team to perform the Security audit of the Snow Thrive 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 11th, 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

Snow Thrive Contract is an ERC20 smart contract, having functions like stake, withdraw, epoch, claimReward, burn, mint, add and update pool, deposit, etc.

Audit scope

Name	**Code Review and Security Analysis Report for Snow Thrive Protocol Smart Contracts**
Platform	**Avalanche / Solidity**
File 1	HalfPipe.sol
File 1 MD5 Hash	A51C068CDC9E26FD3164B6E7B80D817B
File 2	Oracle.sol
File 2 MD5 Hash	342F8657975AB65AC5804897748148DA
File 3	STBond.sol
File 3 MD5 Hash	7EABFF7B2CEDEC0EF5B019565DF77610
File 4	Thrive.sol
File 4 MD5 Hash	D9A3FB1989576691A05946EB64598087
File 5	Treasury.sol
File 5 MD5 Hash	37AEC2296DC52D4ED5231805075ECEA4

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

File 6	Powder.sol
File 6 MD5 Hash	84B737600773E2E36F9F3653F26CFC93
File 7	ThriveGenesisRewardPool.sol
File 7 MD5 Hash	8BF0D3B40A169150A02BB8BE7E426C07
File 8	PowderRewardPool.sol
File 8 MD5 Hash	80D884E0BA8DE5471605436F8198D4D5
File 9	PowderGenesisRewardPool.sol
File 9 MD5 Hash	823F944FFAC043DCBA53BDA3FA0497B7
Audit Date	March 11th,2022
Revise Audit Date	March 16th, 2022

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
File 1 HalfPipe.sol ● Withdraw Lockup Epochs: 6 Epochs ● Reward Lockup Epochs: 3	YES, This is valid.
File 2 Oracle.sol ● The Oracle contract can inherit Epoch class.	YES, This is valid.
File 3 STBond.sol ● Name: Snow Thrive Bonds ● Symbol: STBOND	YES, This is valid.
File 4 Thrive.sol ● Name: THRIVE ● Symbol: THRIVE ● Decimals: 18 ● Initial distribution for the genesis pools: 24000 THRIVE ● Burn Threshold: 1.1 THRIVE ● Tax Tiers rate: 14 ● Tax Tiers Twaps count:14 ● Total Supply: 1 THRIVE ● Maximum Tax : 4.9%	YES, This is valid.
File 5 Treasury.sol ● Period: 8 hours	YES, This is valid.
File 6 Powder.sol ● Name: Thrive Shares ● Symbol: POWDER ● Decimals: 18 ● Farming Pool Reward Allocation: 35000 POWDER	YES, This is valid.

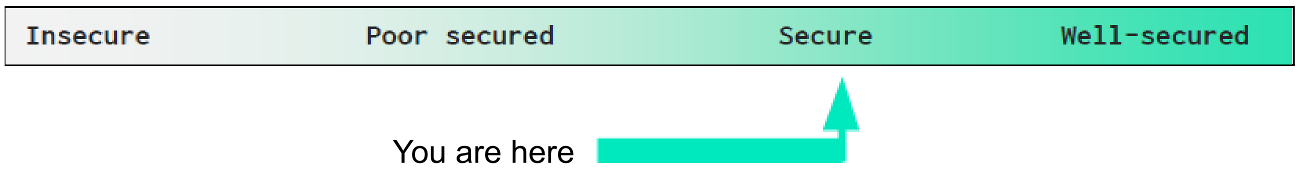
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

• Community Fund Pool Allocation: 5000 POWDER • Dev Fund Pool Allocation: 5000 POWDER • Digits Dao Allocation: 5000 POWDER • Powder Genesis Reward Allocation: 100 POWDER • Vesting Duration: 365 days • Total Supply: 1 POWDER	
File 7 ThriveGenesisRewardPool.sol • Total Rewards: 24000 THRIVE • Running Time: 72 hours • Maximum deposit fee : 5% • Maximum withdraw fee : 5%	**YES, This is valid. Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**
File 8 PowderRewardPool.sol • Total Rewards: 35000 POWDER • Running Time: 365 days • Maximum deposit fee : 5% • Maximum withdraw fee : 5%	**YES, This is valid. Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**
File 9 PowderGenesisRewardPool.sol • Total Rewards: 100 POWDER • Running Time: 72 hours • Maximum deposit fee : 5% • Maximum withdraw fee : 5%	**YES, This is valid. Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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 3 low and some very low level issues.

And those issues are resolved/acknowledged by the dev team and thus the contract is ready for the deployment.

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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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	Moderated
	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

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Code Quality

This audit scope has 10 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 Snow Thrive 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 Snow Thrive Protocol.

The Snow Thrive 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 Snow Thrive 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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

AS-IS overview

HalfPipe.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	getThrivePrice	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	allocateSeigniorage	external	access only One Block	No Issue
24	claimReward	write	Passed	No Issue
25	governanceRecoverUnsupported	external	access only One Block	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	onlyOneBlock	modifier	Passed	No Issue
31	checkSameOriginReentranted	internal	Passed	No Issue
32	checkSameSenderReentranted	internal	Passed	No Issue

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Oracle.sol

Functions

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

STBond.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	access only Operator	No Issue
5	name	read	Passed	No Issue
6	symbol	read	Passed	No Issue
7	decimals	read	Passed	No Issue
8	totalSupply	read	Passed	No Issue
9	balanceOf	read	Passed	No Issue
10	transfer	write	Passed	No Issue
11	allowance	read	Passed	No Issue
12	approve	write	Passed	No Issue
13	transferFrom	write	Passed	No Issue
14	increaseAllowance	write	Passed	No Issue
15	decreaseAllowance	write	Passed	No Issue
16	_transfer	internal	Passed	No Issue
17	_mint	internal	Passed	No Issue
18	_burn	internal	Passed	No Issue
19	_approve	internal	Passed	No Issue

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

20	_beforeTokenTransfer	internal	Passed	No Issue
21	_afterTokenTransfer	internal	Passed	No Issue
22	burn	write	Passed	No Issue
23	burnFrom	write	Passed	No Issue
24	owner	read	Passed	No Issue
25	onlyOwner	modifier	Passed	No Issue
26	renounceOwnership	write	access only Owner	No Issue
27	transferOwnership	write	access only Owner	No Issue
28	_transferOwnership	internal	Passed	No Issue
29	operator	read	Passed	No Issue
30	onlyOperator	modifier	Passed	No Issue
31	isOperator	read	Passed	No Issue
32	transferOperator	write	access only Owner	No Issue
33	_transferOperator	internal	Passed	No Issue

Thrive.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	name	read	Passed	No Issue
3	symbol	read	Passed	No Issue
4	decimals	read	Passed	No Issue
5	totalSupply	read	Passed	No Issue
6	balanceOf	read	Passed	No Issue
7	transfer	write	Passed	No Issue
8	allowance	read	Passed	No Issue
9	approve	write	Passed	No Issue
10	transferFrom	write	Passed	No Issue
11	increaseAllowance	write	Passed	No Issue
12	decreaseAllowance	write	Passed	No Issue
13	_transfer	internal	Passed	No Issue
14	_mint	internal	Passed	No Issue
15	_burn	internal	Passed	No Issue
16	_approve	internal	Passed	No Issue
17	_beforeTokenTransfer	internal	Passed	No Issue
18	_afterTokenTransfer	internal	Passed	No Issue
19	burn	write	Passed	No Issue
20	burnFrom	write	Passed	No Issue
21	owner	read	Passed	No Issue
22	onlyOwner	modifier	Passed	No Issue
23	renounceOwnership	write	access only Owner	No Issue
24	transferOwnership	write	access only Owner	No Issue
25	_transferOwnership	internal	Passed	No Issue
26	operator	read	Passed	No Issue
27	onlyOperator	modifier	Passed	No Issue
28	isOperator	read	Passed	No Issue

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io

29	transferOperator	write	access only Owner	No Issue
30	_transferOperator	internal	Passed	No Issue
31	onlyTaxOffice	modifier	Passed	No Issue
32	onlyOperatorOrTaxOffice	modifier	Passed	No Issue
33	getTaxTiersTwapsCount	read	Passed	No Issue
34	getTaxTiersRatesCount	read	Passed	No Issue
35	isAddressExcluded	read	Passed	No Issue
36	setTaxTiersTwap	write	access only Tax Office	No Issue
37	setTaxTiersRate	write	access only Tax Office	No Issue
38	setBurnThreshold	write	access only Tax Office	No Issue
39	_getThrivePrice	internal	Passed	No Issue
40	_updateTaxRate	internal	Passed	No Issue
41	enableAutoCalculateTax	write	access only Tax Office	No Issue
42	disableAutoCalculateTax	write	access only Tax Office	No Issue
43	setThriveOracle	write	access only Operator Or Tax Office	No Issue
44	setTaxOffice	write	access only Tax Office	No Issue
45	setTaxCollectorAddress	write	access only Tax Office	No Issue
46	setTaxRate	write	access only Tax Office	No Issue
47	excludeAddress	write	access only Tax Office	No Issue
48	includeAddress	write	access only Tax Office	No Issue
49	mint	write	access only Operator	No Issue
50	burn	write	Passed	No Issue
51	burnFrom	write	access only Operator	No Issue
52	transferFrom	write	Passed	No Issue
53	_transferWithTax	internal	Passed	No Issue
54	distributeReward	external	access only Operator	No Issue
55	governanceRecoverUnsupported	external	Function input parameters lack of check	Refer Audit Findings

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

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io

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	getThrivePrice	read	Passed	No Issue
10	getThriveUpdatedPrice	read	Passed	No Issue
11	getReserver	read	Passed	No Issue
12	getBurnableThriveLeft	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	setMasonry	external	access only Operator	No Issue
19	setThriveOracle	external	access only Operator	No Issue
20	setThrivePriceCeiling	external	access only Operator	No Issue
21	setMaxSupplyExpansionPercent s	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	setMaxSupplyContractionPercen t	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

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

35	_updateThrivePrice	internal	Passed	No Issue
36	getThriveCirculatingSupply	read	Passed	No Issue
37	buyBonds	external	access only One Block	No Issue
38	redeemBonds	external	access only One Block	No Issue
39	_sendToMasonry	internal	Passed	No Issue
40	_calculateMaxSupplyExpansion Percent	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	Function input parameters lack of check	Refer Audit Findings
45	masonrySetOperator	external	access only Operator	No Issue
46	masonrySetLockUp	external	access only Operator	No Issue
47	masonryAllocateSeigniorage	external	access only Operator	No Issue
48	masonryGovernanceRecoverUn supported	external	Function input parameters lack of check	Refer Audit Findings
49	checkSameOriginReentranted	internal	Passed	No Issue
50	checkSameSenderReentranted	internal	Passed	No Issue
51	onlyOneBlock	modifier	Passed	No Issue

Powder.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	name	read	Passed	No Issue
3	symbol	read	Passed	No Issue
4	decimals	read	Passed	No Issue
5	totalSupply	read	Passed	No Issue
6	balanceOf	read	Passed	No Issue
7	transfer	write	Passed	No Issue
8	allowance	read	Passed	No Issue
9	approve	write	Passed	No Issue
10	transferFrom	write	Passed	No Issue
11	increaseAllowance	write	Passed	No Issue
12	decreaseAllowance	write	Passed	No Issue
13	_transfer	internal	Passed	No Issue

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

14	_mint	internal	Passed	No Issue
15	_burn	internal	Passed	No Issue
16	_approve	internal	Passed	No Issue
17	_beforeTokenTransfer	internal	Passed	No Issue
18	_afterTokenTransfer	internal	Passed	No Issue
19	burn	write	Passed	No Issue
20	burnFrom	write	Passed	No Issue
21	owner	read	Passed	No Issue
22	onlyOwner	modifier	Passed	No Issue
23	renounceOwnership	write	access only Owner	No Issue
24	transferOwnership	write	access only Owner	No Issue
25	_transferOwnership	internal	Passed	No Issue
26	operator	read	Passed	No Issue
27	onlyOperator	modifier	Passed	No Issue
28	isOperator	read	Passed	No Issue
29	transferOperator	write	access only Owner	No Issue
30	_transferOperator	internal	Passed	No Issue
31	setTreasuryFund	external	Passed	No Issue
32	setDevFund	external	Passed	No Issue
33	unclaimedTreasuryFund	read	Passed	No Issue
34	unclaimedDevFund	read	Passed	No Issue
35	unclaimedDigitsDaoFund	read	Passed	No Issue
36	claimRewards	external	Passed	No Issue
37	distributeReward	external	access only Operator	No Issue
38	burn	write	Passed	No Issue
39	governanceRecoverUnsupported	external	Function input parameters lack of check	Refer Audit Findings

ThriveGenesisRewardPool.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	pendingTHRIVE	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	safeThriveTransfer	internal	Passed	No Issue
14	setOperator	external	access only Operator	No Issue

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

15	governanceRecoverUnsupported	external	Function input parameters lack of check	Refer Audit Findings

PowderRewardPool.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	pendingShare	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	Function input parameters lack of check	Refer Audit Findings

PowderGenesisRewardPool.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	pendingPOWDER	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	safeThriveTransfer	internal	Passed	No Issue

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io

14	setOperator	external	access only Operator	No Issue
15	governanceRecoverUnsupported	external	Function input parameters lack of check	Refer Audit Findings

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
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) Function input parameters lack of check: **PowderGenesisRewardPool.sol, HalfPipe.sol, PowderRewardPool, ThriveGenesisRewardPool.sol, Powder.sol, Thrive.sol, Treasury.sol**

Variable validation is not performed in below functions : governanceRecoverUnsupported, masonryGovernanceRecoverUnsupported

Resolution: We advise using validation like address type variables should not be address(0).

(2) Missing Event Log: **PowderGenesisRewardPool.sol, PowderRewardPool, ThriveGenesisRewardPool.sol**

Some functions need an event log.

- add
- set
- updatePool
- governanceRecoverUnsupported

Resolution: We suggest adding a log for listed events.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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.

PowderGenesisRewardPool.sol

powder, poolStartTime, poolEndTime, feeAddress

PowderRewardPool.sol

tshare, poolStartTime, poolEndTime, feeAddress

ThriveGenesisRewardPool.sol

thrive, poolStartTime, poolEndTime, feeAddress

Treasury.sol

thrive, stbond, powder, seigniorageExpansionFloorPercent

Powder.sol

startTime, endTime, communityFundRewardRate, devFundRewardRate, digitsDaoRewardRate

Resolution: We suggest setting these variables as immutable.

(2) Make variables constant:

PowderGenesisRewardPool.sol

runningTime, powderPerSecond

PowderRewardPool.sol

runningTime, tSharePerSecond

ThriveGenesisRewardPool.sol

runningTime, thrivePerSecond

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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:

- update: The Oracle checkEpoch owner can update 1-day EMA price from Uniswap.
- mint: The STBond Operator owner can mint an amount of token.
- burnFrom: The STBond Operator owner can burn an amount from the address.
- setTaxTiersTwap: The Thrive TaxOffice owner can set tax tiers.
- setTaxTiersRate: The Thrive TaxOffice owner can set tax tiers rate.
- setBurnThreshold: The Thrive TaxOffice owner can set burn threshold.
- enableAutoCalculateTax: The Thrive TaxOffice owner can enable auto calculate tax.
- disableAutoCalculateTax: The Thrive TaxOffice owner can disable auto calculate tax.
- setThriveOracle: The Thrive Operator Or TaxOffice owner can set the thrive oracle address.
- setTaxOffice: The Thrive Operator Or TaxOffice owner can set tax office address.
- setTaxCollectorAddress: The Thrive TaxOffice owner can set tax collector address.
- setTaxRate: The Thrive TaxOffice owner can set the tax rate.
- excludeAddress: The Thrive Operator Or TaxOffice owner can check if the address can't be excluded.
- includeAddress: The Thrive Operator Or TaxOffice owner can check if the address can't be included.
- mint: The Thrive Operator owner can mints THRIVE to a recipient address.
- burnFrom: The Thrive Operator owner can burn an amount from the address.
- distributeReward: The Thrive Operator owner can distribute to the reward pool.
- governanceRecoverUnsupported: The Thrive Operator owner can governance recover unsupported.
- setOperator: The Treasury Operator can set the operator address.
- setMasonry: The Treasury Operator can set the masonry address.
- setThriveOracle: The Treasury Operator can set the thrive oracle address.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

- setThrivePriceCeiling: The Treasury Operator can set the thrive price ceiling amount.
- setMaxSupplyExpansionPercents: The Treasury Operator can set maximum supply expansion percentages.
- setSupplyTiersEntry: The Treasury Operator can set supply tires entry value.
- 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 range.
- setExtraFunds: The Treasury Operator can set extra funds.
- setMaxDiscountRate: The Treasury Operator can set a maximum discount rate.
- setMaxPremiumRate: The Treasury Operator can set the maximum premium rate.
- setDiscountPercent: The Treasury Operator can set the discount percentage.
- setPremiumThreshold: The Treasury Operator can set a premium threshold.
- setPremiumPercent: The Treasury Operator can set a premium percentage.
- setMintingFactorForPayingDebt: The Treasury Operator can set the minting factor for paying debt.
- buyBonds: The Treasury Operator can buy bonds amount.
- redeemBonds: The Treasury OneBlock can redeem bonds amount.
- allocateSeigniorage: The Treasury OneBlock can allocate seigniorage.
- excludeFromTotalSupply: The Treasury Operator can check if the address is excluded From TotalSupply or not.
- includeToTotalSupply: The Treasury Operator can check if the address is included From TotalSupply or not.
- governanceRecoverUnsupported: The Treasury Operator can transfer the amount to governance to recover unsupported addresses.
- masonrySetLockUp: The Treasury Operator can set masonry lockup.
- masonryAllocateSeigniorage: The Treasury Operator can set masonry allocate seigniorage.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

- masonryGovernanceRecoverUnsupported: The Treasury Operator can set masonry governance to recover unsupported token.
- distributeReward: The Powder Operator can distribute to the reward pool.
- governanceRecoverUnsupported: The Powder Operator can transfer governance and transfer the amount to governance to recover unsupported addresses.
- add: The ThriveGenesisRewardPool Operator owner can add a new lp to the pool.
- set: The ThriveGenesisRewardPool Operator owner can update the given pool's THRIVE allocation point.
- setOperator: The ThriveGenesisRewardPool Operator owner can set the operator address.
- governanceRecoverUnsupported: The ThriveGenesisRewardPool Operator owner can transfer the amount to governance to recover unsupported addresses.
- add: The PowderRewardPool Operator owner can add a new lp to the pool.
- set: The PowderRewardPool Operator owner can update the given pool's tSHARE allocation point.
- setOperator: The PowderRewardPool Operator owner can set the operator address.
- governanceRecoverUnsupported: The PowderRewardPool Operator owner can transfer the amount to governance to recover unsupported addresses.
- add: The PowderGenesisRewardPool Operator owner can add a new lp to the pool.
- set: The PowderGenesisRewardPool Operator owner can update the given pool's POWDER allocation point.
- setOperator: The PowderGenesisRewardPool Operator owner can set the operator address.
- governanceRecoverUnsupported: The PowderGenesisRewardPool Operator owner can transfer the amount to governance to recover unsupported addresses.
- setOperator: The HalfPipe Operator can set the operator address.
- setLockUp: The HalfPipe Operator can set lockup addresses.
- stake: The HalfPipe OneBlock can add new stake.
- withdraw: The HalfPipe OneBlock can withdraw an amount.
- allocateSeigniorage: The HalfPipe OneBlock can allocate seigniorage.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

- governanceRecoverUnsupported: The HalfPipe Operator can not allow drain core tokens.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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".**

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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

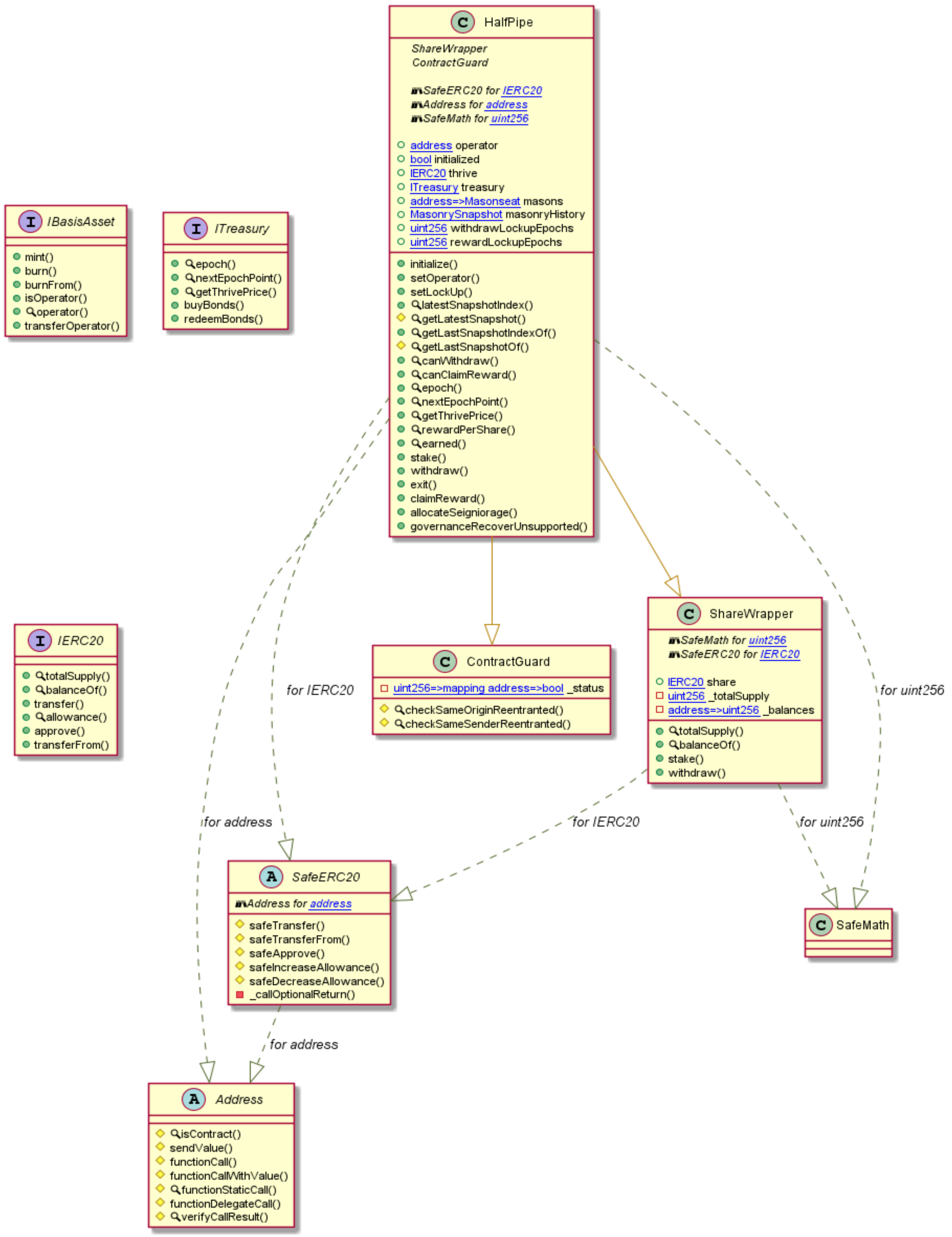
Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can’t guarantee explicit security of the audited smart contracts.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Appendix

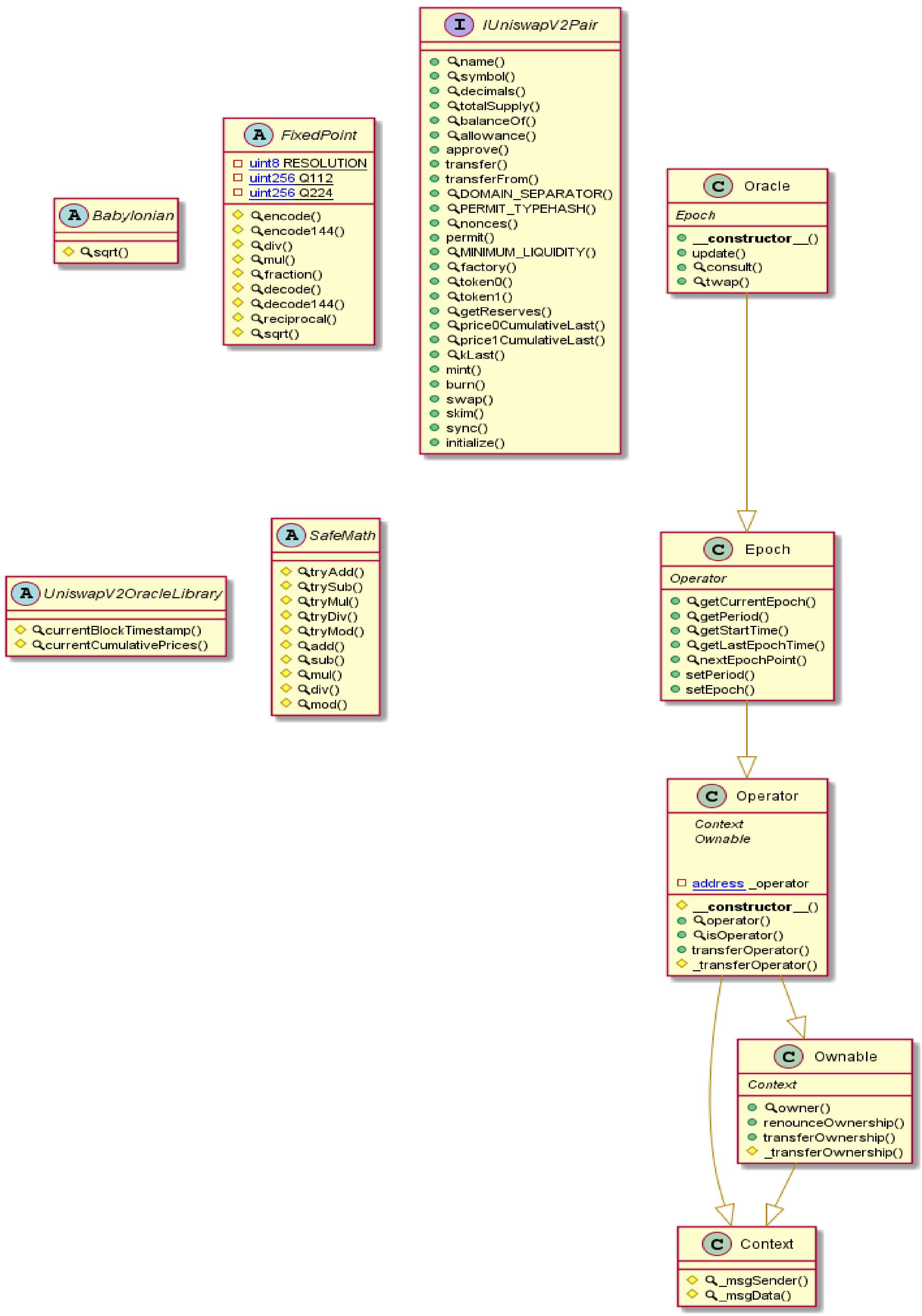
Code Flow Diagram - Snow Thrive Protocol

HalfPipe Diagram

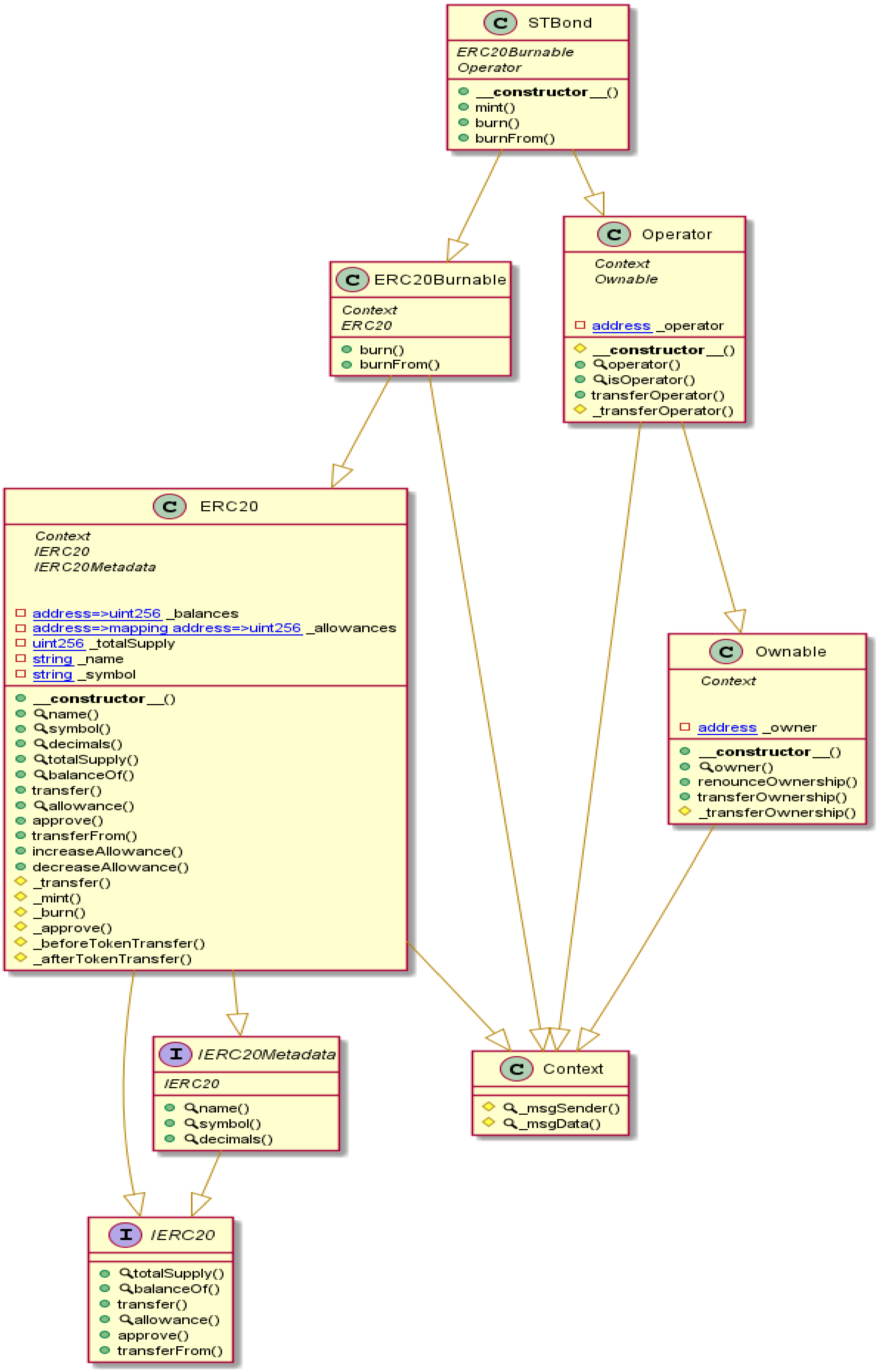


This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Oracle Diagram

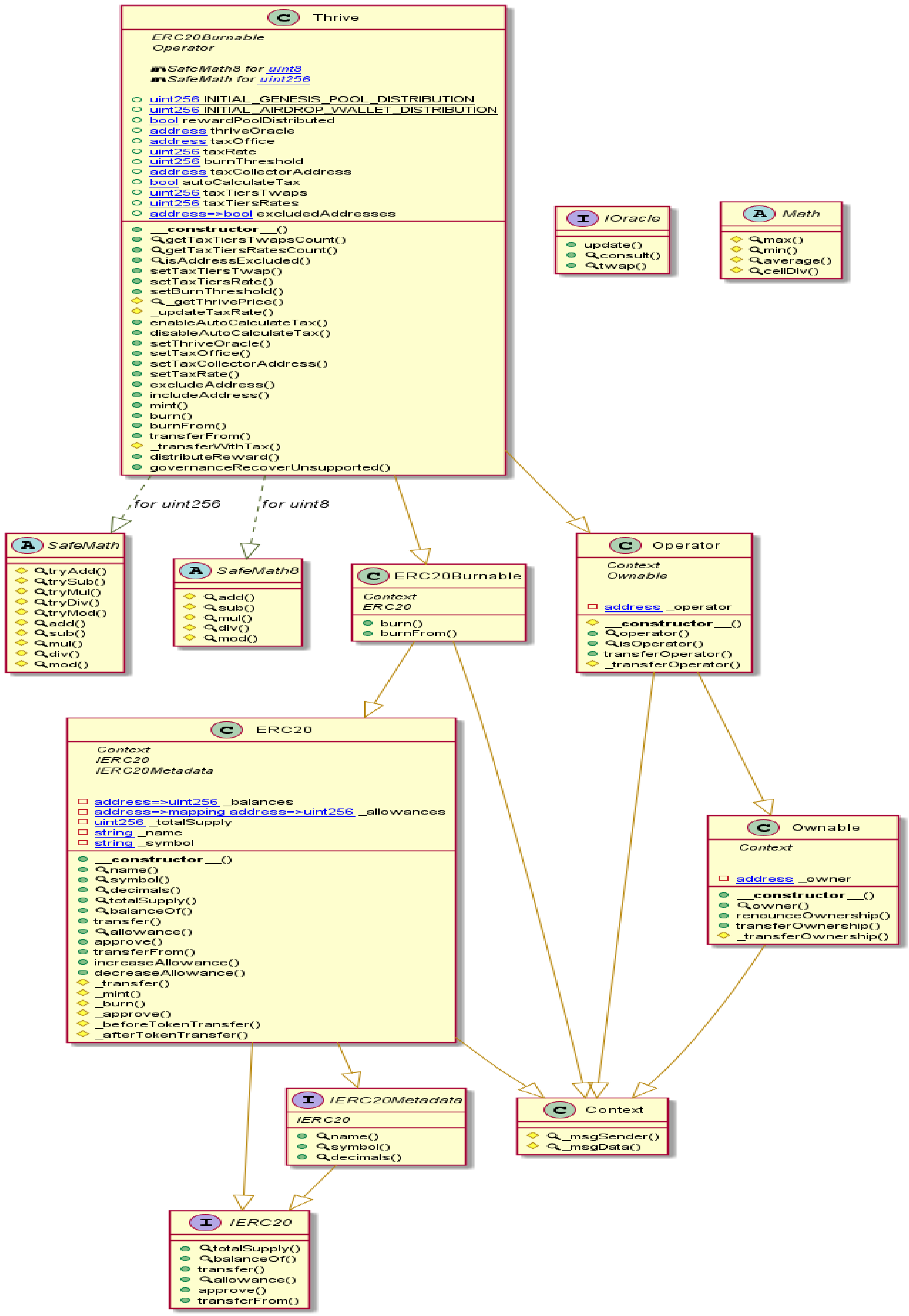


STBond Diagram



This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

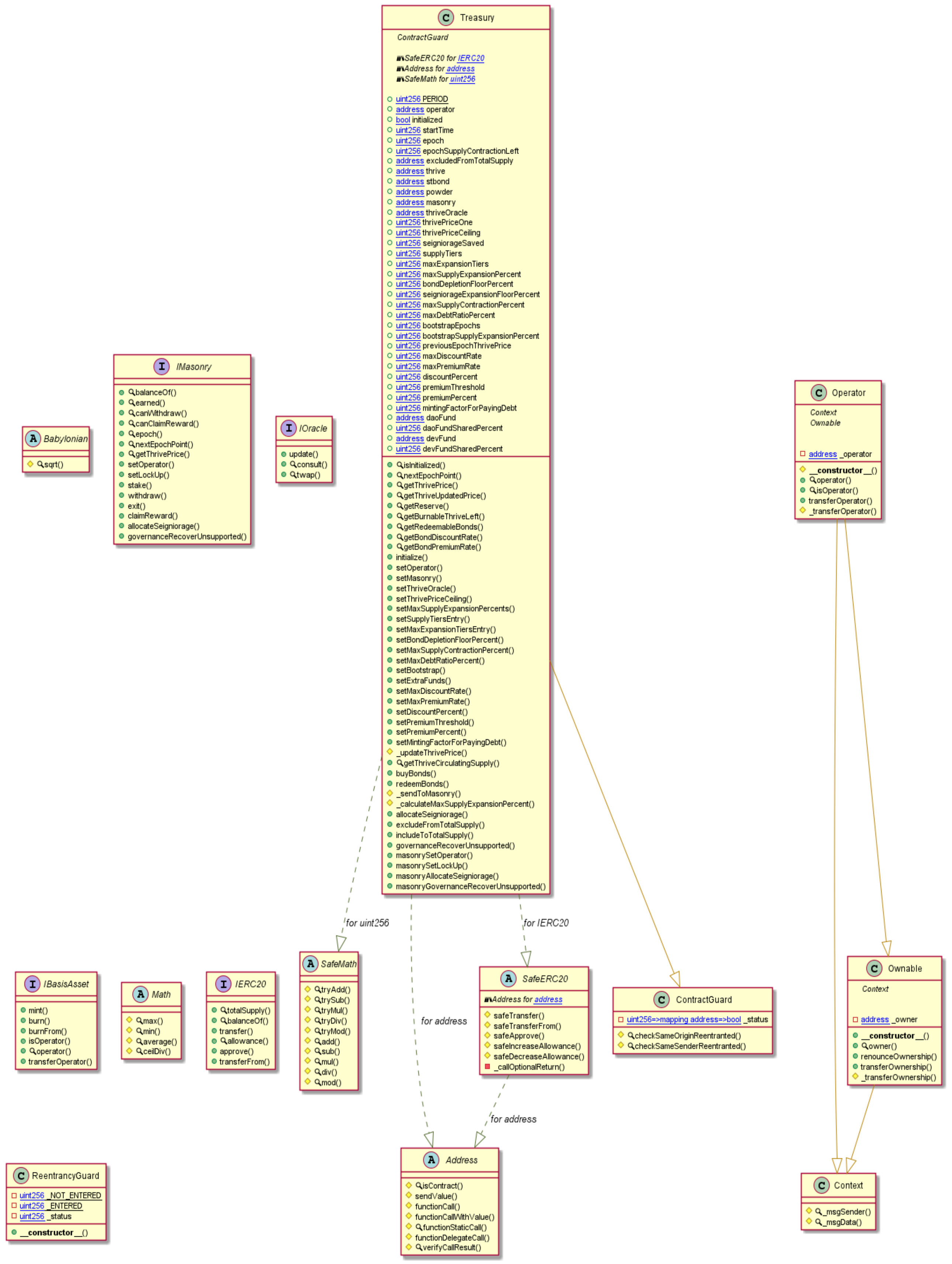
Thrive Diagram



This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

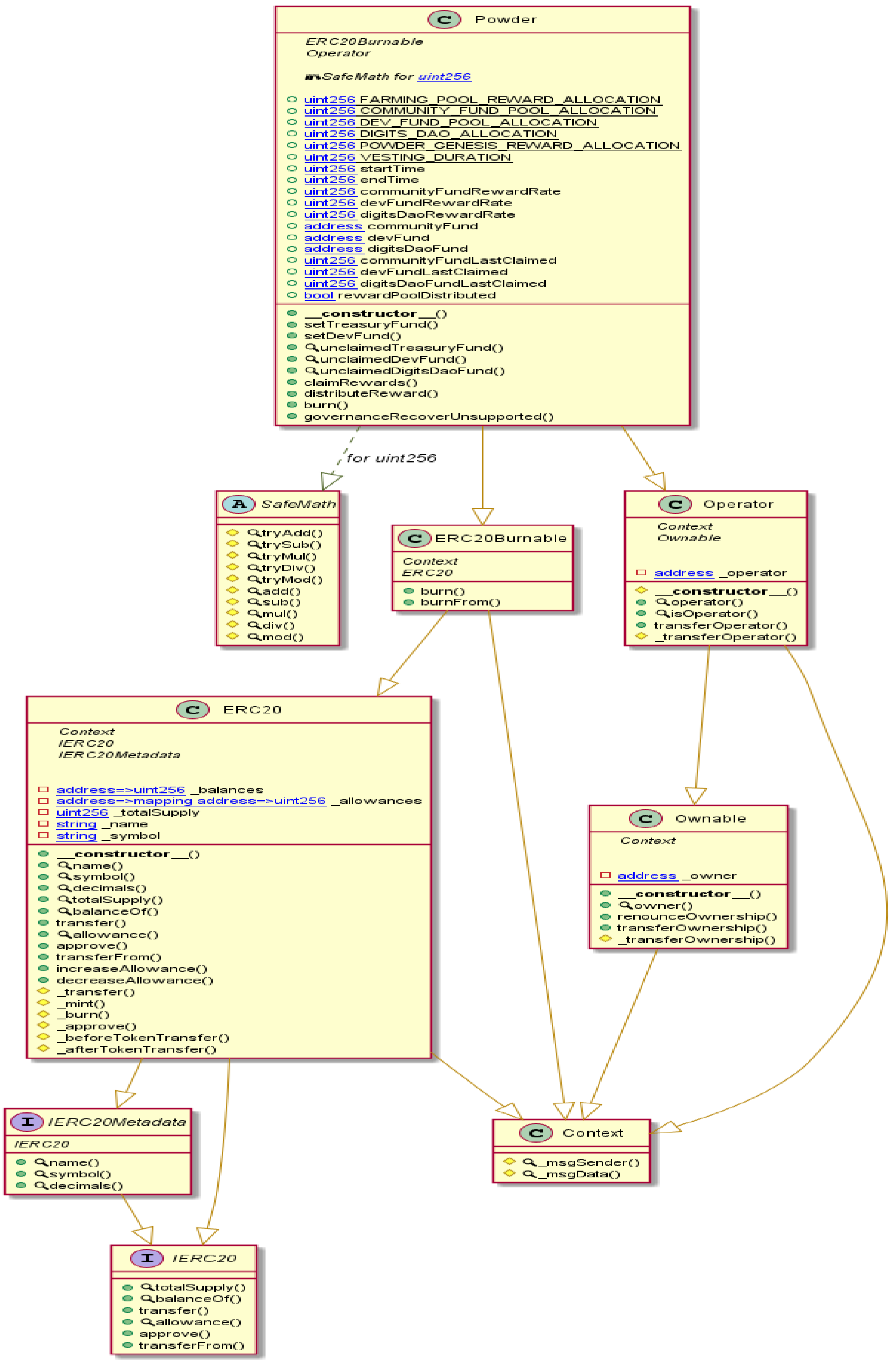
Email: audit@EtherAuthority.io

Treasury Diagram



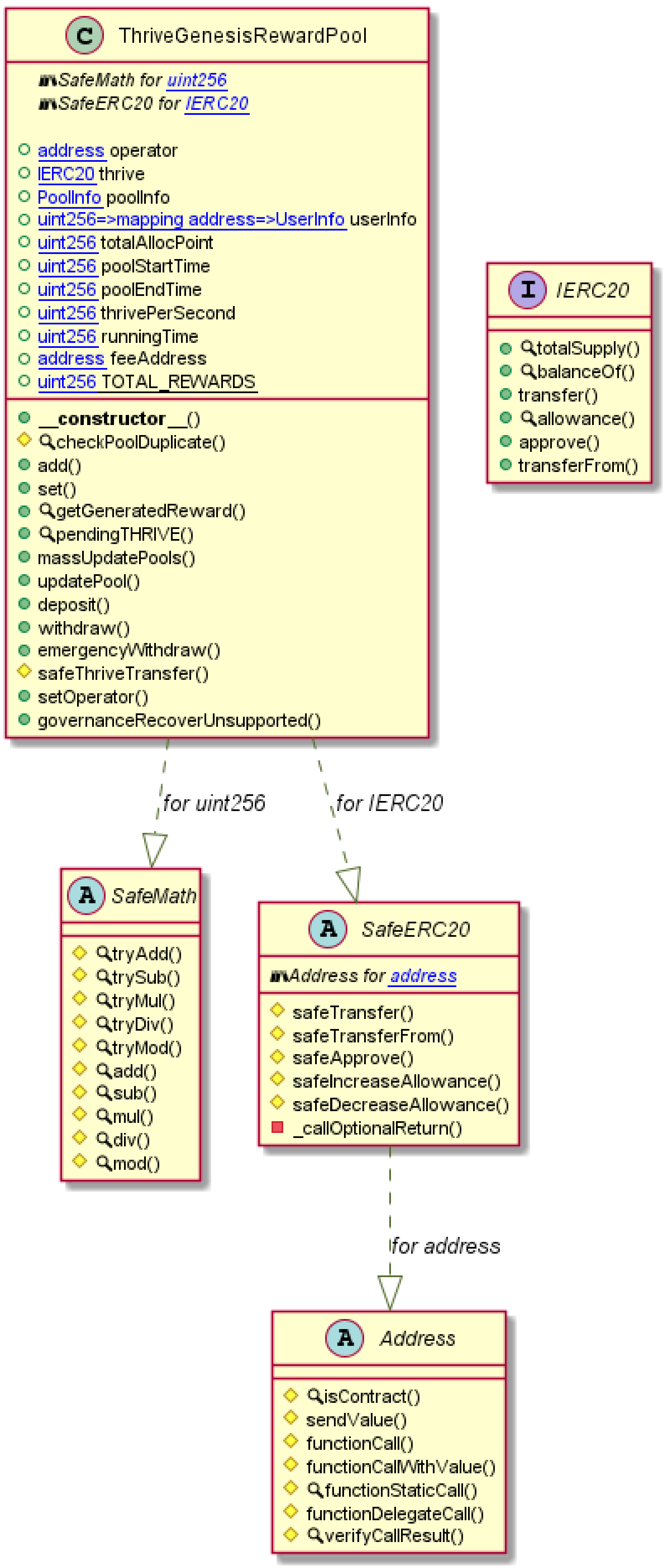
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Powder Diagram

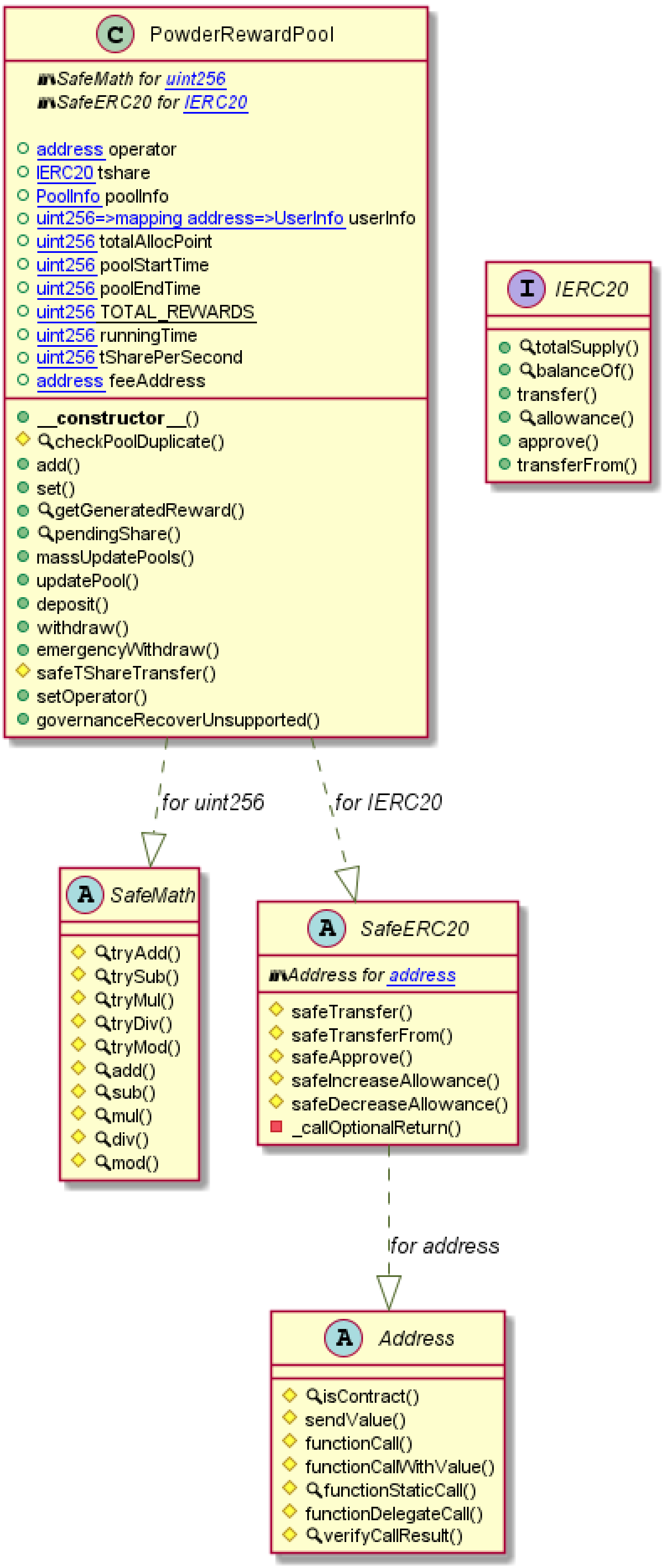


This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

ThriveGenesisRewardPool Diagram

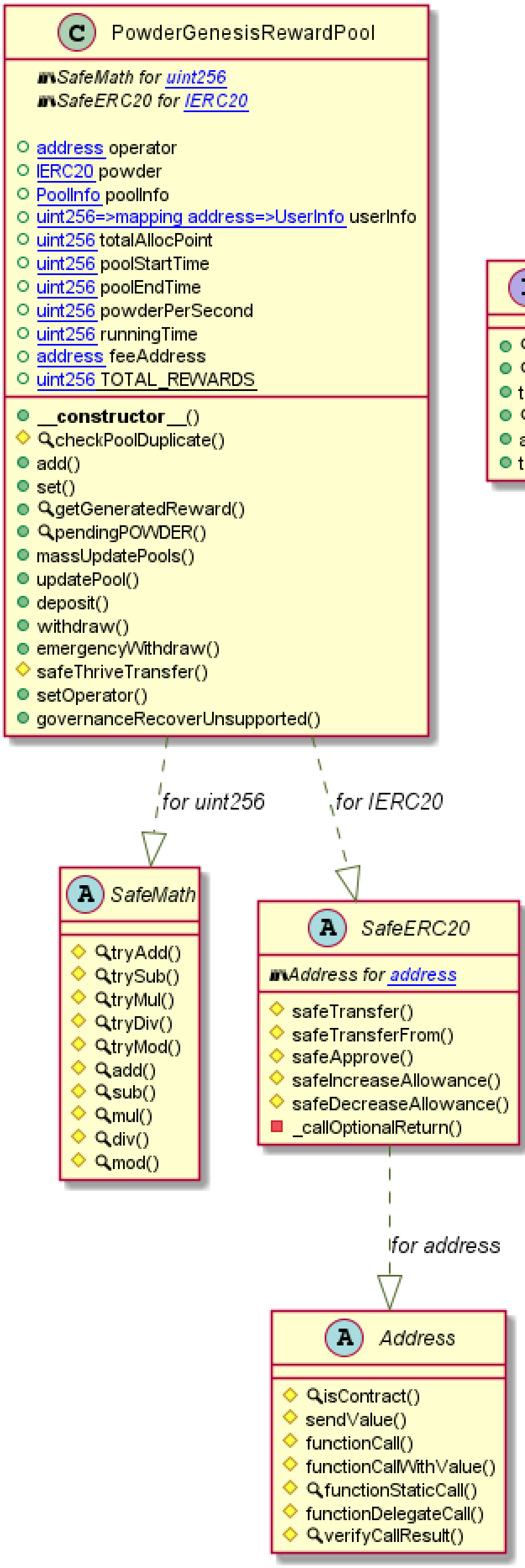


PowderRewardPool Diagram



This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

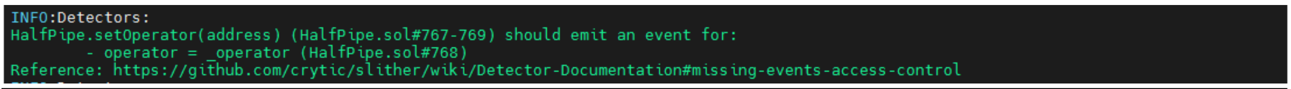
PowderGenesisRewardPool Diagram



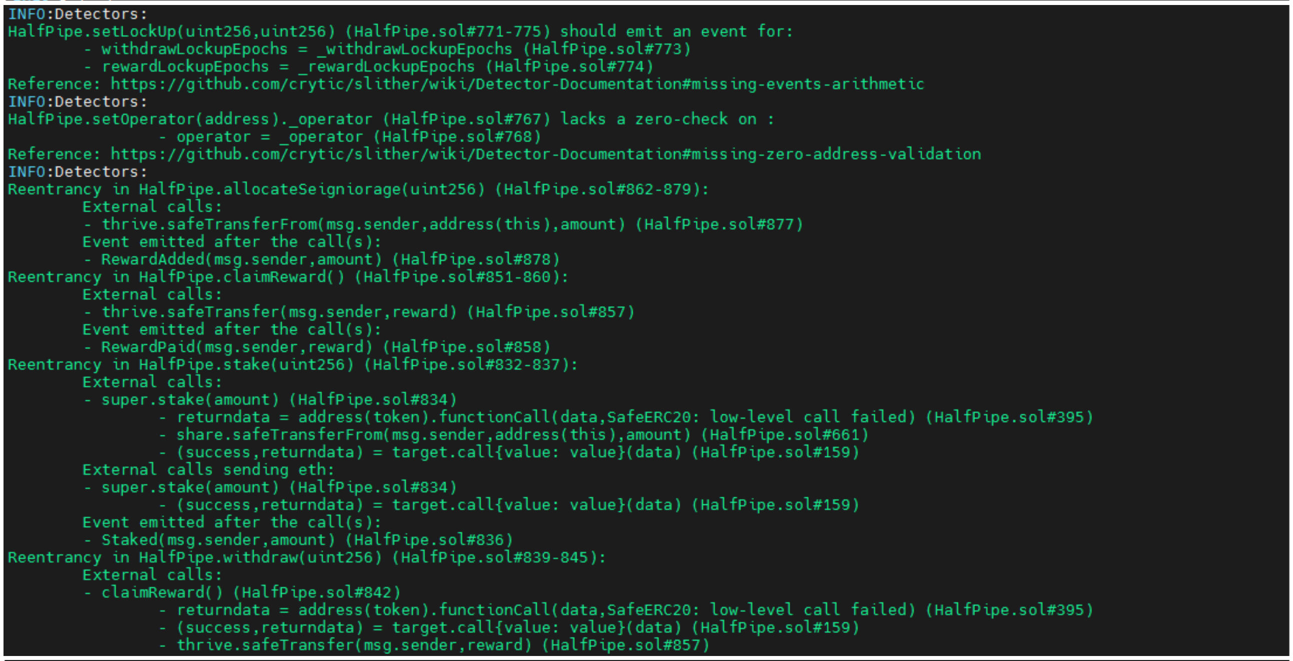
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Slither Results Log

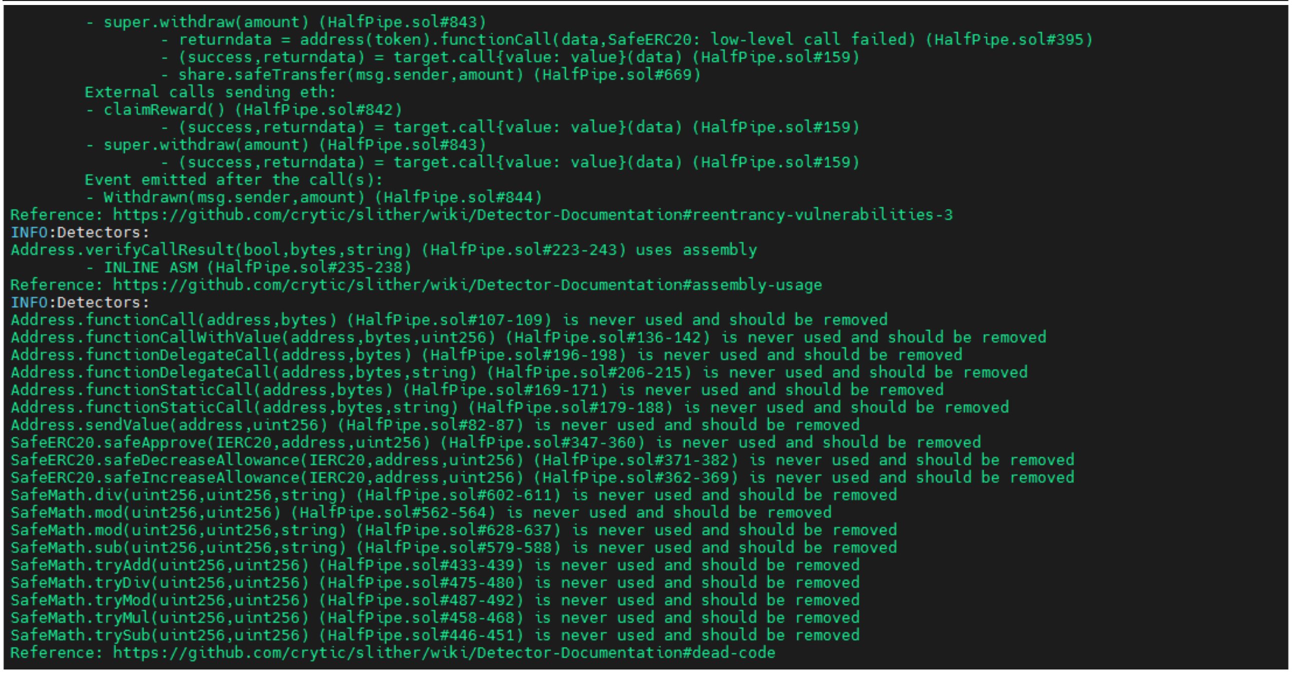
Slither log >> HalfPipe.sol



```
INFO:Detectors:
HalfPipe.setOperator(address) (HalfPipe.sol#767-769) should emit an event for:
        - operator = _operator (HalfPipe.sol#768)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
```

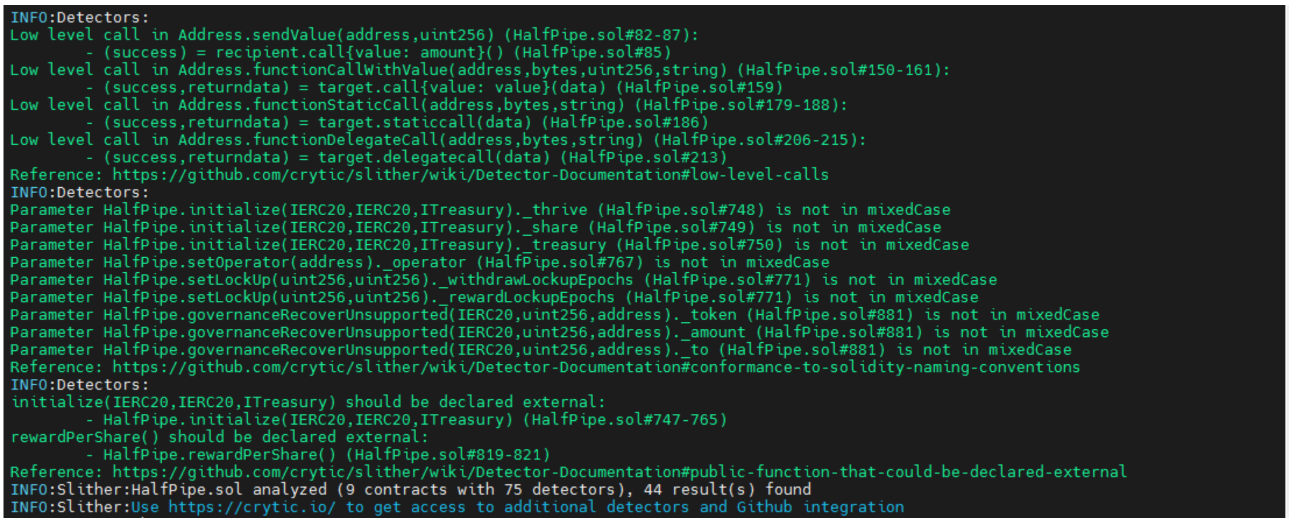


```
INFO:Detectors:
HalfPipe.setLockUp(uint256,uint256) (HalfPipe.sol#771-775) should emit an event for:
        - withdrawLockupEpochs = _withdrawLockupEpochs (HalfPipe.sol#773)
        - rewardLockupEpochs = _rewardLockupEpochs (HalfPipe.sol#774)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
HalfPipe.setOperator(address)._operator (HalfPipe.sol#767) lacks a zero-check on :
                - operator = _operator (HalfPipe.sol#768)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in HalfPipe.allocateSeigniorage(uint256) (HalfPipe.sol#862-879):
        External calls:
        - thrive.safeTransferFrom(msg.sender,address(this),amount) (HalfPipe.sol#877)
        Event emitted after the call(s):
        - RewardAdded(msg.sender,amount) (HalfPipe.sol#878)
Reentrancy in HalfPipe.claimReward() (HalfPipe.sol#851-860):
        External calls:
        - thrive.safeTransfer(msg.sender,reward) (HalfPipe.sol#857)
        Event emitted after the call(s):
        - RewardPaid(msg.sender,reward) (HalfPipe.sol#858)
Reentrancy in HalfPipe.stake(uint256) (HalfPipe.sol#832-837):
        External calls:
        - super.stake(amount) (HalfPipe.sol#834)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (HalfPipe.sol#395)
                - share.safeTransferFrom(msg.sender,address(this),amount) (HalfPipe.sol#661)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
        External calls sending eth:
        - super.stake(amount) (HalfPipe.sol#834)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
        Event emitted after the call(s):
        - Staked(msg.sender,amount) (HalfPipe.sol#836)
Reentrancy in HalfPipe.withdraw(uint256) (HalfPipe.sol#839-845):
        External calls:
        - claimReward() (HalfPipe.sol#842)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (HalfPipe.sol#395)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
                - thrive.safeTransfer(msg.sender,reward) (HalfPipe.sol#857)
```



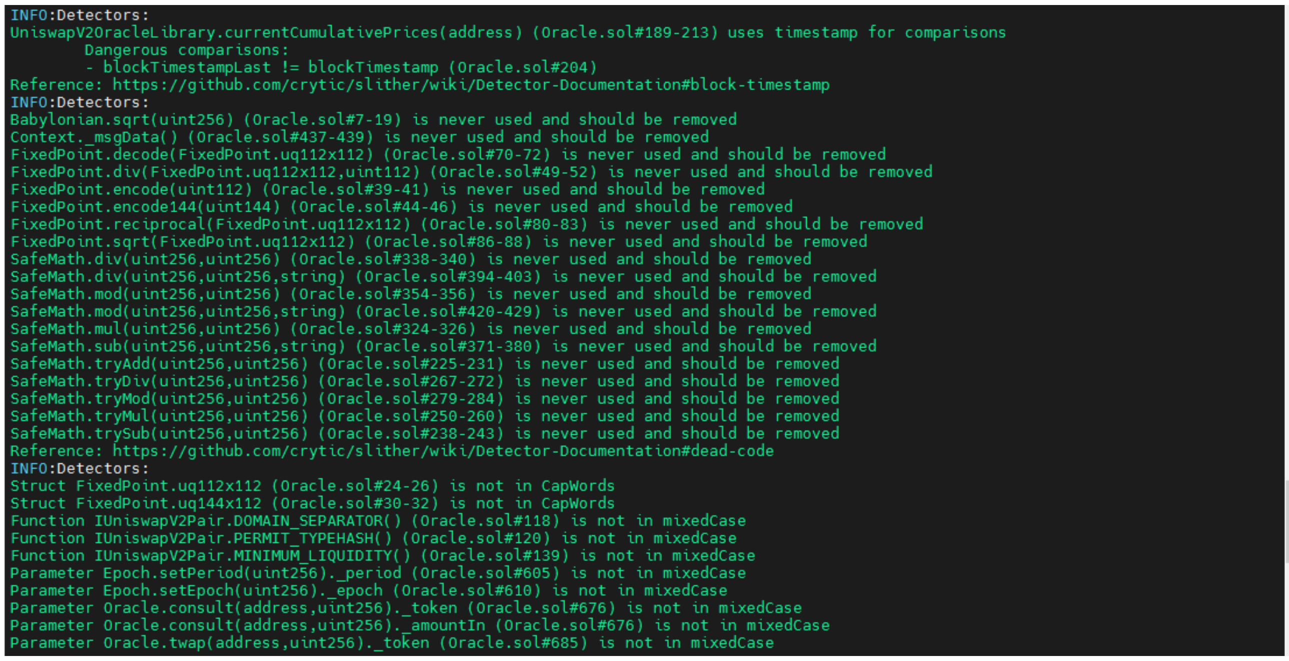
```
        - super.withdraw(amount) (HalfPipe.sol#843)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (HalfPipe.sol#395)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
                - share.safeTransfer(msg.sender,amount) (HalfPipe.sol#669)
        External calls sending eth:
        - claimReward() (HalfPipe.sol#842)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
        - super.withdraw(amount) (HalfPipe.sol#843)
                - (success,returndata) = target.call{value: value}(data) (HalfPipe.sol#159)
        Event emitted after the call(s):
        - Withdrawn(msg.sender,amount) (HalfPipe.sol#844)
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#347-360) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (HalfPipe.sol#371-382) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (HalfPipe.sol#362-369) is never used and should be removed
SafeMath.div(uint256,uint256,string) (HalfPipe.sol#602-611) is never used and should be removed
SafeMath.mod(uint256,uint256) (HalfPipe.sol#562-564) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (HalfPipe.sol#628-637) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (HalfPipe.sol#579-588) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (HalfPipe.sol#433-439) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (HalfPipe.sol#475-480) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (HalfPipe.sol#487-492) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (HalfPipe.sol#458-468) is never used and should be removed
SafeMath.trySub(uint256,uint256) (HalfPipe.sol#446-451) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

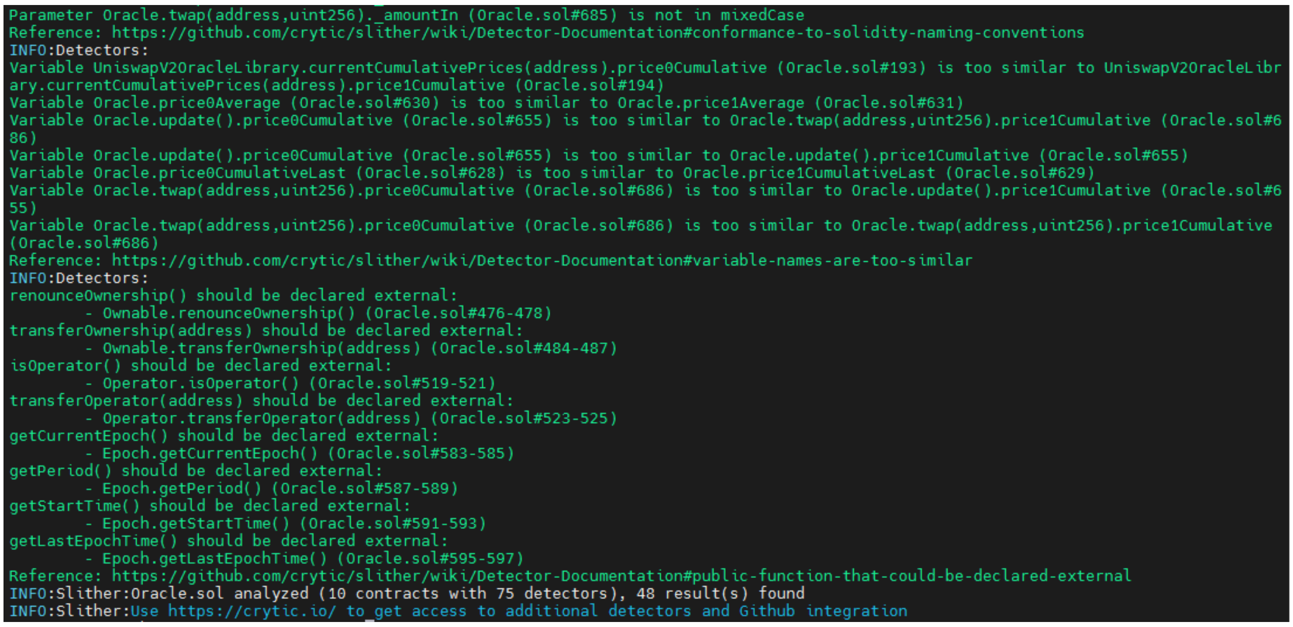


```
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (HalfPipe.sol#82-87):
        - (success) = recipient.call{value: amount}() (HalfPipe.sol#85)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (HalfPipe.sol#150-161):
        - (success,returndata) = 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)._thrive (HalfPipe.sol#748) is not in mixedCase
Parameter HalfPipe.initialize(IERC20,IERC20,ITreasury)._share (HalfPipe.sol#749) is not in mixedCase
Parameter HalfPipe.initialize(IERC20,IERC20,ITreasury)._treasury (HalfPipe.sol#750) is not in mixedCase
Parameter HalfPipe.setOperator(address)._operator (HalfPipe.sol#767) is not in mixedCase
Parameter HalfPipe.setLockUp(uint256,uint256)._withdrawLockupEpochs (HalfPipe.sol#771) is not in mixedCase
Parameter HalfPipe.setLockUp(uint256,uint256)._rewardLockupEpochs (HalfPipe.sol#771) is not in mixedCase
Parameter HalfPipe.governanceRecoverUnsupported(IERC20,uint256,address)._token (HalfPipe.sol#881) is not in mixedCase
Parameter HalfPipe.governanceRecoverUnsupported(IERC20,uint256,address)._amount (HalfPipe.sol#881) is not in mixedCase
Parameter HalfPipe.governanceRecoverUnsupported(IERC20,uint256,address)._to (HalfPipe.sol#881) 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#747-765)
rewardPerShare() should be declared external:
        - HalfPipe.rewardPerShare() (HalfPipe.sol#819-821)
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), 44 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Oracle.sol



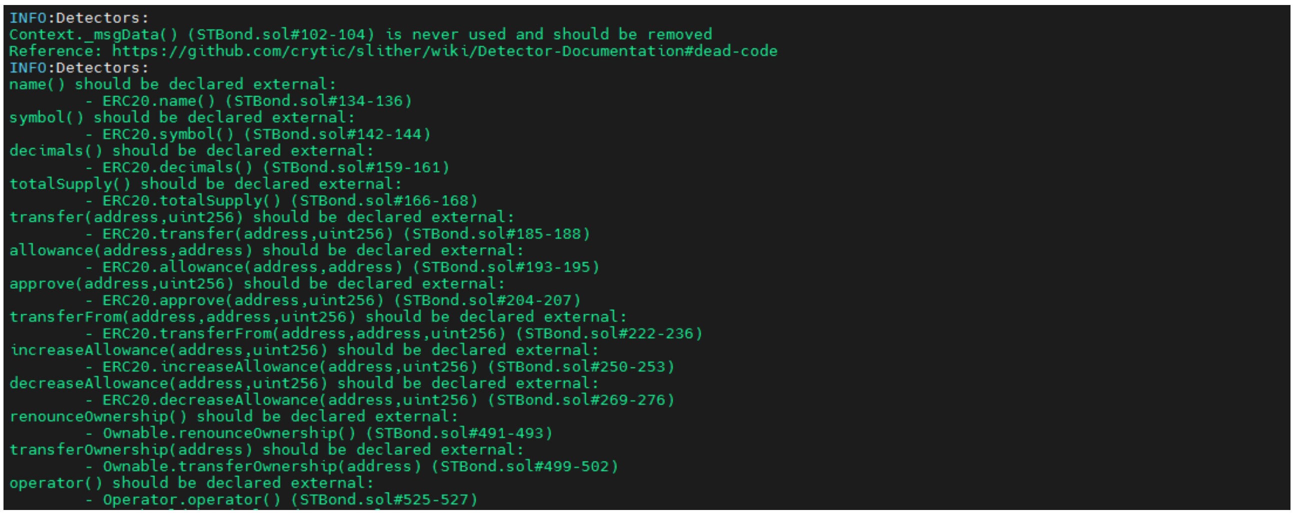
```
INFO:Detectors:
UniswapV2OracleLibrary.currentCumulativePrices(address) (Oracle.sol#189-213) uses timestamp for comparisons
        Dangerous comparisons:
        - blockTimestampLast != blockTimestamp (Oracle.sol#204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Babylonian.sqrt(uint256) (Oracle.sol#7-19) is never used and should be removed
Context._msgData() (Oracle.sol#437-439) 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#338-340) is never used and should be removed
SafeMath.div(uint256,uint256,string) (Oracle.sol#394-403) is never used and should be removed
SafeMath.mod(uint256,uint256) (Oracle.sol#354-356) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Oracle.sol#420-429) is never used and should be removed
SafeMath.mul(uint256,uint256) (Oracle.sol#324-326) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (Oracle.sol#371-380) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (Oracle.sol#225-231) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (Oracle.sol#267-272) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (Oracle.sol#279-284) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (Oracle.sol#250-260) is never used and should be removed
SafeMath.trySub(uint256,uint256) (Oracle.sol#238-243) 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#118) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (Oracle.sol#120) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (Oracle.sol#139) is not in mixedCase
Parameter Epoch.setPeriod(uint256)._period (Oracle.sol#605) is not in mixedCase
Parameter Epoch.setEpoch(uint256)._epoch (Oracle.sol#610) is not in mixedCase
Parameter Oracle.consult(address,uint256)._token (Oracle.sol#676) is not in mixedCase
Parameter Oracle.consult(address,uint256)._amountIn (Oracle.sol#676) is not in mixedCase
Parameter Oracle.twap(address,uint256)._token (Oracle.sol#685) is not in mixedCase
```



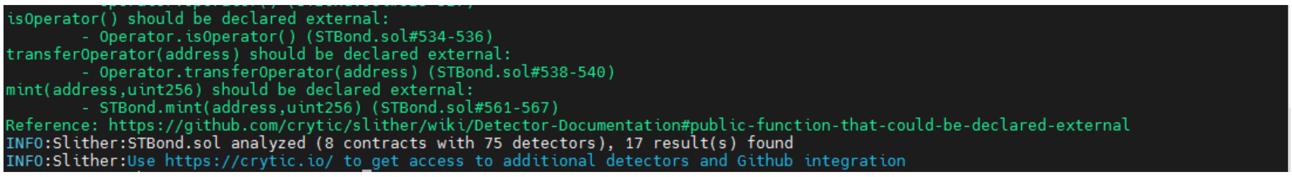
```
Parameter Oracle.twap(address,uint256)._amountIn (Oracle.sol#685) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable UniswapV2OracleLibrary.currentCumulativePrices(address).price0Cumulative (Oracle.sol#193) is too similar to UniswapV2OracleLibr
ary.currentCumulativePrices(address).price1Cumulative (Oracle.sol#194)
Variable Oracle.price0Average (Oracle.sol#630) is too similar to Oracle.price1Average (Oracle.sol#631)
Variable Oracle.update().price0Cumulative (Oracle.sol#655) is too similar to Oracle.twap(address,uint256).price1Cumulative (Oracle.sol#6
86)
Variable Oracle.update().price0Cumulative (Oracle.sol#655) is too similar to Oracle.update().price1Cumulative (Oracle.sol#655)
Variable Oracle.price0CumulativeLast (Oracle.sol#628) is too similar to Oracle.price1CumulativeLast (Oracle.sol#629)
Variable Oracle.twap(address,uint256).price0Cumulative (Oracle.sol#686) is too similar to Oracle.update().price1Cumulative (Oracle.sol#6
55)
Variable Oracle.twap(address,uint256).price0Cumulative (Oracle.sol#686) is too similar to Oracle.twap(address,uint256).price1Cumulative
(Oracle.sol#686)
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#476-478)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Oracle.sol#484-487)
isOperator() should be declared external:
        - Operator.isOperator() (Oracle.sol#519-521)
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (Oracle.sol#523-525)
getCurrentEpoch() should be declared external:
        - Epoch.getCurrentEpoch() (Oracle.sol#583-585)
getPeriod() should be declared external:
        - Epoch.getPeriod() (Oracle.sol#587-589)
getStartTime() should be declared external:
        - Epoch.getStartTime() (Oracle.sol#591-593)
getLastEpochTime() should be declared external:
        - Epoch.getLastEpochTime() (Oracle.sol#595-597)
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
```

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Slither log >> STBond.sol

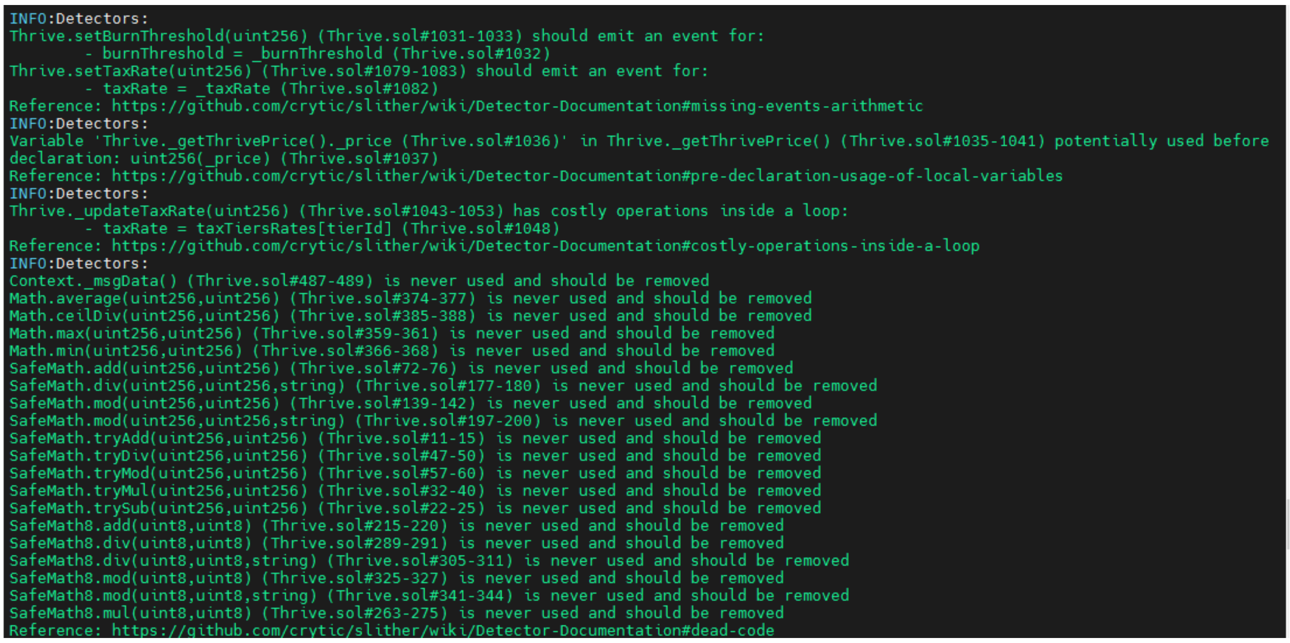


```
INFO:Detectors:
Context._msgData() (STBond.sol#102-104) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (STBond.sol#134-136)
symbol() should be declared external:
        - ERC20.symbol() (STBond.sol#142-144)
decimals() should be declared external:
        - ERC20.decimals() (STBond.sol#159-161)
totalSupply() should be declared external:
        - ERC20.totalSupply() (STBond.sol#166-168)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (STBond.sol#185-188)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (STBond.sol#193-195)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (STBond.sol#204-207)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (STBond.sol#222-236)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (STBond.sol#250-253)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (STBond.sol#269-276)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (STBond.sol#491-493)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (STBond.sol#499-502)
operator() should be declared external:
        - Operator.operator() (STBond.sol#525-527)
```



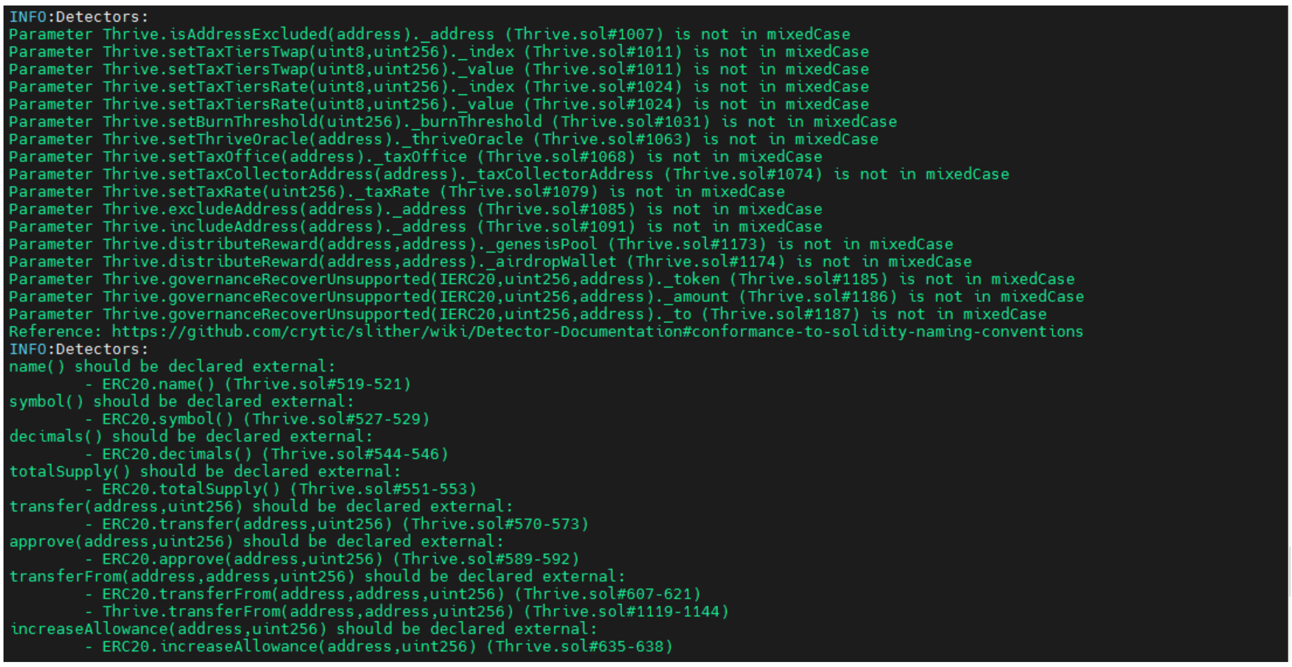
```
isOperator() should be declared external:
        - Operator.isOperator() (STBond.sol#534-536)
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (STBond.sol#538-540)
mint(address,uint256) should be declared external:
        - STBond.mint(address,uint256) (STBond.sol#561-567)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:STBond.sol analyzed (8 contracts with 75 detectors), 17 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Thrive.sol

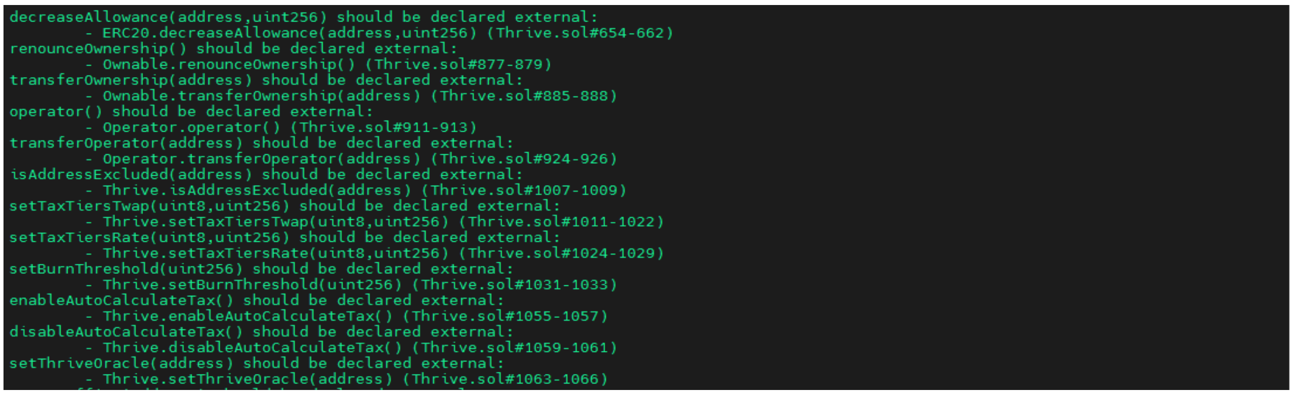


```
INFO:Detectors:
Thrive.setBurnThreshold(uint256) (Thrive.sol#1031-1033) should emit an event for:
        - burnThreshold = _burnThreshold (Thrive.sol#1032)
Thrive.setTaxRate(uint256) (Thrive.sol#1079-1083) should emit an event for:
        - taxRate = _taxRate (Thrive.sol#1082)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Variable 'Thrive._getThrivePrice()._price (Thrive.sol#1036)' in Thrive._getThrivePrice() (Thrive.sol#1035-1041) potentially used before
declaration: uint256(_price) (Thrive.sol#1037)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables
INFO:Detectors:
Thrive._updateTaxRate(uint256) (Thrive.sol#1043-1053) has costly operations inside a loop:
        - taxRate = taxTiersRates[tierId] (Thrive.sol#1048)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop
INFO:Detectors:
Context._msgData() (Thrive.sol#487-489) is never used and should be removed
Math.average(uint256,uint256) (Thrive.sol#374-377) is never used and should be removed
Math.ceilDiv(uint256,uint256) (Thrive.sol#385-388) is never used and should be removed
Math.max(uint256,uint256) (Thrive.sol#359-361) is never used and should be removed
Math.min(uint256,uint256) (Thrive.sol#366-368) is never used and should be removed
SafeMath.add(uint256,uint256) (Thrive.sol#72-76) is never used and should be removed
SafeMath.div(uint256,uint256,string) (Thrive.sol#177-180) is never used and should be removed
SafeMath.mod(uint256,uint256) (Thrive.sol#139-142) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Thrive.sol#197-200) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (Thrive.sol#11-15) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (Thrive.sol#47-50) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (Thrive.sol#57-60) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (Thrive.sol#32-40) is never used and should be removed
SafeMath.trySub(uint256,uint256) (Thrive.sol#22-25) is never used and should be removed
SafeMath8.add(uint8,uint8) (Thrive.sol#215-220) is never used and should be removed
SafeMath8.div(uint8,uint8) (Thrive.sol#289-291) is never used and should be removed
SafeMath8.div(uint8,uint8,string) (Thrive.sol#305-311) is never used and should be removed
SafeMath8.mod(uint8,uint8) (Thrive.sol#325-327) is never used and should be removed
SafeMath8.mod(uint8,uint8,string) (Thrive.sol#341-344) is never used and should be removed
SafeMath8.mul(uint8,uint8) (Thrive.sol#263-275) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

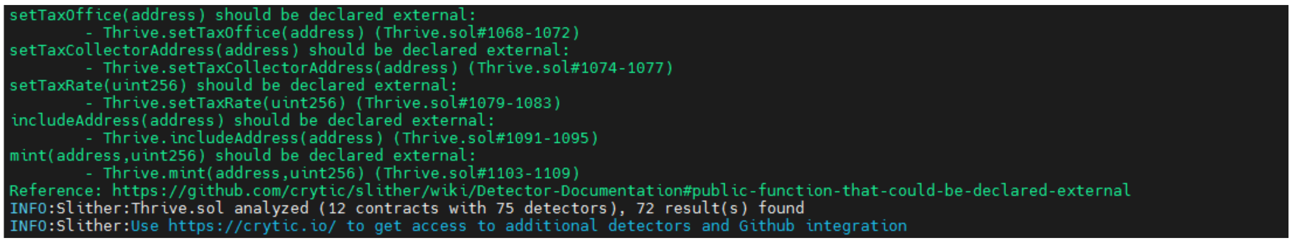
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io



```
INFO:Detectors:
Parameter Thrive.isAddressExcluded(address)._address (Thrive.sol#1007) is not in mixedCase
Parameter Thrive.setTaxTiersTwap(uint8,uint256)._index (Thrive.sol#1011) is not in mixedCase
Parameter Thrive.setTaxTiersTwap(uint8,uint256)._value (Thrive.sol#1011) is not in mixedCase
Parameter Thrive.setTaxTiersRate(uint8,uint256)._index (Thrive.sol#1024) is not in mixedCase
Parameter Thrive.setTaxTiersRate(uint8,uint256)._value (Thrive.sol#1024) is not in mixedCase
Parameter Thrive.setBurnThreshold(uint256)._burnThreshold (Thrive.sol#1031) is not in mixedCase
Parameter Thrive.setThriveOracle(address)._thriveOracle (Thrive.sol#1063) is not in mixedCase
Parameter Thrive.setTaxOffice(address)._taxOffice (Thrive.sol#1068) is not in mixedCase
Parameter Thrive.setTaxCollectorAddress(address)._taxCollectorAddress (Thrive.sol#1074) is not in mixedCase
Parameter Thrive.setTaxRate(uint256)._taxRate (Thrive.sol#1079) is not in mixedCase
Parameter Thrive.excludeAddress(address)._address (Thrive.sol#1085) is not in mixedCase
Parameter Thrive.includeAddress(address)._address (Thrive.sol#1091) is not in mixedCase
Parameter Thrive.distributeReward(address,address)._genesisPool (Thrive.sol#1173) is not in mixedCase
Parameter Thrive.distributeReward(address,address)._airdropWallet (Thrive.sol#1174) is not in mixedCase
Parameter Thrive.governanceRecoverUnsupported(IERC20,uint256,address)._token (Thrive.sol#1185) is not in mixedCase
Parameter Thrive.governanceRecoverUnsupported(IERC20,uint256,address)._amount (Thrive.sol#1186) is not in mixedCase
Parameter Thrive.governanceRecoverUnsupported(IERC20,uint256,address)._to (Thrive.sol#1187) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (Thrive.sol#519-521)
symbol() should be declared external:
        - ERC20.symbol() (Thrive.sol#527-529)
decimals() should be declared external:
        - ERC20.decimals() (Thrive.sol#544-546)
totalSupply() should be declared external:
        - ERC20.totalSupply() (Thrive.sol#551-553)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (Thrive.sol#570-573)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (Thrive.sol#589-592)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (Thrive.sol#607-621)
        - Thrive.transferFrom(address,address,uint256) (Thrive.sol#1119-1144)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (Thrive.sol#635-638)
```

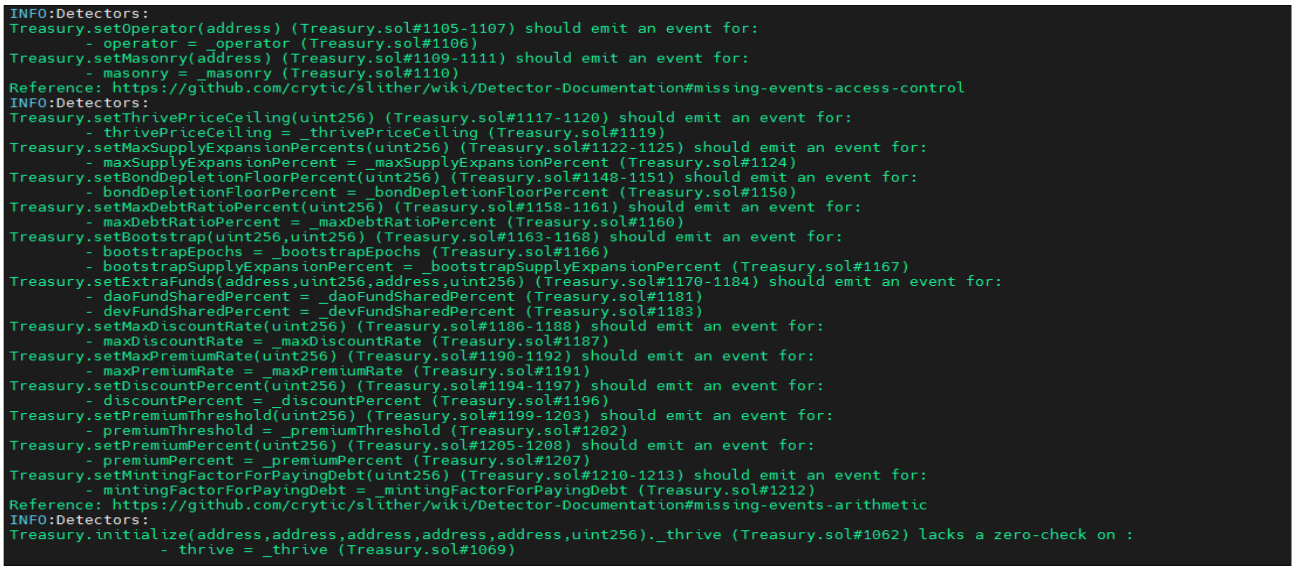


```
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (Thrive.sol#654-662)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Thrive.sol#877-879)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Thrive.sol#885-888)
operator() should be declared external:
        - Operator.operator() (Thrive.sol#911-913)
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (Thrive.sol#924-926)
isAddressExcluded(address) should be declared external:
        - Thrive.isAddressExcluded(address) (Thrive.sol#1007-1009)
setTaxTiersTwap(uint8,uint256) should be declared external:
        - Thrive.setTaxTiersTwap(uint8,uint256) (Thrive.sol#1011-1022)
setTaxTiersRate(uint8,uint256) should be declared external:
        - Thrive.setTaxTiersRate(uint8,uint256) (Thrive.sol#1024-1029)
setBurnThreshold(uint256) should be declared external:
        - Thrive.setBurnThreshold(uint256) (Thrive.sol#1031-1033)
enableAutoCalculateTax() should be declared external:
        - Thrive.enableAutoCalculateTax() (Thrive.sol#1055-1057)
disableAutoCalculateTax() should be declared external:
        - Thrive.disableAutoCalculateTax() (Thrive.sol#1059-1061)
setThriveOracle(address) should be declared external:
        - Thrive.setThriveOracle(address) (Thrive.sol#1063-1066)
```

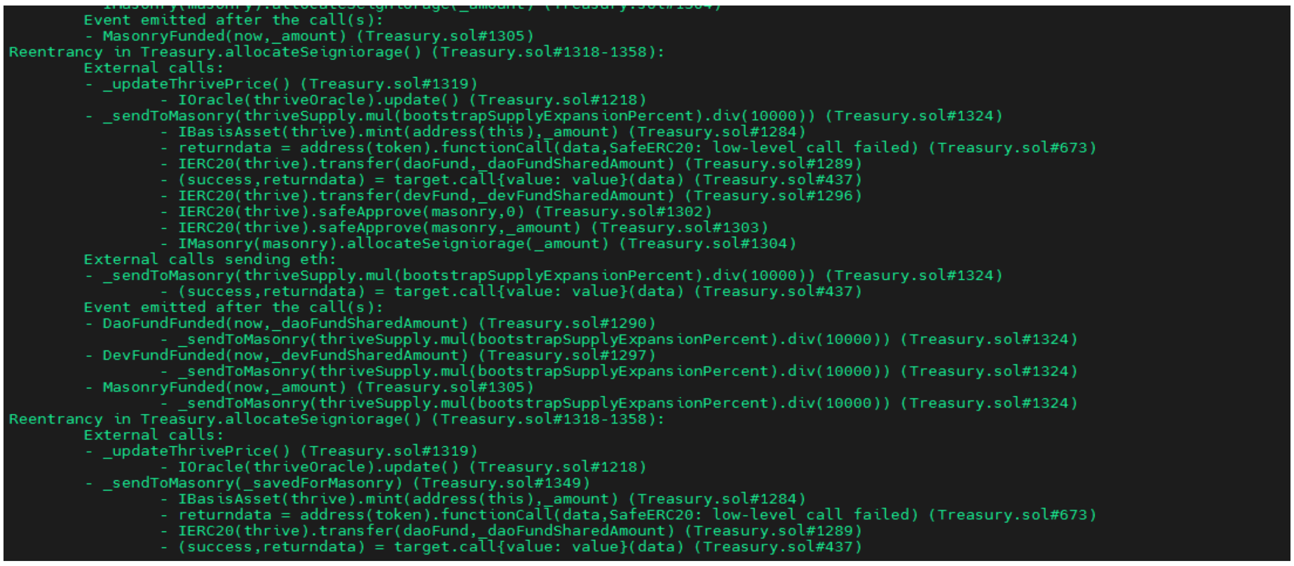


```
setTaxOffice(address) should be declared external:
        - Thrive.setTaxOffice(address) (Thrive.sol#1068-1072)
setTaxCollectorAddress(address) should be declared external:
        - Thrive.setTaxCollectorAddress(address) (Thrive.sol#1074-1077)
setTaxRate(uint256) should be declared external:
        - Thrive.setTaxRate(uint256) (Thrive.sol#1079-1083)
includeAddress(address) should be declared external:
        - Thrive.includeAddress(address) (Thrive.sol#1091-1095)
mint(address,uint256) should be declared external:
        - Thrive.mint(address,uint256) (Thrive.sol#1103-1109)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Thrive.sol analyzed (12 contracts with 75 detectors), 72 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

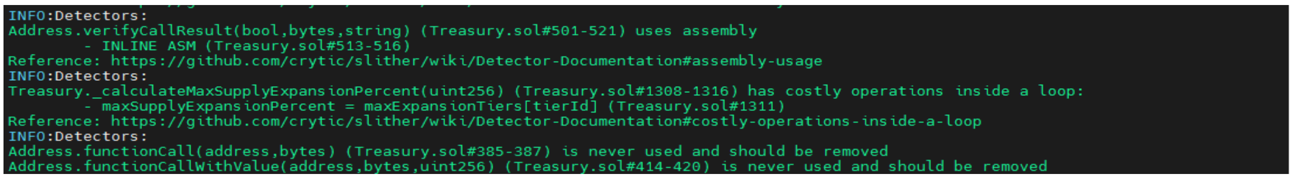
Slither log >> Treasury.sol



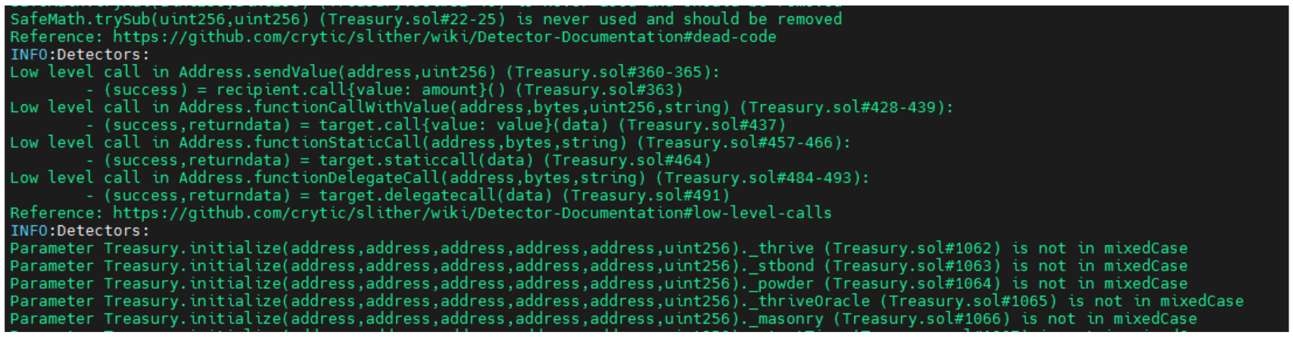
```
INFO:Detectors:
Treasury.setOperator(address) (Treasury.sol#1105-1107) should emit an event for:
        - operator = _operator (Treasury.sol#1106)
Treasury.setMasonry(address) (Treasury.sol#1109-1111) should emit an event for:
        - masonry = _masonry (Treasury.sol#1110)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
Treasury.setThrivePriceCeiling(uint256) (Treasury.sol#1117-1120) should emit an event for:
        - thrivePriceCeiling = _thrivePriceCeiling (Treasury.sol#1119)
Treasury.setMaxSupplyExpansionPercents(uint256) (Treasury.sol#1122-1125) should emit an event for:
        - maxSupplyExpansionPercent = _maxSupplyExpansionPercent (Treasury.sol#1124)
Treasury.setBondDepletionFloorPercent(uint256) (Treasury.sol#1148-1151) should emit an event for:
        - bondDepletionFloorPercent = _bondDepletionFloorPercent (Treasury.sol#1150)
Treasury.setMaxDebtRatioPercent(uint256) (Treasury.sol#1158-1161) should emit an event for:
        - maxDebtRatioPercent = _maxDebtRatioPercent (Treasury.sol#1160)
Treasury.setBootstrap(uint256,uint256) (Treasury.sol#1163-1168) should emit an event for:
        - bootstrapEpochs = _bootstrapEpochs (Treasury.sol#1166)
        - bootstrapSupplyExpansionPercent = _bootstrapSupplyExpansionPercent (Treasury.sol#1167)
Treasury.setExtraFunds(address,uint256,address,uint256) (Treasury.sol#1170-1184) should emit an event for:
        - daoFundSharedPercent = _daoFundSharedPercent (Treasury.sol#1181)
        - devFundSharedPercent = _devFundSharedPercent (Treasury.sol#1183)
Treasury.setMaxDiscountRate(uint256) (Treasury.sol#1186-1188) should emit an event for:
        - maxDiscountRate = _maxDiscountRate (Treasury.sol#1187)
Treasury.setMaxPremiumRate(uint256) (Treasury.sol#1190-1192) should emit an event for:
        - maxPremiumRate = _maxPremiumRate (Treasury.sol#1191)
Treasury.setDiscountPercent(uint256) (Treasury.sol#1194-1197) should emit an event for:
        - discountPercent = _discountPercent (Treasury.sol#1196)
Treasury.setPremiumThreshold(uint256) (Treasury.sol#1199-1203) should emit an event for:
        - premiumThreshold = _premiumThreshold (Treasury.sol#1202)
Treasury.setPremiumPercent(uint256) (Treasury.sol#1205-1208) should emit an event for:
        - premiumPercent = _premiumPercent (Treasury.sol#1207)
Treasury.setMintingFactorForPayingDebt(uint256) (Treasury.sol#1210-1213) should emit an event for:
        - mintingFactorForPayingDebt = _mintingFactorForPayingDebt (Treasury.sol#1212)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Treasury.initialize(address,address,address,address,address,uint256)._thrive (Treasury.sol#1062) lacks a zero-check on :
                - thrive = _thrive (Treasury.sol#1069)
```



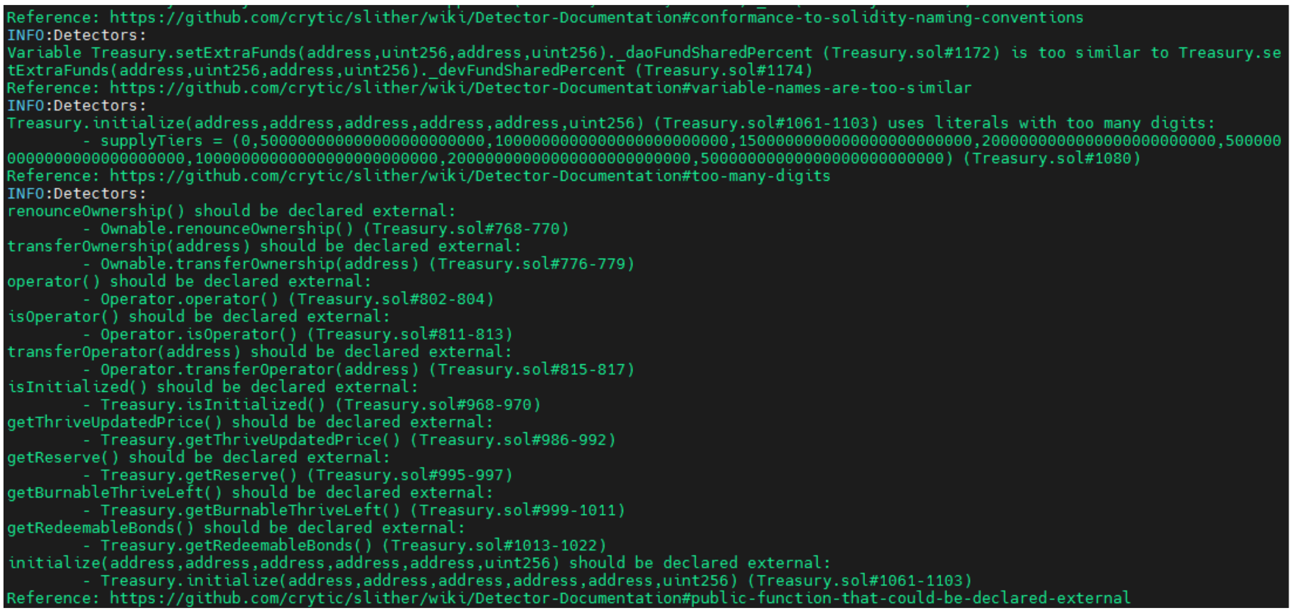
```
        Event emitted after the call(s):
        - MasonryFunded(now,_amount) (Treasury.sol#1305)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1318-1358):
        External calls:
        - _updateThrivePrice() (Treasury.sol#1319)
                - IOracle(thriveOracle).update() (Treasury.sol#1218)
        - _sendToMasonry(thriveSupply.mul(bootstrapSupplyExpansionPercent).div(10000)) (Treasury.sol#1324)
                - IBasisAsset(thrive).mint(address(this),_amount) (Treasury.sol#1284)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Treasury.sol#673)
                - IERC20(thrive).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1289)
                - (success,returndata) = target.call{value: value}(data) (Treasury.sol#437)
                - IERC20(thrive).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1296)
                - IERC20(thrive).safeApprove(masonry,0) (Treasury.sol#1302)
                - IERC20(thrive).safeApprove(masonry,_amount) (Treasury.sol#1303)
                - IMasonry(masonry).allocateSeigniorage(_amount) (Treasury.sol#1304)
        External calls sending eth:
        - _sendToMasonry(thriveSupply.mul(bootstrapSupplyExpansionPercent).div(10000)) (Treasury.sol#1324)
                - (success,returndata) = target.call{value: value}(data) (Treasury.sol#437)
        Event emitted after the call(s):
        - DaoFundFunded(now,_daoFundSharedAmount) (Treasury.sol#1290)
                - _sendToMasonry(thriveSupply.mul(bootstrapSupplyExpansionPercent).div(10000)) (Treasury.sol#1324)
        - DevFundFunded(now,_devFundSharedAmount) (Treasury.sol#1297)
                - _sendToMasonry(thriveSupply.mul(bootstrapSupplyExpansionPercent).div(10000)) (Treasury.sol#1324)
        - MasonryFunded(now,_amount) (Treasury.sol#1305)
                - _sendToMasonry(thriveSupply.mul(bootstrapSupplyExpansionPercent).div(10000)) (Treasury.sol#1324)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1318-1358):
        External calls:
        - _updateThrivePrice() (Treasury.sol#1319)
                - IOracle(thriveOracle).update() (Treasury.sol#1218)
        - _sendToMasonry(_savedForMasonry) (Treasury.sol#1349)
                - IBasisAsset(thrive).mint(address(this),_amount) (Treasury.sol#1284)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Treasury.sol#673)
                - IERC20(thrive).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1289)
                - (success,returndata) = target.call{value: value}(data) (Treasury.sol#437)
```



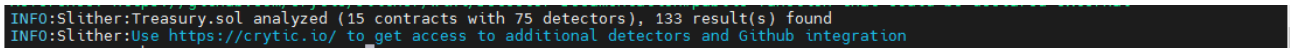
```
INFO:Detectors:
Address.verifyCallResult(bool,bytes,string) (Treasury.sol#501-521) uses assembly
        - INLINE ASM (Treasury.sol#513-516)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Treasury._calculateMaxSupplyExpansionPercent(uint256) (Treasury.sol#1308-1316) has costly operations inside a loop:
        - maxSupplyExpansionPercent = maxExpansionTiers[tierId] (Treasury.sol#1311)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop
INFO:Detectors:
Address.functionCall(address,bytes) (Treasury.sol#385-387) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (Treasury.sol#414-420) is never used and should be removed
```



```
SafeMath.trySub(uint256,uint256) (Treasury.sol#22-25) 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#360-365):
        - (success) = recipient.call{value: amount}() (Treasury.sol#363)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (Treasury.sol#428-439):
        - (success,returndata) = target.call{value: value}(data) (Treasury.sol#437)
Low level call in Address.functionStaticCall(address,bytes,string) (Treasury.sol#457-466):
        - (success,returndata) = target.staticcall(data) (Treasury.sol#464)
Low level call in Address.functionDelegateCall(address,bytes,string) (Treasury.sol#484-493):
        - (success,returndata) = target.delegatecall(data) (Treasury.sol#491)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter Treasury.initialize(address,address,address,address,address,uint256)._thrive (Treasury.sol#1062) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._stbond (Treasury.sol#1063) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._powder (Treasury.sol#1064) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._thriveOracle (Treasury.sol#1065) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._masonry (Treasury.sol#1066) is not in mixedCase
```



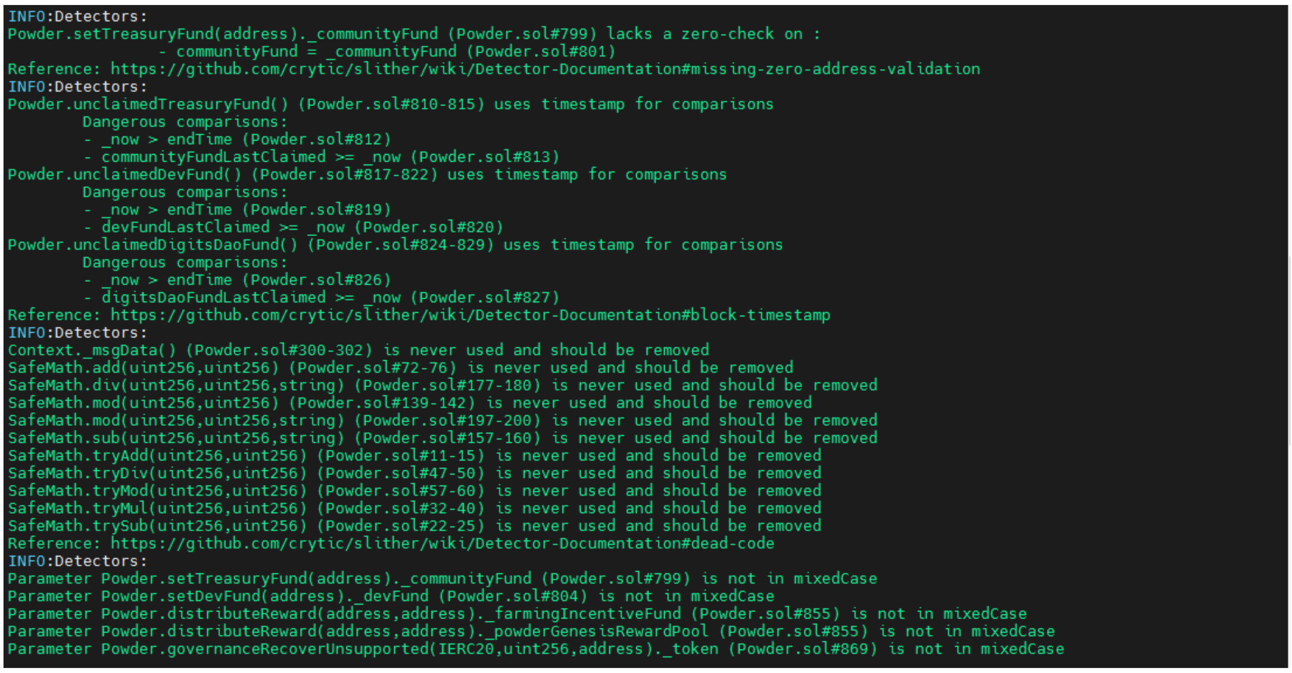
```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable Treasury.setExtraFunds(address,uint256,address,uint256)._daoFundSharedPercent (Treasury.sol#1172) is too similar to Treasury.se
tExtraFunds(address,uint256,address,uint256)._devFundSharedPercent (Treasury.sol#1174)
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#1061-1103) uses literals with too many digits:
        - supplyTiers = (0,500000000000000000000000,1000000000000000000000000,1500000000000000000000000,2000000000000000000000000,500000
0000000000000000000,10000000000000000000000000,20000000000000000000000000,50000000000000000000000000) (Treasury.sol#1080)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Treasury.sol#768-770)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Treasury.sol#776-779)
operator() should be declared external:
        - Operator.operator() (Treasury.sol#802-804)
isOperator() should be declared external:
        - Operator.isOperator() (Treasury.sol#811-813)
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (Treasury.sol#815-817)
isInitialized() should be declared external:
        - Treasury.isInitialized() (Treasury.sol#968-970)
getThriveUpdatedPrice() should be declared external:
        - Treasury.getThriveUpdatedPrice() (Treasury.sol#986-992)
getReserve() should be declared external:
        - Treasury.getReserve() (Treasury.sol#995-997)
getBurnableThriveLeft() should be declared external:
        - Treasury.getBurnableThriveLeft() (Treasury.sol#999-1011)
getRedeemableBonds() should be declared external:
        - Treasury.getRedeemableBonds() (Treasury.sol#1013-1022)
initialize(address,address,address,address,address,uint256) should be declared external:
        - Treasury.initialize(address,address,address,address,address,uint256) (Treasury.sol#1061-1103)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```



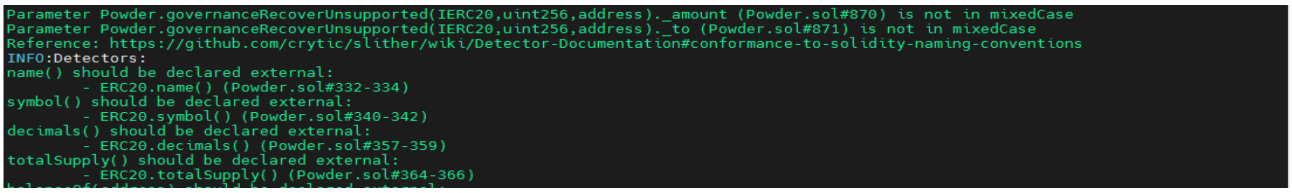
```
INFO:Slither:Treasury.sol analyzed (15 contracts with 75 detectors), 133 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

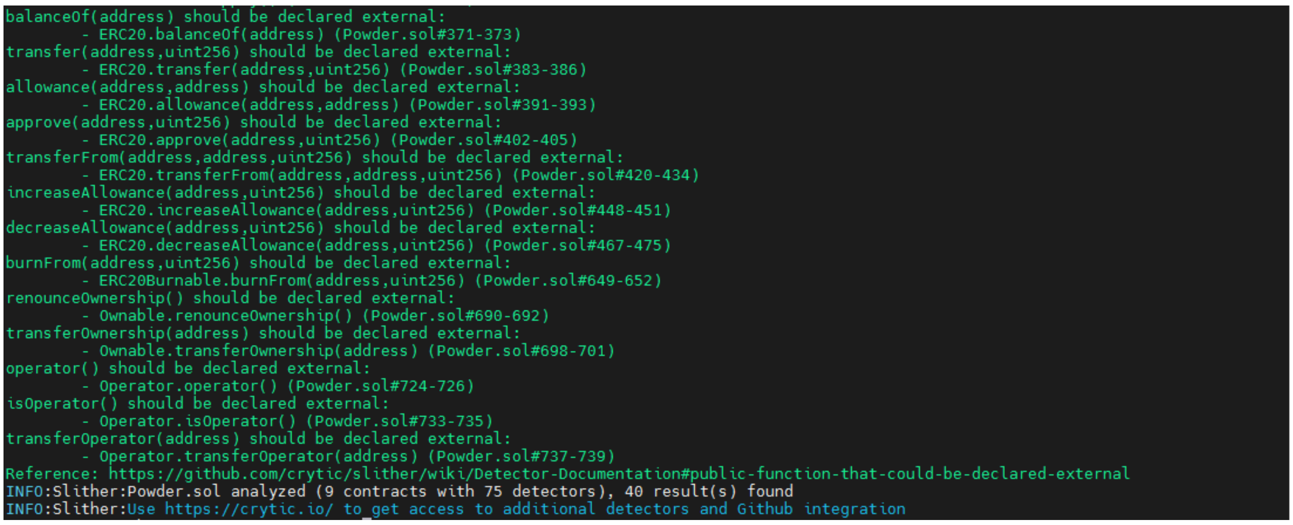
Slither log >> Powder.sol



```
INFO:Detectors:
Powder.setTreasuryFund(address)._communityFund (Powder.sol#799) lacks a zero-check on :
                - communityFund = _communityFund (Powder.sol#801)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Powder.unclaimedTreasuryFund() (Powder.sol#810-815) uses timestamp for comparisons
        Dangerous comparisons:
        - _now > endTime (Powder.sol#812)
        - communityFundLastClaimed >= _now (Powder.sol#813)
Powder.unclaimedDevFund() (Powder.sol#817-822) uses timestamp for comparisons
        Dangerous comparisons:
        - _now > endTime (Powder.sol#819)
        - devFundLastClaimed >= _now (Powder.sol#820)
Powder.unclaimedDigitsDaoFund() (Powder.sol#824-829) uses timestamp for comparisons
        Dangerous comparisons:
        - _now > endTime (Powder.sol#826)
        - digitsDaoFundLastClaimed >= _now (Powder.sol#827)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Context._msgData() (Powder.sol#300-302) is never used and should be removed
SafeMath.add(uint256,uint256) (Powder.sol#72-76) is never used and should be removed
SafeMath.div(uint256,uint256,string) (Powder.sol#177-180) is never used and should be removed
SafeMath.mod(uint256,uint256) (Powder.sol#139-142) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Powder.sol#197-200) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (Powder.sol#157-160) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (Powder.sol#11-15) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (Powder.sol#47-50) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (Powder.sol#57-60) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (Powder.sol#32-40) is never used and should be removed
SafeMath.trySub(uint256,uint256) (Powder.sol#22-25) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Parameter Powder.setTreasuryFund(address)._communityFund (Powder.sol#799) is not in mixedCase
Parameter Powder.setDevFund(address)._devFund (Powder.sol#804) is not in mixedCase
Parameter Powder.distributeReward(address,address)._farmingIncentiveFund (Powder.sol#855) is not in mixedCase
Parameter Powder.distributeReward(address,address)._powderGenesisRewardPool (Powder.sol#855) is not in mixedCase
Parameter Powder.governanceRecoverUnsupported(IERC20,uint256,address)._token (Powder.sol#869) is not in mixedCase
```

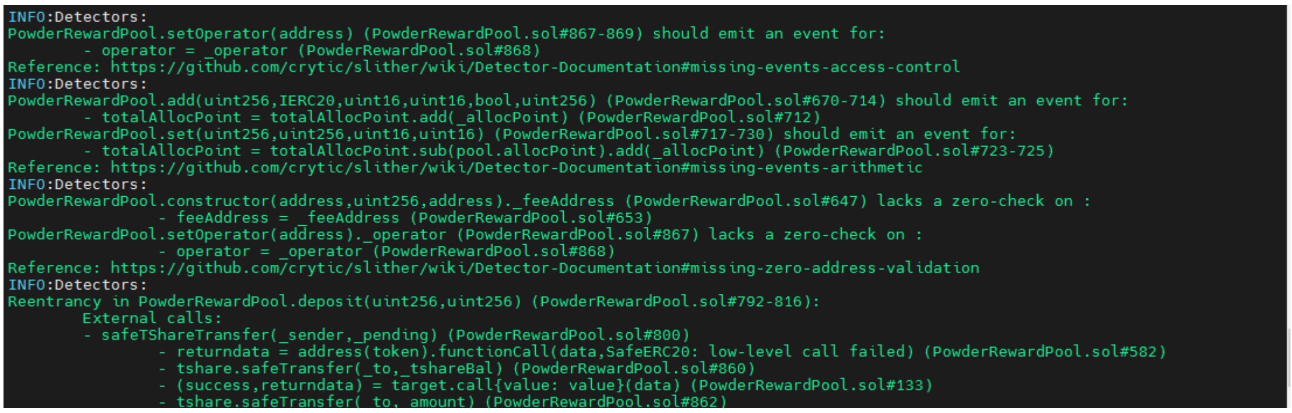


```
Parameter Powder.governanceRecoverUnsupported(IERC20,uint256,address)._amount (Powder.sol#870) is not in mixedCase
Parameter Powder.governanceRecoverUnsupported(IERC20,uint256,address)._to (Powder.sol#871) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (Powder.sol#332-334)
symbol() should be declared external:
        - ERC20.symbol() (Powder.sol#340-342)
decimals() should be declared external:
        - ERC20.decimals() (Powder.sol#357-359)
totalSupply() should be declared external:
        - ERC20.totalSupply() (Powder.sol#364-366)
```



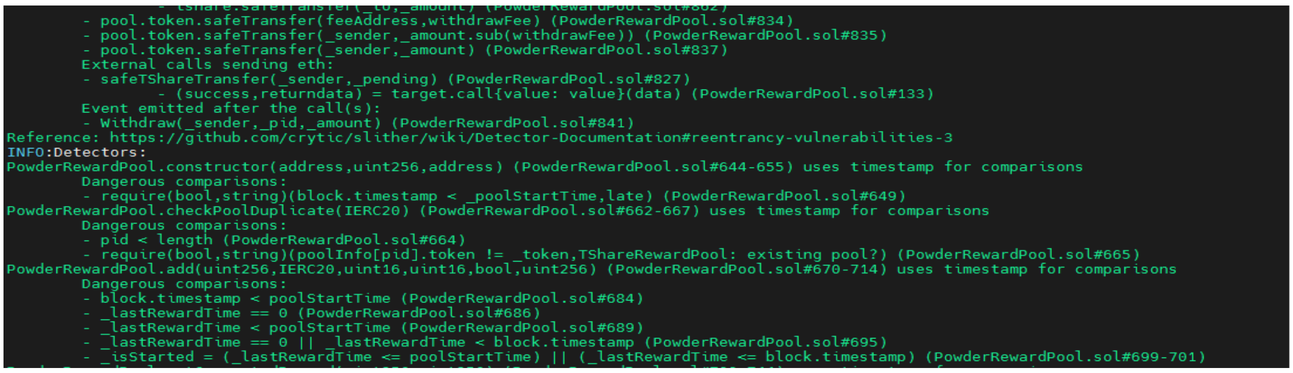
```
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (Powder.sol#371-373)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (Powder.sol#383-386)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (Powder.sol#391-393)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (Powder.sol#402-405)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (Powder.sol#420-434)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (Powder.sol#448-451)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (Powder.sol#467-475)
burnFrom(address,uint256) should be declared external:
        - ERC20Burnable.burnFrom(address,uint256) (Powder.sol#649-652)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Powder.sol#690-692)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Powder.sol#698-701)
operator() should be declared external:
        - Operator.operator() (Powder.sol#724-726)
isOperator() should be declared external:
        - Operator.isOperator() (Powder.sol#733-735)
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (Powder.sol#737-739)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Powder.sol analyzed (9 contracts with 75 detectors), 40 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> PowderRewardPool.sol

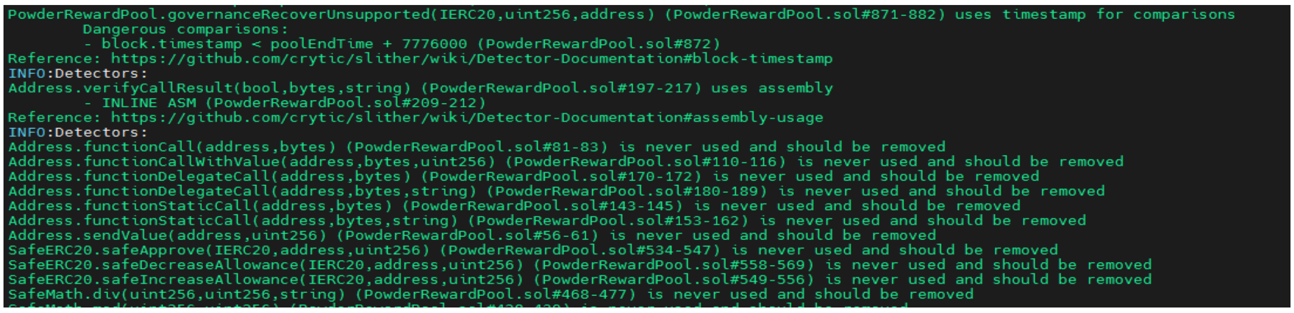


```
INFO:Detectors:
PowderRewardPool.setOperator(address) (PowderRewardPool.sol#867-869) should emit an event for:
        - operator = _operator (PowderRewardPool.sol#868)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
PowderRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256) (PowderRewardPool.sol#670-714) should emit an event for:
        - totalAllocPoint = totalAllocPoint.add(_allocPoint) (PowderRewardPool.sol#712)
PowderRewardPool.set(uint256,uint256,uint16,uint16) (PowderRewardPool.sol#717-730) should emit an event for:
        - totalAllocPoint = totalAllocPoint.sub(pool.allocPoint).add(_allocPoint) (PowderRewardPool.sol#723-725)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
PowderRewardPool.constructor(address,uint256,address)._feeAddress (PowderRewardPool.sol#647) lacks a zero-check on :
                - feeAddress = _feeAddress (PowderRewardPool.sol#653)
PowderRewardPool.setOperator(address)._operator (PowderRewardPool.sol#867) lacks a zero-check on :
                - operator = _operator (PowderRewardPool.sol#868)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in PowderRewardPool.deposit(uint256,uint256) (PowderRewardPool.sol#792-816):
        External calls:
        - safeTShareTransfer(_sender,_pending) (PowderRewardPool.sol#800)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (PowderRewardPool.sol#582)
                - tshare.safeTransfer(_to,_tshareBal) (PowderRewardPool.sol#860)
                - (success,returndata) = target.call{value: value}(data) (PowderRewardPool.sol#133)
                - tshare.safeTransfer(_to,_amount) (PowderRewardPool.sol#862)
```

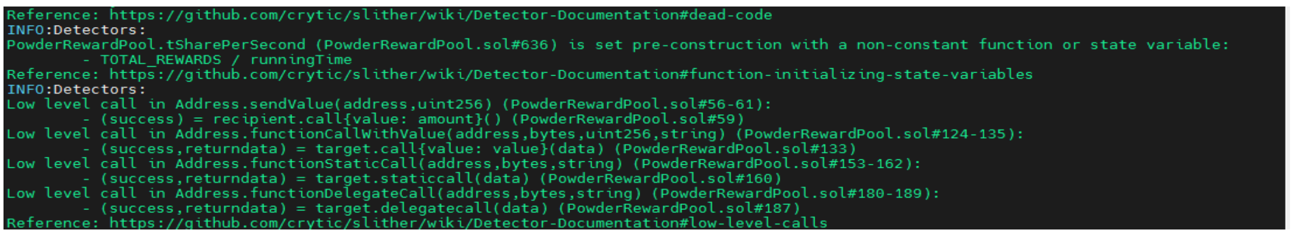
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io



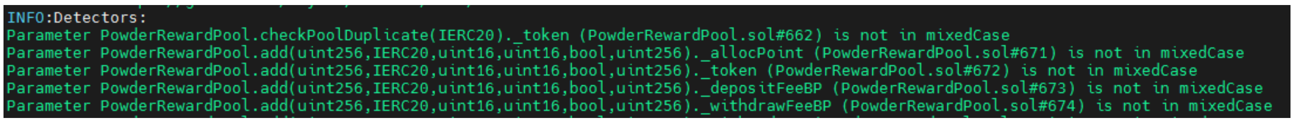
```
                - tshare.safeTransfer(_to,_amount) (PowderRewardPool.sol#862)
        - pool.token.safeTransfer(feeAddress,withdrawFee) (PowderRewardPool.sol#834)
        - pool.token.safeTransfer(_sender,_amount.sub(withdrawFee)) (PowderRewardPool.sol#835)
        - pool.token.safeTransfer(_sender,_amount) (PowderRewardPool.sol#837)
        External calls sending eth:
        - safeTShareTransfer(_sender,_pending) (PowderRewardPool.sol#827)
                - (success,returndata) = target.call{value: value}(data) (PowderRewardPool.sol#133)
        Event emitted after the call(s):
        - Withdraw(_sender,_pid,_amount) (PowderRewardPool.sol#841)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
PowderRewardPool.constructor(address,uint256,address) (PowderRewardPool.sol#644-655) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp < _poolStartTime,late) (PowderRewardPool.sol#649)
PowderRewardPool.checkPoolDuplicate(IERC20) (PowderRewardPool.sol#662-667) uses timestamp for comparisons
        Dangerous comparisons:
        - pid < length (PowderRewardPool.sol#664)
        - require(bool,string)(poolInfo[pid].token != _token,TShareRewardPool: existing pool?) (PowderRewardPool.sol#665)
PowderRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256) (PowderRewardPool.sol#670-714) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < poolStartTime (PowderRewardPool.sol#684)
        - _lastRewardTime == 0 (PowderRewardPool.sol#686)
        - _lastRewardTime < poolStartTime (PowderRewardPool.sol#689)
        - _lastRewardTime == 0 || _lastRewardTime < block.timestamp (PowderRewardPool.sol#695)
        - _isStarted = (_lastRewardTime <= poolStartTime) || (_lastRewardTime <= block.timestamp) (PowderRewardPool.sol#699-701)
```



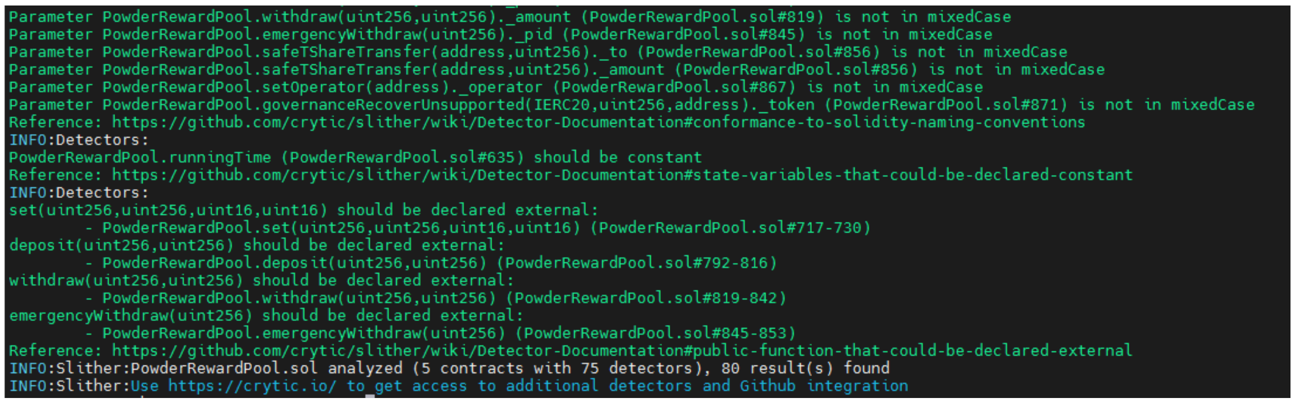
```
PowderRewardPool.governanceRecoverUnsupported(IERC20,uint256,address) (PowderRewardPool.sol#871-882) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < poolEndTime + 7776000 (PowderRewardPool.sol#872)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.verifyCallResult(bool,bytes,string) (PowderRewardPool.sol#197-217) uses assembly
        - INLINE ASM (PowderRewardPool.sol#209-212)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (PowderRewardPool.sol#81-83) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (PowderRewardPool.sol#110-116) is never used and should be removed
Address.functionDelegateCall(address,bytes) (PowderRewardPool.sol#170-172) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (PowderRewardPool.sol#180-189) is never used and should be removed
Address.functionStaticCall(address,bytes) (PowderRewardPool.sol#143-145) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (PowderRewardPool.sol#153-162) is never used and should be removed
Address.sendValue(address,uint256) (PowderRewardPool.sol#56-61) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (PowderRewardPool.sol#534-547) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (PowderRewardPool.sol#558-569) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (PowderRewardPool.sol#549-556) is never used and should be removed
SafeMath.div(uint256,uint256,string) (PowderRewardPool.sol#468-477) is never used and should be removed
```



```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
PowderRewardPool.tSharePerSecond (PowderRewardPool.sol#636) 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) (PowderRewardPool.sol#56-61):
        - (success) = recipient.call{value: amount}() (PowderRewardPool.sol#59)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (PowderRewardPool.sol#124-135):
        - (success,returndata) = target.call{value: value}(data) (PowderRewardPool.sol#133)
Low level call in Address.functionStaticCall(address,bytes,string) (PowderRewardPool.sol#153-162):
        - (success,returndata) = target.staticcall(data) (PowderRewardPool.sol#160)
Low level call in Address.functionDelegateCall(address,bytes,string) (PowderRewardPool.sol#180-189):
        - (success,returndata) = target.delegatecall(data) (PowderRewardPool.sol#187)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
```



```
INFO:Detectors:
Parameter PowderRewardPool.checkPoolDuplicate(IERC20)._token (PowderRewardPool.sol#662) is not in mixedCase
Parameter PowderRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256)._allocPoint (PowderRewardPool.sol#671) is not in mixedCase
Parameter PowderRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256)._token (PowderRewardPool.sol#672) is not in mixedCase
Parameter PowderRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256)._depositFeeBP (PowderRewardPool.sol#673) is not in mixedCase
Parameter PowderRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256)._withdrawFeeBP (PowderRewardPool.sol#674) is not in mixedCase
```

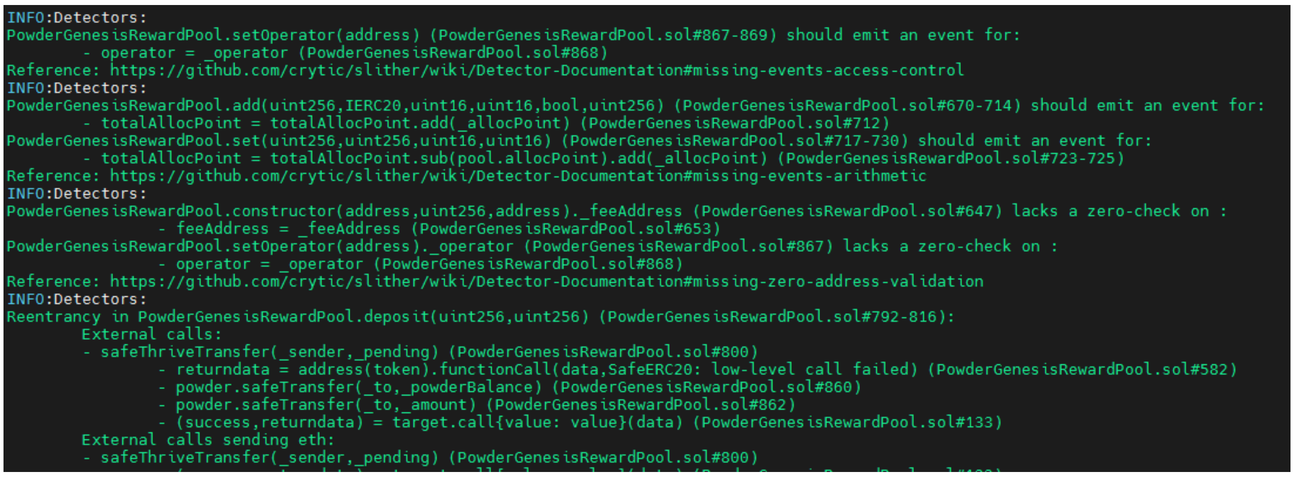


```
Parameter PowderRewardPool.withdraw(uint256,uint256)._amount (PowderRewardPool.sol#819) is not in mixedCase
Parameter PowderRewardPool.emergencyWithdraw(uint256)._pid (PowderRewardPool.sol#845) is not in mixedCase
Parameter PowderRewardPool.safeTShareTransfer(address,uint256)._to (PowderRewardPool.sol#856) is not in mixedCase
Parameter PowderRewardPool.safeTShareTransfer(address,uint256)._amount (PowderRewardPool.sol#856) is not in mixedCase
Parameter PowderRewardPool.setOperator(address)._operator (PowderRewardPool.sol#867) is not in mixedCase
Parameter PowderRewardPool.governanceRecoverUnsupported(IERC20,uint256,address)._token (PowderRewardPool.sol#871) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
PowderRewardPool.runningTime (PowderRewardPool.sol#635) 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:
        - PowderRewardPool.set(uint256,uint256,uint16,uint16) (PowderRewardPool.sol#717-730)
deposit(uint256,uint256) should be declared external:
        - PowderRewardPool.deposit(uint256,uint256) (PowderRewardPool.sol#792-816)
withdraw(uint256,uint256) should be declared external:
        - PowderRewardPool.withdraw(uint256,uint256) (PowderRewardPool.sol#819-842)
emergencyWithdraw(uint256) should be declared external:
        - PowderRewardPool.emergencyWithdraw(uint256) (PowderRewardPool.sol#845-853)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:PowderRewardPool.sol analyzed (5 contracts with 75 detectors), 80 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

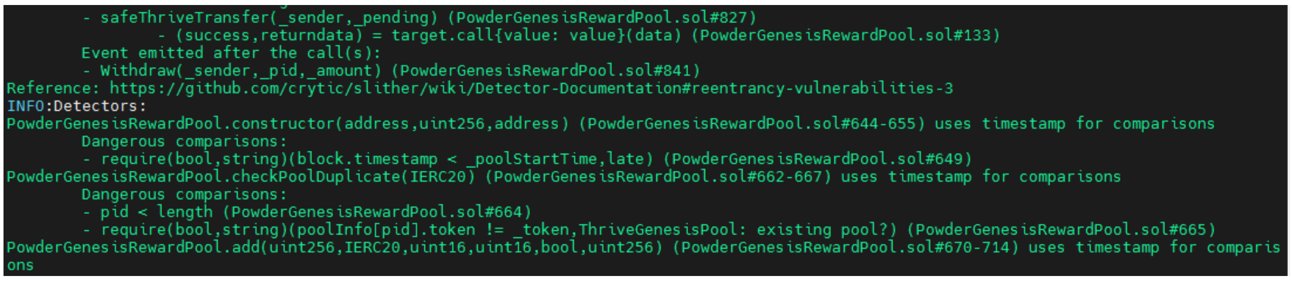
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io

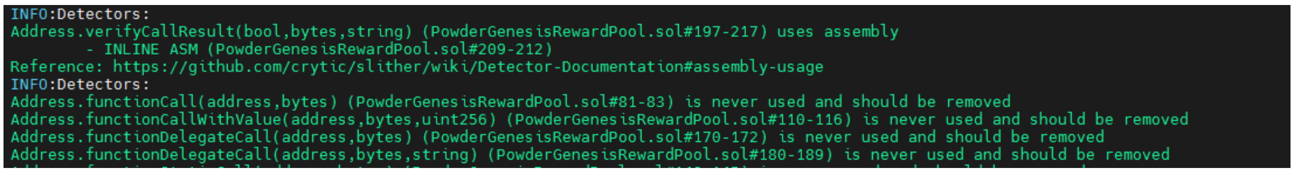
Slither log >> PowderGenesisRewardPool.sol



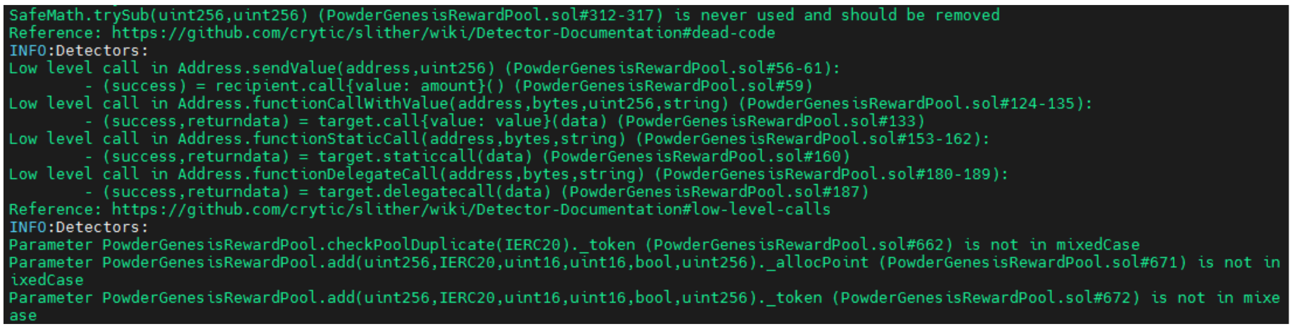
```
INFO:Detectors:
PowderGenesisRewardPool.setOperator(address) (PowderGenesisRewardPool.sol#867-869) should emit an event for:
        - operator = _operator (PowderGenesisRewardPool.sol#868)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
PowderGenesisRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256) (PowderGenesisRewardPool.sol#670-714) should emit an event for:
        - totalAllocPoint = totalAllocPoint.add(_allocPoint) (PowderGenesisRewardPool.sol#712)
PowderGenesisRewardPool.set(uint256,uint256,uint16,uint16) (PowderGenesisRewardPool.sol#717-730) should emit an event for:
        - totalAllocPoint = totalAllocPoint.sub(pool.allocPoint).add(_allocPoint) (PowderGenesisRewardPool.sol#723-725)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
PowderGenesisRewardPool.constructor(address,uint256,address)._feeAddress (PowderGenesisRewardPool.sol#647) lacks a zero-check on :
                - feeAddress = _feeAddress (PowderGenesisRewardPool.sol#653)
PowderGenesisRewardPool.setOperator(address)._operator (PowderGenesisRewardPool.sol#867) lacks a zero-check on :
                - operator = _operator (PowderGenesisRewardPool.sol#868)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in PowderGenesisRewardPool.deposit(uint256,uint256) (PowderGenesisRewardPool.sol#792-816):
        External calls:
        - safeThriveTransfer(_sender,_pending) (PowderGenesisRewardPool.sol#800)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (PowderGenesisRewardPool.sol#582)
                - powder.safeTransfer(_to,_powderBalance) (PowderGenesisRewardPool.sol#860)
                - powder.safeTransfer(_to,_amount) (PowderGenesisRewardPool.sol#862)
                - (success,returndata) = target.call{value: value}(data) (PowderGenesisRewardPool.sol#133)
        External calls sending eth:
        - safeThriveTransfer(_sender,_pending) (PowderGenesisRewardPool.sol#800)
```



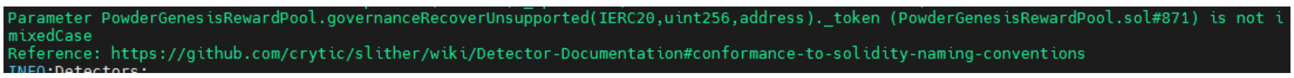
```
        - safeThriveTransfer(_sender,_pending) (PowderGenesisRewardPool.sol#827)
                - (success,returndata) = target.call{value: value}(data) (PowderGenesisRewardPool.sol#133)
        Event emitted after the call(s):
        - Withdraw(_sender,_pid,_amount) (PowderGenesisRewardPool.sol#841)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
PowderGenesisRewardPool.constructor(address,uint256,address) (PowderGenesisRewardPool.sol#644-655) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp < _poolStartTime,late) (PowderGenesisRewardPool.sol#649)
PowderGenesisRewardPool.checkPoolDuplicate(IERC20) (PowderGenesisRewardPool.sol#662-667) uses timestamp for comparisons
        Dangerous comparisons:
        - pid < length (PowderGenesisRewardPool.sol#664)
        - require(bool,string)(poolInfo[pid].token != _token,ThriveGenesisPool: existing pool?) (PowderGenesisRewardPool.sol#665)
PowderGenesisRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256) (PowderGenesisRewardPool.sol#670-714) uses timestamp for comparis
ons
```



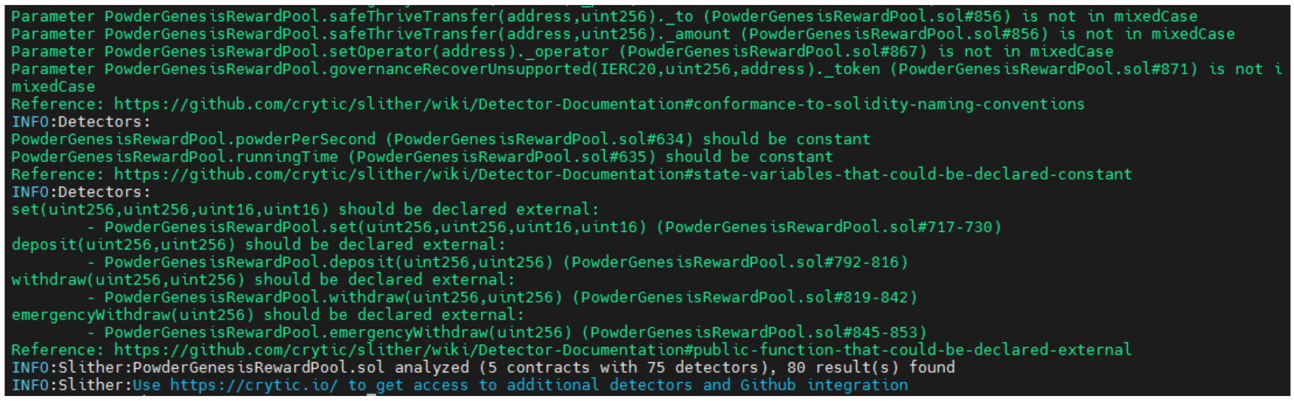
```
INFO:Detectors:
Address.verifyCallResult(bool,bytes,string) (PowderGenesisRewardPool.sol#197-217) uses assembly
        - INLINE ASM (PowderGenesisRewardPool.sol#209-212)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (PowderGenesisRewardPool.sol#81-83) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (PowderGenesisRewardPool.sol#110-116) is never used and should be removed
Address.functionDelegateCall(address,bytes) (PowderGenesisRewardPool.sol#170-172) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (PowderGenesisRewardPool.sol#180-189) is never used and should be removed
```



```
SafeMath.trySub(uint256,uint256) (PowderGenesisRewardPool.sol#312-317) 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) (PowderGenesisRewardPool.sol#56-61):
        - (success) = recipient.call{value: amount}() (PowderGenesisRewardPool.sol#59)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (PowderGenesisRewardPool.sol#124-135):
        - (success,returndata) = target.call{value: value}(data) (PowderGenesisRewardPool.sol#133)
Low level call in Address.functionStaticCall(address,bytes,string) (PowderGenesisRewardPool.sol#153-162):
        - (success,returndata) = target.staticcall(data) (PowderGenesisRewardPool.sol#160)
Low level call in Address.functionDelegateCall(address,bytes,string) (PowderGenesisRewardPool.sol#180-189):
        - (success,returndata) = target.delegatecall(data) (PowderGenesisRewardPool.sol#187)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter PowderGenesisRewardPool.checkPoolDuplicate(IERC20)._token (PowderGenesisRewardPool.sol#662) is not in mixedCase
Parameter PowderGenesisRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256)._allocPoint (PowderGenesisRewardPool.sol#671) is not in
ixedCase
Parameter PowderGenesisRewardPool.add(uint256,IERC20,uint16,uint16,bool,uint256)._token (PowderGenesisRewardPool.sol#672) is not in mixe
ase
```

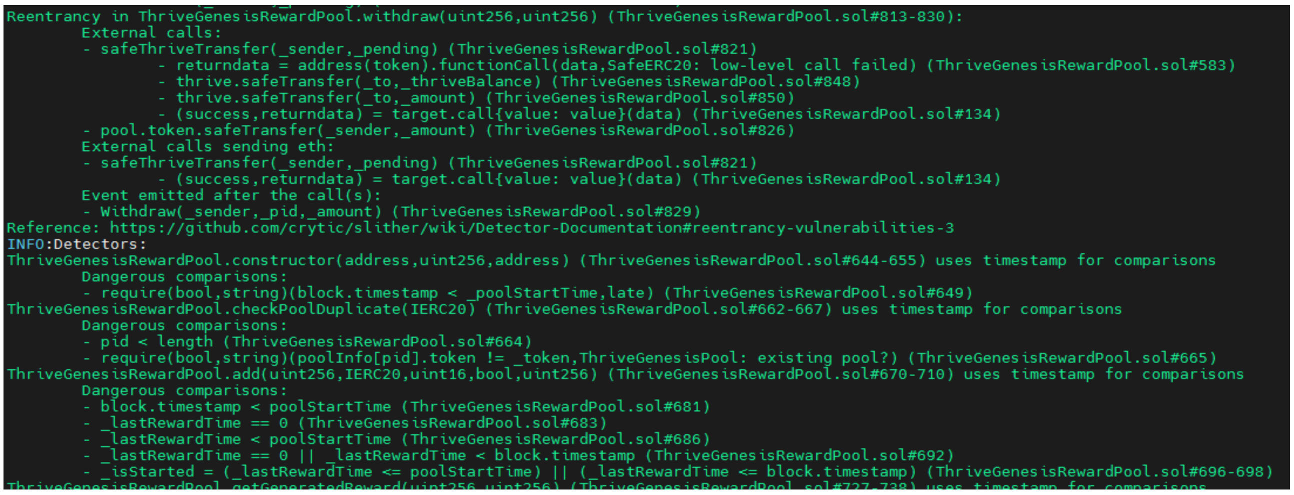


```
Parameter PowderGenesisRewardPool.governanceRecoverUnsupported(IERC20,uint256,address)._token (PowderGenesisRewardPool.sol#871) is not i
mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
```

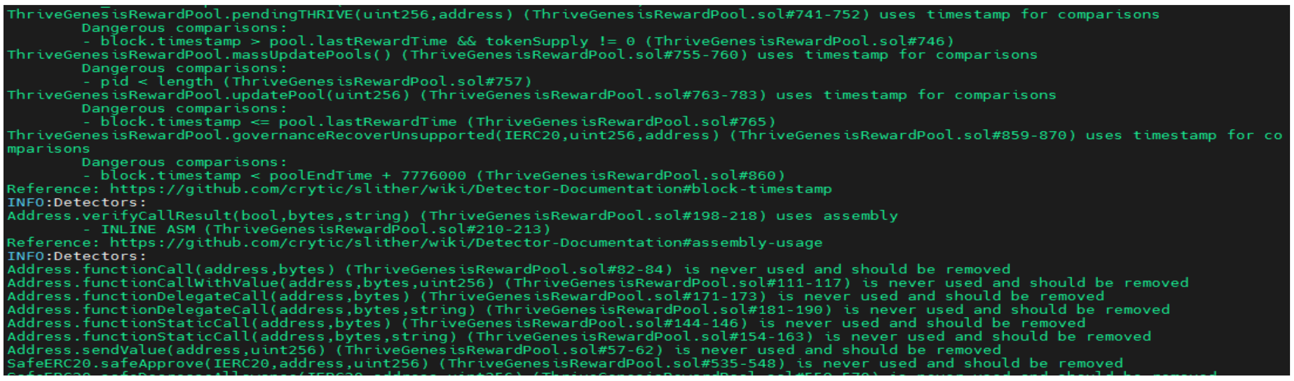


```
Parameter PowderGenesisRewardPool.safeThriveTransfer(address,uint256)._to (PowderGenesisRewardPool.sol#856) is not in mixedCase
Parameter PowderGenesisRewardPool.safeThriveTransfer(address,uint256)._amount (PowderGenesisRewardPool.sol#856) is not in mixedCase
Parameter PowderGenesisRewardPool.setOperator(address)._operator (PowderGenesisRewardPool.sol#867) is not in mixedCase
Parameter PowderGenesisRewardPool.governanceRecoverUnsupported(IERC20,uint256,address)._token (PowderGenesisRewardPool.sol#871) is not i
mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
PowderGenesisRewardPool.powderPerSecond (PowderGenesisRewardPool.sol#634) should be constant
PowderGenesisRewardPool.runningTime (PowderGenesisRewardPool.sol#635) 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:
        - PowderGenesisRewardPool.set(uint256,uint256,uint16,uint16) (PowderGenesisRewardPool.sol#717-730)
deposit(uint256,uint256) should be declared external:
        - PowderGenesisRewardPool.deposit(uint256,uint256) (PowderGenesisRewardPool.sol#792-816)
withdraw(uint256,uint256) should be declared external:
        - PowderGenesisRewardPool.withdraw(uint256,uint256) (PowderGenesisRewardPool.sol#819-842)
emergencyWithdraw(uint256) should be declared external:
        - PowderGenesisRewardPool.emergencyWithdraw(uint256) (PowderGenesisRewardPool.sol#845-853)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:PowderGenesisRewardPool.sol analyzed (5 contracts with 75 detectors), 80 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

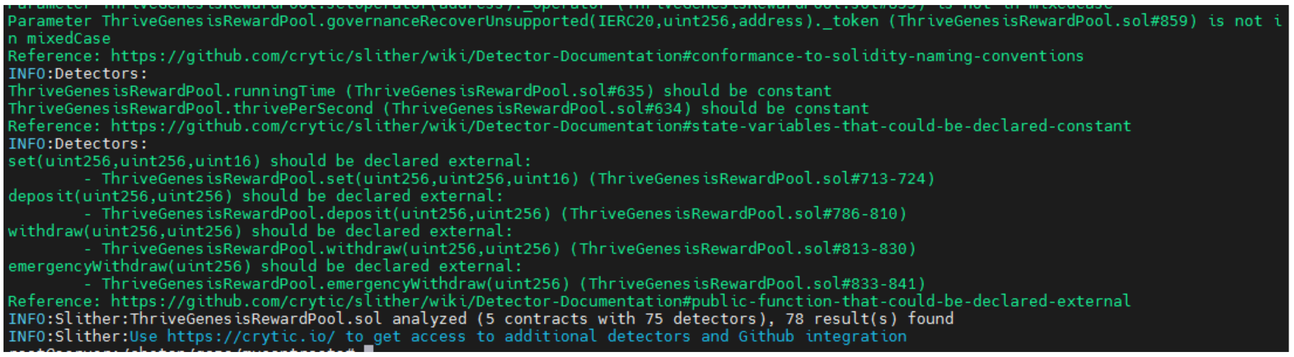
Slither log >> ThriveGenesisRewardPool.sol



```
Reentrancy in ThriveGenesisRewardPool.withdraw(uint256,uint256) (ThriveGenesisRewardPool.sol#813-830):
        External calls:
        - safeThriveTransfer(_sender,_pending) (ThriveGenesisRewardPool.sol#821)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (ThriveGenesisRewardPool.sol#583)
                - thrive.safeTransfer(_to,_thriveBalance) (ThriveGenesisRewardPool.sol#848)
                - thrive.safeTransfer(_to,_amount) (ThriveGenesisRewardPool.sol#850)
                - (success,returndata) = target.call{value: value}(data) (ThriveGenesisRewardPool.sol#134)
        - pool.token.safeTransfer(_sender,_amount) (ThriveGenesisRewardPool.sol#826)
        External calls sending eth:
        - safeThriveTransfer(_sender,_pending) (ThriveGenesisRewardPool.sol#821)
                - (success,returndata) = target.call{value: value}(data) (ThriveGenesisRewardPool.sol#134)
        Event emitted after the call(s):
        - Withdraw(_sender,_pid,_amount) (ThriveGenesisRewardPool.sol#829)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
ThriveGenesisRewardPool.constructor(address,uint256,address) (ThriveGenesisRewardPool.sol#644-655) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp < _poolStartTime,late) (ThriveGenesisRewardPool.sol#649)
ThriveGenesisRewardPool.checkPoolDuplicate(IERC20) (ThriveGenesisRewardPool.sol#662-667) uses timestamp for comparisons
        Dangerous comparisons:
        - pid < length (ThriveGenesisRewardPool.sol#664)
        - require(bool,string)(poolInfo[pid].token != _token,ThriveGenesisPool: existing pool?) (ThriveGenesisRewardPool.sol#665)
ThriveGenesisRewardPool.add(uint256,IERC20,uint16,bool,uint256) (ThriveGenesisRewardPool.sol#670-710) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < poolStartTime (ThriveGenesisRewardPool.sol#681)
        - _lastRewardTime == 0 (ThriveGenesisRewardPool.sol#683)
        - _lastRewardTime < poolStartTime (ThriveGenesisRewardPool.sol#686)
        - _lastRewardTime == 0 || _lastRewardTime < block.timestamp (ThriveGenesisRewardPool.sol#692)
        - _isStarted = (_lastRewardTime <= poolStartTime) || (_lastRewardTime <= block.timestamp) (ThriveGenesisRewardPool.sol#696-698)
ThriveGenesisRewardPool.getGeneratedReward(uint256,uint256) (ThriveGenesisRewardPool.sol#727-738) uses timestamp for comparisons
```



```
ThriveGenesisRewardPool.pendingTHRIVE(uint256,address) (ThriveGenesisRewardPool.sol#741-752) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp > pool.lastRewardTime && tokenSupply != 0 (ThriveGenesisRewardPool.sol#746)
ThriveGenesisRewardPool.massUpdatePools() (ThriveGenesisRewardPool.sol#755-760) uses timestamp for comparisons
        Dangerous comparisons:
        - pid < length (ThriveGenesisRewardPool.sol#757)
ThriveGenesisRewardPool.updatePool(uint256) (ThriveGenesisRewardPool.sol#763-783) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp <= pool.lastRewardTime (ThriveGenesisRewardPool.sol#765)
ThriveGenesisRewardPool.governanceRecoverUnsupported(IERC20,uint256,address) (ThriveGenesisRewardPool.sol#859-870) uses timestamp for co
mparisons
        Dangerous comparisons:
        - block.timestamp < poolEndTime + 7776000 (ThriveGenesisRewardPool.sol#860)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.verifyCallResult(bool,bytes,string) (ThriveGenesisRewardPool.sol#198-218) uses assembly
        - INLINE ASM (ThriveGenesisRewardPool.sol#210-213)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (ThriveGenesisRewardPool.sol#82-84) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (ThriveGenesisRewardPool.sol#111-117) is never used and should be removed
Address.functionDelegateCall(address,bytes) (ThriveGenesisRewardPool.sol#171-173) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (ThriveGenesisRewardPool.sol#181-190) is never used and should be removed
Address.functionStaticCall(address,bytes) (ThriveGenesisRewardPool.sol#144-146) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (ThriveGenesisRewardPool.sol#154-163) is never used and should be removed
Address.sendValue(address,uint256) (ThriveGenesisRewardPool.sol#57-62) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (ThriveGenesisRewardPool.sol#535-548) is never used and should be removed
```

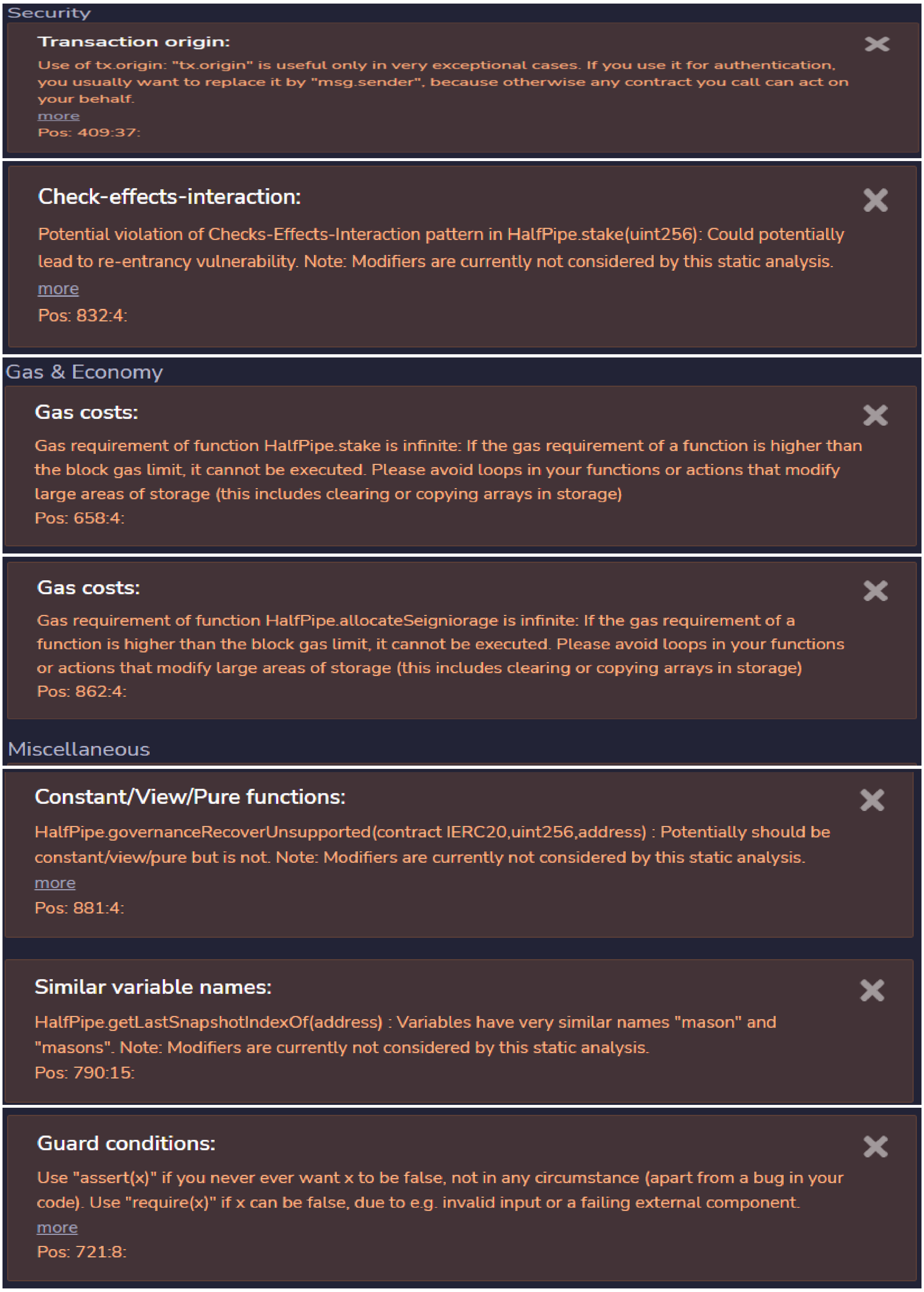


```
Parameter ThriveGenesisRewardPool.governanceRecoverUnsupported(IERC20,uint256,address)._token (ThriveGenesisRewardPool.sol#859) is not i
n mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
ThriveGenesisRewardPool.runningTime (ThriveGenesisRewardPool.sol#635) should be constant
ThriveGenesisRewardPool.thrivePerSecond (ThriveGenesisRewardPool.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) should be declared external:
        - ThriveGenesisRewardPool.set(uint256,uint256,uint16) (ThriveGenesisRewardPool.sol#713-724)
deposit(uint256,uint256) should be declared external:
        - ThriveGenesisRewardPool.deposit(uint256,uint256) (ThriveGenesisRewardPool.sol#786-810)
withdraw(uint256,uint256) should be declared external:
        - ThriveGenesisRewardPool.withdraw(uint256,uint256) (ThriveGenesisRewardPool.sol#813-830)
emergencyWithdraw(uint256) should be declared external:
        - ThriveGenesisRewardPool.emergencyWithdraw(uint256) (ThriveGenesisRewardPool.sol#833-841)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:ThriveGenesisRewardPool.sol analyzed (5 contracts with 75 detectors), 78 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Solidity Static Analysis

HalfPipe.sol



Security

Transaction origin:

Use of tx.origin: "tx.origin" is useful only in very exceptional cases. If you use it for authentication, you usually want to replace it by "msg.sender", because otherwise any contract you call can act on your behalf.

more

Pos: 409:37:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in HalfPipe.stake(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 832:4:

Gas & Economy

Gas costs:

Gas requirement of function HalfPipe.stake is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 658:4:

Gas costs:

Gas requirement of function HalfPipe.allocateSeigniorage is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 862:4:

Miscellaneous

Constant/View/Pure functions:

HalfPipe.governanceRecoverUnsupported(contract IERC20,uint256,address) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 881:4:

Similar variable names:

HalfPipe.getLastSnapshotIndexOf(address) : Variables have very similar names "mason" and "masons". Note: Modifiers are currently not considered by this static analysis.

Pos: 790:15:

Guard conditions:

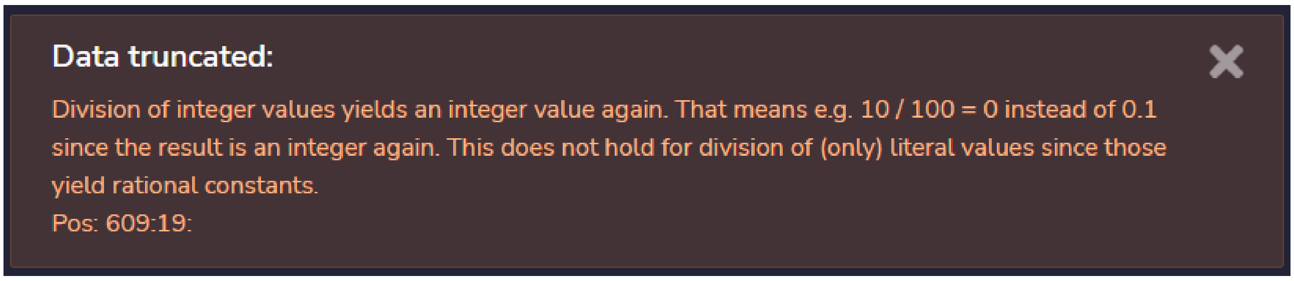
Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 721:8:

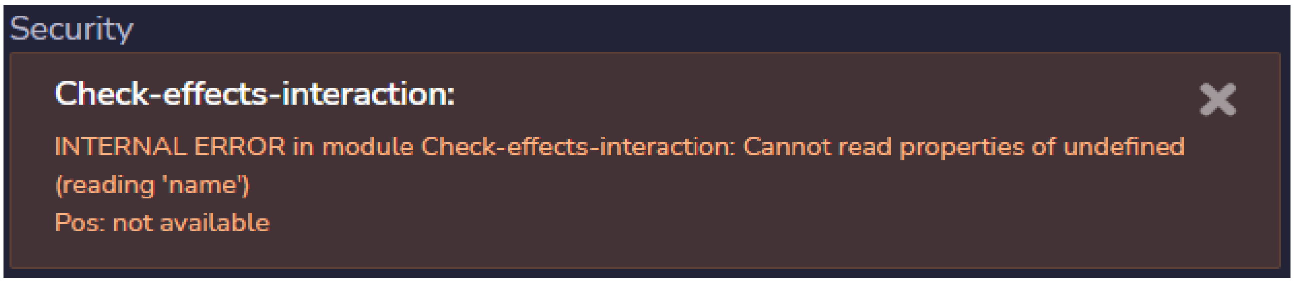
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io


Data truncated:

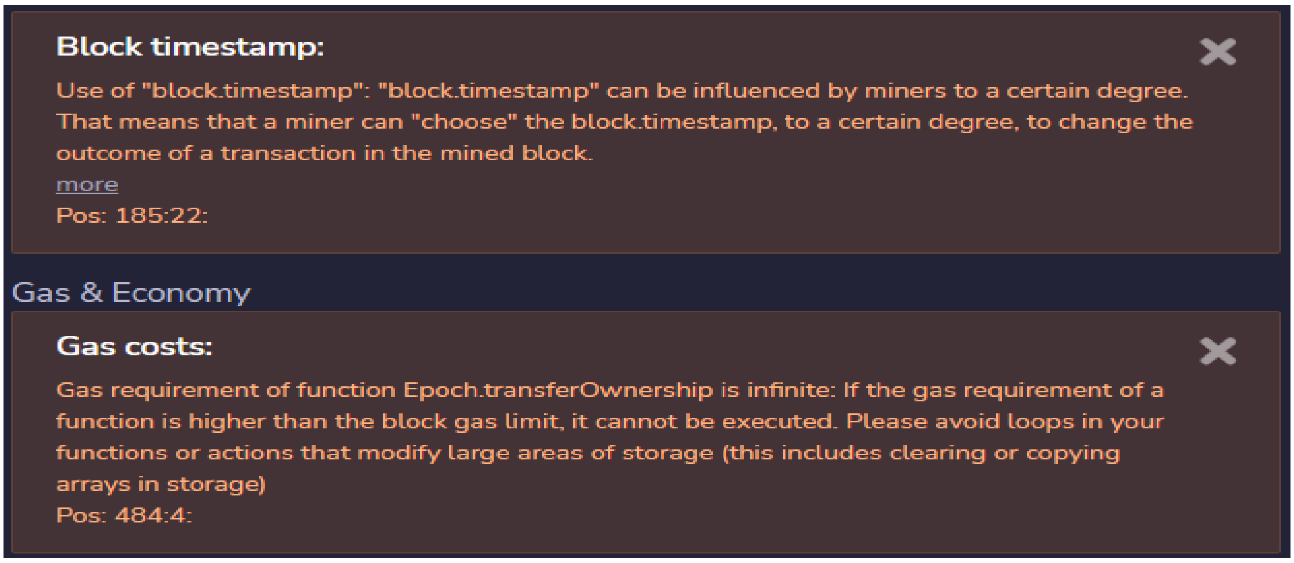
Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 609:19:

Oracle.sol


Security

Check-effects-interaction:

INTERNAL ERROR in module Check-effects-interaction: Cannot read properties of undefined (reading 'name')
Pos: not available

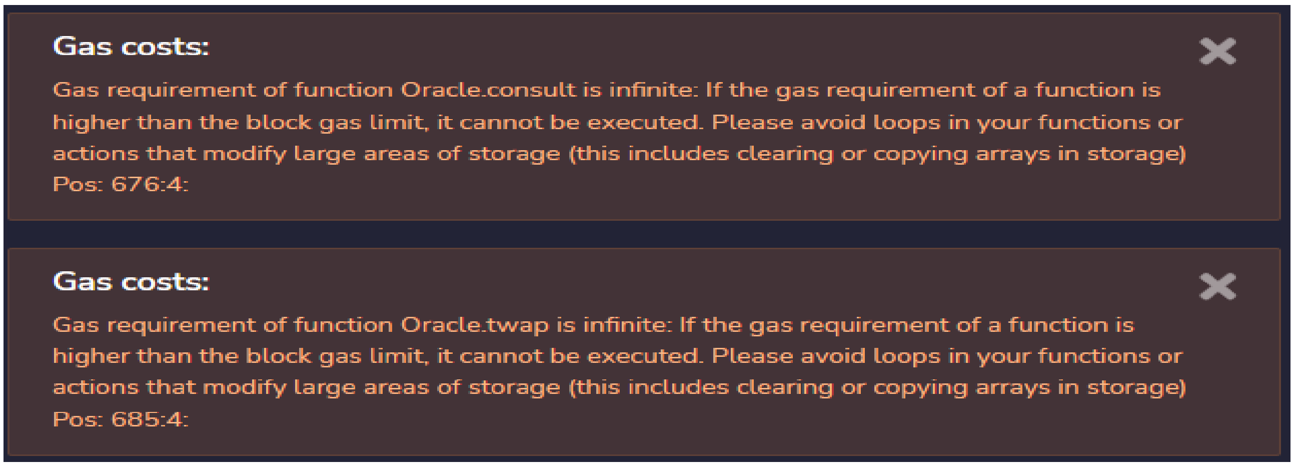

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 185:22:

Gas & Economy

Gas costs:

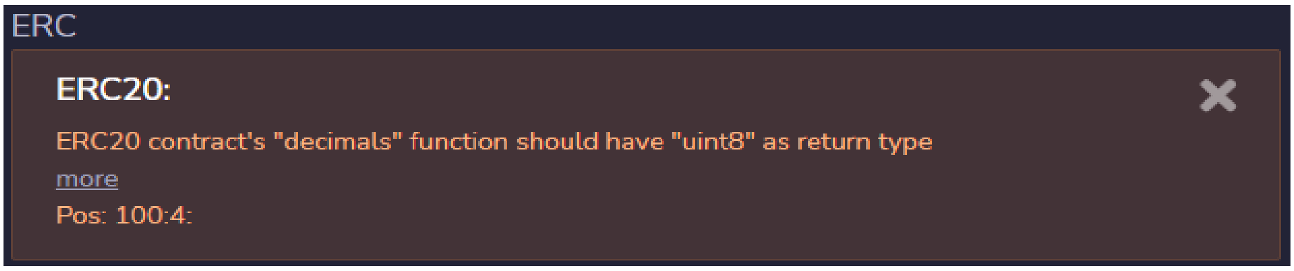
Gas requirement of function Epoch.transferOwnership is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 484:4:


Gas costs:

Gas requirement of function Oracle.consult is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 676:4:

Gas costs:

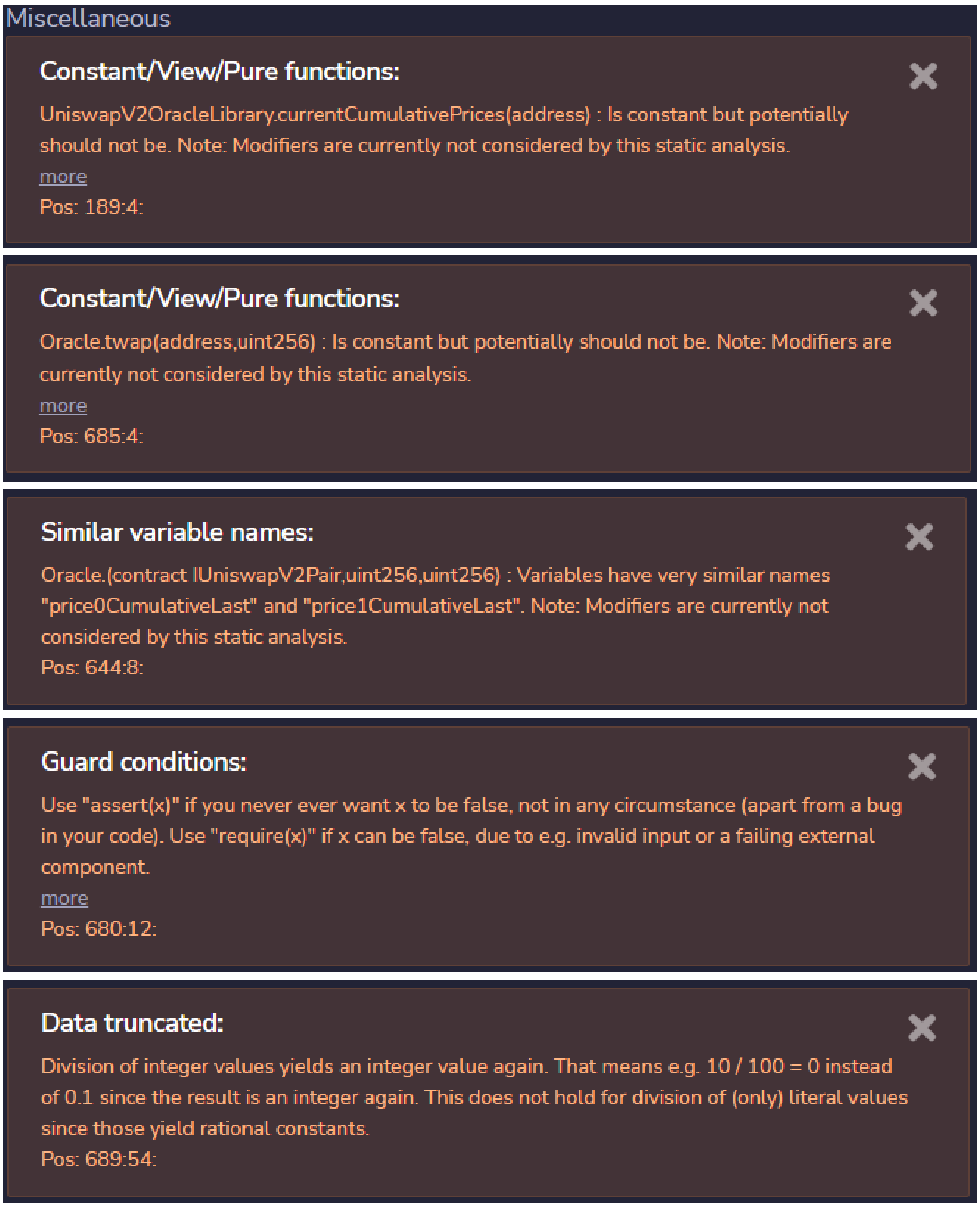
Gas requirement of function Oracle.twap is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 685:4:


ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 100:4:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io


Miscellaneous

Constant/View/Pure functions:

UniswapV2OracleLibrary.currentCumulativePrices(address) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 189:4:

Constant/View/Pure functions:

Oracle.twap(address,uint256) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 685:4:

Similar variable names:

Oracle.(contract IUniswapV2Pair,uint256,uint256) : Variables have very similar names "price0CumulativeLast" and "price1CumulativeLast". Note: Modifiers are currently not considered by this static analysis.
Pos: 644:8:

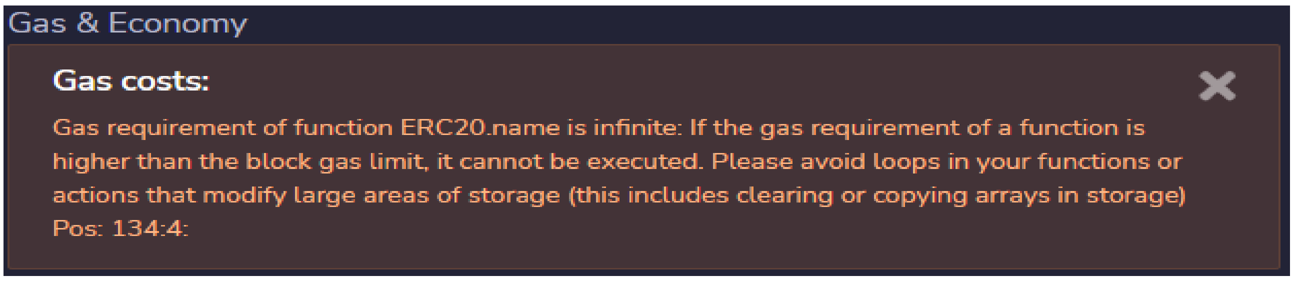
Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 680:12:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 689:54:

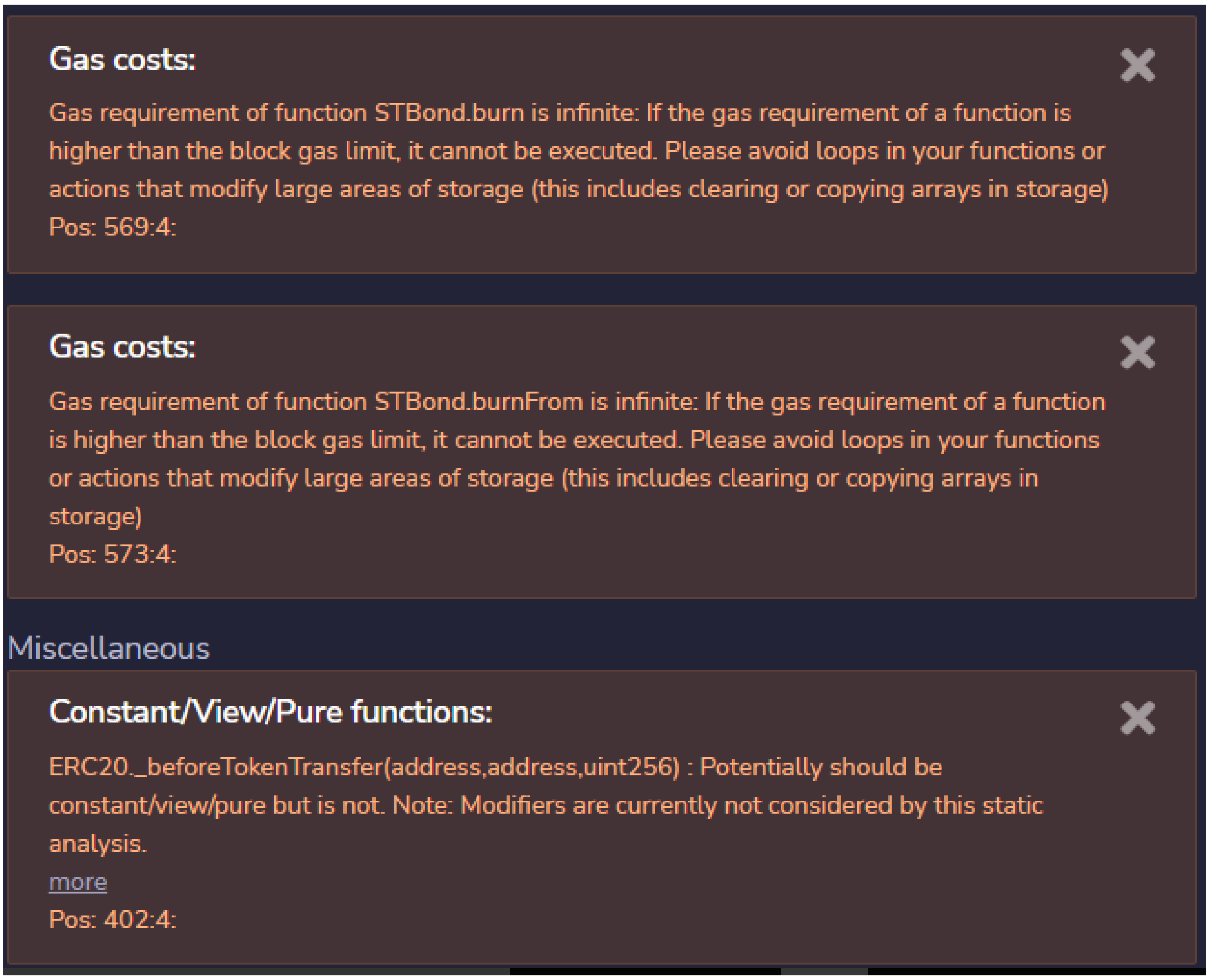
STBond.sol


Gas & Economy

Gas costs:

Gas requirement of function ERC20.name is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 134:4:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io



Gas costs:

Gas requirement of function STBond.burn is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 569:4:

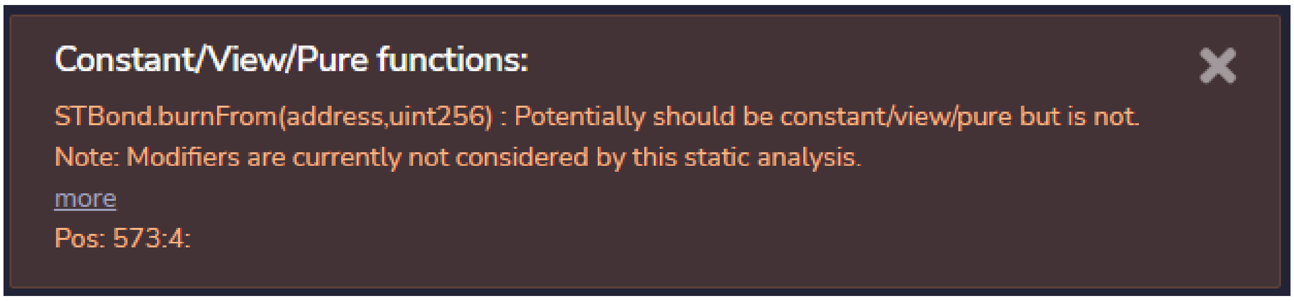
Gas costs:

Gas requirement of function STBond.burnFrom is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 573:4:

Miscellaneous

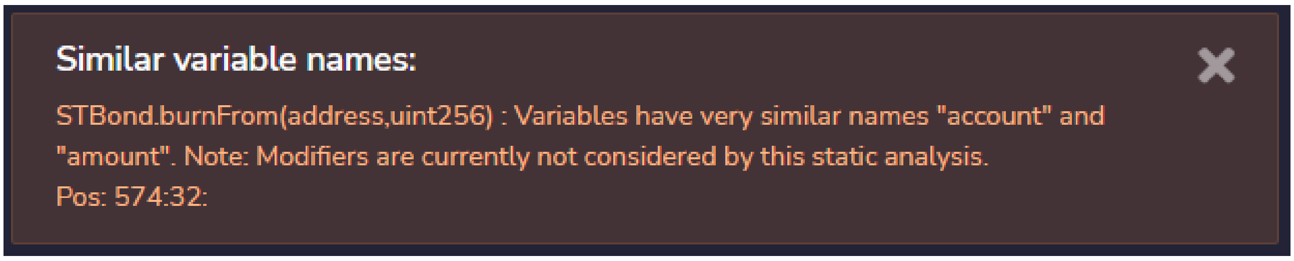
Constant/View/Pure functions:

ERC20._beforeTokenTransfer(address,address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 402:4:



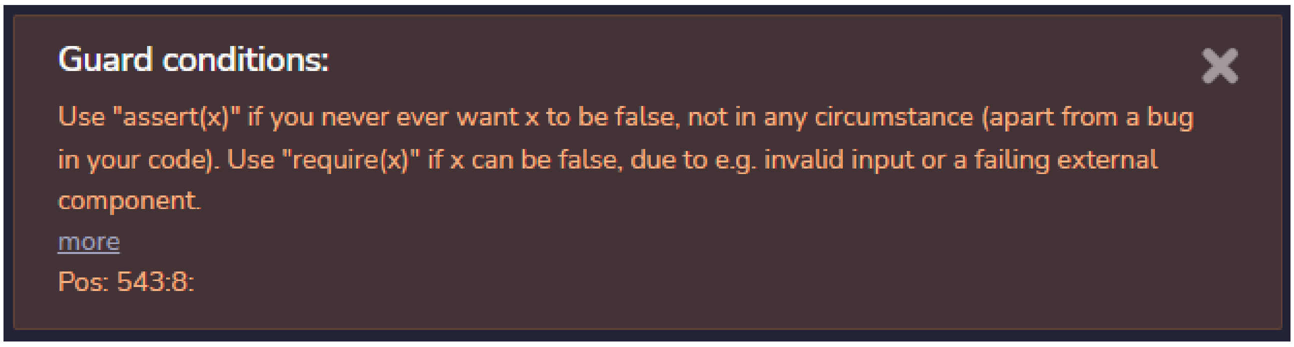
Constant/View/Pure functions:

STBond.burnFrom(address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 573:4:



Similar variable names:

STBond.burnFrom(address,uint256) : Variables have very similar names "account" and "amount". Note: Modifiers are currently not considered by this static analysis.
Pos: 574:32:

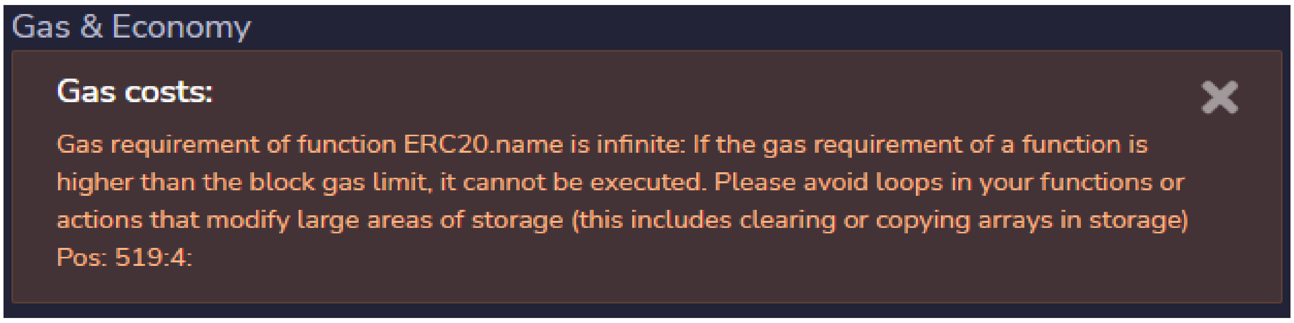


Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 543:8:

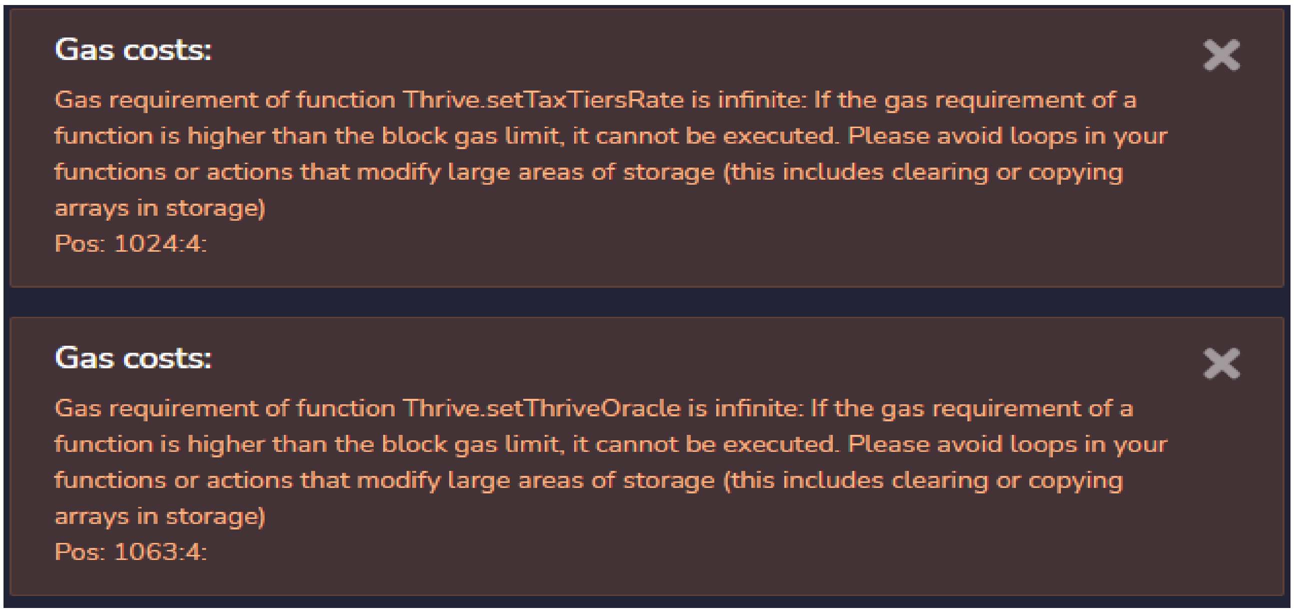
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Thrive.sol


Gas & Economy

Gas costs:

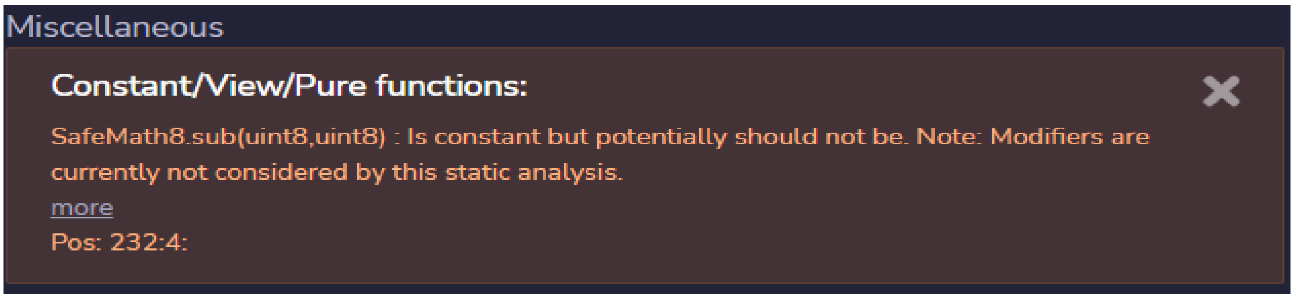
Gas requirement of function ERC20.name is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 519:4:


Gas costs:

Gas requirement of function Thrive.setTaxTiersRate is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 1024:4:

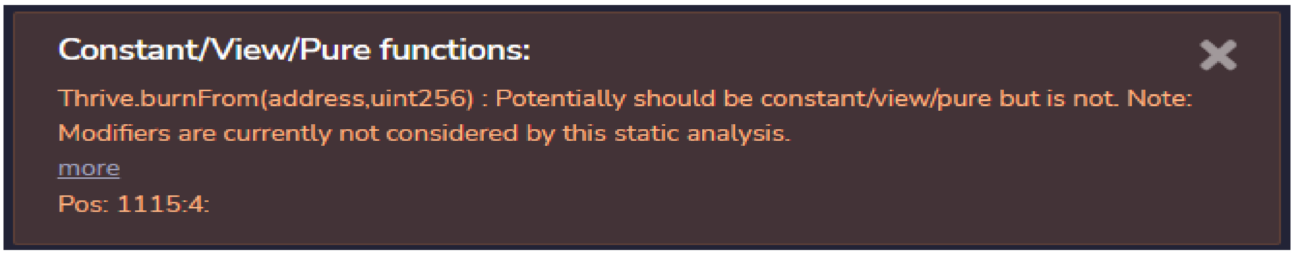
Gas costs:

Gas requirement of function Thrive.setThriveOracle is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 1063:4:

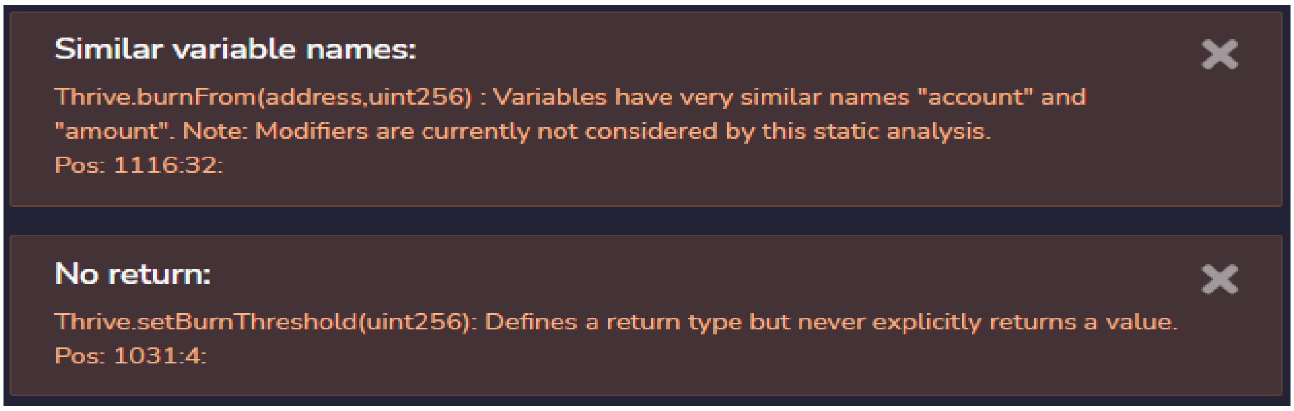

Miscellaneous

Constant/View/Pure functions:

SafeMath8.sub(uint8,uint8) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 232:4:


Constant/View/Pure functions:

Thrive.burnFrom(address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1115:4:

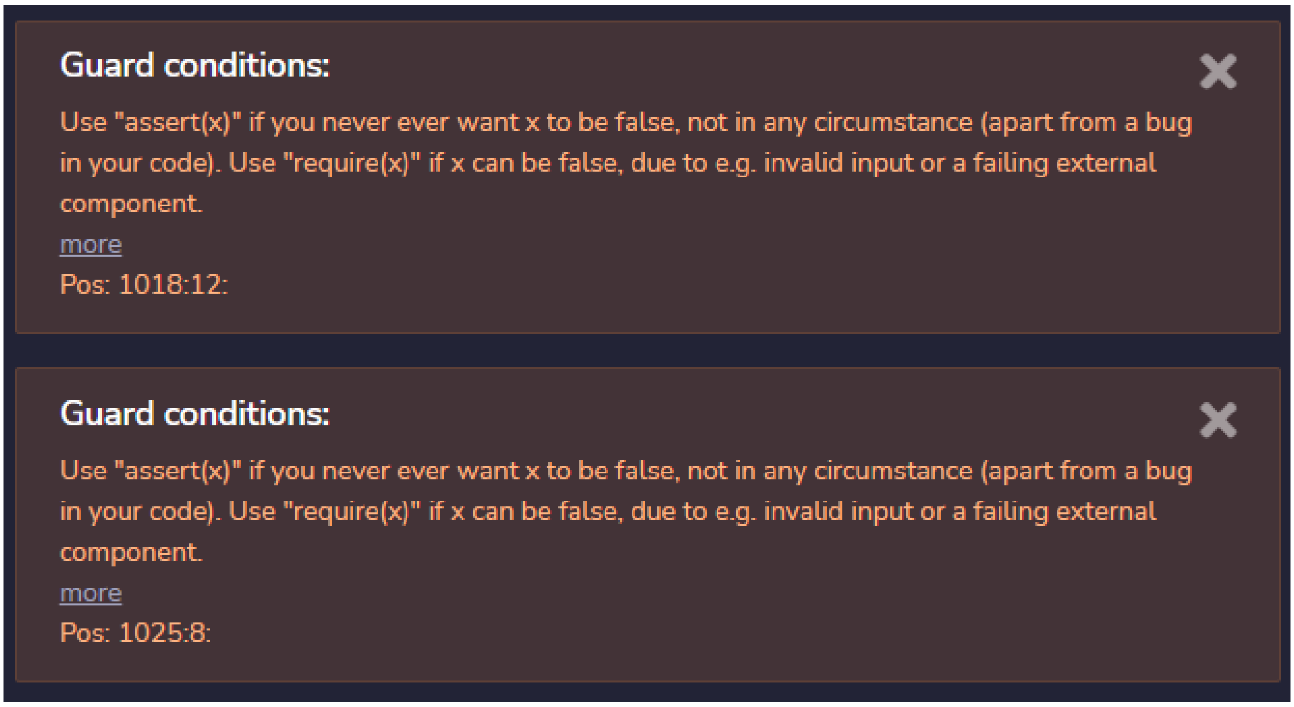

Similar variable names:

Thrive.burnFrom(address,uint256) : Variables have very similar names "account" and "amount". Note: Modifiers are currently not considered by this static analysis.
Pos: 1116:32:

No return:

Thrive.setBurnThreshold(uint256): Defines a return type but never explicitly returns a value.
Pos: 1031:4:

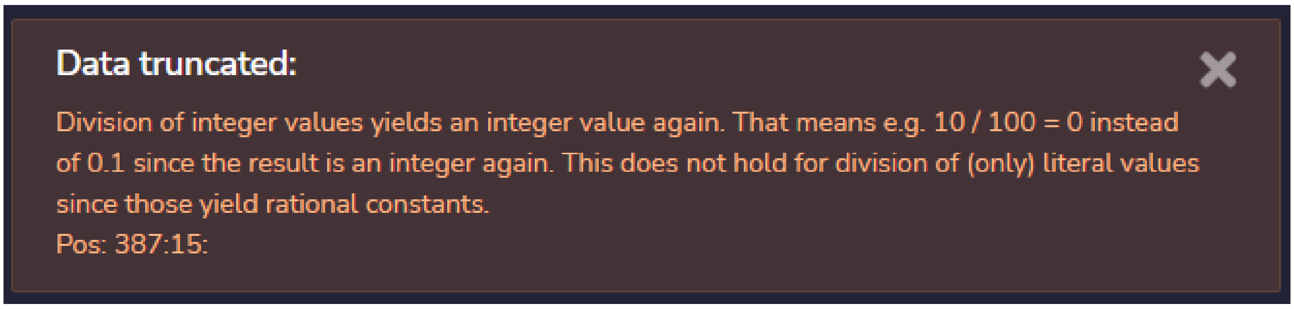
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io


Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 1018:12:

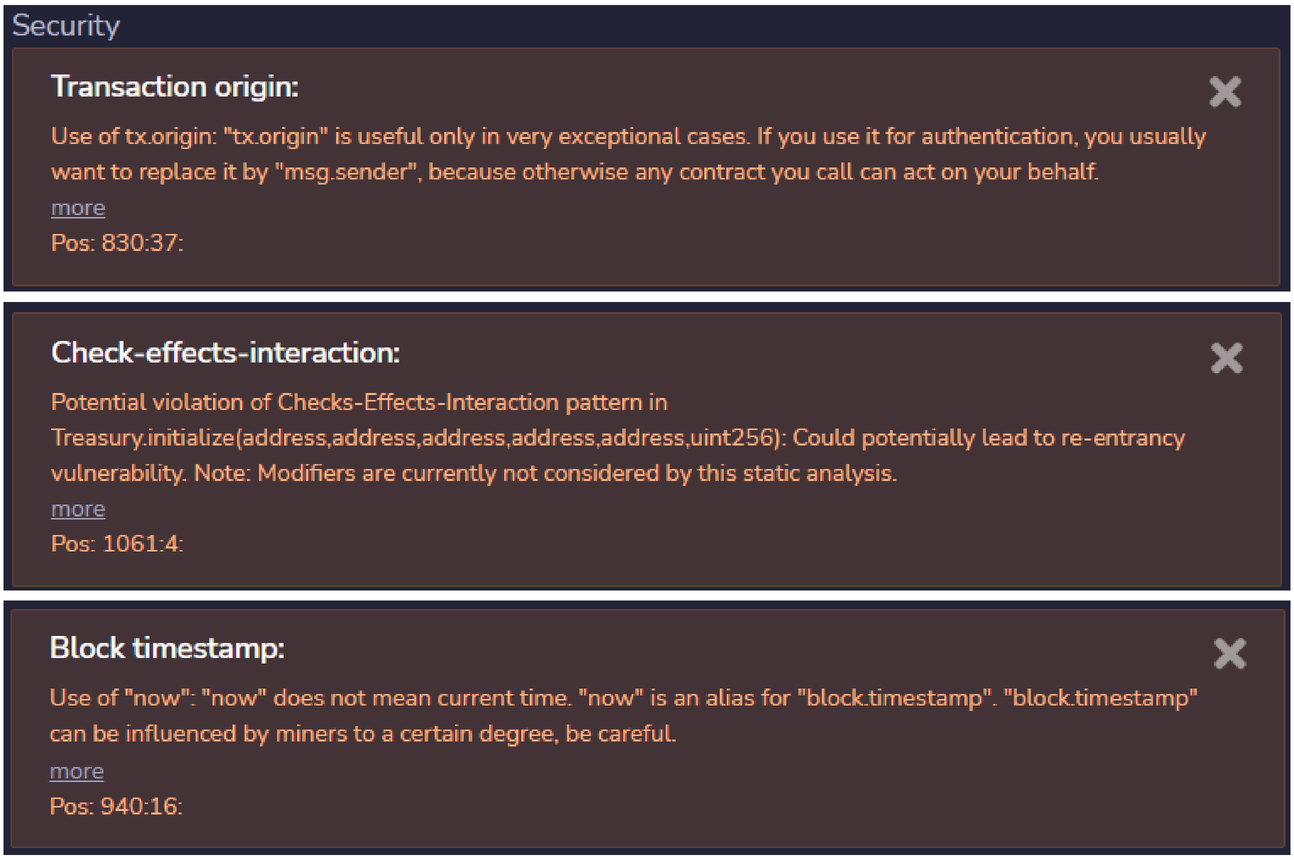
Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 1025:8:


Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 387:15:

Treasury.sol


Security

Transaction origin:

Use of tx.origin: "tx.origin" is useful only in very exceptional cases. If you use it for authentication, you usually want to replace it by "msg.sender", because otherwise any contract you call can act on your behalf.
more
Pos: 830:37:

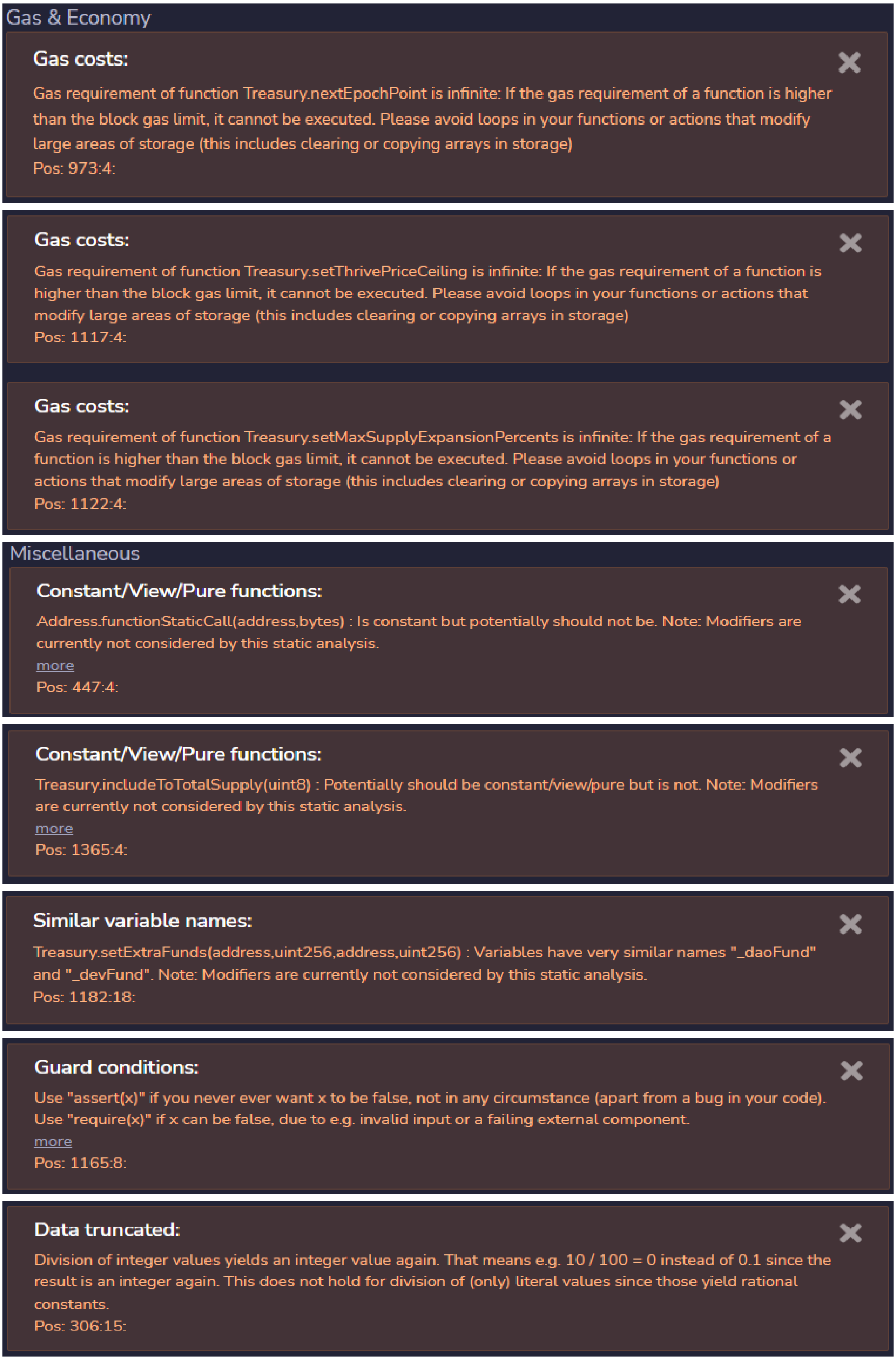
Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Treasury.initialize(address,address,address,address,address,uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1061:4:

Block timestamp:

Use of "now": "now" does not mean current time. "now" is an alias for "block.timestamp". "block.timestamp" can be influenced by miners to a certain degree, be careful.
more
Pos: 940:16:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io



Gas & Economy

Gas costs:

Gas requirement of function Treasury.nextEpochPoint is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 973:4:

Gas costs:

Gas requirement of function Treasury.setThrivePriceCeiling is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 1117:4:

Gas costs:

Gas requirement of function Treasury.setMaxSupplyExpansionPercents is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 1122:4:

Miscellaneous

Constant/View/Pure functions:

Address.functionStaticCall(address,bytes) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 447:4:

Constant/View/Pure functions:

Treasury.includeToTotalSupply(uint8) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1365:4:

Similar variable names:

Treasury.setExtraFunds(address,uint256,address,uint256) : Variables have very similar names "_daoFund" and "_devFund". Note: Modifiers are currently not considered by this static analysis.
Pos: 1182:18:

Guard conditions:

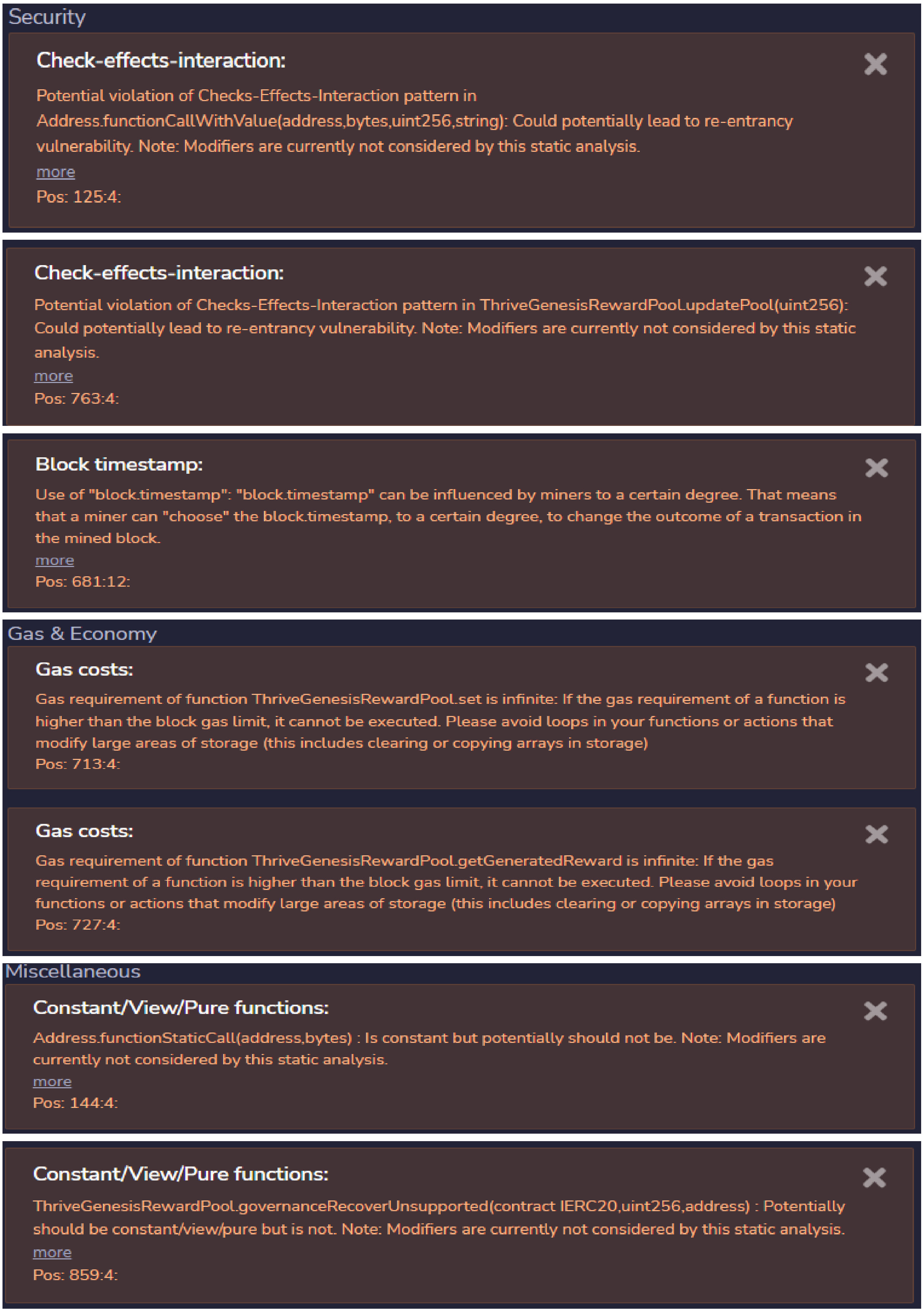
Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 1165:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 306:15:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

ThriveGenesisRewardPool.sol



Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Address.functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 125:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in ThriveGenesisRewardPool.updatePool(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 763:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 681:12:

Gas & Economy

Gas costs:

Gas requirement of function ThriveGenesisRewardPool.set is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 713:4:

Gas costs:

Gas requirement of function ThriveGenesisRewardPool.getGeneratedReward is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 727:4:

Miscellaneous

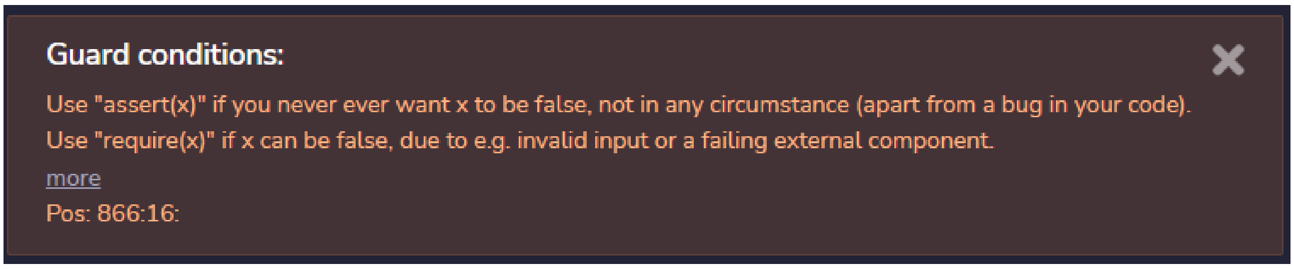
Constant/View/Pure functions:

Address.functionStaticCall(address,bytes) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 144:4:

Constant/View/Pure functions:

ThriveGenesisRewardPool.governanceRecoverUnsupported(contract IERC20,uint256,address) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 859:4:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

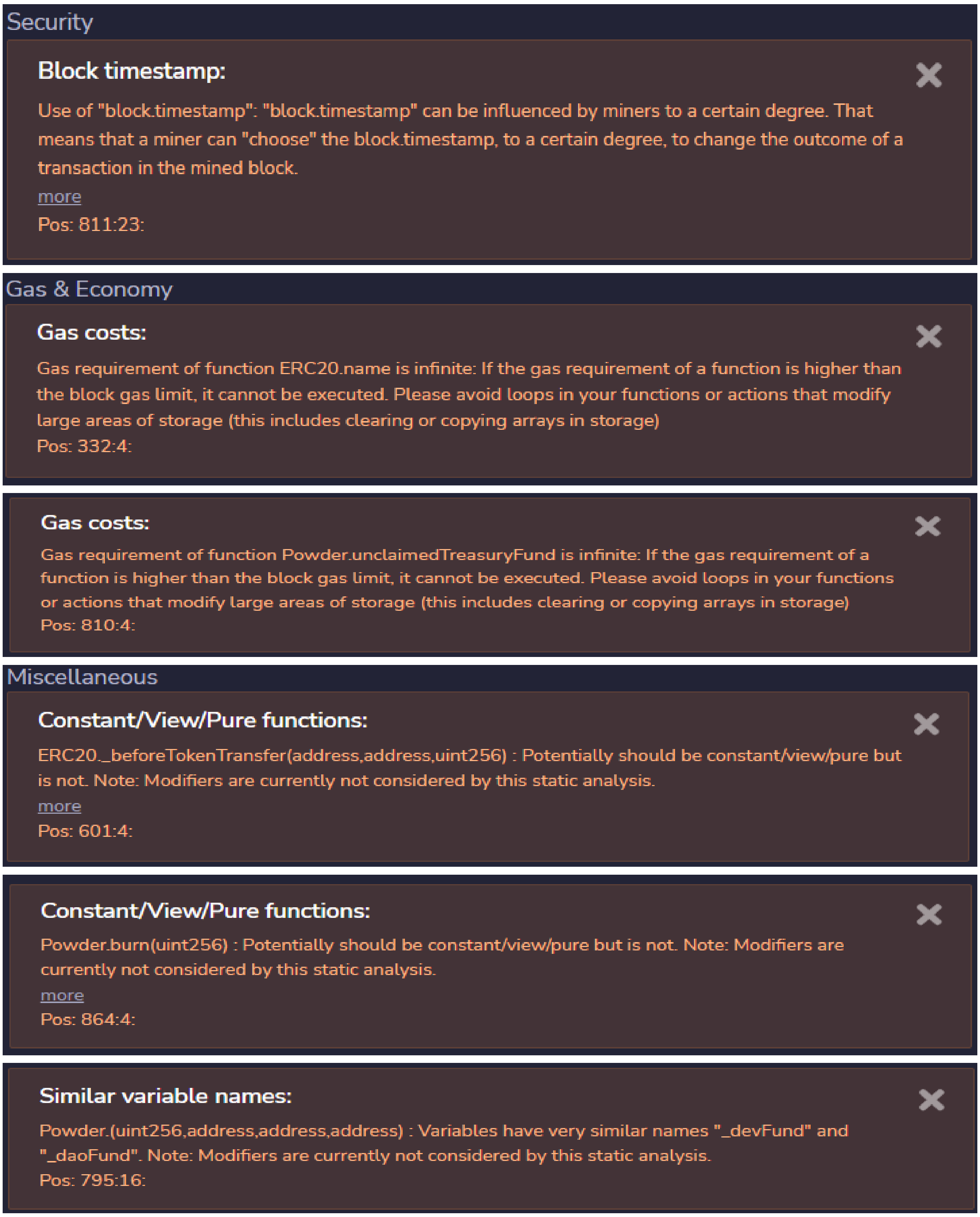

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 866:16:

Powder.sol


Security

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 811:23:

Gas & Economy

Gas costs:

Gas requirement of function ERC20.name is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 332:4:

Gas costs:

Gas requirement of function Powder.unclaimedTreasuryFund is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 810:4:

Miscellaneous

Constant/View/Pure functions:

ERC20._beforeTokenTransfer(address,address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 601:4:

Constant/View/Pure functions:

Powder.burn(uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

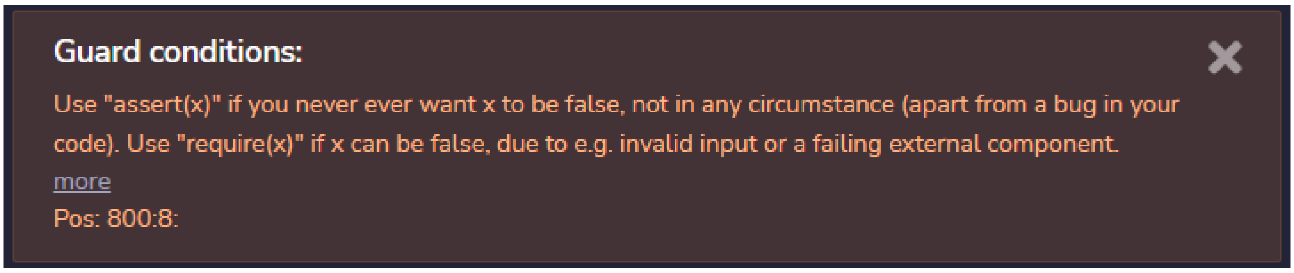
Pos: 864:4:

Similar variable names:

Powder.(uint256,address,address,address) : Variables have very similar names "_devFund" and "_daoFund". Note: Modifiers are currently not considered by this static analysis.

Pos: 795:16:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io



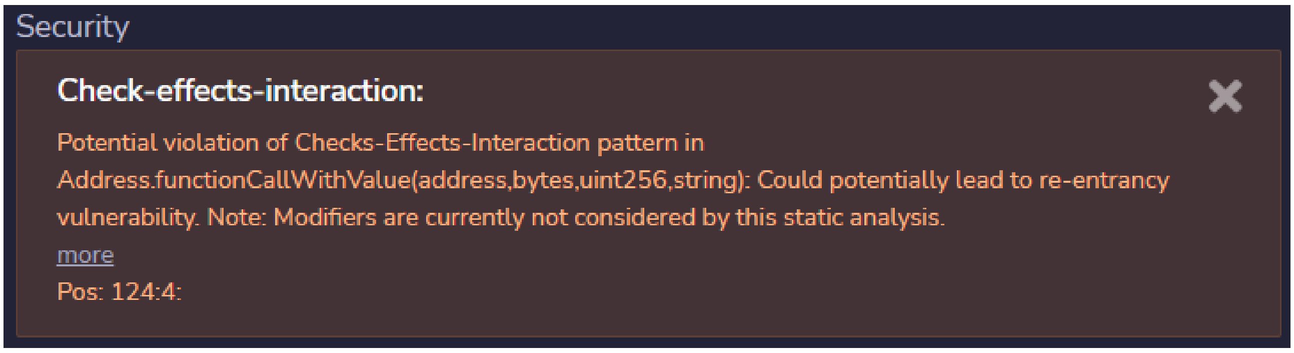
Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 800:8:

PowderRewardPool.sol



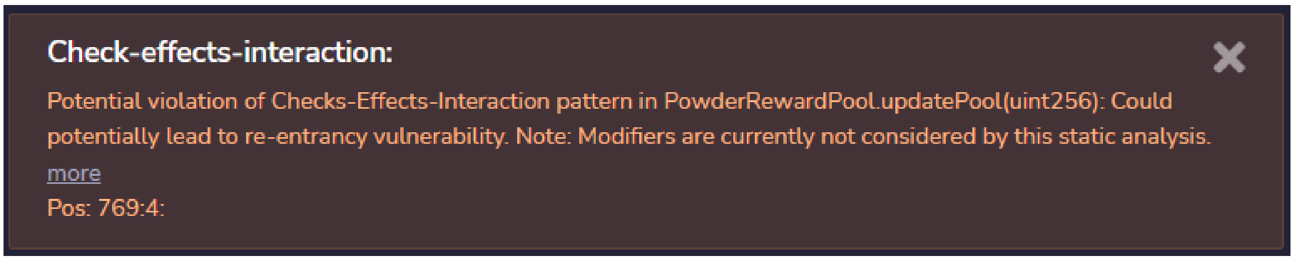
Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Address.functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 124:4:

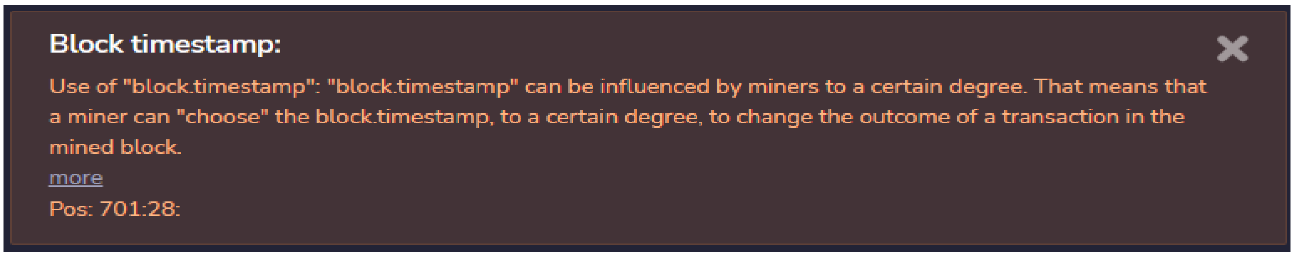


Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in PowderRewardPool.updatePool(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 769:4:

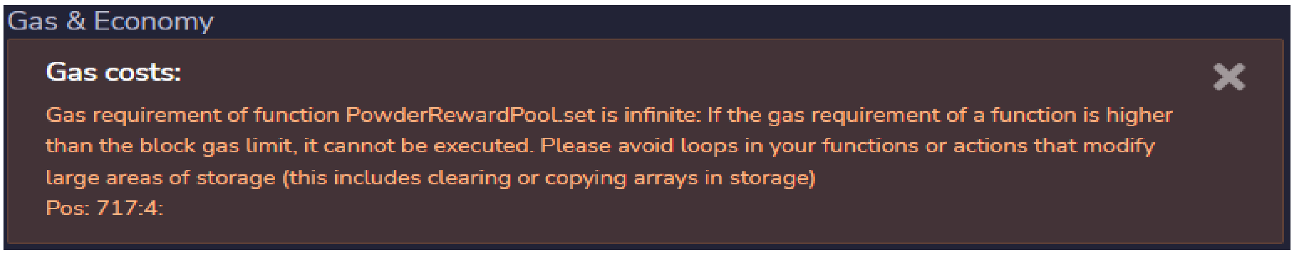


Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 701:28:

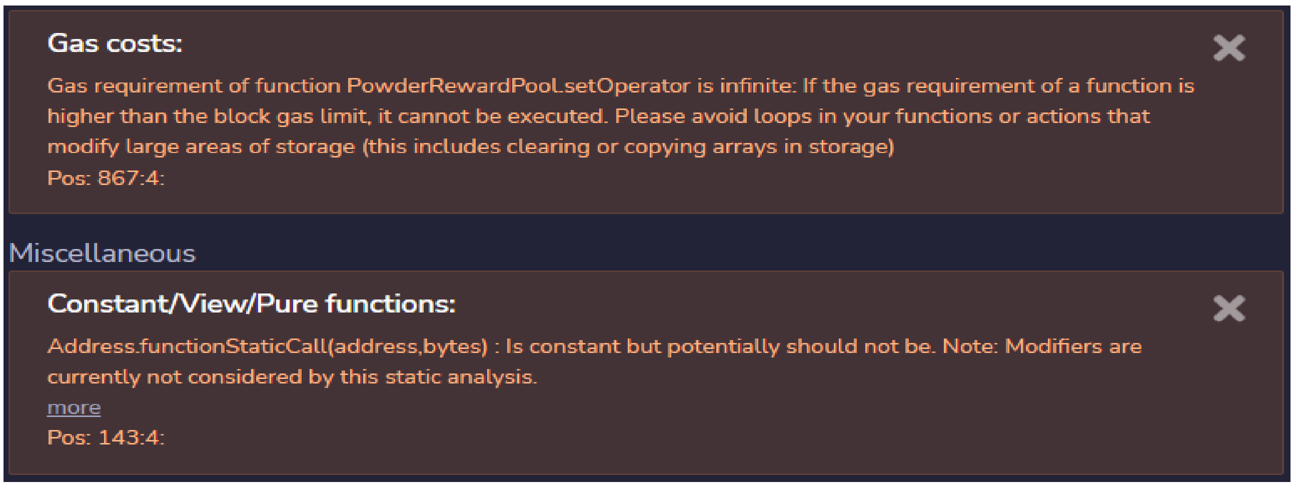


Gas & Economy

Gas costs:

Gas requirement of function PowderRewardPool.set is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 717:4:



Gas costs:

Gas requirement of function PowderRewardPool.setOperator is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 867:4:

Miscellaneous

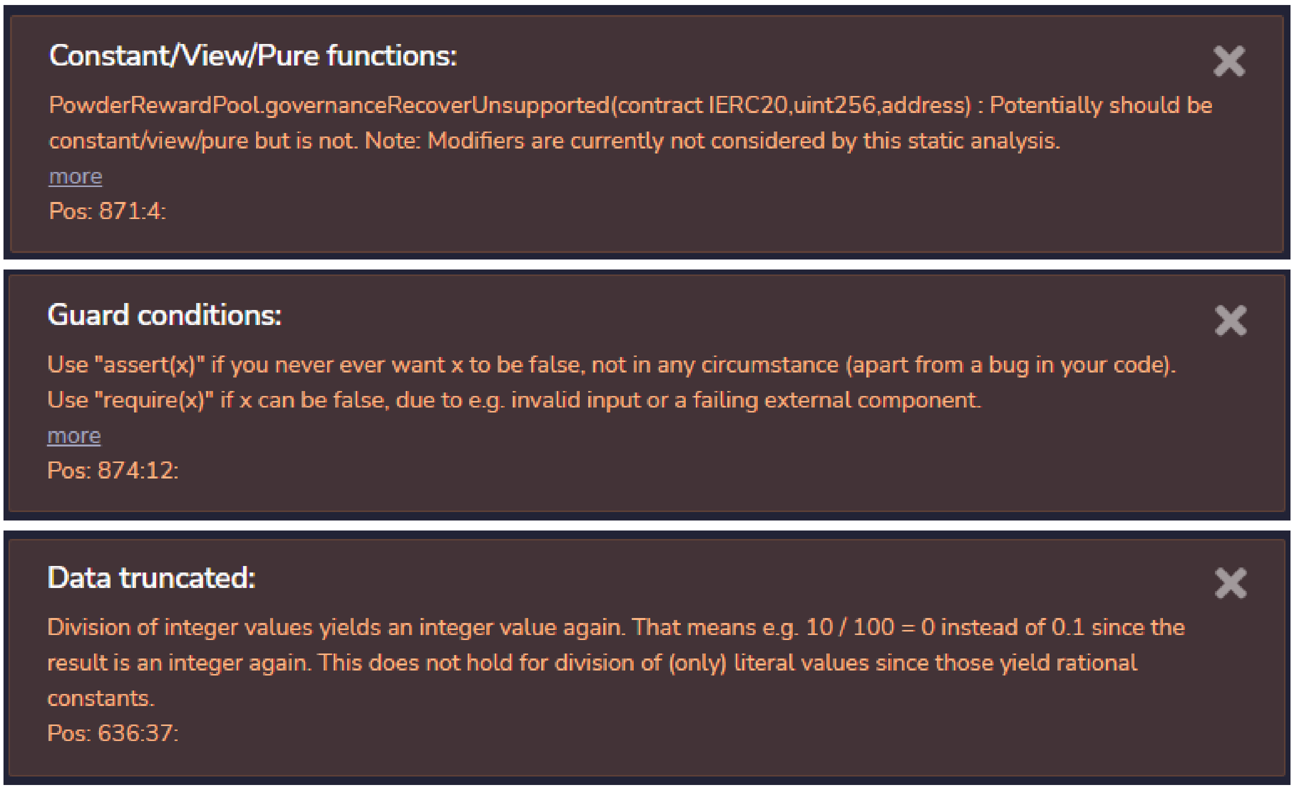
Constant/View/Pure functions:

Address.functionStaticCall(address,bytes) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 143:4:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io



Constant/View/Pure functions:

PowderRewardPool.governanceRecoverUnsupported(contract IERC20,uint256,address) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 871:4:

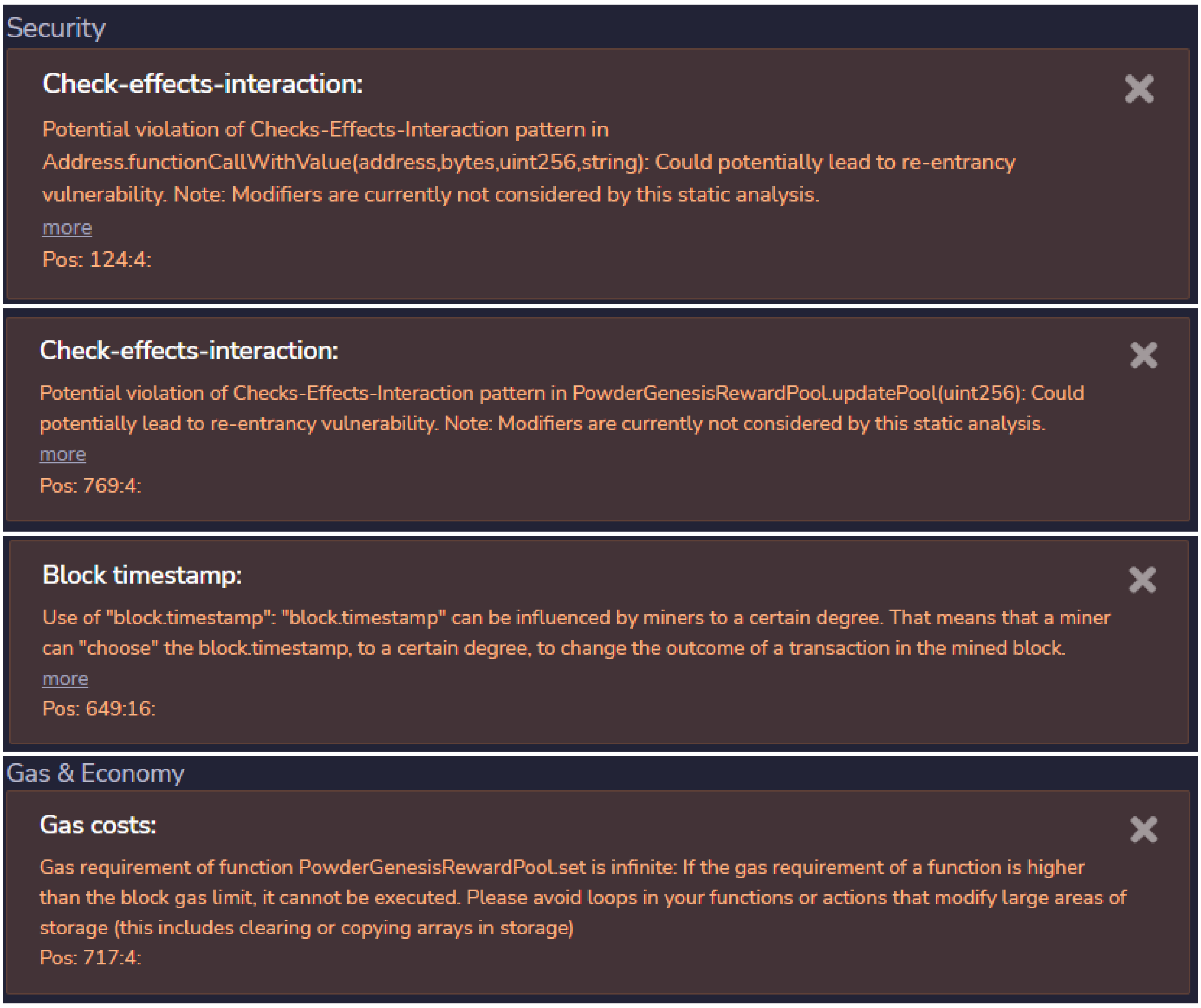
Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 874:12:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 636:37:

PowderGenesisRewardPool.sol



Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Address.functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 124:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in PowderGenesisRewardPool.updatePool(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 769:4:

Block timestamp:

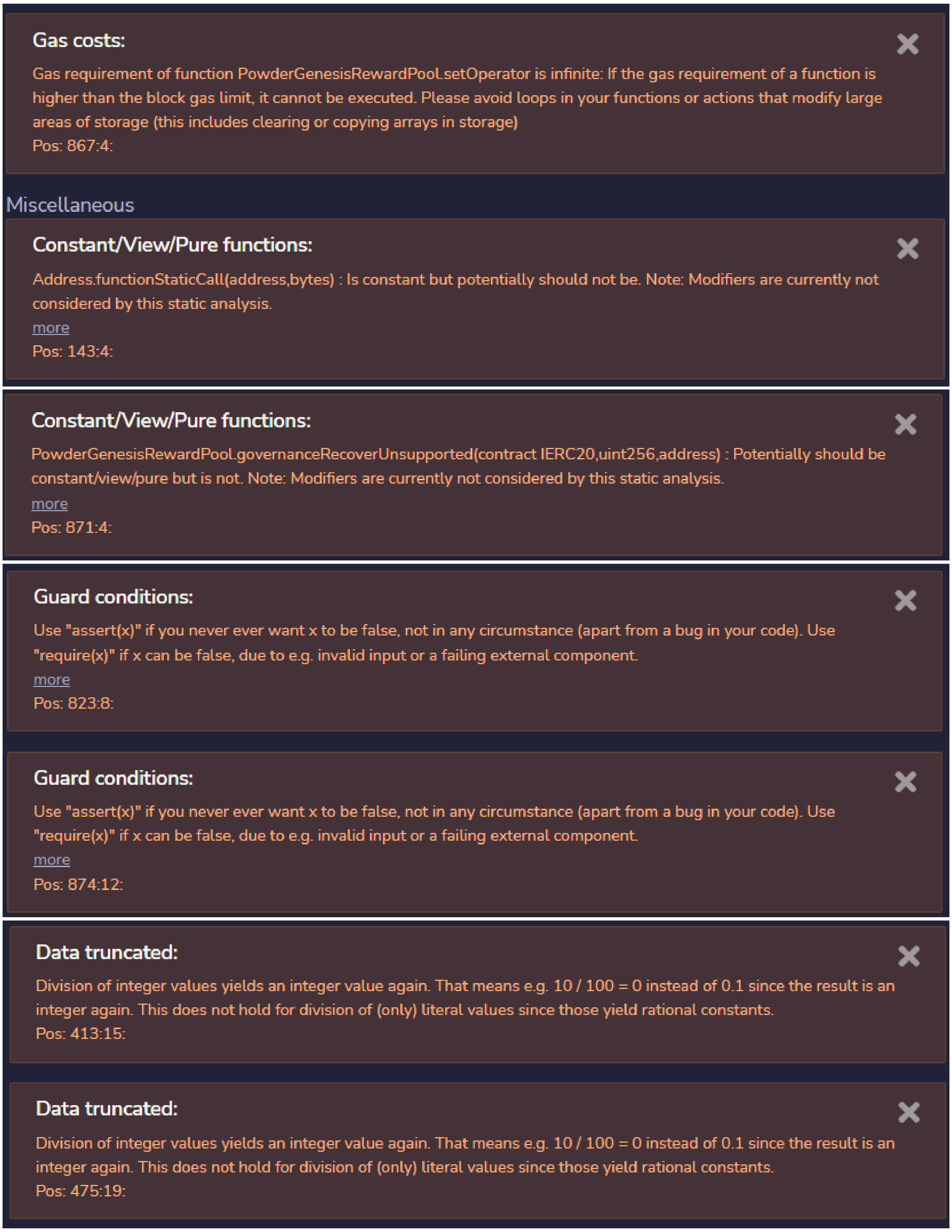
Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 649:16:

Gas & Economy

Gas costs:

Gas requirement of function PowderGenesisRewardPool.set is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 717:4:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io



Gas costs:

Gas requirement of function PowderGenesisRewardPool.setOperator is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 867:4:

Miscellaneous

Constant/View/Pure functions:

Address.functionStaticCall(address,bytes) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 143:4:

Constant/View/Pure functions:

PowderGenesisRewardPool.governanceRecoverUnsupported(contract IERC20,uint256,address) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 871:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 823:8:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 874:12:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 413:15:

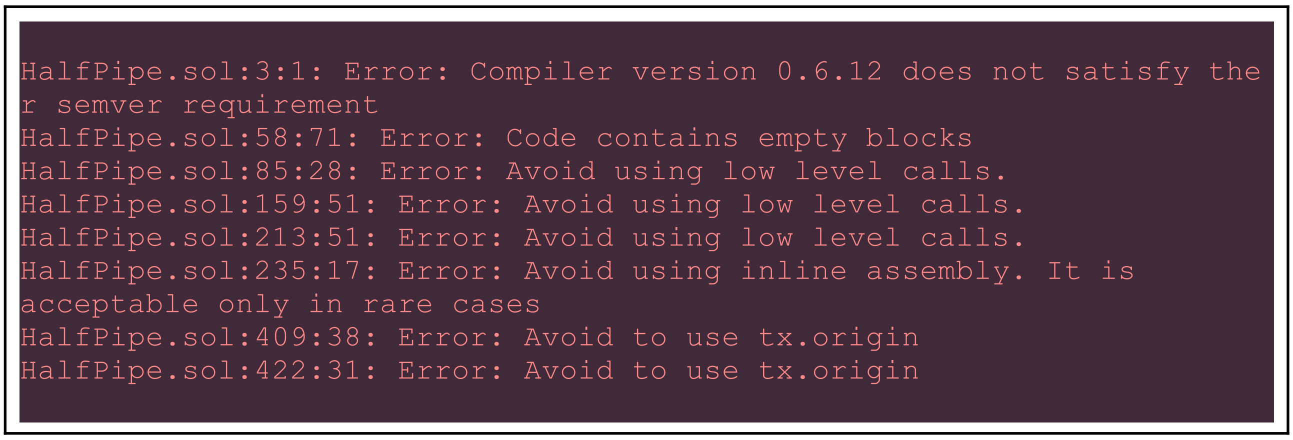
Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 475:19:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

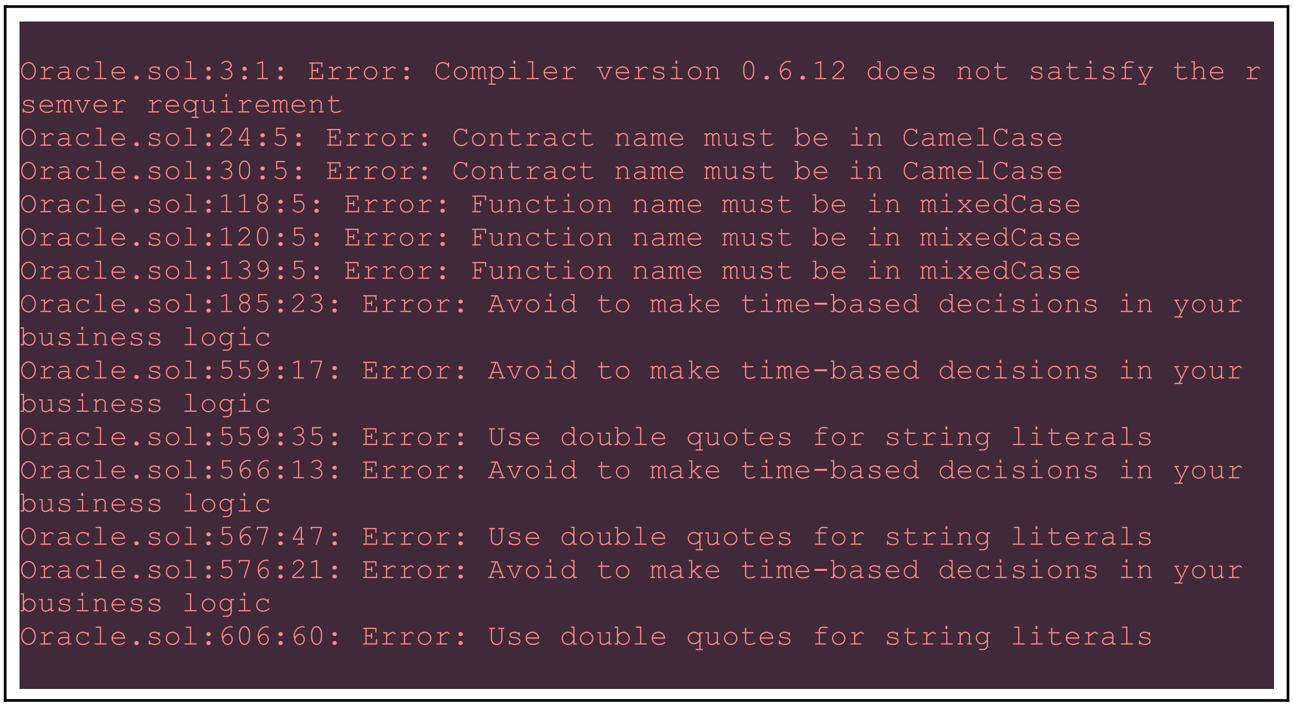
Solhint Linter

HalfPipe.sol



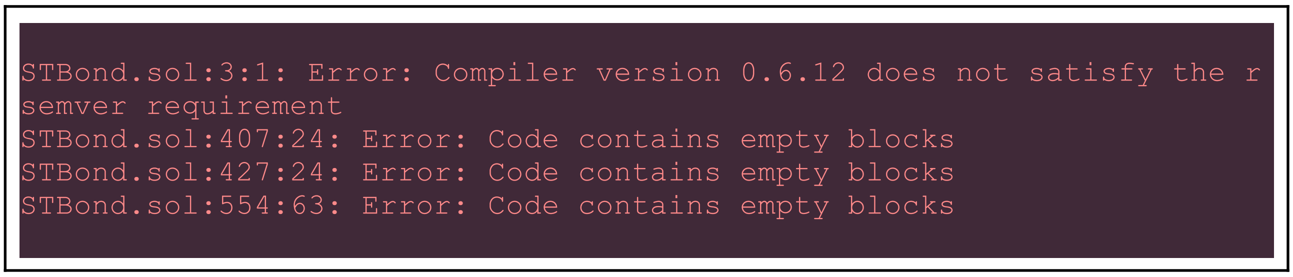
```
HalfPipe.sol:3:1: Error: Compiler version 0.6.12 does not satisfy the
r semver requirement
HalfPipe.sol:58:71: Error: Code contains empty blocks
HalfPipe.sol:85:28: Error: Avoid using low level calls.
HalfPipe.sol:159:51: Error: Avoid using low level calls.
HalfPipe.sol:213:51: Error: Avoid using low level calls.
HalfPipe.sol:235:17: Error: Avoid using inline assembly. It is
acceptable only in rare cases
HalfPipe.sol:409:38: Error: Avoid to use tx.origin
HalfPipe.sol:422:31: Error: Avoid to use tx.origin
```

Oracle.sol



```
Oracle.sol:3:1: Error: Compiler version 0.6.12 does not satisfy the r
semver requirement
Oracle.sol:24:5: Error: Contract name must be in CamelCase
Oracle.sol:30:5: Error: Contract name must be in CamelCase
Oracle.sol:118:5: Error: Function name must be in mixedCase
Oracle.sol:120:5: Error: Function name must be in mixedCase
Oracle.sol:139:5: Error: Function name must be in mixedCase
Oracle.sol:185:23: Error: Avoid to make time-based decisions in your
business logic
Oracle.sol:559:17: Error: Avoid to make time-based decisions in your
business logic
Oracle.sol:559:35: Error: Use double quotes for string literals
Oracle.sol:566:13: Error: Avoid to make time-based decisions in your
business logic
Oracle.sol:567:47: Error: Use double quotes for string literals
Oracle.sol:576:21: Error: Avoid to make time-based decisions in your
business logic
Oracle.sol:606:60: Error: Use double quotes for string literals
```

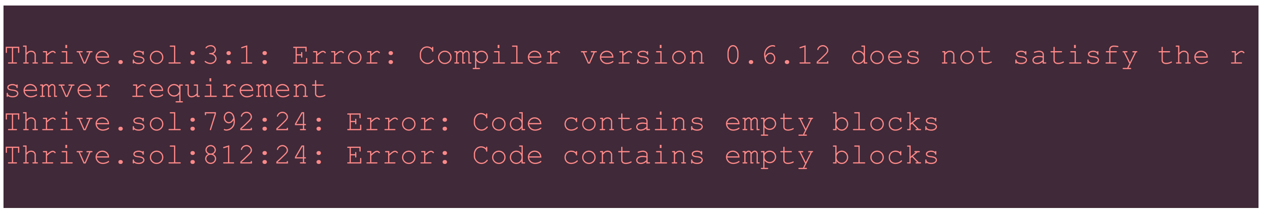
STBond.sol



```
STBond.sol:3:1: Error: Compiler version 0.6.12 does not satisfy the r
semver requirement
STBond.sol:407:24: Error: Code contains empty blocks
STBond.sol:427:24: Error: Code contains empty blocks
STBond.sol:554:63: Error: Code contains empty blocks
```

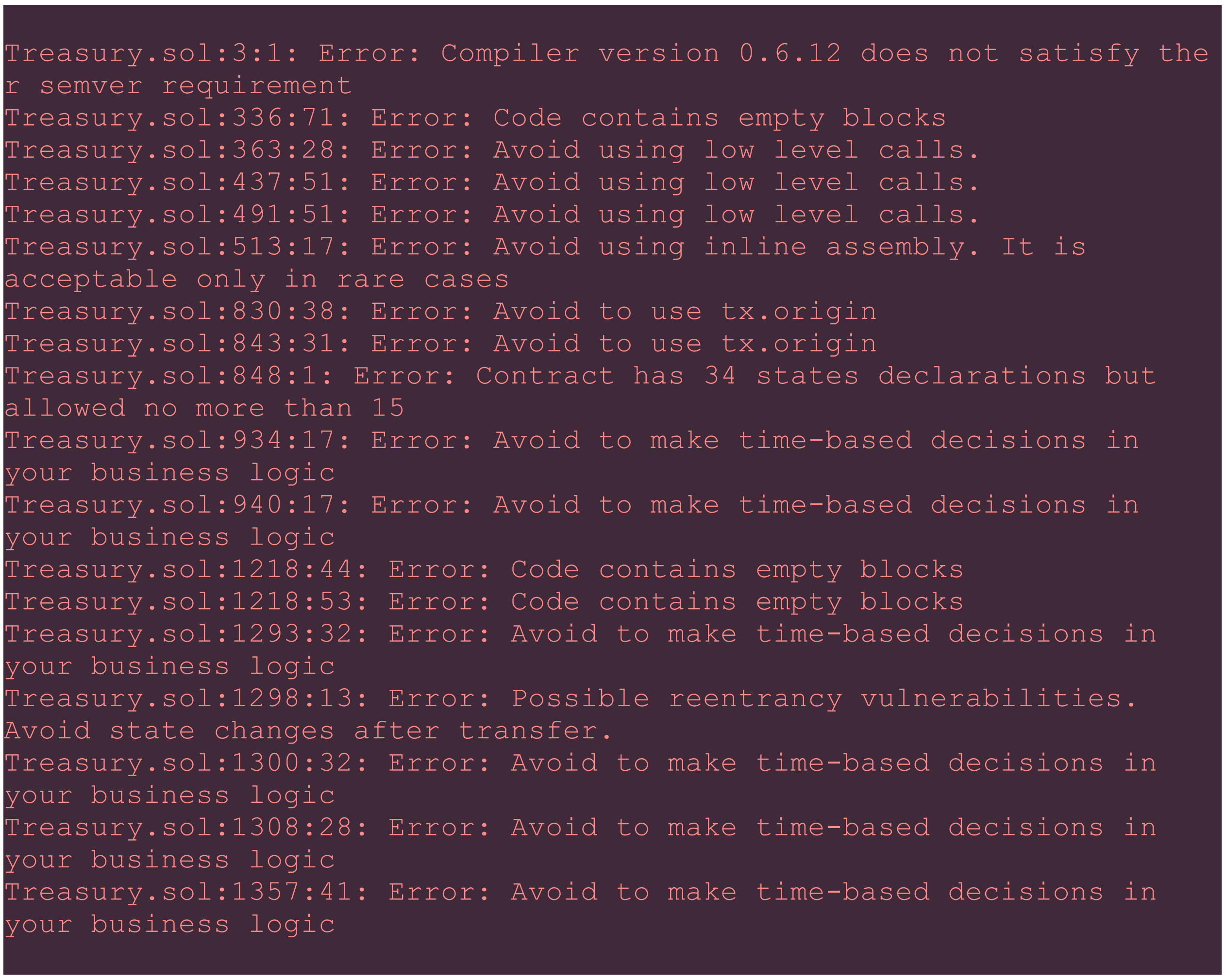
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Thrive.sol



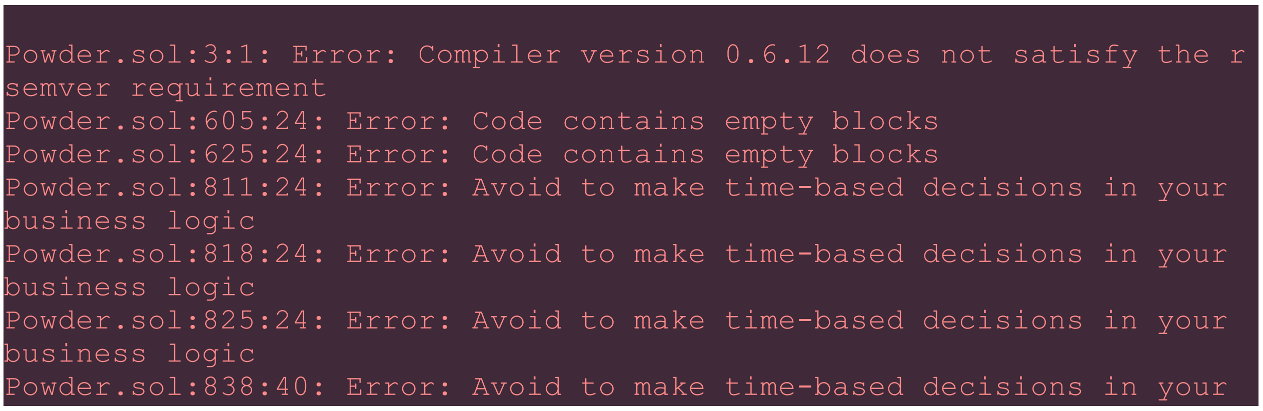
```
Thrive.sol:3:1: Error: Compiler version 0.6.12 does not satisfy the r
semver requirement
Thrive.sol:792:24: Error: Code contains empty blocks
Thrive.sol:812:24: Error: Code contains empty blocks
```

Treasury.sol



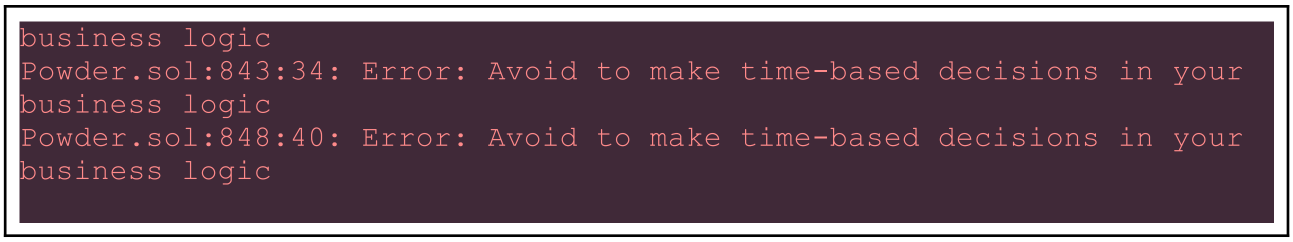
```
Treasury.sol:3:1: Error: Compiler version 0.6.12 does not satisfy the
r semver requirement
Treasury.sol:336:71: Error: Code contains empty blocks
Treasury.sol:363:28: Error: Avoid using low level calls.
Treasury.sol:437:51: Error: Avoid using low level calls.
Treasury.sol:491:51: Error: Avoid using low level calls.
Treasury.sol:513:17: Error: Avoid using inline assembly. It is
acceptable only in rare cases
Treasury.sol:830:38: Error: Avoid to use tx.origin
Treasury.sol:843:31: Error: Avoid to use tx.origin
Treasury.sol:848:1: Error: Contract has 34 states declarations but
allowed no more than 15
Treasury.sol:934:17: Error: Avoid to make time-based decisions in
your business logic
Treasury.sol:940:17: Error: Avoid to make time-based decisions in
your business logic
Treasury.sol:1218:44: Error: Code contains empty blocks
Treasury.sol:1218:53: Error: Code contains empty blocks
Treasury.sol:1293:32: Error: Avoid to make time-based decisions in
your business logic
Treasury.sol:1298:13: Error: Possible reentrancy vulnerabilities.
Avoid state changes after transfer.
Treasury.sol:1300:32: Error: Avoid to make time-based decisions in
your business logic
Treasury.sol:1308:28: Error: Avoid to make time-based decisions in
your business logic
Treasury.sol:1357:41: Error: Avoid to make time-based decisions in
your business logic
```

Powder.sol



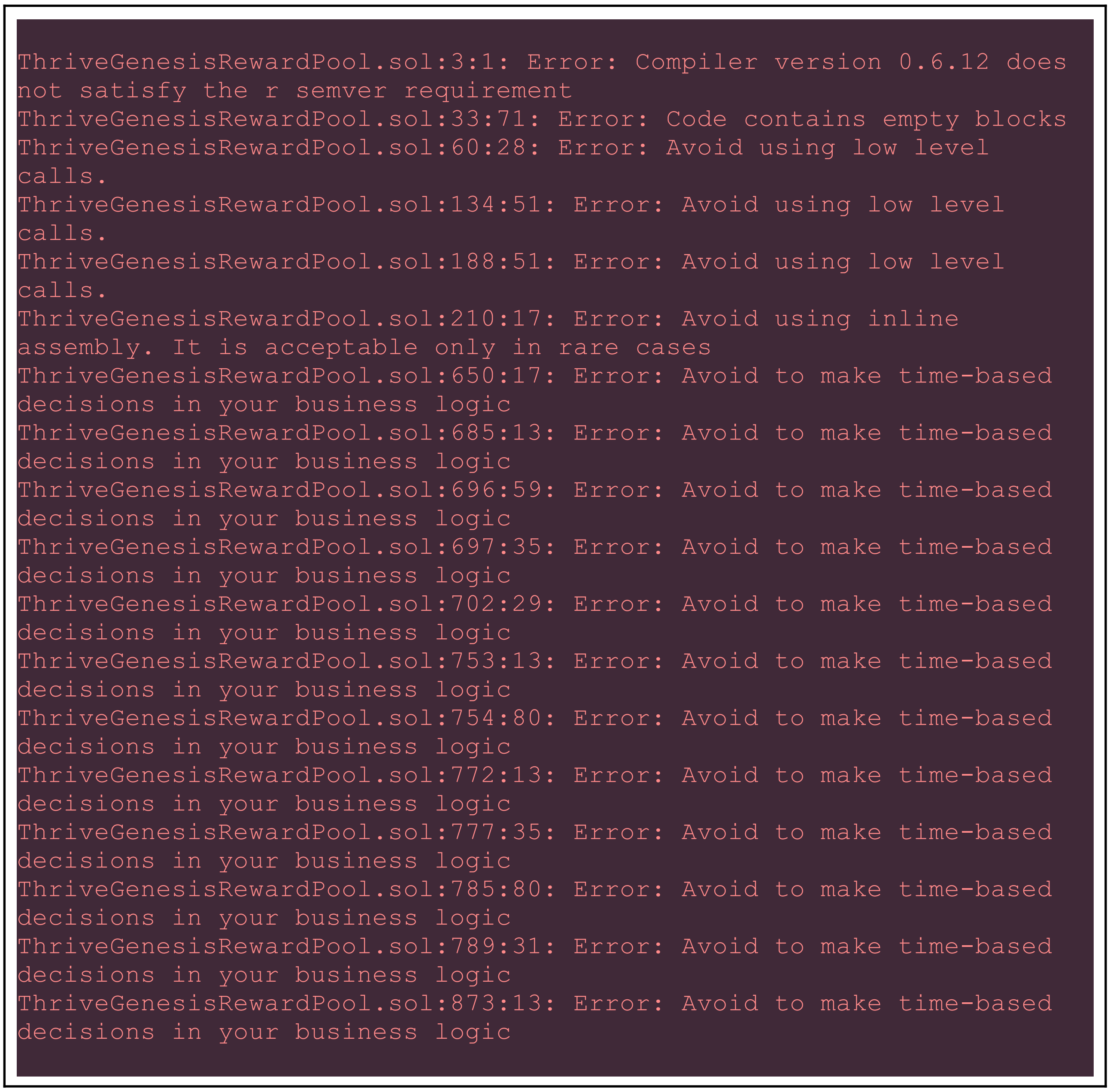
```
Powder.sol:3:1: Error: Compiler version 0.6.12 does not satisfy the r
semver requirement
Powder.sol:605:24: Error: Code contains empty blocks
Powder.sol:625:24: Error: Code contains empty blocks
Powder.sol:811:24: Error: Avoid to make time-based decisions in your
business logic
Powder.sol:818:24: Error: Avoid to make time-based decisions in your
business logic
Powder.sol:825:24: Error: Avoid to make time-based decisions in your
business logic
Powder.sol:838:40: Error: Avoid to make time-based decisions in your
```

This is a private and confidential document. No part of this document should
be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io



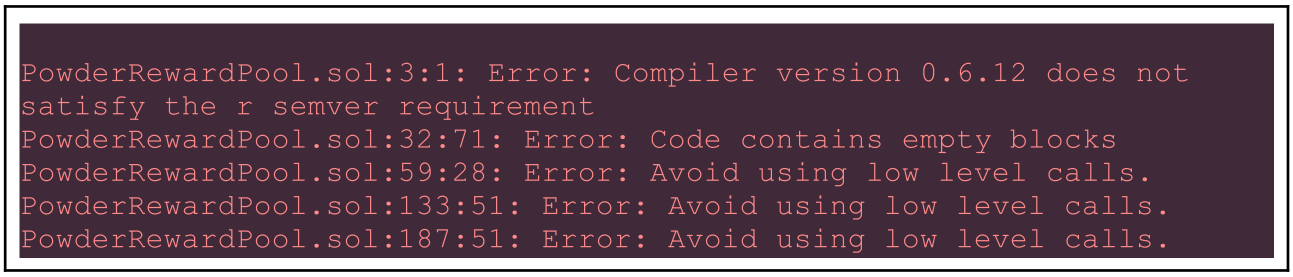
```
business logic
Powder.sol:843:34: Error: Avoid to make time-based decisions in your
business logic
Powder.sol:848:40: Error: Avoid to make time-based decisions in your
business logic
```

ThriveGenesisRewardPool.sol



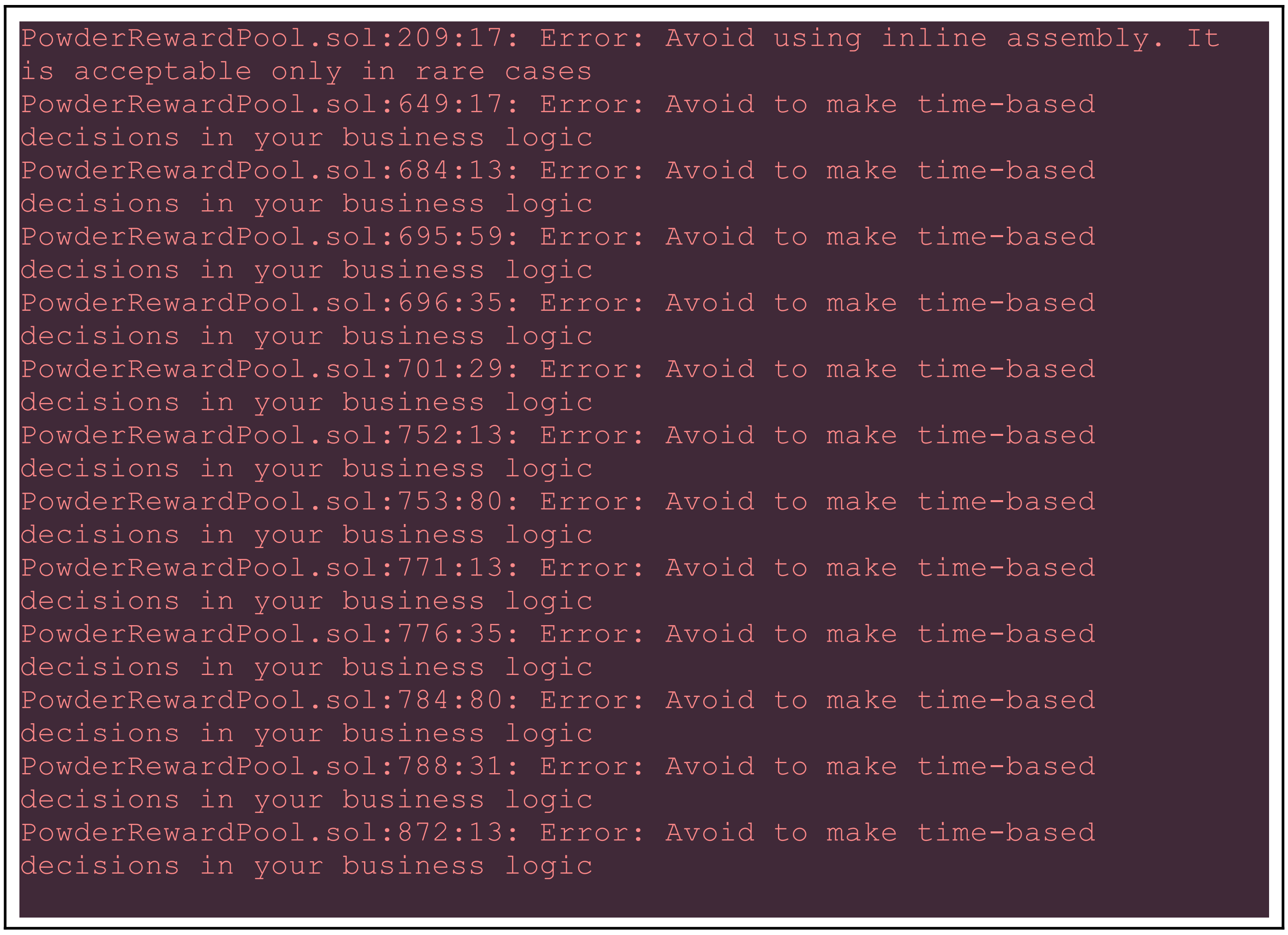
```
ThriveGenesisRewardPool.sol:3:1: Error: Compiler version 0.6.12 does
not satisfy the r semver requirement
ThriveGenesisRewardPool.sol:33:71: Error: Code contains empty blocks
ThriveGenesisRewardPool.sol:60:28: Error: Avoid using low level
calls.
ThriveGenesisRewardPool.sol:134:51: Error: Avoid using low level
calls.
ThriveGenesisRewardPool.sol:188:51: Error: Avoid using low level
calls.
ThriveGenesisRewardPool.sol:210:17: Error: Avoid using inline
assembly. It is acceptable only in rare cases
ThriveGenesisRewardPool.sol:650:17: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:685:13: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:696:59: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:697:35: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:702:29: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:753:13: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:754:80: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:772:13: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:777:35: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:785:80: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:789:31: Error: Avoid to make time-based
decisions in your business logic
ThriveGenesisRewardPool.sol:873:13: Error: Avoid to make time-based
decisions in your business logic
```

PowderRewardPool.sol



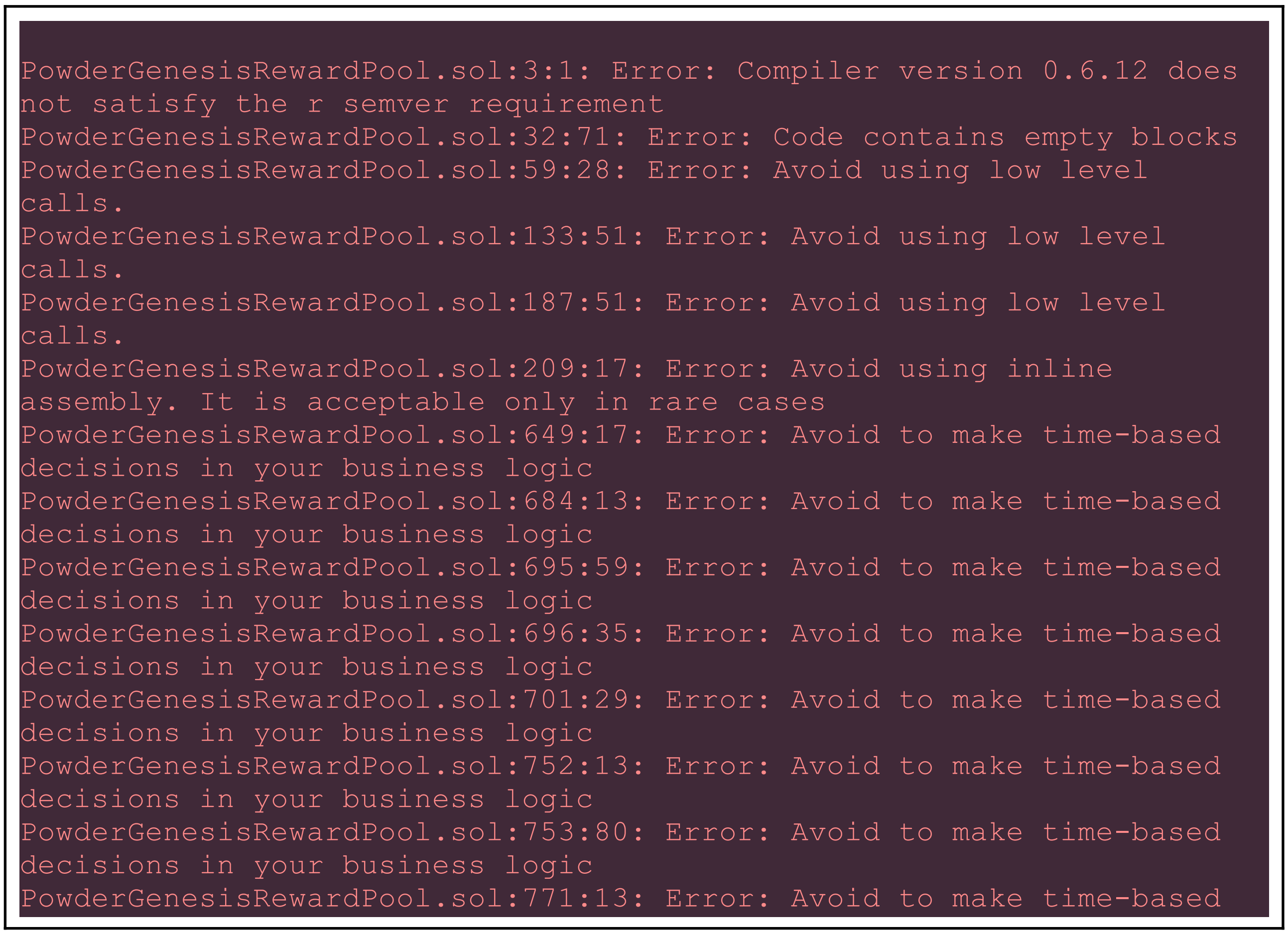
```
PowderRewardPool.sol:3:1: Error: Compiler version 0.6.12 does not
satisfy the r semver requirement
PowderRewardPool.sol:32:71: Error: Code contains empty blocks
PowderRewardPool.sol:59:28: Error: Avoid using low level calls.
PowderRewardPool.sol:133:51: Error: Avoid using low level calls.
PowderRewardPool.sol:187:51: Error: Avoid using low level calls.
```

This is a private and confidential document. No part of this document should
be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io



```
PowderRewardPool.sol:209:17: Error: Avoid using inline assembly. It
is acceptable only in rare cases
PowderRewardPool.sol:649:17: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:684:13: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:695:59: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:696:35: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:701:29: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:752:13: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:753:80: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:771:13: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:776:35: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:784:80: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:788:31: Error: Avoid to make time-based
decisions in your business logic
PowderRewardPool.sol:872:13: Error: Avoid to make time-based
decisions in your business logic
```

PowderGenesisRewardPool.sol



```
PowderGenesisRewardPool.sol:3:1: Error: Compiler version 0.6.12 does
not satisfy the r semver requirement
PowderGenesisRewardPool.sol:32:71: Error: Code contains empty blocks
PowderGenesisRewardPool.sol:59:28: Error: Avoid using low level
calls.
PowderGenesisRewardPool.sol:133:51: Error: Avoid using low level
calls.
PowderGenesisRewardPool.sol:187:51: Error: Avoid using low level
calls.
PowderGenesisRewardPool.sol:209:17: Error: Avoid using inline
assembly. It is acceptable only in rare cases
PowderGenesisRewardPool.sol:649:17: Error: Avoid to make time-based
decisions in your business logic
PowderGenesisRewardPool.sol:684:13: Error: Avoid to make time-based
decisions in your business logic
PowderGenesisRewardPool.sol:695:59: Error: Avoid to make time-based
decisions in your business logic
PowderGenesisRewardPool.sol:696:35: Error: Avoid to make time-based
decisions in your business logic
PowderGenesisRewardPool.sol:701:29: Error: Avoid to make time-based
decisions in your business logic
PowderGenesisRewardPool.sol:752:13: Error: Avoid to make time-based
decisions in your business logic
PowderGenesisRewardPool.sol:753:80: Error: Avoid to make time-based
decisions in your business logic
PowderGenesisRewardPool.sol:771:13: Error: Avoid to make time-based
```

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

```
decisions in your business logic
PowderGenesisRewardPool.sol:776:35: Error: Avoid to make time-based
decisions in your business logic
PowderGenesisRewardPool.sol:784:80: Error: Avoid to make time-based
decisions in your business logic
PowderGenesisRewardPool.sol:788:31: Error: Avoid to make time-based
decisions in your business logic
PowderGenesisRewardPool.sol:872:13: Error: Avoid to make time-based
decisions in your business logic
```

Software analysis result:

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

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io