Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project: Tosha Protocol
Website: https://tosha.io
Language: Solidity
Date: April 6th, 2022

Table of contents

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

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by the Tosha team to perform the Security audit of the Tosha 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 April 6th, 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

- Tosha.io is a decentralized multi-chain smart yield optimizer platform where users earn auto-compounded interest on their crypto investments.
- It aggregates farming pools from various DeFi projects that offer rewards when tokens are staked.
- Tosha IO automatically collects rewards and reinvests them periodically. This is accomplished by employing various strategies that aim to optimize and maximize the yield on the return.
- The Tosha Contracts have functions like setKeeper, setCallFee, setUnirouter, setVault, setStrategist, harvest, panic, etc.

Audit scope

Name	Code Review and Security Analysis Report for Tosha Protocol Smart Contracts
File 1	FeeManager.sol
File 1 MD5 Hash	8908949D80545F09A27DC569F93D2AA4
File 2	FeeManagerLP.sol
File 2 MD5 Hash	67AD8555D93261FA580927219F0254ED
File 3	LPTokenWrapper.sol

File 3 MD5 Hash	8E0E37FFBE9FF03E0AA074DA593367C0
File 4	StratManager.sol
File 4 MD5 Hash	0CA7C357109AE172232FE52297D78B9E
File 5	StratManagerLP.sol
File 5 MD5 Hash	D23CDA5B8238A73778BF7CECC6DCF0D2
File 6	StrategyCommonLP.sol
File 6 MD5 Hash	903D21AE10D725B04A9F89454D786C8F
File 7	StrategyDualLP.sol
File 7 MD5 Hash	9708E2F78341B3E0A7AD3F5E45294879
Updated File 7 MD5 Hash	0A0711465B404B230B689EFC553E7DA5
File 8	StrategyTosha.sol
File 8 MD5 Hash	7A67613E022416F5DC3D293D151CF385
File 9	ToshaVault.sol
File 9 MD5 Hash	88E1190FEE4048D2D945F88B48C21A4A
File 10	ToshaLPVault.sol
File 10 MD5 Hash	AF81B1378E86F5BCE64B582DE5F8B65F
File 11	Materchef.sol
File 11 MD5 Hash	727C3696ACF2C6CE8AD1CA6FC4F8A51A
Updated File 11 MD5 Hash	FF0933DF80E7D372C7ABE3051A58180A
File 12	Tosha.sol
File 12 MD5 Hash	AF81B1378E86F5BCE64B582DE5F8B65F
Updated File 12 MD5 Hash	529AD794392BD4B256D88149A9D3A788
Audit Date	April 6th,2022
Revise Audit Date	April 11th,2022

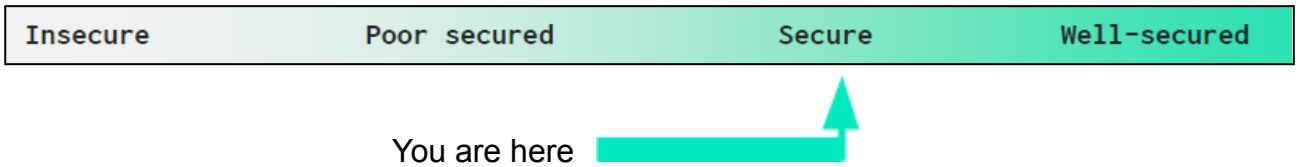
Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
File 1 FeeManager.sol • Maximum Fee Cap: 0.005% • Withdrawal Fee Cap: 0.005% • Withdrawal Fee: 0.0005% • Call Fee: 0.0005%	**YES, This is valid.**
File 2 FeeManagerLP.sol • Strategist Fee: 0.112% • Maximum Call Fee: 0.111% • Withdrawal Fee Cap: 0.005% • Withdrawal Fee: 0.001% • Call Fee: 0.111%	**YES, This is valid.**
File 3 LPTokenWrapper.sol • The LPTokenWeapper can access stake and withdraw a token.	**YES, This is valid.**
File 4 StratManager.sol • The StratManager can access functions like: setKeeper, setUnirouter, etc.	**YES, This is valid.**
File 5 StratManagerLP.sol • The StratManagerLP can access functions like: setStrategist, beforeDeposit, etc.	**YES, This is valid.**
File 6 StrategyCommonLP.sol • The StrategyCommonLP can access functions like: deposit, withdraw, harvest, etc.	**YES, This is valid.**
File 7 StrategyDualLP.sol • The StrategyDualLP can access functions like:harvest, managerHarvest, etc.	**YES, This is valid.**

File 8 StrategyTosha.sol • The StrategyTosha can access functions like: deposit, beforeDeposit, etc.	**YES, This is valid.**
File 9 ToshaVault.sol • Decimals: 18	**YES, This is valid.**
File 10 ToshaLPVault.sol • Decimals: 18.	**YES, This is valid.**
File 11 Materchef.sol • reserve funds: 10% • farming rewards: 90%	**YES, This is valid.** **Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**
File 12 Tosha.sol • Name: Tosha.IO • Symbol: TOSHA	**YES, This is valid.**

Audit Summary

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



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

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

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

Technical Quick Stats

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

Overall Audit Result: **PASSED**

Code Quality

This audit scope has 12 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 Tosha 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 Tosha Protocol.

The Tosha 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 Tosha Protocol smart contract code in the form of a Github Web Link. The hash of that code is mentioned above in the table.

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

Use of Dependencies

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

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

AS-IS overview

FeeManager.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	setCallFee	write	access only Manager	No Issue
3	setWithdrawalFee	write	access only Manager	No Issue
4	onlyManager	modifier	Passed	No Issue
5	setKeeper	external	access only Manager	No Issue
6	setUnirouter	external	access only Owner	No Issue
7	setVault	external	access only Owner	No Issue
8	beforeDeposit	external	Passed	No Issue
9	owner	read	Passed	No Issue
10	onlyOwner	modifier	Passed	No Issue
11	renounceOwnership	write	access only Owner	No Issue
12	transferOwnership	write	access only Owner	No Issue
13	paused	read	Passed	No Issue
14	whenNotPaused	modifier	Passed	No Issue
15	whenPaused	modifier	Passed	No Issue
16	_pause	internal	Passed	No Issue
17	_unpause	internal	Passed	No Issue

FeeManagerLP.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	onlyManager	modifier	Passed	No Issue
3	setKeeper	external	access only Manager	No Issue
4	setUnirouter	external	access only Owner	No Issue
5	setVault	external	access only Owner	No Issue
6	beforeDeposit	external	Passed	No Issue
7	owner	read	Passed	No Issue
8	onlyOwner	modifier	Passed	No Issue
9	renounceOwnership	write	access only Owner	No Issue
10	transferOwnership	write	access only Owner	No Issue
11	paused	read	Passed	No Issue
12	whenNotPaused	modifier	Passed	No Issue
13	whenPaused	modifier	Passed	No Issue
14	_pause	internal	Passed	No Issue
15	_unpause	internal	Passed	No Issue
16	setCallFee	write	access only Manager	No Issue

17	setWithdrawalFee	write	access only Manager	No Issue

LPTokenWrapper.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	totalSupply	read	Passed	No Issue
3	balanceOf	read	Passed	No Issue
4	stakeToken	write	Passed	No Issue
5	withdrawToken	write	Passed	No Issue

StratManager.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	paused	read	Passed	No Issue
7	whenNotPaused	modifier	Passed	No Issue
8	whenPaused	modifier	Passed	No Issue
9	_pause	internal	Passed	No Issue
10	_unpause	internal	Passed	No Issue
11	onlyManager	modifier	Passed	No Issue
12	setKeeper	external	access only Manager	No Issue
13	setUnirouter	external	access only Owner	No Issue
14	setVault	external	access only Owner	No Issue
15	beforeDeposit	external	Passed	No Issue

StratManagerLP.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	paused	read	Passed	No Issue
7	whenNotPaused	modifier	Passed	No Issue

8	whenPaused	modifier	Passed	No Issue
9	_pause	internal	Passed	No Issue
10	_unpause	internal	Passed	No Issue
11	onlyManager	modifier	Passed	No Issue
12	setKeeper	external	access only Manager	No Issue
13	setStrategist	external	Passed	No Issue
14	setUnirouter	external	access only Owner	No Issue
15	setVault	external	access only Owner	No Issue
16	setToshaFeeRecipient	external	access only Owner	No Issue
17	beforeDeposit	external	Passed	No Issue

StrategyCommonLP.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	gasThrottle	modifier	Passed	No Issue
3	owner	read	Passed	No Issue
4	onlyOwner	modifier	Passed	No Issue
5	renounceOwnership	write	access only Owner	No Issue
6	transferOwnership	write	access only Owner	No Issue
7	paused	read	Passed	No Issue
8	whenNotPaused	modifier	Passed	No Issue
9	whenPaused	modifier	Passed	No Issue
10	_unpause	internal	Passed	No Issue
11	_pause	internal	Passed	No Issue
12	_unpause	internal	Passed	No Issue
13	onlyManager	modifier	Passed	No Issue
14	setKeeper	external	access only Manager	No Issue
15	setUnirouter	external	access only Owner	No Issue
16	setStrategist	external	Passed	No Issue
17	setVault	external	access only Owner	No Issue
18	setToshaFeeRecipient	external	access only Owner	No Issue
19	beforeDeposit	external	Passed	No Issue
20	setCallFee	write	access only Manager	No Issue
21	setWithdrawalFee	write	access only Manager	No Issue
22	deposit	write	Passed	No Issue
23	withdraw	external	Passed	No Issue
24	beforeDeposit	external	Passed	No Issue
25	harvest	external	Passed	No Issue
26	harvestWithCallFeeRecipient	external	Passed	No Issue
27	managerHarvest	external	access only Manager	No Issue

28	_harvest	internal	Passed	No Issue
29	chargeFees	internal	Passed	No Issue
30	_deposit	internal	Passed	No Issue
31	totalStake	read	Passed	No Issue
32	balanceOf	read	Passed	No Issue
33	balanceOfWant	read	Passed	No Issue
34	balanceOfPool	read	Passed	No Issue
35	retireStrat	external	Passed	No Issue
36	setPendingRewardsFunctionName	external	access only Manager	No Issue
37	rewardsAvailable	read	Passed	No Issue
38	callReward	read	Passed	No Issue
39	setHarvestOnDeposit	external	access only Manager	No Issue
40	setShouldGasThrottle	external	access only Manager	No Issue
41	panic	write	access only Manager	No Issue
42	pause	write	access only Manager	No Issue
43	unpause	external	access only Manager	No Issue
44	_giveAllowances	internal	Passed	No Issue
45	_removeAllowances	internal	Passed	No Issue

StrategyDualLP.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	gasThrottle	modifier	Passed	No Issue
3	owner	read	Passed	No Issue
4	onlyOwner	modifier	Passed	No Issue
5	renounceOwnership	write	access only Owner	No Issue
6	transferOwnership	write	access only Owner	No Issue
7	paused	read	Passed	No Issue
8	whenNotPaused	modifier	Passed	No Issue
9	whenPaused	modifier	Passed	No Issue
10	_unpause	internal	Passed	No Issue
11	_pause	internal	Passed	No Issue
12	_unpause	internal	Passed	No Issue
13	onlyManager	modifier	Passed	No Issue
14	setKeeper	external	access only Manager	No Issue
15	setUnirouter	external	access only Owner	No Issue
16	setStrategist	external	Passed	No Issue
17	setVault	external	access only Owner	No Issue

18	setToshaFeeRecipient	external	access only Owner	No Issue
19	beforeDeposit	external	Passed	No Issue
20	setCallFee	write	access only Manager	No Issue
21	setWithdrawalFee	write	access only Manager	No Issue
22	deposit	write	Passed	No Issue
23	withdraw	external	Passed	No Issue
24	beforeDeposit	external	Passed	No Issue
25	harvest	external	Passed	No Issue
26	harvestWithCallFeeRecipient	external	Passed	No Issue
27	managerHarvest	external	access only Manager	No Issue
28	_harvest	internal	Passed	No Issue
29	chargeFees	internal	Passed	No Issue
30	_deposit	internal	Passed	No Issue
31	totalStake	read	Passed	No Issue
32	balanceOf	read	Passed	No Issue
33	balanceOfWant	read	Passed	No Issue
34	balanceOfPool	read	Passed	No Issue
35	retireStrat	external	Passed	No Issue
36	setPendingRewardsFunctionName	external	access only Manager	No Issue
37	rewardsAvailable	read	Passed	No Issue
38	callReward	read	Passed	No Issue
39	setHarvestOnDeposit	external	access only Manager	No Issue
40	setShouldGasThrottle	external	access only Manager	No Issue
41	panic	external	access only Manager	No Issue
42	pause	external	access only Manager	No Issue
43	unpause	external	access only Manager	No Issue
44	_giveAllowances	internal	Passed	No Issue
45	_removeAllowances	internal	Passed	No Issue

StrategyTosha.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	setCallFee	write	access only Manager	No Issue
3	setWithdrawalFee	write	access only Manager	No Issue

4	onlyManager	modifier	Passed	No Issue
5	setKeeper	external	access only Manager	No Issue
6	setUnirouter	external	access only Owner	No Issue
7	setVault	external	access only Owner	No Issue
8	beforeDeposit	external	Passed	No Issue
9	owner	read	Passed	No Issue
10	onlyOwner	modifier	Passed	No Issue
11	renounceOwnership	write	access only Owner	No Issue
12	transferOwnership	write	access only Owner	No Issue
13	paused	read	Passed	No Issue
14	whenNotPaused	modifier	Passed	No Issue
15	whenPaused	modifier	Passed	No Issue
16	_pause	internal	Passed	No Issue
17	_unpause	internal	Passed	No Issue
18	deposit	write	Passed	No Issue
19	withdraw	external	Passed	No Issue
20	beforeDeposit	external	Passed	No Issue
21	harvest	external	Passed	No Issue
22	harvest	external	Passed	No Issue
23	_harvest	internal	Passed	No Issue
24	chargeFees	internal	Passed	No Issue
25	swapRewards	internal	Passed	No Issue
26	balanceOf	read	Passed	No Issue
27	balanceOfWant	read	Passed	No Issue
28	balanceOfPool	read	Passed	No Issue
29	rewardsAvailable	read	Passed	No Issue
30	callReward	read	Passed	No Issue
31	setHarvestOnDeposit	external	access only Manager	No Issue
32	retireStrat	external	Passed	No Issue
33	panic	write	access only Manager	No Issue
34	pause	write	access only Manager	No Issue
35	unpause	external	access only Manager	No Issue
36	_giveAllowances	internal	Passed	No Issue
37	_removeAllowances	internal	Passed	No Issue
38	outputToWant	external	Passed	No Issue

ToshaVault.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	name	read	Passed	No Issue
7	symbol	read	Passed	No Issue
8	decimals	read	Passed	No Issue
9	totalSupply	read	Passed	No Issue
10	balanceOf	read	Passed	No Issue
11	transfer	write	Passed	No Issue
12	allowance	read	Passed	No Issue
13	approve	write	Passed	No Issue
14	transferFrom	write	Passed	No Issue
15	increaseAllowance	write	Passed	No Issue
16	decreaseAllowance	write	Passed	No Issue
17	_transfer	internal	Passed	No Issue
18	_mint	internal	Passed	No Issue
19	_burn	internal	Passed	No Issue
20	_approve	internal	Passed	No Issue
21	_setupDecimals	internal	Passed	No Issue
22	_beforeTokenTransfer	internal	Passed	No Issue
23	want	read	Passed	No Issue
24	balance	read	Passed	No Issue
25	available	read	Passed	No Issue
26	getPricePerFullShare	read	Passed	No Issue
27	depositAll	external	Passed	No Issue
28	deposit	write	Passed	No Issue
29	earn	write	Passed	No Issue
30	withdrawAll	external	Passed	No Issue
31	withdraw	write	Passed	No Issue
32	proposeStrat	write	access only Owner	No Issue
33	upgradeStrat	write	access only Owner	No Issue
34	inCaseTokensGetStuck	external	access only Owner	No Issue

ToshaLPVault.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue

5	transferOwnership	write	access only Owner	No Issue
6	name	read	Passed	No Issue
7	symbol	read	Passed	No Issue
8	decimals	read	Passed	No Issue
9	totalSupply	read	Passed	No Issue
10	balanceOf	read	Passed	No Issue
11	transfer	write	Passed	No Issue
12	allowance	read	Passed	No Issue
13	approve	write	Passed	No Issue
14	transferFrom	write	Passed	No Issue
15	increaseAllowance	write	Passed	No Issue
16	decreaseAllowance	write	Passed	No Issue
17	_transfer	internal	Passed	No Issue
18	_mint	internal	Passed	No Issue
19	_burn	internal	Passed	No Issue
20	_approve	internal	Passed	No Issue
21	_setupDecimals	internal	Passed	No Issue
22	_beforeTokenTransfer	internal	Passed	No Issue
23	want	read	Passed	No Issue
24	balance	read	Passed	No Issue
25	available	read	Passed	No Issue
26	getPricePerFullShare	read	Passed	No Issue
27	depositAll	external	Passed	No Issue
28	deposit	write	Passed	No Issue
29	earn	write	Passed	No Issue
30	withdrawAll	external	Passed	No Issue
31	withdraw	write	Passed	No Issue
32	_getPercent	write	Passed	No Issue
33	_claimRewards	write	Passed	No Issue
34	_safeCoreTransfer	write	Passed	No Issue
35	notifyRewards	external	Passed	No Issue
36	totalAutoCoreShares	read	Passed	No Issue
37	_coreBalance	read	Passed	No Issue
38	proposeStrat	write	access only Owner	No Issue
39	upgradeStrat	write	access only Owner	No Issue
40	inCaseTokensGetStuck	external	access only Owner	No Issue

MasterChef.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	name	read	Passed	No Issue

7	symbol	read	Passed	No Issue
8	decimals	read	Passed	No Issue
9	totalSupply	read	Passed	No Issue
10	balanceOf	read	Passed	No Issue
11	transfer	write	Passed	No Issue
12	allowance	read	Passed	No Issue
13	approve	write	Passed	No Issue
14	transferFrom	write	Passed	No Issue
15	increaseAllowance	write	Passed	No Issue
16	decreaseAllowance	write	Passed	No Issue
17	_transfer	internal	Passed	No Issue
18	_mint	internal	Passed	No Issue
19	_burn	internal	Passed	No Issue
20	_approve	internal	Passed	No Issue
21	_setupDecimals	internal	Passed	No Issue
22	_beforeTokenTransfer	internal	Passed	No Issue
23	mint	write	Passed	No Issue
24	setGovernance	write	Passed	No Issue
25	addMinter	write	Passed	No Issue
26	removeMinter	write	Passed	No Issue
27	harvest	write	Passed	No Issue
28	updateReserveFundsAddress	write	access only Owner	No Issue
29	updateFarmingRewarderAddress	write	access only Owner	No Issue
30	updateEmissionRate	write	access only Owner	No Issue
31	updateRewardsRate	write	access only Owner	No Issue

Tosha.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	mint	write	access only Owner	No Issue
3	setGovernance	write	Passed	No Issue
4	addMinter	write	Passed	No Issue
5	removeMinter	write	Passed	No Issue
6	name	read	Passed	No Issue
7	symbol	read	Passed	No Issue
8	decimals	read	Passed	No Issue
9	totalSupply	read	Passed	No Issue
10	balanceOf	read	Passed	No Issue
11	transfer	write	Passed	No Issue
12	allowance	read	Passed	No Issue
13	approve	write	Passed	No Issue
14	transferFrom	write	Passed	No Issue
15	increaseAllowance	write	Passed	No Issue

16	decreaseAllowance	write	Passed	No Issue
17	_transfer	internal	Passed	No Issue
18	_mint	internal	Passed	No Issue
19	_burn	internal	Passed	No Issue
20	_approve	internal	Passed	No Issue
21	_setupDecimals	internal	Passed	No Issue
22	_beforeTokenTransfer	internal	Passed	No Issue

Severity Definitions

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

Email: audit@EtherAuthority.io

Audit Findings

Critical Severity

No Critical severity vulnerabilities were found.

High Severity

No High severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

(1) Owner can mint unlimited tokens: **Tosha.sol**
There is no limit for minting TOSHA tokens. Thus the owner can mint unlimited tokens to any account.

Resolution: There should be a limit for minting or need to confirm, if it is a part of the plan then disregard this issue.
Status: Fixed

Very Low / Informational / Best practices:

(1) Use latest solidity version: **ToshaVault.sol, MasterChef.sol, FeeManager.sol, FeeManagerLP.sol, StratManager.sol, StratManagerLP.sol, LPTokenWrapper.sol, StrategyCommonLP.sol, StrategyDualLP.sol, StrategyTosha.sol, ToshaLPVault.sol, Tosha.sol**

```
pragma solidity 0.6.12;
```

Using the latest solidity will prevent any compiler level bugs.

Resolution: We suggest using version > 0.8.0.
Status: Acknowledged

(2) Unused event / variable:

MasterChef.sol

Event - Harvest

StratManager.sol

Variables - strategist

Resolution: We suggest removing the unused events and variables. Harvest event can be used in the harvest function.

Status: Fixed

(3) Same contract name: **StrategyDualLP.sol**

StrategyDualLP and StrategyCommonLP contract files have the same contract name in both files - “StrategyCommonLP”.

Resolution: We suggest giving the appropriate name to the contract to identify them better and use them.

Status: Fixed

(4) If condition can be replaced by require: **MasterChef.sol**

```
function harvest() public {

    if (block.number > lastRewardBlock) {
        uint256 blocksSinceLastReward = block.number - lastRewardBlock;

        // rewards for these many blocks
        uint256 tokenRewards = blocksSinceLastReward * tokensPerBlock;
```

In the harvest() function the execution is only working if the condition is satisfied. Though if that condition is not satisfied, the function will run and cost gas.

Resolution: We suggest using require with proper error message instead of if condition.

Status: Fixed

(5) Irrelevant error message: **MasterChef.sol**

```
function updateRewardsRate(uint256 _rewardsPerBlock) public onlyOwner {
    require(
        _rewardsPerBlock <= 4e18,
        "maximum emission rate of 4 tokens per block exceeded"
    );
```

In updateRewardsRate, an error message was mentioned for the emission rate per block instead of the reward rate per block.

Resolution: We suggest correcting the error message.
Status: Fixed

(6) rewardPoolAddress should be made immutable: **MasterChef.sol**
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.

Resolution: Consider marking this variable as immutable.
Status: Fixed

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:

- setUnirouter: The StratManager owner can update the router that will be used for swaps.
- setVault: The StratManager owner can update the parent vault.
- setUnirouter: The StratManagerLP owner can update the router that will be used for swaps.
- setVault: The StratManagerLP owner can update the parent vault.
- setToshaFeeRecipient: The StratManagerLP owner can update the tosha fee recipient.
- proposeStrat: The ToshaVault owner can set the candidate for the new strat to use with this vault.
- upgradeStrat: The ToshaVault owner can update switches to the active strat for the strat candidate.
- inCaseTokensGetStuck: The ToshaVault owner can rescue random funds stuck that the strat can't handle.
- proposeStrat: The ToshaLPVault owner can set the candidate for the new strat to use with this vault.
- upgradeStrat: The ToshaLPVault owner can update switches to the active strat for the strat candidate.
- inCaseTokensGetStuck: The ToshaLPVault owner can rescue random funds stuck that the strat can't handle.

- updateReserveFundsAddress: The MasterChef owner can update reserve funds addressed.
- updateFarmingRewarderAddress: The MasterChef owner can update farming rewarder addresses.
- updateEmissionRate: The MasterChef owner can update the emission rate.
- updateRewardsRate: The MasterChef owner can update the rewards rate.

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"**.

Our Methodology

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

Manual Code Review:

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

Vulnerability Analysis:

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

Documenting Results:

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

Suggested Solutions:

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

Disclaimers

EtherAuthority.io Disclaimer

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

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

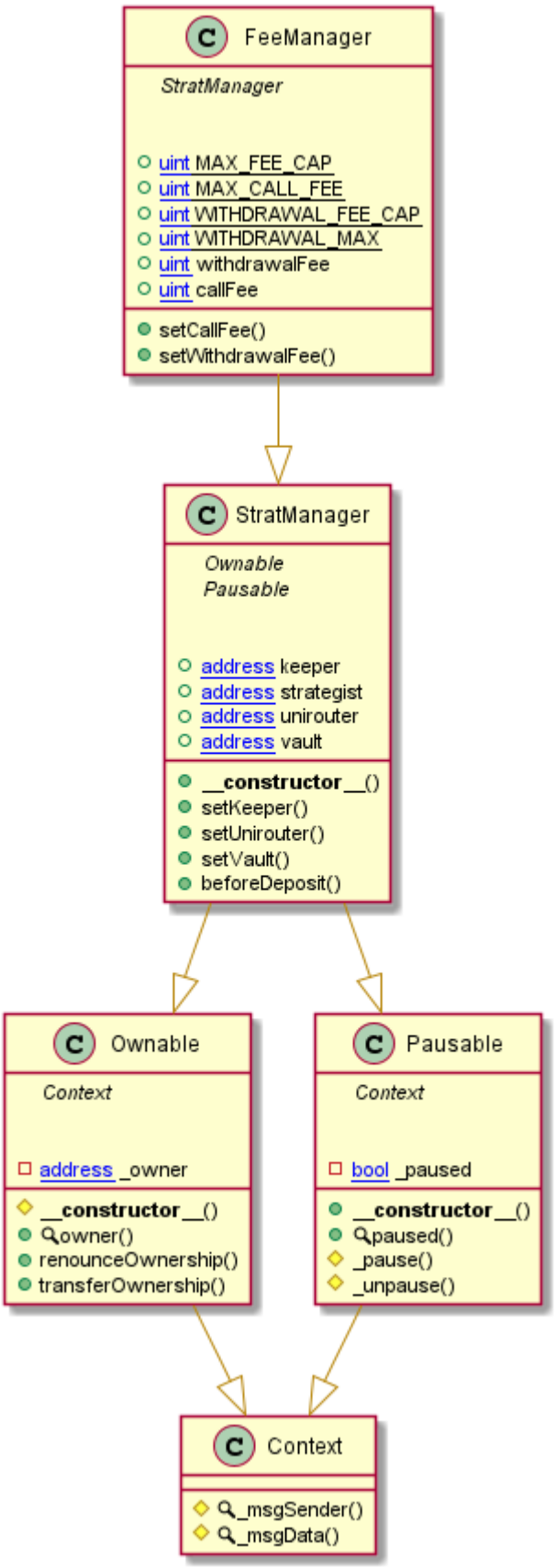
Technical Disclaimer

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

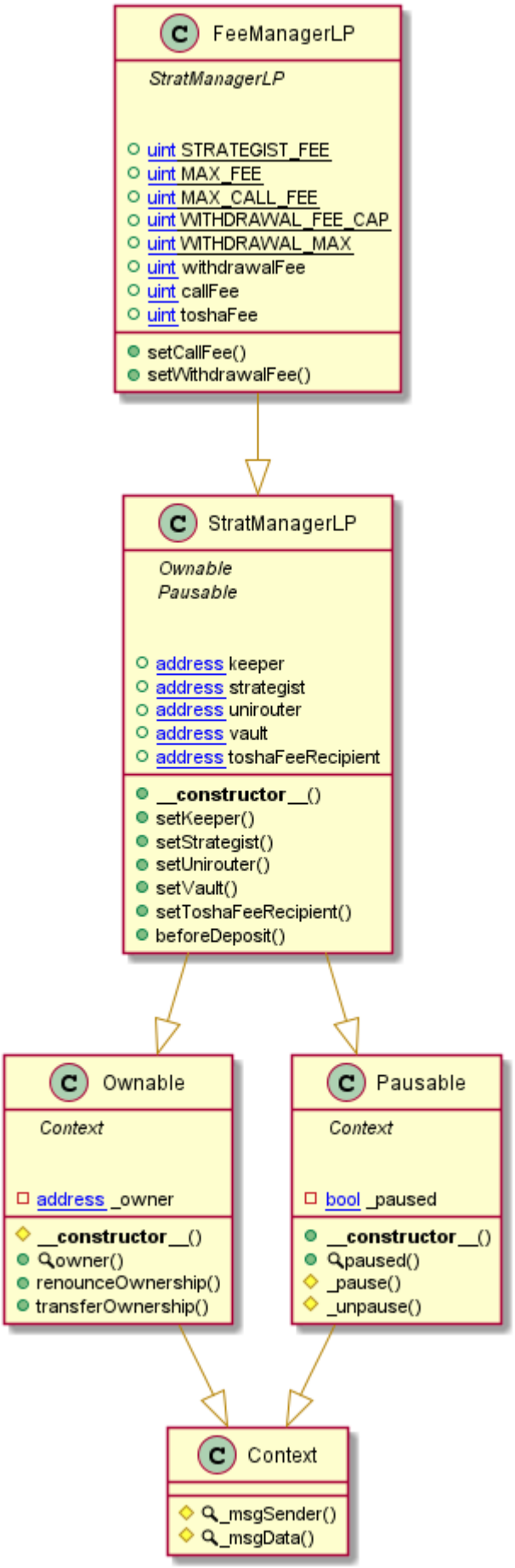
Appendix

Code Flow Diagram - Tosha Protocol

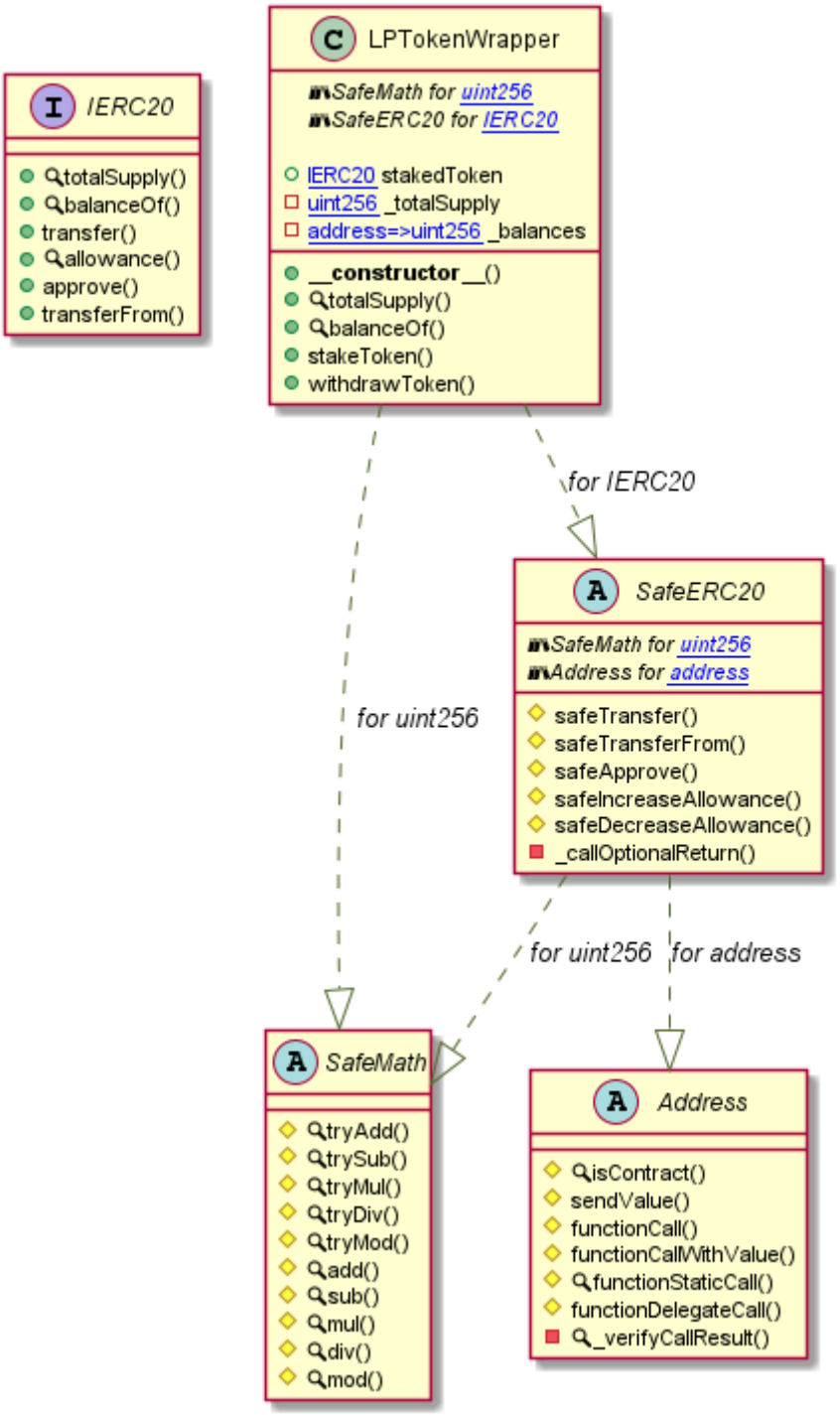
FeeManager Diagram



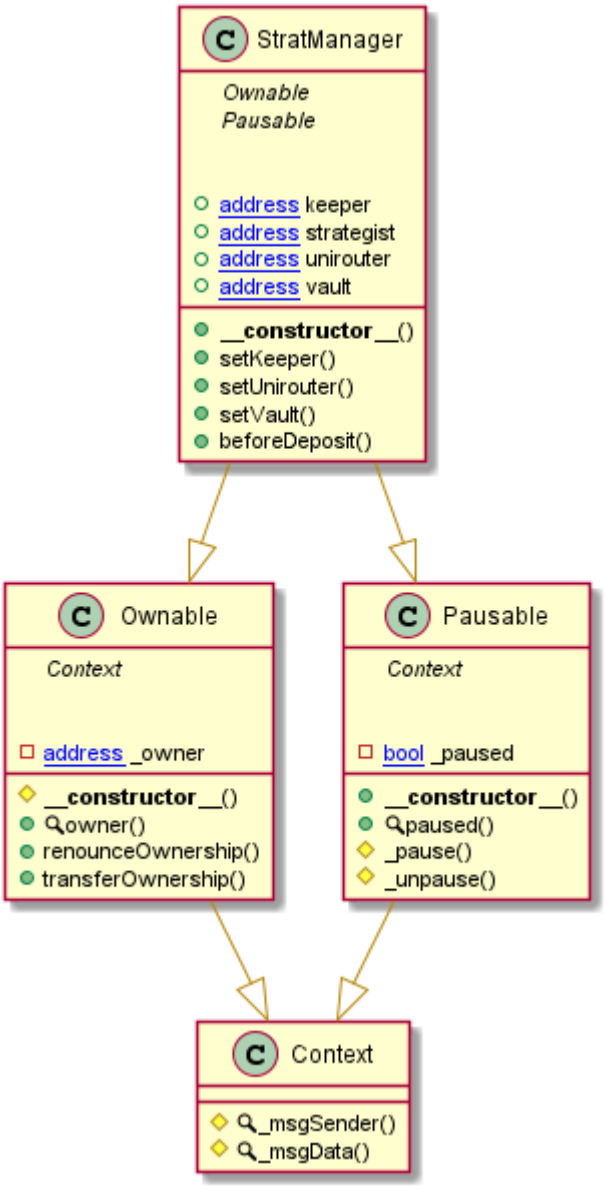
FeeManagerLP Diagram



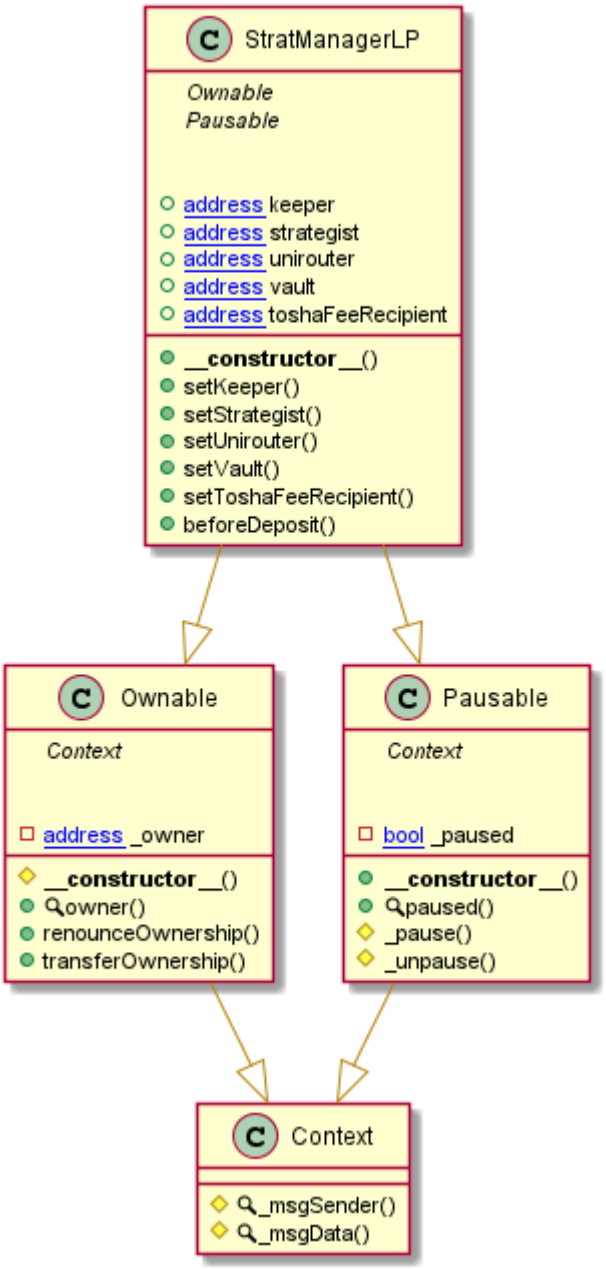
LPTokenWrapper Diagram



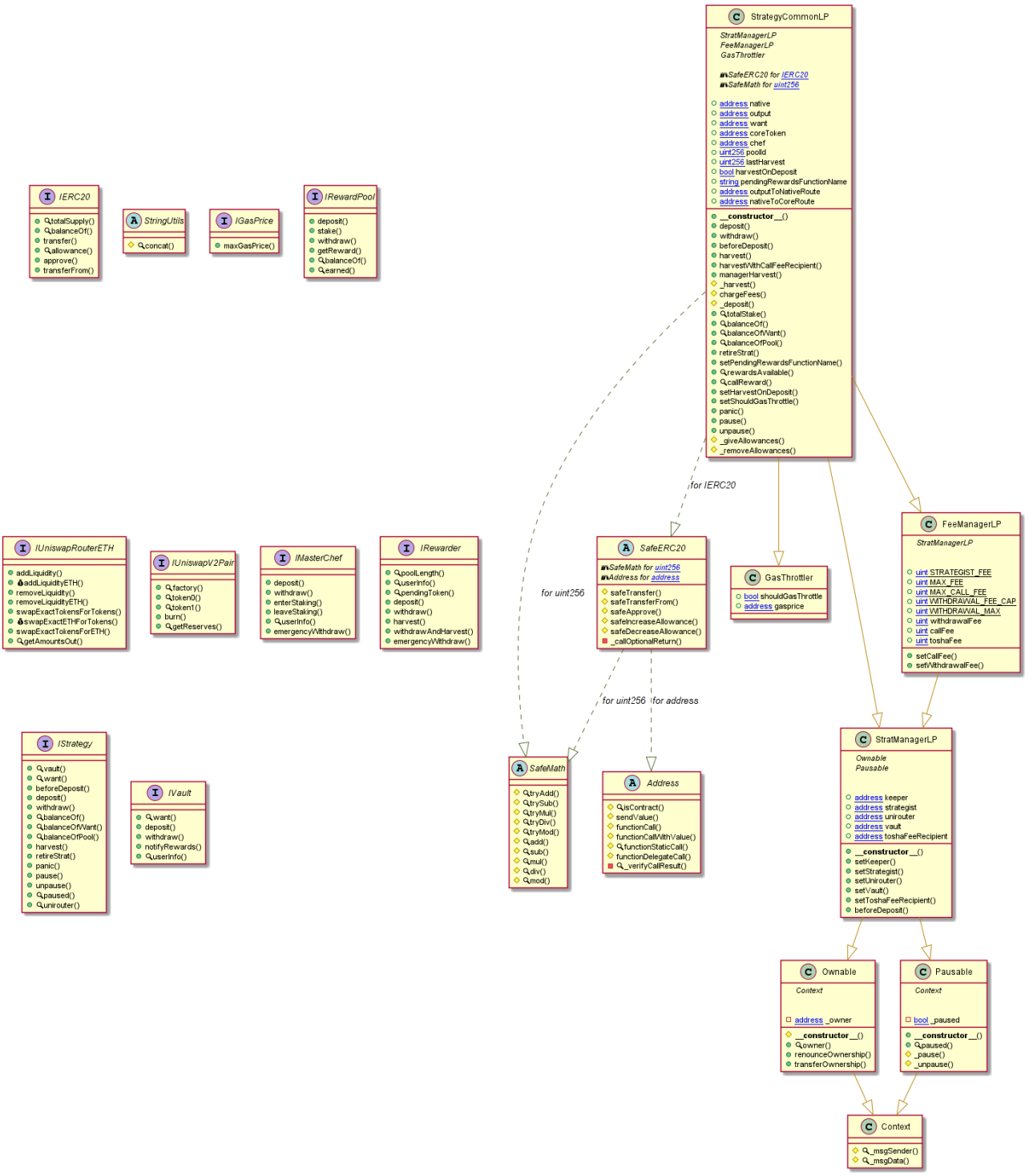
StratManager Diagram



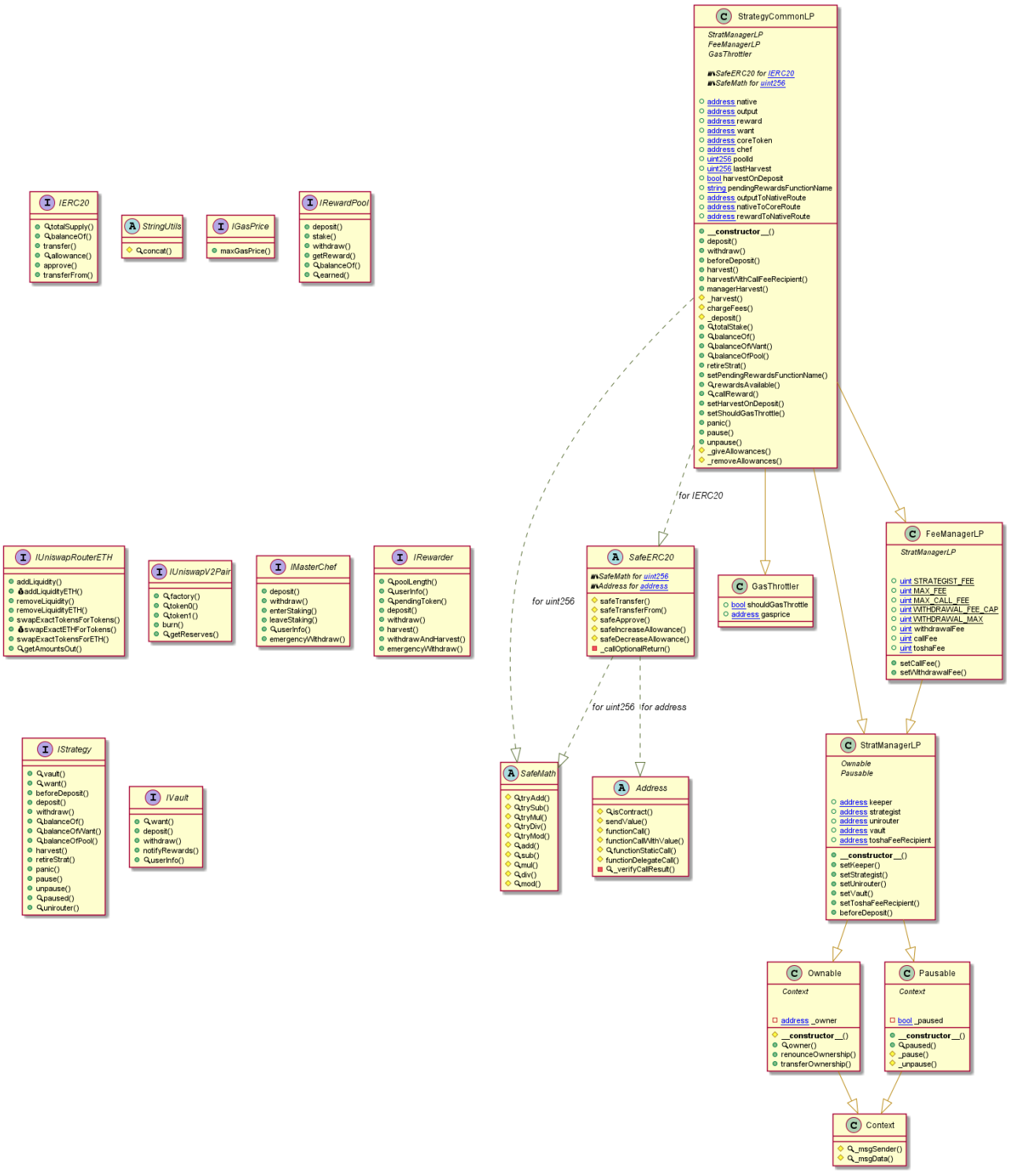
StratManagerLP Diagram



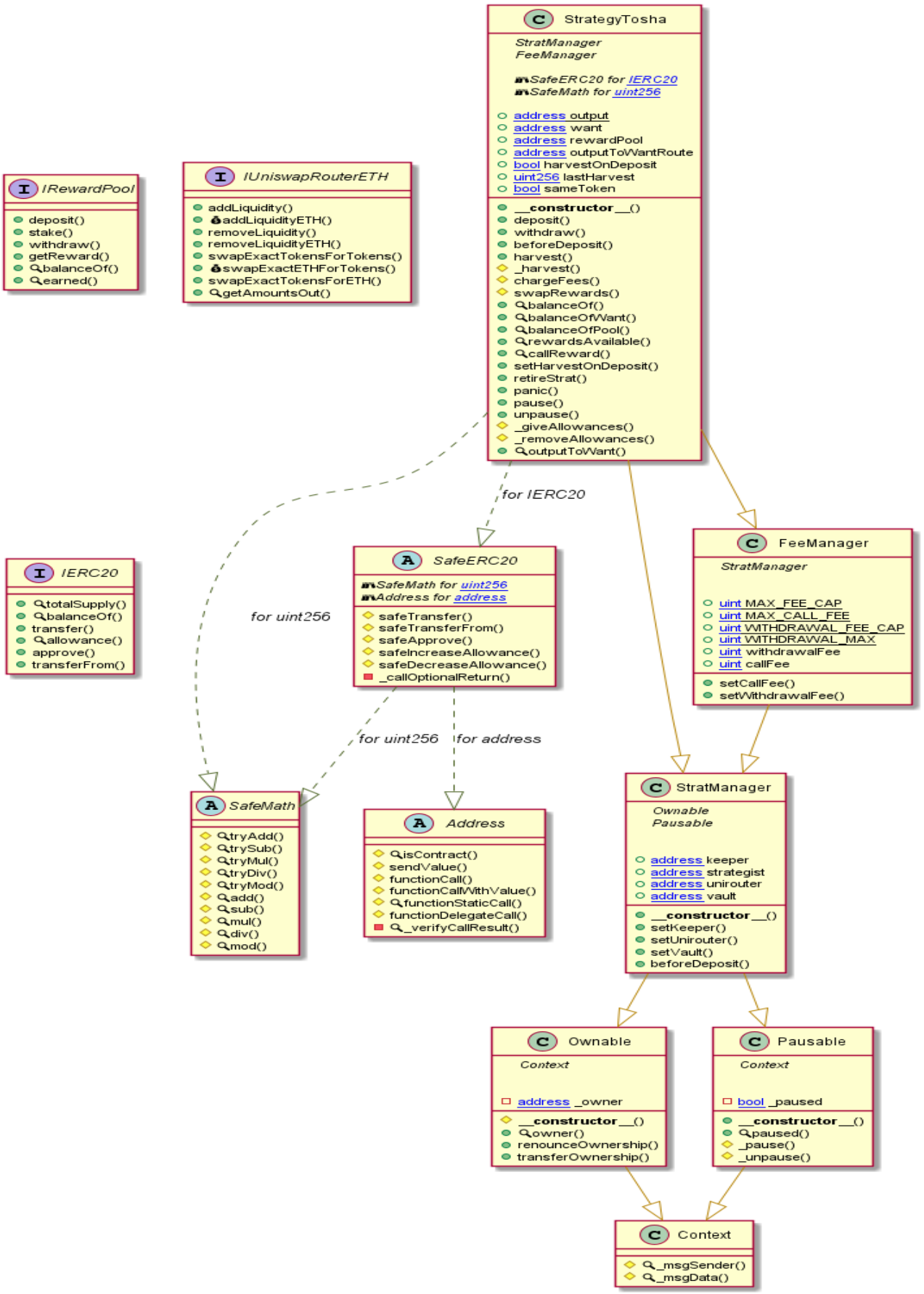
StrategyCommonLP Diagram



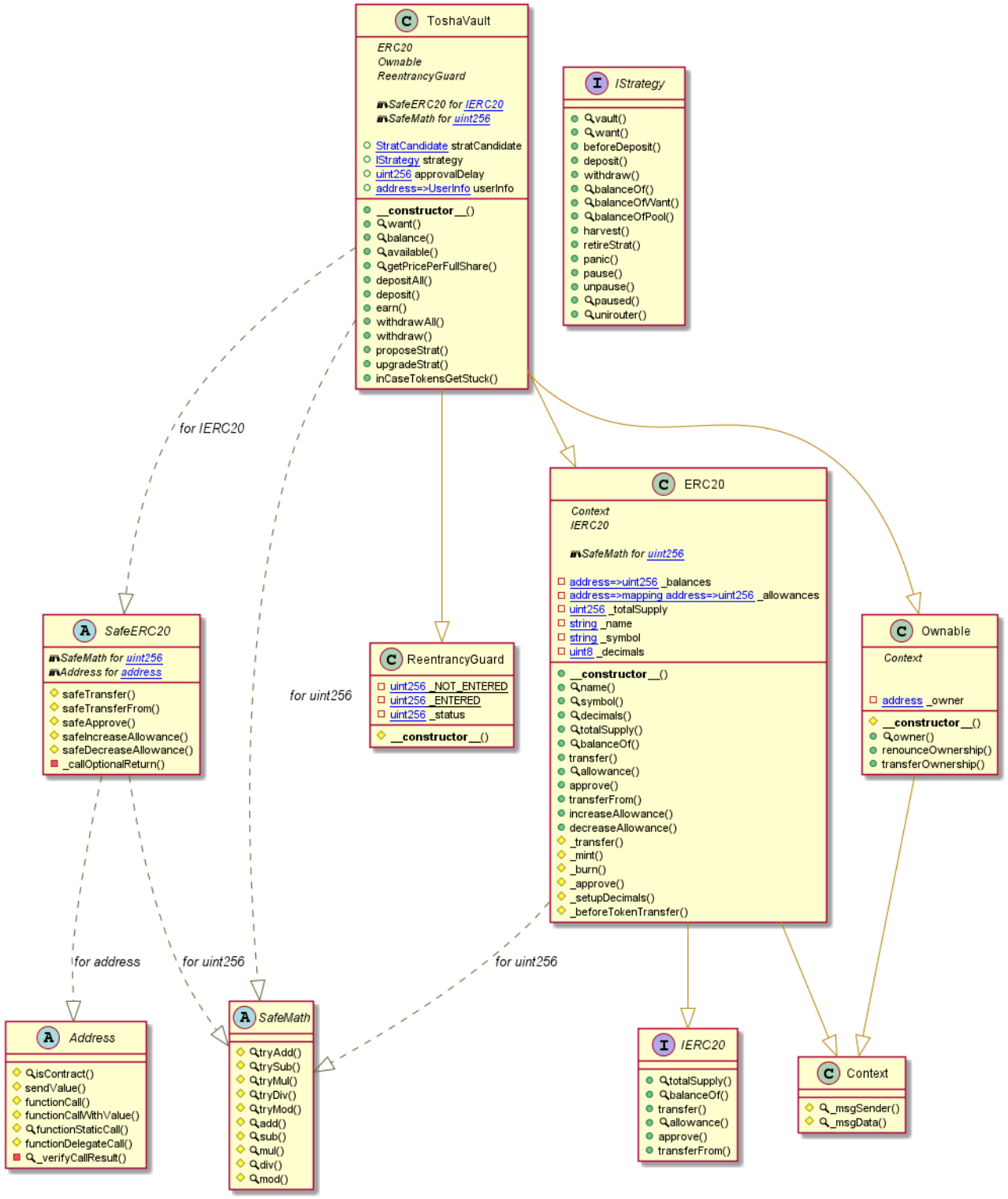
StrategyDualLP Diagram



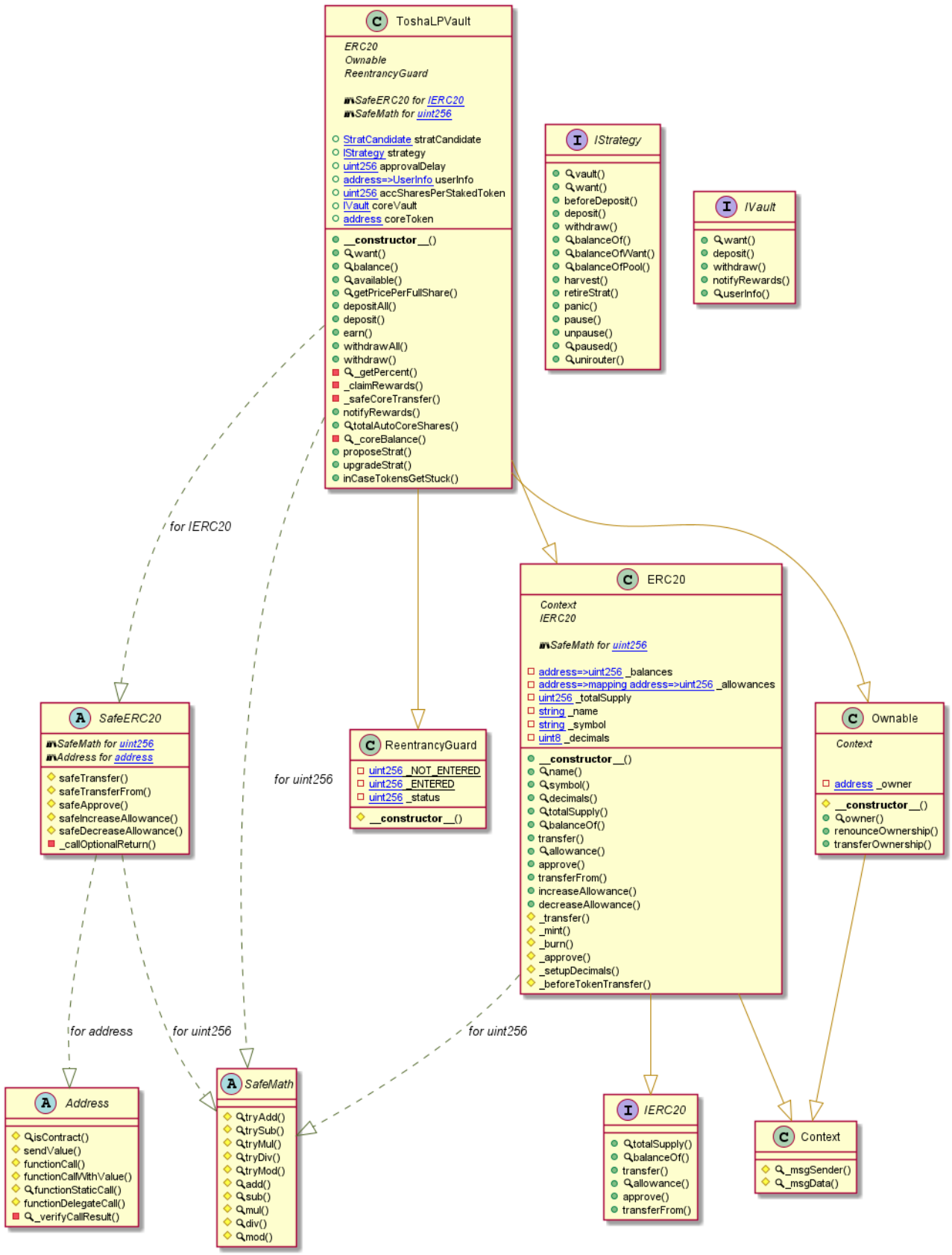
StrategyTosha Diagram



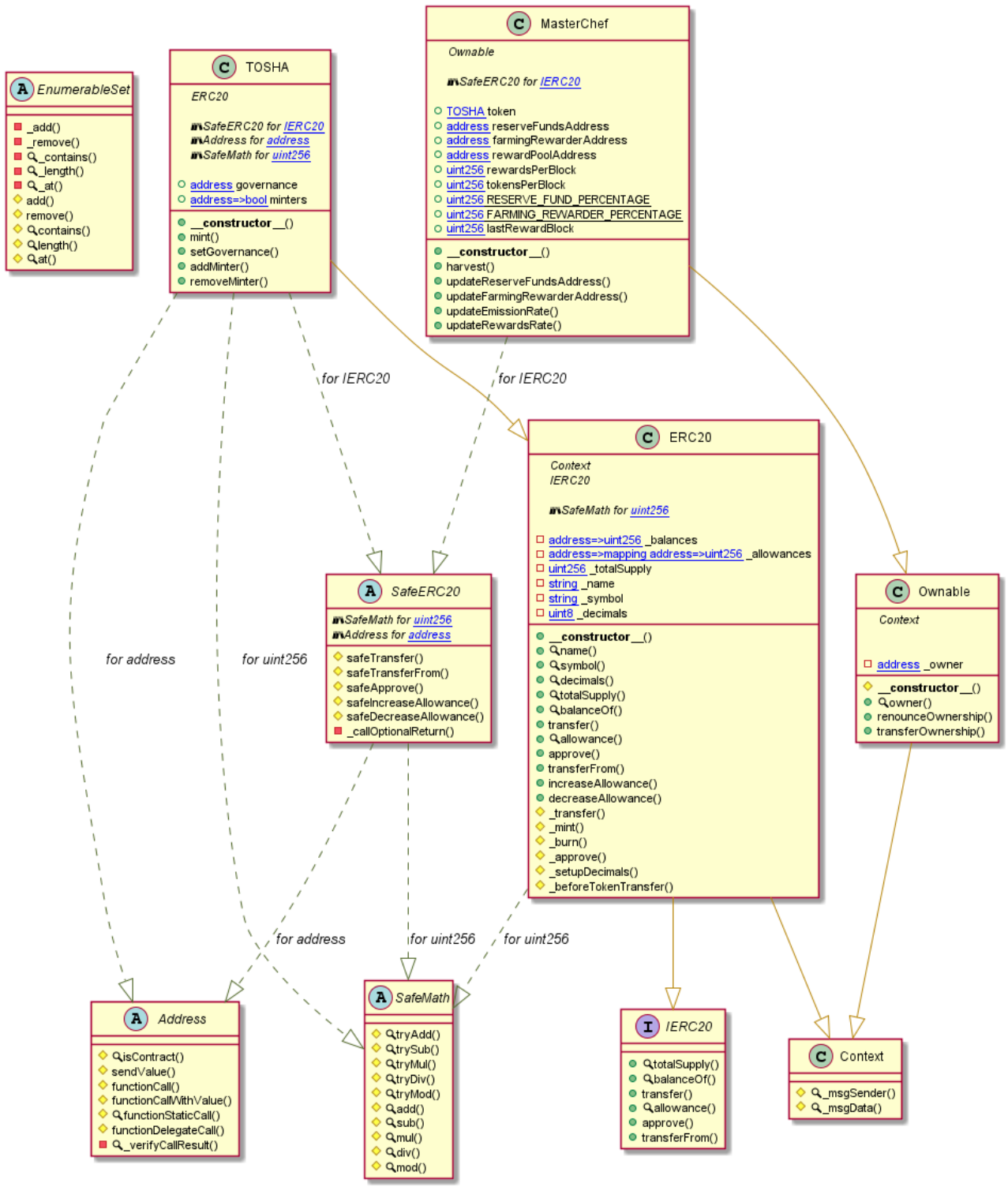
ToshaVault Diagram



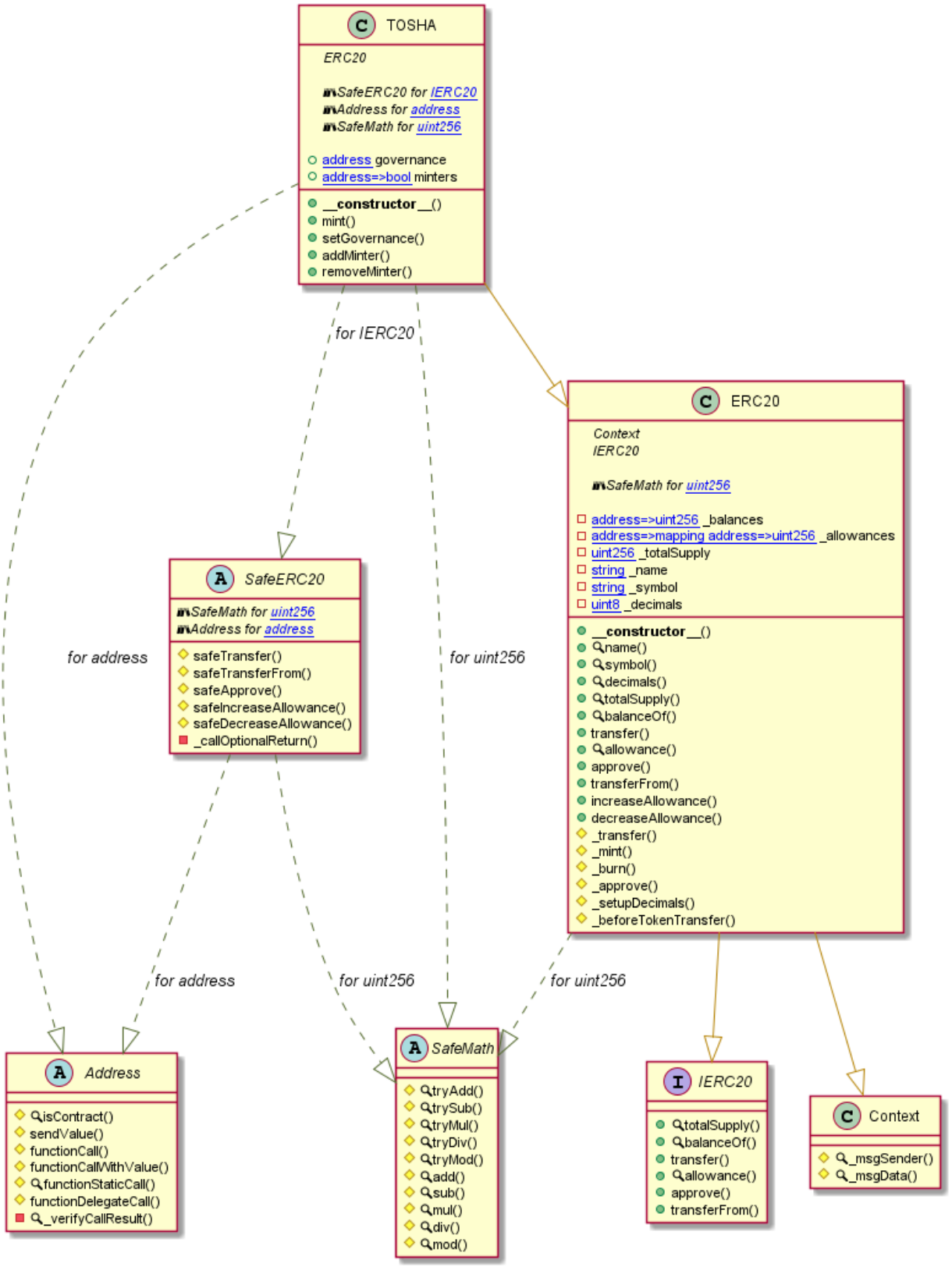
ToshaLPVault Diagram



MasterChef Diagram



Tosha Diagram



Slither Results Log

Slither log >> FeeManager.sol

```
INFO:Detectors:
StratManager.setKeeper(address) (FeeManager.sol#180-182) should emit an event for:
        - keeper = _keeper (FeeManager.sol#181)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
StratManager.constructor(address,address,address)._keeper (FeeManager.sol#161) lacks a zero-check on :
                - keeper = _keeper (FeeManager.sol#165)
StratManager.constructor(address,address,address)._unirouter (FeeManager.sol#162) lacks a zero-check on :
                - unirouter = _unirouter (FeeManager.sol#166)
StratManager.constructor(address,address,address)._vault (FeeManager.sol#163) lacks a zero-check on :
                - vault = _vault (FeeManager.sol#167)
StratManager.setKeeper(address)._keeper (FeeManager.sol#180) lacks a zero-check on :
                - keeper = _keeper (FeeManager.sol#181)
StratManager.setUnirouter(address)._unirouter (FeeManager.sol#188) lacks a zero-check on :
                - unirouter = _unirouter (FeeManager.sol#189)
StratManager.setVault(address)._vault (FeeManager.sol#196) lacks a zero-check on :
                - vault = _vault (FeeManager.sol#197)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Context._msgData() (FeeManager.sol#10-13) is never used and should be removed
Pausable._pause() (FeeManager.sol#124-127) is never used and should be removed
Pausable._unpause() (FeeManager.sol#136-139) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Parameter StratManager.setKeeper(address)._keeper (FeeManager.sol#180) is not in mixedCase
Parameter StratManager.setUnirouter(address)._unirouter (FeeManager.sol#188) is not in mixedCase
Parameter StratManager.setVault(address)._vault (FeeManager.sol#196) is not in mixedCase
Parameter FeeManager.setCallFee(uint256)._fee (FeeManager.sol#220) is not in mixedCase
Parameter FeeManager.setWithdrawalFee(uint256)._fee (FeeManager.sol#226) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (FeeManager.sol#11)" inContext (FeeManager.sol#5-14)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
StratManager.strategist (FeeManager.sol#150) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
```

```
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (FeeManager.sol#51-54)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (FeeManager.sol#60-64)
setCallFee(uint256) should be declared external:
        - FeeManager.setCallFee(uint256) (FeeManager.sol#220-224)
setWithdrawalFee(uint256) should be declared external:
        - FeeManager.setWithdrawalFee(uint256) (FeeManager.sol#226-230)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:FeeManager.sol analyzed (5 contracts with 75 detectors), 21 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> FeeManagerLP.sol

```
INFO:Detectors:
StratManagerLP.setKeeper(address) (FeeManagerLP.sol#185-187) should emit an event for:
        - keeper = _keeper (FeeManagerLP.sol#186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
FeeManagerLP.setCallFee(uint256) (FeeManagerLP.sol#233-238) should emit an event for:
        - callFee = _fee (FeeManagerLP.sol#236)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
StratManagerLP.constructor(address,address,address,address,address)._keeper (FeeManagerLP.sol#164) lacks a zero-check on :
                - keeper = _keeper (FeeManagerLP.sol#170)
StratManagerLP.constructor(address,address,address,address,address)._strategist (FeeManagerLP.sol#165) lacks a zero-check on
 :
                - strategist = _strategist (FeeManagerLP.sol#171)
StratManagerLP.constructor(address,address,address,address,address)._unirouter (FeeManagerLP.sol#166) lacks a zero-check on
:
                - unirouter = _unirouter (FeeManagerLP.sol#172)
StratManagerLP.constructor(address,address,address,address,address)._vault (FeeManagerLP.sol#167) lacks a zero-check on :
                - vault = _vault (FeeManagerLP.sol#173)
StratManagerLP.constructor(address,address,address,address,address)._toshaFeeRecipient (FeeManagerLP.sol#168) lacks a zero-c
heck on :
                - toshaFeeRecipient = _toshaFeeRecipient (FeeManagerLP.sol#174)
StratManagerLP.setKeeper(address)._keeper (FeeManagerLP.sol#185) lacks a zero-check on :
                - keeper = _keeper (FeeManagerLP.sol#186)
StratManagerLP.setStrategist(address)._strategist (FeeManagerLP.sol#192) lacks a zero-check on :
                - strategist = _strategist (FeeManagerLP.sol#194)
StratManagerLP.setUnirouter(address)._unirouter (FeeManagerLP.sol#200) lacks a zero-check on :
                - unirouter = _unirouter (FeeManagerLP.sol#201)
StratManagerLP.setVault(address)._vault (FeeManagerLP.sol#207) lacks a zero-check on :
                - vault = _vault (FeeManagerLP.sol#208)
StratManagerLP.setToshaFeeRecipient(address)._toshaFeeRecipient (FeeManagerLP.sol#214) lacks a zero-check on :
                - toshaFeeRecipient = _toshaFeeRecipient (FeeManagerLP.sol#215)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Context._msgData() (FeeManagerLP.sol#9-12) is never used and should be removed
Pausable._pause() (FeeManagerLP.sol#123-126) is never used and should be removed
Pausable._unpause() (FeeManagerLP.sol#135-138) is never used and should be removed
```

```
Pausable._unpause() (FeeManagerLP.sol#135-138) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
FeeManagerLP.toshaFee (FeeManagerLP.sol#232) is set pre-construction with a non-constant function or state variable:
        - MAX_FEE - STRATEGIST_FEE - callFee
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state-variables
INFO:Detectors:
Parameter StratManagerLP.setKeeper(address)._keeper (FeeManagerLP.sol#185) is not in mixedCase
Parameter StratManagerLP.setStrategist(address)._strategist (FeeManagerLP.sol#192) is not in mixedCase
Parameter StratManagerLP.setUnirouter(address)._unirouter (FeeManagerLP.sol#200) is not in mixedCase
Parameter StratManagerLP.setVault(address)._vault (FeeManagerLP.sol#207) is not in mixedCase
Parameter StratManagerLP.setToshaFeeRecipient(address)._toshaFeeRecipient (FeeManagerLP.sol#214) is not in mixedCase
Parameter FeeManagerLP.setCallFee(uint256)._fee (FeeManagerLP.sol#233) is not in mixedCase
Parameter FeeManagerLP.setWithdrawalFee(uint256)._fee (FeeManagerLP.sol#239) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (FeeManagerLP.sol#10)" inContext (FeeManagerLP.sol#4-13)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (FeeManagerLP.sol#50-53)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (FeeManagerLP.sol#59-63)
setCallFee(uint256) should be declared external:
        - FeeManagerLP.setCallFee(uint256) (FeeManagerLP.sol#233-238)
setWithdrawalFee(uint256) should be declared external:
        - FeeManagerLP.setWithdrawalFee(uint256) (FeeManagerLP.sol#239-242)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:FeeManagerLP.sol analyzed (5 contracts with 75 detectors), 28 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> LPTokenWrapper.sol

```
INFO:Detectors:
Address.isContract(address) (LPTokenWrapper.sol#93-102) uses assembly
        - INLINE ASM (LPTokenWrapper.sol#100)
Address._verifyCallResult(bool,bytes,string) (LPTokenWrapper.sol#238-255) uses assembly
        - INLINE ASM (LPTokenWrapper.sol#247-250)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (LPTokenWrapper.sol#146-148) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (LPTokenWrapper.sol#171-173) is never used and should be removed
Address.functionDelegateCall(address,bytes) (LPTokenWrapper.sol#220-222) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (LPTokenWrapper.sol#230-236) is never used and should be removed
Address.functionStaticCall(address,bytes) (LPTokenWrapper.sol#196-198) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (LPTokenWrapper.sol#206-212) is never used and should be removed
Address.sendValue(address,uint256) (LPTokenWrapper.sol#120-126) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (LPTokenWrapper.sol#475-484) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (LPTokenWrapper.sol#491-494) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (LPTokenWrapper.sol#486-489) is never used and should be removed
SafeMath.div(uint256,uint256) (LPTokenWrapper.sol#375-378) is never used and should be removed
SafeMath.div(uint256,uint256,string) (LPTokenWrapper.sol#430-433) is never used and should be removed
SafeMath.mod(uint256,uint256) (LPTokenWrapper.sol#392-395) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (LPTokenWrapper.sol#450-453) is never used and should be removed
SafeMath.mul(uint256,uint256) (LPTokenWrapper.sol#356-361) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (LPTokenWrapper.sol#410-413) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (LPTokenWrapper.sol#264-268) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (LPTokenWrapper.sol#300-303) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (LPTokenWrapper.sol#310-313) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (LPTokenWrapper.sol#285-293) is never used and should be removed
SafeMath.trySub(uint256,uint256) (LPTokenWrapper.sol#275-278) 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) (LPTokenWrapper.sol#120-126):
        - (success) = recipient.call{value: amount}() (LPTokenWrapper.sol#124)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (LPTokenWrapper.sol#181-188):
        - (success,returndata) = target.call{value: value}(data) (LPTokenWrapper.sol#186)
Low level call in Address.functionStaticCall(address,bytes,string) (LPTokenWrapper.sol#206-212):
        - (success,returndata) = target.staticcall(data) (LPTokenWrapper.sol#210)
Low level call in Address.functionDelegateCall(address,bytes,string) (LPTokenWrapper.sol#230-236):
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
totalSupply() should be declared external:
        - LPTokenWrapper.totalSupply() (LPTokenWrapper.sol#527-529)
balanceOf(address) should be declared external:
        - LPTokenWrapper.balanceOf(address) (LPTokenWrapper.sol#531-533)
stakeToken(uint256) should be declared external:
        - LPTokenWrapper.stakeToken(uint256) (LPTokenWrapper.sol#535-539)
withdrawToken(uint256) should be declared external:
        - LPTokenWrapper.withdrawToken(uint256) (LPTokenWrapper.sol#541-545)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:LPTokenWrapper.sol analyzed (5 contracts with 75 detectors), 31 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> StratManager.sol

```
INFO:Detectors:
StratManager.setKeeper(address) (StratManager.sol#180-182) should emit an event for:
        - keeper = _keeper (StratManager.sol#181)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
StratManager.constructor(address,address,address)._keeper (StratManager.sol#161) lacks a zero-check on :
                - keeper = _keeper (StratManager.sol#165)
StratManager.constructor(address,address,address)._unirouter (StratManager.sol#162) lacks a zero-check on :
                - unirouter = _unirouter (StratManager.sol#166)
```

```
                - unirouter = _unirouter (StratManager.sol#166)
StratManager.constructor(address,address,address)._vault (StratManager.sol#163) lacks a zero-check on :
                - vault = _vault (StratManager.sol#167)
StratManager.setKeeper(address)._keeper (StratManager.sol#180) lacks a zero-check on :
                - keeper = _keeper (StratManager.sol#181)
StratManager.setUnirouter(address)._unirouter (StratManager.sol#188) lacks a zero-check on :
                - unirouter = _unirouter (StratManager.sol#189)
StratManager.setVault(address)._vault (StratManager.sol#196) lacks a zero-check on :
                - vault = _vault (StratManager.sol#197)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Context._msgData() (StratManager.sol#10-13) is never used and should be removed
Pausable._pause() (StratManager.sol#124-127) is never used and should be removed
Pausable._unpause() (StratManager.sol#136-139) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Parameter StratManager.setKeeper(address)._keeper (StratManager.sol#180) is not in mixedCase
Parameter StratManager.setUnirouter(address)._unirouter (StratManager.sol#188) is not in mixedCase
Parameter StratManager.setVault(address)._vault (StratManager.sol#196) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (StratManager.sol#11)" inContext (StratManager.sol#5-14)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
StratManager.strategist (StratManager.sol#150) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (StratManager.sol#51-54)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (StratManager.sol#60-64)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:StratManager.sol analyzed (4 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 >> StratManagerLP.sol

```
INFO:Detectors:
StratManagerLP.setKeeper(address) (StratManagerLP.sol#185-187) should emit an event for:
        - keeper = _keeper (StratManagerLP.sol#186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
StratManagerLP.constructor(address,address,address,address,address)._keeper (StratManagerLP.sol#164) lacks a zero-check on :
                - keeper = _keeper (StratManagerLP.sol#170)
StratManagerLP.constructor(address,address,address,address,address)._strategist (StratManagerLP.sol#165) lacks a zero-check
on :
                - strategist = _strategist (StratManagerLP.sol#171)
StratManagerLP.constructor(address,address,address,address,address)._unirouter (StratManagerLP.sol#166) lacks a zero-check o
n :
                - unirouter = _unirouter (StratManagerLP.sol#172)
StratManagerLP.constructor(address,address,address,address,address)._vault (StratManagerLP.sol#167) lacks a zero-check on :
                - vault = _vault (StratManagerLP.sol#173)
StratManagerLP.constructor(address,address,address,address,address)._toshaFeeRecipient (StratManagerLP.sol#168) lacks a zero
-check on :
                - toshaFeeRecipient = _toshaFeeRecipient (StratManagerLP.sol#174)
StratManagerLP.setKeeper(address)._keeper (StratManagerLP.sol#185) lacks a zero-check on :
                - keeper = _keeper (StratManagerLP.sol#186)
StratManagerLP.setStrategist(address)._strategist (StratManagerLP.sol#192) lacks a zero-check on :
                - strategist = _strategist (StratManagerLP.sol#194)
StratManagerLP.setUnirouter(address)._unirouter (StratManagerLP.sol#200) lacks a zero-check on :
                - unirouter = _unirouter (StratManagerLP.sol#201)
StratManagerLP.setVault(address)._vault (StratManagerLP.sol#207) lacks a zero-check on :
                - vault = _vault (StratManagerLP.sol#208)
StratManagerLP.setToshaFeeRecipient(address)._toshaFeeRecipient (StratManagerLP.sol#214) lacks a zero-check on :
                - toshaFeeRecipient = _toshaFeeRecipient (StratManagerLP.sol#215)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Context._msgData() (StratManagerLP.sol#9-12) is never used and should be removed
Pausable._pause() (StratManagerLP.sol#123-126) is never used and should be removed
Pausable._unpause() (StratManagerLP.sol#135-138) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

```
INFO:Detectors:
Parameter StratManagerLP.setKeeper(address)._keeper (StratManagerLP.sol#185) is not in mixedCase
Parameter StratManagerLP.setStrategist(address)._strategist (StratManagerLP.sol#192) is not in mixedCase
Parameter StratManagerLP.setUnirouter(address)._unirouter (StratManagerLP.sol#200) is not in mixedCase
Parameter StratManagerLP.setVault(address)._vault (StratManagerLP.sol#207) is not in mixedCase
Parameter StratManagerLP.setToshaFeeRecipient(address)._toshaFeeRecipient (StratManagerLP.sol#214) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (StratManagerLP.sol#10)" inContext (StratManagerLP.sol#4-13)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (StratManagerLP.sol#50-53)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (StratManagerLP.sol#59-63)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:StratManagerLP.sol analyzed (4 contracts with 75 detectors), 22 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> StrategyCommonLP.sol

```
INFO:Detectors:
StratManagerLP.setKeeper(address) (StrategyCommonLP.sol#843-845) should emit an event for:
        - keeper = _keeper (StrategyCommonLP.sol#844)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
```

```
INFO:Detectors:
FeeManagerLP.setCallFee(uint256) (StrategyCommonLP.sol#891-896) should emit an event for:
        - callFee = _fee (StrategyCommonLP.sol#894)
        - toshaFee = MAX_FEE - STRATEGIST_FEE - callFee (StrategyCommonLP.sol#895)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
StratManagerLP.constructor(address,address,address,address,address)._keeper (StrategyCommonLP.sol#822) lacks a zero-check on
 :
                - keeper = _keeper (StrategyCommonLP.sol#828)
StratManagerLP.constructor(address,address,address,address,address)._strategist (StrategyCommonLP.sol#823) lacks a zero-chec
k on :
                - strategist = _strategist (StrategyCommonLP.sol#829)
StratManagerLP.constructor(address,address,address,address,address)._unirouter (StrategyCommonLP.sol#824) lacks a zero-check
 on :
                - unirouter = _unirouter (StrategyCommonLP.sol#830)
StratManagerLP.constructor(address,address,address,address,address)._vault (StrategyCommonLP.sol#825) lacks a zero-check on
:
                - vault = _vault (StrategyCommonLP.sol#831)
StratManagerLP.constructor(address,address,address,address,address)._toshaFeeRecipient (StrategyCommonLP.sol#826) lacks a ze
ro-check on :
                - toshaFeeRecipient = _toshaFeeRecipient (StrategyCommonLP.sol#832)
StratManagerLP.setKeeper(address)._keeper (StrategyCommonLP.sol#843) lacks a zero-check on :
                - keeper = _keeper (StrategyCommonLP.sol#844)
StratManagerLP.setStrategist(address)._strategist (StrategyCommonLP.sol#850) lacks a zero-check on :
                - strategist = _strategist (StrategyCommonLP.sol#852)
StratManagerLP.setUnirouter(address)._unirouter (StrategyCommonLP.sol#858) lacks a zero-check on :
                - unirouter = _unirouter (StrategyCommonLP.sol#859)
StratManagerLP.setVault(address)._vault (StrategyCommonLP.sol#865) lacks a zero-check on :
                - vault = _vault (StrategyCommonLP.sol#866)
StratManagerLP.setToshaFeeRecipient(address)._toshaFeeRecipient (StrategyCommonLP.sol#872) lacks a zero-check on :
                - toshaFeeRecipient = _toshaFeeRecipient (StrategyCommonLP.sol#873)
StrategyCommonLP.constructor(address,uint256,address,address,address,address,address,address,address[],address[])._want (Str
ategyCommonLP.sol#935) lacks a zero-check on :
                - want = _want (StrategyCommonLP.sol#946)
StrategyCommonLP.constructor(address,uint256,address,address,address,address,address,address,address[],address[])._chef (Str
ategyCommonLP.sol#937) lacks a zero-check on :
                - chef = _chef (StrategyCommonLP.sol#948)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Variable 'StrategyCommonLP.callReward().amountOut (StrategyCommonLP.sol#1090)' in StrategyCommonLP.callReward() (StrategyCom
monLP.sol#1085-1098) potentially used before declaration: nativeOut = amountOut[amountOut.length - 1] (StrategyCommonLP.sol#
1092)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables
INFO:Detectors:
Reentrancy in StrategyCommonLP._harvest(address) (StrategyCommonLP.sol#1006-1016):
        External calls:
        - IMasterChef(chef).deposit(poolId,0) (StrategyCommonLP.sol#1007)
        - chargeFees(callFeeRecipient) (StrategyCommonLP.sol#1010)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (StrategyCommonLP.sol#508)
                - IUniswapRouterETH(unirouter).swapExactTokensForTokens(toNative,0,outputToNativeRoute,address(this),block.t
imestamp) (StrategyCommonLP.sol#1020)
                - IERC20(native).safeTransfer(callFeeRecipient,callFeeAmount) (StrategyCommonLP.sol#1023)
                - (success,returndata) = target.call{value: value}(data) (StrategyCommonLP.sol#187)
                - IERC20(native).safeTransfer(toshaFeeRecipient,toshaFeeAmount) (StrategyCommonLP.sol#1025)
                - IERC20(native).safeTransfer(strategist,strategistFee) (StrategyCommonLP.sol#1027)
        - _deposit() (StrategyCommonLP.sol#1012)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (StrategyCommonLP.sol#508)
                - IUniswapRouterETH(unirouter).swapExactTokensForTokens(nativeToken,0,nativeToCoreRoute,address(this),block.
timestamp) (StrategyCommonLP.sol#1033)
                - (success,returndata) = target.call{value: value}(data) (StrategyCommonLP.sol#187)
                - IERC20(coreToken).safeTransfer(vault,coreBal) (StrategyCommonLP.sol#1037)
                - IVault(vault).notifyRewards(totalStake()) (StrategyCommonLP.sol#1038)
        External calls sending eth:
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Address.isContract(address) (StrategyCommonLP.sol#94-103) uses assembly
        - INLINE ASM (StrategyCommonLP.sol#101)
Address._verifyCallResult(bool,bytes,string) (StrategyCommonLP.sol#239-256) uses assembly
        - INLINE ASM (StrategyCommonLP.sol#248-251)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (StrategyCommonLP.sol#147-149) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (StrategyCommonLP.sol#172-174) is never used and should be removed
Address.functionDelegateCall(address,bytes) (StrategyCommonLP.sol#221-223) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (StrategyCommonLP.sol#231-237) is never used and should be removed
Address.sendValue(address,uint256) (StrategyCommonLP.sol#121-127) is never used and should be removed
Context._msgData() (StrategyCommonLP.sol#667-670) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (StrategyCommonLP.sol#492-495) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (StrategyCommonLP.sol#487-490) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (StrategyCommonLP.sol#465-467) is never used and should be remove
d
SafeMath.div(uint256,uint256,string) (StrategyCommonLP.sol#431-434) is never used and should be removed
SafeMath.mod(uint256,uint256) (StrategyCommonLP.sol#393-396) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (StrategyCommonLP.sol#451-454) is never used and should be removed
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
GasThrottler.gasprice (StrategyCommonLP.sol#652) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (StrategyCommonLP.sol#708-711)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (StrategyCommonLP.sol#717-721)
setCallFee(uint256) should be declared external:
        - FeeManagerLP.setCallFee(uint256) (StrategyCommonLP.sol#891-896)
callReward() should be declared external:
        - StrategyCommonLP.callReward() (StrategyCommonLP.sol#1085-1098)
panic() should be declared external:
        - StrategyCommonLP.panic() (StrategyCommonLP.sol#1113-1116)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:StrategyCommonLP.sol analyzed (20 contracts with 75 detectors), 79 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> StrategyDualLP.sol

```
INFO:Detectors:
StratManagerLP.setKeeper(address) (StrategyDualLP.sol#843-845) should emit an event for:
        - keeper = _keeper (StrategyDualLP.sol#844)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
FeeManagerLP.setCallFee(uint256) (StrategyDualLP.sol#891-896) should emit an event for:
        - callFee = _fee (StrategyDualLP.sol#894)
        - toshaFee = MAX_FEE - STRATEGIST_FEE - callFee (StrategyDualLP.sol#895)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
StratManagerLP.constructor(address,address,address,address,address)._keeper (StrategyDualLP.sol#822) lacks a zero-check on :
                - keeper = _keeper (StrategyDualLP.sol#828)
StratManagerLP.constructor(address,address,address,address,address)._strategist (StrategyDualLP.sol#823) lacks a zero-check
on :
                - strategist = _strategist (StrategyDualLP.sol#829)
StratManagerLP.constructor(address,address,address,address,address)._unirouter (StrategyDualLP.sol#824) lacks a zero-check o
n :
                - unirouter = _unirouter (StrategyDualLP.sol#830)
StratManagerLP.constructor(address,address,address,address,address)._vault (StrategyDualLP.sol#825) lacks a zero-check on :
                - vault = _vault (StrategyDualLP.sol#831)
StratManagerLP.constructor(address,address,address,address,address)._toshaFeeRecipient (StrategyDualLP.sol#826) lacks a zero
-check on :
                - toshaFeeRecipient = _toshaFeeRecipient (StrategyDualLP.sol#832)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Variable 'StrategyCommonLP.callReward().amountOut (StrategyDualLP.sol#1102)' in StrategyCommonLP.callReward() (StrategyDualL
P.sol#1097-1110) potentially used before declaration: nativeOut = amountOut[amountOut.length - 1] (StrategyDualLP.sol#1104)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables
INFO:Detectors:
Reentrancy in StrategyCommonLP._harvest(address) (StrategyDualLP.sol#1012-1023):
        External calls:
        - IMasterChef(chef).deposit(poolId,0) (StrategyDualLP.sol#1013)
        - chargeFees(callFeeRecipient) (StrategyDualLP.sol#1017)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (StrategyDualLP.sol#508)
                - IUniswapRouterETH(unirouter).swapExactTokensForTokens(toNative,0,outputToNativeRoute,address(this),block.t
imestamp) (StrategyDualLP.sol#1027)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Address.isContract(address) (StrategyDualLP.sol#94-103) uses assembly
        - INLINE ASM (StrategyDualLP.sol#101)
Address._verifyCallResult(bool,bytes,string) (StrategyDualLP.sol#239-256) uses assembly
        - INLINE ASM (StrategyDualLP.sol#248-251)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (StrategyDualLP.sol#147-149) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (StrategyDualLP.sol#172-174) is never used and should be removed
Address.functionDelegateCall(address,bytes) (StrategyDualLP.sol#221-223) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (StrategyDualLP.sol#231-237) is never used and should be removed
Address.sendValue(address,uint256) (StrategyDualLP.sol#121-127) is never used and should be removed
Context._msgData() (StrategyDualLP.sol#667-670) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (StrategyDualLP.sol#492-495) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (StrategyDualLP.sol#487-490) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (StrategyDualLP.sol#465-467) is never used and should be removed
SafeMath.div(uint256,uint256,string) (StrategyDualLP.sol#431-434) is never used and should be removed
SafeMath.mod(uint256,uint256) (StrategyDualLP.sol#393-396) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (StrategyDualLP.sol#451-454) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (StrategyDualLP.sol#411-414) is never used and should be removed
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
FeeManagerLP.toshaFee (StrategyDualLP.sol#890) is set pre-construction with a non-constant function or state variable:
        - MAX_FEE - STRATEGIST_FEE - callFee
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state-variables
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (StrategyDualLP.sol#121-127):
        - (success) = recipient.call{value: amount}() (StrategyDualLP.sol#125)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (StrategyDualLP.sol#182-189):
        - (success,returndata) = target.call{value: value}(data) (StrategyDualLP.sol#187)
Low level call in Address.functionStaticCall(address,bytes,string) (StrategyDualLP.sol#207-213):
        - (success,returndata) = target.staticcall(data) (StrategyDualLP.sol#211)
Low level call in Address.functionDelegateCall(address,bytes,string) (StrategyDualLP.sol#231-237):
        - (success,returndata) = target.delegatecall(data) (StrategyDualLP.sol#235)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter StratManagerLP.setKeeper(address)._keeper (StrategyDualLP.sol#843) is not in mixedCase
Parameter StratManagerLP.setStrategist(address)._strategist (StrategyDualLP.sol#850) is not in mixedCase
Parameter StratManagerLP.setUnirouter(address)._unirouter (StrategyDualLP.sol#858) is not in mixedCase
Parameter StratManagerLP.setVault(address)._vault (StrategyDualLP.sol#865) is not in mixedCase
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (StrategyDualLP.sol#668)" inContext (StrategyDualLP.sol#662-671)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable IUniswapRouterETH.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Str
ategyDualLP.sol#538) is too similar to IUniswapRouterETH.addLiquidity(address,address,uint256,uint256,uint256,uint256,addres
s,uint256).amountBDesired (StrategyDualLP.sol#539)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
GasThrottler.gasprice (StrategyDualLP.sol#652) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (StrategyDualLP.sol#708-711)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (StrategyDualLP.sol#717-721)
setCallFee(uint256) should be declared external:
        - FeeManagerLP.setCallFee(uint256) (StrategyDualLP.sol#891-896)
callReward() should be declared external:
        - StrategyCommonLP.callReward() (StrategyDualLP.sol#1097-1110)
panic() should be declared external:
        - StrategyCommonLP.panic() (StrategyDualLP.sol#1125-1128)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:StrategyDualLP.sol analyzed (20 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 >> StrategyTosha.sol

```
INFO:Detectors:
StrategyTosha.chargeFees(address).callFee (StrategyTosha.sol#923) shadows:
        - FeeManager.callFee (StrategyTosha.sol#796) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
StratManager.setKeeper(address) (StrategyTosha.sol#760-762) should emit an event for:
        - keeper = _keeper (StrategyTosha.sol#761)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
FeeManager.setCallFee(uint256) (StrategyTosha.sol#798-802) should emit an event for:
        - callFee = _fee (StrategyTosha.sol#801)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
StratManager.constructor(address,address,address)._keeper (StrategyTosha.sol#741) lacks a zero-check on :
                - keeper = _keeper (StrategyTosha.sol#745)
StratManager.constructor(address,address,address)._unirouter (StrategyTosha.sol#742) lacks a zero-check on :
                - unirouter = _unirouter (StrategyTosha.sol#746)
StratManager.constructor(address,address,address)._vault (StrategyTosha.sol#743) lacks a zero-check on :
                - vault = _vault (StrategyTosha.sol#747)
StratManager.setKeeper(address)._keeper (StrategyTosha.sol#760) lacks a zero-check on :
                - keeper = _keeper (StrategyTosha.sol#761)
StratManager.setUnirouter(address)._unirouter (StrategyTosha.sol#768) lacks a zero-check on :
                - unirouter = _unirouter (StrategyTosha.sol#769)
StratManager.setVault(address)._vault (StrategyTosha.sol#776) lacks a zero-check on :
                - vault = _vault (StrategyTosha.sol#777)
StrategyTosha.constructor(address,address,address,address,address)._want (StrategyTosha.sol#839) lacks a zero-check on :
                - want = _want (StrategyTosha.sol#845)
StrategyTosha.constructor(address,address,address,address,address)._rewardPool (StrategyTosha.sol#840) lacks a zero-check on
 :
                - rewardPool = _rewardPool (StrategyTosha.sol#846)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in StrategyTosha._harvest(address) (StrategyTosha.sol#905-919):
        External calls:
        - IRewardPool(rewardPool).getReward() (StrategyTosha.sol#906)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Address.isContract(address) (StrategyTosha.sol#163-172) uses assembly
        - INLINE ASM (StrategyTosha.sol#170)
Address._verifyCallResult(bool,bytes,string) (StrategyTosha.sol#308-325) uses assembly
        - INLINE ASM (StrategyTosha.sol#317-320)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (StrategyTosha.sol#216-218) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (StrategyTosha.sol#241-243) is never used and should be removed
Address.functionDelegateCall(address,bytes) (StrategyTosha.sol#290-292) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (StrategyTosha.sol#300-306) is never used and should be removed
Address.functionStaticCall(address,bytes) (StrategyTosha.sol#266-268) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (StrategyTosha.sol#276-282) is never used and should be removed
Address.sendValue(address,uint256) (StrategyTosha.sol#190-196) is never used and should be removed
Context._msgData() (StrategyTosha.sol#590-593) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (StrategyTosha.sol#561-564) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (StrategyTosha.sol#556-559) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (StrategyTosha.sol#534-536) is never used and should be removed
SafeMath.div(uint256,uint256,string) (StrategyTosha.sol#500-503) is never used and should be removed
SafeMath.mod(uint256,uint256) (StrategyTosha.sol#462-465) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (StrategyTosha.sol#520-523) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (StrategyTosha.sol#480-483) is never used and should be removed
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (StrategyTosha.sol#591)" inContext (StrategyTosha.sol#585-594)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable IUniswapRouterETH.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Str
ategyTosha.sol#19) is too similar to IUniswapRouterETH.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,
uint256).amountBDesired (StrategyTosha.sol#20)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
StratManager.strategist (StrategyTosha.sol#730) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (StrategyTosha.sol#631-634)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (StrategyTosha.sol#640-644)
setCallFee(uint256) should be declared external:
        - FeeManager.setCallFee(uint256) (StrategyTosha.sol#798-802)
callReward() should be declared external:
        - StrategyTosha.callReward() (StrategyTosha.sol#952-954)
panic() should be declared external:
        - StrategyTosha.panic() (StrategyTosha.sol#977-980)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:StrategyTosha.sol analyzed (12 contracts with 75 detectors), 61 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> ToshaVault.sol

```
INFO:Detectors:
ToshaVault.constructor(IStrategy,string,string,uint256)._name (ToshaVault.sol#958) shadows:
        - ERC20._name (ToshaVault.sol#649) (state variable)
ToshaVault.constructor(IStrategy,string,string,uint256)._symbol (ToshaVault.sol#959) shadows:
        - ERC20._symbol (ToshaVault.sol#650) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Reentrancy in ToshaVault.deposit(uint256) (ToshaVault.sol#1011-1030):
        External calls:
        - strategy.beforeDeposit() (ToshaVault.sol#1012)
        - want().safeTransferFrom(msg.sender,address(this),_amount) (ToshaVault.sol#1015)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
ToshaVault.getPricePerFullShare() (ToshaVault.sol#996-998) uses timestamp for comparisons
        Dangerous comparisons:
        - totalSupply() == 0 (ToshaVault.sol#997)
ToshaVault.deposit(uint256) (ToshaVault.sol#1011-1030) uses timestamp for comparisons
        Dangerous comparisons:
        - totalSupply() == 0 (ToshaVault.sol#1020)
ToshaVault.withdraw(uint256) (ToshaVault.sol#1054-1075) uses timestamp for comparisons
        Dangerous comparisons:
        - b < r (ToshaVault.sol#1061)
        - _diff < _withdraw (ToshaVault.sol#1066)
ToshaVault.upgradeStrat() (ToshaVault.sol#1097-1109) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(stratCandidate.implementation != address(0),There is no candidate) (ToshaVault.sol#1098)
        - require(bool,string)(stratCandidate.proposedTime.add(approvalDelay) < block.timestamp,Delay has not passed) (Tosha
Vault.sol#1099)
ToshaVault.inCaseTokensGetStuck(address) (ToshaVault.sol#1115-1120) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(_token != address(want()),!token) (ToshaVault.sol#1116)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (ToshaVault.sol#114-123) uses assembly
        - INLINE ASM (ToshaVault.sol#121)
Address._verifyCallResult(bool,bytes,string) (ToshaVault.sol#259-276) uses assembly
        - INLINE ASM (ToshaVault.sol#268-271)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (ToshaVault.sol#167-169) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (ToshaVault.sol#192-194) is never used and should be removed
Address.functionDelegateCall(address,bytes) (ToshaVault.sol#241-243) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (ToshaVault.sol#251-257) is never used and should be removed
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter ToshaVault.deposit(uint256)._amount (ToshaVault.sol#1011) is not in mixedCase
Parameter ToshaVault.withdraw(uint256)._shares (ToshaVault.sol#1054) is not in mixedCase
Parameter ToshaVault.proposeStrat(address)._implementation (ToshaVault.sol#1081) is not in mixedCase
Parameter ToshaVault.inCaseTokensGetStuck(address)._token (ToshaVault.sol#1115) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (ToshaVault.sol#584)" inContext (ToshaVault.sol#578-587)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
ToshaVault.upgradeStrat() (ToshaVault.sol#1097-1109) uses literals with too many digits:
        - stratCandidate.proposedTime = 5000000000 (ToshaVault.sol#1106)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (ToshaVault.sol#624-627)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (ToshaVault.sol#633-637)
name() should be declared external:
        - ERC20.name() (ToshaVault.sol#671-673)
symbol() should be declared external:
        - ERC20.symbol() (ToshaVault.sol#679-681)
decimals() should be declared external:
        - ERC20.decimals() (ToshaVault.sol#696-698)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (ToshaVault.sol#722-725)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (ToshaVault.sol#730-732)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (ToshaVault.sol#741-744)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (ToshaVault.sol#759-763)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (ToshaVault.sol#777-780)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (ToshaVault.sol#796-799)
```

```
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (ToshaVault.sol#796-799)
getPricePerFullShare() should be declared external:
        - ToshaVault.getPricePerFullShare() (ToshaVault.sol#996-998)
proposeStrat(address) should be declared external:
        - ToshaVault.proposeStrat(address) (ToshaVault.sol#1081-1089)
upgradeStrat() should be declared external:
        - ToshaVault.upgradeStrat() (ToshaVault.sol#1097-1109)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:ToshaVault.sol analyzed (10 contracts with 75 detectors), 59 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> ToshaLPVault.sol

```
INFO:Detectors:
ToshaLPVault.constructor(IVault,IStrategy,string,string,uint256)._name (ToshaLPVault.sol#977) shadows:
        - ERC20._name (ToshaLPVault.sol#658) (state variable)
ToshaLPVault.constructor(IVault,IStrategy,string,string,uint256)._symbol (ToshaLPVault.sol#978) shadows:
        - ERC20._symbol (ToshaLPVault.sol#659) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Reentrancy in ToshaLPVault.deposit(uint256) (ToshaLPVault.sol#1033-1055):
        External calls:
        - strategy.beforeDeposit() (ToshaLPVault.sol#1034)
        - want().safeTransferFrom(msg.sender,address(this),_amount) (ToshaLPVault.sol#1037)
        - earn() (ToshaLPVault.sol#1038)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (ToshaLPVault.sol#537)
                - want().safeTransfer(address(strategy),_bal) (ToshaLPVault.sol#1063)
                - strategy.deposit() (ToshaLPVault.sol#1064)
                - (success,returndata) = target.call{value: value}(data) (ToshaLPVault.sol#216)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in ToshaLPVault.deposit(uint256) (ToshaLPVault.sol#1033-1055):
        External calls:
        - strategy.beforeDeposit() (ToshaLPVault.sol#1034)
        - want().safeTransferFrom(msg.sender,address(this),_amount) (ToshaLPVault.sol#1037)
        - earn() (ToshaLPVault.sol#1038)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (ToshaLPVault.sol#537)
                - want().safeTransfer(address(strategy),_bal) (ToshaLPVault.sol#1063)
                - strategy.deposit() (ToshaLPVault.sol#1064)
                - (success,returndata) = target.call{value: value}(data) (ToshaLPVault.sol#216)
        External calls sending eth:
```

```
Parameter ToshaLPVault.withdraw(uint256)._shares (ToshaLPVault.sol#1079) is not in mixedCase
Parameter ToshaLPVault.notifyRewards(uint256)._totalStake (ToshaLPVault.sol#1140) is not in mixedCase
Parameter ToshaLPVault.proposeStrat(address)._implementation (ToshaLPVault.sol#1166) is not in mixedCase
Parameter ToshaLPVault.inCaseTokensGetStuck(address)._token (ToshaLPVault.sol#1200) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (ToshaLPVault.sol#593)" inContext (ToshaLPVault.sol#587-596)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
ToshaLPVault.upgradeStrat() (ToshaLPVault.sol#1182-1194) uses literals with too many digits:
        - stratCandidate.proposedTime = 5000000000 (ToshaLPVault.sol#1191)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (ToshaLPVault.sol#633-636)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (ToshaLPVault.sol#642-646)
name() should be declared external:
        - ERC20.name() (ToshaLPVault.sol#680-682)
symbol() should be declared external:
        - ERC20.symbol() (ToshaLPVault.sol#688-690)
decimals() should be declared external:
        - ERC20.decimals() (ToshaLPVault.sol#705-707)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (ToshaLPVault.sol#731-734)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (ToshaLPVault.sol#739-741)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (ToshaLPVault.sol#750-753)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (ToshaLPVault.sol#768-772)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (ToshaLPVault.sol#786-789)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (ToshaLPVault.sol#805-808)
getPricePerFullShare() should be declared external:
        - ToshaLPVault.getPricePerFullShare() (ToshaLPVault.sol#1018-1020)
```

```
getPricePerFullShare() should be declared external:
        - ToshaLPVault.getPricePerFullShare() (ToshaLPVault.sol#1018-1020)
proposeStrat(address) should be declared external:
        - ToshaLPVault.proposeStrat(address) (ToshaLPVault.sol#1166-1174)
upgradeStrat() should be declared external:
        - ToshaLPVault.upgradeStrat() (ToshaLPVault.sol#1182-1194)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:ToshaLPVault.sol analyzed (11 contracts with 75 detectors), 68 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
root@server:/chotan/gaza/mycontracts# slither MasterChef.sol
```

Slither log >> Materchef.sol

```
INFO:Detectors:
TOSHA.setGovernance(address)._governance (MasterChef.sol#1087) lacks a zero-check on :
                - governance = _governance (MasterChef.sol#1089)
MasterChef.constructor(TOSHA,address,address,address,uint256,uint256)._reserveFundsAddress (MasterChef.sol#1139) lacks a zer
o-check on :
                - reserveFundsAddress = _reserveFundsAddress (MasterChef.sol#1150)
MasterChef.constructor(TOSHA,address,address,address,uint256,uint256)._farmingRewarderAddress (MasterChef.sol#1140) lacks a
zero-check on :
                - farmingRewarderAddress = _farmingRewarderAddress (MasterChef.sol#1151)
MasterChef.constructor(TOSHA,address,address,address,uint256,uint256)._rewardPoolAddress (MasterChef.sol#1141) lacks a zero-
check on :
                - rewardPoolAddress = _rewardPoolAddress (MasterChef.sol#1152)
MasterChef.updateReserveFundsAddress(address)._reserveFundsAddress (MasterChef.sol#1184) lacks a zero-check on :
                - reserveFundsAddress = _reserveFundsAddress (MasterChef.sol#1185)
MasterChef.updateFarmingRewarderAddress(address)._farmingRewarderAddress (MasterChef.sol#1189) lacks a zero-check on :
                - farmingRewarderAddress = _farmingRewarderAddress (MasterChef.sol#1190)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Address.isContract(address) (MasterChef.sol#311-320) uses assembly
        - INLINE ASM (MasterChef.sol#318)
Address._verifyCallResult(bool,bytes,string) (MasterChef.sol#456-473) uses assembly
        - INLINE ASM (MasterChef.sol#465-468)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._verifyCallResult(bool,bytes,string) (MasterChef.sol#456-473) is never used and should be removed
Address.functionCall(address,bytes) (MasterChef.sol#364-366) is never used and should be removed
Address.functionCall(address,bytes,string) (MasterChef.sol#374-376) is never used and should be removed
```

```
SafeMath.sub(uint256,uint256) (MasterChef.sol#559-562) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (MasterChef.sol#482-486) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (MasterChef.sol#518-521) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (MasterChef.sol#528-531) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (MasterChef.sol#503-511) is never used and should be removed
SafeMath.trySub(uint256,uint256) (MasterChef.sol#493-496) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
```

```
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (MasterChef.sol#338-344):
        - (success) = recipient.call{value: amount}() (MasterChef.sol#342)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (MasterChef.sol#399-406):
        - (success,returndata) = target.call{value: value}(data) (MasterChef.sol#404)
Low level call in Address.functionStaticCall(address,bytes,string) (MasterChef.sol#424-430):
        - (success,returndata) = target.staticcall(data) (MasterChef.sol#428)
Low level call in Address.functionDelegateCall(address,bytes,string) (MasterChef.sol#448-454):
        - (success,returndata) = target.delegatecall(data) (MasterChef.sol#452)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter TOSHA.setGovernance(address)._governance (MasterChef.sol#1087) is not in mixedCase
Parameter TOSHA.addMinter(address)._minter (MasterChef.sol#1092) is not in mixedCase
Parameter TOSHA.removeMinter(address)._minter (MasterChef.sol#1097) is not in mixedCase
Parameter MasterChef.updateReserveFundsAddress(address)._reserveFundsAddress (MasterChef.sol#1184) is not in mixedCase
Parameter MasterChef.updateFarmingRewarderAddress(address)._farmingRewarderAddress (MasterChef.sol#1189) is not in mixedCase
Parameter MasterChef.updateEmissionRate(uint256)._tokensPerBlock (MasterChef.sol#1193) is not in mixedCase
Parameter MasterChef.updateRewardsRate(uint256)._rewardsPerBlock (MasterChef.sol#1202) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (MasterChef.sol#739)" inContext (MasterChef.sol#733-742)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (MasterChef.sol#779-782)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (MasterChef.sol#788-792)
name() should be declared external:
        - ERC20.name() (MasterChef.sol#826-828)
symbol() should be declared external:
        - ERC20.symbol() (MasterChef.sol#834-836)
decimals() should be declared external:
        - ERC20.decimals() (MasterChef.sol#851-853)
totalSupply() should be declared external:
```

```
        - ERC20.decimals() (MasterChef.sol#851-853)
totalSupply() should be declared external:
        - ERC20.totalSupply() (MasterChef.sol#858-860)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (MasterChef.sol#865-867)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (MasterChef.sol#877-880)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (MasterChef.sol#885-887)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (MasterChef.sol#896-899)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (MasterChef.sol#914-918)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (MasterChef.sol#932-935)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (MasterChef.sol#951-954)
mint(address,uint256) should be declared external:
        - TOSHA.mint(address,uint256) (MasterChef.sol#1082-1085)
setGovernance(address) should be declared external:
        - TOSHA.setGovernance(address) (MasterChef.sol#1087-1090)
addMinter(address) should be declared external:
        - TOSHA.addMinter(address) (MasterChef.sol#1092-1095)
removeMinter(address) should be declared external:
        - TOSHA.removeMinter(address) (MasterChef.sol#1097-1100)
harvest() should be declared external:
        - MasterChef.harvest() (MasterChef.sol#1158-1181)
updateReserveFundsAddress(address) should be declared external:
        - MasterChef.updateReserveFundsAddress(address) (MasterChef.sol#1184-1186)
updateFarmingRewarderAddress(address) should be declared external:
        - MasterChef.updateFarmingRewarderAddress(address) (MasterChef.sol#1189-1191)
updateEmissionRate(uint256) should be declared external:
        - MasterChef.updateEmissionRate(uint256) (MasterChef.sol#1193-1200)
updateRewardsRate(uint256) should be declared external:
        - MasterChef.updateRewardsRate(uint256) (MasterChef.sol#1202-1209)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:MasterChef.sol analyzed (10 contracts with 75 detectors), 89 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Tosha.sol

```
INFO:Detectors:
TOSHA.setGovernance(address)._governance (Tosha.sol#821) lacks a zero-check on :
                - governance = _governance (Tosha.sol#823)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Address.isContract(address) (Tosha.sol#95-104) uses assembly
        - INLINE ASM (Tosha.sol#102)
Address._verifyCallResult(bool,bytes,string) (Tosha.sol#240-257) uses assembly
        - INLINE ASM (Tosha.sol#249-252)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._verifyCallResult(bool,bytes,string) (Tosha.sol#240-257) is never used and should be removed
Address.functionCall(address,bytes) (Tosha.sol#148-150) is never used and should be removed
Address.functionCall(address,bytes,string) (Tosha.sol#158-160) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (Tosha.sol#173-175) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (Tosha.sol#183-190) is never used and should be removed
Address.functionDelegateCall(address,bytes) (Tosha.sol#222-224) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (Tosha.sol#232-238) is never used and should be removed
Address.functionStaticCall(address,bytes) (Tosha.sol#198-200) is never used and should be removed
```

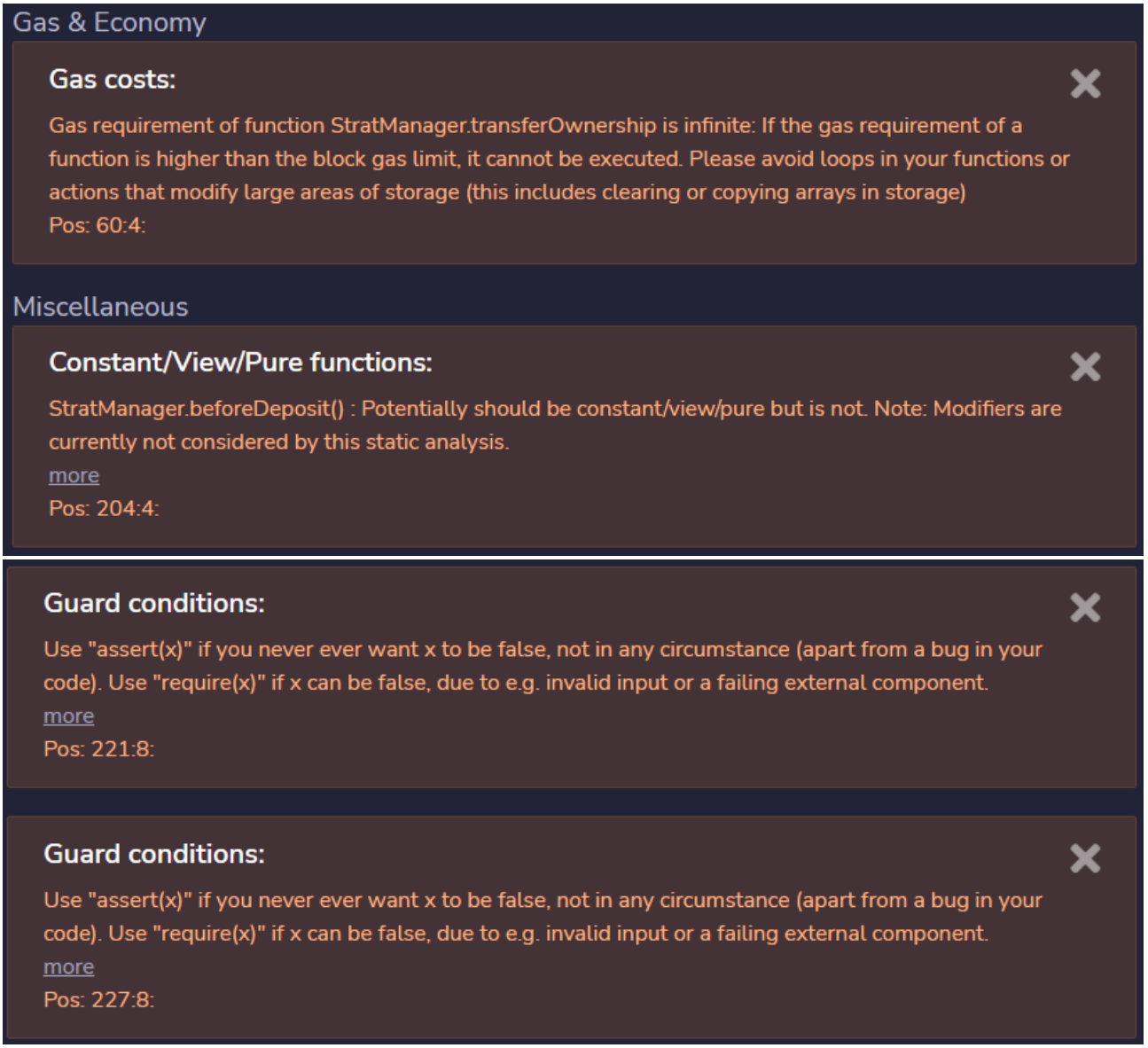
```
Context._msgData() (Tosha.sol#522-525) is never used and should be removed
ERC20._burn(address,uint256) (Tosha.sol#744-752) is never used and should be removed
ERC20._setupDecimals(uint8) (Tosha.sol#782-784) is never used and should be removed
SafeERC20._callOptionalReturn(IERC20,bytes) (Tosha.sol#504-514) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (Tosha.sol#477-486) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (Tosha.sol#493-496) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (Tosha.sol#488-491) is never used and should be removed
SafeERC20.safeTransfer(IERC20,address,uint256) (Tosha.sol#462-464) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (Tosha.sol#466-468) is never used and should be removed
SafeMath.div(uint256,uint256) (Tosha.sol#377-380) is never used and should be removed
SafeMath.div(uint256,uint256,string) (Tosha.sol#432-435) is never used and should be removed
SafeMath.mod(uint256,uint256) (Tosha.sol#394-397) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Tosha.sol#452-455) is never used and should be removed
SafeMath.mul(uint256,uint256) (Tosha.sol#358-363) is never used and should be removed
SafeMath.sub(uint256,uint256) (Tosha.sol#343-346) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (Tosha.sol#266-270) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (Tosha.sol#302-305) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (Tosha.sol#312-315) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (Tosha.sol#287-295) is never used and should be removed
SafeMath.trySub(uint256,uint256) (Tosha.sol#277-280) 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) (Tosha.sol#122-128):
        - (success) = recipient.call{value: amount}() (Tosha.sol#126)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (Tosha.sol#183-190):
        - (success,returndata) = target.call{value: value}(data) (Tosha.sol#188)
Low level call in Address.functionStaticCall(address,bytes,string) (Tosha.sol#208-214):
        - (success,returndata) = target.staticcall(data) (Tosha.sol#212)
Low level call in Address.functionDelegateCall(address,bytes,string) (Tosha.sol#232-238):
        - (success,returndata) = target.delegatecall(data) (Tosha.sol#236)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter TOSHA.setGovernance(address)._governance (Tosha.sol#821) is not in mixedCase
Parameter TOSHA.addMinter(address)._minter (Tosha.sol#826) is not in mixedCase
Parameter TOSHA.removeMinter(address)._minter (Tosha.sol#831) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (Tosha.sol#523)" inContext (Tosha.sol#517-526)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (Tosha.sol#559-561)
symbol() should be declared external:
        - ERC20.symbol() (Tosha.sol#567-569)
decimals() should be declared external:
        - ERC20.decimals() (Tosha.sol#584-586)
totalSupply() should be declared external:
        - ERC20.totalSupply() (Tosha.sol#591-593)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (Tosha.sol#598-600)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (Tosha.sol#610-613)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (Tosha.sol#618-620)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (Tosha.sol#629-632)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (Tosha.sol#647-651)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (Tosha.sol#665-668)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (Tosha.sol#684-687)
mint(address,uint256) should be declared external:
        - TOSHA.mint(address,uint256) (Tosha.sol#816-819)
setGovernance(address) should be declared external:
        - TOSHA.setGovernance(address) (Tosha.sol#821-824)
addMinter(address) should be declared external:
        - TOSHA.addMinter(address) (Tosha.sol#826-829)
removeMinter(address) should be declared external:
        - TOSHA.removeMinter(address) (Tosha.sol#831-834)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

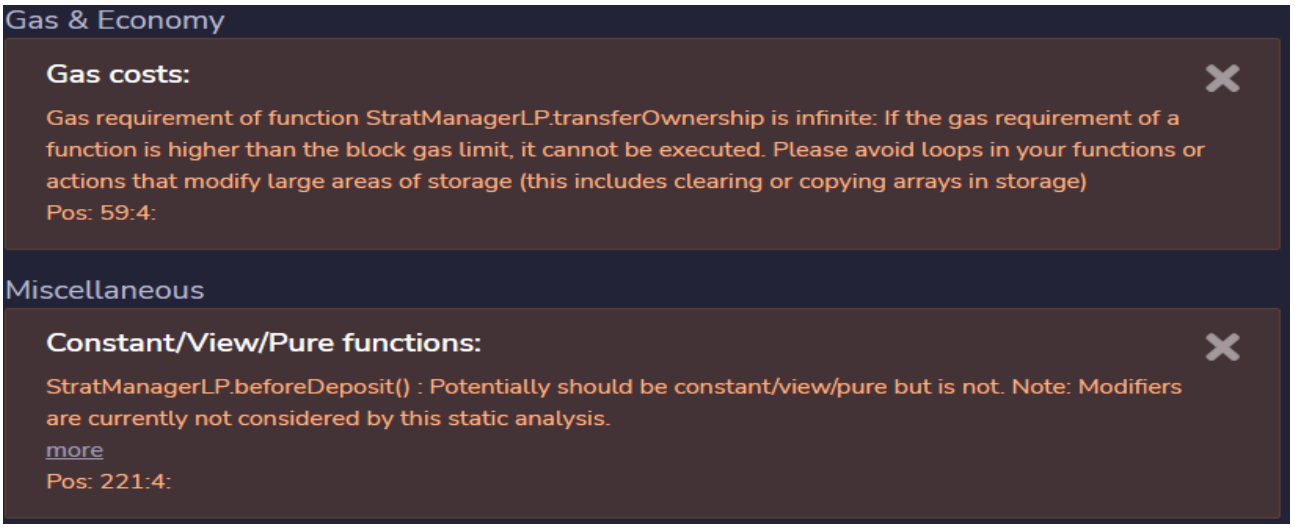
```
INFO:Slither:Tosha.sol analyzed (7 contracts with 75 detectors), 57 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

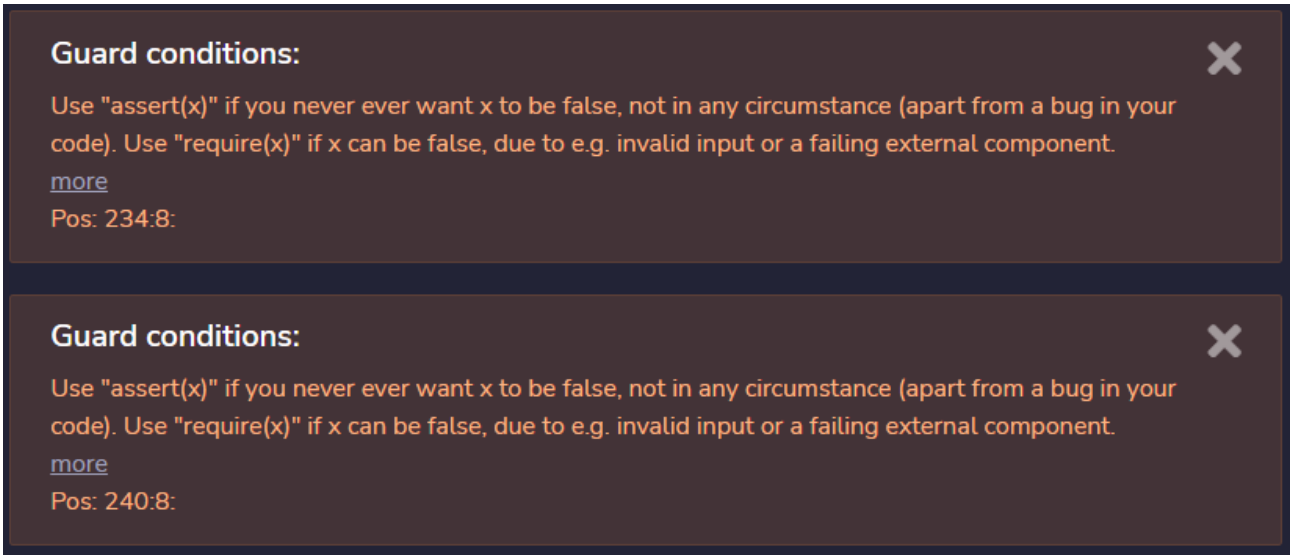
Solidity Static Analysis

FeeManager.sol

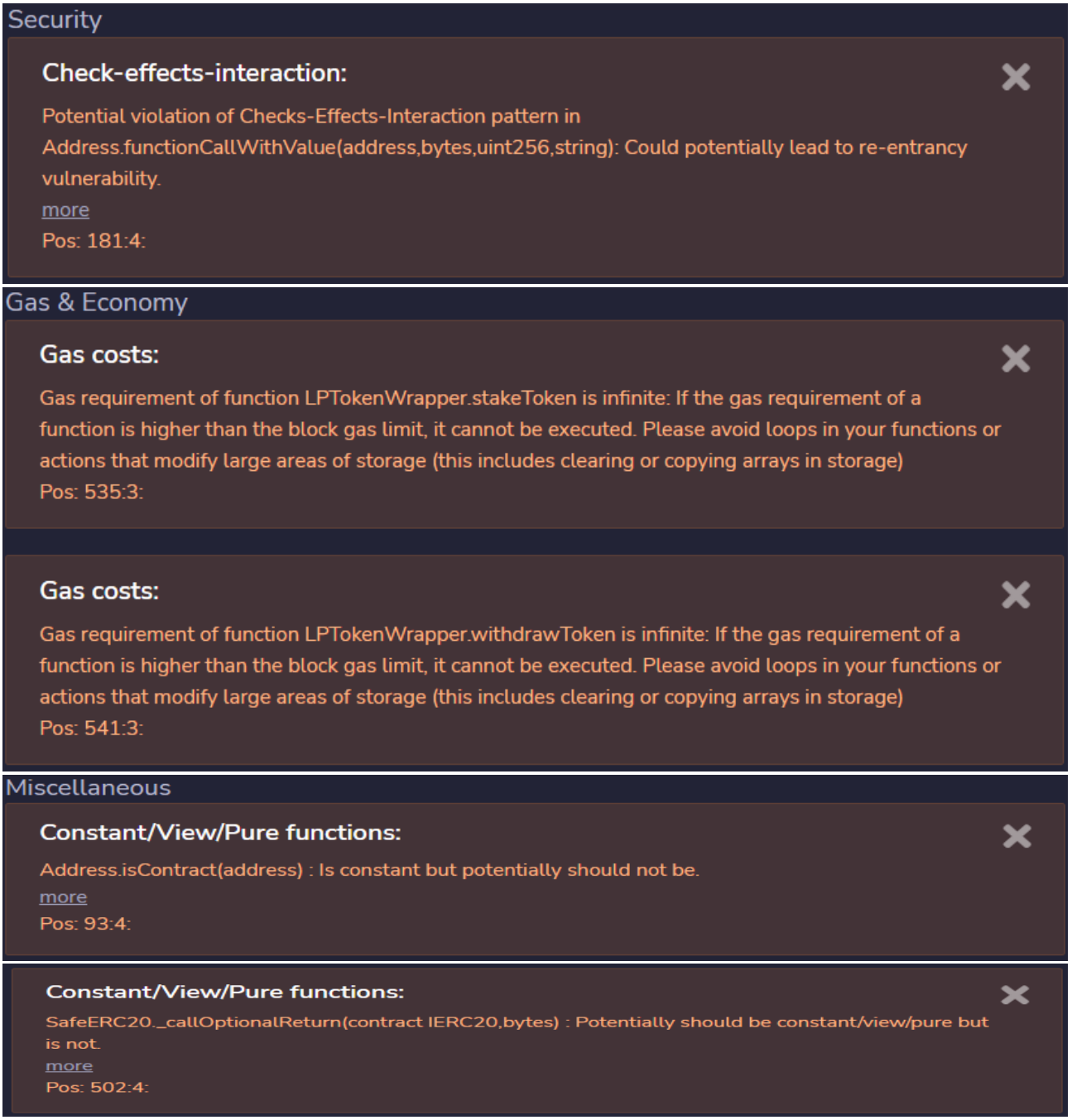


FeeManagerLP.sol





LPTokenWrapper.sol



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: 510:12:

StratManager.sol

Gas & Economy

Gas costs:

Gas requirement of function StratManager.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: 60:4:

Miscellaneous

Constant/View/Pure functions:

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

more

Pos: 204:4:

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: 172:8:

StratManagerLP.sol

Gas & Economy

Gas costs:

Gas requirement of function StratManagerLP.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: 59:4:

Miscellaneous

Constant/View/Pure functions:

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

more

Pos: 221:4:

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: 178:8:

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: 193:8:

StrategyCommonLP.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: 980:12:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in StrategyCommonLP.withdraw(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 968:4:

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: 1020:111:

Gas & Economy

Gas costs:

Gas requirement of function StratManagerLP.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: 717:4:

Gas costs:

Gas requirement of function StrategyCommonLP.beforeDeposit 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: 989:4:

Miscellaneous

Constant/View/Pure functions:

Address.isContract(address) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 94:4:

Constant/View/Pure functions:

StrategyCommonLP._removeAllowances() : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1134:4:

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: 955:8:

StrategyDualLP.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: 986:12:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in StrategyCommonLP._harvest(address): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1012:4:

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: 1031:114:

Gas & Economy

Gas costs:

Gas requirement of function StratManagerLP.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: 717:4:

Gas costs:

Gas requirement of function StrategyCommonLP.harvestWithCallFeeRecipient 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: 1005:4:

Miscellaneous

Constant/View/Pure functions:

Address.isContract(address) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 94:4:

Constant/View/Pure functions:

StrategyCommonLP._removeAllowances() : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1146:4:

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: 1074:8:

StrategyTosha.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: 879:12:

Check-effects-interaction:

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

more

Pos: 905:4:

Gas & Economy

Gas costs:

Gas requirement of function StratManager.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: 640:4:

Gas costs:

Gas requirement of function StrategyTosha.beforeDeposit 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: 889:4:

Miscellaneous

Constant/View/Pure functions:

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

more

Pos: 163:4:

Constant/View/Pure functions:

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

more

Pos: 996:4:

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: 968:8:

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: 502:15:

ToshaVault.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: 202:4:

Check-effects-interaction:

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

more

Pos: 1097:4:

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: 1099:65:

Gas & Economy

Gas costs:

Gas requirement of function ToshaVault.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: 633:4:

Gas costs:

Gas requirement of function ToshaVault.inCaseTokensGetStuck 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: 1115:4:

Miscellaneous

Constant/View/Pure functions:

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

more

Pos: 114:4:

Constant/View/Pure functions:

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

more

Pos: 1115:4:

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: 1116:8:

ToshaLPVault.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: 211:4:

Check-effects-interaction:

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

more

Pos: 1140:4:

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: 1184:65:

Gas & Economy

Gas costs:

Gas requirement of function ToshaLPVault.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: 642:4:

Gas costs:

Gas requirement of function ToshaLPVault.inCaseTokensGetStuck 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: 1200:4:

Miscellaneous

Constant/View/Pure functions:

Address.isContract(address) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 123:4:

Constant/View/Pure functions:

ToshaLPVault.inCaseTokensGetStuck(address) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1200:4:

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: 1141:8:

Materchef.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: 399:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in MasterChef.harvest(): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1158:3:

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: 826:4:

Gas costs:

Gas requirement of function MasterChef.updateRewardsRate 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: 1202:4:

Miscellaneous

Constant/View/Pure functions:

EnumerableSet.contains(struct EnumerableSet.AddressSet,address) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 141:4:

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: 1067:4:

Similar variable names:

ERC20._burn(address,uint256) : Variables have very similar names "account" and "amount". Note: Modifiers are currently not considered by this static analysis.
Pos: 1016:39:

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: 1145:8:

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: 1166:36:

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: 1167:37:

Tosha.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in SafeERC20.safeDecreaseAllowance(contract IERC20,address,uint256): Could potentially lead to re-entrancy vulnerability.

more

Pos: 493:4:

Gas & Economy

Gas costs:

Gas requirement of function TOSHA.mint 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: 816:4:

Miscellaneous

Constant/View/Pure functions:

ERC20._beforeTokenTransfer(address,address,uint256) : Potentially should be constant/view/pure but is not.

more

Pos: 800:4:

Similar variable names:

TOSHA.removeMinter(address) : Variables have very similar names "minters" and "_minter".

Pos: 833:16:

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: 832:8:

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: 434:15:

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: 822:8:

Email: audit@EtherAuthority.io

Solhint Linter

FeeManager.sol

```
FeeManager.sol:3:1: Error: Compiler version 0.6.12 does not satisfy
the r semver requirement
FeeManager.sol:204:47: Error: Code contains empty blocks
```

FeeManagerLP.sol

```
FeeManagerLP.sol:3:1: Error: Compiler version 0.6.12 does not satisfy
the r semver requirement
FeeManagerLP.sol:221:47: Error: Code contains empty blocks
```

LPTokenWrapper.sol

```
LPTokenWrapper.sol:3:1: Error: Compiler version 0.6.12 does not
satisfy the r semver requirement
```

StratManager.sol

```
StratManager.sol:3:1: Error: Compiler version 0.6.12 does not satisfy
the r semver requirement
StratManager.sol:204:47: Error: Code contains empty blocks
```

StratManagerLP.sol

```
StratManagerLP.sol:3:1: Error: Compiler version 0.6.12 does not
satisfy the r semver requirement
StratManagerLP.sol:221:47: Error: Code contains empty blocks
```

StrategyCommonLP.sol

```
StrategyCommonLP.sol:3:1: Error: Compiler version 0.6.12 does not
satisfy the r semver requirement
StrategyCommonLP.sol:879:47: Error: Code contains empty blocks
StrategyCommonLP.sol:945:84: Error: Visibility modifier must be first
in list of modifiers
```

```
StrategyCommonLP.sol:980:13: Error: Avoid to use tx.origin
StrategyCommonLP.sol:992:22: Error: Avoid to use tx.origin
StrategyCommonLP.sol:997:18: Error: Avoid to use tx.origin
StrategyCommonLP.sol:1003:18: Error: Avoid to use tx.origin
StrategyCommonLP.sol:1013:27: Error: Avoid to make time-based
decisions in your business logic
StrategyCommonLP.sol:1020:112: Error: Avoid to make time-based
decisions in your business logic
StrategyCommonLP.sol:1033:117: Error: Avoid to make time-based
decisions in your business logic
StrategyCommonLP.sol:1094:19: Error: Code contains empty blocks
```

StrategyDualLP.sol

```
StrategyDualLP.sol:3:1: Error: Compiler version 0.6.12 does not
satisfy the r semver requirement
StrategyDualLP.sol:879:47: Error: Code contains empty blocks
StrategyDualLP.sol:947:84: Error: Visibility modifier must be first
in list of modifiers
StrategyDualLP.sol:986:13: Error: Avoid to use tx.origin
StrategyDualLP.sol:998:22: Error: Avoid to use tx.origin
StrategyDualLP.sol:1003:18: Error: Avoid to use tx.origin
StrategyDualLP.sol:1009:18: Error: Avoid to use tx.origin
StrategyDualLP.sol:1020:27: Error: Avoid to make time-based decisions
in your business logic
StrategyDualLP.sol:1027:112: Error: Avoid to make time-based
decisions in your business logic
StrategyDualLP.sol:1031:115: Error: Avoid to make time-based
decisions in your business logic
StrategyDualLP.sol:1045:117: Error: Avoid to make time-based
decisions in your business logic
StrategyDualLP.sol:1106:19: Error: Code contains empty blocks
```

StrategyTosha.sol

```
StrategyTosha.sol:3:1: Error: Compiler version 0.6.12 does not
satisfy the r semver requirement
StrategyTosha.sol:784:47: Error: Code contains empty blocks
StrategyTosha.sol:816:29: Error: Constant name must be in capitalized
SNAKE_CASE
StrategyTosha.sol:844:49: Error: Visibility modifier must be first in
list of modifiers
StrategyTosha.sol:879:13: Error: Avoid to use tx.origin
StrategyTosha.sol:892:22: Error: Avoid to use tx.origin
StrategyTosha.sol:897:18: Error: Avoid to use tx.origin
StrategyTosha.sol:916:27: Error: Avoid to make time-based decisions
in your business logic
StrategyTosha.sol:930:108: Error: Avoid to make time-based decisions
in your business logic
```

ToshaVault.sol

```
ToshaVault.sol:3:1: Error: Compiler version 0.6.12 does not satisfy
the r semver requirement
ToshaVault.sol:912:94: Error: Code contains empty blocks
ToshaVault.sol:1085:27: Error: Avoid to make time-based decisions in
your business logic
ToshaVault.sol:1099:66: Error: Avoid to make time-based decisions in
your business logic
```

ToshaLPVault.sol

```
ToshaLPVault.sol:3:1: Error: Compiler version 0.6.12 does not satisfy
the r semver requirement
ToshaLPVault.sol:921:94: Error: Code contains empty blocks
ToshaLPVault.sol:1052:34: Error: Avoid to make time-based decisions
in your business logic
ToshaLPVault.sol:1113:60: Error: Visibility modifier must be first in
list of modifiers
ToshaLPVault.sol:1170:27: Error: Avoid to make time-based decisions
in your business logic
ToshaLPVault.sol:1184:66: Error: Avoid to make time-based decisions
in your business logic
```

Materchef.sol

```
Materchef.sol:3:1: Error: Compiler version 0.6.12 does not satisfy
the r semver requirement
Materchef.sol:1067:94: Error: Code contains empty blocks
```

Tosha.sol

```
Tosha.sol:3:1: Error: Compiler version 0.6.12 does not satisfy the r
semver requirement
Tosha.sol:800:94: Error: Code contains empty blocks
```

Software analysis result:

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

Email: audit@EtherAuthority.io