Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project: Tally-ho Protocol
Website: app.tally-ho.org
Platform: Binance Smart Chain
Language: Solidity
Date: April 15th, 2022

Table of contents

Introduction 4
Project Background 4
Audit Scope 5
Claimed Smart Contract Features 6
Audit Summary 8
Technical Quick Stats 9
Code Quality 10
Documentation 10
Use of Dependencies 10
AS-IS overview 11
Severity Definitions 17
Audit Findings 18
Conclusion 24
Our Methodology 25
Disclaimers 27
Appendix

- Code Flow Diagram 28
- Slither Results Log 33
- Solidity static analysis 41
- Solhint Linter 50

Email: audit@EtherAuthority.io

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

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by Tally-ho to perform the Security audit of the Tally Exchange 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 15th, 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

Tally-ho Contracts will provide Swap Feature(with token giveaway feature), Add Liquidity, Staking, Farming, NFT Marketplace, Launchpad, Raffles, Expansion to other blockchains than BSC etc.

Email: audit@EtherAuthority.io

Audit scope

Name	Code Review and Security Analysis Report for Tally-ho Protocol Smart Contracts
Platform	**BSC / Solidity**
File 1	SwapFeeReward.sol
File 1 MD5 Hash	AFE1346EE64743E6B1D1307456177CCC
Updated File 1 MD5 Hash	02BC72F895435207989B3595A58860A5
File 2	Tally.sol
File 2 MD5 Hash	3BE52C97A80419E6091A9F9B6B60AA2D
Updated File 2 MD5 Hash	E970BA651DB98061496FFAAFAFCACA63
File 3	MasterChef.sol
File 3 MD5 Hash	622F7F36703DEA77BFE8271FB19A03B7
Updated File 3 MD5 Hash	585527118190E9344BEEB7B5679AC2C2
File 4	TallyswapFactory.sol
File 4 MD5 Hash	EBCAE118C5783194B8C60C2D9F9850B4
Updated File 4 MD5 Hash	9E21744DF216FFB55CC8A57E2ADEAFF5
File 5	TallyswapRouter.sol
File 5 MD5 Hash	9C6926868D119CF253C43F5505568213
Updated File 5 MD5 Hash	284ECE13B08AA3E63CFFDED96492E7A4
Audit Date	April 15th,2022

Email: audit@EtherAuthority.io

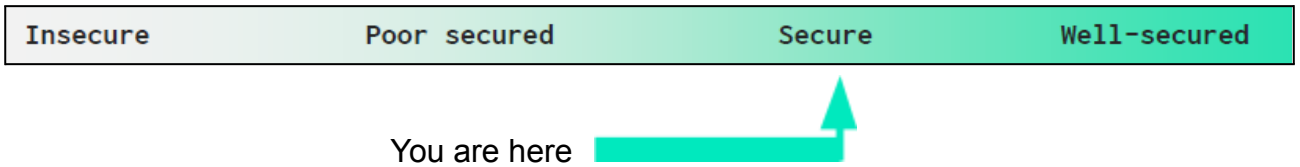
Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
File 1 SwapFeeReward.sol • Maximum Mining Amount: 100 Million • Maximum Mining In Phase: 5000 • Current Phase: 1	**YES, This is valid.**
File 2 Tally.sol • Name: Tally Token • Symbol: TALLY • Decimals: 9 • Tax Fee: 5% • Reflections for Selling Fee : 5% • Reflections for buying Fee: 3% • Liquidity Pool Fee: 3% • Burned Fee: 2% • Marketing: 3.66% • Charity Fee: 0.34% • Maximum Transaction Amount: 0.2% • BuyBack Fee: 0.5% • Raffle Fee: 5% • Pool Fee: 1% • Number Tokens Sell To Add To Liquidity: 500 Billion	**YES, This is valid.** **Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**
File 3 MasterChef.sol • percentDec: 1000000 • stakingPercent: 40% • Marketing Reserve: 0.15% • Platform Maintenance & Security: 0.008% • BUY BACK RESERVES: 0.1% • Operation Manager: 0.142% • TALLY Per Block: 30 Tally	**YES, This is valid.** **Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**

• Bonus multiplier: 1	
File 4 TallyswapFactory.sol • The TallyswapFactory contract can create pairs, set pair fees, etc.	**YES, This is valid.**
File 5 TallyswapRouter.sol • The TallyswapRouter contract can add new Liquidity, remove Liquidity, swap tokens, etc.	**YES, This is valid.**

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **"Secured"**. Also, 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, 1 medium and 1 low and some very low level issues.

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

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Passed
	Integer overflow/underflow	Passed
	Function input parameters lack of check	Passed
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Passed
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	N/A
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Passed
	Other programming issues	Moderated
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	Moderated
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: **PASSED**

Code Quality

This audit scope has 5 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 Tally-ho 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 Tally-ho Protocol.

The Tally-ho 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 Tally-ho 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.

Another source of information was its official website https://app.tally-ho.org which provided rich information about the project architecture and tokenomics.

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

SwapFeeReward.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Visibility for constructor is ignored	Refer Audit Findings
2	onlyRouter	modifier	Passed	No Issue
3	sortTokens	write	Passed	No Issue
4	pairFor	read	Passed	No Issue
5	getSwapFee	internal	Passed	No Issue
6	setPhase	write	access only Owner	No Issue
7	checkPairExist	read	Passed	No Issue
8	swap	write	access only Router	No Issue
9	rewardBalance	read	Passed	No Issue
10	permit	write	Passed	No Issue
11	withdraw	write	Passed	No Issue
12	getQuantity	read	Passed	No Issue
13	addWhitelist	write	access only Owner	No Issue
14	delWhitelist	write	access only Owner	No Issue
15	getWhitelistLength	read	Passed	No Issue
16	isWhitelist	read	Passed	No Issue
17	getWhitelist	read	Passed	No Issue
18	setRouter	write	access only Owner	No Issue
19	setOracle	write	access only Owner	No Issue
20	setFactory	write	access only Owner	No Issue
21	setInitCodeHash	write	access only Owner	No Issue
22	pairsListLength	read	Passed	No Issue
23	addPair	write	access only Owner	No Issue
24	setPair	write	access only Owner	No Issue
25	setPairEnabled	write	access only Owner	No Issue
26	owner	read	Passed	No Issue
27	isOwner	read	Passed	No Issue
28	renounceOwnership	write	access only Owner	No Issue
29	_transferOwnership	internal	Passed	No Issue
30	transferOwnership	write	access only Owner	No Issue
31	onlyOwner	modifier	Passed	No Issue

Tally.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Visibility for constructor is ignored	Refer Audit Findings
2	owner	read	Passed	No Issue
3	isOwner	read	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	_transferOwnership	internal	Passed	No Issue
6	transferOwnership	write	access only Owner	No Issue
7	onlyOwner	modifier	Passed	No Issue
8	lockTheSwap	modifier	Passed	No Issue
9	name	read	Passed	No Issue
10	symbol	read	Passed	No Issue
11	decimals	read	Passed	No Issue
12	totalSupply	read	Passed	No Issue
13	balanceOf	read	Passed	No Issue
14	transfer	write	Passed	No Issue
15	allowance	read	Passed	No Issue
16	approve	write	Passed	No Issue
17	transferFrom	write	Passed	No Issue
18	increaseAllowance	write	Passed	No Issue
19	decreaseAllowance	write	Passed	No Issue
20	isExcludedFromReward	read	Passed	No Issue
21	totalFees	read	Passed	No Issue
22	deliver	write	Passed	No Issue
23	reflectionFromToken	read	Passed	No Issue
24	tokenFromReflection	read	Passed	No Issue
25	excludeFromReward	write	access only Owner	No Issue
26	includeInReward	external	access only Owner	No Issue
27	_transferBothExcluded	write	Passed	No Issue
28	excludeFromFee	write	access only Owner	No Issue
29	includeInFee	write	access only Owner	No Issue
30	setTaxFeePercent	external	access only Owner	No Issue
31	getTaxFee	read	Passed	No Issue
32	setSellTaxFeePercent	external	access only Owner	No Issue
33	setBuyTaxFeePercent	external	access only Owner	No Issue
34	setLiquidityFeePercent	external	access only Owner	No Issue
35	setBurnFeePercent	external	access only Owner	No Issue
36	setMarketingFeePercent	external	access only Owner	No Issue
37	setCharityFeePercent	external	access only Owner	No Issue
38	setBuyBackFeePercent	external	access only Owner	No Issue
39	setRaffleFeePercent	external	access only Owner	No Issue
40	setPoolFeePercent	external	access only Owner	No Issue
41	setMarketingWallet	external	access only Owner	No Issue
42	setCharityWallet	external	access only Owner	No Issue

43	setBuyBackWallet	external	access only Owner	No Issue
44	setRaffleWallet	external	access only Owner	No Issue
45	setPoolWallet	external	access only Owner	No Issue
46	setPoolCharityWallet	external	access only Owner	No Issue
47	setMaxTxPercent	external	access only Owner	No Issue
48	setSwapAndLiquifyEnabled	external	access only Owner	No Issue
49	_reflectFee	write	Passed	No Issue
50	receive	external	Passed	No Issue
51	_getValues	write	Passed	No Issue
52	_getTValues	write	Passed	No Issue
53	_getRValues	write	Passed	No Issue
54	_getRate	read	Passed	No Issue
55	_getCurrentSupply	read	Passed	No Issue
56	_takeLiquidity	write	Passed	No Issue
57	_takeBurn	write	Passed	No Issue
58	_takeMarketing	write	Passed	No Issue
59	_takeCharity	write	Passed	No Issue
60	_takeBuyBack	write	Passed	No Issue
61	_takeRaffle	write	Passed	No Issue
62	_takePool	write	Passed	No Issue
63	calculateTaxFee	read	Passed	No Issue
64	calculateLiquidityFee	read	Passed	No Issue
65	calculateBurnFee	read	Passed	No Issue
66	calculateMarketingFee	read	Passed	No Issue
67	calculateCharityFee	read	Passed	No Issue
68	calculateBuyBackFee	read	Passed	No Issue
69	calculateRaffleFee	read	Passed	No Issue
70	calculatePoolFee	read	Passed	No Issue
71	removeMainFee	write	Passed	No Issue
72	removeDirectWalletFee	write	Passed	No Issue
73	removeRaffleFee	write	Passed	No Issue
74	removePoolFee	write	Passed	No Issue
75	restoreMainFee	write	Passed	No Issue
76	restoreRaffleFee	write	Passed	No Issue
77	restoreDirectWalletFee	write	Passed	No Issue
78	restorePoolFee	write	Passed	No Issue
79	isExcludedFromFee	read	Passed	No Issue
80	_approve	write	Passed	No Issue
81	_transfer	write	Passed	No Issue
82	swapAndLiquify	write	Passed	No Issue
83	swapTokensForEth	write	Passed	No Issue
84	swapAndSendToWallet	write	Passed	No Issue
85	addLiquidity	write	Passed	No Issue
86	_tokenTransfer	write	Passed	No Issue
87	_transferStandard	write	Passed	No Issue
88	_transferToExcluded	write	Passed	No Issue
89	_transferFromExcluded	write	Passed	No Issue

90	mint	write	access only Minter	No Issue
91	_mint	internal	Set require with error message	Refer Audit Findings
92	addMinter	write	access only Owner	No Issue
93	delMinter	write	access only Owner	No Issue
94	getMinterLength	read	Passed	No Issue
95	isMinter	read	Passed	No Issue
96	getMinter	read	access only Owner	No Issue
97	onlyMinter	modifier	Passed	No Issue

MasterChef.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Visibility for constructor is ignored	Refer Audit Findings
2	updateMultiplier	write	access only Owner	No Issue
3	owner	read	Passed	No Issue
4	isOwner	read	Passed	No Issue
5	renounceOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	transferOwnership	write	access only Owner	No Issue
8	onlyOwner	modifier	Passed	No Issue
9	poolLength	external	Passed	No Issue
10	withdrawDevAndRefFee	write	Passed	No Issue
11	add	write	The owner can add the same lp token multiple times	Refer Audit Findings
12	set	write	access only Owner	No Issue
13	setMigrator	write	access only Owner	No Issue
14	migrate	write	Passed	No Issue
15	getMultiplier	read	Passed	No Issue
16	pendingTALLY	external	Passed	No Issue
17	massUpdatePools	write	Passed	No Issue
18	updatePool	write	Passed	No Issue
19	deposit	write	Passed	No Issue
20	withdraw	write	Passed	No Issue
21	enterStaking	write	Passed	No Issue
22	leaveStaking	write	Passed	No Issue
23	emergencyWithdraw	write	Passed	No Issue
24	safeTALLYTransfer	internal	Passed	No Issue
25	setReservAddress	write	Passed	No Issue
26	setBuyBackReservesAddress	write	access only Owner	No Issue
27	setPlatformMaintenanceSecurityAddress	write	access only Owner	No Issue

28	setOperationManagerAddress	write	access only Owner	No Issue
29	updateTALLYPerBlock	write	access only Owner	No Issue
30	setStakingPercent	write	access only Owner	No Issue
31	setReservPercent	write	access only Owner	No Issue
32	setMaintenanceSecurityPercent	write	access only Owner	No Issue
33	setBuyBackReservesPercent	write	access only Owner	No Issue
34	setOperationManagerPercent	write	access only Owner	No Issue

TallyswapFactory.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Visibility for constructor is ignored	Refer Audit Findings
2	allPairsLength	external	Passed	No Issue
3	createPair	external	Passed	No Issue
4	setFeeTo	external	Passed	No Issue
5	setFeeToSetter	external	Passed	No Issue
6	setDevFee	external	Passed	No Issue
7	setSwapFee	external	Passed	No Issue

TallyswapRouter.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Visibility for constructor is ignored	Refer Audit Findings
2	owner	read	Passed	No Issue
3	isOwner	read	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	_transferOwnership	internal	Passed	No Issue
6	transferOwnership	write	access only Owner	No Issue
7	onlyOwner	modifier	Passed	No Issue
8	ensure	modifier	Passed	No Issue
9	receive	external	Passed	No Issue
10	setSwapFeeReward	write	access only Owner	No Issue
11	_addLiquidity	internal	Passed	No Issue
12	addLiquidity	external	Passed	No Issue
13	addLiquidityETH	external	Passed	No Issue

14	removeLiquidity	write	Passed	No Issue
15	removeLiquidityETH	write	Passed	No Issue
16	removeLiquidityWithPermit	external	Passed	No Issue
17	removeLiquidityETHWithPermit	external	Passed	No Issue
18	removeLiquidityETHSupportingFeeOnTransferTokens	write	Passed	No Issue
19	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	external	Passed	No Issue
20	_swap	internal	Passed	No Issue
21	swapExactTokensForTokens	external	Passed	No Issue
22	swapTokensForExactTokens	external	Passed	No Issue
23	swapExactETHForTokens	external	Passed	No Issue
24	swapTokensForExactETH	external	Passed	No Issue
25	swapExactTokensForETH	external	Passed	No Issue
26	swapETHForExactTokens	external	Passed	No Issue
27	_swapSupportingFeeOnTransferTokens	internal	Passed	No Issue
28	swapExactTokensForTokensSupportingFeeOnTransferTokens	external	Passed	No Issue
29	swapExactETHForTokensSupportingFeeOnTransferTokens	external	Passed	No Issue
30	swapExactTokensForETHSupportingFeeOnTransferTokens	external	Passed	No Issue
31	quote	write	Passed	No Issue
32	getAmountOut	write	Passed	No Issue
33	getAmountIn	write	Passed	No Issue
34	getAmountsOut	write	Passed	No Issue
35	getAmountsIn	write	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

(1) The owner can add the same lp token more than once: **MasterChef.sol**

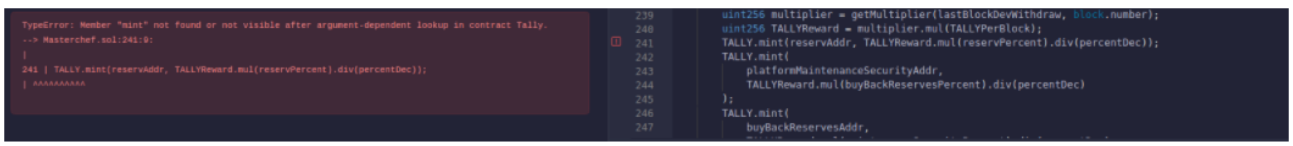
```
// Add a new lp to the pool. Can only be called by the owner.
// XXX DO NOT add the same LP token more than once. Rewards will be messed up if you do.
function add(
    uint256 _allocPoint,
    IERC20 _lpToken,
    bool _withUpdate
) public onlyOwner {
    if (_withUpdate) {
        massUpdatePools();
    }
    uint256 lastRewardBlock = block.number > startBlock
        ? block.number
```

As per comment on the add() function, the same LP token should not be added more than once.

Resolution: There is no validation or check for adding duplicate _lpToken in add() function, check duplication of _lpToken when the owner can add new _lpToken in add() function.
Status: Acknowledged

Low

(1) Member "mint" not found or not visible: **MasterChef.sol**



TypeError: Member "mint" not found or not visible after argument-dependent lookup in contract Tally.

Resolution: Please add the mint function in the TALLY Token contract and after that use it in the MasterChef contract.

Status: **Fixed**

Very Low / Informational / Best practices:

(1) Spelling mistake: **MasterChef.sol**

```
// Last block then develeper withdraw dev and ref fee
uint256 public lastBlockDevWithdraw;
```

```
// Bonus muliplier for early TALLY makers.
uint256 public BONUS_MULTIPLIER = 1;
```

```
// Total allocation poitns. Must be the sum of all allocation points in all pools.
uint256 public totalAllocPoint = 0;
```

Spelling mistake in comments:

1. develeper word should be developer.
2. muliplier word should be multiplier.
3. poitns word should be points.

Resolution: We suggest correcting the spelling.

Status: Acknowledged

(2) SafeMath Library: **MasterChef.sol, TallyswapRouter.sol, SwapfeeReward.sol**

SafeMath Library is used in this contract code, but the compiler version is greater than or equal to 0.8.0, Then it will not be required to use, solidity automatically handles overflow/underflow.

Resolution: We suggest removing the SafeMath library and use normal math operators, It will improve code size, and less gas consumption.

Status: Acknowledged

(3) Make variables constant: **MasterChef.sol**

These variables will be unchanged. So, please make it constant. It will save some gas.

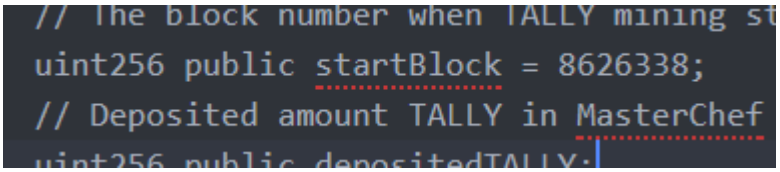
Variables are:

- percentDec
- startBlock

Resolution: We suggest declaring those variables as constant. Just put a constant keyword.

Status: Acknowledged

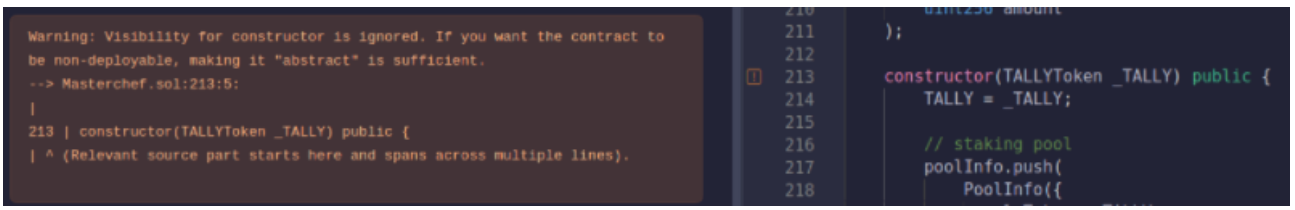
(4) startBlock must be set to the proper value: **MasterChef.sol**
While deploying a masterChef smart contract, startBlock must be set to the proper value, so that rewards can be handled properly.



Resolution: We suggest double confirm before deploying the masterChef smart contract.
Status: Acknowledged

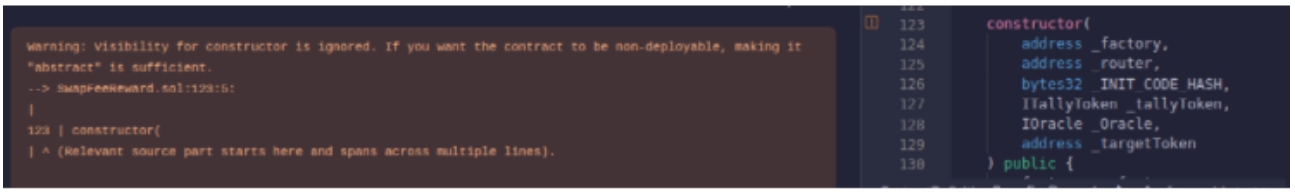
(5) Visibility for constructor is ignored:

MasterChef.sol



TallyswapRouter.sol

SwapFeeReward.sol



TallyswapFactory.sol

Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.

Resolution: We suggest removing the public keywords.

Status: Acknowledged

Centralization

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

- setPhase: The SWAPFeeReward Owner can set phase.
- addWhitelist: The SWAPFeeReward Owner can add an address to the whitelist.
- delWhitelist: The SWAPFeeReward Owner can delete addresses from whitelist.
- setRouter: The SWAPFeeReward Owner can set the router address.
- setOracle: The SWAPFeeReward Owner can set the oracle address.
- setFactory: The SWAPFeeReward Owner can set the factory address.
- setInitCodeHash: The SWAPFeeReward Owner can set initial hash code.
- addPair: The SWAPFeeReward Owner can add a pair address.
- setPair: The SWAPFeeReward Owner can set pair id, reward percentage,
- setPairEnabled: The SWAPFeeReward Owner can set pair status.
- excludeFromReward: The Tally token owner can check if the address is excluded or not and set it.
- includeInReward: The Tally token owner can check if the address is excluded or not and set it.
- excludeFromFee: The Tally token owner can set the status “true” from the excluded account address.
- includeInFee: The Tally token owner can set the status “false” from the excluded account address.
- setTaxFeePercent: The Tally token owner can set the tax fee percentage.
- updateMultiplier: The Masterchef owner can update the multiplier number.
- add: The Masterchef owner can add a new lp to the pool.
- set: The Masterchef owner can update the given pool's TALLY allocation point.
- setMigrator: The Masterchef owner can set the migrator contract.
- setReservAddress: The Masterchef owner can set a reserve address.
- setBuyBackReservesAddress: The Masterchef owner can set a buy back reserve address.
- setPlatformMaintenanceSecurityAddress: The Masterchef owner can set a platform maintenance address.

- setOperationManagerAddress: The Masterchef owner can set the operation manager address.
- updateTALLYPerBlock: The Masterchef owner can update the tally per block amount.
- setStakingPercent: The Masterchef owner can set the staking percentage value.
- setReservPercent:The Masterchef owner can set reserve percentage value
- setMaintenanceSecurityPercent: The Masterchef owner can set maintenance security percentage value.
- setBuyBackReservesPercent:The Masterchef owner can set the buy back reserve percentage value.
- setOperationManagerPercent:The Masterchef owner can set the operation manager percentage value.
- setSwapFeeReward: The TallyswapRouter owner can set a swap fee reward address.
- addMinter: The Tally token owner can add a new minter address.
- delMinter: The Tally token owner can remove the minter address.
- getMinter: The Tally token owner can get minter details.

To make the smart contract 100% decentralized, we suggest renouncing ownership in the airdrop smart contract once its function is completed.

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 had observed some issues in the smart contracts, but they were resolved in the revised smart contract code. **So, the smart contracts are ready for the mainnet deployment**.

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.

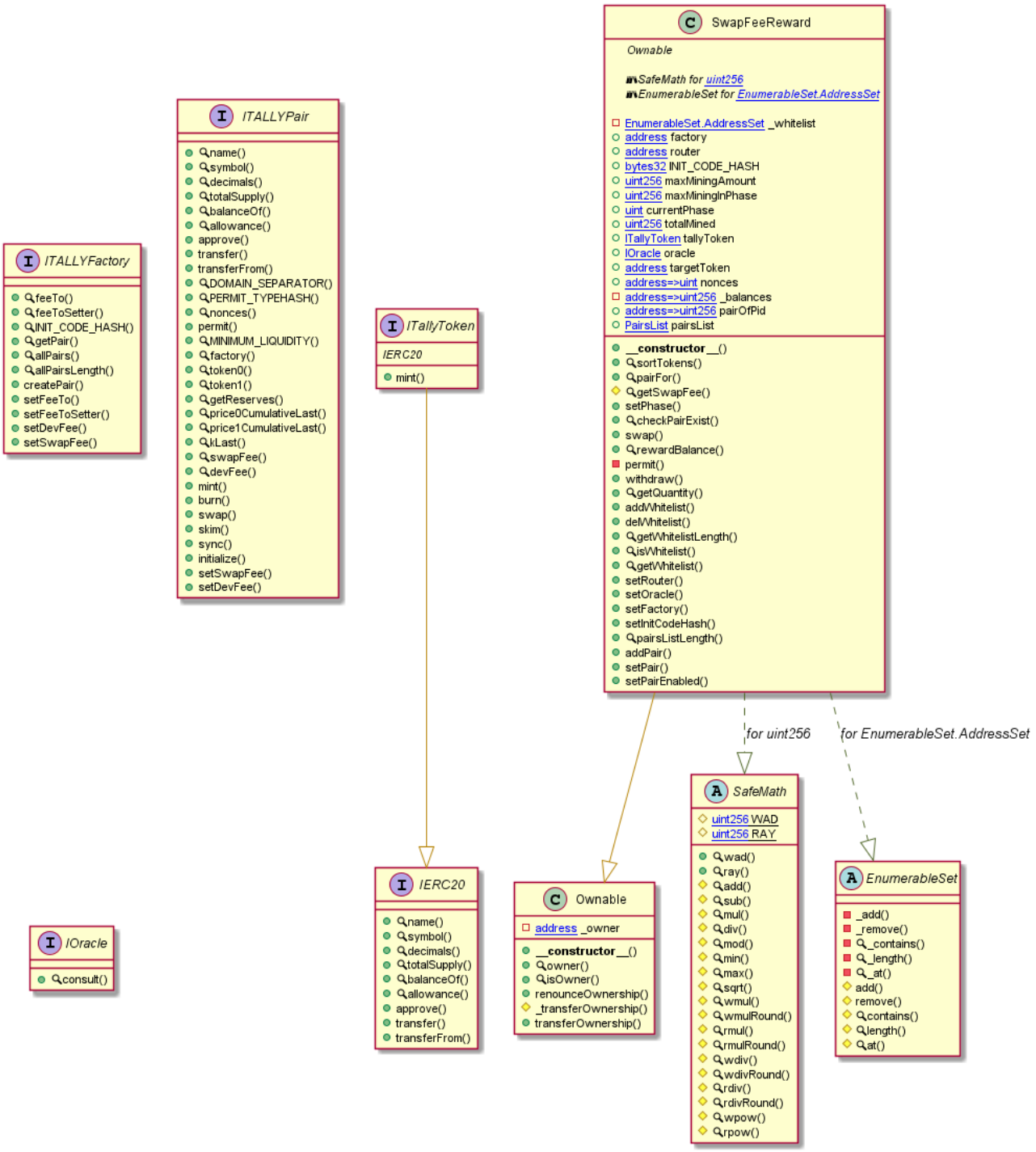
Technical Disclaimer

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.

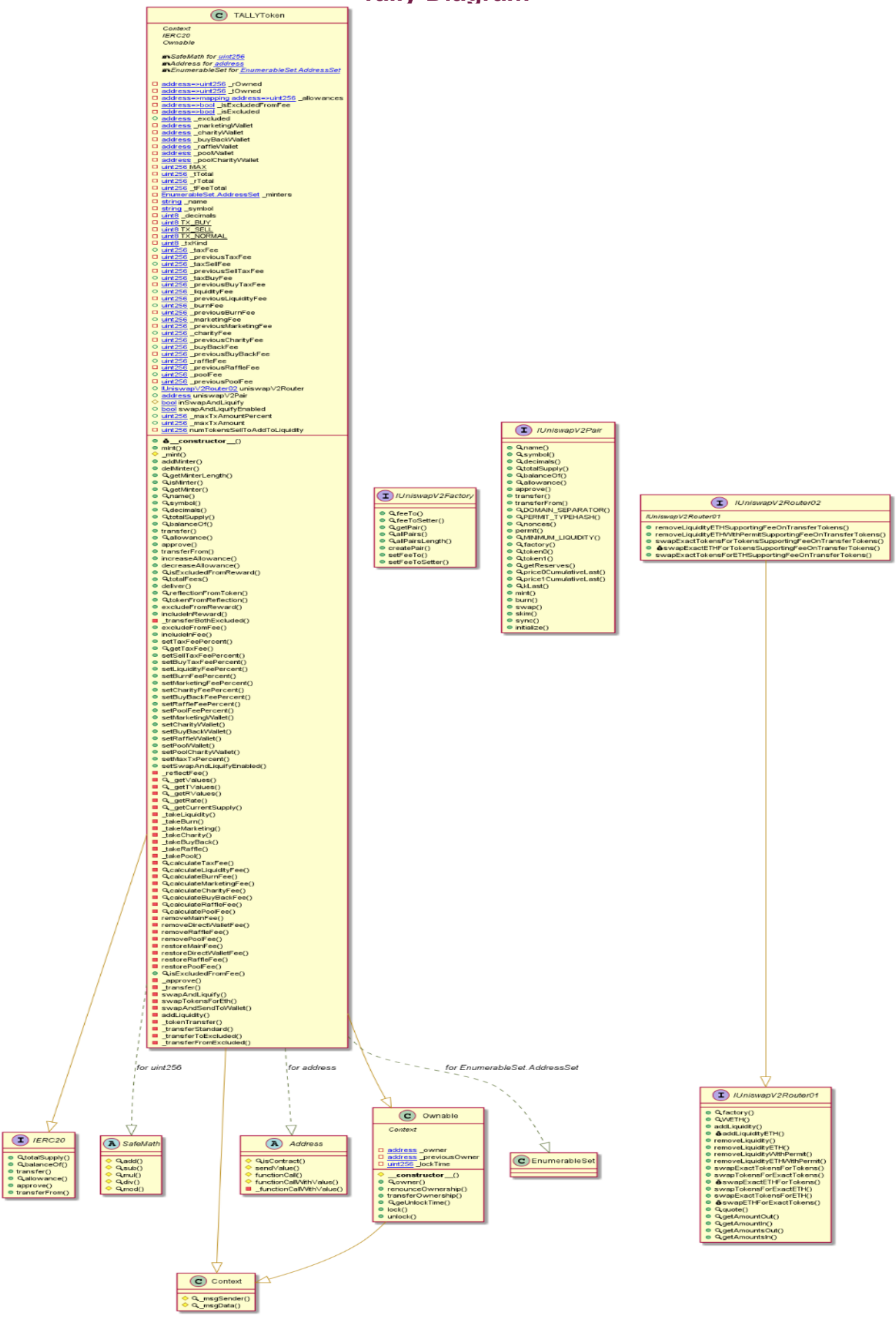
Appendix

Code Flow Diagram - Tally-ho Protocol

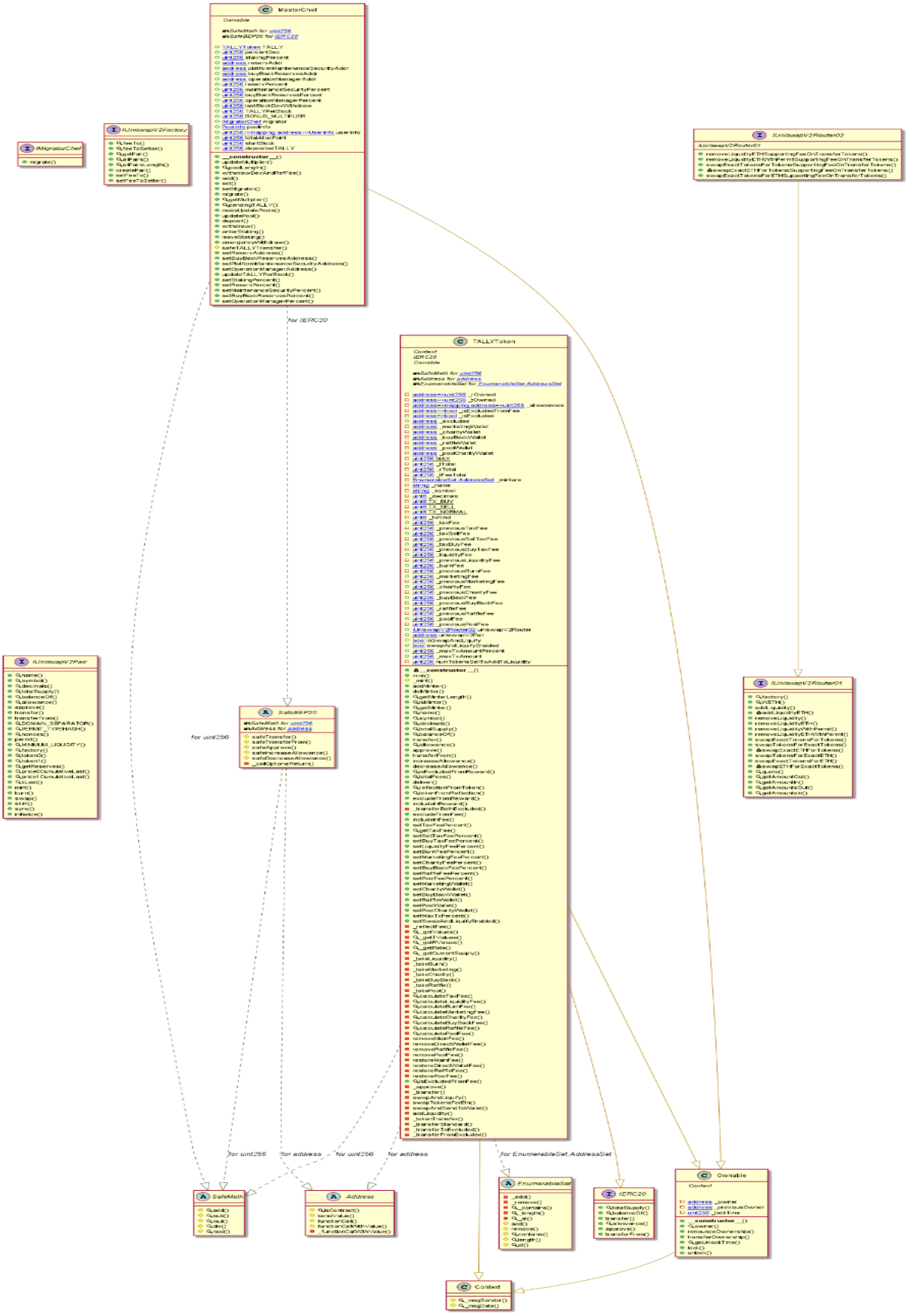
SwapFeeReward Diagram



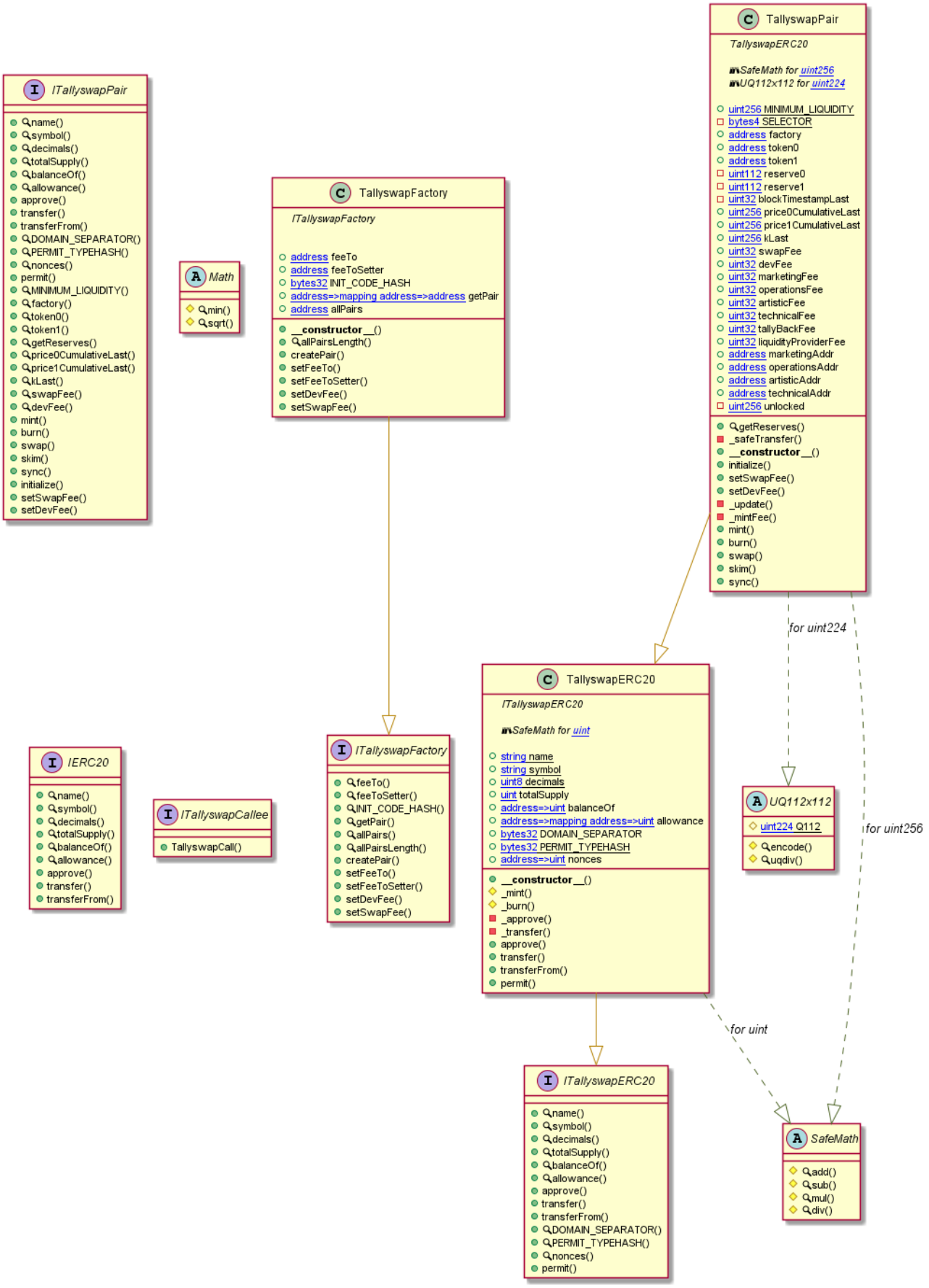
Tally Diagram



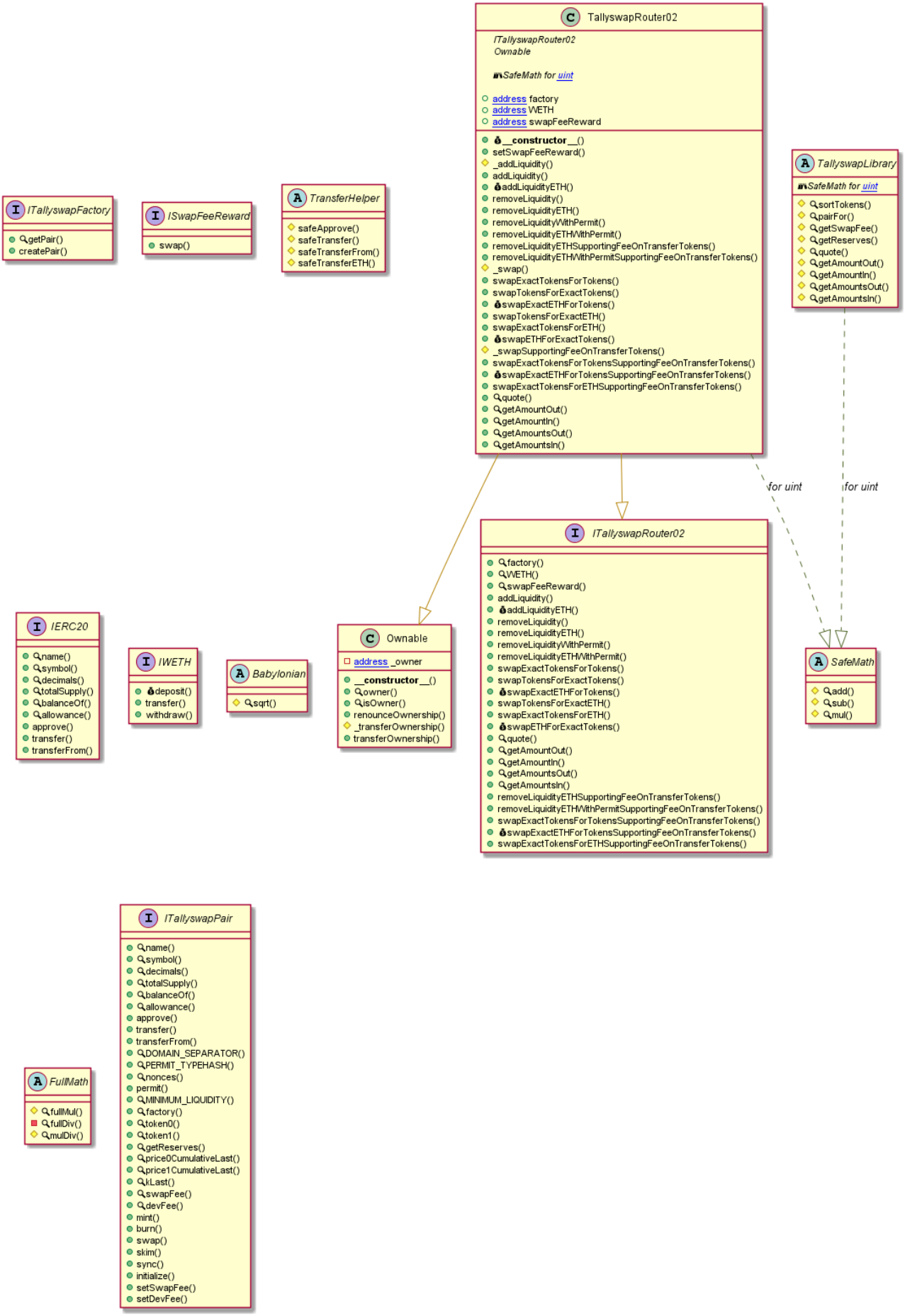
MasterChef Diagram



TallyswapFactory Diagram



TallyswapRouter Diagram



Slither Results Log

Slither log >> SwapFeeReward.sol

```
INFO:Detectors:
SwapFeeReward.setRouter(address) (SwapFeeReward.sol#737-740) should emit an event for:
        - router = newRouter (SwapFeeReward.sol#739)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
SwapFeeReward.constructor(address,address,bytes32,ITallyToken,IOracle,address)._factory (SwapFeeReward.sol#596) lacks a zero-check on :
                - factory = _factory (SwapFeeReward.sol#603)
SwapFeeReward.constructor(address,address,bytes32,ITallyToken,IOracle,address)._router (SwapFeeReward.sol#597) lacks a zero-check on :
                - router = _router (SwapFeeReward.sol#604)
SwapFeeReward.constructor(address,address,bytes32,ITallyToken,IOracle,address)._targetToken (SwapFeeReward.sol#601) lacks a zero-check o
:
                - targetToken = _targetToken (SwapFeeReward.sol#608)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
SwapFeeReward.getQuantity(address,uint256,address) (SwapFeeReward.sol#694-712) has external calls inside a loop: ITALLYFactory(factory).
tPair(outputToken,intermediate) != address(0) && ITALLYFactory(factory).getPair(intermediate,anchorToken) != address(0) && checkPairExis
intermediate,anchorToken) (SwapFeeReward.sol#704)
SwapFeeReward.getQuantity(address,uint256,address) (SwapFeeReward.sol#694-712) has external calls inside a loop: interQuantity = IOracle
racle).consult(outputToken,outputAmount,intermediate) (SwapFeeReward.sol#705)
SwapFeeReward.getQuantity(address,uint256,address) (SwapFeeReward.sol#694-712) has external calls inside a loop: quantity = IOracle(orac
).consult(intermediate,interQuantity,anchorToken) (SwapFeeReward.sol#706)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
INFO:Detectors:
Reentrancy in SwapFeeReward.withdraw(uint8,bytes32,bytes32) (SwapFeeReward.sol#679-692):
        External calls:
        - tallyToken.mint(msg.sender,balance) (SwapFeeReward.sol#685)
        Event emitted after the call(s):
        - Withdraw(msg.sender,balance) (SwapFeeReward.sol#687)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
SwapFeeReward.swap(address,address,address,uint256) (SwapFeeReward.sol#645-667) compares to a boolean constant:
        -pool.pair != pair || pool.enabled == false (SwapFeeReward.sol#654)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality
INFO:Detectors:
EnumerableSet.add(EnumerableSet.Bytes32Set,bytes32) (SwapFeeReward.sol#265-267) is never used and should be removed
EnumerableSet.add(EnumerableSet.UintSet,uint256) (SwapFeeReward.sol#374-376) is never used and should be removed
EnumerableSet.at(EnumerableSet.Bytes32Set,uint256) (SwapFeeReward.sol#303-305) is never used and should be removed
```

```
EnumerableSet.at(EnumerableSet.UintSet,uint256) (SwapFeeReward.sol#412-414) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes32) (SwapFeeReward.sol#282-284) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,uint256) (SwapFeeReward.sol#391-393) is never used and should be removed
EnumerableSet.length(EnumerableSet.Bytes32Set) (SwapFeeReward.sol#289-291) is never used and should be removed
EnumerableSet.length(EnumerableSet.UintSet) (SwapFeeReward.sol#398-400) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (SwapFeeReward.sol#275-277) is never used and should be removed
EnumerableSet.remove(EnumerableSet.UintSet,uint256) (SwapFeeReward.sol#384-386) is never used and should be removed
SafeMath.max(uint256,uint256) (SwapFeeReward.sol#73-75) is never used and should be removed
SafeMath.min(uint256,uint256) (SwapFeeReward.sol#69-71) is never used and should be removed
SafeMath.mod(uint256,uint256) (SwapFeeReward.sol#60-62) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (SwapFeeReward.sol#64-67) is never used and should be removed
SafeMath.rdiv(uint256,uint256) (SwapFeeReward.sol#114-116) is never used and should be removed
SafeMath.rdivRound(uint256,uint256) (SwapFeeReward.sol#118-120) is never used and should be removed
SafeMath.rmul(uint256,uint256) (SwapFeeReward.sol#98-100) is never used and should be removed
SafeMath.rmulRound(uint256,uint256) (SwapFeeReward.sol#102-104) is never used and should be removed
SafeMath.rpow(uint256,uint256) (SwapFeeReward.sol#134-144) is never used and should be removed
SafeMath.sqrt(uint256) (SwapFeeReward.sol#77-88) is never used and should be removed
SafeMath.wdiv(uint256,uint256) (SwapFeeReward.sol#106-108) is never used and should be removed
SafeMath.wdivRound(uint256,uint256) (SwapFeeReward.sol#110-112) is never used and should be removed
SafeMath.wmul(uint256,uint256) (SwapFeeReward.sol#90-92) is never used and should be removed
SafeMath.wmulRound(uint256,uint256) (SwapFeeReward.sol#94-96) is never used and should be removed
SafeMath.wpow(uint256,uint256) (SwapFeeReward.sol#122-132) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version0.8.4 (SwapFeeReward.sol#1) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Function ITALLYFactory.INIT_CODE_HASH() (SwapFeeReward.sol#447) is not in mixedCase
Function ITALLYPair.DOMAIN_SEPARATOR() (SwapFeeReward.sol#476) is not in mixedCase
Function ITALLYPair.PERMIT_TYPEHASH() (SwapFeeReward.sol#477) is not in mixedCase
Function ITALLYPair.MINIMUM_LIQUIDITY() (SwapFeeReward.sol#494) is not in mixedCase
Parameter SwapFeeReward.setPhase(uint256)._newPhase (SwapFeeReward.sol#631) is not in mixedCase
Parameter SwapFeeReward.addWhitelist(address)._addToken (SwapFeeReward.sol#714) is not in mixedCase
Parameter SwapFeeReward.delWhitelist(address)._delToken (SwapFeeReward.sol#719) is not in mixedCase
Parameter SwapFeeReward.isWhitelist(address)._token (SwapFeeReward.sol#728) is not in mixedCase
Parameter SwapFeeReward.getWhitelist(uint256)._index (SwapFeeReward.sol#732) is not in mixedCase
```

```
Parameter SwapFeeReward.setPairEnabled(uint256,bool)._pid (SwapFeeReward.sol#777) is not in mixedCase
Parameter SwapFeeReward.setPairEnabled(uint256,bool)._enabled (SwapFeeReward.sol#777) is not in mixedCase
Variable SwapFeeReward.INIT_CODE_HASH (SwapFeeReward.sol#567) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
SwapFeeReward.slitherConstructorVariables() (SwapFeeReward.sol#560-780) uses literals with too many digits:
        - maxMiningAmount = 100000000 * 1e18 (SwapFeeReward.sol#568)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
SwapFeeReward.maxMiningAmount (SwapFeeReward.sol#568) should be constant
SwapFeeReward.maxMiningInPhase (SwapFeeReward.sol#569) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
wad() should be declared external:
        - SafeMath.wad() (SwapFeeReward.sol#7-9)
ray() should be declared external:
        - SafeMath.ray() (SwapFeeReward.sol#11-13)
owner() should be declared external:
        - Ownable.owner() (SwapFeeReward.sol#528-530)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (SwapFeeReward.sol#536-539)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (SwapFeeReward.sol#547-549)
setPhase(uint256) should be declared external:
        - SwapFeeReward.setPhase(uint256) (SwapFeeReward.sol#631-634)
swap(address,address,address,uint256) should be declared external:
        - SwapFeeReward.swap(address,address,address,uint256) (SwapFeeReward.sol#645-667)
rewardBalance(address) should be declared external:
        - SwapFeeReward.rewardBalance(address) (SwapFeeReward.sol#669-671)
```

```
withdraw(uint8,bytes32,bytes32) should be declared external:
        - SwapFeeReward.withdraw(uint8,bytes32,bytes32) (SwapFeeReward.sol#679-692)
addWhitelist(address) should be declared external:
        - SwapFeeReward.addWhitelist(address) (SwapFeeReward.sol#714-717)
delWhitelist(address) should be declared external:
        - SwapFeeReward.delWhitelist(address) (SwapFeeReward.sol#719-722)
setRouter(address) should be declared external:
        - SwapFeeReward.setRouter(address) (SwapFeeReward.sol#737-740)
setOracle(IOracle) should be declared external:
        - SwapFeeReward.setOracle(IOracle) (SwapFeeReward.sol#742-745)
setFactory(address) should be declared external:
        - SwapFeeReward.setFactory(address) (SwapFeeReward.sol#747-750)
setInitCodeHash(bytes32) should be declared external:
        - SwapFeeReward.setInitCodeHash(bytes32) (SwapFeeReward.sol#752-754)
addPair(uint256,address) should be declared external:
        - SwapFeeReward.addPair(uint256,address) (SwapFeeReward.sol#760-771)
setPair(uint256,uint256) should be declared external:
        - SwapFeeReward.setPair(uint256,uint256) (SwapFeeReward.sol#773-775)
setPairEnabled(uint256,bool) should be declared external:
        - SwapFeeReward.setPairEnabled(uint256,bool) (SwapFeeReward.sol#777-779)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:SwapFeeReward.sol analyzed (9 contracts with 75 detectors), 78 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> MasterChef.sol

```
INFO:Detectors:
TALLYToken.allowance(address,address).owner (MasterChef.sol#1351) shadows:
        - Ownable.owner() (MasterChef.sol#451-453) (function)
TALLYToken._approve(address,address,uint256).owner (MasterChef.sol#1917) shadows:
        - Ownable.owner() (MasterChef.sol#451-453) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
TALLYToken.setMarketingWallet(address).account (MasterChef.sol#1576) lacks a zero-check on :
                - _marketingWallet = account (MasterChef.sol#1577)
TALLYToken.setCharityWallet(address).account (MasterChef.sol#1580) lacks a zero-check on :
                - _charityWallet = account (MasterChef.sol#1581)
TALLYToken.setBuyBackWallet(address).account (MasterChef.sol#1584) lacks a zero-check on :
                - _buyBackWallet = account (MasterChef.sol#1585)
TALLYToken.setRaffleWallet(address).account (MasterChef.sol#1588) lacks a zero-check on :
                - _raffleWallet = account (MasterChef.sol#1589)
TALLYToken.setPoolWallet(address).account (MasterChef.sol#1592) lacks a zero-check on :
                - _poolWallet = account (MasterChef.sol#1593)
TALLYToken.setPoolCharityWallet(address).account (MasterChef.sol#1596) lacks a zero-check on :
                - _poolCharityWallet = account (MasterChef.sol#1597)
MasterChef.setReservAddress(address)._reservAddr (MasterChef.sol#2676) lacks a zero-check on :
                - reservAddr = _reservAddr (MasterChef.sol#2677)
MasterChef.setBuyBackReservesAddress(address)._buyBackReservesAddr (MasterChef.sol#2680) lacks a zero-check on :
                - buyBackReservesAddr = _buyBackReservesAddr (MasterChef.sol#2684)
MasterChef.setPlatformMaintenanceSecurityAddress(address)._platformMaintenanceSecurityAddr (MasterChef.sol#2688) lacks a zero-check on :
                - platformMaintenanceSecurityAddr = _platformMaintenanceSecurityAddr (MasterChef.sol#2690)
MasterChef.setOperationManagerAddress(address)._operationManagerAddr (MasterChef.sol#2693) lacks a zero-check on :
                - operationManagerAddr = _operationManagerAddr (MasterChef.sol#2697)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in TALLYToken._transfer(address,address,uint256) (MasterChef.sol#1928-1975):
        External calls:
        - swapAndLiquify(contractTokenBalance) (MasterChef.sol#1962)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (MasterChef.s
#2036-2043)
```

```
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (MasterChef.s
#2036-2043)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (
sterChef.sol#2009-2015)
                - (success) = address(account).call{value: dividends}() (MasterChef.sol#2024)
        External calls sending eth:
        - _takeMarketing(values[7]) (MasterChef.sol#2149)
                - (success) = address(account).call{value: dividends}() (MasterChef.sol#2024)
        - _takeCharity(values[8]) (MasterChef.sol#2150)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (MasterChef.s
#2036-2043)
                - (success) = address(account).call{value: dividends}() (MasterChef.sol#2024)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (MasterChef.sol#1925)
                - _takeCharity(values[8]) (MasterChef.sol#2150)
        - SendBNBToWallet(tokens,account) (MasterChef.sol#2027)
                - _takeCharity(values[8]) (MasterChef.sol#2150)
        - SwapAndLiquify(half,newBalance,otherHalf) (MasterChef.sol#1997)
                - _takeCharity(values[8]) (MasterChef.sol#2150)
        - Transfer(sender,recipient,values[3]) (MasterChef.sol#2132)
                - _takeCharity(values[8]) (MasterChef.sol#2150)
        - Transfer(sender,recipient,values[3]) (MasterChef.sol#2157)
                - _takeCharity(values[8]) (MasterChef.sol#2150)
        - Transfer(sender,recipient,values[3]) (MasterChef.sol#2182)
                - _takeCharity(values[8]) (MasterChef.sol#2150)
        - Transfer(sender,recipient,values[3]) (MasterChef.sol#1515)
                - _takeCharity(values[8]) (MasterChef.sol#2150)
Reentrancy in TALLYToken._transferToExcluded(address,address,uint256,uint8) (MasterChef.sol#2135-2158):
        External calls:
        - _takeMarketing(values[7]) (MasterChef.sol#2149)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (
sterChef.sol#2009-2015)
                - (success) = address(account).call{value: dividends}() (MasterChef.sol#2024)
        - _takeCharity(values[8]) (MasterChef.sol#2150)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (MasterChef.s
#2036-2043)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Ownable.unlock() (MasterChef.sol#501-509) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp > _lockTime,Contract is locked until 7 days) (MasterChef.sol#506)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
```

```
INFO:Detectors:
Address.isContract(address) (MasterChef.sol#273-284) uses assembly
        - INLINE ASM (MasterChef.sol#280-282)
Address._functionCallWithValue(address,bytes,uint256,string) (MasterChef.sol#399-427) uses assembly
        - INLINE ASM (MasterChef.sol#419-422)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (MasterChef.sol#334-339) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (MasterChef.sol#366-378) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (MasterChef.sol#386-397) is never used and should be removed
Address.sendValue(address,uint256) (MasterChef.sol#302-314) is never used and should be removed
Context._msgData() (MasterChef.sol#248-251) is never used and should be removed
EnumerableSet.add(EnumerableSet.UintSet,uint256) (MasterChef.sol#1055-1057) is never used and should be removed
EnumerableSet.at(EnumerableSet.UintSet,uint256) (MasterChef.sol#1100-1106) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,uint256) (MasterChef.sol#1075-1081) is never used and should be removed
EnumerableSet.length(EnumerableSet.UintSet) (MasterChef.sol#1086-1088) is never used and should be removed
EnumerableSet.remove(EnumerableSet.UintSet,uint256) (MasterChef.sol#1065-1070) is never used and should be removed
SafeBEP20.safeDecreaseAllowance(IERC20,address,uint256) (MasterChef.sol#2259-2276) is never used and should be removed
SafeBEP20.safeIncreaseAllowance(IERC20,address,uint256) (MasterChef.sol#2241-2257) is never used and should be removed
SafeMath.mod(uint256,uint256) (MasterChef.sol#217-219) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (MasterChef.sol#233-240) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
TALLYToken._rTotal (MasterChef.sol#1132) is set pre-construction with a non-constant function or state variable:
        - (MAX - (MAX % _tTotal))
TALLYToken._previousTaxFee (MasterChef.sol#1149) is set pre-construction with a non-constant function or state variable:
        - _taxFee
TALLYToken._previousSellTaxFee (MasterChef.sol#1152) is set pre-construction with a non-constant function or state variable:
        - _taxSellFee
TALLYToken._previousBuyTaxFee (MasterChef.sol#1155) is set pre-construction with a non-constant function or state variable:
        - _taxBuyFee
TALLYToken._previousLiquidityFee (MasterChef.sol#1158) is set pre-construction with a non-constant function or state variable:
        - _liquidityFee
TALLYToken._previousBurnFee (MasterChef.sol#1161) is set pre-construction with a non-constant function or state variable:
        - _burnFee
TALLYToken._previousMarketingFee (MasterChef.sol#1164) is set pre-construction with a non-constant function or state variable:
        - _marketingFee
```

```
TALLYToken._previousRaffleFee (MasterChef.sol#1173) is set pre-construction with a non-constant function or state variable:
        - _raffleFee
TALLYToken._previousPoolFee (MasterChef.sol#1176) is set pre-construction with a non-constant function or state variable:
        - _poolFee
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state-variables
INFO:Detectors:
Pragma version0.8.4 (MasterChef.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (MasterChef.sol#302-314):
        - (success) = recipient.call{value: amount}() (MasterChef.sol#309)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (MasterChef.sol#399-427):
        - (success,returndata) = target.call{value: weiValue}(data) (MasterChef.sol#408-410)
Low level call in TALLYToken.swapAndSendToWallet(uint256,address) (MasterChef.sol#2018-2029):
        - (success) = address(account).call{value: dividends}() (MasterChef.sol#2024)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (MasterChef.sol#576) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (MasterChef.sol#578) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (MasterChef.sol#609) is not in mixedCase
Function IUniswapV2Router01.WETH() (MasterChef.sol#655) is not in mixedCase
Parameter TALLYToken.mint(address,uint256)._to (MasterChef.sol#1255) is not in mixedCase
Parameter TALLYToken.mint(address,uint256)._amount (MasterChef.sol#1255) is not in mixedCase
Parameter TALLYToken.addMinter(address)._addMinter (MasterChef.sol#1286) is not in mixedCase
Parameter TALLYToken.delMinter(address)._delMinter (MasterChef.sol#1294) is not in mixedCase
Parameter TALLYToken.getMinter(uint256)._index (MasterChef.sol#1310) is not in mixedCase
Parameter TALLYToken.setSwapAndLiquifyEnabled(bool)._enabled (MasterChef.sol#1605) is not in mixedCase
Parameter TALLYToken.calculateTaxFee(uint256,uint8)._amount (MasterChef.sol#1781) is not in mixedCase
Parameter TALLYToken.calculateLiquidityFee(uint256)._amount (MasterChef.sol#1790) is not in mixedCase
```

```
Variable MasterChef.TALLY (MasterChef.sol#2335) is not in mixedCase
Variable MasterChef.TALLYPerBlock (MasterChef.sol#2364) is not in mixedCase
Variable MasterChef.BONUS_MULTIPLIER (MasterChef.sol#2366) 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#249)" inContext (MasterChef.sol#243-252)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (MasterChef.sol
60) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (M
terChef.sol#661)
Variable TALLYToken._getRValues(uint256,uint256[9],uint256).rTransferAmount (MasterChef.sol#1706-1710) is too similar to TALLYToken._get
alues(uint256,uint8).tTransferAmount (MasterChef.sol#1662-1666)
Variable TALLYToken._getValues(uint256,uint8).rTransferAmount (MasterChef.sol#1624) is too similar to TALLYToken._getTValues(uint256,uin
).tTransferAmount (MasterChef.sol#1662-1666)
Variable TALLYToken.reflectionFromToken(uint256,bool,uint8).rTransferAmount (MasterChef.sol#1451) is too similar to TALLYToken._getTValu
(uint256,uint8).tTransferAmount (MasterChef.sol#1662-1666)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
TALLYToken.slitherConstructorVariables() (MasterChef.sol#1109-2184) uses literals with too many digits:
        - _tTotal = 1000000000 * 10 ** 9 (MasterChef.sol#1131)
TALLYToken.slitherConstructorVariables() (MasterChef.sol#1109-2184) uses literals with too many digits:
        - numTokensSellToAddToLiquidity = 500000 * 10 ** 6 * 10 ** 9 (MasterChef.sol#1186)
MasterChef.slitherConstructorVariables() (MasterChef.sol#2308-2734) uses literals with too many digits:
        - percentDec = 1000000 (MasterChef.sol#2337)
MasterChef.slitherConstructorVariables() (MasterChef.sol#2308-2734) uses literals with too many digits:
        - stakingPercent = 400000 (MasterChef.sol#2339)
MasterChef.slitherConstructorVariables() (MasterChef.sol#2308-2734) uses literals with too many digits:
        - buyBackReservesPercent = 100000 (MasterChef.sol#2357)
MasterChef.slitherConstructorVariables() (MasterChef.sol#2308-2734) uses literals with too many digits:
        - TALLYPerBlock = 3000000000000000000 (MasterChef.sol#2364)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
```

```
INFO:Detectors:
MasterChef.percentDec (MasterChef.sol#2337) should be constant
MasterChef.startBlock (MasterChef.sol#2376) should be constant
TALLYToken._decimals (MasterChef.sol#1140) should be constant
TALLYToken._name (MasterChef.sol#1138) should be constant
TALLYToken._symbol (MasterChef.sol#1139) should be constant
TALLYToken._tTotal (MasterChef.sol#1131) should be constant
TALLYToken.numTokensSellToAddToLiquidity (MasterChef.sol#1186) 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() (MasterChef.sol#470-473)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (MasterChef.sol#479-486)
geUnlockTime() should be declared external:
        - Ownable.geUnlockTime() (MasterChef.sol#488-490)
lock(uint256) should be declared external:
        - Ownable.lock(uint256) (MasterChef.sol#493-498)
unlock() should be declared external:
        - Ownable.unlock() (MasterChef.sol#501-509)
mint(address,uint256) should be declared external:
        - TALLYToken.mint(address,uint256) (MasterChef.sol#1255-1263)
addMinter(address) should be declared external:
        - TALLYToken.addMinter(address) (MasterChef.sol#1286-1292)
delMinter(address) should be declared external:
        - TALLYToken.delMinter(address) (MasterChef.sol#1294-1300)
getMinter(uint256) should be declared external:
        - TALLYToken.getMinter(uint256) (MasterChef.sol#1310-1313)
name() should be declared external:
        - TALLYToken.name() (MasterChef.sol#1321-1323)
symbol() should be declared external:
        - TALLYToken.symbol() (MasterChef.sol#1325-1327)
decimals() should be declared external:
        - TALLYToken.decimals() (MasterChef.sol#1329-1331)
totalSupply() should be declared external:
        - TALLYToken.totalSupply() (MasterChef.sol#1333-1335)
transfer(address,uint256) should be declared external:
        - TALLYToken.transfer(address,uint256) (MasterChef.sol#1342-1349)
allowance(address,address) should be declared external:
        - TALLYToken.allowance(address,address) (MasterChef.sol#1351-1358)
approve(address,uint256) should be declared external:
        - TALLYToken.approve(address,uint256) (MasterChef.sol#1360-1367)
transferFrom(address,address,uint256) should be declared external:
        - TALLYToken.transferFrom(address,address,uint256) (MasterChef.sol#1369-1384)
increaseAllowance(address,uint256) should be declared external:
        - TALLYToken.increaseAllowance(address,uint256) (MasterChef.sol#1386-1397)
```

```
increaseAllowance(address,uint256) should be declared external:
        - TALLYToken.increaseAllowance(address,uint256) (MasterChef.sol#1386-1397)
decreaseAllowance(address,uint256) should be declared external:
        - TALLYToken.decreaseAllowance(address,uint256) (MasterChef.sol#1399-1413)
isExcludedFromReward(address) should be declared external:
        - TALLYToken.isExcludedFromReward(address) (MasterChef.sol#1415-1417)
totalFees() should be declared external:
        - TALLYToken.totalFees() (MasterChef.sol#1419-1421)
deliver(uint256) should be declared external:
        - TALLYToken.deliver(uint256) (MasterChef.sol#1423-1436)
reflectionFromToken(uint256,bool,uint8) should be declared external:
        - TALLYToken.reflectionFromToken(uint256,bool,uint8) (MasterChef.sol#1438-1454)
excludeFromReward(address) should be declared external:
        - TALLYToken.excludeFromReward(address) (MasterChef.sol#1469-1477)
excludeFromFee(address) should be declared external:
        - TALLYToken.excludeFromFee(address) (MasterChef.sol#1518-1520)
includeInFee(address) should be declared external:
        - TALLYToken.includeInFee(address) (MasterChef.sol#1522-1524)
setSwapAndLiquifyEnabled(bool) should be declared external:
        - TALLYToken.setSwapAndLiquifyEnabled(bool) (MasterChef.sol#1605-1608)
isExcludedFromFee(address) should be declared external:
        - TALLYToken.isExcludedFromFee(address) (MasterChef.sol#1912-1914)
updateMultiplier(uint256) should be declared external:
        - MasterChef.updateMultiplier(uint256) (MasterChef.sol#2404-2406)
withdrawDevAndRefFee() should be declared external:
        - MasterChef.withdrawDevAndRefFee() (MasterChef.sol#2412-2430)
add(uint256,IERC20,bool) should be declared external:
        - MasterChef.add(uint256,IERC20,bool) (MasterChef.sol#2434-2454)
set(uint256,uint256,bool) should be declared external:
        - MasterChef.set(uint256,uint256,bool) (MasterChef.sol#2457-2469)
setMigrator(IMigratorChef) should be declared external:
        - MasterChef.setMigrator(IMigratorChef) (MasterChef.sol#2472-2474)
migrate(uint256) should be declared external:
        - MasterChef.migrate(uint256) (MasterChef.sol#2477-2486)
deposit(uint256,uint256) should be declared external:
        - MasterChef.deposit(uint256,uint256) (MasterChef.sol#2565-2587)
withdraw(uint256,uint256) should be declared external:
```

```
withdraw(uint256,uint256) should be declared external:
        - MasterChef.withdraw(uint256,uint256) (MasterChef.sol#2590-2605)
enterStaking(uint256) should be declared external:
        - MasterChef.enterStaking(uint256) (MasterChef.sol#2608-2633)
leaveStaking(uint256) should be declared external:
        - MasterChef.leaveStaking(uint256) (MasterChef.sol#2636-2654)
emergencyWithdraw(uint256) should be declared external:
        - MasterChef.emergencyWithdraw(uint256) (MasterChef.sol#2657-2664)
setReservAddress(address) should be declared external:
        - MasterChef.setReservAddress(address) (MasterChef.sol#2676-2678)
setBuyBackReservesAddress(address) should be declared external:
        - MasterChef.setBuyBackReservesAddress(address) (MasterChef.sol#2680-2685)
setPlatformMaintenanceSecurityAddress(address) should be declared external:
        - MasterChef.setPlatformMaintenanceSecurityAddress(address) (MasterChef.sol#2687-2691)
setOperationManagerAddress(address) should be declared external:
        - MasterChef.setOperationManagerAddress(address) (MasterChef.sol#2693-2698)
updateTALLYPerBlock(uint256) should be declared external:
        - MasterChef.updateTALLYPerBlock(uint256) (MasterChef.sol#2700-2704)
setStakingPercent(uint256) should be declared external:
        - MasterChef.setStakingPercent(uint256) (MasterChef.sol#2706-2708)
setReservPercent(uint256) should be declared external:
        - MasterChef.setReservPercent(uint256) (MasterChef.sol#2710-2712)
setMaintenanceSecurityPercent(uint256) should be declared external:
        - MasterChef.setMaintenanceSecurityPercent(uint256) (MasterChef.sol#2714-2719)
setBuyBackReservesPercent(uint256) should be declared external:
        - MasterChef.setBuyBackReservesPercent(uint256) (MasterChef.sol#2721-2726)
setOperationManagerPercent(uint256) should be declared external:
        - MasterChef.setOperationManagerPercent(uint256) (MasterChef.sol#2728-2733)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:MasterChef.sol analyzed (14 contracts with 75 detectors), 263 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> TallyswapFactory.sol

```
INFO:Detectors:
TallyswapPair.setSwapFee(uint32) (TallyswapFactory.sol#350-355) should emit an event for:
        - swapFee = _swapFee (TallyswapFactory.sol#354)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
TallyswapPair.initialize(address,address)._token0 (TallyswapFactory.sol#344) lacks a zero-check on :
                - token0 = _token0 (TallyswapFactory.sol#346)
TallyswapPair.initialize(address,address)._token1 (TallyswapFactory.sol#344) lacks a zero-check on :
                - token1 = _token1 (TallyswapFactory.sol#347)
TallyswapFactory.constructor(address)._feeToSetter (TallyswapFactory.sol#596) lacks a zero-check on :
                - feeToSetter = _feeToSetter (TallyswapFactory.sol#597)
TallyswapFactory.setFeeTo(address)._feeTo (TallyswapFactory.sol#621) lacks a zero-check on :
                - feeTo = _feeTo (TallyswapFactory.sol#623)
TallyswapFactory.setFeeToSetter(address)._feeToSetter (TallyswapFactory.sol#626) lacks a zero-check on :
                - feeToSetter = _feeToSetter (TallyswapFactory.sol#628)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in TallyswapPair.burn(address) (TallyswapFactory.sol#461-490):
        External calls:
        - _safeTransfer(_token0,to,amount0) (TallyswapFactory.sol#482)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (TallyswapFactory.sol#315-317)
        - _safeTransfer(_token1,to,amount1) (TallyswapFactory.sol#483)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (TallyswapFactory.sol#315-317)
        State variables written after the call(s):
        - _update(balance0,balance1,_reserve0,_reserve1) (TallyswapFactory.sol#487)
                - price0CumulativeLast += uint256(UQ112x112.encode(_reserve1).uqdiv(_reserve0)) * timeElapsed (TallyswapFactory.sol#379-
1)
        - _update(balance0,balance1,_reserve0,_reserve1) (TallyswapFactory.sol#487)
                - price1CumulativeLast += uint256(UQ112x112.encode(_reserve0).uqdiv(_reserve1)) * timeElapsed (TallyswapFactory.sol#382-
4)
Reentrancy in TallyswapFactory.createPair(address,address) (TallyswapFactory.sol#604-619):
        External calls:
        - ITallyswapPair(pair).initialize(token0,token1) (TallyswapFactory.sol#614)
        State variables written after the call(s):
        - allPairs.push(pair) (TallyswapFactory.sol#617)
Reentrancy in TallyswapPair.swap(uint256,uint256,address,bytes) (TallyswapFactory.sol#493-557):
        External calls:
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in TallyswapPair.burn(address) (TallyswapFactory.sol#461-490):
        External calls:
        - _safeTransfer(_token0,to,amount0) (TallyswapFactory.sol#482)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (TallyswapFactory.sol#315-317)
        - _safeTransfer(_token1,to,amount1) (TallyswapFactory.sol#483)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (TallyswapFactory.sol#315-317)
        Event emitted after the call(s):
        - Burn(msg.sender,amount0,amount1,to) (TallyswapFactory.sol#489)
        - Sync(reserve0,reserve1) (TallyswapFactory.sol#389)
                - _update(balance0,balance1,_reserve0,_reserve1) (TallyswapFactory.sol#487)
Reentrancy in TallyswapFactory.createPair(address,address) (TallyswapFactory.sol#604-619):
        External calls:
        - ITallyswapPair(pair).initialize(token0,token1) (TallyswapFactory.sol#614)
        Event emitted after the call(s):
        - PairCreated(token0,token1,pair,allPairs.length) (TallyswapFactory.sol#618)
Reentrancy in TallyswapPair.swap(uint256,uint256,address,bytes) (TallyswapFactory.sol#493-557):
        External calls:
        - _safeTransfer(_token0,to,amount0Out) (TallyswapFactory.sol#517)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (TallyswapFactory.sol#315-317)
        - _safeTransfer(_token1,to,amount1Out) (TallyswapFactory.sol#518)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (TallyswapFactory.sol#315-317)
        - ITallyswapCallee(to).TallyswapCall(msg.sender,amount0Out,amount1Out,data) (TallyswapFactory.sol#520-525)
        Event emitted after the call(s):
        - Swap(msg.sender,amount0In,amount1In,amount0Out,amount1Out,to) (TallyswapFactory.sol#556)
        - Sync(reserve0,reserve1) (TallyswapFactory.sol#389)
                - _update(balance0,balance1,_reserve0,_reserve1) (TallyswapFactory.sol#555)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
```

```
INFO:Detectors:
TallyswapERC20.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (TallyswapFactory.sol#234-246) uses timestamp for compariso
        Dangerous comparisons:
        - require(bool,string)(deadline >= block.timestamp,Tallyswap: EXPIRED) (TallyswapFactory.sol#235)
TallyswapPair._update(uint256,uint256,uint112,uint112) (TallyswapFactory.sol#365-390) uses timestamp for comparisons
        Dangerous comparisons:
        - timeElapsed > 0 && _reserve0 != 0 && _reserve1 != 0 (TallyswapFactory.sol#377)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
TallyswapFactory.createPair(address,address) (TallyswapFactory.sol#604-619) uses assembly
        - INLINE ASM (TallyswapFactory.sol#611-613)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Pragma version0.8.4 (TallyswapFactory.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in TallyswapPair._safeTransfer(address,address,uint256) (TallyswapFactory.sol#310-322):
        - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (TallyswapFactory.sol#315-317)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function ITallyswapPair.DOMAIN_SEPARATOR() (TallyswapFactory.sol#20) is not in mixedCase
Function ITallyswapPair.PERMIT_TYPEHASH() (TallyswapFactory.sol#21) is not in mixedCase
Function ITallyswapPair.MINIMUM_LIQUIDITY() (TallyswapFactory.sol#38) is not in mixedCase
Function ITallyswapCallee.TallyswapCall(address,uint256,uint256,bytes) (TallyswapFactory.sol#111) is not in mixedCase
Function ITallyswapERC20.DOMAIN_SEPARATOR() (TallyswapFactory.sol#128) is not in mixedCase
Function ITallyswapERC20.PERMIT_TYPEHASH() (TallyswapFactory.sol#129) is not in mixedCase
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable TallyswapPair.swap(uint256,uint256,address,bytes).balance0Adjusted (TallyswapFactory.sol#542-544) is too similar to TallyswapPa
.swap(uint256,uint256,address,bytes).balance1Adjusted (TallyswapFactory.sol#545-547)
Variable TallyswapPair.price0CumulativeLast (TallyswapFactory.sol#266) is too similar to TallyswapPair.price1CumulativeLast (TallyswapFa
ory.sol#267)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
```

```
INFO:Detectors:
TallyswapFactory.createPair(address,address) (TallyswapFactory.sol#604-619) uses literals with too many digits:
        - bytecode = type()(TallyswapPair).creationCode (TallyswapFactory.sol#609)
TallyswapFactory.slitherConstructorVariables() (TallyswapFactory.sol#586-641) uses literals with too many digits:
        - INIT_CODE_HASH = keccak256(bytes)(abi.encodePacked(type()(TallyswapPair).creationCode)) (TallyswapFactory.sol#589)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
TallyswapERC20.DOMAIN_SEPARATOR (TallyswapFactory.sol#183) should be constant
TallyswapPair.artisticAddr (TallyswapFactory.sol#283) should be constant
TallyswapPair.artisticFee (TallyswapFactory.sol#276) should be constant
TallyswapPair.factory (TallyswapFactory.sol#258) should be constant
TallyswapPair.liquidityProviderFee (TallyswapFactory.sol#279) should be constant
TallyswapPair.marketingAddr (TallyswapFactory.sol#281) should be constant
TallyswapPair.marketingFee (TallyswapFactory.sol#274) should be constant
TallyswapPair.operationsAddr (TallyswapFactory.sol#282) should be constant
TallyswapPair.operationsFee (TallyswapFactory.sol#275) should be constant
TallyswapPair.tallyBackFee (TallyswapFactory.sol#278) should be constant
TallyswapPair.technicalAddr (TallyswapFactory.sol#284) should be constant
TallyswapPair.technicalFee (TallyswapFactory.sol#277) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Slither:TallyswapFactory.sol analyzed (11 contracts with 75 detectors), 60 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> TallyswapRouter.sol

```
INFO:Detectors:
TallyswapRouter02.constructor(address,address)._factory (TallyswapRouter02.sol#516) lacks a zero-check on :
                - factory = _factory (TallyswapRouter02.sol#517)
TallyswapRouter02.constructor(address,address)._WETH (TallyswapRouter02.sol#516) lacks a zero-check on :
                - WETH = _WETH (TallyswapRouter02.sol#518)
TallyswapRouter02.setSwapFeeReward(address)._swapFeeReward (TallyswapRouter02.sol#525) lacks a zero-check on :
                - swapFeeReward = _swapFeeReward (TallyswapRouter02.sol#526)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
TallyswapRouter02._swap(uint256[],address[],address) (TallyswapRouter02.sol#709-723) has external calls inside a loop: ISwapFeeReward(swFeeReward).swap(msg.sender,input,output,amountOut) (TallyswapRouter02.sol#716)
TallyswapRouter02._swap(uint256[],address[],address) (TallyswapRouter02.sol#709-723) has external calls inside a loop: ITallyswapPair(TayswapLibrary.pairFor(factory,input,output)).swap(amount0Out,amount1Out,to,new bytes(0)) (TallyswapRouter02.sol#719-721)
TallyswapRouter02._swapSupportingFeeOnTransferTokens(address[],address) (TallyswapRouter02.sol#821-841) has external calls inside a loop (reserve0,reserve1) = pair.getReserves() (TallyswapRouter02.sol#829)
TallyswapRouter02._swapSupportingFeeOnTransferTokens(address[],address) (TallyswapRouter02.sol#821-841) has external calls inside a loop amountInput = IERC20(input).balanceOf(address(pair)).sub(reserveInput) (TallyswapRouter02.sol#831)
TallyswapRouter02._swapSupportingFeeOnTransferTokens(address[],address) (TallyswapRouter02.sol#821-841) has external calls inside a loop amountOutput = TallyswapLibrary.getAmountOut(amountInput,reserveInput,reserveOutput,pair.swapFee()) (TallyswapRouter02.sol#832)
TallyswapRouter02._swapSupportingFeeOnTransferTokens(address[],address) (TallyswapRouter02.sol#821-841) has external calls inside a loop ISwapFeeReward(swapFeeReward).swap(msg.sender,input,output,amountOutput) (TallyswapRouter02.sol#835)
TallyswapRouter02._swapSupportingFeeOnTransferTokens(address[],address) (TallyswapRouter02.sol#821-841) has external calls inside a loop pair.swap(amount0Out,amount1Out,to,new bytes(0)) (TallyswapRouter02.sol#839)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
INFO:Detectors:
Babylonian.sqrt(uint256) (TallyswapRouter02.sol#357-399) is never used and should be removed
FullMath.fullDiv(uint256,uint256,uint256) (TallyswapRouter02.sol#409-428) is never used and should be removed
FullMath.fullMul(uint256,uint256) (TallyswapRouter02.sol#402-407) is never used and should be removed
FullMath.mulDiv(uint256,uint256,uint256) (TallyswapRouter02.sol#430-445) is never used and should be removed
TransferHelper.safeApprove(address,address,uint256) (TallyswapRouter02.sol#56-67) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version0.8.4 (TallyswapRouter02.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in TransferHelper.safeApprove(address,address,uint256) (TallyswapRouter02.sol#56-67):
        - (success,data) = token.call(abi.encodeWithSelector(0x095ea7b3,to,value)) (TallyswapRouter02.sol#62)
Low level call in TransferHelper.safeTransfer(address,address,uint256) (TallyswapRouter02.sol#69-80):
        - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value)) (TallyswapRouter02.sol#75)
Low level call in TransferHelper.safeTransferFrom(address,address,address,uint256) (TallyswapRouter02.sol#82-94):
        - (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,to,value)) (TallyswapRouter02.sol#89)
Low level call in TransferHelper.safeTransferETH(address,uint256) (TallyswapRouter02.sol#96-99):
        - (success) = to.call{value: value}(new bytes(0)) (TallyswapRouter02.sol#97)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function ITallyswapRouter02.WETH() (TallyswapRouter02.sol#104) is not in mixedCase
Function ITallyswapPair.DOMAIN_SEPARATOR() (TallyswapRouter02.sol#463) is not in mixedCase
Function ITallyswapPair.PERMIT_TYPEHASH() (TallyswapRouter02.sol#464) is not in mixedCase
Function ITallyswapPair.MINIMUM_LIQUIDITY() (TallyswapRouter02.sol#481) is not in mixedCase
Parameter TallyswapRouter02.setSwapFeeReward(address)._swapFeeReward (TallyswapRouter02.sol#525) is not in mixedCase
Variable TallyswapRouter02.WETH (TallyswapRouter02.sol#508) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable ITallyswapRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (TallyswapRoute2.sol#110) is too similar to ITallyswapRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesed (TallyswapRouter02.sol#111)
Variable TallyswapRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (TallyswapRouter.sol#561) is too similar to ITallyswapRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesid (TallyswapRouter02.sol#111)
Variable TallyswapRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountADesired (TallyswapRouter02.sol#533) is o similar to TallyswapRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBDesired (TallyswapRouter02.sol#534)
Variable TallyswapRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (TallyswapRouter.sol#561) is too similar to TallyswapRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesir (TallyswapRouter02.sol#562)
Variable TallyswapRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountADesired (TallyswapRouter02.sol#533) is o similar to TallyswapRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (TallyswapRoer02.sol#562)
Variable TallyswapRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountADesired (TallyswapRouter02.sol#533) is o similar to ITallyswapRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (TallyswapRter02.sol#111)
```

```
INFO:Detectors:
Babylonian.sqrt(uint256) (TallyswapRouter02.sol#357-399) uses literals with too many digits:
        - xx >= 0x100000000000000000000000000000000 (TallyswapRouter02.sol#363)
Babylonian.sqrt(uint256) (TallyswapRouter02.sol#357-399) uses literals with too many digits:
        - xx >= 0x10000000000000000 (TallyswapRouter02.sol#367)
Babylonian.sqrt(uint256) (TallyswapRouter02.sol#357-399) uses literals with too many digits:
        - xx >= 0x100000000 (TallyswapRouter02.sol#371)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
owner() should be declared external:
        - Ownable.owner() (TallyswapRouter02.sol#18-20)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (TallyswapRouter02.sol#26-29)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (TallyswapRouter02.sol#37-39)
setSwapFeeReward(address) should be declared external:
        - TallyswapRouter02.setSwapFeeReward(address) (TallyswapRouter02.sol#525-527)
quote(uint256,uint256,uint256) should be declared external:
        - TallyswapRouter02.quote(uint256,uint256,uint256) (TallyswapRouter02.sol#906-908)
getAmountOut(uint256,uint256,uint256,uint256) should be declared external:
        - TallyswapRouter02.getAmountOut(uint256,uint256,uint256,uint256) (TallyswapRouter02.sol#910-918)
getAmountIn(uint256,uint256,uint256,uint256) should be declared external:
        - TallyswapRouter02.getAmountIn(uint256,uint256,uint256,uint256) (TallyswapRouter02.sol#920-928)
getAmountsOut(uint256,address[]) should be declared external:
        - TallyswapRouter02.getAmountsOut(uint256,address[]) (TallyswapRouter02.sol#930-938)
getAmountsIn(uint256,address[]) should be declared external:
```

```
getAmountIn(uint256,uint256,uint256,uint256) should be declared external:
        - TallyswapRouter02.getAmountIn(uint256,uint256,uint256,uint256) (TallyswapRouter02.sol#920-928)
getAmountsOut(uint256,address[]) should be declared external:
        - TallyswapRouter02.getAmountsOut(uint256,address[]) (TallyswapRouter02.sol#930-938)
getAmountsIn(uint256,address[]) should be declared external:
        - TallyswapRouter02.getAmountsIn(uint256,address[]) (TallyswapRouter02.sol#940-948)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:TallyswapRouter02.sol analyzed (13 contracts with 75 detectors), 65 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Tally.sol

```
INFO:Detectors:
Tally.allowance(address,address).owner (Tally.sol#811) shadows:
        - Ownable.owner() (Tally.sol#383-385) (function)
Tally._approve(address,address,uint256).owner (Tally.sol#1246) shadows:
        - Ownable.owner() (Tally.sol#383-385) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Tally.setMarketingWallet(address).account (Tally.sol#977) lacks a zero-check on :
                - _marketingWallet = account (Tally.sol#978)
Tally.setCharityWallet(address).account (Tally.sol#981) lacks a zero-check on :
                - _charityWallet = account (Tally.sol#982)
Tally.setBuyBackWallet(address).account (Tally.sol#985) lacks a zero-check on :
                - _buyBackWallet = account (Tally.sol#986)
Tally.setRaffleWallet(address).account (Tally.sol#989) lacks a zero-check on :
                - _raffleWallet = account (Tally.sol#990)
Tally.setPoolWallet(address).account (Tally.sol#993) lacks a zero-check on :
                - _poolWallet = account (Tally.sol#994)
Tally.setPoolCharityWallet(address).account (Tally.sol#997) lacks a zero-check on :
                - _poolCharityWallet = account (Tally.sol#998)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in Tally._transfer(address,address,uint256) (Tally.sol#1254-1298):
        External calls:
        - swapAndLiquify(contractTokenBalance) (Tally.sol#1285)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (Tally.sol#13
-1366)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (
lly.sol#1332-1338)
        - _tokenTransfer(from,to,amount,takeFee) (Tally.sol#1297)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (Tally.sol#13
-1366)
                - (success) = address(account).call{value: dividends}() (Tally.sol#1347)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (
```

```
        - require(bool,string)(now > _lockTime,Contract is locked until 7 days) (Tally.sol#432)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (Tally.sol#247-256) uses assembly
        - INLINE ASM (Tally.sol#254)
Address._functionCallWithValue(address,bytes,uint256,string) (Tally.sol#340-361) uses assembly
        - INLINE ASM (Tally.sol#353-356)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._functionCallWithValue(address,bytes,uint256,string) (Tally.sol#340-361) is never used and should be removed
Address.functionCall(address,bytes) (Tally.sol#300-302) is never used and should be removed
Address.functionCall(address,bytes,string) (Tally.sol#310-312) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (Tally.sol#325-327) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (Tally.sol#335-338) is never used and should be removed
Address.isContract(address) (Tally.sol#247-256) is never used and should be removed
Address.sendValue(address,uint256) (Tally.sol#274-280) is never used and should be removed
Context._msgData() (Tally.sol#223-226) is never used and should be removed
SafeMath.mod(uint256,uint256) (Tally.sol#196-198) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Tally.sol#212-215) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Tally._rTotal (Tally.sol#665) is set pre-construction with a non-constant function or state variable:
        - (MAX - (MAX % _tTotal))
Tally._previousTaxFee (Tally.sol#679) is set pre-construction with a non-constant function or state variable:
        - _taxFee
Tally._previousSellTaxFee (Tally.sol#682) is set pre-construction with a non-constant function or state variable:
        - _taxSellFee
Tally._previousBuyTaxFee (Tally.sol#685) is set pre-construction with a non-constant function or state variable:
        - _taxBuyFee
Tally._previousLiquidityFee (Tally.sol#688) is set pre-construction with a non-constant function or state variable:
        - _liquidityFee
Tally._previousBurnFee (Tally.sol#691) is set pre-construction with a non-constant function or state variable:
        - _burnFee
Tally._previousMarketingFee (Tally.sol#694) is set pre-construction with a non-constant function or state variable:
        - _marketingFee
Tally._previousCharityFee (Tally.sol#697) is set pre-construction with a non-constant function or state variable:
        - _charityFee
```

```
Tally._previousCharityFee (Tally.sol#697) is set pre-construction with a non-constant function or state variable:
        - _charityFee
Tally._previousBuyBackFee (Tally.sol#700) is set pre-construction with a non-constant function or state variable:
        - _buyBackFee
Tally._previousRaffleFee (Tally.sol#703) is set pre-construction with a non-constant function or state variable:
        - _raffleFee
Tally._previousPoolFee (Tally.sol#706) is set pre-construction with a non-constant function or state variable:
        - _poolFee
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state-variables
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (Tally.sol#274-280):
        - (success) = recipient.call{value: amount}() (Tally.sol#278)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (Tally.sol#340-361):
        - (success,returndata) = target.call{value: weiValue}(data) (Tally.sol#344)
Low level call in Tally.swapAndSendToWallet(uint256,address) (Tally.sol#1341-1352):
        - (success) = address(account).call{value: dividends}() (Tally.sol#1347)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (Tally.sol#470) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (Tally.sol#471) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (Tally.sol#488) is not in mixedCase
Function IUniswapV2Router01.WETH() (Tally.sol#508) is not in mixedCase
Parameter Tally.setSwapAndLiquifyEnabled(bool)._enabled (Tally.sol#1008) is not in mixedCase
Parameter Tally.calculateTaxFee(uint256,uint8)._amount (Tally.sol#1127) is not in mixedCase
Parameter Tally.calculateLiquidityFee(uint256)._amount (Tally.sol#1134) is not in mixedCase
Parameter Tally.calculateBurnFee(uint256)._amount (Tally.sol#1140) is not in mixedCase
Parameter Tally.calculateMarketingFee(uint256)._amount (Tally.sol#1146) is not in mixedCase
Parameter Tally.calculateCharityFee(uint256)._amount (Tally.sol#1152) is not in mixedCase
Parameter Tally.calculateBuyBackFee(uint256)._amount (Tally.sol#1158) is not in mixedCase
Parameter Tally.calculateRaffleFee(uint256)._amount (Tally.sol#1164) is not in mixedCase
Parameter Tally.calculatePoolFee(uint256)._amount (Tally.sol#1170) is not in mixedCase
Variable Tally._excluded (Tally.sol#653) is not in mixedCase
Variable Tally._taxFee (Tally.sol#678) is not in mixedCase
```

```
Variable Tally._maxTxAmountPercent (Tally.sol#714) is not in mixedCase
Variable Tally._maxTxAmount (Tally.sol#715) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (Tally.sol#224)" inContext (Tally.sol#218-227)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Tally.sol#513)
s too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Tally.
l#514)
Variable Tally._getValues(uint256,uint8).rTransferAmount (Tally.sol#1023) is too similar to Tally._getTValues(uint256,uint8).tTransferAm
nt (Tally.sol#1041)
Variable Tally._getRValues(uint256,uint256[9],uint256).rTransferAmount (Tally.sol#1059) is too similar to Tally._getTValues(uint256,uint
.tTransferAmount (Tally.sol#1041)
Variable Tally.reflectionFromToken(uint256,bool,uint8).rTransferAmount (Tally.sol#865) is too similar to Tally._getTValues(uint256,uint8
tTransferAmount (Tally.sol#1041)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
Tally.slitherConstructorVariables() (Tally.sol#642-1501) uses literals with too many digits:
        - _tTotal = 1000000000 * 10 ** 9 (Tally.sol#664)
Tally.slitherConstructorVariables() (Tally.sol#642-1501) uses literals with too many digits:
        - numTokensSellToAddToLiquidity = 500000 * 10 ** 6 * 10 ** 9 (Tally.sol#716)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
Tally._decimals (Tally.sol#670) should be constant
Tally._name (Tally.sol#668) should be constant
Tally._symbol (Tally.sol#669) should be constant
Tally._tTotal (Tally.sol#664) should be constant
Tally.numTokensSellToAddToLiquidity (Tally.sol#716) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-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() (Tally.sol#402-405)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Tally.sol#411-415)
geUnlockTime() should be declared external:
        - Ownable.geUnlockTime() (Tally.sol#417-419)
lock(uint256) should be declared external:
        - Ownable.lock(uint256) (Tally.sol#422-427)
unlock() should be declared external:
        - Ownable.unlock() (Tally.sol#430-435)
name() should be declared external:
        - Tally.name() (Tally.sol#785-787)
symbol() should be declared external:
        - Tally.symbol() (Tally.sol#789-791)
decimals() should be declared external:
        - Tally.decimals() (Tally.sol#793-795)
totalSupply() should be declared external:
        - Tally.totalSupply() (Tally.sol#797-799)
transfer(address,uint256) should be declared external:
        - Tally.transfer(address,uint256) (Tally.sol#806-809)
allowance(address,address) should be declared external:
        - Tally.allowance(address,address) (Tally.sol#811-813)
approve(address,uint256) should be declared external:
        - Tally.approve(address,uint256) (Tally.sol#815-818)
transferFrom(address,address,uint256) should be declared external:
        - Tally.transferFrom(address,address,uint256) (Tally.sol#820-824)
increaseAllowance(address,uint256) should be declared external:
        - Tally.increaseAllowance(address,uint256) (Tally.sol#826-829)
decreaseAllowance(address,uint256) should be declared external:
        - Tally.decreaseAllowance(address,uint256) (Tally.sol#831-834)
isExcludedFromReward(address) should be declared external:
        - Tally.isExcludedFromReward(address) (Tally.sol#836-838)
totalFees() should be declared external:
        - Tally.totalFees() (Tally.sol#840-842)
```

```
deliver(uint256) should be declared external:
        - Tally.deliver(uint256) (Tally.sol#844-854)
reflectionFromToken(uint256,bool,uint8) should be declared external:
        - Tally.reflectionFromToken(uint256,bool,uint8) (Tally.sol#856-868)
excludeFromReward(address) should be declared external:
        - Tally.excludeFromReward(address) (Tally.sol#876-884)
excludeFromFee(address) should be declared external:
        - Tally.excludeFromFee(address) (Tally.sol#920-922)
includeInFee(address) should be declared external:
        - Tally.includeInFee(address) (Tally.sol#924-926)
setSwapAndLiquifyEnabled(bool) should be declared external:
        - Tally.setSwapAndLiquifyEnabled(bool) (Tally.sol#1008-1011)
isExcludedFromFee(address) should be declared external:
        - Tally.isExcludedFromFee(address) (Tally.sol#1242-1244)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Tally.sol analyzed (10 contracts with 75 detectors), 154 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Solidity Static Analysis

SwapFeeReward.sol

Security

Check-effects-interaction:

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

Gas & Economy

Gas costs:

Gas requirement of function SwapFeeReward.sortTokens 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: 611:4:

Gas costs:

Gas requirement of function SwapFeeReward.pairFor 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: 617:4:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 424:4:

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 467:4:

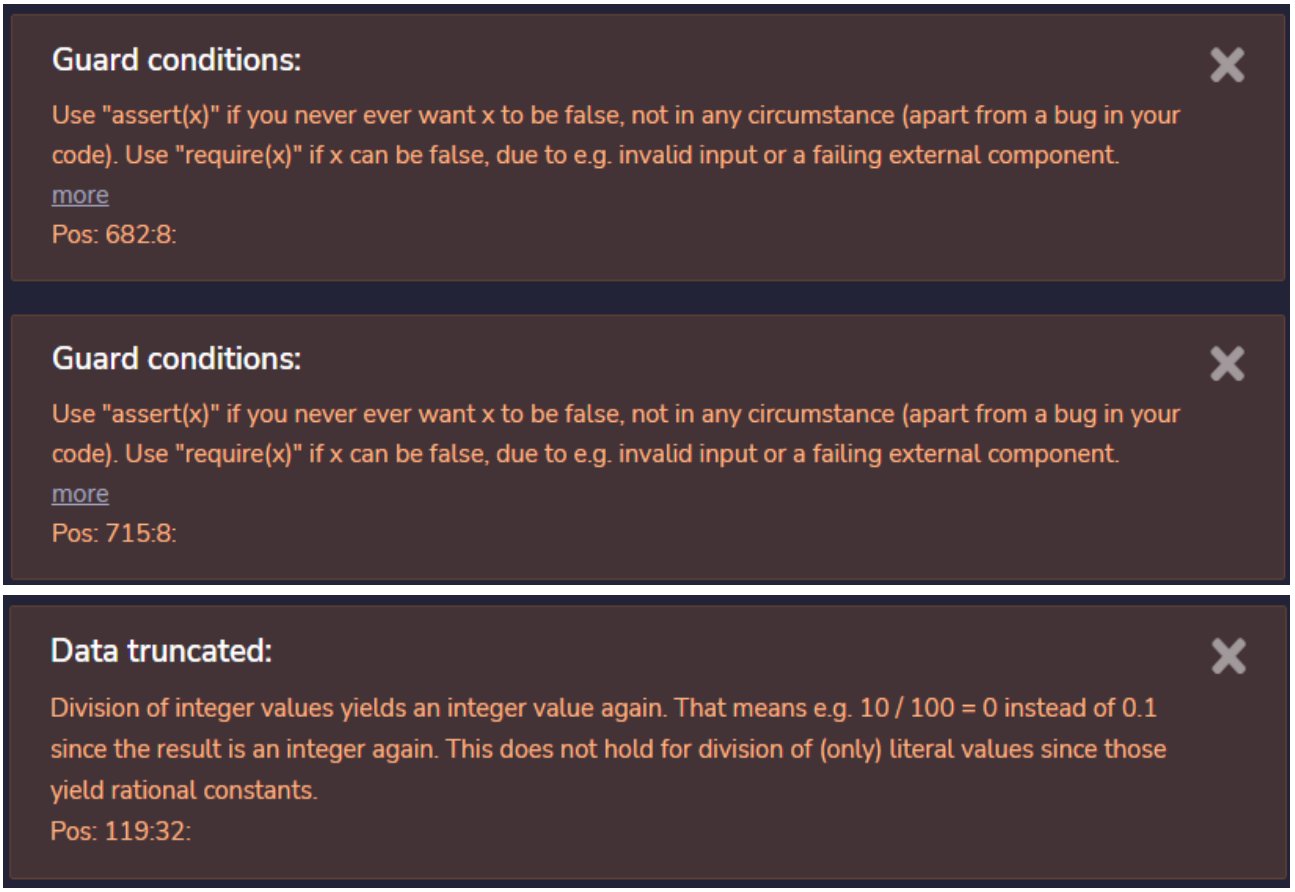
Miscellaneous

Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: Cannot read properties of undefined (reading 'name')
Pos: not available

Similar variable names:

SwapFeeReward.sortTokens(address,address) : Variables have very similar names "token0" and "tokenA". Note: Modifiers are currently not considered by this static analysis.
Pos: 613:46:



Tally.sol

Low level calls:

Use of "call": should be avoided whenever possible. It can lead to unexpected behavior if return value is not handled properly. Please use Direct Calls via specifying the called contract's interface.
more
Pos: 1347:26:

Gas & Economy

Gas costs:

Gas requirement of function Tally.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: 411:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.
more
Pos: 1073:8:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 461:4:

Miscellaneous

Constant/View/Pure functions:

SafeMath.sub(uint256,uint256) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 103:4:

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: 247:4:

Similar variable names:

Tally.() : Variables have very similar names "_tTotal" and "_rTotal". Note: Modifiers are currently not considered by this static analysis.
Pos: 782:48:

Similar variable names:

Tally.totalSupply() : Variables have very similar names "_tTotal" and "_rTotal". Note: Modifiers are currently not considered by this static analysis.
Pos: 798:15:

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: 1261:8:

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: 1263:12:

MasterChef.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.withdrawDevAndRefFee(): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 2412:4:

Low level calls:

Use of "call": should be avoided whenever possible. It can lead to unexpected behavior if return value is not handled properly. Please use Direct Calls via specifying the called contract's interface.
more
Pos: 2024:27:

Gas & Economy

Gas costs:

Gas requirement of function MasterChef.lock 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: 493:4:

Gas costs:

Gas requirement of function MasterChef.leaveStaking 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: 2636:4:

Gas costs:

Gas requirement of function MasterChef.emergencyWithdraw 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: 2657:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.
more
Pos: 1724:8:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 555:4:

Miscellaneous

Constant/View/Pure functions:

IERC20.transfer(address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 22:4:

Constant/View/Pure functions:

IMigratorChef.migrate(contract IERC20) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 2305:4:

No return:

IMigratorChef.migrate(contract IERC20): Defines a return type but never explicitly returns a value.
Pos: 2305:4:

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: 2701:8:

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: 2702:8:

TallyswapFactory.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in TallyswapPair._mintFee(uint112,uint112): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 393:4:

Inline assembly:

The Contract uses inline assembly, this is only advised in rare cases. Additionally static analysis modules do not parse inline Assembly, this can lead to wrong analysis results.

more

Pos: 611:8:

Gas & Economy

Gas costs:

Gas requirement of function TallyswapERC20.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: 176:4:

Gas costs:

Gas requirement of function TallyswapFactory.setDevFee 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: 631:4:

Gas costs:

Gas requirement of function TallyswapFactory.setSwapFee 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: 637:4:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type

more

Pos: 11:4:

Miscellaneous

Constant/View/Pure functions:

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

more

Pos: 16:4:

Similar variable names:

TallyswapFactory.createPair(address,address) : Variables have very similar names "token0" and "tokenB". Note: Modifiers are currently not considered by this static analysis.

Pos: 607:16:

Similar variable names:

TallyswapFactory.createPair(address,address) : Variables have very similar names "token0" and "tokenB". Note: Modifiers are currently not considered by this static analysis.

Pos: 608:24:

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: 622:8:

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: 627:8:

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: 476:18:

TallyswapRouter.sol

Security

Check-effects-interaction:

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

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 512:28:

Gas & Economy

Gas costs:

Gas requirement of function TallyswapRouter02.quote 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: 270:4:

Gas costs:

Gas requirement of function TallyswapRouter02.getAmountOut 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: 910:4:

Gas costs:

Gas requirement of function TallyswapRouter02.getAmountIn 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: 920:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.
more
Pos: 838:25:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 338:4:

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 454:4:

Miscellaneous

Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: Cannot read properties of undefined (reading 'name')
Pos: not available

Similar variable names:

TallyswapRouter02._swap(uint256[],address[],address) : Variables have very similar names "amount0Out" and "amount1Out". Note: Modifiers are currently not considered by this static analysis.
Pos: 720:28:

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: 894:8:

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: 900:8:

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: 441:27:

Solhint Linter

SwapFeeReward.sol

```
SwapFeeReward.sol:1:1: Error: Compiler version 0.8.4 does not satisfy
the r semver requirement
SwapFeeReward.sol:4:5: Error: Explicitly mark visibility of state
SwapFeeReward.sol:5:5: Error: Explicitly mark visibility of state
SwapFeeReward.sol:447:5: Error: Function name must be in mixedCase
SwapFeeReward.sol:476:5: Error: Function name must be in mixedCase
SwapFeeReward.sol:477:5: Error: Function name must be in mixedCase
SwapFeeReward.sol:494:5: Error: Function name must be in mixedCase
SwapFeeReward.sol:523:5: Error: Explicitly mark visibility in
function (Set ignoreConstructors to true if using solidity >=0.7.0)
SwapFeeReward.sol:567:20: Error: Variable name must be in mixedCase
SwapFeeReward.sol:598:9: Error: Variable name must be in mixedCase
SwapFeeReward.sol:600:9: Error: Variable name must be in mixedCase
SwapFeeReward.sol:612:35: Error: Use double quotes for string
literals
SwapFeeReward.sol:614:39: Error: Use double quotes for string
literals
SwapFeeReward.sol:676:80: Error: Use double quotes for string
literals
SwapFeeReward.sol:680:47: Error: Use double quotes for string
literals
SwapFeeReward.sol:682:80: Error: Use double quotes for string
literals
SwapFeeReward.sol:752:30: Error: Variable name must be in mixedCase
```

Tally.sol

```
Tally.sol:4:1: Error: Compiler version ^0.6.12 does not satisfy the r
semver requirement
Tally.sol:425:21: Error: Avoid making time-based decisions in your
business logic
Tally.sol:432:17: Error: Avoid making time-based decisions in your
business logic
Tally.sol:470:5: Error: Function name must be in mixedCase
Tally.sol:471:5: Error: Function name must be in mixedCase
Tally.sol:488:5: Error: Function name must be in mixedCase
Tally.sol:508:5: Error: Function name must be in mixedCase
Tally.sol:642:1: Error: Contract has 44 states declarations but
allowed no more than 15
Tally.sol:711:5: Error: Explicitly mark visibility of state
Tally.sol:1014:32: Error: Code contains empty blocks
Tally.sol:1337:13: Error: Avoid making time-based decisions in your
business logic
Tally.sol:1347:27: Error: Avoid using low level calls.
Tally.sol:1365:13: Error: Avoid making time-based decisions in your
business logic
```

Email: audit@EtherAuthority.io

MasterChef.sol

```
MasterChef.sol:2:1: Error: Compiler version 0.8.4 does not satisfy
the r semver requirement
MasterChef.sol:442:5: Error: Explicitly mark visibility in function
(Set ignoreConstructors to true if using solidity >=0.7.0)
MasterChef.sol:496:21: Error: Avoid making time-based decisions in
your business logic
MasterChef.sol:506:17: Error: Avoid making time-based decisions in
your business logic
MasterChef.sol:576:5: Error: Function name must be in mixedCase
MasterChef.sol:578:5: Error: Function name must be in mixedCase
MasterChef.sol:609:5: Error: Function name must be in mixedCase
MasterChef.sol:655:5: Error: Function name must be in mixedCase
MasterChef.sol:1109:1: Error: Contract has 45 states declarations but
allowed no more than 15
MasterChef.sol:1181:5: Error: Explicitly mark visibility of state
MasterChef.sol:1611:32: Error: Code contains empty blocks
MasterChef.sol:2014:13: Error: Avoid making time-based decisions in
your business logic
MasterChef.sol:2024:28: Error: Avoid using low level calls.
MasterChef.sol:2042:13: Error: Avoid making time-based decisions in
your business logic
MasterChef.sol:2308:1: Error: Contract has 20 states declarations but
allowed no more than 15
MasterChef.sol:2335:23: Error: Variable name must be in mixedCase
MasterChef.sol:2364:20: Error: Variable name must be in mixedCase
MasterChef.sol:2366:20: Error: Variable name must be in mixedCase
MasterChef.sol:2388:17: Error: Variable name must be in mixedCase
MasterChef.sol:2415:9: Error: Variable name must be in mixedCase
MasterChef.sol:2515:13: Error: Variable name must be in mixedCase
MasterChef.sol:2551:9: Error: Variable name must be in mixedCase
MasterChef.sol:2668:9: Error: Variable name must be in mixedCase
```

TallyswapFactory.sol

```
TallyswapFactory.sol:2:1: Error: Compiler version 0.8.4 does not
satisfy the r semver requirement
TallyswapFactory.sol:20:5: Error: Function name must be in mixedCase
TallyswapFactory.sol:21:5: Error: Function name must be in mixedCase
TallyswapFactory.sol:38:5: Error: Function name must be in mixedCase
TallyswapFactory.sol:82:5: Error: Explicitly mark visibility of state
TallyswapFactory.sol:111:5: Error: Function name must be in mixedCase
TallyswapFactory.sol:128:5: Error: Function name must be in mixedCase
TallyswapFactory.sol:129:5: Error: Function name must be in mixedCase
TallyswapFactory.sol:159:5: Error: Function name must be in mixedCase
TallyswapFactory.sol:176:37: Error: Constant name must be in
capitalized SNAKE_CASE
TallyswapFactory.sol:176:44: Error: Use double quotes for string
literals
TallyswapFactory.sol:177:37: Error: Constant name must be in
capitalized SNAKE_CASE
TallyswapFactory.sol:177:46: Error: Use double quotes for string
literals
```

```
TallyswapFactory.sol:178:36: Error: Constant name must be in
capitalized SNAKE_CASE
TallyswapFactory.sol:183:29: Error: Variable name must be in
mixedCase
TallyswapFactory.sol:235:29: Error: Avoid making time-based decisions
in your business logic
TallyswapFactory.sol:235:46: Error: Use double quotes for string
literals
TallyswapFactory.sol:238:17: Error: Use double quotes for string
literals
TallyswapFactory.sol:244:78: Error: Use double quotes for string
literals
TallyswapFactory.sol:250:1: Error: Contract has 22 states
declarations but allowed no more than 15
TallyswapFactory.sol:315:45: Error: Avoid using low level calls.
TallyswapFactory.sol:375:40: Error: Avoid making time-based decisions
in your business logic
TallyswapFactory.sol:589:29: Error: Variable name must be in
mixedCase
TallyswapFactory.sol:605:35: Error: Use double quotes for string
literals
TallyswapFactory.sol:607:39: Error: Use double quotes for string
literals
TallyswapFactory.sol:608:56: Error: Use double quotes for string
literals
TallyswapFactory.sol:611:9: Error: Avoid using inline assembly. It is
acceptable only in rare cases
TallyswapFactory.sol:622:44: Error: Use double quotes for string
literals
TallyswapFactory.sol:627:44: Error: Use double quotes for string
literals
TallyswapFactory.sol:632:44: Error: Use double quotes for string
literals
TallyswapFactory.sol:633:30: Error: Use double quotes for string
literals
TallyswapFactory.sol:638:44: Error: Use double quotes for string
literals
```

TallyswapRouter.sol

```
TallyswapRouter.sol:2:1: Error: Compiler version 0.8.4 does not
satisfy the r semver requirement
TallyswapRouter.sol:13:5: Error: Explicitly mark visibility in
function (Set ignoreConstructors to true if using solidity >=0.7.0)
TallyswapRouter.sol:62:45: Error: Avoid using low level calls.
TallyswapRouter.sol:65:13: Error: Use double quotes for string
literals
TallyswapRouter.sol:75:45: Error: Avoid using low level calls.
TallyswapRouter.sol:78:13: Error: Use double quotes for string
literals
TallyswapRouter.sol:89:45: Error: Avoid using low level calls.
TallyswapRouter.sol:92:13: Error: Use double quotes for string
literals
TallyswapRouter.sol:97:28: Error: Avoid using low level calls.
TallyswapRouter.sol:98:26: Error: Use double quotes for string
```

```
literals
TallyswapRouter.sol:104:5: Error: Function name must be in mixedCase
TallyswapRouter.sol:242:35: Error: Use double quotes for string
literals
TallyswapRouter.sol:244:39: Error: Use double quotes for string
literals
TallyswapRouter.sol:271:30: Error: Use double quotes for string
literals
TallyswapRouter.sol:272:47: Error: Use double quotes for string
literals
TallyswapRouter.sol:278:31: Error: Use double quotes for string
literals
TallyswapRouter.sol:279:50: Error: Use double quotes for string
literals
TallyswapRouter.sol:288:32: Error: Use double quotes for string
literals
TallyswapRouter.sol:289:50: Error: Use double quotes for string
literals
TallyswapRouter.sol:297:35: Error: Use double quotes for string
literals
TallyswapRouter.sol:308:35: Error: Use double quotes for string
literals
TallyswapRouter.sol:320:35: Error: Use double quotes for string
literals
TallyswapRouter.sol:324:35: Error: Use double quotes for string
literals
TallyswapRouter.sol:328:49: Error: Use double quotes for string
literals
TallyswapRouter.sol:402:67: Error: Avoid to use letters 'I', 'l', 'O'
as identifiers
TallyswapRouter.sol:410:9: Error: Avoid to use letters 'I', 'l', 'O'
as identifiers
TallyswapRouter.sol:435:10: Error: Avoid to use letters 'I', 'l', 'O'
as identifiers
TallyswapRouter.sol:443:24: Error: Use double quotes for string
literals
TallyswapRouter.sol:463:5: Error: Function name must be in mixedCase
TallyswapRouter.sol:464:5: Error: Function name must be in mixedCase
TallyswapRouter.sol:481:5: Error: Function name must be in mixedCase
TallyswapRouter.sol:508:39: Error: Variable name must be in mixedCase
TallyswapRouter.sol:512:29: Error: Avoid making time-based decisions
in your business logic
TallyswapRouter.sol:512:46: Error: Use double quotes for string
literals
TallyswapRouter.sol:516:35: Error: Variable name must be in mixedCase
TallyswapRouter.sol:548:55: Error: Use double quotes for string
literals
TallyswapRouter.sol:553:55: Error: Use double quotes for string
literals
TallyswapRouter.sol:614:40: Error: Use double quotes for string
literals
TallyswapRouter.sol:615:40: Error: Use double quotes for string
literals
TallyswapRouter.sol:732:62: Error: Use double quotes for string
literals
TallyswapRouter.sol:746:44: Error: Use double quotes for string
literals
TallyswapRouter.sol:760:34: Error: Use double quotes for string
literals
```

```
TallyswapRouter.sol:762:62: Error: Use double quotes for string
literals
TallyswapRouter.sol:774:48: Error: Use double quotes for string
literals
TallyswapRouter.sol:776:44: Error: Use double quotes for string
literals
TallyswapRouter.sol:791:48: Error: Use double quotes for string
literals
TallyswapRouter.sol:793:62: Error: Use double quotes for string
literals
TallyswapRouter.sol:809:34: Error: Use double quotes for string
literals
TallyswapRouter.sol:811:42: Error: Use double quotes for string
literals
TallyswapRouter.sol:856:13: Error: Use double quotes for string
literals
TallyswapRouter.sol:871:34: Error: Use double quotes for string
literals
TallyswapRouter.sol:879:13: Error: Use double quotes for string
literals
TallyswapRouter.sol:894:48: Error: Use double quotes for string
literals
TallyswapRouter.sol:900:44: Error: Use double quotes for string
literals
```

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