Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Customer:	ViralataSwap
Website:	viralata.finance
Platform:	Binance Smart Chain
Language:	Solidity
Date:	August 12th, 2021

Table of contents

Email: audit@EtherAuthority.io

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

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by the viralataSwap team to perform the Security audit of the viralataSwap 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 August 12th, 2021.

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

ViraLata Finance ($REAU) is an open source, decentralized project, created to help include Brazilians in the cryptocurrency world. REAU provides a safe financial system that is easy to use, and helps make a difference through charitable donations.

Audit scope

Name	**Code Review and Security Analysis Report for ViralataSwap Protocol Smart Contracts**
Platform	**BSC / Solidity**
File 1	ViralataSwapToken.sol
Smart Contract Online Code 1	https://github.com/viralatafinance/viralataswap-contracts/blob/master/contracts/auro/ViralataSwapToken.sol
File 1 MD5 Hash	7E47EDA063062D2AC9404FF7A5786A84
File 2	AuroDistributor.sol
Smart Contract Online Code 2	https://github.com/viralatafinance/viralataswap-contracts/blob/master/contracts/farm/AuroDistributor.sol
File 2 MD5 Hash	9D3825888CA6F3BE27C606B9A033D4E5
File 3	ViralataERC20.sol
Smart Contract Online Code 3	https://github.com/viralatafinance/viralataswap-contracts/blob/master/contracts/uniswapv2/ViralataERC20.sol

File 3 MD5 Hash	010A7C4ECA08ED7C90564BBC87EE7931
File 4	ViralataFactory.sol
Smart Contract Online Code 4	https://github.com/viralatafinance/viralataswap-contracts/blob/master/contracts/uniswapv2/ViralataFactory.sol
File 4 MD5 Hash	163E8A023D032EE63EB68D5DCACB6536
File 5	ViralataPair.sol
Smart Contract Online Code 5	https://github.com/viralatafinance/viralataswap-contracts/blob/master/contracts/uniswapv2/ViralataPair.sol
File 5 MD5 Hash	8B6CD8E61A452B53541F8158CDEAD737
File 6	ViralataRouter02.sol
Smart Contract Online Code 6	https://github.com/viralatafinance/viralataswap-contracts/blob/master/contracts/uniswapv2/ViralataRouter02.sol
File 6 MD5 Hash	99397B16488F8762B365E280CD956488
Audit Date	August 12th, 2021
Revision Commit	ad70257437306a7e3d51806d351a254a7f213d25
Revision Date	August 21st, 2021

PS: There are 10 external imports from open zeppelin. These files are not included in the audit scope and thus they are not audited.

Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
File 1: ViralataSwapToken.sol • Name: ViralataSwap Token • Symbol: AURO • Maximum Supply is 20 billion tokens. • Initial Supply is 200,000 tokens.	**YES, This is valid.**
File 2: AuroDistributor.sol • Max harvest interval: 14 days. • Maximum deposit fee rate: 10%	**YES, This is valid.**
File 3: ViralataERC20.sol • Name: ViralataSwap LP Token • Symbol: VLP • Decimals: 18	**YES, This is valid.**
File 4: ViralataFactory.sol • The ViralataFactory owner can access functions like create Pair, set Migrator, set Fee To Setter, etc.	**YES, This is valid.**
File 5: ViralataPair.sol • Name: ViralataSwap LP Token • Symbol: VLP • Decimals: 18	**YES, This is valid.**
File 6: ViralataRouter02.sol • The ViralataRouter02 owner can add Liquidity, remove Liquidity, swap, etc.	**YES, This is valid.**

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **"Secured"**. These contracts also have owner functions (described in the centralization section below), which does not make everything 100% decentralized. Thus, the owner must execute those smart contract functions as per the business plan.

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

We found 0 critical, 0 high, 0 medium and 4 low and some very low level issues.
These issues are fixed/acknowledged in the revised smart contract.

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

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Moderated
	Integer overflow/underflow	Passed
	Function input parameters lack of check	Moderated
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Moderated
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	Passed
	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
	Other code specification issues	Moderated
Gas Optimization	“Out of Gas” Issue	Passed
	High consumption ‘for/while’ loop	Passed
	High consumption ‘storage’ storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage not set	Passed
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: **PASSED**

Code Quality

These audit scope have 6 smart contracts. These smart contracts also contain Libraries, Smart contracts inherits and Interfaces. These are compact and well written contracts.

The libraries in the ViralataSwap 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 ViralataSwap Protocol.

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

Some code parts are not well commented on smart contracts.

Documentation

We were given a ViralataSwap smart contracts code in the form of a github link. The hash of that code is mentioned above in the table.

As mentioned above, some code parts are **not well** commented. So it is difficult 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://viralata.finance/ which provided rich information about the project architecture and tokenomics.

Use of Dependencies

As per our observation, the libraries are used in this smart contract infrastructure that are based on well known industry standard open source projects. And their core code blocks are written well.

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

AS-IS overview

ViralataSwapToken.sol

(1) Interface

(a) IERC20

(b) IERC20Metadata

(c) IERC20Permit

(d) IERC165

(e) IAccessControl

(2) Inherited contracts

(a) ERC20Permit

(b) Pausable

(c) AccessControl

(d) ERC20

(3) Usages

(a) using SafeERC20 for IERC20;

(b) using SafeMath for uint256;

(4) Events

(a) event TokensRescued(address indexed sender, address indexed token, uint256 value);

(b) event MetaTxnsEnabled(address indexed caller);

(c) event MetaTxnsDisabled(address indexed caller);

(5) Functions

Sl.	Functions	Type	Observation	Conclusion
1	isTrustedForwarder	read	Passed	No Issue
2	_msgSender	internal	Passed	No Issue
3	_msgData	internal	Passed	No Issue
4	maxSupply	read	Passed	No Issue
5	pause	write	access only Role	No Issue
6	unpause	write	access only Role	No Issue
7	mint	write	access only Role	No Issue
8	_beforeTokenTransfer	internal	Passed	No Issue

9	rescueTokens	external	access only Role	No Issue
10	enableMetaTxns	write	access only Role	No Issue
11	disableMetaTxns	write	access only Role	No Issue
12	name	read	Passed	No Issue
13	symbol	read	Passed	No Issue
14	decimals	read	Passed	No Issue
15	totalSupply	read	Passed	No Issue
16	balanceOf	read	Passed	No Issue
17	transfer	write	Passed	No Issue
18	allowance	read	Passed	No Issue
19	approve	write	Passed	No Issue
20	transferFrom	write	Passed	No Issue
21	increaseAllowance	write	Passed	No Issue
22	decreaseAllowance	write	Passed	No Issue
23	_transfer	internal	Passed	No Issue
24	_mint	internal	Passed	No Issue
25	_burn	internal	Passed	No Issue
26	_approve	internal	Passed	No Issue
27	_beforeTokenTransfer	internal	Passed	No Issue
28	_afterTokenTransfer	internal	Passed	No Issue
29	permit	write	Passed	No Issue
30	nonces	read	Passed	No Issue
31	DOMAIN_SEPARATOR	external	Passed	No Issue
32	_useNonce	read	Passed	No Issue
33	paused	read	Passed	No Issue
34	whenNotPaused	modifier	Passed	No Issue
35	whenPaused	modifier	Passed	No Issue
36	_pause	internal	Passed	No Issue
37	_unpause	internal	Passed	No Issue
38	onlyRole	modifier	Passed	No Issue
39	supportsInterface	read	Passed	No Issue
40	hasRole	read	Passed	No Issue
41	_checkRole	internal	Passed	No Issue
42	getRoleAdmin	read	Passed	No Issue
43	grantRole	write	Passed	No Issue
44	revokeRole	write	Passed	No Issue
45	renounceRole	write	Passed	No Issue
46	_setupRole	internal	Passed	No Issue
47	_setRoleAdmin	internal	Passed	No Issue
48	_grantRole	write	Passed	No Issue
49	_revokeRole	write	Passed	No Issue

AuroDistributor.sol

(1) Interface

(a) IERC20

(b) IAuroERC20

(2) Inherited contracts

(a) Ownable

(b) ReentrancyGuard

(3) Usages

(a) using SafeMath for uint256;

(b) using SafeERC20 for IERC20;

(4) Struct

(a) UserInfo

(5) Events

(a) event Deposit(address indexed user, uint256 indexed pid, uint256 amount);

(b) event Withdraw(address indexed user, uint256 indexed pid, uint256 amount);

(c) event EmergencyWithdraw(address indexed user, uint256 indexed pid, uint256 amount);

(d) event EmissionRateUpdated(address indexed caller, uint256 previousAmount, uint256 newAmount);

(e) event RewardLockedUp(address indexed user, uint256 indexed pid, uint256 amountLockedUp);

(f) event OperatorTransferred(address indexed previousOperator, address indexed newOperator);

(g) event DevAddressChanged(address indexed caller, address oldAddress, address newAddress);

(h) event FeeAddressChanged(address indexed caller, address oldAddress, address newAddress);

(i) event AllocPointsUpdated(address indexed caller, uint256 previousAmount, uint256 newAmount);

(j) event MetaTxnsEnabled(address indexed caller);

(k) event MetaTxnsDisabled(address indexed caller);

(6) Functions

Sl.	Functions	Type	Observation	Conclusion
1	onlyOperator	modifier	Passed	No Issue
2	isTrustedForwarder	read	Passed	No Issue
3	_msgSender	internal	Passed	No Issue
4	_msgData	internal	Passed	No Issue
5	operator	read	Passed	No Issue
6	getMultiplier	write	Passed	No Issue
7	transferOperator	write	access only Operator	No Issue
8	startFarming	write	Infinite loop possibility	Refer audit finding section
9	poolLength	external	Passed	No Issue
10	add	write	Critical operation lacks event log	Refer audit finding section
11	set	write	Critical operation lacks event log	Refer audit finding section
12	pendingAuro	external	Range validation missing	Refer audit finding section
13	canHarvest	read	Passed	No Issue
14	massUpdatePools	write	Infinite loop possibility	Refer audit finding section
15	updatePool	write	Critical operation lacks event log	Refer audit finding section
16	deposit	write	Other code specification issues	Refer audit finding section
17	withdraw	write	Passed	No Issue
18	emergencyWithdraw	write	Passed	No Issue
19	payOrLockupPendingAuro	internal	Passed	No Issue
20	safeAuroTransfer	internal	Passed	No Issue
21	setDevAddress	write	Passed	No Issue
22	setFeeAddress	write	Passed	No Issue
23	updateEmissionRate	write	access only Operator	No Issue
24	updateAllocPoint	write	access only Operator	No Issue
25	enableMetaTxns	write	access only Operator	No Issue
26	disableMetaTxns	write	access only Operator	No Issue
27	onlyOwner	modifier	Passed	No Issue
28	renounceOwnership	write	access only Owner	No Issue
29	transferOwnership	write	access only Owner	No Issue
30	_setOwner	write	Passed	No Issue
31	nonReentrant	modifier	Passed	No Issue

ViralataERC20.sol

(1) Usages

(a) using SafeMathViralata for uint;

(2) Events

(a) event Approval(address indexed owner, address indexed spender, uint value);

(b) event Transfer(address indexed from, address indexed to, uint value);

(c) event MetaTxnsEnabled(address indexed caller);

(d) event MetaTxnsDisabled(address indexed caller);

(3) Functions

Sl.	Functions	Type	Observation	Conclusion
1	isTrustedForwarder	read	Passed	No Issue
2	_msgSender	internal	Passed	No Issue
3	_msgData	internal	Passed	No Issue
4	_mint	internal	Passed	No Issue
5	_burn	internal	Passed	No Issue
6	_approve	write	Passed	No Issue
7	_transfer	write	Passed	No Issue
8	approve	external	Passed	No Issue
9	transfer	external	Passed	No Issue
10	transferFrom	external	Passed	No Issue
11	permit	external	Passed	No Issue

ViralataFactory.sol

(1) Interface

(a) IViralataFactory

(b) IERC20Viralata

(c) IViralataFactory

(d) IViralataCallee

(e) IMigrator

(2) Inherited contracts

(a) IViralataFactory

(3) Events

(a) event PairCreated(address indexed token0, address indexed token1, address pair, uint);

(4) Functions

Sl.	Functions	Type	Observation	Conclusion
1	allPairsLength	external	Passed	No Issue
2	createPair	external	Passed	No Issue
3	setFeeTo	external	Passed	No Issue
4	setMigrator	external	Passed	No Issue
5	setFeeToSetter	external	Passed	No Issue
6	enableMetaTxnsPair	external	Passed	No Issue
7	disableMetaTxnsPair	external	Passed	No Issue

ViralataPair.sol

(1) Interface

(a) IERC20Viralata

(b) IViralataFactory

(c) IViralataCallee

(d) IMigrator

(2) Inherited contracts

(a) ViralataERC20

(3) Usages

(a) using SafeMathViralata for uint;

(b) using UQ112x112 for uint224;

(4) Events

(a) event Mint(address indexed sender, uint amount0, uint amount1);

(b) event Burn(address indexed sender, uint amount0, uint amount1, address indexed to);

(c) event Swap(address indexed sender, uint amount0In,uint amount1In, uint amount0Out,uint amount1Out, address indexed to);

(d) event Sync(uint112 reserve0, uint112 reserve1);

(5) Functions

Sl.	Functions	Type	Observation	Conclusion
1	lock	modifier	Passed	No Issue
2	getReserves	read	Passed	No Issue
3	_safeTransfer	write	Passed	No Issue
4	initialize	external	Passed	No Issue
5	_update	write	Passed	No Issue
6	_mintFee	write	Passed	No Issue
7	mint	external	Passed	No Issue
8	burn	external	Passed	No Issue
9	swap	external	Passed	No Issue
10	skim	external	Passed	No Issue
11	sync	external	Passed	No Issue
12	disableMetaTxns	external	Passed	No Issue
13	enableMetaTxns	external	Passed	No Issue
14	isTrustedForwarder	read	Passed	No Issue
15	_msgSender	internal	Passed	No Issue
16	_msgData	internal	Passed	No Issue
17	_mint	internal	Passed	No Issue
18	_burn	internal	Passed	No Issue
19	_approve	write	Passed	No Issue
20	_transfer	write	Passed	No Issue
2`	approve	external	Passed	No Issue
22	transfer	external	Passed	No Issue
23	transferFrom	external	Passed	No Issue
24	permit	external	Passed	No Issue

ViralataRouter02.sol

(1) Interface

(a) IViralataPair

(b) IERC20Viralata

(c) IWETH

(d) IViralataFactory

(e) IViralataRouter01

(f) IViralataRouter02

(2) Usages

(a) using SafeMathViralata for uint;

(3) Functions

Sl.	Functions	Type	Observation	Conclusion
1	ensure	modifier	Passed	No Issue
2	receive	external	Passed	No Issue
3	_addLiquidity	internal	Passed	No Issue
4	addLiquidity	external	Passed	No Issue
5	addLiquidityETH	external	Passed	No Issue
6	removeLiquidity	write	Passed	No Issue
7	removeLiquidityETH	write	Passed	No Issue
8	removeLiquidityWithPermit	write	Passed	No Issue
9	removeLiquidityETHWithPermit	external	Passed	No Issue
10	removeLiquidityETHSupportingFeeOnTransferTokens	external	Passed	No Issue
11	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	external	Passed	No Issue
12	_swap	internal	Passed	No Issue
13	swapExactTokensForTokens	external	Passed	No Issue
14	swapTokensForExactTokens	external	Passed	No Issue
15	swapExactETHForTokens	external	Passed	No Issue
16	swapTokensForExactETH	external	Passed	No Issue
17	swapExactTokensForETH	external	Passed	No Issue
18	swapETHForExactTokens	external	Passed	No Issue
19	_swapSupportingFeeOnTransferTokens	internal	Passed	No Issue
20	swapExactTokensForTokensSupportingFeeOnTransferTokens	external	Passed	No Issue
21	swapExactETHForTokensSupportingFeeOnTransferTokens	external	Passed	No Issue
22	swapExactTokensForETHSupportingFeeOnTransferTokens	external	Passed	No Issue
23	quote	write	Passed	No Issue
24	getAmountOut	write	Passed	No Issue
25	getAmountIn	write	Passed	No Issue
26	getAmountsOut	read	Passed	No Issue
27	getAmountsIn	read	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.

Audit Findings

ViralataSwapToken.sol

Critical

No Critical severity vulnerabilities were found.

High

No High severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

(1) Missing zero address validation:

```
constructor(address trustedForwarder) ERC20("ViralataSwap Token", "AURO")
ERC20Permit("ViralataSwap Token") {
    _setupRole(DEFAULT_ADMIN_ROLE, _msgSender());
    _setupRole(PAUSER_ROLE, _msgSender());
    _setupRole(MINTER_ROLE, _msgSender());
    _setupRole(RESCUER_ROLE, _msgSender());

    _trustedForwarder = trustedForwarder;

    _mint(_msgSender(), _initialSupply);
}
```

Detects missing zero address validation.

Resolution: Line no : 36 Check that the address is not zero.

Status: **acknowledged**

Very Low / Discussion / Best practices:

(1) Use latest solidity version:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.2;
```

Using the latest solidity will prevent any compiler level bugs.

Resolution: Please use 0.8.7 which is the latest version.

Status: **acknowledged**

(2) Function input parameters lack of check:

Variable validation is not performed in below functions : mint = amount | rescueTokens = value & address validation missing In isTrustedForwarder function.

Resolution: There should be some validations to check the variable is not empty or greater than 0 and address validation.

Status: **acknowledged**

(3) Make variables constant:

```
uint256 private _maxSupply = 2000000000 * 10**decimals();
// 2 billion tokens is maximum supply
uint256 private _initialSupply = 200000 * 10**decimals();
// 200,000 tokens is the initial supply
```

._maxSupply, _initialSupply. These variables will not be changed. So, please make it constant. It will save some gas.

Resolution: Declare those variables as constant.

Status: acknowledged

AuroDistributor.sol

Critical

No Critical severity vulnerabilities were found.

High

No High severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

(1) Hardcoded address used:

```
// remember to change for mainnet deploy
address constant _trustedForwarder = 0xEa9983bBb6FD1f95cd0A40275c6aC51B56Ae6176;
```

If the private key of the hard coded address is compromised, then it may create problems.

Resolution: Line no : 14 don't use hard coded addresses or make sure to handle the private key of it very securely.

Status: acknowledged

(2) Infinite loop possibility:

```
// Set farming start, can call only once
function startFarming() public onlyOwner {
    require(block.number < startBlock, "Error::Farm started already");

    uint256 length = poolInfo.length;
    for (uint256 pid = 0; pid < length; ++pid) {
        PoolInfo storage pool = poolInfo[pid];
        pool.lastRewardBlock = block.number;
    }

    startBlock = block.number;
}
```

```
// Update reward vairables for all pools. Be careful of gas spending!
function massUpdatePools() public {
    uint256 length = poolInfo.length;
    for (uint256 pid = 0; pid < length; ++pid) {
        updatePool(pid);
    }
}
```

If there are so many Pools, then this logic will fail, as it might hit the block's gas limit. If there are very limited pools, then this will work, but will cost more gas.

Resolution: Line no : 153, Line no : 230.

Status: **acknowledged**

(3) Missing zero address validation:

```
constructor(
    IAuroERC20 _auro,
    uint256 _auroPerBlock
) {
    //StartBlock always many years later from contract construct, will be
    //set later in StartFarming function
    startBlock = block.number + (10 * 365 * 24 * 60 * 60);

    auro = _auro;
    auroPerBlock = _auroPerBlock;

    devAddress = msg.sender;
    feeAddress = msg.sender;
    _operator = msg.sender;
    emit OperatorTransferred(address(0), _operator);
}
```

Detects missing zero address validation.

Resolution: Line no : 98 Check that the address is not zero.

Status: **acknowledged**

Very Low / Discussion / Best practices:

(1) Use latest solidity version:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.2;
```

Using the latest solidity will prevent any compiler level bugs.

Resolution: Please use 0.8.6 which is the latest version.

Status: **acknowledged**

(2) Function input parameters lack of check:

Variable validation is not performed in below function :

- getMultiplier -- from , to.

Resolution: There should be some validations to check the variable is not empty or greater than 0 and address validation.

Status: **acknowledged**

(3) Critical operation lacks event log:

Emit/ Event log is not written for add, set, updatePool, massUpdatePools.

Resolution: Please add emit for all listed functions.

Status: acknowledged

(4) External instead of public:

If any function is not called from inside the smart contract, then it is better to declare it as external instead of public. As it saves some gas as well.

https://ethereum.stackexchange.com/questions/19380/external-vs-public-best-practices

Status: acknowledged

ViralataERC20.sol

Critical

No Critical severity vulnerabilities were found.

High

No High severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

No Low severity vulnerabilities were found.

Very Low / Informational / Best practices

(1) Unused functions

```
function _mint(address to, uint value) internal {
    totalSupply = totalSupply.add(value);
    balanceOf[to] = balanceOf[to].add(value);
    emit Transfer(address(0), to, value);
}

function _burn(address from, uint value) internal {
    balanceOf[from] = balanceOf[from].sub(value);
    totalSupply = totalSupply.sub(value);
    emit Transfer(from, address(0), value);
}
```

The functions _mint and _burn are not called from any other functions. And they are internal functions, so they can not be called from outside.

Resolution: We suggest implementing the logic which uses them. And if not needed, then they can be removed to make the code clean.

Status: **acknowledged**

ViralataFactory.sol, ViralataPair.sol, ViralataRouter02.sol

No Critical, High, Low severity vulnerabilities were found.

Centralization

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

- startFarming: The AuroDistributor Owner can Set farming start, can call only once.
- add: The AuroDistributor Owner can add multiple pools with the same lp token without messing up rewards, because each pool's balance is tracked using its own totalLp.
- set: The AuroDistributor Owner can update the given poole's AURO allocation point and deposit fee.
- pause and unpause: The ViralataSwapToken Pauser wallet can pause and unpause the token transfer
- mint: The ViralataSwapToken Minter wallet can mint new tokens capped at max supply.
- rescueTokens: The ViralataSwapToken Rescuer wallet can take any tokens from its own smart contract to another wallet.
- setFeeTo and setMigrator in ViralataFactory contract can be set by the feeTo wallet.
- setFeeToSetter, setAuroAddress, enableMetaTxnsPair and disableMetaTxnsPair in ViralataFactory contract can be set by the feeToSetter wallet.

Conclusion

We were given a contract code. And we have used all possible tests based on given objects as files. We observed some issues, but they are not critical. So, **it's good to go to production**.

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

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

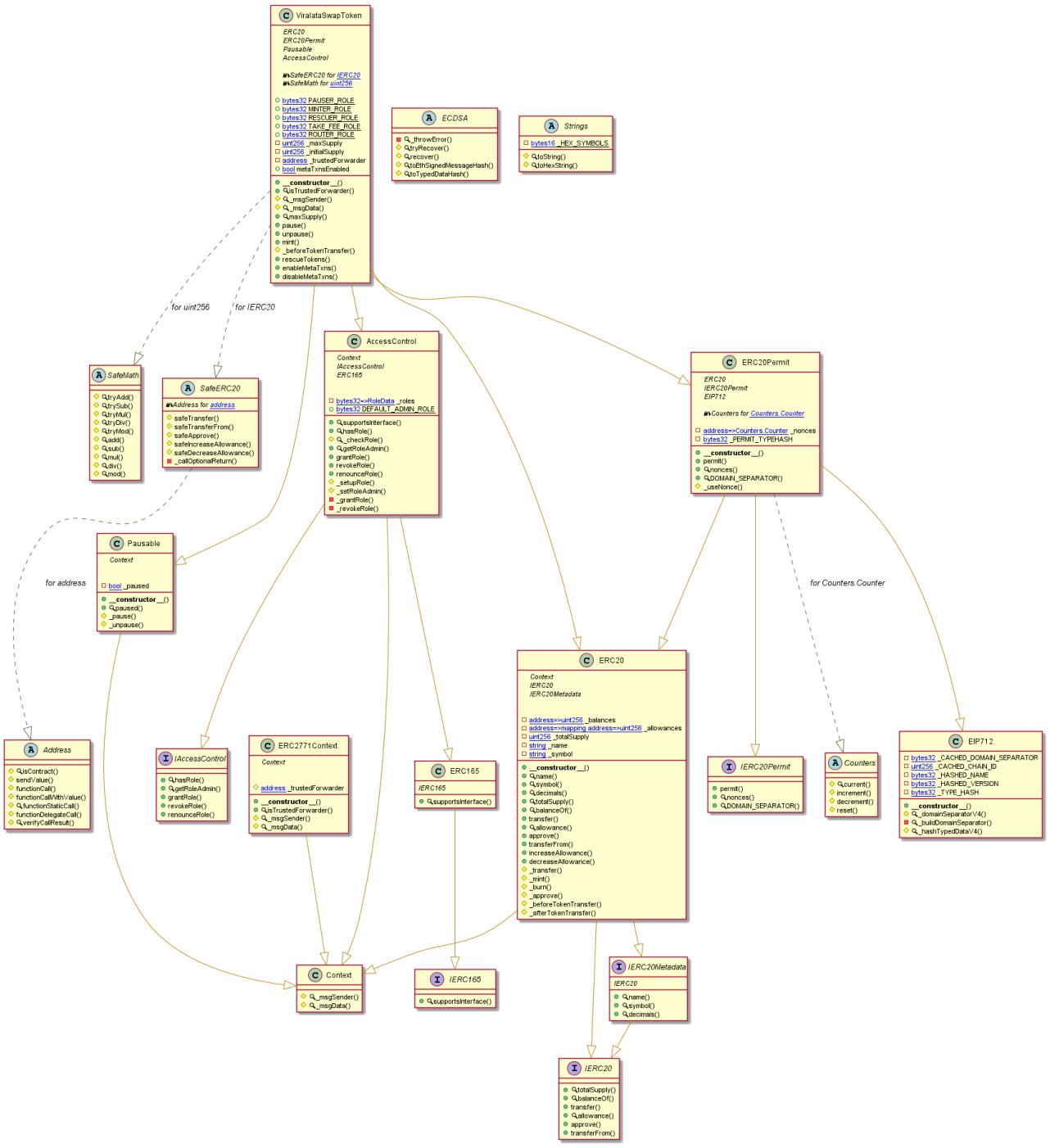
Technical Disclaimer

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.

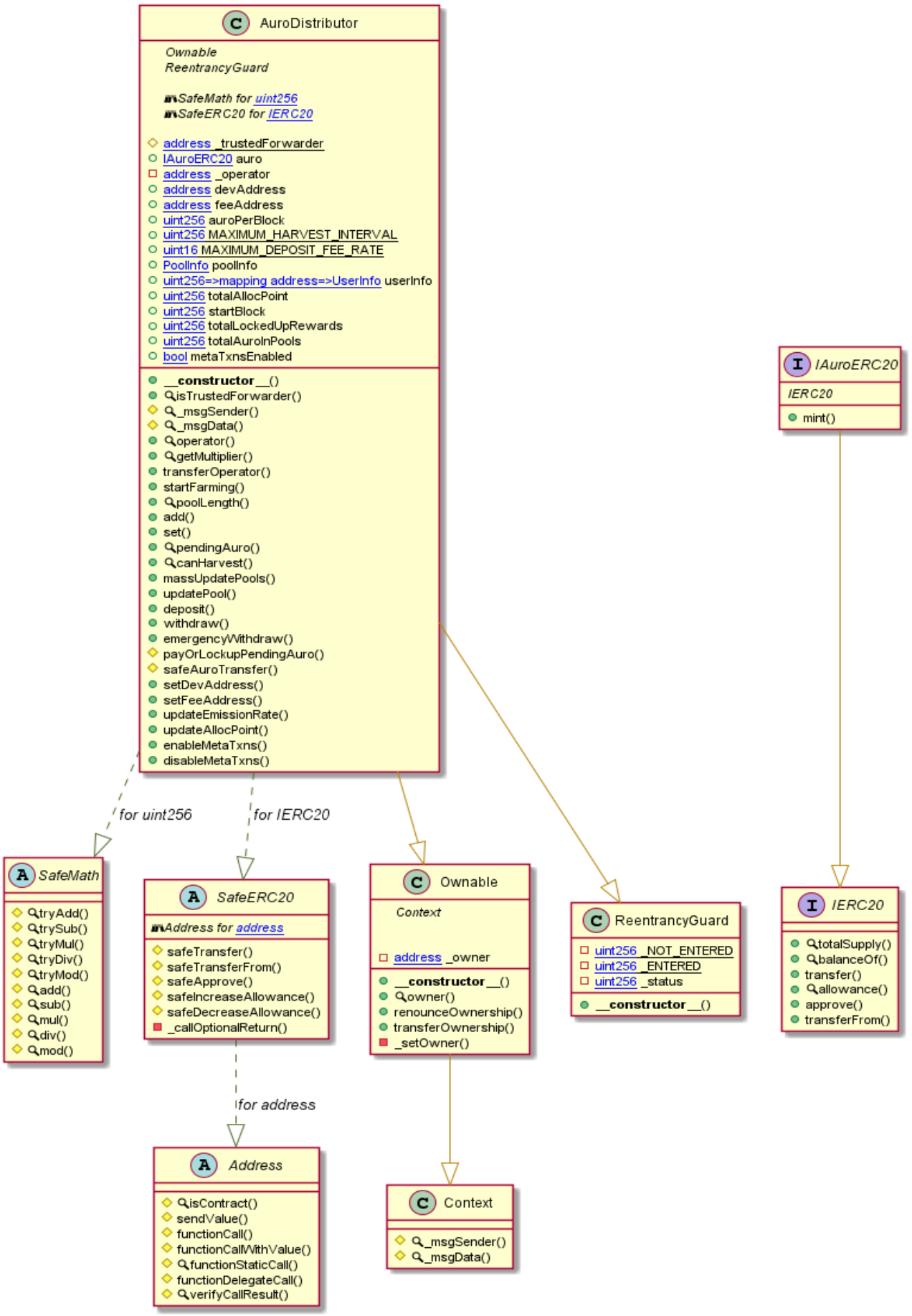
Appendix

Code Flow Diagram - ViralataSwap

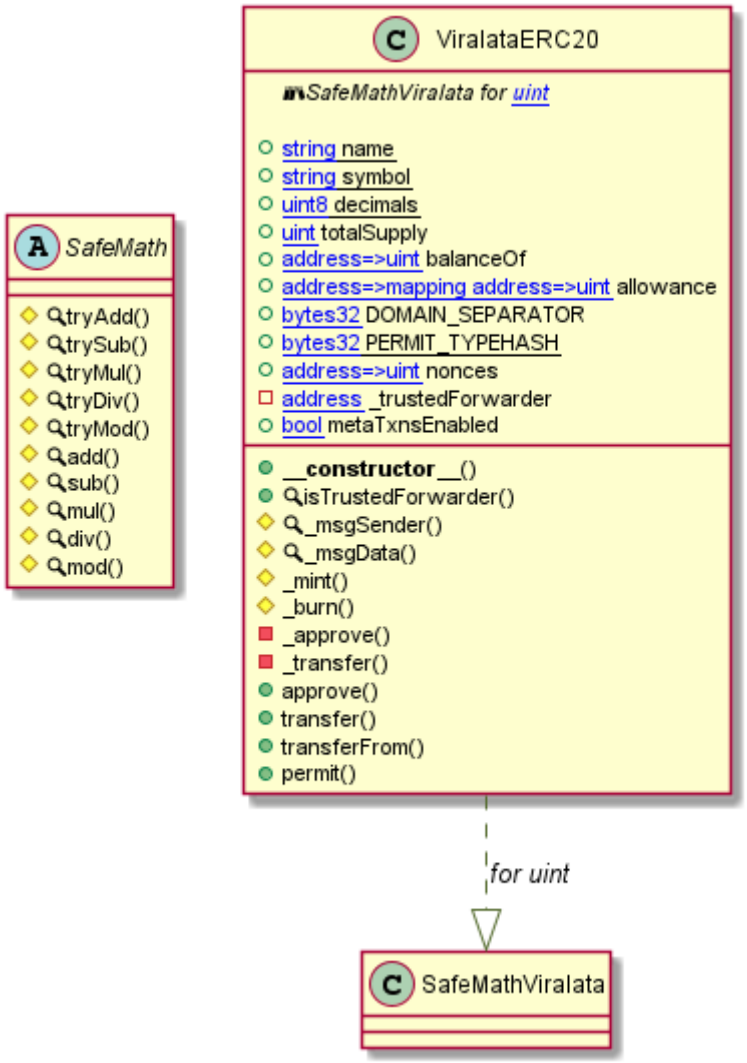
ViralataSwap Diagram



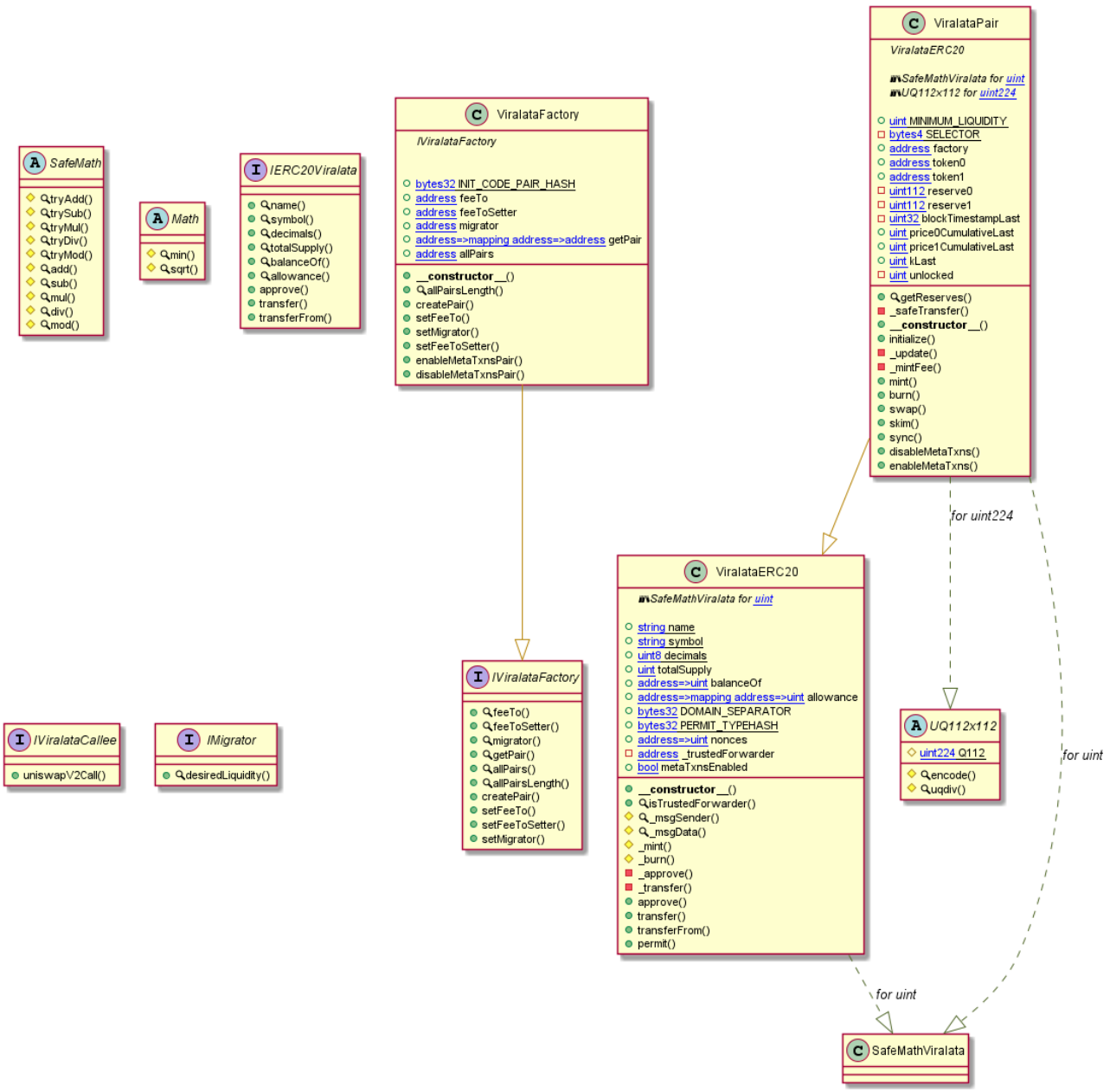
AuroDistributor Diagram



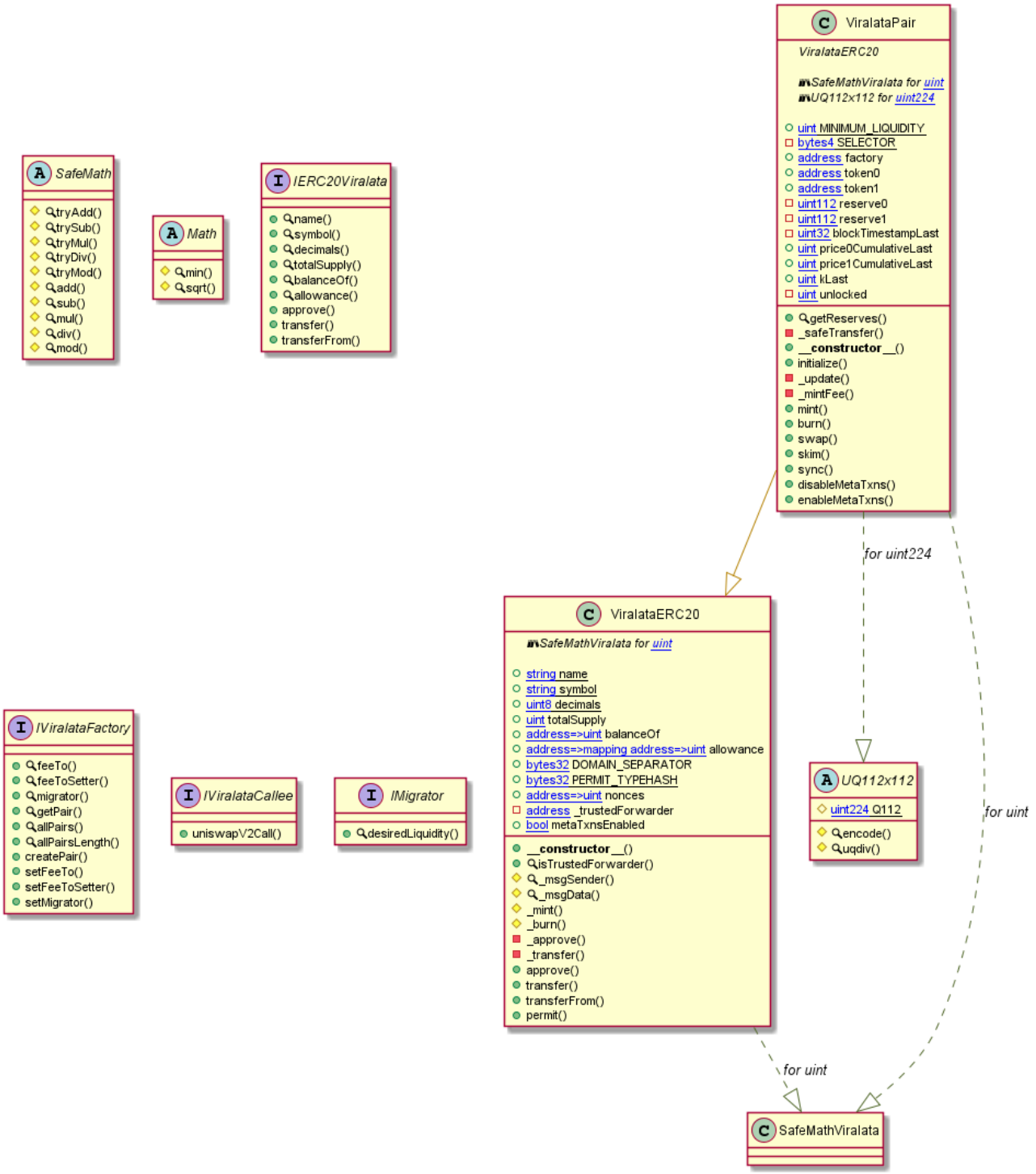
ViralataERC20 Diagram



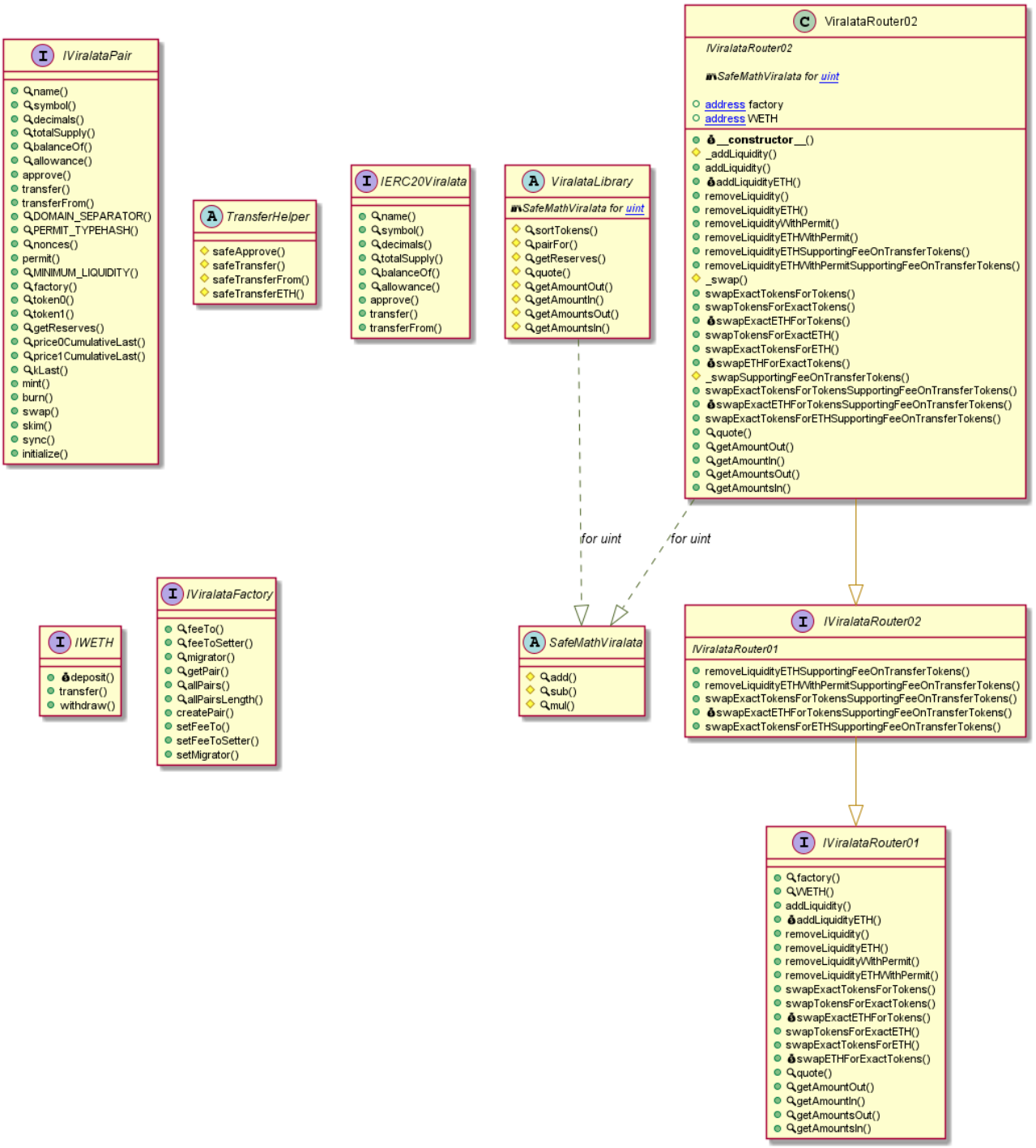
ViralataFactory Diagram



ViralataPair Diagram



ViralataRouter02 Diagram



Slither Results Log

Slither log >> ViralataSwapToken.sol

```
INFO:Detectors:
ViralataSwapToken.rescueTokens(IERC20,uint256) (ViralataSwapToken.sol#1444-1448) ignores return value by token.transfer(_msgSender(),valu
e) (ViralataSwapToken.sol#1445)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer
INFO:Detectors:
ERC20Permit.constructor(string).name (ViralataSwapToken.sol#840) shadows:
        - ERC20.name() (ViralataSwapToken.sol#320-322) (function)
        - IERC20Metadata.name() (ViralataSwapToken.sol#290) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
ViralataSwapToken.constructor(address).trustedForwarder (ViralataSwapToken.sol#1382) lacks a zero-check on :
                - _trustedForwarder = trustedForwarder (ViralataSwapToken.sol#1388)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Variable 'ECDSA.tryRecover(bytes32,bytes).r (ViralataSwapToken.sol#650)' in ECDSA.tryRecover(bytes32,bytes) (ViralataSwapToken.sol#645-67
4) potentially used before declaration: r = mload(uint256)(signature + 0x20) (ViralataSwapToken.sol#667)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables
INFO:Detectors:
Reentrancy in ViralataSwapToken.rescueTokens(IERC20,uint256) (ViralataSwapToken.sol#1444-1448):
        External calls:
        - token.transfer(_msgSender(),value) (ViralataSwapToken.sol#1445)
        Event emitted after the call(s):
        - TokensRescued(_msgSender(),address(token),value) (ViralataSwapToken.sol#1447)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
ERC20Permit.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (ViralataSwapToken.sol#845-864) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp <= deadline,ERC20Permit: expired deadline) (ViralataSwapToken.sol#854)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (ViralataSwapToken.sol#98-108) uses assembly
```

```
INFO:Detectors:
Address.isContract(address) (ViralataSwapToken.sol#98-108) uses assembly
        - INLINE ASM (ViralataSwapToken.sol#104-106)
Address.verifyCallResult(bool,bytes,string) (ViralataSwapToken.sol#191-211) uses assembly
        - INLINE ASM (ViralataSwapToken.sol#203-206)
ECDSA.tryRecover(bytes32,bytes) (ViralataSwapToken.sol#645-674) uses assembly
        - INLINE ASM (ViralataSwapToken.sol#655-659)
        - INLINE ASM (ViralataSwapToken.sol#666-669)
ECDSA.tryRecover(bytes32,bytes32,bytes32) (ViralataSwapToken.sol#696-708) uses assembly
        - INLINE ASM (ViralataSwapToken.sol#703-706)
ERC2771Context._msgSender() (ViralataSwapToken.sol#1141-1148) uses assembly
        - INLINE ASM (ViralataSwapToken.sol#1144)
ViralataSwapToken._msgSender() (ViralataSwapToken.sol#1397-1406) uses assembly
        - INLINE ASM (ViralataSwapToken.sol#1400-1402)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
AccessControl._setRoleAdmin(bytes32,bytes32) (ViralataSwapToken.sol#1339-1343) is never used and should be removed
Address.functionCall(address,bytes) (ViralataSwapToken.sol#118-120) is never used and should be removed
Address.functionCall(address,bytes,string) (ViralataSwapToken.sol#123-129) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (ViralataSwapToken.sol#131-137) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (ViralataSwapToken.sol#140-151) is never used and should be removed
Address.functionDelegateCall(address,bytes) (ViralataSwapToken.sol#169-171) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (ViralataSwapToken.sol#174-183) is never used and should be removed
Address.functionStaticCall(address,bytes) (ViralataSwapToken.sol#153-155) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (ViralataSwapToken.sol#157-166) is never used and should be removed
Address.isContract(address) (ViralataSwapToken.sol#98-108) is never used and should be removed
Address.sendValue(address,uint256) (ViralataSwapToken.sol#111-116) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (ViralataSwapToken.sol#191-211) is never used and should be removed
Context._msgData() (ViralataSwapToken.sol#300-302) is never used and should be removed
Counters.decrement(Counters.Counter) (ViralataSwapToken.sol#589-595) is never used and should be removed
Counters.reset(Counters.Counter) (ViralataSwapToken.sol#597-599) is never used and should be removed
ECDSA.recover(bytes32,bytes) (ViralataSwapToken.sol#690-694) is never used and should be removed
ECDSA.recover(bytes32,bytes32,bytes32) (ViralataSwapToken.sol#710-718) is never used and should be removed
ECDSA.toEthSignedMessageHash(bytes32) (ViralataSwapToken.sol#754-758) is never used and should be removed
ECDSA.tryRecover(bytes32,bytes) (ViralataSwapToken.sol#645-674) is never used and should be removed
ECDSA.tryRecover(bytes32,bytes32,bytes32) (ViralataSwapToken.sol#696-708) is never used and should be removed
ERC20._burn(address,uint256) (ViralataSwapToken.sol#439-454) is never used and should be removed
```

```
ECDSA.tryRecover(bytes32,bytes) (ViralataSwapToken.sol#645-674) is never used and should be removed
ECDSA.tryRecover(bytes32,bytes32,bytes32) (ViralataSwapToken.sol#696-708) is never used and should be removed
ERC20._burn(address,uint256) (ViralataSwapToken.sol#439-454) is never used and should be removed
ERC2771Context._msgData() (ViralataSwapToken.sol#1150-1156) is never used and should be removed
ERC2771Context._msgSender() (ViralataSwapToken.sol#1141-1148) is never used and should be removed
SafeERC20._callOptionalReturn(IERC20,bytes) (ViralataSwapToken.sol#278-285) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (ViralataSwapToken.sol#241-254) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (ViralataSwapToken.sol#265-276) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (ViralataSwapToken.sol#256-263) is never used and should be removed
SafeERC20.safeTransfer(IERC20,address,uint256) (ViralataSwapToken.sol#217-223) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (ViralataSwapToken.sol#225-232) is never used and should be removed
SafeMath.add(uint256,uint256) (ViralataSwapToken.sol#29-31) is never used and should be removed
SafeMath.div(uint256,uint256) (ViralataSwapToken.sol#38-40) is never used and should be removed
SafeMath.div(uint256,uint256,string) (ViralataSwapToken.sol#51-60) is never used and should be removed
SafeMath.mod(uint256,uint256) (ViralataSwapToken.sol#41-43) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (ViralataSwapToken.sol#61-70) is never used and should be removed
SafeMath.mul(uint256,uint256) (ViralataSwapToken.sol#35-37) is never used and should be removed
SafeMath.sub(uint256,uint256) (ViralataSwapToken.sol#32-34) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (ViralataSwapToken.sol#44-50) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (ViralataSwapToken.sol#4-8) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (ViralataSwapToken.sol#19-22) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (ViralataSwapToken.sol#23-28) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (ViralataSwapToken.sol#13-18) is never used and should be removed
SafeMath.trySub(uint256,uint256) (ViralataSwapToken.sol#9-12) is never used and should be removed
Strings.toHexString(uint256) (ViralataSwapToken.sol#1020-1031) is never used and should be removed
Strings.toString(uint256) (ViralataSwapToken.sol#995-1015) is never used and should be removed
ViralataSwapToken._beforeTokenTransfer(address,address,uint256) (ViralataSwapToken.sol#1436-1442) is never used and should be removed
ViralataSwapToken._msgData() (ViralataSwapToken.sol#1408-1414) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
ViralataSwapToken._maxSupply (ViralataSwapToken.sol#1369) is set pre-construction with a non-constant function or state variable:
        - 2000000000 * 10 ** decimals()
ViralataSwapToken._initialSupply (ViralataSwapToken.sol#1370) is set pre-construction with a non-constant function or state variable:
        - 200000 * 10 ** decimals()
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state-variables
INFO:Detectors:
Pragma version^0.8.0 (ViralataSwapToken.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.0 is not recommended for deployment
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state-variables
INFO:Detectors:
Pragma version^0.8.0 (ViralataSwapToken.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.0 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) (ViralataSwapToken.sol#111-116):
        - (success) = recipient.call{value: amount}() (ViralataSwapToken.sol#114)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (ViralataSwapToken.sol#140-151):
        - (success,returndata) = target.call{value: value}(data) (ViralataSwapToken.sol#149)
Low level call in Address.functionStaticCall(address,bytes,string) (ViralataSwapToken.sol#157-166):
        - (success,returndata) = target.staticcall(data) (ViralataSwapToken.sol#164)
Low level call in Address.functionDelegateCall(address,bytes,string) (ViralataSwapToken.sol#174-183):
        - (success,returndata) = target.delegatecall(data) (ViralataSwapToken.sol#181)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IERC20Permit.DOMAIN_SEPARATOR() (ViralataSwapToken.sol#568) is not in mixedCase
Variable EIP712._CACHED_DOMAIN_SEPARATOR (ViralataSwapToken.sol#768) is not in mixedCase
Variable EIP712._CACHED_CHAIN_ID (ViralataSwapToken.sol#769) is not in mixedCase
Variable EIP712._HASHED_NAME (ViralataSwapToken.sol#771) is not in mixedCase
Variable EIP712._HASHED_VERSION (ViralataSwapToken.sol#772) is not in mixedCase
Variable EIP712._TYPE_HASH (ViralataSwapToken.sol#773) is not in mixedCase
Function ERC20Permit.DOMAIN_SEPARATOR() (ViralataSwapToken.sol#877-879) is not in mixedCase
Variable ERC20Permit._PERMIT_TYPEHASH (ViralataSwapToken.sol#832-833) is not in mixedCase
Variable ERC2771Context._trustedForwarder (ViralataSwapToken.sol#1131) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
ViralataSwapToken.slitherConstructorVariables() (ViralataSwapToken.sol#1359-1462) uses literals with too many digits:
        - _maxSupply = 2000000000 * 10 ** decimals() (ViralataSwapToken.sol#1369)
ViralataSwapToken.slitherConstructorVariables() (ViralataSwapToken.sol#1359-1462) uses literals with too many digits:
        - _initialSupply = 200000 * 10 ** decimals() (ViralataSwapToken.sol#1370)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
ERC20Permit._PERMIT_TYPEHASH (ViralataSwapToken.sol#832-833) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (ViralataSwapToken.sol#320-322)
```

```
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (ViralataSwapToken.sol#320-322)
symbol() should be declared external:
        - ERC20.symbol() (ViralataSwapToken.sol#324-326)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (ViralataSwapToken.sol#337-339)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (ViralataSwapToken.sol#342-345)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (ViralataSwapToken.sol#347-349)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (ViralataSwapToken.sol#351-354)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (ViralataSwapToken.sol#356-370)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (ViralataSwapToken.sol#371-374)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (ViralataSwapToken.sol#376-384)
permit(address,address,uint256,uint256,uint8,bytes32,bytes32) should be declared external:
        - ERC20Permit.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (ViralataSwapToken.sol#845-864)
nonces(address) should be declared external:
        - ERC20Permit.nonces(address) (ViralataSwapToken.sol#869-871)
grantRole(bytes32,address) should be declared external:
        - AccessControl.grantRole(bytes32,address) (ViralataSwapToken.sol#1277-1279)
revokeRole(bytes32,address) should be declared external:
        - AccessControl.revokeRole(bytes32,address) (ViralataSwapToken.sol#1290-1292)
renounceRole(bytes32,address) should be declared external:
        - AccessControl.renounceRole(bytes32,address) (ViralataSwapToken.sol#1308-1312)
maxSupply() should be declared external:
        - ViralataSwapToken.maxSupply() (ViralataSwapToken.sol#1419-1421)
pause() should be declared external:
        - ViralataSwapToken.pause() (ViralataSwapToken.sol#1423-1425)
unpause() should be declared external:
        - ViralataSwapToken.unpause() (ViralataSwapToken.sol#1427-1429)
mint(address,uint256) should be declared external:
        - ViralataSwapToken.mint(address,uint256) (ViralataSwapToken.sol#1431-1434)
enableMetaTxns() should be declared external:
```

```
enableMetaTxns() should be declared external:
        - ViralataSwapToken.enableMetaTxns() (ViralataSwapToken.sol#1449-1454)
disableMetaTxns() should be declared external:
        - ViralataSwapToken.disableMetaTxns() (ViralataSwapToken.sol#1455-1460)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:ViralataSwapToken.sol analyzed (20 contracts with 75 detectors), 98 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> AuroDistributor.sol

```
INFO:Detectors:
AuroDistributor.safeAuroTransfer(address,uint256) (AuroDistributor.sol#1071-1081) ignores return value by auro.transfer(_to,auroBal) (Aur
oDistributor.sol#1076)
AuroDistributor.safeAuroTransfer(address,uint256) (AuroDistributor.sol#1071-1081) ignores return value by auro.transfer(_to,_amount) (Aur
oDistributor.sol#1078)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer
INFO:Detectors:
AuroDistributor.pendingAuro(uint256,address) (AuroDistributor.sol#895-909) performs a multiplication on the result of a division:
        -auroReward = multiplier.mul(auroPerBlock).mul(pool.allocPoint).div(totalAllocPoint) (AuroDistributor.sol#903)
        -accAuroPerShare = accAuroPerShare.add(auroReward.mul(1e12).div(lpSupply)) (AuroDistributor.sol#904)
AuroDistributor.updatePool(uint256) (AuroDistributor.sol#926-953) performs a multiplication on the result of a division:
        -auroReward = multiplier.mul(auroPerBlock).mul(pool.allocPoint).div(totalAllocPoint) (AuroDistributor.sol#939-942)
        -pool.accAuroPerShare = pool.accAuroPerShare.add(auroReward.mul(1e12).div(pool.totalLp)) (AuroDistributor.sol#947-949)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
INFO:Detectors:
AuroDistributor.deposit(uint256,uint256) (AuroDistributor.sol#956-989) uses a dangerous strict equality:
        - address(pool.lpToken) == address(auro) (AuroDistributor.sol#983)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
Reentrancy in AuroDistributor.add(uint256,IERC20,uint16,uint256,bool) (AuroDistributor.sol#858-873):
        External calls:
        - massUpdatePools() (AuroDistributor.sol#868)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        State variables written after the call(s):
        - poolInfo.push(PoolInfo(_lpToken,_allocPoint,lastRewardBlock,0,_depositFeeBP,_harvestInterval,0)) (AuroDistributor.sol#872)
        - totalAllocPoint = totalAllocPoint.add(_allocPoint) (AuroDistributor.sol#871)
Reentrancy in AuroDistributor.deposit(uint256,uint256) (AuroDistributor.sol#956-989):
        External calls:
        - updatePool(_pid) (AuroDistributor.sol#962)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#964)
                - auro.transfer(_to,auroBal) (AuroDistributor.sol#1076)
                - auro.transfer(_to,_amount) (AuroDistributor.sol#1078)
```

```
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#964)
                - auro.transfer(_to,auroBal) (AuroDistributor.sol#1076)
                - auro.transfer(_to,_amount) (AuroDistributor.sol#1078)
        State variables written after the call(s):
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#964)
                - user.nextHarvestUntil = block.timestamp.add(pool.harvestInterval) (AuroDistributor.sol#1047)
                - user.rewardLockedUp = 0 (AuroDistributor.sol#1057)
                - user.nextHarvestUntil = block.timestamp.add(pool.harvestInterval) (AuroDistributor.sol#1058)
                - user.rewardLockedUp = user.rewardLockedUp.add(pending) (AuroDistributor.sol#1064)
Reentrancy in AuroDistributor.deposit(uint256,uint256) (AuroDistributor.sol#956-989):
        External calls:
        - updatePool(_pid) (AuroDistributor.sol#962)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#964)
                - auro.transfer(_to,auroBal) (AuroDistributor.sol#1076)
                - auro.transfer(_to,_amount) (AuroDistributor.sol#1078)
        - pool.lpToken.safeTransferFrom(_msgSender(),address(this),_amount) (AuroDistributor.sol#968)
        - pool.lpToken.safeTransfer(feeAddress,depositFee) (AuroDistributor.sol#975)
        State variables written after the call(s):
        - pool.totalLp = pool.totalLp.add(_amount) (AuroDistributor.sol#981)
        - totalAuroInPools = totalAuroInPools.add(_amount) (AuroDistributor.sol#984)
        - user.amount = user.amount.add(_amount) (AuroDistributor.sol#980)
        - user.rewardDebt = user.amount.mul(pool.accAuroPerShare).div(1e12) (AuroDistributor.sol#987)
Reentrancy in AuroDistributor.set(uint256,uint256,uint16,uint256,bool) (AuroDistributor.sol#876-892):
        External calls:
        - massUpdatePools() (AuroDistributor.sol#886)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        State variables written after the call(s):
        - poolInfo[_pid].allocPoint = _allocPoint (AuroDistributor.sol#889)
        - poolInfo[_pid].depositFeeBP = _depositFeeBP (AuroDistributor.sol#890)
        - poolInfo[_pid].harvestInterval = _harvestInterval (AuroDistributor.sol#891)
        - totalAllocPoint = totalAllocPoint.sub(poolInfo[_pid].allocPoint).add(_allocPoint) (AuroDistributor.sol#888)
Reentrancy in AuroDistributor.updateAllocPoint(uint256,uint256,bool) (AuroDistributor.sol#1110-1123):
        External calls:
```

```
Reentrancy in AuroDistributor.updateAllocPoint(uint256,uint256,bool) (AuroDistributor.sol#1110-1123):
        External calls:
        - massUpdatePools() (AuroDistributor.sol#1116)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        State variables written after the call(s):
        - poolInfo[_pid].allocPoint = _allocPoint (AuroDistributor.sol#1122)
        - totalAllocPoint = totalAllocPoint.sub(poolInfo[_pid].allocPoint).add(_allocPoint) (AuroDistributor.sol#1121)
Reentrancy in AuroDistributor.updateEmissionRate(uint256) (AuroDistributor.sol#1103-1108):
        External calls:
        - massUpdatePools() (AuroDistributor.sol#1104)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        State variables written after the call(s):
        - auroPerBlock = _auroPerBlock (AuroDistributor.sol#1107)
Reentrancy in AuroDistributor.updatePool(uint256) (AuroDistributor.sol#926-953):
        External calls:
        - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
        - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        State variables written after the call(s):
        - pool.accAuroPerShare = pool.accAuroPerShare.add(auroReward.mul(1e12).div(pool.totalLp)) (AuroDistributor.sol#947-949)
        - pool.lastRewardBlock = block.number (AuroDistributor.sol#950)
Reentrancy in AuroDistributor.withdraw(uint256,uint256) (AuroDistributor.sol#992-1016):
        External calls:
        - updatePool(_pid) (AuroDistributor.sol#1002)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#1004)
                - auro.transfer(_to,auroBal) (AuroDistributor.sol#1076)
                - auro.transfer(_to,_amount) (AuroDistributor.sol#1078)
        State variables written after the call(s):
        - pool.totalLp = pool.totalLp.sub(_amount) (AuroDistributor.sol#1008)
        - totalAuroInPools = totalAuroInPools.sub(_amount) (AuroDistributor.sol#1010)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#1004)
                - user.nextHarvestUntil = block.timestamp.add(pool.harvestInterval) (AuroDistributor.sol#1047)
                - user.rewardLockedUp = 0 (AuroDistributor.sol#1057)
                - user.nextHarvestUntil = block.timestamp.add(pool.harvestInterval) (AuroDistributor.sol#1058)
                - user.rewardLockedUp = user.rewardLockedUp.add(pending) (AuroDistributor.sol#1064)
```

```
                - user.rewardLockedUp = user.rewardLockedUp.add(pending) (AuroDistributor.sol#1064)
        - user.amount = user.amount.sub(_amount) (AuroDistributor.sol#1007)
Reentrancy in AuroDistributor.withdraw(uint256,uint256) (AuroDistributor.sol#992-1016):
        External calls:
        - updatePool(_pid) (AuroDistributor.sol#1002)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#1004)
                - auro.transfer(_to,auroBal) (AuroDistributor.sol#1076)
                - auro.transfer(_to,_amount) (AuroDistributor.sol#1078)
        - pool.lpToken.safeTransfer(_msgSender(),_amount) (AuroDistributor.sol#1012)
        State variables written after the call(s):
        - user.rewardDebt = user.amount.mul(pool.accAuroPerShare).div(1e12) (AuroDistributor.sol#1014)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
INFO:Detectors:
AuroDistributor.add(uint256,IERC20,uint16,uint256,bool) (AuroDistributor.sol#858-873) should emit an event for:
        - totalAllocPoint = totalAllocPoint.add(_allocPoint) (AuroDistributor.sol#871)
AuroDistributor.set(uint256,uint256,uint16,uint256,bool) (AuroDistributor.sol#876-892) should emit an event for:
        - totalAllocPoint = totalAllocPoint.sub(poolInfo[_pid].allocPoint).add(_allocPoint) (AuroDistributor.sol#888)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Reentrancy in AuroDistributor.deposit(uint256,uint256) (AuroDistributor.sol#956-989):
        External calls:
        - updatePool(_pid) (AuroDistributor.sol#962)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#964)
                - auro.transfer(_to,auroBal) (AuroDistributor.sol#1076)
                - auro.transfer(_to,_amount) (AuroDistributor.sol#1078)
        Event emitted after the call(s):
        - RewardLockedUp(_msgSender(),_pid,pending) (AuroDistributor.sol#1066)
                - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#964)
Reentrancy in AuroDistributor.deposit(uint256,uint256) (AuroDistributor.sol#956-989):
        External calls:
        - updatePool(_pid) (AuroDistributor.sol#962)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#964)
```

```
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#964)
                - auro.transfer(_to,auroBal) (AuroDistributor.sol#1076)
                - auro.transfer(_to,_amount) (AuroDistributor.sol#1078)
        - pool.lpToken.safeTransferFrom(_msgSender(),address(this),_amount) (AuroDistributor.sol#968)
        - pool.lpToken.safeTransfer(feeAddress,depositFee) (AuroDistributor.sol#975)
        Event emitted after the call(s):
        - Deposit(_msgSender(),_pid,_amount) (AuroDistributor.sol#988)
Reentrancy in AuroDistributor.emergencyWithdraw(uint256) (AuroDistributor.sol#1019-1039):
        External calls:
        - pool.lpToken.safeTransfer(_msgSender(),amount) (AuroDistributor.sol#1036)
        Event emitted after the call(s):
        - EmergencyWithdraw(_msgSender(),_pid,amount) (AuroDistributor.sol#1038)
Reentrancy in AuroDistributor.updateAllocPoint(uint256,uint256,bool) (AuroDistributor.sol#1110-1123):
        External calls:
        - massUpdatePools() (AuroDistributor.sol#1116)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        Event emitted after the call(s):
        - AllocPointsUpdated(_msgSender(),poolInfo[_pid].allocPoint,_allocPoint) (AuroDistributor.sol#1119)
Reentrancy in AuroDistributor.updateEmissionRate(uint256) (AuroDistributor.sol#1103-1108):
        External calls:
        - massUpdatePools() (AuroDistributor.sol#1104)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        Event emitted after the call(s):
        - EmissionRateUpdated(msg.sender,auroPerBlock,_auroPerBlock) (AuroDistributor.sol#1106)
Reentrancy in AuroDistributor.withdraw(uint256,uint256) (AuroDistributor.sol#992-1016):
        External calls:
        - updatePool(_pid) (AuroDistributor.sol#1002)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#1004)
                - auro.transfer(_to,auroBal) (AuroDistributor.sol#1076)
                - auro.transfer(_to,_amount) (AuroDistributor.sol#1078)
        Event emitted after the call(s):
        - RewardLockedUp(_msgSender(),_pid,pending) (AuroDistributor.sol#1066)
                - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#1004)
```

```
        Event emitted after the call(s):
        - RewardLockedUp(_msgSender(),_pid,pending) (AuroDistributor.sol#1066)
                - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#1004)
Reentrancy in AuroDistributor.withdraw(uint256,uint256) (AuroDistributor.sol#992-1016):
        External calls:
        - updatePool(_pid) (AuroDistributor.sol#1002)
                - auro.mint(devAddress,auroReward.div(10)) (AuroDistributor.sol#944)
                - auro.mint(address(this),auroReward) (AuroDistributor.sol#945)
        - payOrLockupPendingAuro(_pid) (AuroDistributor.sol#1004)
                - auro.transfer(_to,auroBal) (AuroDistributor.sol#1076)
                - auro.transfer(_to,_amount) (AuroDistributor.sol#1078)
        - pool.lpToken.safeTransfer(_msgSender(),_amount) (AuroDistributor.sol#1012)
        Event emitted after the call(s):
        - Withdraw(_msgSender(),_pid,_amount) (AuroDistributor.sol#1015)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
AuroDistributor.canHarvest(uint256,address) (AuroDistributor.sol#912-915) uses timestamp for comparisons
        Dangerous comparisons:
        - block.number >= startBlock && block.timestamp >= user.nextHarvestUntil (AuroDistributor.sol#914)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (AuroDistributor.sol#316-326) uses assembly
        - INLINE ASM (AuroDistributor.sol#322-324)
Address.verifyCallResult(bool,bytes,string) (AuroDistributor.sol#485-505) uses assembly
        - INLINE ASM (AuroDistributor.sol#497-500)
AuroDistributor._msgSender() (AuroDistributor.sol#805-814) uses assembly
        - INLINE ASM (AuroDistributor.sol#808-810)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (AuroDistributor.sol#369-371) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (AuroDistributor.sol#398-404) is never used and should be removed
Address.functionDelegateCall(address,bytes) (AuroDistributor.sol#458-460) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (AuroDistributor.sol#468-477) is never used and should be removed
Address.functionStaticCall(address,bytes) (AuroDistributor.sol#431-433) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (AuroDistributor.sol#441-450) is never used and should be removed
Address.sendValue(address,uint256) (AuroDistributor.sol#344-349) is never used and should be removed
AuroDistributor._msgData() (AuroDistributor.sol#816-822) is never used and should be removed
Context._msgData() (AuroDistributor.sol#596-598) is never used and should be removed
```

```
AuroDistributor._msgData() (AuroDistributor.sol#816-822) is never used and should be removed
Context._msgData() (AuroDistributor.sol#596-598) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (AuroDistributor.sol#535-548) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (AuroDistributor.sol#559-570) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (AuroDistributor.sol#550-557) is never used and should be removed
SafeMath.div(uint256,uint256,string) (AuroDistributor.sol#179-188) is never used and should be removed
SafeMath.mod(uint256,uint256) (AuroDistributor.sol#139-141) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (AuroDistributor.sol#205-214) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (AuroDistributor.sol#156-165) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (AuroDistributor.sol#10-16) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (AuroDistributor.sol#52-57) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (AuroDistributor.sol#64-69) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (AuroDistributor.sol#35-45) is never used and should be removed
SafeMath.trySub(uint256,uint256) (AuroDistributor.sol#23-28) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.4 (AuroDistributor.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) (AuroDistributor.sol#344-349):
        - (success) = recipient.call{value: amount}() (AuroDistributor.sol#347)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (AuroDistributor.sol#412-423):
        - (success,returndata) = target.call{value: value}(data) (AuroDistributor.sol#421)
Low level call in Address.functionStaticCall(address,bytes,string) (AuroDistributor.sol#441-450):
        - (success,returndata) = target.staticcall(data) (AuroDistributor.sol#448)
Low level call in Address.functionDelegateCall(address,bytes,string) (AuroDistributor.sol#468-477):
        - (success,returndata) = target.delegatecall(data) (AuroDistributor.sol#475)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter AuroDistributor.getMultiplier(uint256,uint256)._from (AuroDistributor.sol#829) is not in mixedCase
Parameter AuroDistributor.getMultiplier(uint256,uint256)._to (AuroDistributor.sol#829) is not in mixedCase
Parameter AuroDistributor.add(uint256,IERC20,uint16,uint256,bool)._allocPoint (AuroDistributor.sol#859) is not in mixedCase
Parameter AuroDistributor.add(uint256,IERC20,uint16,uint256,bool)._lpToken (AuroDistributor.sol#860) is not in mixedCase
Parameter AuroDistributor.add(uint256,IERC20,uint16,uint256,bool)._depositFeeBP (AuroDistributor.sol#861) is not in mixedCase
Parameter AuroDistributor.add(uint256,IERC20,uint16,uint256,bool)._harvestInterval (AuroDistributor.sol#862) is not in mixedCase
Parameter AuroDistributor.add(uint256,IERC20,uint16,uint256,bool)._withUpdate (AuroDistributor.sol#863) is not in mixedCase
Parameter AuroDistributor.set(uint256,uint256,uint16,uint256,bool)._pid (AuroDistributor.sol#877) is not in mixedCase
```

```
Parameter AuroDistributor.add(uint256,IERC20,uint16,uint256,bool)._withUpdate (AuroDistributor.sol#863) is not in mixedCase
Parameter AuroDistributor.set(uint256,uint256,uint16,uint256,bool)._pid (AuroDistributor.sol#877) is not in mixedCase
Parameter AuroDistributor.set(uint256,uint256,uint16,uint256,bool)._allocPoint (AuroDistributor.sol#878) is not in mixedCase
Parameter AuroDistributor.set(uint256,uint256,uint16,uint256,bool)._depositFeeBP (AuroDistributor.sol#879) is not in mixedCase
Parameter AuroDistributor.set(uint256,uint256,uint16,uint256,bool)._harvestInterval (AuroDistributor.sol#880) is not in mixedCase
Parameter AuroDistributor.set(uint256,uint256,uint16,uint256,bool)._withUpdate (AuroDistributor.sol#881) is not in mixedCase
Parameter AuroDistributor.pendingAuro(uint256,address)._pid (AuroDistributor.sol#895) is not in mixedCase
Parameter AuroDistributor.pendingAuro(uint256,address)._user (AuroDistributor.sol#895) is not in mixedCase
Parameter AuroDistributor.canHarvest(uint256,address)._pid (AuroDistributor.sol#912) is not in mixedCase
Parameter AuroDistributor.canHarvest(uint256,address)._user (AuroDistributor.sol#912) is not in mixedCase
Parameter AuroDistributor.updatePool(uint256)._pid (AuroDistributor.sol#926) is not in mixedCase
Parameter AuroDistributor.deposit(uint256,uint256)._pid (AuroDistributor.sol#956) is not in mixedCase
Parameter AuroDistributor.deposit(uint256,uint256)._amount (AuroDistributor.sol#956) is not in mixedCase
Parameter AuroDistributor.withdraw(uint256,uint256)._pid (AuroDistributor.sol#992) is not in mixedCase
Parameter AuroDistributor.withdraw(uint256,uint256)._amount (AuroDistributor.sol#992) is not in mixedCase
Parameter AuroDistributor.emergencyWithdraw(uint256)._pid (AuroDistributor.sol#1019) is not in mixedCase
Parameter AuroDistributor.payOrLockupPendingAuro(uint256)._pid (AuroDistributor.sol#1042) is not in mixedCase
Parameter AuroDistributor.safeAuroTransfer(address,uint256)._to (AuroDistributor.sol#1071) is not in mixedCase
Parameter AuroDistributor.safeAuroTransfer(address,uint256)._amount (AuroDistributor.sol#1071) is not in mixedCase
Parameter AuroDistributor.setDevAddress(address)._devAddress (AuroDistributor.sol#1084) is not in mixedCase
Parameter AuroDistributor.setFeeAddress(address)._feeAddress (AuroDistributor.sol#1093) is not in mixedCase
Parameter AuroDistributor.updateEmissionRate(uint256)._auroPerBlock (AuroDistributor.sol#1103) is not in mixedCase
Parameter AuroDistributor.updateAllocPoint(uint256,uint256,bool)._pid (AuroDistributor.sol#1111) is not in mixedCase
Parameter AuroDistributor.updateAllocPoint(uint256,uint256,bool)._allocPoint (AuroDistributor.sol#1112) is not in mixedCase
Parameter AuroDistributor.updateAllocPoint(uint256,uint256,bool)._withUpdate (AuroDistributor.sol#1113) is not in mixedCase
Constant AuroDistributor._trustedForwarder (AuroDistributor.sol#701) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (AuroDistributor.sol#635-637)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (AuroDistributor.sol#643-646)
operator() should be declared external:
        - AuroDistributor.operator() (AuroDistributor.sol#824-826)
transferOperator(address) should be declared external:
        - AuroDistributor.transferOperator(address) (AuroDistributor.sol#833-837)
startFarming() should be declared external:
        - AuroDistributor.startFarming() (AuroDistributor.sol#840-850)
```

```
        - AuroDistributor.add(uint256,IERC20,uint16,uint256,bool) (AuroDistributor.sol#858-873)
set(uint256,uint256,uint16,uint256,bool) should be declared external:
        - AuroDistributor.set(uint256,uint256,uint16,uint256,bool) (AuroDistributor.sol#876-892)
withdraw(uint256,uint256) should be declared external:
        - AuroDistributor.withdraw(uint256,uint256) (AuroDistributor.sol#992-1016)
emergencyWithdraw(uint256) should be declared external:
        - AuroDistributor.emergencyWithdraw(uint256) (AuroDistributor.sol#1019-1039)
setDevAddress(address) should be declared external:
        - AuroDistributor.setDevAddress(address) (AuroDistributor.sol#1084-1091)
setFeeAddress(address) should be declared external:
        - AuroDistributor.setFeeAddress(address) (AuroDistributor.sol#1093-1100)
updateEmissionRate(uint256) should be declared external:
        - AuroDistributor.updateEmissionRate(uint256) (AuroDistributor.sol#1103-1108)
updateAllocPoint(uint256,uint256,bool) should be declared external:
        - AuroDistributor.updateAllocPoint(uint256,uint256,bool) (AuroDistributor.sol#1110-1123)
enableMetaTxns() should be declared external:
        - AuroDistributor.enableMetaTxns() (AuroDistributor.sol#1126-1131)
disableMetaTxns() should be declared external:
        - AuroDistributor.disableMetaTxns() (AuroDistributor.sol#1134-1139)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:AuroDistributor.sol analyzed (9 contracts with 75 detectors), 101 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
root@server:/chetan/gaza/mycontracts#
```

Slither log >> ViralataERC20.sol

```
INFO:Detectors:
ViralataERC20._trustedForwarder (ViralataERC20.sol#250) is never initialized. It is used in:
        - ViralataERC20.isTrustedForwarder(address) (ViralataERC20.sol#276-278)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-state-variables
INFO:Detectors:
ViralataERC20.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (ViralataERC20.sol#337-349) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(deadline >= block.timestamp,ViralataSwap: EXPIRED) (ViralataERC20.sol#338)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
ViralataERC20.constructor() (ViralataERC20.sol#260-274) uses assembly
        - INLINE ASM (ViralataERC20.sol#262-264)
ViralataERC20._msgSender() (ViralataERC20.sol#280-289) uses assembly
        - INLINE ASM (ViralataERC20.sol#283-285)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
SafeMath.add(uint256,uint256) (ViralataERC20.sol#98-100) is never used and should be removed
SafeMath.div(uint256,uint256) (ViralataERC20.sol#140-142) is never used and should be removed
SafeMath.div(uint256,uint256,string) (ViralataERC20.sol#196-205) is never used and should be removed
SafeMath.mod(uint256,uint256) (ViralataERC20.sol#156-158) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (ViralataERC20.sol#222-231) is never used and should be removed
SafeMath.mul(uint256,uint256) (ViralataERC20.sol#126-128) is never used and should be removed
SafeMath.sub(uint256,uint256) (ViralataERC20.sol#112-114) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (ViralataERC20.sol#173-182) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (ViralataERC20.sol#27-33) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (ViralataERC20.sol#69-74) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (ViralataERC20.sol#81-86) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (ViralataERC20.sol#52-62) is never used and should be removed
SafeMath.trySub(uint256,uint256) (ViralataERC20.sol#40-45) is never used and should be removed
SafeMathViralata.mul(uint256,uint256) (ViralataERC20.sol#15-17) is never used and should be removed
```

```
SafeMath.trySub(uint256,uint256) (ViralataERC20.sol#40-45) is never used and should be removed
SafeMathViralata.mul(uint256,uint256) (ViralataERC20.sol#15-17) is never used and should be removed
ViralataERC20._burn(address,uint256) (ViralataERC20.sol#305-309) is never used and should be removed
ViralataERC20._mint(address,uint256) (ViralataERC20.sol#299-303) is never used and should be removed
ViralataERC20._msgData() (ViralataERC20.sol#291-297) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version0.8.0 (ViralataERC20.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.0 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Variable ViralataERC20.DOMAIN_SEPARATOR (ViralataERC20.sol#245) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
ViralataERC20._trustedForwarder (ViralataERC20.sol#250) should be constant
ViralataERC20.metaTxnsEnabled (ViralataERC20.sol#253) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Slither:ViralataERC20.sol analyzed (3 contracts with 75 detectors), 26 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> ViralataFactory.sol

```
INFO:Detectors:
ViralataPair._update(uint256,uint256,uint112,uint112) (ViralataFactory.sol#496-509) uses a weak PRNG: "blockTimestamp = uint32(block.time
stamp % 2 ** 32) (ViralataFactory.sol#498)"
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#weak-PRNG
INFO:Detectors:
ViralataERC20._trustedForwarder (ViralataFactory.sol#250) is never initialized. It is used in:
        - ViralataERC20.isTrustedForwarder(address) (ViralataFactory.sol#276-278)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-state-variables
INFO:Detectors:
ViralataPair._safeTransfer(address,address,uint256) (ViralataFactory.sol#467-470) uses a dangerous strict equality:
        - require(bool,string)(success && (data.length == 0 || abi.decode(data,(bool))),ViralataSwap: TRANSFER_FAILED) (ViralataFactory.s
ol#469)
ViralataPair.mint(address) (ViralataFactory.sol#535-563) uses a dangerous strict equality:
        - _totalSupply == 0 (ViralataFactory.sol#544)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
Reentrancy in ViralataPair.burn(address) (ViralataFactory.sol#566-588):
        External calls:
        - _safeTransfer(_token0,to,amount0) (ViralataFactory.sol#580)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        - _safeTransfer(_token1,to,amount1) (ViralataFactory.sol#581)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        State variables written after the call(s):
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#585)
                - blockTimestampLast = blockTimestamp (ViralataFactory.sol#507)
        - kLast = uint256(reserve0).mul(reserve1) (ViralataFactory.sol#586)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#585)
                - reserve0 = uint112(balance0) (ViralataFactory.sol#505)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#585)
                - reserve1 = uint112(balance1) (ViralataFactory.sol#506)
Reentrancy in ViralataFactory.createPair(address,address) (ViralataFactory.sol#673-688):
        External calls:
        - ViralataPair(pair).initialize(token0,token1) (ViralataFactory.sol#683)
        State variables written after the call(s):
        - getPair[token0][token1] = pair (ViralataFactory.sol#684)
        - getPair[token1][token0] = pair (ViralataFactory.sol#685)
Reentrancy in ViralataPair.swap(uint256,uint256,address,bytes) (ViralataFactory.sol#591-619):
```

```
        - getPair[token1][token0] = pair (ViralataFactory.sol#685)
Reentrancy in ViralataPair.swap(uint256,uint256,address,bytes) (ViralataFactory.sol#591-619):
        External calls:
        - _safeTransfer(_token0,to,amount0Out) (ViralataFactory.sol#602)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        - _safeTransfer(_token1,to,amount1Out) (ViralataFactory.sol#603)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        - IViralataCallee(to).uniswapV2Call(_msgSender(),amount0Out,amount1Out,data) (ViralataFactory.sol#604)
        State variables written after the call(s):
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#617)
                - blockTimestampLast = blockTimestamp (ViralataFactory.sol#507)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#617)
                - reserve0 = uint112(balance0) (ViralataFactory.sol#505)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#617)
                - reserve1 = uint112(balance1) (ViralataFactory.sol#506)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
INFO:Detectors:
ViralataPair.initialize(address,address)._token0 (ViralataFactory.sol#489) lacks a zero-check on :
                - token0 = _token0 (ViralataFactory.sol#491)
ViralataPair.initialize(address,address)._token1 (ViralataFactory.sol#489) lacks a zero-check on :
                - token1 = _token1 (ViralataFactory.sol#492)
ViralataFactory.constructor(address)._feeToSetter (ViralataFactory.sol#665) lacks a zero-check on :
                - feeToSetter = _feeToSetter (ViralataFactory.sol#666)
ViralataFactory.setFeeTo(address)._feeTo (ViralataFactory.sol#690) lacks a zero-check on :
                - feeTo = _feeTo (ViralataFactory.sol#692)
ViralataFactory.setMigrator(address)._migrator (ViralataFactory.sol#695) lacks a zero-check on :
                - migrator = _migrator (ViralataFactory.sol#697)
ViralataFactory.setFeeToSetter(address)._feeToSetter (ViralataFactory.sol#700) lacks a zero-check on :
                - feeToSetter = _feeToSetter (ViralataFactory.sol#702)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in ViralataPair.burn(address) (ViralataFactory.sol#566-588):
        External calls:
        - _safeTransfer(_token0,to,amount0) (ViralataFactory.sol#580)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        - _safeTransfer(_token1,to,amount1) (ViralataFactory.sol#581)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        State variables written after the call(s):
```

```
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        State variables written after the call(s):
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#585)
                - price0CumulativeLast += uint256(UQ112x112.encode(_reserve1).uqdiv(_reserve0)) * timeElapsed (ViralataFactory.sol#502)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#585)
                - price1CumulativeLast += uint256(UQ112x112.encode(_reserve0).uqdiv(_reserve1)) * timeElapsed (ViralataFactory.sol#503)
Reentrancy in ViralataFactory.createPair(address,address) (ViralataFactory.sol#673-688):
        External calls:
        - ViralataPair(pair).initialize(token0,token1) (ViralataFactory.sol#683)
        State variables written after the call(s):
        - allPairs.push(pair) (ViralataFactory.sol#686)
Reentrancy in ViralataPair.swap(uint256,uint256,address,bytes) (ViralataFactory.sol#591-619):
        External calls:
        - _safeTransfer(_token0,to,amount0Out) (ViralataFactory.sol#602)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        - _safeTransfer(_token1,to,amount1Out) (ViralataFactory.sol#603)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        - IViralataCallee(to).uniswapV2Call(_msgSender(),amount0Out,amount1Out,data) (ViralataFactory.sol#604)
        State variables written after the call(s):
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#617)
                - price0CumulativeLast += uint256(UQ112x112.encode(_reserve1).uqdiv(_reserve0)) * timeElapsed (ViralataFactory.sol#502)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#617)
                - price1CumulativeLast += uint256(UQ112x112.encode(_reserve0).uqdiv(_reserve1)) * timeElapsed (ViralataFactory.sol#503)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in ViralataPair.burn(address) (ViralataFactory.sol#566-588):
        External calls:
        - _safeTransfer(_token0,to,amount0) (ViralataFactory.sol#580)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        - _safeTransfer(_token1,to,amount1) (ViralataFactory.sol#581)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        Event emitted after the call(s):
        - Burn(_msgSender(),amount0,amount1,to) (ViralataFactory.sol#587)
        - Sync(reserve0,reserve1) (ViralataFactory.sol#508)
                - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#585)
Reentrancy in ViralataFactory.createPair(address,address) (ViralataFactory.sol#673-688):
        External calls:
        - ViralataPair(pair).initialize(token0,token1) (ViralataFactory.sol#683)
```

```
        - ViralataPair(pair).initialize(token0,token1) (ViralataFactory.sol#683)
        Event emitted after the call(s):
        - PairCreated(token0,token1,pair,allPairs.length) (ViralataFactory.sol#687)
Reentrancy in ViralataPair.swap(uint256,uint256,address,bytes) (ViralataFactory.sol#591-619):
        External calls:
        - _safeTransfer(_token0,to,amount0Out) (ViralataFactory.sol#602)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        - _safeTransfer(_token1,to,amount1Out) (ViralataFactory.sol#603)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
        - IViralataCallee(to).uniswapV2Call(_msgSender(),amount0Out,amount1Out,data) (ViralataFactory.sol#604)
        Event emitted after the call(s):
        - Swap(_msgSender(),amount0In,amount1In,amount0Out,amount1Out,to) (ViralataFactory.sol#618)
        - Sync(reserve0,reserve1) (ViralataFactory.sol#508)
                - _update(balance0,balance1,_reserve0,_reserve1) (ViralataFactory.sol#617)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
ViralataERC20.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (ViralataFactory.sol#340-352) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(deadline >= block.timestamp,ViralataSwap: EXPIRED) (ViralataFactory.sol#341)
ViralataPair._update(uint256,uint256,uint112,uint112) (ViralataFactory.sol#496-509) uses timestamp for comparisons
        Dangerous comparisons:
        - timeElapsed > 0 && _reserve0 != 0 && _reserve1 != 0 (ViralataFactory.sol#500)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
ViralataERC20.constructor() (ViralataFactory.sol#260-274) uses assembly
        - INLINE ASM (ViralataFactory.sol#262-264)
ViralataERC20._msgSender() (ViralataFactory.sol#280-289) uses assembly
        - INLINE ASM (ViralataFactory.sol#283-285)
ViralataFactory.createPair(address,address) (ViralataFactory.sol#673-688) uses assembly
        - INLINE ASM (ViralataFactory.sol#680-682)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
SafeMath.add(uint256,uint256) (ViralataFactory.sol#98-100) is never used and should be removed
SafeMath.div(uint256,uint256) (ViralataFactory.sol#140-142) is never used and should be removed
SafeMath.div(uint256,uint256,string) (ViralataFactory.sol#196-205) is never used and should be removed
SafeMath.mod(uint256,uint256) (ViralataFactory.sol#156-158) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (ViralataFactory.sol#222-231) is never used and should be removed
SafeMath.mul(uint256,uint256) (ViralataFactory.sol#126-128) is never used and should be removed
```

```
SafeMath.sub(uint256,uint256,string) (ViralataFactory.sol#173-182) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (ViralataFactory.sol#27-33) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (ViralataFactory.sol#69-74) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (ViralataFactory.sol#81-86) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (ViralataFactory.sol#52-62) is never used and should be removed
SafeMath.trySub(uint256,uint256) (ViralataFactory.sol#40-45) is never used and should be removed
ViralataERC20._msgData() (ViralataFactory.sol#291-297) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Low level call in ViralataPair._safeTransfer(address,address,uint256) (ViralataFactory.sol#467-470):
        - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataFactory.sol#468)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Variable ViralataERC20.DOMAIN_SEPARATOR (ViralataFactory.sol#245) is not in mixedCase
Parameter ViralataPair.initialize(address,address)._token0 (ViralataFactory.sol#489) is not in mixedCase
Parameter ViralataPair.initialize(address,address)._token1 (ViralataFactory.sol#489) is not in mixedCase
Parameter ViralataFactory.setFeeTo(address)._feeTo (ViralataFactory.sol#690) is not in mixedCase
Parameter ViralataFactory.setMigrator(address)._migrator (ViralataFactory.sol#695) is not in mixedCase
Parameter ViralataFactory.setFeeToSetter(address)._feeToSetter (ViralataFactory.sol#700) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable ViralataPair.swap(uint256,uint256,address,bytes).balance0Adjusted (ViralataFactory.sol#612) is too similar to ViralataPair.swap(
uint256,uint256,address,bytes).balance1Adjusted (ViralataFactory.sol#613)
Variable ViralataPair.price0CumulativeLast (ViralataFactory.sol#449) is too similar to ViralataPair.price1CumulativeLast (ViralataFactory
.sol#450)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
ViralataFactory.createPair(address,address) (ViralataFactory.sol#673-688) uses literals with too many digits:
        - bytecode = type()(ViralataPair).creationCode (ViralataFactory.sol#678)
ViralataFactory.slitherConstructorConstantVariables() (ViralataFactory.sol#653-727) uses literals with too many digits:
        - INIT_CODE_PAIR_HASH = keccak256(bytes)(abi.encodePacked(type()(ViralataPair).creationCode)) (ViralataFactory.sol#654)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
ViralataERC20._trustedForwarder (ViralataFactory.sol#250) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Slither:ViralataFactory.sol analyzed (11 contracts with 75 detectors), 50 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
root@server:/chetan/gaza/mycontracts#
```

Slither log >> ViralataPair.sol

```
INFO:Detectors:
ViralataPair._update(uint256,uint256,uint112,uint112) (ViralataPair.sol#487-500) uses a weak PRNG: "blockTimestamp = uint32(block.timesta
mp % 2 ** 32) (ViralataPair.sol#489)"
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#weak-PRNG
INFO:Detectors:
ViralataERC20._trustedForwarder (ViralataPair.sol#248) is never initialized. It is used in:
        - ViralataERC20.isTrustedForwarder(address) (ViralataPair.sol#274-276)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-state-variables
INFO:Detectors:
ViralataPair._safeTransfer(address,address,uint256) (ViralataPair.sol#465-468) uses a dangerous strict equality:
        - require(bool,string)(success && (data.length == 0 || abi.decode(data,(bool))),ViralataSwap: TRANSFER_FAILED) (ViralataPair.sol#
467)
ViralataPair.mint(address) (ViralataPair.sol#526-554) uses a dangerous strict equality:
        - _totalSupply == 0 (ViralataPair.sol#535)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
Reentrancy in ViralataPair.burn(address) (ViralataPair.sol#557-579):
        External calls:
        - _safeTransfer(_token0,to,amount0) (ViralataPair.sol#571)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        - _safeTransfer(_token1,to,amount1) (ViralataPair.sol#572)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        State variables written after the call(s):
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#576)
                - blockTimestampLast = blockTimestamp (ViralataPair.sol#498)
        - kLast = uint256(reserve0).mul(reserve1) (ViralataPair.sol#577)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#576)
                - reserve0 = uint112(balance0) (ViralataPair.sol#496)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#576)
                - reserve1 = uint112(balance1) (ViralataPair.sol#497)
Reentrancy in ViralataPair.swap(uint256,uint256,address,bytes) (ViralataPair.sol#582-610):
        External calls:
        - _safeTransfer(_token0,to,amount0Out) (ViralataPair.sol#593)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        - _safeTransfer(_token1,to,amount1Out) (ViralataPair.sol#594)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        - IViralataCallee(to).uniswapV2Call(_msgSender(),amount0Out,amount1Out,data) (ViralataPair.sol#595)
```

```
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        - IViralataCallee(to).uniswapV2Call(_msgSender(),amount0Out,amount1Out,data) (ViralataPair.sol#595)
        State variables written after the call(s):
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#608)
                - blockTimestampLast = blockTimestamp (ViralataPair.sol#498)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#608)
                - reserve0 = uint112(balance0) (ViralataPair.sol#496)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#608)
                - reserve1 = uint112(balance1) (ViralataPair.sol#497)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
INFO:Detectors:
ViralataPair.initialize(address,address)._token0 (ViralataPair.sol#480) lacks a zero-check on :
                - token0 = _token0 (ViralataPair.sol#482)
ViralataPair.initialize(address,address)._token1 (ViralataPair.sol#480) lacks a zero-check on :
                - token1 = _token1 (ViralataPair.sol#483)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in ViralataPair.burn(address) (ViralataPair.sol#557-579):
        External calls:
        - _safeTransfer(_token0,to,amount0) (ViralataPair.sol#571)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        - _safeTransfer(_token1,to,amount1) (ViralataPair.sol#572)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        State variables written after the call(s):
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#576)
                - price0CumulativeLast += uint256(UQ112x112.encode(_reserve1).uqdiv(_reserve0)) * timeElapsed (ViralataPair.sol#493)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#576)
                - price1CumulativeLast += uint256(UQ112x112.encode(_reserve0).uqdiv(_reserve1)) * timeElapsed (ViralataPair.sol#494)
Reentrancy in ViralataPair.swap(uint256,uint256,address,bytes) (ViralataPair.sol#582-610):
        External calls:
        - _safeTransfer(_token0,to,amount0Out) (ViralataPair.sol#593)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        - _safeTransfer(_token1,to,amount1Out) (ViralataPair.sol#594)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        - IViralataCallee(to).uniswapV2Call(_msgSender(),amount0Out,amount1Out,data) (ViralataPair.sol#595)
        State variables written after the call(s):
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#608)
                - price0CumulativeLast += uint256(UQ112x112.encode(_reserve1).uqdiv(_reserve0)) * timeElapsed (ViralataPair.sol#493)
```

```
                - price0CumulativeLast += uint256(UQ112x112.encode(_reserve1).uqdiv(_reserve0)) * timeElapsed (ViralataPair.sol#493)
        - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#608)
                - price1CumulativeLast += uint256(UQ112x112.encode(_reserve0).uqdiv(_reserve1)) * timeElapsed (ViralataPair.sol#494)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in ViralataPair.burn(address) (ViralataPair.sol#557-579):
        External calls:
        - _safeTransfer(_token0,to,amount0) (ViralataPair.sol#571)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        - _safeTransfer(_token1,to,amount1) (ViralataPair.sol#572)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        Event emitted after the call(s):
        - Burn(_msgSender(),amount0,amount1,to) (ViralataPair.sol#578)
        - Sync(reserve0,reserve1) (ViralataPair.sol#499)
                - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#576)
Reentrancy in ViralataPair.swap(uint256,uint256,address,bytes) (ViralataPair.sol#582-610):
        External calls:
        - _safeTransfer(_token0,to,amount0Out) (ViralataPair.sol#593)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        - _safeTransfer(_token1,to,amount1Out) (ViralataPair.sol#594)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
        - IViralataCallee(to).uniswapV2Call(_msgSender(),amount0Out,amount1Out,data) (ViralataPair.sol#595)
        Event emitted after the call(s):
        - Swap(_msgSender(),amount0In,amount1In,amount0Out,amount1Out,to) (ViralataPair.sol#609)
        - Sync(reserve0,reserve1) (ViralataPair.sol#499)
                - _update(balance0,balance1,_reserve0,_reserve1) (ViralataPair.sol#608)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
ViralataERC20.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (ViralataPair.sol#338-350) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(deadline >= block.timestamp,ViralataSwap: EXPIRED) (ViralataPair.sol#339)
ViralataPair._update(uint256,uint256,uint112,uint112) (ViralataPair.sol#487-500) uses timestamp for comparisons
        Dangerous comparisons:
        - timeElapsed > 0 && _reserve0 != 0 && _reserve1 != 0 (ViralataPair.sol#491)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
ViralataERC20.constructor() (ViralataPair.sol#258-272) uses assembly
        - INLINE ASM (ViralataPair.sol#260-262)
```

```
        - INLINE ASM (ViralataPair.sol#281-283)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
SafeMath.add(uint256,uint256) (ViralataPair.sol#96-98) is never used and should be removed
SafeMath.div(uint256,uint256) (ViralataPair.sol#138-140) is never used and should be removed
SafeMath.div(uint256,uint256,string) (ViralataPair.sol#194-203) is never used and should be removed
SafeMath.mod(uint256,uint256) (ViralataPair.sol#154-156) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (ViralataPair.sol#220-229) is never used and should be removed
SafeMath.mul(uint256,uint256) (ViralataPair.sol#124-126) is never used and should be removed
SafeMath.sub(uint256,uint256) (ViralataPair.sol#110-112) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (ViralataPair.sol#171-180) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (ViralataPair.sol#25-31) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (ViralataPair.sol#67-72) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (ViralataPair.sol#79-84) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (ViralataPair.sol#50-60) is never used and should be removed
SafeMath.trySub(uint256,uint256) (ViralataPair.sol#38-43) is never used and should be removed
ViralataERC20._msgData() (ViralataPair.sol#289-295) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Low level call in ViralataPair._safeTransfer(address,address,uint256) (ViralataPair.sol#465-468):
        - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (ViralataPair.sol#466)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Variable ViralataERC20.DOMAIN_SEPARATOR (ViralataPair.sol#243) is not in mixedCase
Parameter ViralataPair.initialize(address,address)._token0 (ViralataPair.sol#480) is not in mixedCase
Parameter ViralataPair.initialize(address,address)._token1 (ViralataPair.sol#480) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable ViralataPair.swap(uint256,uint256,address,bytes).balance0Adjusted (ViralataPair.sol#603) is too similar to ViralataPair.swap(uin
t256,uint256,address,bytes).balance1Adjusted (ViralataPair.sol#604)
Variable ViralataPair.price0CumulativeLast (ViralataPair.sol#447) is too similar to ViralataPair.price1CumulativeLast (ViralataPair.sol#4
48)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
ViralataERC20._trustedForwarder (ViralataPair.sol#248) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Slither:ViralataPair.sol analyzed (10 contracts with 75 detectors), 37 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> ViralataRouter02.sol

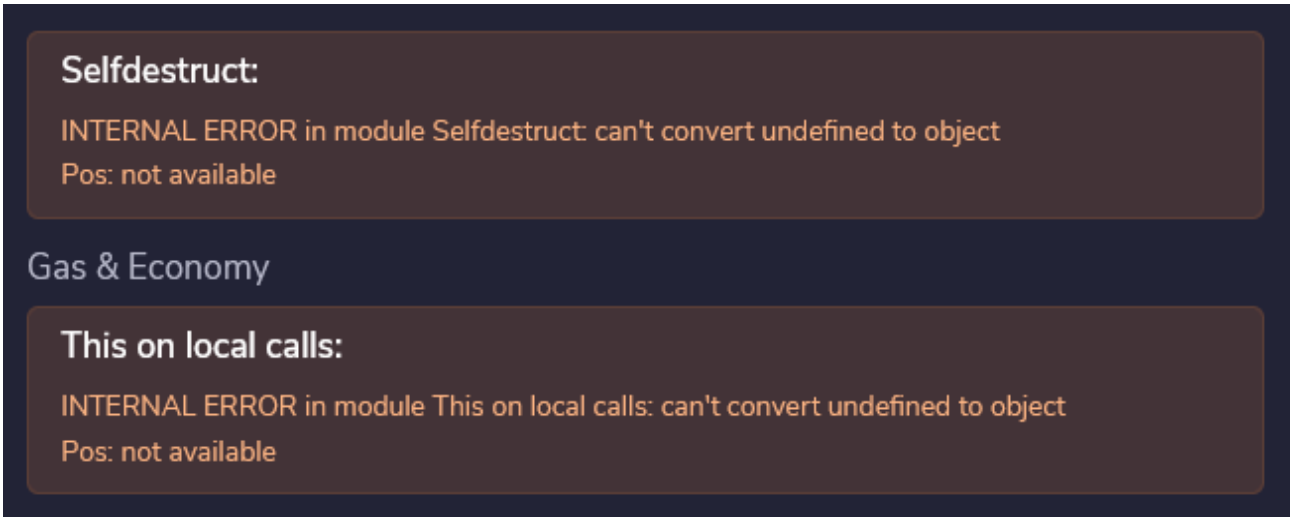
```
INFO:Detectors:
ViralataRouter02.removeLiquidity(address,address,uint256,uint256,uint256,address,uint256) (ViralataRouter02.sol#436-452) ignores return v
alue by IViralataPair(pair).transferFrom(msg.sender,pair,liquidity) (ViralataRouter02.sol#446)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer
INFO:Detectors:
ViralataLibrary.getAmountsOut(address,uint256,address[]).i (ViralataRouter02.sol#113) is a local variable never initialized
ViralataRouter02._swapSupportingFeeOnTransferTokens(address[],address).i (ViralataRouter02.sol#655) is a local variable never initialized
ViralataRouter02._swap(uint256[],address[],address).i (ViralataRouter02.sol#546) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
ViralataRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256) (ViralataRouter02.sol#366-393) ignores return value by IV
iralataFactory(factory).createPair(tokenA,tokenB) (ViralataRouter02.sol#376)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
ViralataRouter02.constructor(address,address)._factory (ViralataRouter02.sol#356) lacks a zero-check on :
                - factory = _factory (ViralataRouter02.sol#357)
ViralataRouter02.constructor(address,address)._WETH (ViralataRouter02.sol#356) lacks a zero-check on :
                - WETH = _WETH (ViralataRouter02.sol#358)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
ViralataRouter02._swap(uint256[],address[],address) (ViralataRouter02.sol#545-556) has external calls inside a loop: IViralataPair(Virala
taLibrary.pairFor(factory,input,output)).swap(amount0Out,amount1Out,to,new bytes(0)) (ViralataRouter02.sol#552-554)
ViralataRouter02._swapSupportingFeeOnTransferTokens(address[],address) (ViralataRouter02.sol#654-671) has external calls inside a loop: (
reserve0,reserve1) = pair.getReserves() (ViralataRouter02.sol#662)
ViralataRouter02._swapSupportingFeeOnTransferTokens(address[],address) (ViralataRouter02.sol#654-671) has external calls inside a loop: a
mountInput = IERC20Viralata(input).balanceOf(address(pair)).sub(reserveInput) (ViralataRouter02.sol#664)
ViralataRouter02._swapSupportingFeeOnTransferTokens(address[],address) (ViralataRouter02.sol#654-671) has external calls inside a loop: p
air.swap(amount0Out,amount1Out,to,new bytes(0)) (ViralataRouter02.sol#669)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
INFO:Detectors:
TransferHelper.safeApprove(address,address,uint256) (ViralataRouter02.sol#146-150) is never used and should be removed
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
INFO:Detectors:
TransferHelper.safeApprove(address,address,uint256) (ViralataRouter02.sol#146-150) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Low level call in TransferHelper.safeApprove(address,address,uint256) (ViralataRouter02.sol#146-150):
        - (success,data) = token.call(abi.encodeWithSelector(0x095ea7b3,to,value)) (ViralataRouter02.sol#148)
Low level call in TransferHelper.safeTransfer(address,address,uint256) (ViralataRouter02.sol#152-156):
        - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value)) (ViralataRouter02.sol#154)
Low level call in TransferHelper.safeTransferFrom(address,address,address,uint256) (ViralataRouter02.sol#158-162):
        - (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,to,value)) (ViralataRouter02.sol#160)
Low level call in TransferHelper.safeTransferETH(address,uint256) (ViralataRouter02.sol#164-167):
        - (success) = to.call{value: value}(new bytes(0)) (ViralataRouter02.sol#165)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IViralataPair.DOMAIN_SEPARATOR() (ViralataRouter02.sol#19) is not in mixedCase
Function IViralataPair.PERMIT_TYPEHASH() (ViralataRouter02.sol#20) is not in mixedCase
Function IViralataPair.MINIMUM_LIQUIDITY() (ViralataRouter02.sol#37) is not in mixedCase
Function IViralataRouter01.WETH() (ViralataRouter02.sol#212) is not in mixedCase
Variable ViralataRouter02.WETH (ViralataRouter02.sol#349) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable IViralataRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (ViralataRouter02
.sol#217) is too similar to IViralataRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesire
d (ViralataRouter02.sol#218)
Variable ViralataRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (ViralataRouter02.
sol#397) is too similar to IViralataRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired
 (ViralataRouter02.sol#218)
Variable ViralataRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountADesired (ViralataRouter02.sol#369) is too
 similar to ViralataRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBDesired (ViralataRouter02.sol#370)
Variable ViralataRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (ViralataRouter02.
sol#397) is too similar to ViralataRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired
(ViralataRouter02.sol#398)
Variable ViralataRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountADesired (ViralataRouter02.sol#369) is too
 similar to IViralataRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (ViralataRoute
r02.sol#218)
Variable ViralataRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountADesired (ViralataRouter02.sol#369) is too
 similar to ViralataRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (ViralataRouter
```

```
Variable ViralataRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountADesired (ViralataRouter02.sol#369) is too
 similar to ViralataRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (ViralataRouter
02.sol#398)
Variable ViralataRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (ViralataRouter02.
sol#397) is too similar to ViralataRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBDesired (ViralataRouter
02.sol#370)
Variable IViralataRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (ViralataRouter02
.sol#217) is too similar to ViralataRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBDesired (ViralataRoute
r02.sol#370)
Variable IViralataRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (ViralataRouter02
.sol#217) is too similar to ViralataRouter02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired
 (ViralataRouter02.sol#398)
Variable ViralataRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountAOptimal (ViralataRouter02.sol#387) is too
 similar to ViralataRouter02._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBOptimal (ViralataRouter02.sol#382)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
quote(uint256,uint256,uint256) should be declared external:
        - ViralataRouter02.quote(uint256,uint256,uint256) (ViralataRouter02.sol#736-738)
getAmountOut(uint256,uint256,uint256) should be declared external:
        - ViralataRouter02.getAmountOut(uint256,uint256,uint256) (ViralataRouter02.sol#740-748)
getAmountIn(uint256,uint256,uint256) should be declared external:
        - ViralataRouter02.getAmountIn(uint256,uint256,uint256) (ViralataRouter02.sol#750-758)
getAmountsOut(uint256,address[]) should be declared external:
        - ViralataRouter02.getAmountsOut(uint256,address[]) (ViralataRouter02.sol#760-768)
getAmountsIn(uint256,address[]) should be declared external:
        - ViralataRouter02.getAmountsIn(uint256,address[]) (ViralataRouter02.sol#770-778)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:ViralataRouter02.sol analyzed (10 contracts with 75 detectors), 36 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Solidity static analysis

ViralataSwapToken.sol



Email: audit@EtherAuthority.io

Delete dynamic array:

INTERNAL ERROR in module Delete dynamic array: can't convert undefined to object

Pos: not available

For loop over dynamic array:

INTERNAL ERROR in module For loop over dynamic array: can't convert undefined to object

Pos: not available

Ether transfer in loop:

INTERNAL ERROR in module Ether transfer in loop: can't convert undefined to object

Pos: not available

ERC

ERC20:

INTERNAL ERROR in module ERC20: can't convert undefined to object

Pos: not available

Miscellaneous

Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: can't convert undefined to object

Pos: not available

Similar variable names:

INTERNAL ERROR in module Similar variable names: can't convert undefined to object

Pos: not available

No return:

INTERNAL ERROR in module No return: can't convert undefined to object

Pos: not available

Guard conditions:

INTERNAL ERROR in module Guard conditions: can't convert undefined to object

Pos: not available

String length:

INTERNAL ERROR in module String length: can't convert undefined to object

Pos: not available

AuroDistributor.sol

SOLIDITY STATIC ANALYSIS

AuroDistributor.sol

Security

Transaction origin:

INTERNAL ERROR in module Transaction origin: can't convert undefined to object

Pos: not available

Check-effects-interaction:

INTERNAL ERROR in module Check-effects-interaction: can't convert undefined to object

Pos: not available

Inline assembly:

INTERNAL ERROR in module Inline assembly: can't convert undefined to object

Pos: not available

Block timestamp:

INTERNAL ERROR in module Block timestamp: can't convert undefined to object

Pos: not available

Low level calls:

INTERNAL ERROR in module Low level calls: can't convert undefined to object

Pos: not available

Selfdestruct:

INTERNAL ERROR in module Selfdestruct: can't convert undefined to object

Pos: not available

Gas & Economy

This on local calls:

INTERNAL ERROR in module This on local calls: can't convert undefined to object
Pos: not available

Delete dynamic array:

INTERNAL ERROR in module Delete dynamic array: can't convert undefined to object
Pos: not available

For loop over dynamic array:

INTERNAL ERROR in module For loop over dynamic array: can't convert undefined to object
Pos: not available

Ether transfer in loop:

INTERNAL ERROR in module Ether transfer in loop: can't convert undefined to object
Pos: not available

ERC

ERC20:

INTERNAL ERROR in module ERC20: can't convert undefined to object
Pos: not available

Miscellaneous

Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: can't convert undefined to object
Pos: not available

Similar variable names:

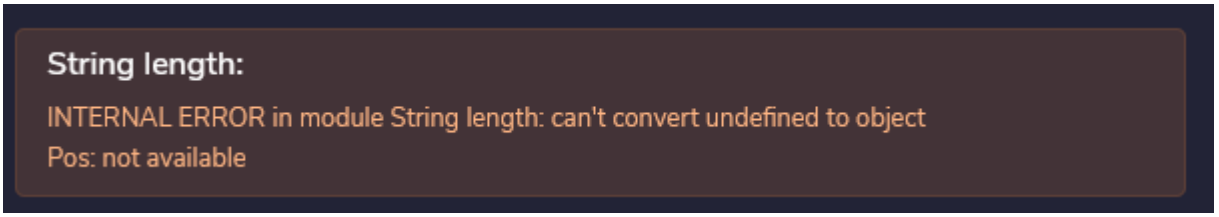
INTERNAL ERROR in module Similar variable names: can't convert undefined to object
Pos: not available

No return:

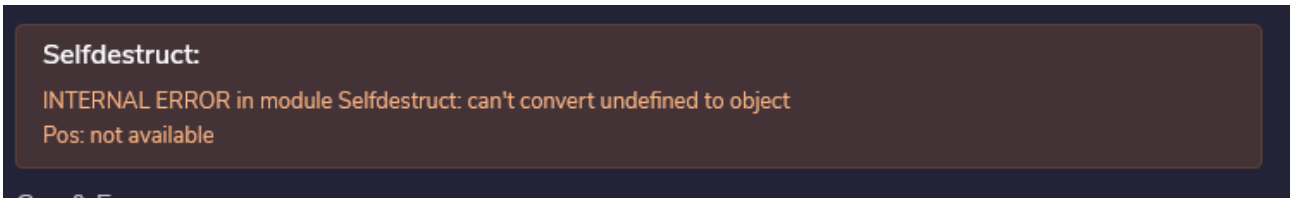
INTERNAL ERROR in module No return: can't convert undefined to object
Pos: not available

Guard conditions:

INTERNAL ERROR in module Guard conditions: can't convert undefined to object
Pos: not available



ViralataERC20.sol



Gas & Economy

This on local calls:

INTERNAL ERROR in module This on local calls: can't convert undefined to object
Pos: not available

Delete dynamic array:

INTERNAL ERROR in module Delete dynamic array: can't convert undefined to object
Pos: not available

For loop over dynamic array:

INTERNAL ERROR in module For loop over dynamic array: can't convert undefined to object
Pos: not available

Ether transfer in loop:

INTERNAL ERROR in module Ether transfer in loop: can't convert undefined to object
Pos: not available

ERC

ERC20:

INTERNAL ERROR in module ERC20: can't convert undefined to object
Pos: not available

Miscellaneous

Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: can't convert undefined to object
Pos: not available

Similar variable names:

INTERNAL ERROR in module Similar variable names: can't convert undefined to object
Pos: not available

No return:

INTERNAL ERROR in module No return: can't convert undefined to object
Pos: not available

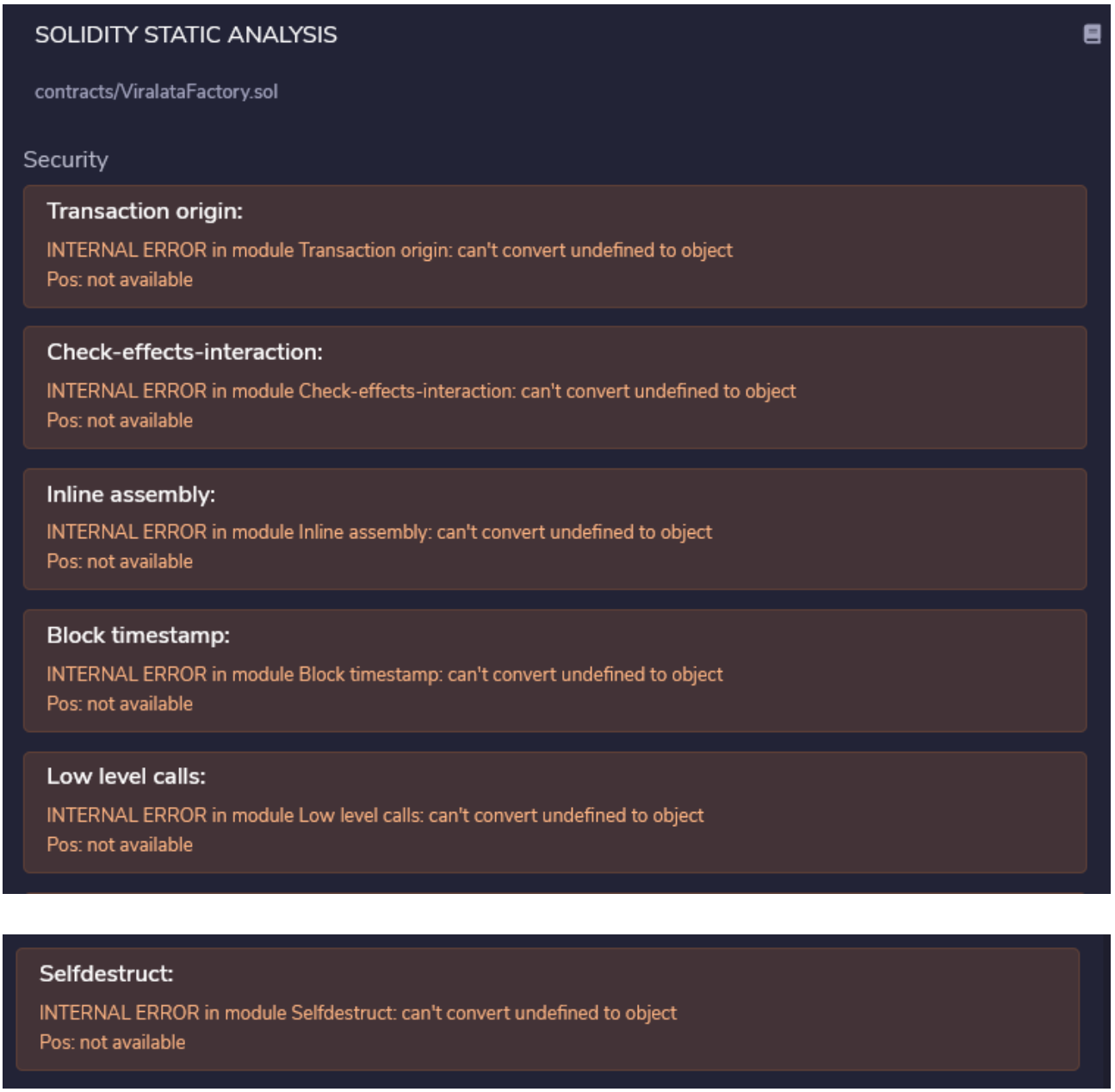
Guard conditions:

INTERNAL ERROR in module Guard conditions: can't convert undefined to object
Pos: not available

String length:

INTERNAL ERROR in module String length: can't convert undefined to object
Pos: not available

ViralataFactory.sol



Gas & Economy

This on local calls:

INTERNAL ERROR in module This on local calls: can't convert undefined to object
Pos: not available

Delete dynamic array:

INTERNAL ERROR in module Delete dynamic array: can't convert undefined to object
Pos: not available

For loop over dynamic array:

INTERNAL ERROR in module For loop over dynamic array: can't convert undefined to object
Pos: not available

Ether transfer in loop:

INTERNAL ERROR in module Ether transfer in loop: can't convert undefined to object
Pos: not available

ERC

ERC20:

INTERNAL ERROR in module ERC20: can't convert undefined to object
Pos: not available

Miscellaneous

Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: can't convert undefined to object
Pos: not available

Similar variable names:

INTERNAL ERROR in module Similar variable names: can't convert undefined to object
Pos: not available

No return:

INTERNAL ERROR in module No return: can't convert undefined to object
Pos: not available

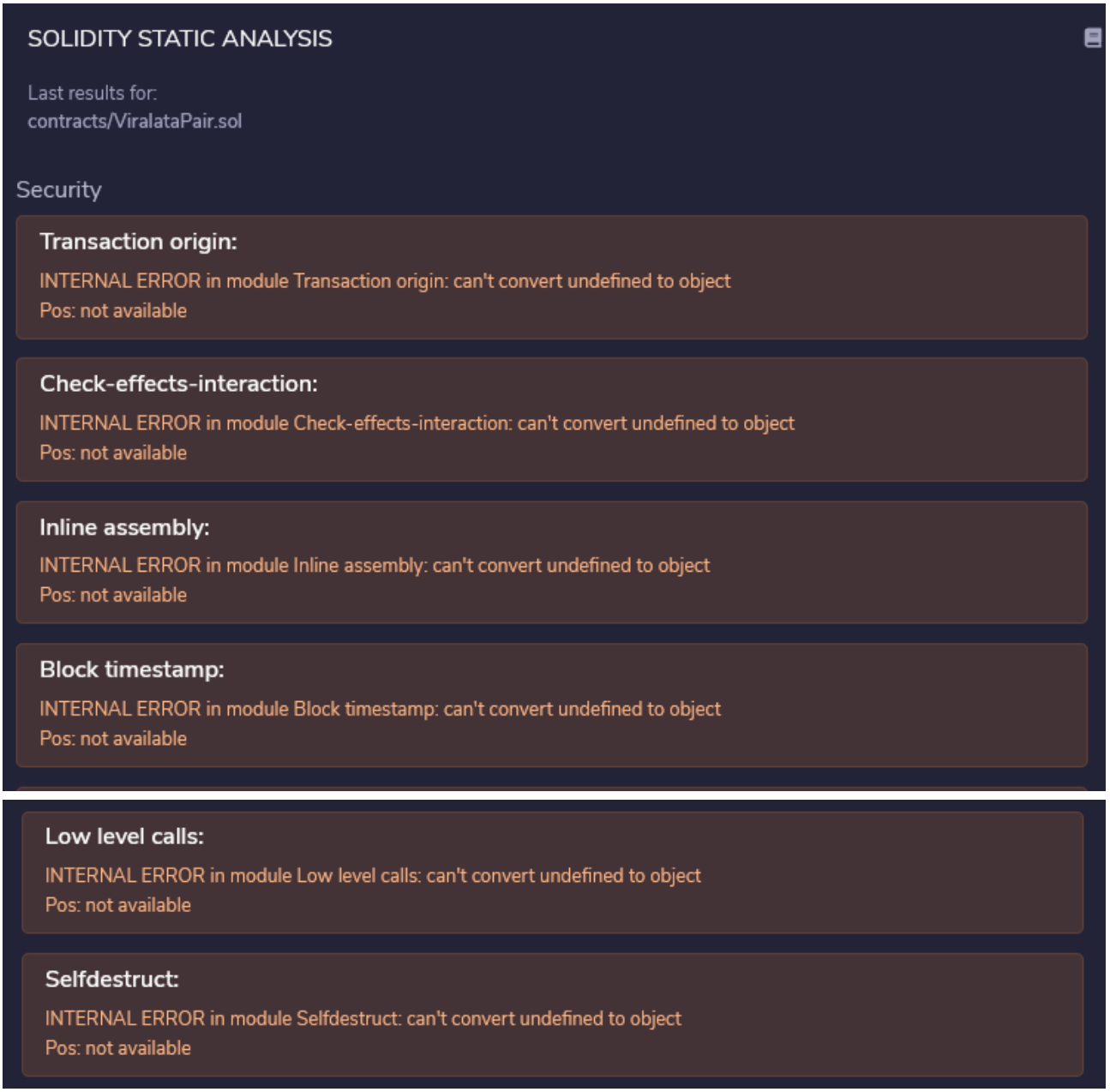
Guard conditions:

INTERNAL ERROR in module Guard conditions: can't convert undefined to object
Pos: not available

String length:

INTERNAL ERROR in module String length: can't convert undefined to object
Pos: not available

ViralataPair.sol



Gas & Economy

This on local calls:

INTERNAL ERROR in module This on local calls: can't convert undefined to object
Pos: not available

Delete dynamic array:

INTERNAL ERROR in module Delete dynamic array: can't convert undefined to object
Pos: not available

For loop over dynamic array:

INTERNAL ERROR in module For loop over dynamic array: can't convert undefined to object
Pos: not available

Ether transfer in loop:

INTERNAL ERROR in module Ether transfer in loop: can't convert undefined to object
Pos: not available

ERC

ERC20:

INTERNAL ERROR in module ERC20: can't convert undefined to object
Pos: not available

Miscellaneous

Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: can't convert undefined to object
Pos: not available

Similar variable names:

INTERNAL ERROR in module Similar variable names: can't convert undefined to object
Pos: not available

No return:

INTERNAL ERROR in module No return: can't convert undefined to object
Pos: not available

Guard conditions:

INTERNAL ERROR in module Guard conditions: can't convert undefined to object
Pos: not available

String length:

INTERNAL ERROR in module String length: can't convert undefined to object
Pos: not available

ViralataRouter02.sol

Gas & Economy

This on local calls:

INTERNAL ERROR in module This on local calls: can't convert undefined to object
Pos: not available

Delete dynamic array:

INTERNAL ERROR in module Delete dynamic array: can't convert undefined to object
Pos: not available

For loop over dynamic array:

INTERNAL ERROR in module For loop over dynamic array: can't convert undefined to object
Pos: not available

Ether transfer in loop:

INTERNAL ERROR in module Ether transfer in loop: can't convert undefined to object
Pos: not available

ERC

ERC20:

INTERNAL ERROR in module ERC20: can't convert undefined to object
Pos: not available

Miscellaneous

Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: can't convert undefined to object
Pos: not available

Similar variable names:

INTERNAL ERROR in module Similar variable names: can't convert undefined to object
Pos: not available

No return:

INTERNAL ERROR in module No return: can't convert undefined to object
Pos: not available

Guard conditions:

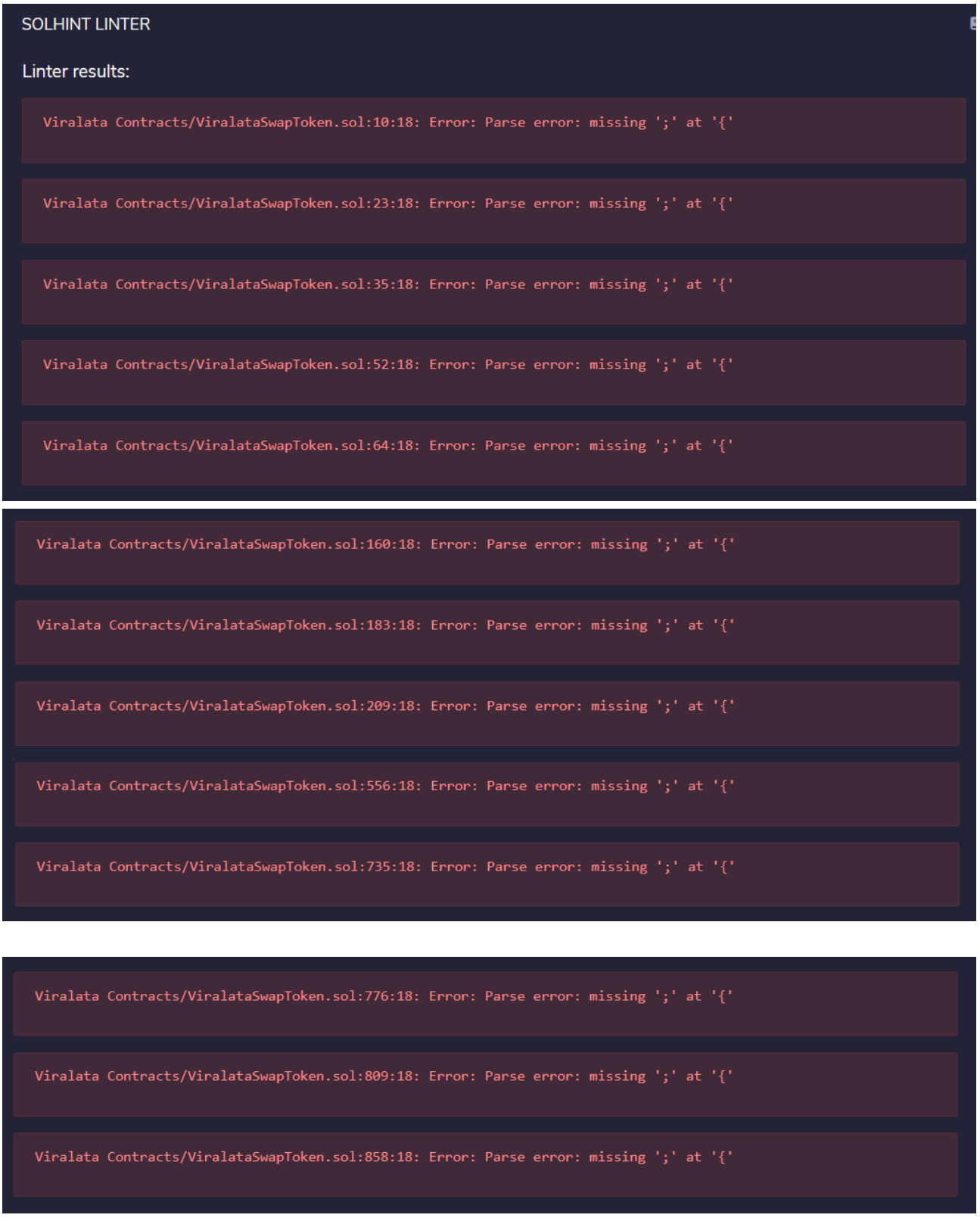
INTERNAL ERROR in module Guard conditions: can't convert undefined to object
Pos: not available

String length:

INTERNAL ERROR in module String length: can't convert undefined to object
Pos: not available

Solhint Linter

ViralataSwapToken.sol



```
Viralata Contracts/ViralataSwapToken.sol:996:18: Error: Parse error: missing ';' at '{'
```

```
Viralata Contracts/ViralataSwapToken.sol:1004:18: Error: Parse error: missing ';' at '{'
```

AuroDistributor.sol

SOLHINT LINTER

Linter results:

```
AuroDistributor.sol:11:18: Error: Parse error: missing ';' at '{'
```

```
AuroDistributor.sol:24:18: Error: Parse error: missing ';' at '{'
```

```
AuroDistributor.sol:36:18: Error: Parse error: missing ';' at '{'
```

```
AuroDistributor.sol:53:18: Error: Parse error: missing ';' at '{'
```

```
AuroDistributor.sol:65:18: Error: Parse error: missing ';' at '{'
```

```
AuroDistributor.sol:161:18: Error: Parse error: missing ';' at '{'
```

```
AuroDistributor.sol:184:18: Error: Parse error: missing ';' at '{'
```

```
AuroDistributor.sol:210:18: Error: Parse error: missing ';' at '{'
```

```
AuroDistributor.sol:564:18: Error: Parse error: missing ';' at '{'
```

ViralataERC20.sol

SOLHINT LINTER

```
contracts/ViralataERC20.sol:12:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataERC20.sol:25:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataERC20.sol:37:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataERC20.sol:54:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataERC20.sol:66:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataERC20.sol:162:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataERC20.sol:185:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataERC20.sol:211:18: Error: Parse error: missing ';' at '{'
```

ViralataFactory.sol

SOLHINT LINTER

Linter results:

```
contracts/ViralataFactory.sol:31:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataFactory.sol:44:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataFactory.sol:56:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataFactory.sol:73:18: Error: Parse error: missing ';' at '{'
```

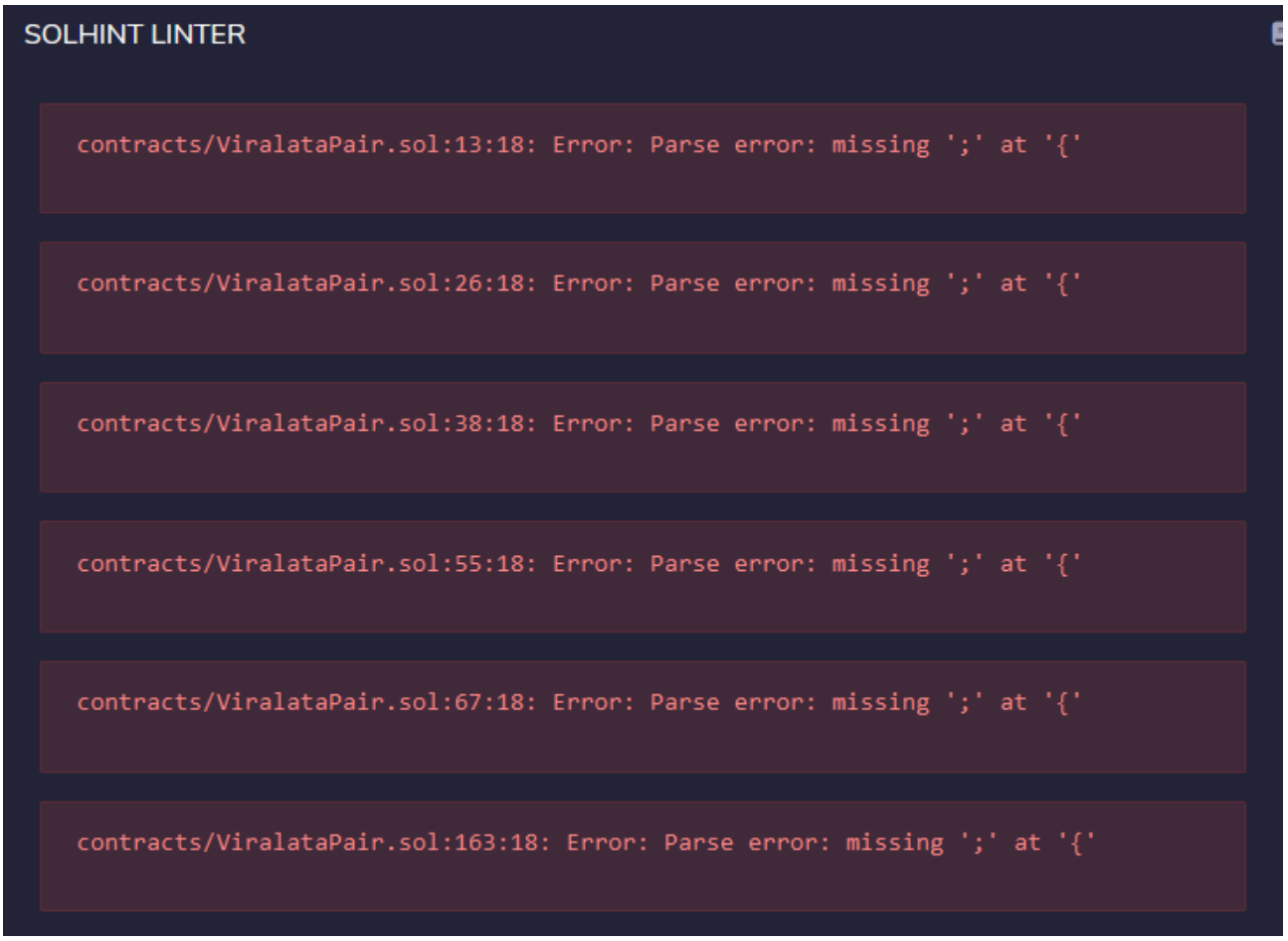
```
contracts/ViralataFactory.sol:85:18: Error: Parse error: missing ';' at '{'
```

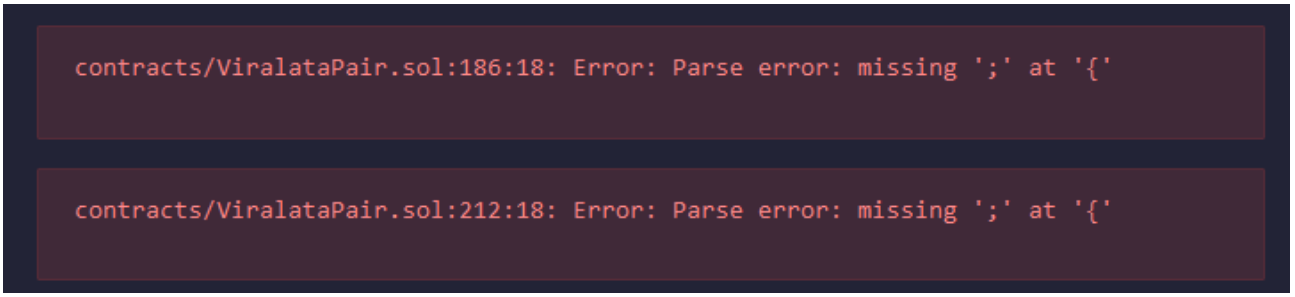
```
contracts/ViralataFactory.sol:181:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataFactory.sol:204:18: Error: Parse error: missing ';' at '{'
```

```
contracts/ViralataFactory.sol:230:18: Error: Parse error: missing ';' at '{'
```

ViralataPair.sol





ViralataRouter02.sol

SOLHINT LINTER

Linter results:

```
ViralataRouter02.sol:3:1: Error: Compiler version =0.6.12 does not satisfy the r
semver requirement
```

```
ViralataRouter02.sol:19:5: Error: Function name must be in mixedCase
```

```
ViralataRouter02.sol:20:5: Error: Function name must be in mixedCase
```

```
ViralataRouter02.sol:37:5: Error: Function name must be in mixedCase
```

```
ViralataRouter02.sol:59:35: Error: Use double quotes for string literals
```

```
ViralataRouter02.sol:61:39: Error: Use double quotes for string literals
```

```
ViralataRouter02.sol:84:30: Error: Use double quotes for string literals

ViralataRouter02.sol:85:47: Error: Use double quotes for string literals

ViralataRouter02.sol:91:31: Error: Use double quotes for string literals

ViralataRouter02.sol:92:50: Error: Use double quotes for string literals

ViralataRouter02.sol:101:32: Error: Use double quotes for string literals

ViralataRouter02.sol:102:50: Error: Use double quotes for string literals
```

```
ViralataRouter02.sol:110:35: Error: Use double quotes for string literals

ViralataRouter02.sol:121:35: Error: Use double quotes for string literals

ViralataRouter02.sol:134:35: Error: Use double quotes for string literals

ViralataRouter02.sol:138:35: Error: Use double quotes for string literals

ViralataRouter02.sol:142:49: Error: Use double quotes for string literals

ViralataRouter02.sol:148:45: Error: Avoid to use low level calls.
```

```
ViralataRouter02.sol:149:76: Error: Use double quotes for string literals

ViralataRouter02.sol:154:45: Error: Avoid to use low level calls.

ViralataRouter02.sol:155:76: Error: Use double quotes for string literals

ViralataRouter02.sol:160:45: Error: Avoid to use low level calls.

ViralataRouter02.sol:161:76: Error: Use double quotes for string literals

ViralataRouter02.sol:165:27: Error: Avoid to use low level calls.
```

```
ViralataRouter02.sol:166:26: Error: Use double quotes for string literals

ViralataRouter02.sol:212:5: Error: Function name must be in mixedCase

ViralataRouter02.sol:349:39: Error: Variable name must be in mixedCase

ViralataRouter02.sol:352:29: Error: Avoid to make time-based decisions in your
business logic

ViralataRouter02.sol:352:46: Error: Use double quotes for string literals

ViralataRouter02.sol:356:35: Error: Variable name must be in mixedCase
```

```
ViralataRouter02.sol:384:55: Error: Use double quotes for string literals

ViralataRouter02.sol:389:55: Error: Use double quotes for string literals

ViralataRouter02.sol:450:40: Error: Use double quotes for string literals

ViralataRouter02.sol:451:40: Error: Use double quotes for string literals

ViralataRouter02.sol:565:62: Error: Use double quotes for string literals

ViralataRouter02.sol:579:44: Error: Use double quotes for string literals
```

```
ViralataRouter02.sol:593:34: Error: Use double quotes for string literals

ViralataRouter02.sol:595:62: Error: Use double quotes for string literals

ViralataRouter02.sol:607:48: Error: Use double quotes for string literals

ViralataRouter02.sol:609:44: Error: Use double quotes for string literals

ViralataRouter02.sol:624:48: Error: Use double quotes for string literals

ViralataRouter02.sol:626:62: Error: Use double quotes for string literals
```

```
ViralataRouter02.sol:642:34: Error: Use double quotes for string literals

ViralataRouter02.sol:644:42: Error: Use double quotes for string literals

ViralataRouter02.sol:686:13: Error: Use double quotes for string literals

ViralataRouter02.sol:701:34: Error: Use double quotes for string literals

ViralataRouter02.sol:709:13: Error: Use double quotes for string literals

ViralataRouter02.sol:724:48: Error: Use double quotes for string literals

ViralataRouter02.sol:730:44: Error: Use double quotes for string literals
```

Email: audit@EtherAuthority.io